PERFORMANCE PREDICTION AND TUNING FOR SYMMETRIC COEXISTENCE OF WIFI AND ZIGBEE NETWORKS

A Dissertation

by

WEI ZHANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2017

Major Subject: Computer Engineering

ABSTRACT

Due to the explosive deployment of WiFi and ZigBee wireless networks, 2.4GHz ISM bands (2.4GHz-2.5GHz) are becoming increasingly crowded, and the co-channel coexistence of these two networks is inevitable. For coexistence networks, people always want to predict their performance (e.g. throughput, energy consumption, etc.) before deployment, or even want to tune parameters to compensate unnecessary performance degradation (owing to the huge differences between these two MAC protocols) or to satisfy some performance requirements (e.g., priority, delay constraint, etc.) of them. However, *predicting and tuning performance* of coexisting WiFi and ZigBee networks has been a challenging task, primarily due to the lack of corresponding simulators and analytical models.

In this dissertation, we addressed the aforementioned problems by presenting simulators and models for the coexistence of WiFi and ZigBee devices. Specifically, based on the energy efficiency and traffic pattern of three practical coexistence scenarios: disaster rescue site, smart hospital and home automation. We first of all classify them into three classes, which are non-sleeping devices with saturated traffic (SAT), non-sleeping devices with unsaturated traffic (UNSAT) and duty-cycling devices with unsaturated traffic (DC-UNSAT). Then a simulator and an analytical model are proposed for each class, where each simulator is verified by simple hardware based experiment. Next, we derive the expressions for performance metrics like throughput, delay etc., and predict them using both the proposed simulator and the model. Due to the higher accuracy of the simulator, the results from them are used as the ground truth to validate the accuracy of the model. Last, according to some common performance tuning requirements for each class, we formulate them into optimization problems and propose the corresponding solving methods. The results show that the proposed simulators have high accuracy in performance prediction, while the models, although are less accurate than the former, can be used in fast prediction. In particular, the models can also be easily used in optimization problems for performance tuning, and the results prove its high efficiency.

DEDICATION

To my family.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Radu Stoleru, for his endless support, encouragement, and guidance throughout the course of my Ph.D. study. I appreciate that he gladly accepted me to take my first step for becoming a researcher under his guidance. His patience allowed me to make steady and consistent progress. Whenever I was stuck on a problem, he has been always with me to give precious advice to help me moving forward. Without his help, I would not have completed this work.

I would also like to thank my committee members, Dr. Alexander Sprintson, Dr. Anxiao Jiang, and Dr. Jyh-Charn Liu for their insightful comments and support. Without the knowledge I learned from their classes and their constructive comments, this work would not have been possible.

My thanks also go to my lab colleagues, Won Myounggyu, Harsha Chenji, Amin Hassanzadeh, Mahima Suresh, Jay Chen, Chen Yang, Ala Altaweel and Mengyuan Chao, for gladly taking their time for discussion and collaboration and making my life at the Laboratory for Embedded & Networked Sensor Systems (LENSS) pleasurable.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis (or) dissertation committee consisting of Professor Radu Stoleru, Anxiao Jiang and Jyh-Charn Liu of the Department of Computer Science and Engineering, and Professor Alexander Sprintson of the Department of Electrical and Computer Engineering.

All work for the thesis (or) dissertation was completed by the student, under the advisement of Radu Stoleru of the Department of Computer Science and Engineering, and in collaboration with Mahima A. Suresh of the Department of Computer Science and Engineering, Yuhan Zhou of the Department of Electrical and Computer Engineering and Harsha Chenji of the Department of Computer Science and Engineering.

Funding Sources

Graduate study was supported by a scholarship from Texas A&M University.

This work was made possible in part by National Science Foundation (NSF) under Grant Number #1127449, #1145858. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

In recent years, due to the remarkable advantages of wireless networks such as "increased mobility and collaboration", "improved responsiveness", "economic access to information", "easier network expansion" and "enhanced guest access", we have witnessed a proliferation of wireless technologies that have now become ubiquitous [1]. For example, cell phone networks, wireless local area networks (WLAN), wireless personal area networks (WPAN), wireless sensor networks (WSN), satellite communication networks, and terrestrial microwave networks are very widely deployed nowadays [2].

Given the scarce availability of RF spectrum, many of these technologies are forced to use the cost-free 2.4GHz unlicensed frequency bands, which are also under the famous name industrial, scientific and medical (ISM) radio bands. The two most popular wireless techniques working on ISM are WiFi [3] and ZigBee [4] for WLAN and WSN, respectively. According to a report by ABI Research, the WiFi chipset accumulative shipments in 2016∼2020 will reach 20 billion [5], while such number for ZigBee realm during the same period will achieve 2.5 billion based on a report by ON World [6].

WLAN network traffic is mostly contributed by mainstream/entertainment applications that exploit the high data rate provided by the IEEE 802.11 MAC protocol [7] of WiFi. Wireless Sensor Networks traffic, on the other hand, has been concentrated towards applications such as building automation, health care monitoring and environmental sensing because of energy economy and low date rate imposed by the IEEE 802.15.4 MAC protocol [8] of ZigBee. Although there are many differences between 802.11 and 802.15.4 protocols, they both employ the Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) as the underlying wireless accessing mechanism, which renders the network operating efficiently without a central controller (i.e. in a distributed manner).

To avoid potential collisions between different wireless networks, people tend to distribute them into non-overlapping channels where signals never affect each other. For example, since there are three orthogonal channels for WiFi (i.e. channel 1, 6 and 11), two WiFi networks can be put on channel 1 and channel 6 respectively to eradicate collisions completely. However, owing to the aforementioned exponential growth in the number of WiFi and ZigBee networks deployments, almost all channels in ISM bands have been extremely crowded. Thus the approaches that utilize orthogonal channels to improve the performance of WiFi and ZigBee networks [9] [10] [11] are rendered ineffective, implying that their coexistence on the same channel is not avertible. Since the same channel coexistence is the focus of this dissertation, hereafter "coexistence" in the context always indicates "coexistence on the same channel", unless otherwise specified.

According to the difference in the transmission power level of WiFi and ZigBee devices, two types of coexistence are defined, i.e. single-cell coexistence (symmetric coexistence) and multi-cell coexistence (asymmetric coexistence). In the former scenario, these two types of devices are placed inside each other's communication ranges (i.e. they are basically within the range of $\sim 20m$ of each other), and they are able to sense each other's signals. While for asymmetric coexistence, ZigBee devices are placed inside the coverage of WiFi devices, but not vice versa, therefore only ZigBee can detect the transmissions of WiFi.

In symmetric coexistence, both WiFi and ZigBee can hear the voice of each other, their underneath CSMA/CA mechanism enables their coexistence intrinsically, i.e. any transmission will not be interrupted arbitrarily. Thus our concerns for symmetric coexistence scenario are more related to performance issues, i.e. can we predict or even tune the performancec of both type of devices.

On the other hand, since in asymmetric coexistence scenario WiFi devices are unaware of the existence of ZigBee, their CSMA/CA protocol cannot detect an ongoing transmission of the latter, i.e. WiFi never backoff for ZigBee. This situation is disastrous for ZigBee devices because their packets may be destroyed by a WiFi communication at any point of time. Due to the severity of this problem (from the perspective of ZigBee), there is much research has been taken to overcome it [12] [13] [14]. For example, [12] tried to analyze the distribution of the white space size in WiFi traffic (i.e. the length of the time gap in which no WiFi is transmitting), and then it segments the packet of ZigBee such that its size can be probabilistically fitted into an empty space of WiFi. Rather than analyzing the traffic characteristic of WiFi, [13] employs a more straightforward idea, i.e. it tries to fix the errors in a ZigBee packet (caused by a WiFi transmission) by using Reed Solomon code. While [14] adds a WiFi signal cancellation circuit to the PHY layer of ZigBee devices to eliminate the negative effects from WiFi transmission.

The solutions above for asymmetric coexistence have shown promising results in improving the performance of ZigBee, however, they are often very complicated, and usually becomes infeasible when network density is high. The main problem of this type of solution is, ZigBee always tries to adapt itself to WiFi, which is too passive to succeed in an intensive competition. Thus another type of solution emerges, i.e. making ZigBee proactive, and the most effective way to do so is letting WiFi knows the existence of ZigBee. In [15], the authors place a high power ZigBee busy toner which transmits dummy packet along with a normal packet (on different ZigBee channel) such that the CSMA/CA mechanism of WiFi begins to function for the ZigBee network.

Now, we want to argue that the idea of transforming asymmetric coexistence to symmetric coexistence is a better strategy because passive solutions reply too much on the competitor (i.e. WiFi), and if the latter is very eager for the resource, the passive ones might get starved easily. Therefore, symmetric coexistence is the focus of this dissertation.

1.1 Problems

1.1.1 Performance prediction

When designing a network, people are always wondering how well the network can actually perform, such that they can change the design when the result is not satisfactory. The most common approach to predict network performance is using simulators, such as ns-2 [16], OPNET [17] etc., which are accurate but usually very time costly. Another option is based on mathematical modeling, i.e. formulating the network as several equations through stochastic techniques and directly compute the performance needed. This type of solution is not as accurate as simulators but outperforms the latter in terms of speed. However, for symmetric coexistence network, so far not only the simulator is lacking, but also there is no mathematical model available to the community.

1.1.2 Performance tuning

In a symmetric coexistence network, it is expected to see performance degradation for both WiFi and ZigBee devices [18] [19], because one precious resource (i.e. the channel) needs to shared by more consumers. Although this seems reasonable, the fact is more complicated than that. By digging inside these two MAC protocols, we know that the design philosophy of them are quite different. Specifically, WiFi protocol is designed to be very aggressive to obtain maximum throughput, while ZigBee is quite passive so as to save energy to its best. Thus when coexisting with each other, ZigBee is very likely to lose the competition with WiFi; but if the passive ZigBee seizes the channel (with low probability), it occupies the channel for a long time due to its low data rate (i.e. 250Kbps), which in turn makes the WiFi suffer. In other words, their coexistence is possible, but is far from ideal. Therefore, given the performance objective constraint of one side (e.g. WiFi), our pursuit is to optimize the performance of the other side as much as possible by tuning the core parameters such as contention window (CW), etc.

For WiFi, for example, under the premise that the throughput has reached a requirement, we can reduce the aggressiveness by increasing its CW to give ZigBee more chances to use the channel, i.e. increasing the throughput of ZigBee. However, it is worth emphasizing that, the performance metric like throughput of WiFi and ZigBee are essentially a pair of contradictions, it is thus not possible to maximize both of them at the same time. Generally, we can formulate the performance tuning process as the following optimization problem,

$$\begin{aligned} \underset{CW,etc.}{\arg\max/\min} \quad & \textit{Metric of device type 1} \\ \text{subject to} \quad & \textit{Metric of device type 2} \geq \textit{Constraint}, \\ & \textit{Other constraints.} \end{aligned}$$

which uses an initial guesses of parameters CW as the input, and outputs the optimal values of them. Note that a valid metric can be the throughput, delay, energy consumption etc., and the metrics for different types are not necessarily the same (e.g. we can maximize the throughput of WiFi, under the delay constraint of ZigBee). Apparently, for optimization problems, simulators are not good choice because obtaining one result takes hours, not to mention searching a massive solution space for an optimum. Thus, the mathematical model becomes the only option.

In this dissertation, we propose mathematical models for predicting and tuning the performance of the symmetric coexistence network of WiFi and ZigBee. Simulators for symmetric coexistence are also proposed, which can not only be used for performance prediction, but also for verifying the accuracy of the proposed models.

1.2 Motivations

In this section, we motivate our research in detail by the following three different symmetric coexistence conditions.

- Disaster response, where many ZigBee sensors and WiFi routers are deployed within a small disastrous area. Then huge amount of data is generated by wireless sensors and responders (photos, videos etc.), which is then conveyed through a WiFi backbone network to the Command and Control (C2). The characteristic of this coexistence scenario is that all WiFi and ZigBee devices are very powerful (e.g. sufficient battery, long communication range), the traffic pattern is saturated, and the duty cycling mechanism is disabled to achieve better throughput. *Since different type of devices mule information with different privilege, the tuning requirement of giving one type of devices higher priority motivates the need for a mathematical model as well as a good simulator for the coexistence of non-sleeping devices with saturated traffic.*

- Smart Hospital, where ZigBee sensors are placed around the body of patients, and the host instruments of the sensors are equipped with WiFi to enhance the mobility. The sensors collect data from the patients and send it wirelessly to the host instruments, which then report the analyses to the doctors/nurses through WiFi. In this type of coexistence, the WiFi and ZigBee devices are also powerful, the traffic pattern, however, is usually unsaturated, and they are not allowed to go to sleep in the consideration of delay constraints and reliability. *It is important for the data from the sensors to be delivered before some deadline, thus needs of maximizing the throughput of WiFi while ensuring service time constraint of ZigBee motivates us to find an analytical model and a corresponding simulator for the coexistence of non-sleeping devices with unsaturated traffic.*

- Home Automation, where many ZigBee sensors/actuators are deployed in the house, and many WiFi devices such as routers, smartphones, tablets are coexisting with them. Since most devices are powered by batteries, the duty cycling mechanism is always enabled. Obviously, the characteristic of this kind of coexistence is the unsaturated traffic plus duty cycling. *Usually the hand-held WiFi devices are expected to work as long as possible before getting charged, and the sensors need to work for years before the batteries get changed, the tuning requirement of lifetime extension for both type of devices motivates us to have a mathematical model and a simulator for the coexistence of duty-cycling devices with unsaturated traffic.*

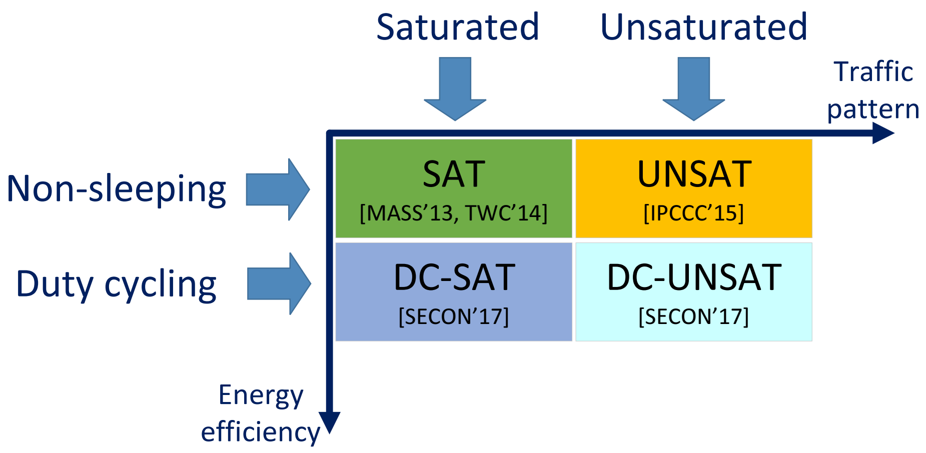


Figure 1.1: Classification of coexisting WiFi and ZigBee

Based on the motivations above, we classify the symmetric coexistence network in terms of traffic pattern and energy efficiency as in Figure 1.1. The research objective is to model and simulate the single-cell coexistence for three different cases, i.e. saturated traffic without duty cycling (SAT), unsaturated traffic without duty cycling (UNSAT) and unsaturated traffic with duty cycling (DC-UNSAT), such that we can predict and tune performances for the devices involve. Note that although the scenario of duty cycling devices with saturated traffic is unreasonable (named DC-SAT), it actually serves as a critical part in the approximation for the modeling of DC-UNSAT.

1.3 Dissertation Statement

Due to the explosive deployment of wireless technology such as WiFi and ZigBee, coexistence network of them has become more and more pervasive. For the case of multi-cell coexistence, transforming it into a single-cell one (i.e., making ZigBee proactive) is more advantageous than the solutions which adapt ZigBee to WiFi because it is near impossible for a passive device to win a very intensive competition. We claim that symmetric coexistence will be the future of this hybrid coexistence network, and thus is the focus of this dissertation. As predicting and tuning network performance are of great importance for network designers, the tools which can help improving the designs are thus highly demanded. However, for symmetric coexistence network there is neither simulator nor mathematical model that is available to use so far.

This research is trying to fill up this gap, and it proposes the first models and simulators for symmetric coexistence network, which can be used for the aforementioned purpose. Specifically, by showing three different coexistence scenarios in terms of traffic pattern and energy efficiency, it classifies symmetric coexistence into three types and proposes a model and a simulator for each type. Then it validates the accuracy of the models by comparing the performance predictions from the models against the ones from the proposed simulators. Lastly, to demonstrate the usages of the models on performance tuning, it formulates three requirements as three optimization problems, one for each type, and obtains three optimal values for the chosen parameter sets by solving the corresponding problems.

1.4 Main Contributions

The contributions of this research are outlined as follows:

For chapter "**Coexistence of non-sleeping WiFi and ZigBee with saturated traffic**",

- it presents the first Monte Carlo based coexistence simulator for SAT;
- it proposes an accurate Markov Chain model for SAT;
- it presents the first analysis and closed form expressions to predict throughput;
- it presents two performance tuning methods that achieve priority and fairness, respectively;
- it validates the accuracy of the model and the feasibility of the tuning method through extensive simulations.

For chapter "**Coexistence of non-sleeping WiFi and ZigBee with unsaturated traffic**",

- it presents the first ns-3 based coexistence simulator for UNSAT;
- it proposes a new accurate Markov Chain model for UNSAT;

- it proposes a $M/G/1$ queueing model that accurately compute buffer empty probabilities;
- it presents the first analysis and closed form expressions to predict throughput and packet delivery delay;
- it presents a performance tuning method, that maximizes WiFi throughput while satisfying 802.15.4 packet delivery delay constraints;
- it validates the accuracy of the model and the feasibility of the tuning method through extensive simulations.

For chapter "**Coexistence of duty-cycling WiFi and ZigBee with unsaturated traffic**",

- it extends the Monte Carlo based coexistence simulator such that it supports the simulation of DC-UNSAT;
- it proposes an accurate Markov Chain model for WiFi power saving mode (PSM), as a step stone to the entire DC-UNSAT modeling objective;
- it presents an approximation based yet accurate model for DC-UNSAT;
- it proposes a $M/G/1$ queueing model that can accurately compute buffer empty probabilities;
- it presents the first analysis and closed form expressions to predict throughput and energy consumption;
- it presents performance tuning method to account for energy consumption;
- it validates the accuracy of the model and the feasibility of the tuning method through extensive simulations.

1.5 Organization

This dissertation is organized as follows. In Section 2, the state of the art and prior works related to this dissertation are reviewed. Section 3 proposes a Monte Carlo simulator, a Markov Chain model for SAT, Section 4 proposes a ns-3 based simulator and a new Markov Chain and M/G/1 based model for UNSAT, and Section 5 describes a extended Monte Carlo simulator, a Markov Chain model for WiFi PSM and an approximation based model for DC-UNSAT. Finally, Section 6 concludes this dissertation and illustrates its future perspective.

2. STATE OF THE ART

2.1 Background

Much research has revealed the performance degradation in coexisting WiFi and ZigBee network [18] [19] [12] [15] [13] [20], where [18] [19] demonstrate this phenomena for symmetric coexistence scenario, while [12] [15] [13] [20] show the problem in asymmetric coexistence networks.

First of all, in symmetric coexistence case, performance degradation are observed on both WiFi and ZigBee sides [18] [19]. Specifically, the results of [18] indicate that packet error rate is more than 90% for 802.15.4 (the degradation of 802.11 is not shown, but can be expected), which the experimental results in [19] show that the throughput loss of IEEE 802.11 can be up to 30%, and may reach 60% when 802.15.4 duty cycles is large. The main reason for this problem is that although these two CSMA/CA based MAC protocols are still functioning in coexistence condition, nevertheless they do not work well because of the huge yet critical differences between them. More precisely, since 802.11 is very aggressive compare to 802.15.4, it can grab the transmission chance more easily than the latter, however, because the data rate for 802.15.4 is low, whenever it grabs a chance to send, it occupies the channel for a long time hence hurts 802.11 significantly.

On the other hand, in [12] [15] [13] [20], all the authors point out that for asymmetric coexistence case (i.e. when ZigBee devices are located far from WiFi), the performance of ZigBee degrade significantly ($> 90\%$), especially when the WiFi traffic is heavy. The reason for this is that WiFi devices are agnostic of the ongoing transmission of ZigBee, which makes ZigBee devices vulnerable to WiFi transmissions.

2.2 General Solutions

Frequency hopping is a general solution for both symmetric coexistence and asymmetric coexistence because whenever the performance becomes too bad (especially for ZigBee in asymmetric coexistence), the victims always have the option to use an emptier channel. Frequency hopping wisely manipulates the usage of the frequency channels that are available. This solution implements channel assignment algorithms that target multi-channel networks. Based on the detected interference level the nodes are assigned different channels to reduce the impact of interference.

Examples of solutions that adopted this technique are [9] [10] and [11]. Han et al. presented a centralized interference mitigation mechanism that is based on dynamic channel selection for cluster tree ZigBee networks [9]. Interference is detected by sensing the channel or using packet error rate. If high level of interference is affirmed, the coordinator announces the start of multi-channel operation mode in its cluster. In this mode, the nodes in the cluster use channel hopping and transmit data on different channels determined by the coordinator. After monitoring the transmission quality, the coordinator finds the best channel and assigns it as the new channel then return back to the single-channel operation mode. [10] and [11] also use the channel hopping idea but without a master nodes who select the best channel for other devices. Each device tracks the current channel of its neighbors and also notifies them when itself changes the channel. However, to determine the interference level and choose the best new channel, different methods are used.

Recently, cognitive radio has become a promising approach to mitigate the interference from other wireless techniques [21] [22] [23] [24] [25] [26]. In principle, cognitive radio is a radio or system that senses its operational electromagnetic environment and can dynamically and autonomously adjust its radio operating parameters to modify system operation, such as maximize throughput, mitigate interference, facilitate interoperability, access secondary markets. Compare to the aforementioned frequency hopping solutions, cognitive radio selects the best working channel more intelligently, thus is usually more complicated. [21] and [23] surveyed the applications, challenges and research trends for cognitive radio in wireless sensor networks, while [22] [24] [25] and [26] studied the interference in cognitive networks by some theoretical means such as statistical model etc.

The main problem in frequency spacing solutions is the limited number of available channels, especially, when interfering with networks that has large bandwidth frequency channels, e.g. a channel in WiFi interferes with three or four channels in ZigBee. Moreover, there is no accurate methodology to determinate the level of interference based on SINR, RSSI, channel quality etc., and an accurate determination of whether a channel is emptier is also not trivial.

2.3 Asymmetric Coexistence Specific Solutions

Since in asymmetric coexistence network, ZigBee devices cannot affect WiFi, the first type of strategy for ZigBee is "adaptation", i.e. the solutions under this category try to adapt the behavior of ZigBee to reduce interference from WiFi [12] [13] [14] [27].

White spaces in WiFi networks were exploited in [12]. The interference between ZigBee and WiFi was studied empirically based on real life traffic traces and used to model white spaces in bursty WiFi networks. Also, they modeled the behavior of Zigbee links under WiFi interference where the probability of collision was analyzed mathematically. Based on that, the authors proposed a frame control protocol called WISE which predicts the white spaces length of WiFi using the developed model and adjusts the frame size of ZigBee accordingly to fit into the available free space.

[13] and [14] tried to reduce the interference by WiFi from a different perspective.

Specifically, [13] utilized a Forward Error Correction based approach named Reed Solomon Code to fix the packet errors caused by 802.11 interference; while [14] recovers ZigBee packet during WiFi/ZigBee collision by extracting WiFi packet first (because WiFi signal is very strong compare to ZigBee), and then subtracts WiFi interference and decodes ZigBee packet. Moreover, [13] also defined two types of interference: symmetric (WiFi device can hear Zigbee transmission) and asymmetric (WiFi device cannot hear Zigbee transmission), which is same as we do.

Multiple-antenna and beamforming techniques were utilized in [27] where a cognitive smart grid network protocol has been proposed. ZigBee nodes monitor the transmission time of frames needed by WiFi devices which is used to compute the beamforming vector for the antennas. This vector guarantees a satisfactory rate for ZigBee nodes while sharing the spectrum with WiFi devices and allows data transmission with the interfering WiFi networks simultaneously.

Note that this strategy (i.e. adaptation) works unsatisfactorily when WiFi traffic load is heavy because there is no enough space to "adapt". Therefore, unlike adapting ZigBee to WiFi, the second strategy is letting WiFi know the existence of ZigBee such that WiFi are forced to share the channel with ZigBee (i.e. more fairly).

In [15], Zhang et al. placed an extra powerful ZigBee device as a busy toner in the original ZigBee network. The busy toner has two antennas and is able to work on two different 802.15.4 channels. The novelty of this work is that, the toner receives the transmission from the ZigBee network on one channel, and uses the other channel to transmit dummy packet such that WiFi devices are forced to backoff. This work needs the two 802.15.4 channels to be covered by one WiFi channel, which is feasible because one 802.11 channel is around four times wider than a 802.15.4 one. Radunovic et al. [20] redesign the preamble of ZigBee based on a key observation that longer preamble sequence can be detected easier. In this case, WiFi will sense the presence of ZigBee, and thus backoff.

Fake WiFi packets were used in [28] where the authors proposed WiCop framework that tends to exploit the white spaces of interfering WiFi networks. In WiCop each ZigBee network has a policing node that implements WiCop framework. Before the start of ZigBee activity the policing node broadcasts a fake WiFi packet that contains a preamble, physical header, and the packet duration, possibly without data. This packet forces nearby WiFi nodes to mute and ZigBee nodes may exploit this interval for data transmission. Another mechanism that was also proposed in [28], is DSSS (Direct Sequence Spread Spectrum) nulling. A band-pass filter is used to reshape a DSSS jamming signal, which result from the continuous transmission of repeated WiFi preambles, to have smaller bandwidth that can jam the WiFi devices on a specified channel while not jamming all Zigbee channels in the same frequency band. Thus, the un-jammed ZigBee channels can still be used for communication.

The aforementioned solutions for asymmetric coexistence, although are promising in the corresponding setups, usually involve significant changes to the PHY layer or MAC layer or both, or even need extra devices (i.e. transform asymmetric coexistence to symmetric coexistence). Moreover, the techniques used are complicated and not applicable to symmetric coexistence, where CSMA/CA works for both types inherently.

2.4 Symmetric Coexistence Specific Solutions

In symmetric coexistence scenario, since WiFi and ZigBee know the existence of each other, their default MAC layers (i.e. CSMA/CA) are already functioning. However, compare with CSMA/CA, coordination based methods (e.g. TDMA) usually provide better performance because the devices involve are guaranteed to have certain level of performance, thus is beneficial especially for the non-aggressive ZigBee.

The main difficulty in coordination is, although both types of devices can detect each other's signal, they actually have no clue about the information embedded, which means they cannot communicate directly. There are two types of solutions for this, one is using central dual-radio gateways to make the communication between WiFi and ZigBee possible [29], the other is utilizing some special techniques [30].

Nodes clustering and dual-radio nodes design were exploited in [29] to mitigate interference in ZigBee networks coexisted with WiFi networks. The nodes were grouped into clusters where ZigBee was used for intracluster communication and WiFi was used for inter-cluster communication. This was enabled by the usage of cluster heads that are equipped with dual radios one for ZigBee and another for WiFi. Data aggregation and delayed transmission were used to reduce interference between the two heterogeneous radios as well as with nearby nodes.

Kim et al utilized time shifting of beacons to mule information across WiFi and ZigBee devices [30]. Specifically, by observing that one type of device can detect the beacon (i.e. a spike signal) of the other type, it proposed a method to accurately compute the difference between the actual beacon time and the supposed beacon time, which is used to convey a specific meaning (i.e. data). And to facilitate the accurate capture of the actual beacon time, it also proposed a statistics based method by observing the channel condition for a sufficiently long time.

The problem for [29] is that it is a centralized method, which needs an extra expensive device, and is not scalable. Although [30] has shown interesting results, it requires that all devices are capable of sending a beacon, thus is not feasible for non-AP devices. Moreover, since the achievable throughput is quite slow (depending on the length of the beacon interval), it is only suitable for very simple applications.

2.5 Coexistence Modeling

To understand the performance of WiFi and ZigBee devices in symmetric coexistence, few papers have proposed simple models for it.

Howitt et al. [31] studied the effect of IEEE 802.15.4 devices on IEEE 802.11b WLANs. They developed an analytical model for the probability of packets collision in WLANs that is caused by the activity of Zigbee-based devices. The developed model was used to set an upper limit on the cluster size of zigbee-based nodes to reduce the interference on WiFi networks.

The throughput of Zigbee networks under the presence of WiFi interference was analyzed in [32]. The model is based on Markov Chain mathematical system and assumed that the Zigbee network has no effect on WiFi. The researchers found that the increase of the WiFi network packet rate caused a decrease in the Zigbee network throughput. Yuan et al. [33] also analyzed the throughput of Zigbee network under the existence of 802.11b/g network. The obtained simulation and experimental results were similar to other studies, the Zigbee throughput dropped severely in the present of WiFi activity which matches the developed mathematical model.

However, these works only consider the impact from one type to another, i.e. ignoring the possible effect from the victim to the interferer, which fail to depict the whole picture in symmetric coexistence. Therefore, our work tries to complete the picture by modeling the interactions between WiFi and ZigBee devices.

2.6 Performance Prediction

2.6.1 Simulation based studies

Generally speaking, there are two main types of simulation based approaches for predicting performance, i.e. based on existing network simulators or based on Monte Carlo methods. For the former, common choices are ns-2 [16], ns-3 [34], OMNeT++ [35], OPNET [17], QualNet [36], TOSSIM [37] etc., where the first three are open sourced free simulations while the other two are commercial ones. However, when the property of the objective network is beyond the functionality of a simulator (for example, it runs a special protocol), one can either write a customized module for the existing simulators or write Monte Carlo simulation code. Sometimes, Monte Carlo simulation is preferred because in prototype stage people might not want to worry about network typologies or environment factors such as channel condition etc.

There are plenty of papers that study the network performance by simulations, either using simulators or Monte Carlo methods [38] [39] [40] [41] [42]. [38] attempted to analyze the performance of the wireless and wired computer networks through OPNET simulator. Specifically, for wired networks, the performance metrics such as delay and throughput have been investigated by changing transmission links and load balances. While in wireless networks the metrics like delay, number of retransmission attempts and throughput have been estimated by varying PHY layer characteristic and buffer sizes. [39] studied the performance of some routing protocols for wired and wireless network using the ns-2 simulator. It analyzes the packet delivery ratio and end-to-end delay for protocols such as ADOV, DSDV, DSR, TORA and ZRP. In [40], the authors conducted a detailed simulation study of stateless anycast routing in a mobile wireless ad hoc network by using a simulator called Winsim [43]. While Mallanda et al. [41] introduced the OMNeT++ simulator, which serves as a faster substitution for the ns-2 for analyzing the performance of wireless sensor networks. Paper [42] proposed a model for WSN simulation, and summarizes the key factors for a good simulation tools, such as reusability, availability, performance, scalability, graphical etc. This paper can be regarded as a guideline for designing a new simulation framework.

In terms of the application of the Monte Carlo based methods, [44] used this idea to simulate a 802.15.4 network to analyze some detailed probabilities like the probability to start sensing, etc.

2.6.2 Modeling based studies

There are plenty of literature that try to address the performance analysis/prediction problem for both WiFi (i.e. 802.11) and ZigBee (i.e. 802.15.4).

In [45], the authors proposed a game theory based model to describe IEEE 802.11 DCF and designed a simple Nash equilibrium backoff strategy to achieve fairness. Moreover, several queuing theory based models are designed using the uniqueness of the fixed point to analyze the performance of IEEE 802.11 DCF [46] [47].

A very well known model for IEEE 802.11 DCF networks was proposed in [48] by Bianchi in 2000, where a two-dimensional Markov Chain was created to characterize the backoff process of each node. It is well validated by simulation results and demonstrated to be a powerful, yet simple, analytical tool to evaluate the performance for saturated WiFi network, i.e., each 802.11 node always has data to transmit.

To further improve the accuracy of Bianchis model, huge amount of works tried to refine it, including but not limited to [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60] which made more practical assumptions such as freezing of backoff counters [49], [51], [52], [53], finite retransmission attempts [50], [54] and imperfect channel conditions [55], [56], [57], [58]. Moreover, a great deal of effort was made to extend Bianchi's model to account for unsaturated networks [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74].

More recently, researchers have attempted to analyze the networks with duty cycling devices. Zheng et al. [58] tried to model the IBSS 802.11 PSM protocol, where a transient analysis is used. However, it relies on an unrealistic assumption of the

MAC service time being either exponential or deterministic. While [75] modeled the 802.11 IBSS PSM as a pure 3-D Markov Chain, where the key probability of a node being reset in a cycle is obtained by simulation, which makes this model futile. In stead of 802.11 PSM, [76] tried to model X-MAC, which is a duty cycling protocol for WSN. This paper analyzes the medium access process and models the transmission queue as a Markov Chain to solve the model, but it ignores the CSMA/CA mechanism for medium access, which attenuates its accuracy.

Researchers have also developed similar models for other protocols [44] [77] [78] [79]. Pollin et. al. [44] proposed an accurate model for IEEE 802.15.4, for both saturated and unsaturated traffic. It is paramount to note that these models are all for single MAC protocols, i.e., non coexistence. Notably, researchers have attempted to build models for coexisting networks [80] [81], but all of them only studied the coexistence of the variants of the 802.11 MAC protocol, e.g., 802.11b and 802.11g.

2.7 Performance Tuning

For performance tuning of wireless networks, methods that employ CW size adaptation have been proposed [82] [83] [84] [85]. These methods are either centralized or distributed. They typically propose models for throughput, delay and fairness, then make estimates for collision probabilities and number of devices. Finally, by solving various optimization problems, optimal CW sizes are derived. However, it is often very difficult to accurately estimate the number of devices in the distributed CW size adaptation methods. To address this problem, recently, game theoretic solutions were proposed [86] [87]. Essentially, these solutions optimize a payoff function defined as the difference between a utility function (e.g. throughput) and a price function (e.g. collision rate). Since each device in a game theoretic solution only observes the price, and needs not to know the number of devices, the solution is distributed inherently. However, all existing performance tuning methods employing game theoretic approaches are for devices of the same type, and can not be employed in coexistence scenarios (symmetric or asymmetric).

3. COEXISTENCE OF NON-SLEEPING WIFI AND ZIGBEE WITH SATURATED TRAFFIC [*]

In this chapter, we discuss a mathematical model as well as a simulator for symmetric coexistence of non-sleeping WiFi and ZigBee with saturated traffic (i.e. SAT).

3.1 Background

3.1.1 The WiFi MAC Protocol

The standard MAC protocol for non-sleeping WiFi devices is named IEEE 802.11 distributed coordination function (DCF) (802.11 for short), which is discribed in the 802.11 standard [88]. In this section, we will briefly summarize 802.11. For efficiency reasons, 802.11 employs a discrete-time backoff scale, i.e. all time involved is slotted and one time slot is represented by σ.

A device with a new packet to transmit randomly chooses an integer from the current contention window (CW) as its backoff counter (BC) , and then it monitors the channel activity. If the channel is not idle, it persists to monitor the channel until it is measured idle for a distributed inter-frame space (DIFS). If the channel is measured idle for DIFS, the device begins to count down its BC. The BC is decremented as long as the channel is sensed idle at the beginning of each time slot, “frozen” when a transmission is detected on the channel, and reactivated when the channel is sensed idle again for more than a DIFS. The device transmits when the backoff counter reaches zero.

If the transmission succeed, the process is finished, otherwise, it doubles the

[*]Reprinted from “On Modeling the Coexistence of 802.11 and 802.15.4 Networks for Performance Tuning” by Wei Zhang, Mahima A. Suresh, Radu Stoleru, Harsha Chenji, 2014. IEEE Transactions on Wireless Communications, Vol. 13, 5855-5866, Copyright 2014 by IEEE.

contention window (i.e. reaching a new backoff stage (BS)), and restart the process by choosing BC from the new CW and monitoring the channel. Note that the contention window only doubles if it has not reached a maximum value W_m.

The aforementioned 802.11 protocol is demonstrated in Figure 3.1.

3.1.2 The ZigBee MAC Protocol

The standard MAC protocol for ZigBee is IEEE 802.15.4 [89], which is, however, not considered in this research because the most popular WSN operating system TinyOS uses BoX-MAC as its MAC protocol [90]. In fact, BoX-MAC is a simplified version of IEEE 802.15.4 because there are only two backoff stages for the former while the latter has five stages.

Each BoX-MAC device maintains three variables: CW BC and CS. CW is the *Contention Window* size. A *Backoff Counter* BC is randomly chosen in $[0, CW-1]$. Since BoX-MAC can transmit data only if the channel is sensed idle for two consecutive times slots, the variable CS (the *CCA Stage*) is used to represent the number of successful CCAs. It decrements if channel is sensed idle and be reset otherwise. The exact mechanism is depicted in Figure 3.2. Initially, $CS = 2$, CW equals the initial contention window CW_{init}, BC is randomly chosen in $[0, CW-1]$, and is decremented every time slot, until it reaches 0. When $BC = 0$, BoX-MAC enters CCA stage, when the MAC layer requests PHY to perform CCA in the following two consecutive time slots. In the CCA stage, if channel is sensed idle, CS is decremented until 0, time when the packet will be transmitted; otherwise (i.e., the CCA fails) CS will be reset to 2, CW will be assigned the congested contention window CW_{cong} value. Then, the same aforementioned backoff rules are applied until data is successfully transmitted.

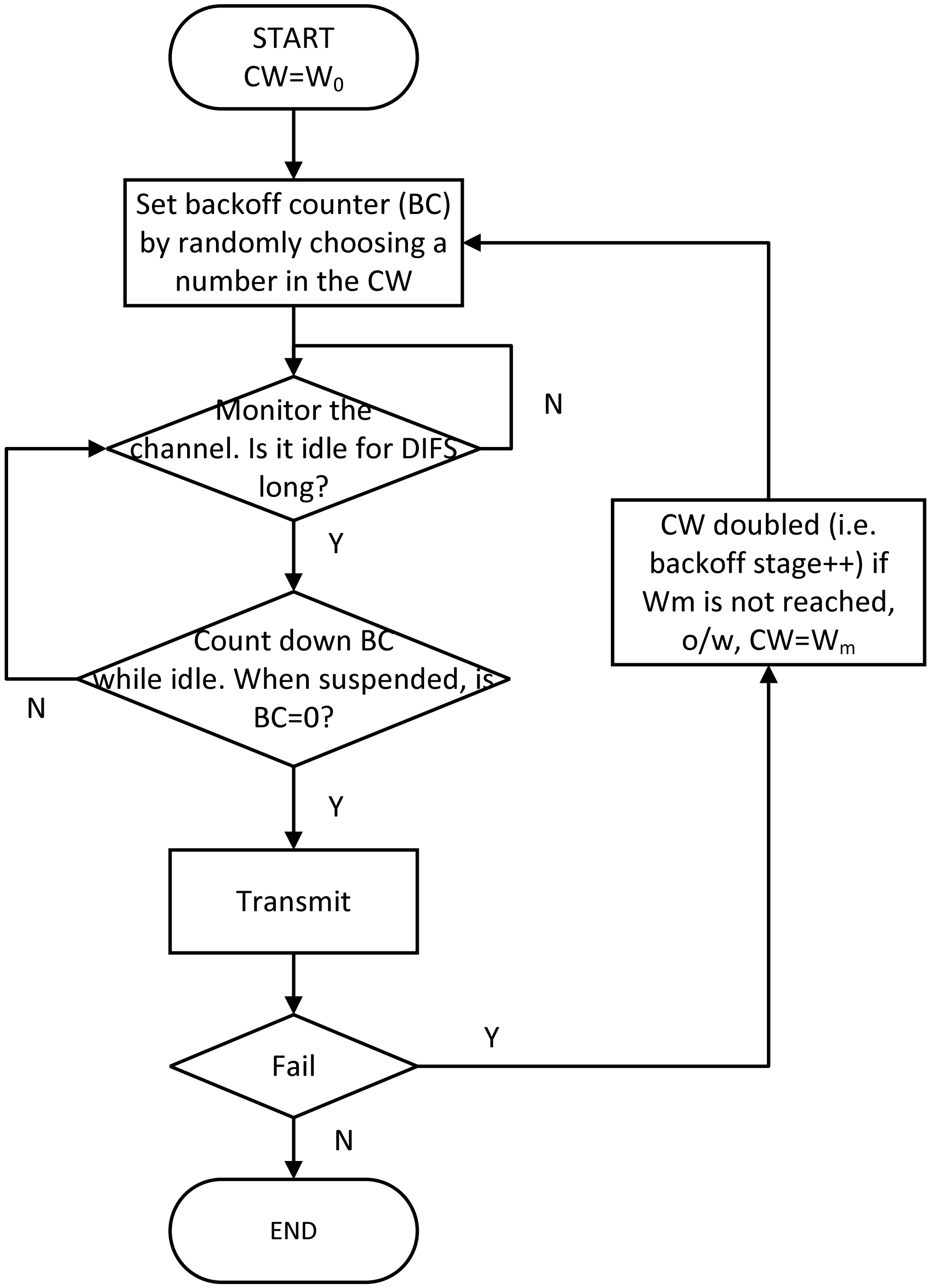

START
$CW=W_0$
Set backoff counter (BC) by randomly choosing a number in the CW
Monitor the channel. Is it idle for DIFS long?
N
Y
Count down BC while idle. When suspended, is BC=0?
N
Y
Transmit
Fail
Y
N
END
CW doubled (i.e. backoff stage++) if Wm is not reached, o/w, $CW=W_m$

Figure 3.1: The IEEE 802.11 DCF protocol.

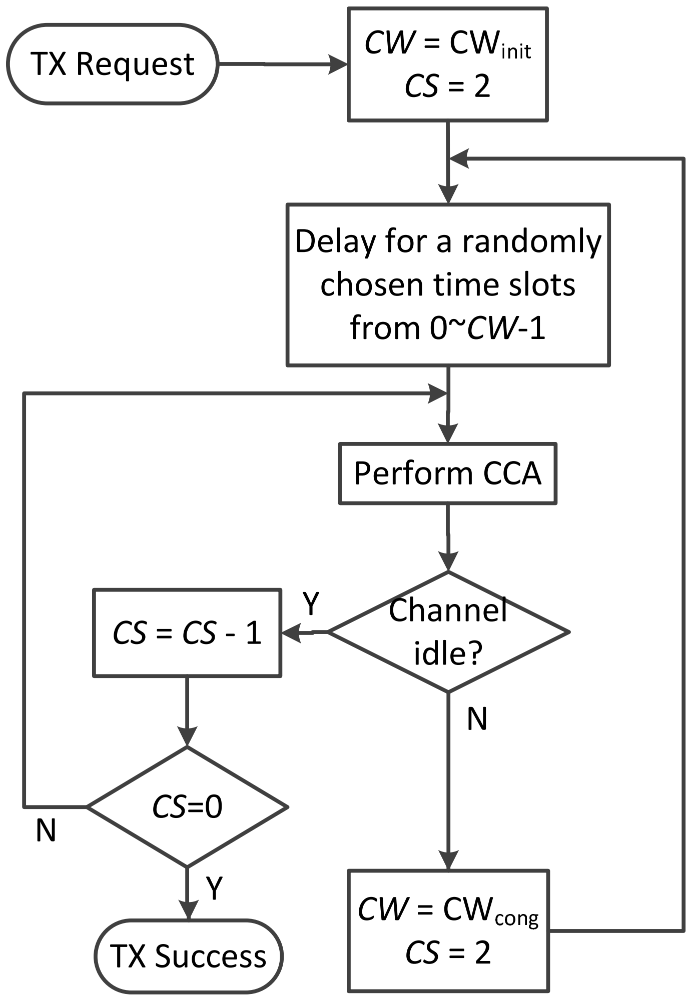


Figure 3.2: The backoff mechanism for the BoX-MAC protocol.

3.2 Mathematical Model for SAT

In this section, we present the mathematical analysis for the coexistence of 802.11 DCF and BoX-MAC. We assume that the traffic is saturated and that the devices are within communication range of each other (i.e., single-cell coexistence). To the best of our knowledge, this is the first analysis for coexistence of these two classes of devices. For analysis, we use independent analytical models for the two MAC protocols, followed by their steady state analysis. A novel Markov Chain based *channel model* estimates the channel busy probability. These enable us to predict network performance (e.g. compute saturation throughput, etc.) for the symmetric coexistence model.

Both 802.11 DCF and BoX-MAC are modeled as Markov Chains. By the Markov property, state transition probabilities are dependent only on the most recent states. We model a state transition at the end of a time slot whose size is dependent on the protocol. It is hard to analyze coexistence if two devices have different time slot sizes. In this case, BoX-MAC has a time slot that is 3 times that of 802.11. To account for the difference in slot sizes, while maintaining the Markovian property, we add two dummy states to each BoX-MAC state, each with transition probability 1, as explained below. Therefore, each state in the Markov Chain corresponds to one third of a BoX-MAC time slot, i.e., each BoX-MAC slot is divided in three equal time slots.

3.2.1 Markov Chain Model for IEEE 802.11

The Markov Chain model that we propose for 802.11 DCF is depicted in Figure 3.3. We extend the Bianchi model [48] to include backoff freezing and we adopt ideas for accurate modeling, as in Felemban and Ekici [74]. Our model uses two parameters $b(t)$ and $r(t)$ for the high level *stage* (e.g., backoff, transmission, etc.) and *counter* respectively. The counter is used as an indicator for the number of time slots in each stage. Each state is represented as $(b(t), r(t))$. This model does not account for inter-frame spacings when the channel is sensed busy in a backoff state.

The stages where $b(t) \geq 0$ correspond to backoffs. When an 802.11 device attempts to transmit a packet, it starts at state $(0, k)$, where k is a random number between 0 and $CW_{\min}$, ($CW_{\min}$ is the minimal contention window). The channel is sensed in each time slot (state). Similar to BoX-MAC, any channel sensing state

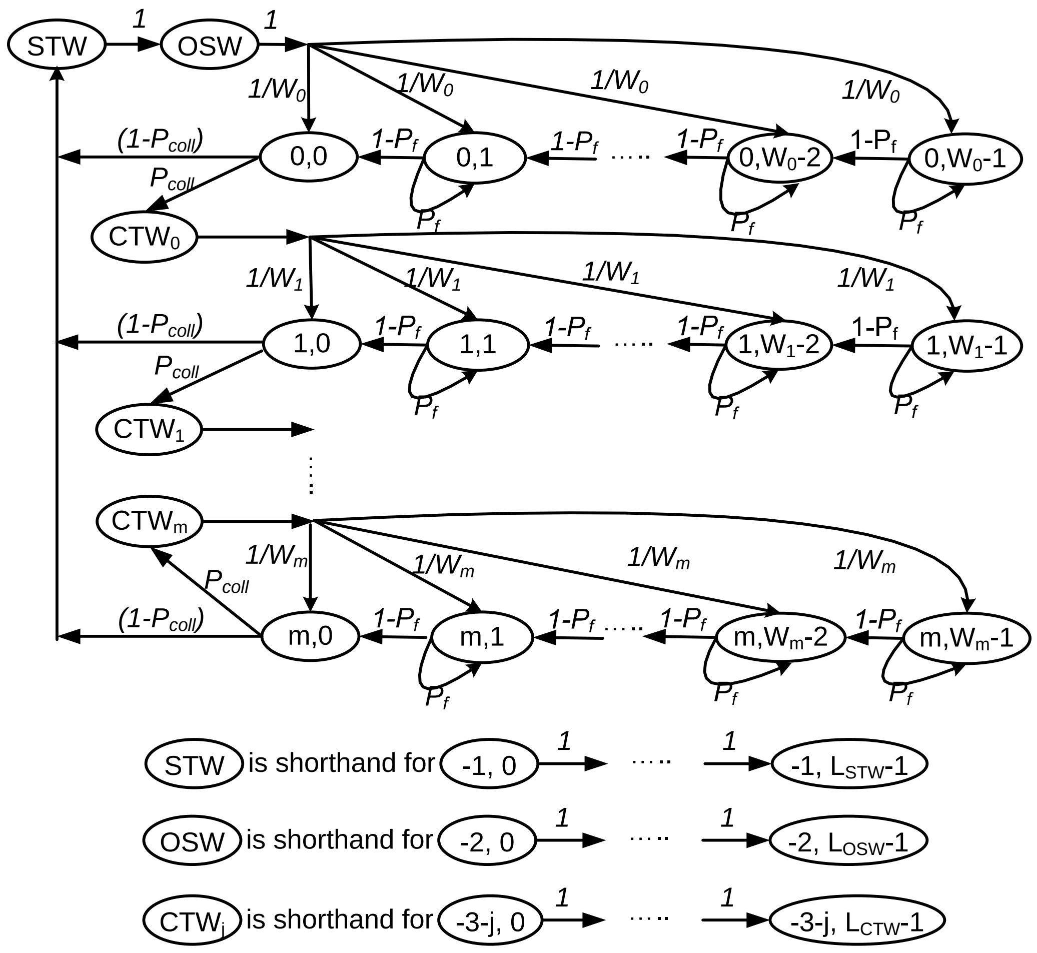

STW
OSW
1
1
1/W0
0,0
0,1
0,W0-2
0,W0-1
1-Pf
Pf
(1-Pcoll)
Pcoll
CTW0
1/W1
1,0
1,1
1,W1-2
1,W1-1
CTW1
CTWm
1/Wm
m,0
m,1
m,Wm-2
m,Wm-1
STW is shorthand for -1, 0 → ….. → -1, LSTW-1
OSW is shorthand for -2, 0 → ….. → -2, LOSW-1
CTWj is shorthand for -3-j, 0 → ….. → -3-j, LCTW-1

Figure 3.3: The Markov Chain describing 802.11 DCF.

for 802.11 is modeled by a process named *SENS_W*. If the channel is sensed busy (with probability P_f), the device remains in the same state, i.e., the backoff counter freezes. If the channel is free, the backoff counter is decremented. When the backoff counter reaches 0, if the channel is sensed idle, the 802.11 device transmits the packet and waits for an acknowledgement (ACK). If an ACK is not received (e.g., due to collision), the device tries to transmit the packet again, with a current contention window W_j, being in backoff stage j. The probability that a transmitted packet collides with others is P_{coll}. Collisions occurring after backoff stage j (i.e., $(j, 0)$), where $j = 1, 2, \ldots, m$, are represented by states CTW_j. In these states $b(t) = -3-j$ and $r(t) = 0, 1, ..., L_{\text{CTW}} - 1$. When backoff stage m is reached, further retries are still made from the same backoff stage and have the maximum contention window size $CW_{\max}$. We note that $m = log_2 \frac{CW_{\max}}{CW_{\min}}$.

When an ACK is received, i.e., the packet is successfully sent, the next packet transmission is attempted after experiencing some delay from the operating system, represented as state OSW, where $b(t) = -2$, and $r(t) = 0, 1, ..., L_{\text{OSW}} - 1$. A successful transmission is represented as state STW, where $b(t) = -1$, and $r(t) = 0, 1, ..., L_{\text{STW}} - 1$. Here, L_{STW} and L_{CTW} are functions of packet size, the available bandwidth and MAC protocol specific delays such as interframe spacing and ACK timeout. L_{OSW} is obtained from hardware experiments. Similar to the notation W_j' in BoX-MAC, we extend W_j such that it can denote L_{STW}, L_{CTW} and L_{OSW} of 802.11.

The single step transition probabilities, as defined by the Markov Chain for

802.11, are:

$$
\begin{aligned}
&Pr[j,k-1|j,k] = 1-P_f \\
&Pr[j,k|j,k] = P_f \\
&Pr[-3-j,0|j,0] = P_{coll} \\
&Pr[j,k|-3-(j-1),L_{\mathrm{CTW}}] = \frac{1}{W_j}, && j=1,...,m \\
&Pr[m,k|-3-m,L_{\mathrm{CTW}}] = \frac{1}{CW_{\max}} \\
&Pr[-1,0|j,0] = 1-P_{coll} \\
&Pr[-2,0|-1,L_{\mathrm{STW}}] = 1 \\
&Pr[0,k|-2,L_{\mathrm{OSW}}] = \frac{1}{CW_{\min}} \\
&Pr[j,k|j,k-1] = 1, && j=-1,-2,...-3-m
\end{aligned}
$$

where $j = 0,...,m$ if not specified explicitly; $k = 0,1,...W_j - 1$, where $W_j = CW_{\min}2^j$ when $j = 0,1,...,m$; and $W_{-1} = L_{\mathrm{STW}}$, $W_{-2} = L_{\mathrm{OSW}}$, $W_l = L_{\mathrm{CTW}}$ when $l = -3,-4,...,-3-m$.

In order to model the coexistence of WiFi with other types of devices, we need to compute the variables P_{coll} and P_f. These probabilities reflect the state of the channel.

3.2.2 Markov Chain Model for BoX-MAC

The BoX-MAC protocol is a simplified version of the IEEE 802.15.4 protocol, which makes it tenable for mathematical analysis. Similar to the approach in [44], we model the BoX-MAC protocol as a Markov Chain (shown in Figure 3.4). Let $s(t)$ and $c(t)$ be the stochastic processes representing the high level *stage* and *counter*, respectively. The process where each state is represented by $(s(t), c(t))$ can be mod-

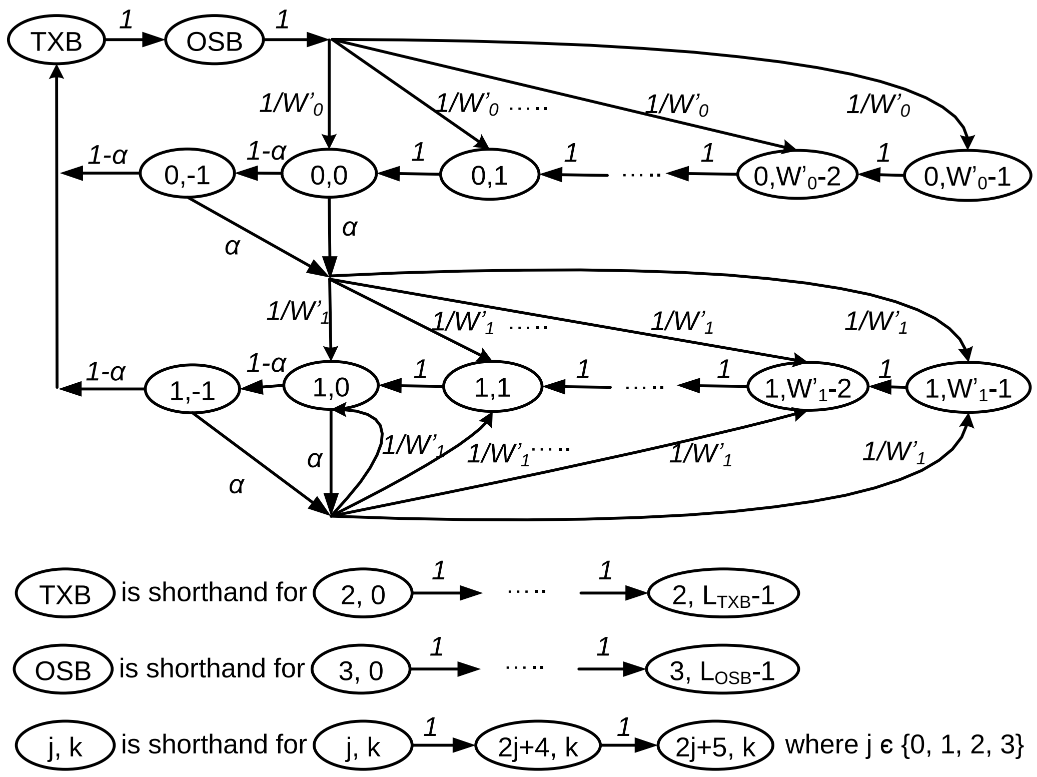

TXB
OSB
1
1/W'_0
1-α
0,-1
0,0
0,1
0,W'_0-2
0,W'_0-1
α
1/W'_1
1,-1
1,0
1,1
1,W'_1-2
1,W'_1-1
TXB is shorthand for 2, 0 → ..... → 2, L_TXB-1
OSB is shorthand for 3, 0 → ..... → 3, L_OSB-1
j, k is shorthand for j, k → 2j+4, k → 2j+5, k where j є {0, 1, 2, 3}

Figure 3.4: The Markov Chain describing the BoX-MAC protocol.

eled as a Markov Chain. The high level stages of a BoX-MAC node are: backoff with initial contention window CW_{init}, backoff with congested contention window CW_{cong}, transmission and operating system delays. The counter process accounts for the number of time slots corresponding to each high level stage.

The states (j, k), $(2j + 4, k)$, and $(2j + 5, k)$, where $j \in \{0, 1, 2, 3\}$, represent one third of a BoX-MAC slot. A node entering state (j, k) transitions to state $(2j+4, k)$, and then to $(2j + 5, k)$ with probability 1 at the end of each time slot, thereby accounting for an entire BoX-MAC slot. We denote by W'_j the current contention window size, where $j \in \{0, 1\}$. When $j = 0$ the stage $s(t) \in \{0, 4, 5\}$ and the node is in backoff stage with a contention window size CW_{init} ($W'_0 = CW_{\text{init}}$). When $j = 1$ the stage $s(t) \in \{1, 6, 7\}$ and $W'_1 = CW_{\text{cong}}$. The backoff delay is represented by states

(j, k), where $j \in \{0, 1\}$ and $k \in \{-1, 0, 1, \ldots, W'_j - 1\}$. The transition probabilities are assumed to be independent. As described by the CSMA/CA mechanism, a device starts from state $(0, k)$, where k is a random number between 0 and $CW_{\text{init}} - 1$. In states $(j, 0)$ and $(j, -1)$, a channel assessment (CCA) is performed. We note that any channel sensing state, such as $(0, 0)$ or $(1, -1)$, is modeled by a process denoted as *SENS_B* and will be used in the channel model (Section III-D). If the channel is sensed busy (with probability α), the device transitions to state $(1, k)$ where k is a random number between 0 and $CW_{\text{cong}} - 1$. If the channel is sensed idle in both states $(j, 0)$ and $(j, -1)$, the packet is transmitted.

Transmission states are represented as TXB; specifically, $s(t) \in \{2, 8, 9\}$ and $c(t) \in \{0, 1, \ldots, L_{\text{TXB}} - 1\}$, where L_{TXB} is the duration of a BoX-MAC transmission, and a function of the packet size and transmission bandwidth. Before sending a packet, the device experiences delay from the operating system. This is represented as state OSB; specifically, $s(t) \in \{3, 10, 11\}$ and $c(t) \in \{0, 1, \ldots, L_{\text{OSB}} - 1\}$, where L_{OSB} is the operating system delay, obtained experimentally. For ease of understanding the equations that follow, we extend the notation W'_j such that it denotes L_{TXB} and L_{OSB}, i.e., $W'_0 = CW_{\text{init}}$, $W'_1 = CW_{\text{cong}}$, $W'_2 = L_{\text{TXB}}$ and $W'_3 = L_{\text{OSB}}$. Thus, the valid values for state j in (j, k) are $\{0, 1, 2, 3\}$.

The single step transition probabilities, as defined by the Markov Chain for BoX-

MAC, are:

$$\begin{aligned}
&Pr[2j+4,k|j,k]=1, && k=-1,0,...,W'_j-1\\
&Pr[2j+5,k|2j+4,k]=1, && k=-1,0,...,W'_j-1\\
&Pr[j,k-1|2j+5,k]=1, && j=0,1;k=1,...,W'_j-1\\
&Pr[j,-1|2j+5,0]=1-\alpha, && j=0,1\\
&Pr[1,k|2j+5,0]=\alpha/W'_1, && j=0,1;k=0,...,W'_1-1\\
&Pr[1,k|2j+5,-1]=\alpha/W'_1, && j=0,1;k=0,...,W'_1-1\\
&Pr[2,0|2j+5,-1]=1-\alpha, && j=0,1\\
&Pr[j,k|2j+5,k-1]=1, && j=2,3;k=1,...,W'_j\\
&Pr[3,0|9,L_{TXB}-1]=1 && \\
&Pr[0,k|11,L_{OSB}-1]=1/W'_0, && k=0,...,W'_0
\end{aligned}$$

where $j \in \{0,1,2,3\}$ if not explicitly specified. The coexistence of BoX-MAC with other types of devices can be modeled by computing α, i.e., the probability that the channel is busy during a given time slot. The probabilities of sensing a busy channel during the first and second CCA are typically correlated [44]. However, we approximate them as independent events. We validate experimentally that this inaccuracy is tolerable.

3.2.3 Steady State Analysis

First we perform steady state analysis in order to obtain the stationary distributions for both BoX-MAC and 802.11 Markov Chains, and their normalization conditions. These are then used to obtain the transmission probabilities and the conditional collision probability *under coexistence.*

Table 3.1: Expressions for the limiting distribution of 802.11 and BoX-MAC Markov Chains

BoX-MAC $(k = 0, ..., W'_j - 1)$	**802.11 DCF** $(k = 0, ..., W_j - 1)$
$b'_{j,k} = \frac{W'_j - k}{W'_j} b'_{j,0},\ j = 0, 1$	$b_{j,0} = b_{0,0} P^j_{coll},\ j = 0, ..., (m-1)$
$b'_{1,0} = \frac{b'_{0,0} x}{1-x},\ j = 0, 1$	$b_{m,0} = b_{0,0} \frac{P^m_{coll}}{1-P_{coll}}$
$b'_{j,-1} = (1-\alpha) b'_{j,0},\ j = 0, 1$	$b_{j,k} = \frac{1}{1-P_f} \frac{W_j - k}{W_j} b_{j,0},\ j = 0, ..., m$
$b'_{j,k} = (1-\alpha)^2 (b'_{0,0} + b'_{1,0}),\ j = 2, 3$	$b_{-3-j,k} = b_{j,0} P_{coll},\ j = 0, ..., m$
$b'_{2j+4,k} = b'_{j,k},\ j = 0, 1, 2, 3$	$b_{-1,k} = b_{0,0}$
$b'_{2j+5,k} = b'_{j,k},\ j = 0, 1, 2, 3$	$b_{-2,k} = b_{0,0}$

Let $b'_{j,k} = lim_{t \to \infty}\ Pr\{s(t) = j, c(t) = k\}$ be the stationary distribution of the BoX-MAC Markov Chain and $b_{j,k} = lim_{t \to \infty}\ Pr\{b(t) = j, r(t) = k\}$ be the stationary distribution of the 802.11 Markov Chain. Expressions for all the terms in the limiting distribution of the Markov Chains are presented in Table 3.1, where $x = (\alpha + (1 - \alpha)\alpha)$, $W'_0 = CW_{\text{init}}$ and $W'_1 = CW_{\text{cong}}$.

The normalization condition is used for obtaining $b'_{0,0}$ and $b_{0,0}$ from the Markov Chains. The following equations are for BoX-MAC:

$$1 = \sum_{j=0}^{1} \sum_{k=0}^{W'_j - 1} 3b'_{j,k} + \sum_{k=0}^{L_{\text{TXB}}-1} 3b'_{2,k} + \sum_{k=0}^{L_{\text{OSB}}-1} 3b'_{3,k}$$

$$b'_{0,0} = \frac{1}{\left(3\frac{W'_0+1}{2} + 3\frac{(W'_1+1)x}{2(1-x)} + 3\frac{1-\alpha}{1-x} + 3L_{\text{TXB}} + 3L_{\text{OSB}}\right)} \tag{3.1}$$

where $W'_0 = CW_{\text{init}}$ and $W'_1 = CW_{\text{cong}}$.

The following equations are for 802.11 DCF:

$$1 = \sum_{j=0}^{m} \sum_{k=0}^{W_j-1} b_{j,k} + \sum_{j=0}^{m} \sum_{k=0}^{L_{\mathrm{CTW}}-1} b_{-3-j,k} + \sum_{k=0}^{L_{\mathrm{STW}}-1} b_{-1,k} + \sum_{k=0}^{L_{\mathrm{OSW}}-1} b_{-2,k}$$

$$b_{0,0} = \frac{1}{\frac{CW_{\min}(1-(2P_{coll})^m)}{2(1-2P_{coll})(1-Pf)} + \frac{CW_{\min}((2P_{coll})^m)+1}{2(1-P_{coll})(1-Pf)} + \frac{L_{\mathrm{CTW}}P_{coll}}{1-P_{coll}} + L_{\mathrm{STW}} + L_{\mathrm{OSW}}} \tag{3.2}$$

Using $b'_{0,0}$ and $b_{0,0}$, we can simply derive the probabilities that a node is transmitting, i.e., τ_W for WiFi and τ_B and BoX-MAC, as follows:

$$\tau_B = 3L_{\mathrm{TXB}} \sum_{j=0}^{1} (1-\alpha) b'_{j,-1} = 3L_{\mathrm{TXB}} b'_{0,0} \tag{3.3}$$

$$\tau_W = L_{\mathrm{STW}} \sum_{j=0}^{m} b_{j,0} = \frac{L_{\mathrm{STW}} b_{0,0}}{1-P_{coll}} \tag{3.4}$$

Knowing τ_B and τ_W, we can calculate the conditional collision probability P_{coll}. For a collision to occur, besides the WiFi device transmitting, there is at least one other device transmitting:

$$P_{coll} = 1 - (1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B} \tag{3.5}$$

where N_W and N_B are the number of WiFi and BoX-MAC nodes, respectively.

We remark that the channel busy probabilities, i.e., α and P_f for BoX-MAC and WiFi respectively, have not been computed yet. The challenge in computing them comes from the fact that the two protocols are extremely different. The main observation that we make is that the channel witnesses all activities of nodes. Consequently, our main idea is to develop a Markov Chain based channel model for symmetric wireless coexistence, the first of its kind.

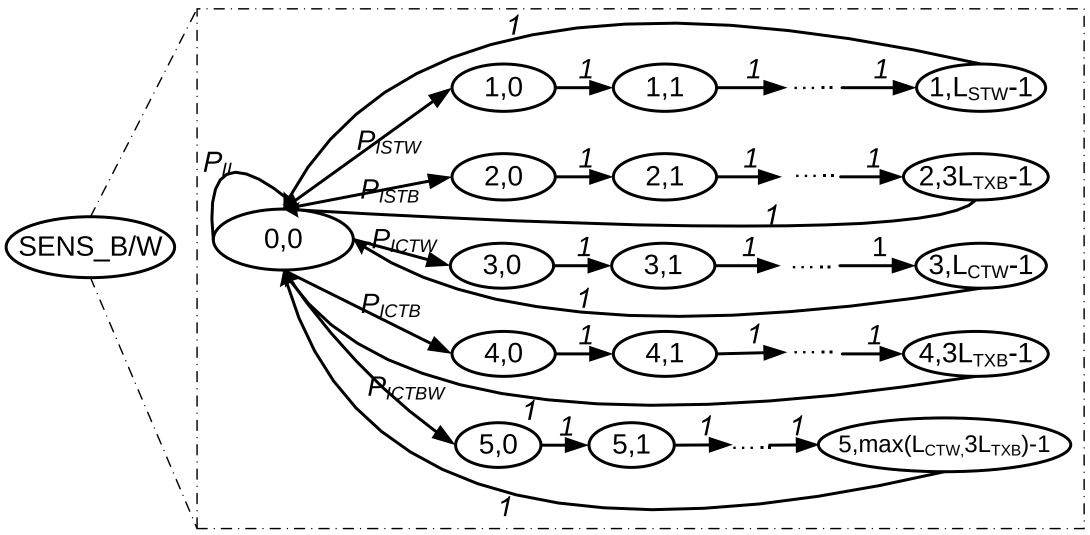


Figure 3.5: Markov Chain describing the channel.

3.2.4 Markov Chain Based Channel Model

We now present the Markov Chain based channel model for coexistence. With this model we aim to compute the steady state transition probabilities and stationary distribution of the channel states. Without a Markov Chain model, α and P_f (as computed from the 802.11 and BoX-MAC Markov Chains) can not be proven to reflect the steady state transition probabilities of the channel. Thus, a Markov Chain model is expected to be more accurate. Evidence was given in [74], where a channel model improved accuracy in P_f computation for 802.11 wireless networks. Additionally, our Markov Chain based channel model simplifies the analysis for α and P_f computation, which we expect to be extremely beneficial when heterogeneous coexistent networks will be considered (i.e., different nodes have different contention windows).

We note that α and P_f are conditional probabilities that depend on nodes "sensing" the channel. From the perspective of a sensing node, which we call "Tagged

Node", the state of the channel (e.g., busy with successful transmission, busy with collision, or idle) and the states of other nodes are interrelated. For a Tagged Node (note: the tagged node simply senses the channel), the channel is busy when any other node transmits; a transmission is successful if only one other node transmits; in all other cases, the channel is idle. Since in our coexistence problem we have devices of different types, the "Tagged Node" can be either a BoX-MAC node or an 802.11 node.

Each channel sensing process, i.e., SENS_B and SENS_W for BoX-MAC and WiFi respectively, is modeled as a Markov Chain as shown in Figure 3.5. Let $v(t)$ and $x(t)$ be the two stochastic processes representing the state of the channel and a counter process, respectively. $v(t) = 0$ represents the state where the channel is idle; $v(t) = 1$ represents the state where the channel is busy with a successful 802.11 transmission; $v(t) = 2$ represents the state where the channel is busy with a successful BoX-MAC transmission; $v(t) = 3$ represents the channel busy with two or more 802.11 nodes transmitting, which leads to a collision; $v(t) = 4$ represents the channel busy with two or more BoX-MAC nodes transmissions, which leads to a collision; and $v(t) = 5$ represents the channel busy with at least one 802.11 node and at least one BoX-MAC transmitting, which leads to a collision. We divide the busy states of the channel in this manner because the duration of each transmission is different (depending on the types of transmitters, i.e., BoX-MAC or WiFi), but deterministic (note that we assume that devices of the same type transmit packets of same length). From each busy state, the channel returns to an idle state after the transmission. We note that if the transmission was not deterministic (i.e., packet length of same type devices can be different), the model would have a single state for the busy channel, that returns to the idle state *with some probability.* This would make the analysis much more difficult and we leave it for future work. The single step transition probabilities are as follows:

$$
\begin{aligned}
&Pr[0,0|0,0] = P_{\mathrm{II}}, && \\
&Pr[1,0|0,0] = P_{\mathrm{ISTW}}, && \\
&Pr[2,0|0,0] = P_{\mathrm{ITSB}}, && \\
&Pr[3,0|0,0] = P_{\mathrm{ICTW}}, && \\
&Pr[4,0|0,0] = P_{\mathrm{ICTB}}, && \\
&Pr[5,0|0,0] = P_{\mathrm{ICTBW}}, && \\
&Pr[i,j|i,j-1] = 1, && \text{for } i = 2,\ldots,5, j = 1,\ldots,L_i - 1 \\
&Pr[0,0|i,L_i - 1] = 1, && \text{for } i = 1,\ldots,5
\end{aligned}
$$

where $L_1 = L_{\mathrm{STW}}$, $L_2 = 3L_{\mathrm{TXB}}$, $L_3 = L_{\mathrm{CTW}}$, $L_4 = 3L_{\mathrm{TXB}}$, and $L_5 = max(L_{\mathrm{CTW}}, 3L_{\mathrm{TXB}})$.

As mentioned, when considering the channel state, there must be a reference node, i.e., the node that is sensing the channel or the *Tagged Node*. Since there are two different types of Tagged Nodes (802.11 and BoX-MAC), two different analyses of the channel are needed.

First, if a BoX-MAC node is the Tagged Node, the probability that it finds the channel idle, namely P'_{II}, is the probability that none of the nodes other than the BoX-MAC node is transmitting. From the idle state, the probability that the channel will contain a successful 802.11 transmission in the next step P'_{ISTW} is the probability that one of the WiFi nodes is transmitting while no other BoX-MAC node is transmitting. Similarly, the probability that the channel goes from an idle state to a state of successful BoX-MAC transmission P'_{ISTB} is the probability that exactly one of the remaining BoX-MAC nodes is transmitting. Collisions are predicted based on the probability that two or more nodes will enter a transmission state simultaneously

(P'_{ICTW}, P'_{ICTB} and P'_{ICTBW} for WiFi devices, for BoX-MAC devices, and for WiFi and BoX-MAC devices, respectively). A node l has a probability of transmission τ_l that is determined by its contention window size. The resulting single step transition probabilities of the channel Markov Chain are:

$$
\begin{aligned}
P'_{\text{II}} &= (1-\tau_W)^{N_W}(1-\tau_B)^{N_B-1} \\
P'_{\text{ISTW}} &= N_W\tau_W(1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B-1} && (3.6) \\
P'_{\text{ISTB}} &= N_B\tau_B(1-\tau_B)^{N_B-2}(1-\tau_W)^{N_W} && (3.7) \\
P'_{\text{ICTW}} &= (1-\tau_B)^{N_B-1}(1-N_W\tau_W(1-\tau_W)^{N_W-1}-(1-\tau_W)^{N_W}) && (3.8) \\
P'_{\text{ICTB}} &= (1-\tau_W)^{N_W}(1-N_B\tau_B(1-\tau_B)^{N_B-2}-(1-\tau_B)^{N_B-1}) && (3.9) \\
P'_{\text{ICTBW}} &= (1-(1-\tau_W)^{N_W})(1-(1-\tau_B)^{N_B-1}) && (3.10)
\end{aligned}
$$

If the Tagged Node is an 802.11 node: from the idle state, the probability that the channel will contain a successful WiFi transmission in the next step is the probability that one of the remaining WiFi nodes is transmitting while none of BoX-MAC nodes is transmitting. To keep the text concise, we omit the descriptions of other probabilities. The resulting single step transition probabilities are as follows:

$$
\begin{aligned}
P_{\text{II}} &= (1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B} \\
P_{\text{ISTW}} &= N_W\tau_W(1-\tau_W)^{N_W-2}(1-\tau_B)^{N_B} && (3.11) \\
P_{\text{ISTB}} &= N_B\tau_B(1-\tau_B)^{N_B-1}(1-\tau_W)^{N_W-1} && (3.12) \\
P_{\text{ICTW}} &= (1-\tau_B)^{N_B}(1-N_W\tau_W(1-\tau_W)^{N_W-2}-(1-\tau_W)^{N_W-1}) && (3.13) \\
P_{\text{ICTB}} &= (1-\tau_W)^{N_W-1}(1-N_B\tau_B(1-\tau_B)^{N_B-1}-(1-\tau_B)^{N_B}) && (3.14) \\
P_{\text{ICTBW}} &= (1-(1-\tau_W)^{N_W-1})(1-(1-\tau_B)^{N_B}) && (3.15)
\end{aligned}
$$

Given the single step transition probabilities, we derive the stationary distributions of this Markov Chain as follows (for the Tagged Node being BoX-MAC):

$$
\begin{aligned}
&v'_{i,j} = v'_{i,0} \text{ for } i \in \{1, 2, \ldots, 5\}, \\
&v'_{1,0} = P'_{\text{ISTW}} v'_{0,0}, \\
&v'_{2,0} = P'_{\text{ISTB}} v_{0,0}, \\
&v'_{3,0} = P'_{\text{ICTW}} v_{0,0}, \\
&v'_{4,0} = P'_{\text{ICTB}} v_{0,0}, \\
&v'_{5,0} = P'_{\text{ICTBW}} v_{0,0}
\end{aligned}
$$

where $v'_{0,0}$ is steady probability that the tagged BoX-MAC node senses the channel idle.

The normalization condition for the channel model yields the following:

$$
v'_{0,0} + \sum_{j=0}^{L_{\text{STW}}-1} v'_{1,j} + \sum_{j=0}^{3L_{\text{TXB}}-1} v'_{2,j} + \sum_{j=0}^{L_{\text{CTW}}-1} v'_{3,j} + \sum_{j=0}^{3L_{\text{TXB}}-1} v'_{4,j} + \sum_{j=0}^{max(L_{\text{CTW}},3L_{\text{TXB}})-1} v'_{5,j} = 1
$$

From the above equation, we obtain:

$$
\begin{aligned}
v'_{0,0} = 1 \Big/ \Big(& 1 + L_{\text{STW}} P'_{\text{ISTW}} + 3L_{\text{TXB}} P'_{\text{ISTB}} + L_{\text{CTW}} P'_{\text{ICTW}} + \\
& 3L_{\text{TXB}} P'_{\text{ICTB}} + max(L_{\text{CTW}}, 3L_{\text{TXB}}) P'_{\text{ICTBW}} \Big) \qquad (3.16)
\end{aligned}
$$

where, from here onwards, the variables in the form of L_{xxx} represent the corresponding duration when event xxx happens.

Using a similar derivation, for an 802.11 node being the Tagged Node, we obtain:

$$v_{0,0} = 1 \Big/ \Big(1 + L_{\mathrm{STW}} P_{\mathrm{ISTW}} + 3L_{\mathrm{TXB}} P_{\mathrm{ISTB}} + L_{\mathrm{CTW}} P_{\mathrm{ICTW}} + \\ 3L_{\mathrm{TXB}} P_{\mathrm{ICTB}} + max(L_{\mathrm{CTW}}, 3L_{\mathrm{TXB}}) P_{\mathrm{ICTBW}}\Big) \tag{3.17}$$

where $v_{0,0}$ is steady probability that the Tagged Node senses the channel idle.

We now derive α and P_f based on the channel model. α and P_f are the conditional probabilities that the channel has a transmission. Therefore, each is the sum of probabilities that the channel is in any state other than the idle state. Consider α as observed by a BoX-MAC node l sensing the channel when the backoff counter is 0. Now the BoX-MAC node is the Tagged Node, i.e., the channel as observed by the Tagged Node contains all N_W 802.11 nodes and $N_B \setminus \{l\}$ BoX-MAC nodes. Similarly, P_f represents the probability that the channel is not in an idle state for an 802.11 Tagged Node s, i.e., $N_W \setminus \{s\}$ WiFi nodes and N_B BoX-MAC nodes. Therefore, α and P_f can be expressed as follows:

$$\alpha = 1 - v'_{0,0} \tag{3.18}$$

$$P_f = 1 - v_{0,0} \tag{3.19}$$

3.3 Performance Predication for SAT

3.3.1 Throughput Analysis

With the help of stationary distribution and normalization conditions, we have successfully derived the variables that reflect the state of the channel, namely α and P_f. These can be determined by numerically solving a set of non-linear equations. Several performance metrics can be derived from these probabilities. In this dissertation, we are interested in the saturation aggregate throughput.

The normalized saturation throughput of BoX-MAC is the fraction of time that the channel is busy with a successful BoX-MAC transmission, given by:

$$\begin{aligned} S_{\text{BoX-MAC}} = 3Lp_{\text{TXB}}P_{\text{STB}}\Big/\Big(&P_{\text{I}} + 3L_{\text{TXB}}P_{\text{STB}} + 3L_{\text{TXB}}P_{\text{CTB}} + L_{\text{STW}}P_{\text{STW}} + \\ &L_{\text{CTW}}P_{\text{CTW}} + 3L_{\text{TXB}}P_{\text{CTBW}}\Big) \end{aligned} \tag{3.20}$$

where Lp_{TXB} is actual packet size and P_{I} is the probability that the channel is in idle state. P_{STB} and P_{STW} are probabilities that a successful transmission occurs, for BoX-MAC and WiFi respectively. Likewise, P_{CTB} and P_{CTW} are collision probabilities among pure BoX-MAC and 802.11 respectively, while P_{CTBW} indicates the probability of collision caused by simultaneous transmission of BoX-MAC and 802.11. The variables in the form of L_{xxx} represent the corresponding duration when event xxx happens. The expression of the state probabilities mentioned above are as follows.

$$P_{\text{I}} = (1-\tau_W)^{N_W}(1-\tau_B)^{N_B}$$

$$P_{\text{STW}} = N_W\tau_W(1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B} \tag{3.21}$$

$$P_{\text{STB}} = N_B\tau_B(1-\tau_B)^{N_B-1}(1-\tau_W)^{N_W} \tag{3.22}$$

$$P_{\text{CTW}} = (1-\tau_B)^{N_B}(1-N_W\tau_W(1-\tau_W)^{N_W-1}-(1-\tau_W)^{N_W}) \tag{3.23}$$

$$P_{\text{CTB}} = (1-\tau_W)^{N_W}(1-N_B\tau_B(1-\tau_B)^{N_B-1}-(1-\tau_B)^{N_B}) \tag{3.24}$$

$$P_{\text{CTBW}} = (1-(1-\tau_W)^{N_W})(1-(1-\tau_B)^{N_B}) \tag{3.25}$$

Notably, these expressions are quite similar to those for the transitional probabilities P_{II}, P_{STW}, etc. as shown in Equations (3.6-3.15); however, the equations here are describing the channel state probabilities while the earlier ones were for the channel model (which depend on an observation node). The normalized saturation throughput of 802.11 is the fraction of time that the channel is busy with a successful 802.11 transmission given by:

$$S_{802.11} = Lp_{\mathrm{STW}} P_{\mathrm{STW}} \Big/ \Big(P_{\mathrm{I}} + 3L_{\mathrm{TXB}} P_{\mathrm{STB}} + 3L_{\mathrm{TXB}} P_{\mathrm{CTB}} + L_{\mathrm{STW}} P_{\mathrm{STW}} + L_{\mathrm{CTW}} P_{\mathrm{CTW}} + 3L_{\mathrm{TXB}} P_{\mathrm{CTBW}} \Big) \tag{3.26}$$

where Lp_{STW} is the actual packet size.

We derive expressions for τ_W and τ_B from their respective Markov Chain models as functions of the contention window sizes and channel states, i.e., $CW_{\min}$, $CW_{\max}$ and P_f for 802.11 nodes, and CW_{init}, CW_{cong} and α for BoX-MAC nodes as:

$$\tau_W = \frac{L_{\mathrm{STW}}}{(1-P_{coll})} \cdot \frac{1}{\frac{CW_{\min}(1-(2P_{coll})^m)}{2(1-2P_{coll})(1-Pf)} + \frac{CW_{\min}((2P_{coll})^m)+1}{2(1-P_{coll})(1-Pf)} + \frac{L_{\mathrm{CTW}} P_{coll}}{1-P_{coll}} + L_{\mathrm{STW}} + L_{\mathrm{OSW}}} \tag{3.27}$$

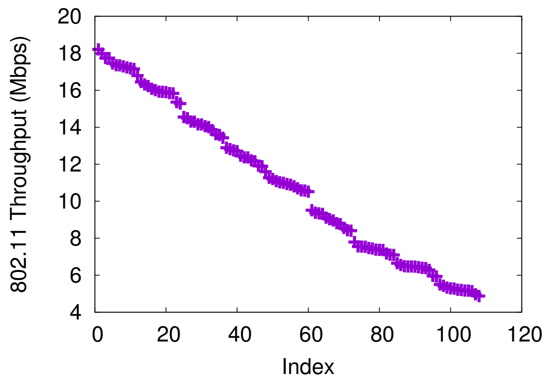
$$\tau_B = \frac{L_{\mathrm{TXB}}}{\frac{W_0'+1}{2} + \frac{(W_1'+1)x}{2(1-x)} + \frac{1-\alpha}{1-x} + L_{\mathrm{TXB}} + L_{\mathrm{OSB}}} \tag{3.28}$$

These can be used to tune protocol parameters based on expected τ_l for a node l.

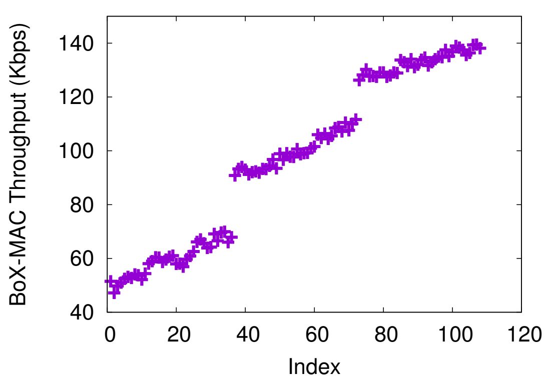
3.4 Performance Tuning for SAT

3.4.1 Motivation

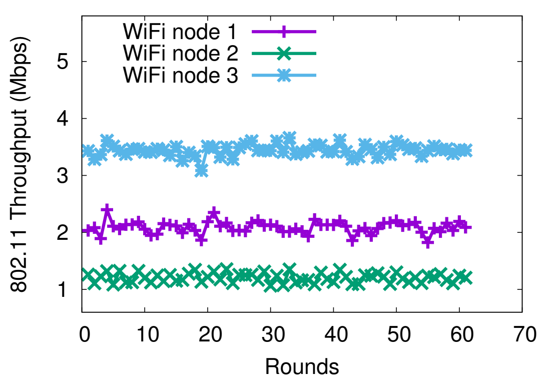
Since contention window size (CW) is a well-known critical parameter that affects throughput and fairness [82] [83] [84] [85], as studied in typical wireless networks employing single MAC protocols, we were curious to investigate how CW sizes of different, coexisting MAC protocols affect throughput and fairness of networks with



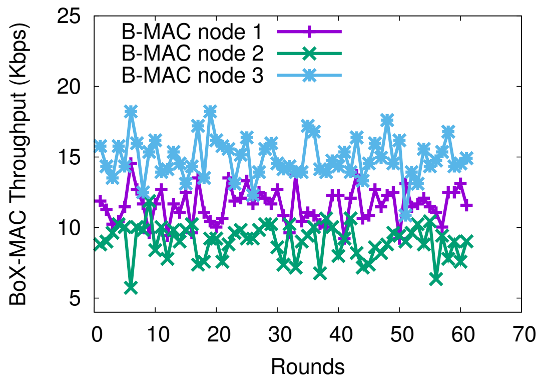
(a)



(b)



(c)



(d)

Figure 3.6: Demonstrating the (a-b) priority problem and (c-d) fairness problem in coexisting WiFi and BoX-MAC networks.

symmetric coexistence.

For this, we used the Monte Carlo simulator for SAT proposed in Section 3.5 and simulated 5 WiFi and 10 BoX-MAC devices, all within one hop, in two sets of experiments. In the first set of experiments we ran multiple simulations, each with a different combination of contention window sizes, CW_W for WiFi and CW_B for BoX-MAC. Importantly, in each simulation, the devices of the same type employ the same contention window size. In the second set of experiments we ran multiple simulations and allowed devices of the same type to choose their contention window size (CW_W for WiFi and CW_B for BoX-MAC) freely. The traffic was saturated in both sets of experiments.

The results for the first sets of experiments are presented in Figures 3.6(a)-(b). As shown, the throughput of WiFi and BoX-MAC are strictly inversely proportional. This motivates us to provide prioritization where a network administrator can decide operating points for different type of devices such that they achieve different throughput. The results for the second set of experiments are presented in Figures 3.6(c)-(d) which show that if devices of the same type are allowed to use different CW, they may experience different throughput, thus experiencing the *fairness problem*. When nodes arbitrarily set their CWs to benefit themselves, channel sharing is unfair.

The state of art research only focused on the effects of CW size on the performance of a single MAC protocol. In a symmetric coexistence scenario, different protocols (WiFi and BoX-MAC) have different CW sizes. Intuitively, CW size will affect the throughput and fairness of both protocols, but the extent to which they will be affected, is not known.

In this section we present Contention Window (CW) tuning mechanisms intended to control priority as well as fairness by changing the CW size on individual nodes. CW is critical for all contention based protocols because it directly controls the transmission probability, thus impacting the throughput [82] [83] [84] [85]. CW tuning is the main objective of most state of the art optimization protocols, where an accurate model plays the key role. Since we have proposed the first Markov Chain based model for coexistence, the capability to tune the CW is important. As shown by literature and confirmed by our experimental results in Section 3.5, the congested CW size of BoX-MAC (CW_{cong}) and the minimum CW size of 802.11 ($CW_{\min}$) have a significant impact on the throughput achieved by individual nodes in a coexisting network, while the initial CW size of BoX-MAC (CW_{init}) and the maximal CW size of 802.11 ($CW_{\max}$) do not. Thus for simplicity, we only consider tuning CW_{cong} and $CW_{\min}$, and treat CW_{init} and $CW_{\max}$ as fixed.

The Markov Chain model presented in the previous section provides a method to estimate the saturation throughput of coexisting networks of 802.11 and BoX-MAC nodes. The model also provides a mechanism to estimate the probability of transmission τ_B or τ_W of a node, given the CW sizes, the packet size, and an observed status of the channel, i.e., α as observed by a BoX-MAC node, and P_{coll} and P_f as observed by a 802.11 node.

3.4.2 Centralized CW Tuning Method for priority

Priority is extremely important when heterogeneous protocols compete for the same medium. For instance, low bit rate protocols (such as ZigBee and BoX-MAC) cannot easily capture the channel due to their non-aggressive nature. On the other hand, these protocols can severely degrade high bit rate protocols (like 802.11) when they capture the channel (because of the former's low transmission rate). Our CW tuning method helps mitigate these effects.

It is problematic to claim that the priority is the ratio of the two absolute throughputs because of the order of magnitude difference in their PHY bit rates. Therefore, we define the priority metric under coexistence as the ratio of the successful transmission probabilities of 802.11 and BoX-MAC [91]. The probability that a transmission is successful is expressed as:

$$SU_B = \tau_B(1-\tau_B)^{N_B-1}(1-\tau_W)^{N_W}$$
$$SU_W = \tau_W(1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B}$$

Hence the priority metric, denoted by ϕ, can be written as:

$$priority = \phi = \frac{SU_B}{SU_W} = \frac{\tau_B(1-\tau_B)^{N_B-1}(1-\tau_W)^{N_W}}{\tau_W(1-\tau_W)^{N_W-1}(1-\tau_B)^{N_B}} = \frac{\tau_B(1-\tau_W)}{\tau_W(1-\tau_B)}$$

We then obtain an expression of τ_W in terms of τ_B and ϕ:

$$\tau_W = \frac{\tau_B}{\frac{1-\tau_B}{\phi} + \tau_B} \tag{3.29}$$

There is a multitude of τ_B and τ_W combinations that satisfy ϕ. In this dissertation, we aim to maximize the total throughput of a given network. Since the aggregate throughput, i.e $S_{802.11}$ and $S_{\text{BoX-MAC}}$, depends on τ_B and τ_W, we maximize the total throughput as $S_{\text{total}} = S_{\text{BoX-MAC}} + S_{802.11}$, while maintaining the priority requirement.

Based on Equations (3.20) and (3.26), we obtain the expression for $S_{\text{BoX-MAC}} + S_{802.11}$ as:

$$\begin{aligned} S_{\text{total}} =& S_{\text{BoX-MAC}} + S_{802.11} \\ =& \Big(3Lp_{\text{TXB}}P_{\text{STB}} + Lp_{\text{STW}}P_{\text{STW}}\Big)\Big/\Big(P_{\text{I}} + 3L_{\text{TXB}}P_{\text{STB}} + \\ & 3L_{\text{TXB}}P_{\text{CTB}} + L_{\text{STW}}P_{\text{STW}} + L_{\text{CTW}}P_{\text{CTW}} + 3L_{\text{TXB}}P_{\text{CTBW}}\Big) \end{aligned} \tag{3.30}$$

All P_{xxx} above are functions of τ_B and τ_W. Using Equation (3.29), we substitute all τ_W with τ_B, and express S_{total} as a function of τ_B only. Note that the priority ratio ϕ is user specified. In order to get the maximum value for S_{total}, we take the derivative of Equation (3.30) w.r.t τ_B, and set it to 0, i.e., $S'_{\text{total}} = 0$. Because of the computational complexity of solving this equation, we use an approximation method.

Under the condition $\tau_B \ll 1$, the following approximation holds:

$$(1-\tau_B)^n \approx 1 - n\tau_B + \frac{n(n-1)}{2}\tau_B^2 \tag{3.31}$$

We can thus significantly simplify the expression for S_{total}, reducing the complexity of solving Equation (3.30) and making it feasible to run on COTS computer hardware, especially when the number of nodes is large. By numerically solving Equation (3.31), we obtain the value of τ_B and subsequently, τ_W through Equation (3.29). A few more unknowns need to be computed before CW sizes can be obtained. α, P_{coll} and P_f are computed using Equations (3.5) and (3.6-3.19). Finally, by substituting τ_B, τ_W, P_{coll}, α and P_f into Equations (3.27-3.28) and treating all other variables as constants, we obtain the contention window sizes for BoX-MAC and 802.11.

3.4.3 Distributed CW Tuning Method for Fairness

As described before, $CW_{\min}$ and CW_{cong} are critical for coexisting wireless networks. Nodes with different $CW_{\min}$ and CW_{cong} achieve different throughput, thereby leading to unfair utilization of the bandwidth. We call this the *fairness problem.* When the nodes of a network are allowed to tune their parameters themselves, it becomes important to show the existence of an equilibrium point from which no node has the incentive to modify its parameters [86].

To solve the fairness problem, we propose a game theoretic approach similar to the one presented by Jin and Kesidis [86]. We define a concave maximization function of τ_i (transmission probability of node i) for each node as $\max(U_i - D_i)$, where U_i is the utility, and D_i is the disutility, or cost experienced by the node n_i for a given τ_i

and α.

$$U_i - D_i = \left(\frac{2(N_W + N_B - 1)log(1+\tau_i)}{L} \right) - \frac{\tau_i \alpha}{L(1+\alpha)}$$

where $L = L_{\mathrm{STW}}$ is for 802.11 nodes and L_{TXB} is for BoX-MAC nodes. Here, $\tau_i \frac{\alpha}{1+\alpha}$ is an approximation for the expected collision probability.

In this CW tuning method, each node always tries to maximize $U_i - D_i$. This can be thought of as a selfish behavior. The game model is such that each node decides $\tau = [\tau_1, \tau_2, \dots \tau_{N_W}, \tau_{N_W+1} \dots \tau_{N_W+N_B}]'$ to maximize $f(\tau) = [U_1 - D_1, U_2 - D_2, \dots U_{N_W+N_B} - D_{N_w+N_B}]'$. For a node n_i, $U_i - D_i = (\frac{2(N_W+N_B-1)log(1+\tau_i)}{L}) - \frac{\tau_i \alpha_i}{L(1+\alpha_i)}$. The gradient, $\nabla f(\tau)_i = (\frac{2(N_W+N_B-1)}{L(1+\tau_i)} - \frac{\alpha_i}{1+\alpha_i})$. To prove that $f(\tau)$ is concave, we show that the Jacobian of $\nabla f(\tau)$, $F(\tau)$ is negative definite, i.e., has only negative eigenvalues.

A diagonal element of F, $F_{ii} = \frac{-2(N_W+N+B-1)}{L(1+\tau_i)^2}$, and non-diagonal element $F_{ij} = \frac{-1}{L} \frac{(1-\alpha_j)^2}{(1+\alpha_j)^2} \Pi_{k \neq i \neq j}(1-\tau_k)$. Since $\frac{(1-\alpha_j)^2}{(1+\alpha_j)^2} \leq 1$ and $\Pi_{k \neq i \neq j}(1-\tau_k) < 1$, $|F_{ij}| < \frac{1}{L}$. Therefore, $\sum_{j \neq i} |F_{ij}| < \frac{N_W+N_B-1}{L}$. Since τ is the probability that a node attempts to begin a transmission, it is always less than 0.414, $(1+\tau_i)^2 < 2$, i.e., $\frac{1}{(1+\tau_i)^2} > \frac{1}{2}$, which proves that $|F_{ii}| < \sum_{j \neq i} |F_{ij}|$ for all i.

Since F_{ii} is negative, and by the Gerschgorin circle theorem, all eigenvalues lie in circles with center at F_{ii} and radius $\sum_{j \neq i} |F_{ij}|$, F is negative definite and this proves that the game has a concave objective. As proven by Rosen [92], an n-player non-cooperative game where each player is selfish and maximizes a concave objective reaches Nash Equilibrium. Hence, there exists a Nash equilibrium, and therefore, our proposed mechanism will reach a stable state where no node benefits from changing their parameters, even though each node behaves selfishly (given that the other nodes do not change their parameters). We also observe that at the Nash equilibrium, nodes of the same kind choose the same parameters.

Rosen also proposed a gradient projection method to iteratively reach Nash Equilibrium. We use this result to define a gradient projection iteration to reach Nash Equilibrium:

$$\tau_{k+1} = \tau_k + \frac{\epsilon}{L}\left(\frac{2(N_W + N_B - 1)}{1+\tau_k} - \frac{\alpha_k}{1+\alpha_k}\right)$$

where ϵ is the step size of the gradient projection method, τ_k is the τ observed by the node in the k^{th} step and α_k is the α observed by the node in the k^{th} step.

As we know from the Markov Chain model, τ is a function of the contention window sizes. We can therefore solve for CW_{min} for 802.11 and CW_{cong} for BoX-MAC at the $(k+1)^{th}$ step from τ_{k+1}. For a sufficiently large k, the system reaches a stable point, i.e. Nash equilibrium, as proven by Rosen. Since the CW sizes have to be chosen from a set (combinatorial), the CW sizes that result in a transmission probability closest to τ_{k+1} are chosen at each step. We have shown that the fairness problem can be solved using this method, by proving that the individual throughput of nodes of the same type are equal at Nash Equilibrium.

3.5 Monte Carlo based Simulator for SAT

As described in Section I, simulator is a tool which is able to predicate the network performance accurately. However, there is no wireless coexistence module that is available in the state of art network simulators (e.g., ns-2, OPNET, QualNet, etc.). Therefore, in this section, we develop a Monte Carlo based simulator for symmetric coexistence of non-sleeping WiFi and ZigBee with saturated traffic. To verify the accuracy of the proposed simulator, hardware based experiments are performed to obtain throughput, which is then used to compare against the result from the simulator. It is worth noting that, since doing hardware experiment is a very tedious

Table 3.2: Parameters and Variables used in the Monte Carlo simulator

Notation	**Description**	**Value**
T	# of time slots for simulator	10,000,000
$channel$	channel state	{IDLE, BUSY}
$N = N_B + N_W$	# of all devices	[5, 50]
$nbColli[1:N]$	total # of collisions	sim variable
$DIFScnt[1:N]$	802.11 DIFS counter	sim variable
$CW[1:N]$	contention window size	sim variable
$delay[1:N]$	backoff counter	sim variable
$state[1:N]$	FSM state (802.11)	{DIFS-T,CSMA-T TRDEL-T, OS-T}
$nbTrans[1:N]$	total # of transmissions	sim variable
$busyFor[1:N]$	delay counter for transmission	sim variable
$softdelay[1:N]$	delay counter of OS delay	sim variable
$nbCCA[1:N]$	total # of CCAs (BoX-MAC)	sim variable
$CS[1:N]$	CCA stage (BoX-MAC)	sim variable

task, only 2 pairs of WiFi and 4 pairs of ZigBee devices are deployed here.

3.5.1 Simulator Design

The pseudocode for our Monte Carlo simulator is presented in Algorithms 1, 2 and 3. All variables used in the simulator, *italicized* in the pseudocode, are listed in Table 3.2.

Similar to the approach in [44], our Monte Carlo simulator is based on time slots. Both MAC protocols (in our case 802.11 DCF and BoX-MAC) have the concept of time slot, a time interval when a specific device behaves consistently; both MAC protocols randomly choose the sizes of their backoff window from a uniform distribution. This works well when we repeatedly sample for a sufficiently large number of time slots randomly (as shown in Table 3.2, $T = 10^7$ samples).

Algorithm 1 is the main function of the simulator. As shown, the simulation is for T time slots. In each time slot, the code first checks the state of the channel (i.e.

Algorithm 1 Monte Carlo Simulator for SAT

```
 1: for $T$ slots do
 2:     check the channel state, and set $channel$.
 3:     for $N$ nodes do
 4:         if node i is WiFi then
 5:             invoke 802.11 DCF($i$) (Algorithm 2)
 6:         else
 7:             if T mod 3 = 0 then
 8:                 invoke BoX-MAC($i$) (Algorithm 3)
 9:             end if
10:         end if
11:     end for
12:     if more than one simultaneous transmission then
13:         increase $nbColli$
14:     end if
15: end for
```

whether any node is transmitting), then executes the MAC protocol for each of the N nodes (note: nodes are either 802.11 or BoX-MAC devices). Since a node may either be an 802.11 or a BoX-MAC node, we have two algorithms: one for 802.11 DCF (Line 5) and the other for BoX-MAC (Line 8). Since the time slot for BoX-MAC is roughly three times that of 802.11 DCF, we invoke the procedure for 802.11 three times more frequently than for BoX-MAC (Line 7 in Algorithm 1). At the end of a slot, Algorithm 1 checks for collision (Line 10).

Algorithm 2 describes the Monte Carlo method for 802.11 DCF for a node i. The simulator maintains a Finite State Machine (FSM) to control the code execution. A node i that has data to send initializes its state to DIFS-T (not shown). Before entering the random backoff (CSMA-T state, in Line 8), a node must sense the channel idle for DIFS time slots. If the channel is sensed busy in this interval, $DIFScnt$ is set to 0 (Line 10), and the node remains in DIFS-T state. After the channel is sensed idle for DIFS time slots (Lines 4), the node checks if the backoff

Algorithm 2 802.11 Procedure of the Monte Carlo Simulator

```
1: if $state(i)$ = DIFS-T then
2:     if $channel(i)$ = IDLE then
3:         $DIFScnt(i) = DIFScnt(i) + 1$
4:         if $DIFScnt(i)$ = DIFS then
5:             $DIFScnt(i) = 0$
6:             if $delay(i) = 0$ then
7:                 $CW(i)$, $delay(i)$ randomly set by 802.11DCF
8:             end if
9:             $state(i)$ = CSMA-T
10:        end if
11:    else
12:        $DIFScnt(i) = 0$
13:    end if
14: else if $state(i)$ = CSMA-T then
15:    if $channel(i)$ = IDLE then
16:        $delay(i) = delay(i)$ - 1
17:        if $delay(i) = 0$ then
18:            $nbTrans(i) = nbTrans(i) + 1$
19:            set $busyFor(i)$
20:            $state(i)$ = TRDEL-T
21:        end if
22:    else
23:        $DIFScnt(i) = 0$
24:        $state(i)$ = DIFS-T
25:    end if
26: else if $state(i)$ == TRDEL-T then
27:    $busyFor(i) = busyFor(i)$ - 1
28:    if $busyFor(i) = 0$ then
29:        set $softdelay(i)$
30:        $state(i)$ = OS-T
31:    end if
32: else if $state(i)$ = OS-T then
33:    $softdelay(i) = softdelay(i)$ -1
34:    if $softdelay(i) = 0$ then
35:        $state(i)$ = DIFS-T
36:    end if
37: end if
```

counter was frozen before entering DIFS-T (Line 6). If the backoff counter was frozen, CSMA-T resumes with that backoff counter. Otherwise, the node first updates CW (i.e., contention window size) according to exponential backoff mechanism (i.e. choosing a value equals $CW_{\min}2^m$), and randomly sets *delay* (i.e., the backoff counter) from a uniform distribution between 0 and CW (Line 7), then goes enter the CSMA-T state. When the node is in the CSMA-T state, it decrements the *delay* variable if the channel is free (Line 13). Otherwise, it freezes the backoff counter and goes back to DIFS-T state (Line 19-20). If *delay* reaches zero, the node sets $busyFor$ and increments *nbTrans* before switching to TRDEL-T state (Line 15-17), in which it transmits the data for $busyFor$ time slots (Line 22). *nbTrans* is used to represent the total number of transmissions. The TRDEL-T state is a special state, in which a node transmits, thus other nodes would sense the channel busy. After the transmission, a node does not begin with a new packet immediately. It delays for $softdelay$ time slots in OS-T state (Line 27), which simulates the operating system (OS) or other hardware delays. The node begins a new transmission by going back to DIFS-T.

Algorithm 3 is for BoX-MAC. Since the BoX-MAC protocol does not continuously sense the channel, the mechanism is simpler. In the beginning, the node with data to transmit sets *delay* to a randomly (uniform) picked value between 0 and initial backoff value $CW = CW_{\text{init}}$ (not shown in the pseudo code), and decrements this number in each BoX-MAC time slot (Line 2). When *delay* reaches zero, the node increments $nbCCA$ (Line 12), (used to represent the total number of CCAs), and checks the channel state. If the channel is busy, the node updates *delay* to a value between 0 and $CW = CW_{\text{cong}}$ and resets CS to 2 (Line 20). Otherwise, the node decrements CS (Line 14) (denotes the number of successful CCAs in one CCA stage). If $CS = 0$, the node increments *nbTrans* before transmitting its data for $busyFor$ BoX-MAC time slots (Line 8). After the transmission, the node waits for $softdelay$

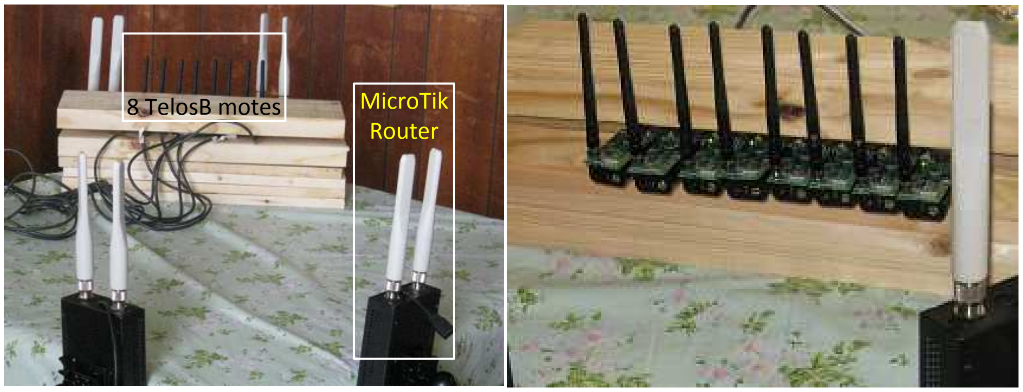
Algorithm 3 BoX-MAC Procedure of the Monte Carlo Simulator

```
1:  if node $i$ is backing off ($delay(i)$ != 0) then
2:      $delay(i) = delay(i)$ - 1
3:  else if $softdelay(i)$ != 0 then
4:      $softdelay(i) = softdelay(i)$ - 1
5:      if $softdelay(i) = 0$ then
6:          set $CS(i)$, $CW(i)$ and $delay(i)$
7:      end if
8:  else if $busyFor(i)$ != 0 then
9:      $busyFor(i) = busyFor$ - 1
10:     if $busyFor(i) = 0$ then
11:         set $softdelay(i)$
12:     end if
13: else
14:     $nbCCA(i) = nbCCA(i) + 1$
15:     if $channel(i)$ == IDLE then
16:         $CS(i) = CS(i)$ - 1
17:         if $CS(i) = 0$ then
18:             $nbTrans(i) = nbTrans + 1$
19:             set $busyFor(i)$
20:         end if
21:     else
22:         if $CS(i)$ != 0 then
23:             update $CS(i)$, $CW(i)$ and $delay(i)$ according to BoX-MAC protocol
24:         end if
25:     end if
26: end if
```

(Line 4), then it resets and starts another transmission.

3.5.2 Simulator Verification

We verify the accuracy of our Monte Carlo simulator by comparing results from it with those from hardware experiments (*∼4 million and ∼60 million transmitted WSN and WiFi packets, respectively*).



(a) (b)

Figure 3.7: Setup of WiFi and TelosB hardware for experiments.

3.5.2.1 Experimental Setup

For our experiments, we use 8 TelosB motes and 4 Microtik WiFi routers. The TelosB motes run TinyOS 2.1, using BoX-MAC as its default MAC layer. The Microtik routers are based on the AR9220 chipset and use the Atheros based R52HN wireless card. The firmware for the MicroTik routers is OpenWrt Backfire 10.03.1 with ath9k wireless driver. OpenWrt is Linux based, thus making the debugging over the serial port relatively easy. We collect the number of packets sent and received for each test scenario, from the TelosB motes through USB cables, and from the Microtik routers over RS232 ports respectively.

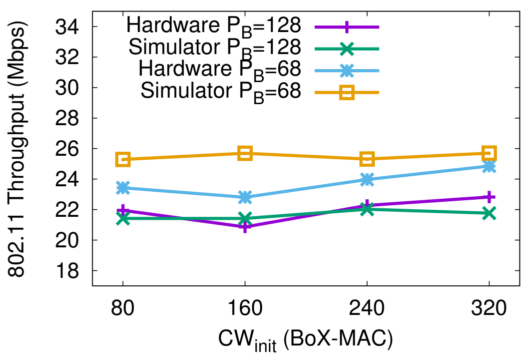
For our experiment, we divide the hardware in two groups: 4 TelosB motes and 2 routers as the transmitters, while the remaining devices are receivers. We use a TinyOS application (called `TxThroughput`) on the TelosB motes to send data continuously. The receiver TelosB motes execute the `BaseStation` application. We configure the MicroTik WiFi routers in 802.11g mode, ran `iperf`, and save the number of packets transmitted/received. To mitigate the irregularity of wireless signal, we place all devices symmetrically, as shown in Figure 3.7.

3.5.2.2 Results for Monte Carlo vs Real Experiments

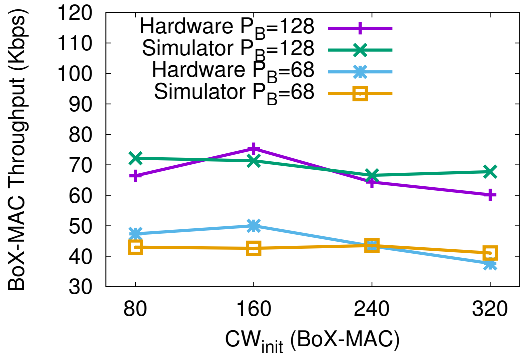
To complete the comparison, we need to first select the protocol parameters to investigate. Intuitively, since 802.11 DCF and BoX-MAC are CSMA/CA-based, the size of the contention window (CW) is a key parameter which could impact the throughput (e.g., smaller window size implies high aggressiveness and more opportunities for collisions). Additionally, the packet size is an important factor since it impacts the time the channel is occupied. Hardware dependent parameters, e.g., OS delay and time slot size cannot be changed on our hardware.

Consequently, the parameters we vary for 802.11 DCF are the minimum contention window size (CW_{min}), the maximum contention window size (CW_{max}) and the packet size (P_W). The values we choose for these parameters are: $CW_{\text{min}} = \{16, 32, 64\}$, $CW_{\text{max}} = \{256, 512, 1024\}$, and $P_W = \{500, 1000, 1500\}$. Similarly, the metrics for BoX-MAC are initial contention window size (CW_{init}), congested contention window size (CW_{cong}), and the packet size (P_B). The values of the parameters are as follows: $CW_{\text{init}} = \{80, 160, 240, 320\}$, $CW_{\text{cong}} = \{40, 60, 80\}$, and $P_B = \{48, 68, 88, 108, 128\}$. We use $\sum_{i=1}^{k}(|T_{S_i} - T_{H_i}| \times 2/(T_{S_i} + T_{H_i}))/k$, the classical formula for the *average difference* between two sets of data (i.e., simulator and real experiment in our case), where k is the number of tests, and T_S and T_H are simulator and hardware throughput, respectively.

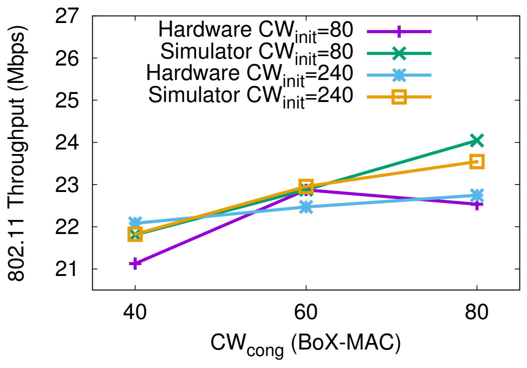
Figure 3.8 shows the throughput for WiFi and ZigBee when varying BoX-MAC parameters only (we keep fixed $CW_{\text{min}} = 16$, $CW_{\text{max}} = 1024$ and $P_W = 1500$). When CW_{init} is varied, and CW_{cong} is set to 80 (Figures 3.8(a-b)), we observe that the throughput of BoX-MAC decreases in both simulator and experiment. This is because at higher CW_{init} BoX-MAC becomes less aggressive. The decrease in



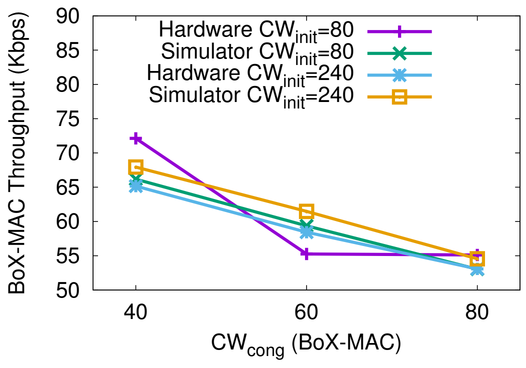
(a)



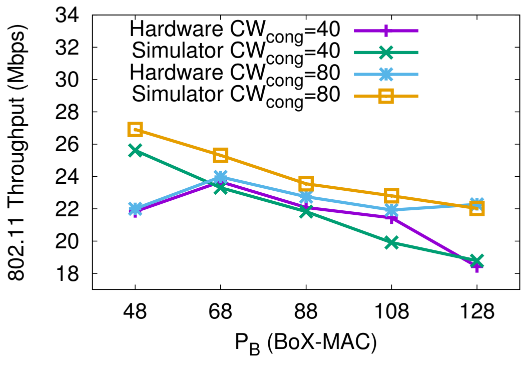
(b)



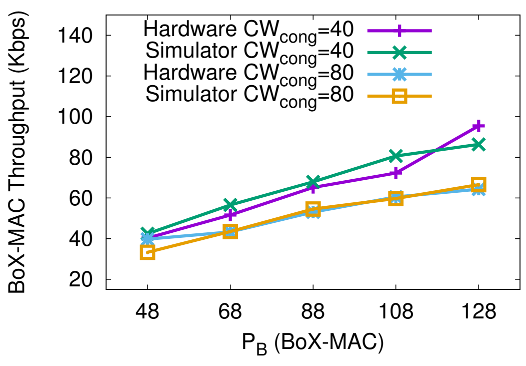
(c)



(d)



(e)

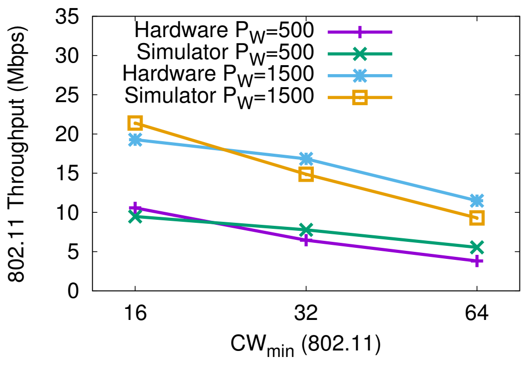


(f)

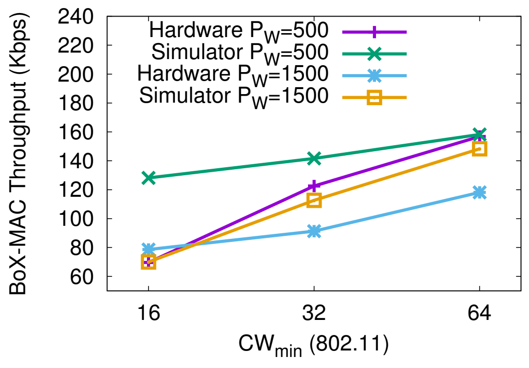
Figure 3.8: 802.11 and BoX-MAC throughput from simulator and hardware when varying BoX-MAC parameters : (a-b) CW_{init}; (c-d) CW_{cong}; and (e-f) P_B

BoX-MAC throughput results in a slight increase in 802.11 throughput. As shown in Figures 3.8(c-d), when CW_{cong} is varied and CW_{init} is set to 240, we observe that for higher CW_{cong} the throughput for BoX-MAC decreases more than for lower CW_{init}. At higher CW_{cong}, BoX-MAC is less aggressive, resulting in a lower throughput. 802.11 benefits from this, and its throughput slightly increases. The results when varying packet size and CW_{cong}, while keeping $CW_{\text{init}} = 240$ are shown in Figures 3.8(e-f). We observe that the packet size has the biggest effect on BoX-MAC throughput. With larger packets, the throughput increases by as much as 100%. As shown, this is at the expense of 802.11 throughput. It is interesting to note that our results show a remarkable agreement (average difference 6%, a worst case of 22%) between simulator and real world.

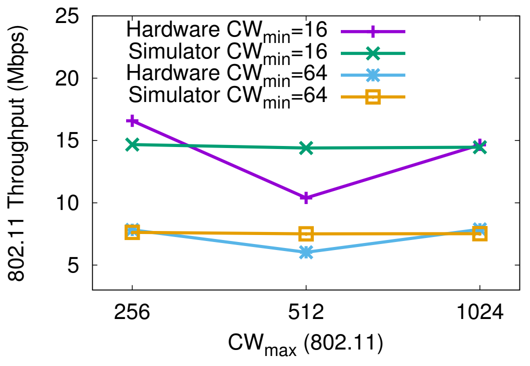
Figure 3.9 shows the throughput of WiFi and BoX-MAC when varying WiFi parameters only (we keep fixed $CW_{\text{init}} = 320$, $CW_{\text{max}} = 80$ and $P_B = 128$). Figures 3.9(a-b) show the results when varying CW_{min} and packet sizes, while keeping $CW_{\text{max}} = 1024$. Upon increasing CW_{min}, throughput of WiFi decreases, since they are less aggressive at larger CW_{min}. This benefits BoX-MAC devices, for which the throughput increases. Figures 3.9(c-d) and 3.9(e-f) show that for larger P_W, the throughput of WiFi increases, since the channel is more efficiently occupied by WiFi transmissions. This is at the expense of degrading BoX-MAC throughput. Real hardware experiments are subject to noise and other irregularities that cannot be modeled in simulation. Therefore we observe that throughput of real hardware fluctuates. We see an average difference of 17%, and a worst case difference of 84% between simulator and hardware experiment as a result of the noise. WiFi devices can take over the channel, since 802.11 DCF is more aggressive than BoX-MAC. By changing WiFi parameters, the throughput of BoX-MAC varies significantly, whereas changing BoX-MAC parameters has a smaller impact, as is apparent in Figures 3.8



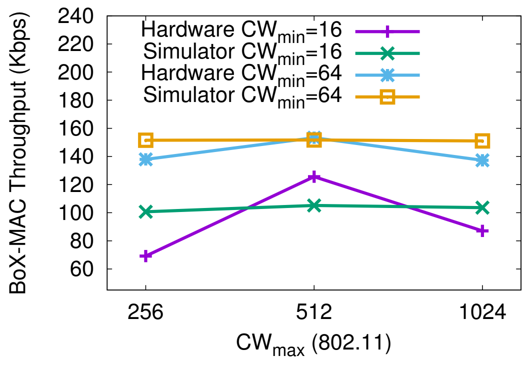
(a)



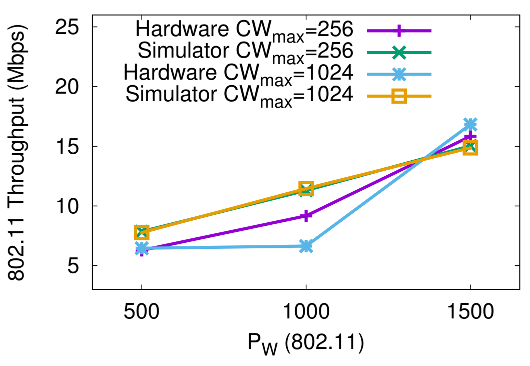
(b)



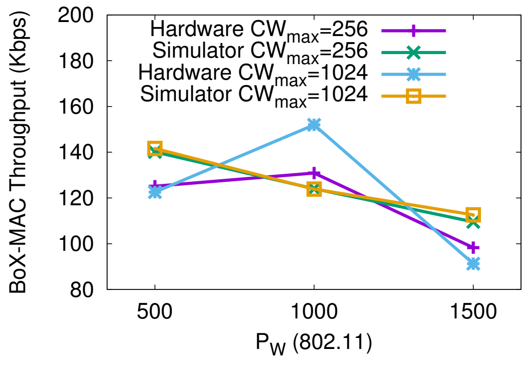
(c)



(d)



(e)



(f)

Figure 3.9: 802.11 and BoX-MAC throughput from simulator and real hardware when varying 802.11 parameters: (a-b) $CW_{\min}$; (c-d) $CW_{\max}$; and (e-f) P_W.

and 3.9.

3.6 Model and Tuning Evaluation

In this section we investigate the accuracy of our analytical model and present the performance evaluation of our CW tuning methods. For this, we use the coexistence simulator presented in Section 3.5. We chose to use our own simulator because no existing existing academic or commercial simulator, e.g., ns2, QualNet, etc., supports wireless coexistence scenarios (please note, coexistence of different wireless MAC protocols). Additionally, we need the ability to control all low level parameters in the simulator (e.g., computations of different collision probabilities), which is a capability typically not exposed to users, in commercial simulators. The proposed Monte Carlo based simulator mimics 802.11 and 802.15.4 protocols at the MAC layer. The simulator's accuracy has been validated through extensive experiments on real hardware (4 MikroTik WiFi routers, 8 TelosB motes and over 60 million transmitted packets). We chose a simulator based approach for validating our analytical model and evaluating our CW tuning methods because it is rather very difficult to obtain results from large scale deployments (i.e., scalability issue), and still maintain the ability to evaluate multiple configuration settings (i.e., various protocol parameters).

3.6.1 Simulation parameters

Since 802.11 DCF and BoX-MAC are CSMA/CA-based, the *contention window size* is a key parameter which impacts the throughput (e.g., a smaller window size is more aggressive, but it gives more opportunities for collisions). The packet size is also an important factor since it impacts the time the channel is occupied. In our evaluations, the metrics we choose for 802.11 DCF are the minimum contention window size $CW_{\min} = \{16, 32, 64\}$, the maximum contention window size $CW_{\max} = \{256, 512, 1024\}$ and the packet size $P_W = \{500, 1000, 1500\}$. Similarly, the metrics for BoX-MAC are the initial contention window size $CW_{\text{init}} = \{80, 160, 240, 320\}$, congested contention window size $CW_{\text{cong}} = \{40, 60, 80\}$, and the packet size $P_B = \{48, 68, 88, 108, 128\}$. We used $\sum_{i=1}^{k}(|T_{S_i} - T_{H_i}| \times 2/(T_{S_i} + T_{H_i}))/k$, the classical *average difference* between two sets of data (i.e., simulator and analytical model in our case) as the evaluation metric, where k is the number of tests, and T_S and T_H are simulator and model throughput, respectively.

3.6.2 Analytical Model Validation

We compare the normalized throughput (as in [48] [74]) obtained from our analytical model with that obtained from the Monte Carlo simulator. The parameters we vary are: the contention window size CW_{min}, maximum contention window size CW_{max}, and the packet size P_W of WiFi, and the initial contention window size CW_{init}, contention window size CW_{cong} and the packet size P_B for BoX-MAC. The default values of these parameters are: $CW_{\text{min}} = 32$, $CW_{\text{max}} = 1024$, $P_W = 1500$ for WiFi, and $CW_{\text{init}} = 320$, $CW_{\text{cong}} = 80$, $P_B = 48$. Due to approximations made in the Markov Chain model (e.g., DIFS delay after backoff freeze), we see differences in results obtained from simulator and the analytical model. We use the aforementioned average difference metric to compare them.

3.6.2.1 The number of devices

Analyzing the coexistence of WiFi and WSN by varying the number of devices in a real implementation is tedious and time consuming. The same can be done by merely varying these parameters in the analytical model and simulator. Figure 3.10 depicts the comparison of normalized throughput when the number of WiFi and BoX-MAC devices are varied, setting other parameters to their default values. The results indicate that increasing the number of WiFi devices increases the throughput of WiFi and degrades the throughput of BoX-MAC, while increasing the number of

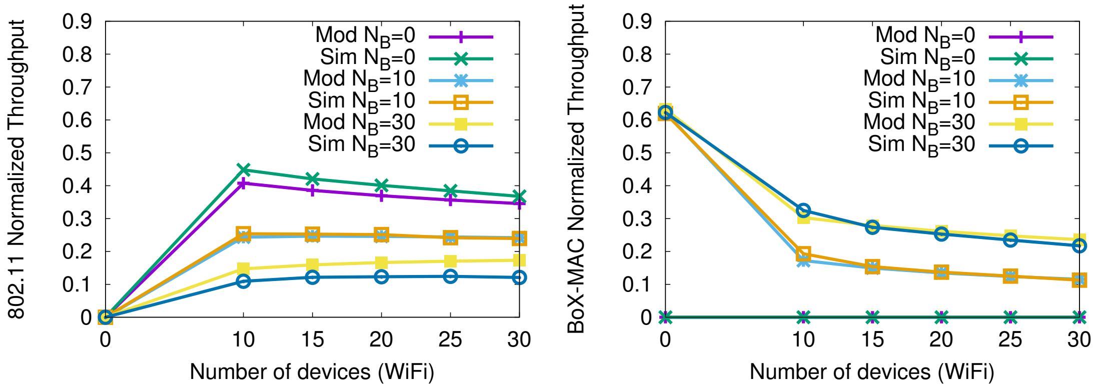
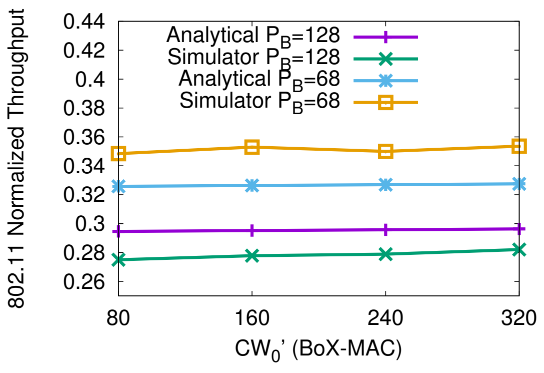


Figure 3.10: Throughput of model and simulator vs number of devices.

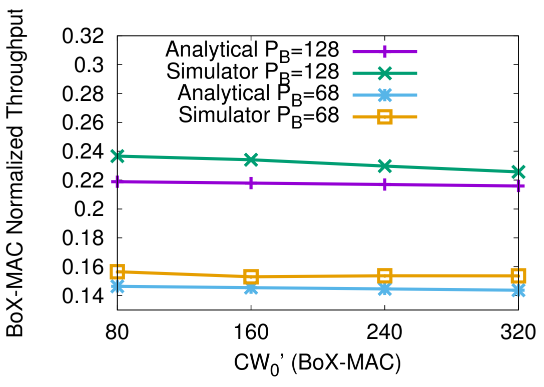
BoX-MAC devices increases BoX-MAC throughput and degrades WiFi throughput. Interestingly, there is good agreement between simulator and analysis with an average difference of 3% and with worst difference of 6%.

3.6.2.2 Throughput Comparison

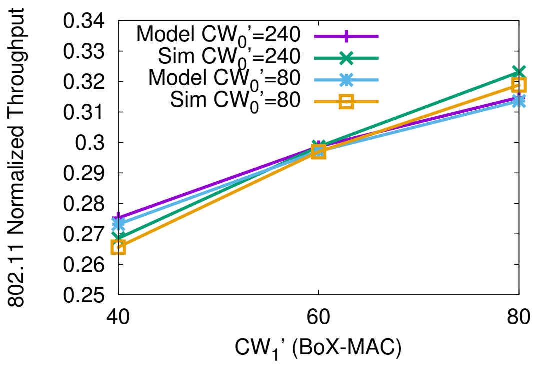
To analyze the impact of BoX-MAC parameters on throughput, as obtained from the simulator and from the analytical model, we considered a scenario with 15 WiFi, 30 BoX-MAC devices and default WiFi parameters. The results are depicted in Figure 3.11. Figures 3.11 (a-b) show that increasing CW_{init} the throughput of BoX-MAC remains almost the same. This is because with a big number of devices, the probability of transmitting after the first backoff attempt is low. As CW_{cong} increases (in Figures 3.11 (c-d)), the backoff overhead of BoX-MAC increases, thus making its throughput decrease. The trend for varying P_B is expected (in Figures. 3.12 (a-b)), since a big packet size implies larger payload and longer channel occupancy, both of which benefit BoX-MAC. Results show an excellent agreement (average difference of 3% and worst case of 6%) between analysis and simulator. Notably, since CW_{init} is not effective in throughput scaling, it should not be considered as a parameter for



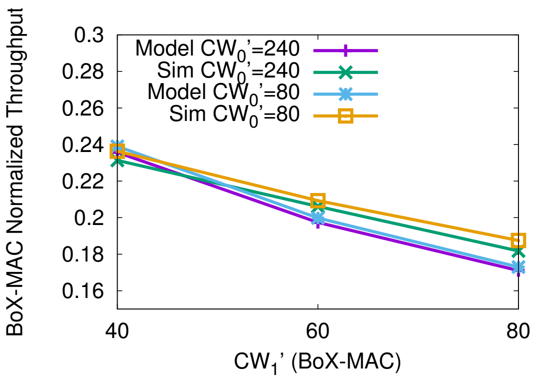
(a)



(b)



(c)

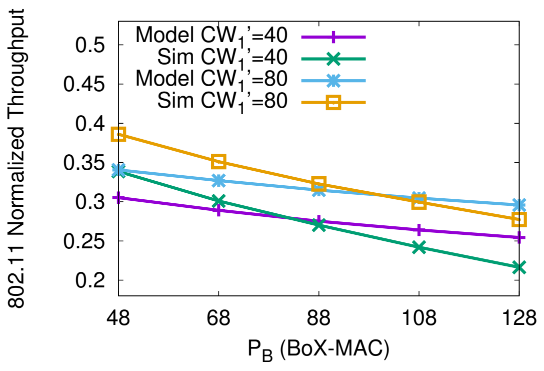


(d)

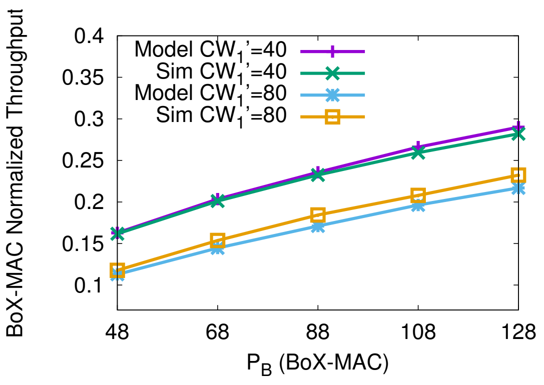
Figure 3.11: Model validation: 802.11 and BoX-MAC throughput from model and simulator when varying BoX-MAC parameters [(a-b) CW_{init}; (c-d) CW_{cong}].

tuning.

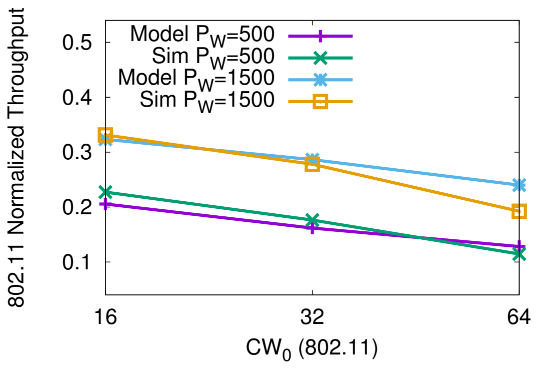
The impact of WiFi parameters on throughput derived from simulator and analytical model was analyzed with the same number of devices and default BoX-MAC parameters. Results are depicted in Figure 3.12 and 3.13. One can observe that CW_{min} has a significant impact on throughput, as shown in Figures 3.12 (c-d). In terms of CW_{max}, since the collision probability is not too high with this number of nodes, it is unlikely that WiFi nodes reach the maximum contention window size, for successful transmissions. Also, CW_{max} does not affect the throughput, as shown in



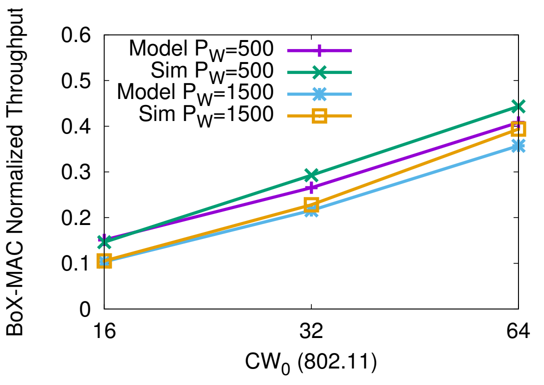
(a)



(b)



(c)



(d)

Figure 3.12: Model validation: 802.11 and BoX-MAC throughput from model and simulator when varying BoX-MAC parameters [(e-f) P_B] and when varying 802.11 parameters [(g-h) $CW_{\min}$].

Figures 3.13 (a-b). As for P_W (in Figures 3.13 (c-d)), the trend is also expected, i.e., it impacts the throughput greatly. These results also show a remarkable agreement (average difference of 2% with a worst case of 5%) between analysis and simulator. The agreement between the simulator and the analysis in all these experiments validates the analytical model as an extremely valuable tool. Similarly, $CW_{\max}$ is not an effective parameter for throughput tuning.

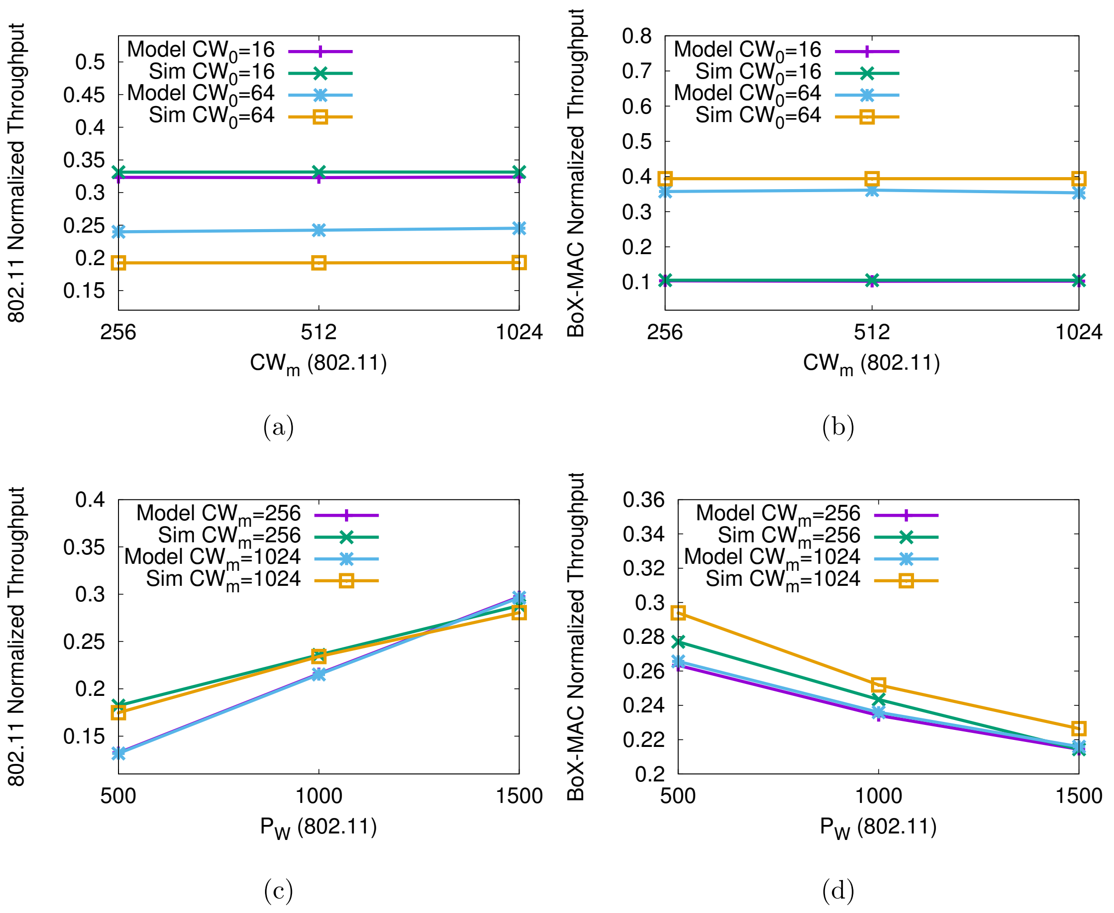


Figure 3.13: Model validation: 802.11 and BoX-MAC throughput from model and simulator when varying 802.11 parameters [(i-j) $CW_{\max}$ and (k-l) P_W].

3.6.3 Contention Window Tuning Evaluation

We evaluate the tuning methods proposed in Section 3.4 using our simulator. First, we demonstrate that the priority can be satisfied through our CW tuning method, and that the total throughput is also maximized. Second, we show that the tuning method we present reaches Nash equilibrium. Then we show that attaining Nash equilibrium can also solve the fairness problem, i.e., unfair bandwidth utilization, which is seen when nodes with different $CW_{\min}$ and CW_{cong} have different

No. of WiFi	No. of BoX-MAC	$\phi1$	$\phi2$	$\phi3$	$\phi4$	$\phi5$
10	10	1	2	5	10	20
15	15	1	2	5	10	20
20	20	1	2	5	10	20
10	15	1	2	5	10	20
15	10	1	2	5	10	20
15	20	1	2	5	10	20
20	15	1	2	5	10	20

Table 3.3: Settings for priority tuning evaluation

throughput.

3.6.3.1 Evaluation of priority tuning

The basic idea of the evaluation is that given the number of BoX-MAC and WiFi nodes, and information like packet size, a centralized master device can communicate with both types of devices and calculate the contention window size based on the CW tuning method. Each node will be informed of the CW size to use, and as a result, the priority requirement is met while the total throughput is maximized. We perform simulations using several combinations of BoX-MAC and WiFi nodes, and different priority settings, i.e., ϕ, as in Table 3.3. We now show the optimal contention window sizes that satisfy the requirement.

As shown in Figures 3.14 (a-b), the optimized CW size is quite different from the default settings ($CW_{\text{min}} = 32$, $CW_{\text{max}} = 1024$, $P_W = 1500$ for WiFi, and $CW_{\text{init}} = 160$, $CW_{\text{cong}} = 80$, $P_B = 48$ for BoX-MAC), which emphasizes the necessity for CW tuning. The realized throughput at these CW sizes is shown in Figure 3.14 (c-d). Notably, the throughput from default contention window size we chose was very close to the optimal value for the priority parameter $\phi = 10$. For other CW combinations, the results were not as good as when the value was obtained from our optimization.

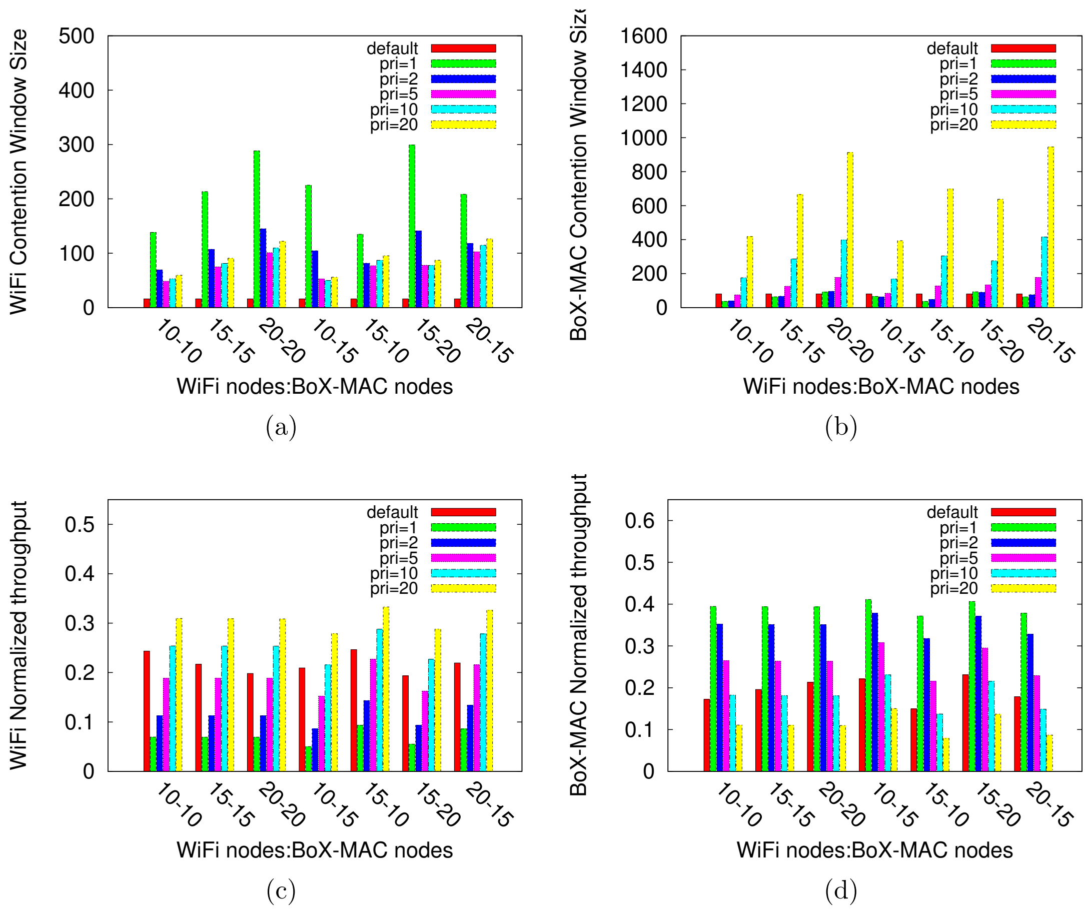


Figure 3.14: CW tuning for priority: Optimized CW size for (a) 802.11 and (b) BoX-MAC nodes based on deployment size. The realized throughput for these CW sizes is shown in (c) for WiFi and (d) for BoX-MAC.

3.6.3.2 Evaluation of Fairness tuning

As mentioned before, the fairness problem is due to nodes choosing different corresponding contention window sizes, i.e., $CW_{\min}$ and CW_{cong}. However, the solution for this problem is quite straightforward, since, generally, nodes of the same kind tend to choose the same parameters when Nash equilibrium is reached which implies that the realized throughputs are equal (they share the bandwidth fairly). We performed simulations for 5 WiFi and 10 BoX-MAC nodes. The nodes initially chose

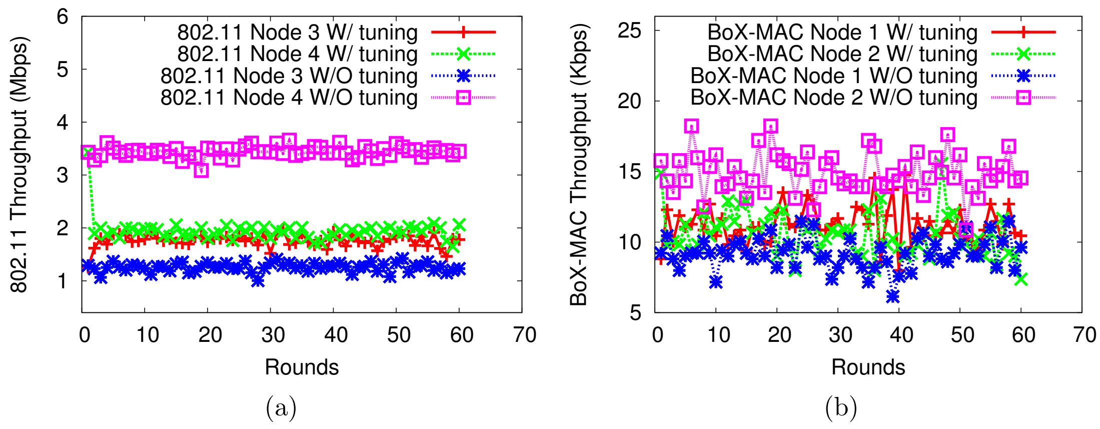


Figure 3.15: Throughput of 802.11 and BoX-MAC are fairly shared by tuning CW_{cong} and CW_{min}.

random CW_{cong} and CW_{min}. To compare the results for fairness, we ran two separate tests, whose results are shown in Figure 3.15. It is obvious that when the Nash equilibrium was attained, the devices fairly shared the throughput, while without tuning, the throughput can be quite different, based on the parameters they choose initially.

4. COEXISTENCE OF NON-SLEEPING WIFI AND ZIGBEE WITH UNSATURATED TRAFFIC

An analytical model and a simulator have been obtained for the coexistence of non-sleeping WiFi and ZigBee with saturated traffic in Chapter III, an endeavor is necessary to understand the more general case where traffic pattern is unsaturated, such that the corresponding performance analysis and tuning can be achieved. Thus, in this chapter we discuss a mathematical model as well as a simulator for symmetric coexistence of non-sleeping WiFi and ZigBee with unsaturated traffic (i.e. UNSAT).

4.1 Mathematical Model for UNSAT

In this section, we first of all briefly describe the BoX-MAC standard and some key assumptions of our Markov Chain (MC) model. Then we introduce the MC model in detail (it is worth noting that, this MC model, which, is quite different from the one for SAT, significantly improves the accuracy). Finally we cover the $M/G/1$ queueing model and derive the expressions of throughput and delay.

4.1.1 Preliminaries and Assumptions

As before, we consider BoX-MAC in this dissertation because it is the MAC protocol of TinyOS, and yet a simplified version of 802.15.4. Generally, 802.15.4 has several double sized contention window (CW) backoff stages up to a maximal one (e.g. $CW = 8, 16, 32$ etc.), while BoX-MAC only uses two such backoff stages (shown as $CW = W_0', W_1'$ in Figure 4.2). Same as 802.15.4, BoX-MAC employs the double-channel-sensing (DCS) mechanism, i.e. a device transmit a packet only if the channel stays idle for two continuous time slots.

We assume that all devices employ *energy based modulation*, and are within a single wireless cell. The same type of devices are homogeneous, i.e. their traffic pattern are Poisson with equal λ , and they transmit *packets of equal size* L (i.e. $\lambda_i = \lambda_j$ and $L_i = L_j$ if nodes i and j are of same type). For simplicity, we also consider an *ideal channel* (i.e. no shadowing, fading and capture effect), implying communication fails only due to collisions. Moreover, we make two fundamental, yet widely used assumptions on the probabilities of a reference device: *a) the probability of a transmission attempt is constant and independent of the attempts of other devices; and b) the collision probability (conditioning on an attempt) is constant and independent of the number of collisions experienced. These assumptions were proven to be accurate [48]- [93].*

4.1.2 Markov Chain model for 802.11

The Markov Chain for 802.11 is shown in Figure 4.1. Unlike the MC for BoX-MAC and the one in last chapter, each state in this one is three dimensional, i.e. $(s(t), c(t), p(t))$. The first two stochastic processes have same meaning as BoX-MAC (the backoff behavior is also similar expect there are m stages instead of 2 for 802.11, and for conciseness the details are not elaborated here), while the third one $p(t)$ represents the channel status in the previous time slot (i.e. $p(t) = 1$ or 0, corresponds to busy or idle). Generally, the purpose of $p(t)$ is to describe the event that the channel being idle for two continuous baseline time slots, which is critical for BoX-MAC devices because they can transmit data only if the double-channel-sensing is passed. Notably, for each backoff stage in Figure 4.1 (i.e. each solid line box), the bigger circles represent the time instances that the channel is idle, while each smaller circle denotes an ongoing transmission by other devices (either BoX-MAC or WiFi or both, see the dashed line box in Figure 4.1). The probability P_f is the transition probability from state $(i, j, 0)$ to a channel busy state (i.e. at least one other node is

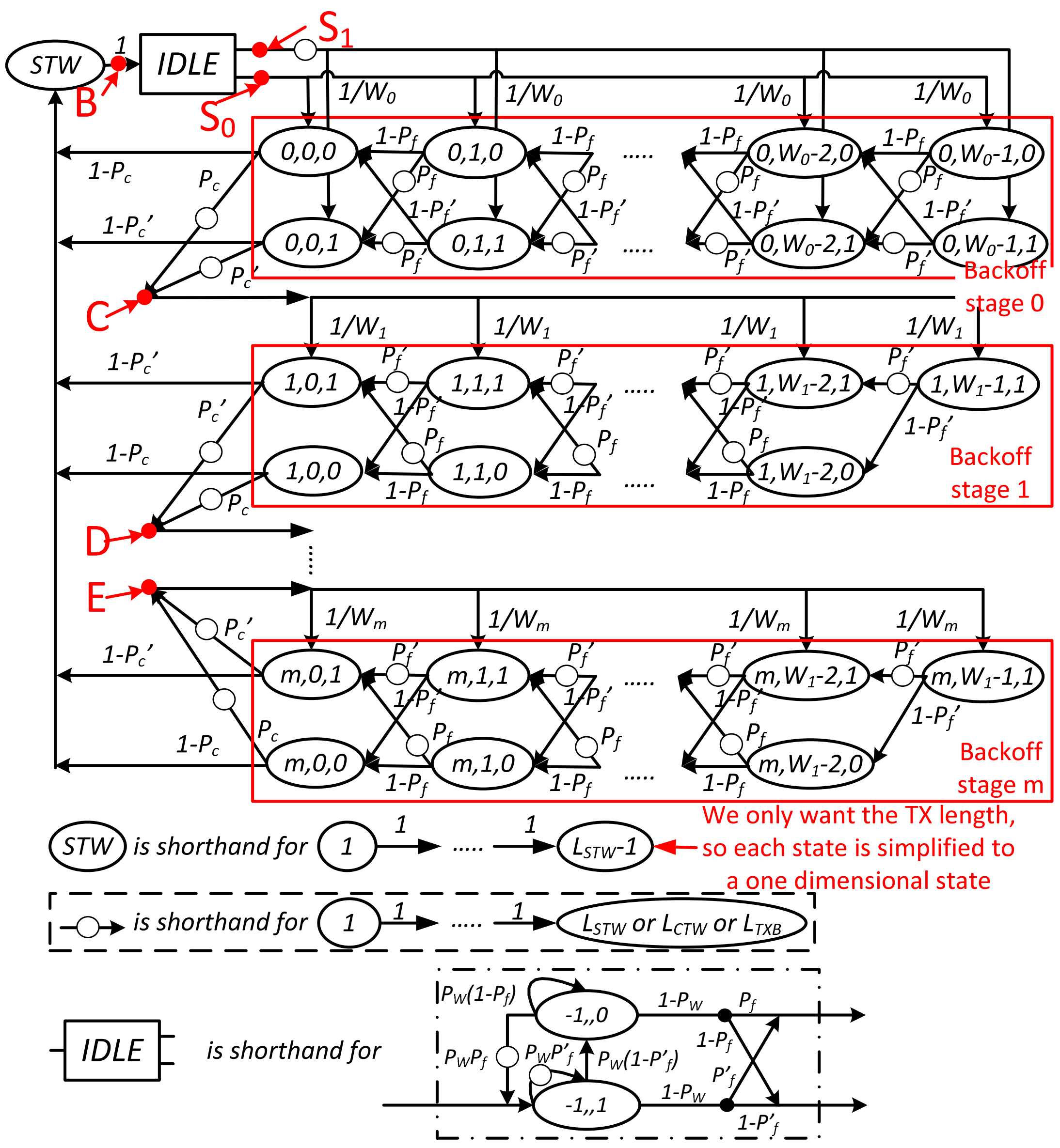


Figure 4.1: Markov Chain describing the 802.11 protocol.

sending). Since $(i, j, 0)$ means that its previous time slot is idle, which implies that there are two continuous idle time slots, BoX-MAC devices have chance to transmit. However, from state $(i, j, 1)$, since there is single idle slot for the channel, BoX-MAC cannot seize the channel, thus the corresponding P_f' is not equal to P_f. Note that for states where the reference device attempts to transmit, i.e. states $(1 \sim m, 0, 0 \sim 1)$, the corresponding transition probabilities are denoted as P_c and P'_c (i.e. collision probability), and since both P_c and P_f imply the same transition probability, we have $P_c = P_f$ (similarly, $P'_c = P'_f$).

The state $IDLE$ is a shorthand for several states (shown in the dot-dash line box in Figure 4.1) according to the behavior of 802.11. When the transmission queue is empty (with probability P_W), the device has to wait in $(-1, , 0)$ or $(-1, , 1)$ state. Note that here we denote by $s(t) = -1$ the queue being empty (abused a little bit), and as before $p(t) = 0$ or 1 depends on the channel status of the last time slot. Since the transitions between $(-1, , 0)$ and $(-1, , 1)$ are fairly simple (e.g. the self-loop for $(-1, , 0)$ happens only if the queue is empty and no other nodes is transmitting, i.e. $P_W(1 - P_f)$), the elaboration is omitted here. When the queue is not empty (w.p. $1 - P_W$) and if no one is using the channel (w.p. $1 - P_f$ or $1 - P'_f$), the device will begin with the backoff process from $(0, j, 0)$; if the channel is being used (w.p. P_f or P'_f), the device simply wait until the transmission finishes and begin to backoff from $(0, j, 1)$. Notably for backoff stages other than 0, there is only one input (i.e. to $(1 \cdots m, j, 1)$) because from their (i.e. stages 1 to m) perspectives the channel is always busy in the previous time slot due to the collided transmission.

Before deriving P_f and P'_f, we first of all discuss the probability that a 802.11 device attempts to transmit, i.e. τ_W, which is needed in the expressions of P_f and P'_f. Similar to ϕ_B we discussed before, τ_W is also a probability conditioned on the channel being idle, thus we have

$$\tau_W = \frac{\sum_{i=0}^{m} \sum_{k=0}^{1} b_{i,0,k}}{\sum_{i=0}^{m} \sum_{j=0}^{W_i} \sum_{k=0}^{1} b_{i,j,k} + b_{IDLE}} \tag{4.1}$$

where $b_{i,j,k}$ (and b_{IDLE}) represents the stationary probability of state (i, j, k) (and state $IDLE$) in the MC.

Unfortunately, for such a complex MC, there is no simple close form expression for τ_W, we therefore resort to a more general method, namely the transition matrix method. Since each transition probability for the 802.11 MC is known (symbols such as P_f, P_c are also considered known because they will be replaced by initial guesses of number when using numerical method), its transition matrix $\mathbf{T}$ can be built. And for a MC being stationary, $\boldsymbol{\pi}\mathbf{T} = \boldsymbol{\pi}$ must hold, where $\boldsymbol{\pi}$ is the row vector stores the stationary probabilities of all states. Then by using the condition that $\sum \pi_i = 1$, each element in $\boldsymbol{\pi}$ (i.e. π_i) can be obtained. Thus the value of τ_W can be computed.

Having obtained τ_W, we are ready to discuss P_f and P'_f. For P_f, since it is conditioned on $p(t) = 0$ (two continuous idle time slots), thus BoX-MAC can transmit, then we have

$$P_f = 1 - (1 - \phi_B)^{N_B}(1 - \tau_W)^{N_W - 1} = P_c \tag{4.2}$$

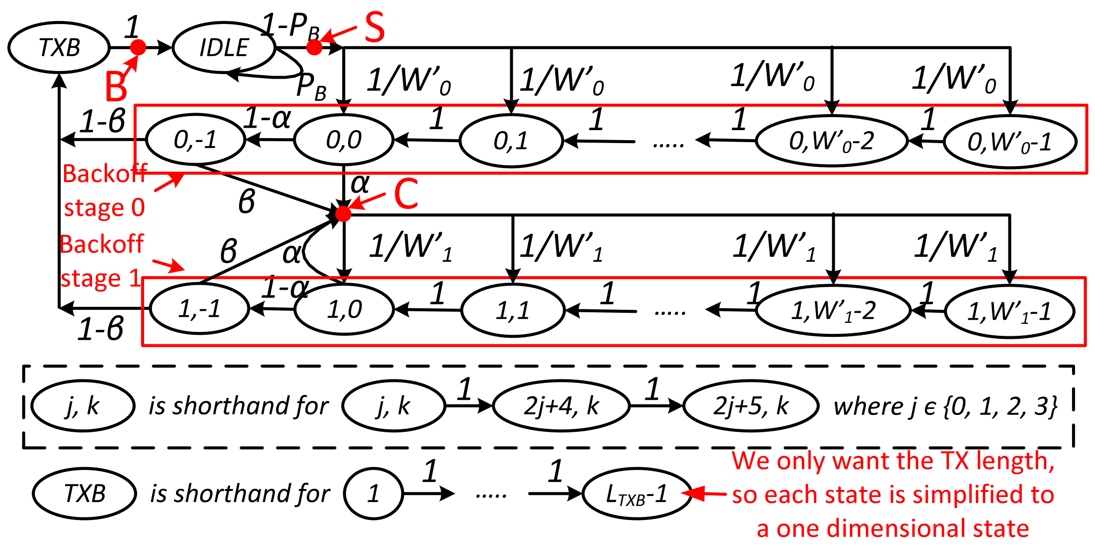
where N_B and N_W are the number of BoX-MAC and WiFi devices, respectively. While for P'_f, since the previous time slot is busy, BoX-MAC has no chance to transmit, thus

$$P'_f = 1 - (1 - \tau_W)^{N_W - 1} = P'_c \tag{4.3}$$

As mentioned before, α and β can be derived using a Markov Chain for the channel, which is discussed next.

4.1.3 Markov Chain model for BoX-MAC

The MC for BoX-MAC is shown in Figure 4.2, where each state is two dimensional $(s(t), c(t))$. Specifically, $s(t)$ is a stochastic process representing the current backoff stage (i.e. two backoff stages as 0 and 1); while $c(t)$ is a process representing the



TXB
1
IDLE
1-P_B
P_B
S
B
C
1/W'_0
1-β
1-α
0,-1
0,0
0,1
0,W'_0-2
0,W'_0-1
1
.....
Backoff stage 0
Backoff stage 1
β
α
1/W'_1
1,-1
1,0
1,1
1,W'_1-2
1,W'_1-1
j, k is shorthand for j, k → 2j+4, k → 2j+5, k where j ϵ {0, 1, 2, 3}
TXB is shorthand for 1 → ..... → L_TXB-1
We only want the TX length, so each state is simplified to a one dimensional state

Figure 4.2: Markov Chain for the BoX-MAC protocol

current backoff counter in the corresponding stage ($0 \leq c(t) \leq W'_0$ or W'_1). Notably, in the dashed-line box in Figure 4.2, since BoX-MAC has a time slot 3 times longer than 802.11 (i.e. 802.11 time slot is our baseline), to account for the difference in slot sizes, for each state (j, k) (*not including the states for double-sensing, i.e.* $(0, -1)$ *and* $(1, -1)$, *because double-sensing is generally fast, and we assume each such state takes one baseline time slot*) in the BoX-MAC MC, we add two dummy states to it (i.e. $(2j + 4, k)$, and $(2j + 5, k)$). Thus the node entering state (j, k) will transit to state $(2j + 4, k)$, and then to $(2j + 5, k)$ with probability 1, thereby accounting for an entire BoX-MAC slot. *For clear presentation, we omit the two dummy states henceforth.*

In the BoX-MAC protocol, a device starts by choosing a random number from 0 to $W'_0 - 1$, and then begin with the backoff procedure by decrementing the backoff counter. In the Markov Chain, this behavior corresponds to starting from state $(0, k)$, where k is a random number between 0 and $W'_0 - 1$, and then move to the $(0, k - 1)$ and so on so forth, until it reaches states $(j, 0)$ and $(j, -1)$, where the double-channel-sensing is performed. If the channel is sensed busy (with probability $\alpha + (1 - \alpha)\beta$, where α and β are the probabilities that the node finds the channel busy for the first and the second time, respectively), the device transitions to state $(1, k)$ where k is a random number between 0 and $W'_1 - 1$. If the channel is sensed idle in both states $(j, 0)$ and $(j, -1)$, the packet is transmitted. Transmission is represented by TXB, which includes L_{TXB} tandem states with all transition probabilities are equal to one (L_{TXB} is the duration of a BoX-MAC transmission). Since two-dimension for TXB is meaningless, each state is simplified to one dimensional. Unlike the saturated traffic scenario, for this unsaturated case, the probability that the transmission queue being empty is not zero, and we denote by P_B such probability. Thus after a transmission, a device enters $IDLE$ state to check the status of its queue, i.e. wherever there is no packet in the queue (w.p. P_B), it stays in the $IDLE$; otherwise, it begins a new backoff process immediately. *Note that since β is conditioned on the success of the first channel sensing, it differs from the model in last chapter where $\beta = \alpha$ is improperly assumed.*

The next step is deriving the expression for the probability that a BoX-MAC device attempts to sense the channel for the first time, i.e. ϕ_B. Notably, ϕ_B is the another difference from Chapter 3 where the probability that a BoX-MAC node attempts to send (i.e. τ_B) is derived. *The reason we use ϕ_B here is that it gives us opportunity to model the details of the coexistence, lacking of which bring inaccuracies.* Moreover, since ϕ_B is a conditional probability given that the reference node is not transmitting, the expression for it is $\phi_B = \sum_{i=0}^{1} b'_{i,0} / (\sum_{i=0}^{1} \sum_{j=0}^{W'_i} b'_{i,j} + b'_{IDLE})$, where $b'_{i,j}$ (and b'_{IDLE}) represents the stationary probability of state (i, j) (and state $IDLE$) in the MC. The denominator of the equation serves as the condition for ϕ_B because it includes all states that the reference node is not transmitting. Then by similar method in [48], we can derive an elegant form of ϕ_B as

$$\phi_B = \frac{2(1-P_B)}{3((3+(1-\alpha)\beta+\alpha)(1-P_B)+2(1-\alpha)(1-\beta))} \tag{4.4}$$

As we can see, Equation (4.4) needs three unknown probabilities, i.e. α, β and P_B. For the derivations of the former two, we will leverage a MC for the channel (shown later), while for P_B, a queueing analysis will be used.

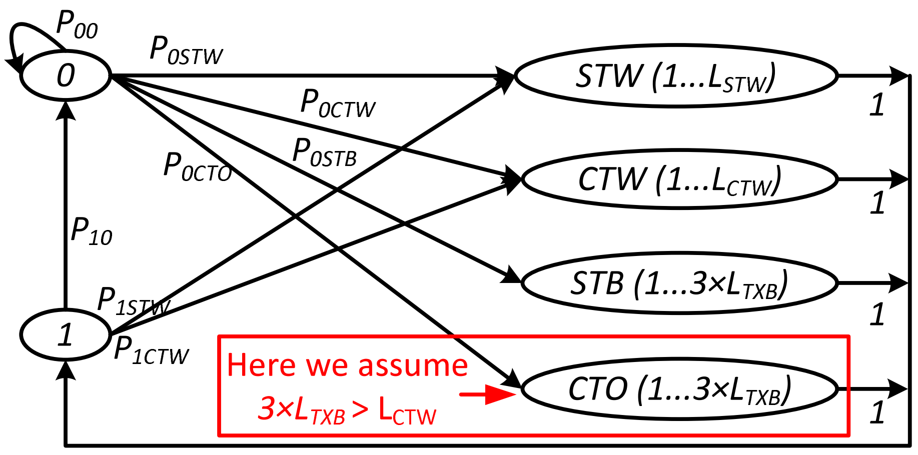


Figure 4.3: Markov Chain describing the channel.

4.1.4 Markov Chain model for the Channel

The Markov Chain describing the channel is shown in Figure 4.3. This MC is simple and only one dimensional. In particular, state 0 represents a currently idle channel with the previous time slot being idle, while state 1 represents a currently idle channel with the previous time slot being busy. States STW, CTW, STB and CTO are similar to STW in the MC of 802.11, and they respectively denote a successful WiFi transmission, a collision WiFi transmission, a successful BoX-MAC transmission and all other cases. Notably, we assume that the occupation of a BoX-MAC packet in the channel is longer than that of WiFi, thus the length for CTO is the length of a BoX-MAC packet. Same as the cases in the 802.11 MC, BoX-MAC can only send data when the channel is idle for two continuous time slots (i.e. state 0), thus from 0 there are four transitions to STW, CTW, STB and CTO respectively. Whereas from 1, there are only two transitions to STW and CTW because BoX-MAC has no chance to transmit in this case. According to the meaning of these transitions, we first of all define $A_0 = (1-\phi_B)^{N_B}$ (i.e. no BoX-MAC attempts to send), $A_1 = N_B\phi_B(1-\phi_B)^{N_B-1}$(i.e. only one BoX-MAC attempts to send), $B_0 = (1-\tau_W)^{N_W}$ (i.e. no WiFi attempts to send) and $B_1 = N_W\tau_W(1-\tau_W)^{N_W-1}$ (i.e. only one WiFi attempts to send), and then we easily have $P_{00} = A_0 \cdot B_0$, $P_{0STW} = A_0 \cdot B_1$, $P_{0CTW} = A_0 \cdot (1 - B_0 - B_1)$, $P_{0STB} = A_1 \cdot B_0$ and $P_{0CTO} = 1 - A_0 - P_{0STB}$ for state 0, and $P_{10} = B_0$, $P_{1STW} = B_1$ and $P_{1CTW} = 1 - B_0 - P_{1STW}$ for state 1. By solving this simple MC (using the same method as in [48]), and denoting $P_{0STW} + P_{0CTW} + P_{0STB} + P_{0CTO}$ by P_{0X} as well as $P_{1STW} + P_{1CTW}$ by P_{1X} for simplicity, we obtain the stationary probabilities of state 0 and 1 as,

$$\begin{aligned} b_0 &= \frac{1-P_{00}}{P_{10}(1+P_{0X}) + (1-P_{00})(1+P_{1X})} \\ b_1 &= \frac{P_{10}}{P_{10}(1+P_{0X}) + (1-P_{00})(1+P_{1X})}. \end{aligned}$$

Then by the definition of α, i.e. the probability that the channel being busy, then $\alpha = 1-$P(channel idle), thus

$$\alpha = 1 - (b_0 + b_1) \tag{4.5}$$

The derivation of β is little tricky because it is a probability of channel being busy in current time slot conditioned on it was idle in last time slot, namely $P(b|i)$. What we do is utilizing the law of total probability i.e. $P(b|i) = P(b,i)/P(i) = (P(i_0)P(b|i_0) + P(i_1)P(b|i_1))/P(i)$, which gives us

$$\beta = \frac{b_0(1 - A_0 \cdot B_0) + b_1(1 - B_0)}{b_0 + b_1} \tag{4.6}$$

So far, we have eight unknowns (i.e. ϕ_B, α, β, τ_W, P_f, P'_f, P_B, P_W) and six Equations (4.4)-(4.6). We move to queueing theory to model the transmission buffers of BoX-MAC and 802.11 as two $M/G/1$ queues to find the remaining two expressions for P_B and P_W. Note that in this dissertation we use the terms buffer and queue interchangeably.

4.1.5 $M/G/1$ queuing models for 802.11 and BoX-MAC

We first define λ_B and μ_B as the arrival rate and the service rate for the queue of a BoX-MAC node, and λ_W and μ_W as the same for a 802.11 node. For $M/G/1$ queue, we have $P_B = 1 - \rho_B$ and $P_W = 1 - \rho_W$, where ρ_B and ρ_W are the traffic intensities (defined as $\rho_B = \lambda_B/\mu_B$ and $\rho_W = \lambda_W/\mu_W$), and since the arrival process is Poisson, we only need to get the expressions for μ_B and μ_W.

Since service processes are based on the CSMA behavior, which is generally distributed, there is no simple expressions for them. However, as mentioned before, what we really need is the service rate, i.e. the mean of the service time. In the following subsections, we first of all introduce the concept of probability generating function (PGF) and how it is related to mean and variance. Then by using PGF, we derive expressions for the mean (i.e. μ_B and μ_W) and variance (i.e. σ_B^2 and σ_W^2). Notably the variance is needed in the computation of delay.

4.1.5.1 Probability generating function (PGF)

PGF is a mathematical tool to describe a discrete random variable (r.v.). Assume X is a discrete r.v. taking values in $\{0, 1, ...\}$, then its PGF is defined as: $G(z) = \sum_{x=0}^{\infty} p(x)z^x$, where p is the probability mass function of X. PGF has two important properties, i.e. the expectation of X is given by $\mathrm{E}\left(X\right) = \frac{dG(z)}{dz}|_{z=1} = G'(1)$, while the variance of X is given by $\mathrm{Var}(X) = G''(1)+G'(1)-(G'(1))^2$, where $G''(1) = \frac{d^2G(z)}{dz^2}|_{z=1}$.

Moreover, as shown in [65], the PGF for any given two different states in a MC is the transfer function (TF) of the corresponding signal flow graph (SFG), which can be obtained by taking Z-transform for each transition probability (along with the time spent in the transition) in the MC. Then we can apply the well-known Mason formula to the SFG to derive the PGF. Figure 4.4 shows a MC and its SFG, where the SFG is very similar to the MC, except that each transition probability is replaced by its Z-transform (i.e. $p \cdot z^{\delta}$, where p is the transition probability and δ is the duration of a baseline time slot). Furthermore, by the property of Z-transform, the SFG can be simplified by combining all states with transition probability 1. For example, the states 1 to L can be simplified as a single state LA, and the PGF between A and L (without the loop) becomes $p_1 z^{(L+1)\delta}$. *Note that $z^{(L+1)\delta}$ here actually means the overall transition between A and L uses $L + 1$ time slots, which is accurate.* Now, by the Mason formula, we can obtain the overall PGF from A to L as $G_{AL}(z) = \frac{p_1 z^{(L+1)\delta}}{1-p_1 z^{(L+1)\delta} p_2 z^{\delta}}$.

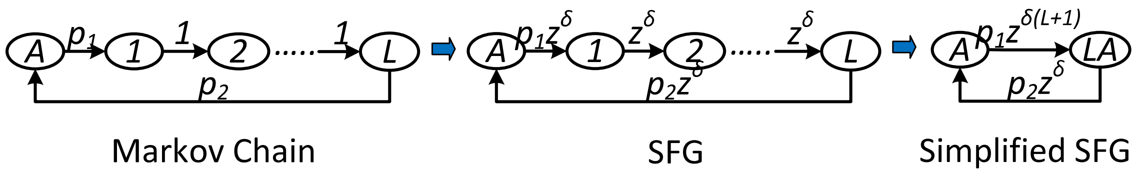


Figure 4.4: Signal Flow Graph of a Markov Chain.

4.1.5.2 BoX-MAC MAC service time characteristics

Let T_B be the r.v. representing the MAC service time of BoX-MAC. As before, to compute $\mathrm{E}(T_B)$, we need to obtain the PGF of T_B, i.e. $BG(z) = \mathrm{E}(z^{T_B})$ by using the SFG of the BoX-MAC MC (Figure 4.2). The SFG is generated by simply adding z to each transition probability in Figure 4.2 (to make the presentation concise, a figure for the SFG is omitted here). *It is worth noting that since T_B is the MAC service time, the PGF of T_B is actually the transfer function between point S and B in Figure 4.2.* Since the entire SFG is a little complex, we first of all derive the PGF between point S and state $(0,0)$ (using the Mason formula) as $BG_{0,0}(z) = \frac{z^{3\delta}}{W_0'} \frac{1-z^{3\delta \cdot W_0'}}{1-z^{3\delta}}$, where 3δ represents a BoX-MAC time slot which is 3 times longer than the baseline (802.11) one. Then since the PGF between $(0,0)$ and $(-1,0)$ is simply $(1-\alpha)z^{\delta}$, the PGF between S and $(0,-1)$ is $BG_{0,-1}(z) = BG_{0,0}(z) \cdot (1-\alpha)z^{\delta}$. Using similar method (note that due to the self loop of C, the derivation is a little harder), we can have the PGF between point C and $(1,-1)$ as $BG_{1,-1}(z)$. Finally by combining all the PGF's of different parts, the overall PGF (between S and B) can be expressed as $BG(z) = BG_{0,0}(z)(1-\alpha)z^{\delta}(1-\beta)z^{3\delta L_{\mathrm{TXB}}} + BG_{0,0}(z)(\alpha z^{\delta} + (1-\alpha)\beta z^{2\delta})BG_{1,-1}(z)(1-\beta)z^{3\delta L_{\mathrm{TXB}}}$.

Then by the property of PGF, we have $1/\mu_B = \mathrm{E}(T_B) = BG'(1)$, thus

$$P_B = 1 - \lambda_B \cdot BG'(1) \tag{4.7}$$

4.1.5.3 802.11 MAC service time characteristics

We define T_W as the r.v. denoting the MAC service time of 802.11. Similar to the analysis for BoX-MAC, here the PGF (i.e. $WG(z) = \mathrm{E}(z^{T_W})$) of T_W is needed to obtain $E(T_W)$. The same method as before is used to convert the MC for 802.11 (in Figure 4.1) to its corresponding SFG. To get the expression for the PGF between point S_0 (or S_1) and B, we also employ the divide-and-conquer strategy, i.e. first expressing the TF's from S_0 (or S_1) to C, then C to D and so on, and finally combining them to generate the overall PGF.

However, due to the increased complexity of the SFG, the derivation for the PGF between S_0 (or S_1) and C is more difficult than that for BoX-MAC. We first of all define the PGFs from S_0 to two states with same backoff counter (e.g. S_0 to $(0, j, 0)$ and to $(0, j, 1)$) as a row vector $\mathbf{WG0_{0,j}}(z)$.Then the PGF from S_0 to $(0, W_0 - 1, 0)$ and to $(0, W_0 - 1, 1)$ (for demonstration purpose) is $\mathbf{WG0_{0,W_0-1}}(z) = \begin{bmatrix} z/W_0, 0 \end{bmatrix}$. From $\mathbf{WG0_{0,W_0-1}}(z)$, $\mathbf{WG0_{0,W_0-2}}(z)$ is also obtained, and so on so forth, until $\mathbf{WG0_{0,0}}(z)$ is derived. Specifically, based on the transitions between $\mathbf{WG0_{0,j}}(z)$ and $\mathbf{WG0_{0,j-1}}(z)$, we define a matrix which describes the TF between them as $\mathbf{Q0} = \begin{bmatrix} (1-P_f)z & a(z) \\ (1-P'_f)z & b(z) \end{bmatrix}$, where $a(z) = A_0C_1z^{\delta L_{\text{STW}}} + A_0(1 - C_0 - C_1)z^{\delta L_{\text{CTW}}} + (1 - A_0 - A_1C_0)z^{3\delta L_{\text{TXB}}}$ and $b(z) = P'_f z^{\delta L_{\text{STW}}} + (P'_f - C_1)z^{\delta L_{\text{CTW}}}$, where A_0, A_1 are defined in Section 4.1.4, and C_0 and C_1 are defined as $(1 - \tau_W)^{N_W-1}$ and $(N_W - 1)\tau_W(1 - \tau_W)^{N_W-2}$, respectively. Although the expressions above look tedious, they are simple Z-transforms of all different transition paths between $\mathbf{WG0_{0,j}}(z)$ and $\mathbf{WG0_{0,j-1}}(z)$. With $\mathbf{Q0}$ (note that $\mathbf{I} - \mathbf{Q0}$ is invertible), we note that $\mathbf{WG0_{0,0}}(z) = \mathbf{WG0_{0,W_0-1}}(z) \cdot (\mathbf{I} + \mathbf{Q0} + \cdots + \mathbf{Q0^{W_0-1}}) = [z/W_0, 0] \cdot \frac{\mathbf{I}-\mathbf{Q0^{W_0}}}{\mathbf{I}-\mathbf{Q0}}$. Since the PGF (denoted by $\mathbf{TG}(z)$) between $[(0, 0, 0), (0, 0, 1)]$ and point C is

$$\mathbf{TG}(z) = \begin{bmatrix} a(z) - A_0C_1z^{\delta L_{\text{STW}}}, b(z) - P'_f z^{\delta L_{\text{STW}}} \end{bmatrix}^T$$

(note $P_c = P_f$ and $P'_c = P'_f$), we define the PGF for backoff stage 0 (represented by $WG0_0(z)$) as the PGF between S_0 and C. We then have $WG0_0(z) = \mathbf{WG0_{0,0}}(z) \cdot \mathbf{TG}(z)$. By similar method, the PGF for backoff stages 1 to $m - 1$ can be derived.

For example, the PGF for stage 1 (i.e. between C and D) is $WG0_1(z) = \mathbf{WG0_{1,0}}(z) \cdot \mathbf{TG}(z)$, where $\mathbf{WG0_{1,0}}(z)$ is the PGF between C and $[(1,0,0),(1,0,1)]$ and $\mathbf{TG}(z)$ is the PGF from $[(1,0,0),(1,0,1)]$ to D. Note that $\mathbf{TG}(z)$ does not change with backoff stage.

Same as BoX-MAC, the PGF for stage m is a little tricky because of the self-loop. The derivation, however, is still trivial by using the Mason formula. The corresponding result is as

$$\mathbf{WG0_{m,0}}(z) = \left[\frac{\mathbf{WG0'_{m,0}}^{(1)}(z)}{1-\mathbf{WG0'_{m,0}}^{(1)}(z)\cdot\mathbf{TG}^{(1)}(z)}, \frac{\mathbf{WG0'_{m,0}}^{(2)}(z)}{1-\mathbf{WG0'_{m,0}}^{(2)}(z)\cdot\mathbf{TG}^{(2)}(z)}\right]$$

where the superscript (i) represent the ith element in a vector, and $\mathbf{WG0'_{m,0}}$ is the forward PGF (i.e. without the self-loop) from point E to $[(m,0,0),(m,0,1)]$.

Having had the PGF's for all stages, the final PGF from S_0 to B can be derived using Mason formula as below

$$\begin{aligned} WG0(z) &= \mathbf{WG0_{0,0}}(z)\cdot\left[(1-P_f)z^{\delta L_{\mathrm{STW}}},(1-P'_f)z^{\delta L_{\mathrm{STW}}}\right]^T \\ &+\sum_{i=1}^{m}\left(\mathbf{WG0_{i,0}}(z)\cdot\left[(1-P_f)z^{\delta L_{\mathrm{STW}}},(1-P'_f)z^{\delta L_{\mathrm{STW}}}\right]^T\cdot\prod_{j=0}^{i-1}WG0_j(z)\right)\end{aligned}$$

By similar derivation as above, the PGF from S_1 to B (i.e. $WG1(z)$) is obtained. Then by the property of PGF, we have

$$\begin{aligned} P_0 &= \frac{b_{IDLE,0}\cdot(1-P_f)+b_{IDLE,1}\cdot(1-P'_f)}{b_{IDLE}} \\ P_1 &= \frac{b_{IDLE,0}\cdot P_f + b_{IDLE,1}\cdot P'_f}{b_{IDLE}} \\ P_W &= 1-\lambda_W\cdot(P_0\cdot WG0'(1)+P_1\cdot WG1'(1)) \qquad (4.8)\end{aligned}$$

where P_0 and P_1 denote the probabilities that the outputs of $IDLE$ state being S_0 and S_1, respectively.

Now, by employing numerical methods to solve Equations (4.4)-(4.8), all eight unknown probabilities are obtained. Several performance metrics can be derived from these probabilities. In this dissertation, we are interested in the aggregate throughput and the total delay, i.e., the time between a packet entering the queue until it is transmitted successfully.

4.2 Performance Prediction of UNSAT

4.2.1 Throughput Analysis

With the help of the Markov Chain model for the channel shown in Section 4.1.4, it is straightforward to derive the expressions for the throughput of 802.11 and BoX-MAC.

4.2.1.1 Throughput for 802.11

Similar to BoX-MAC, we can express the 802.11 throughput as

$$S_W = (b_0 P_{0STW} + b_1 P_{1STW}) Lp_{\mathrm{STW}} \tag{4.9}$$

where b_1 is the stationary probability for state 1 in the channel MC (Section 4.1.4), P_{0STW} is the transition probability for an 802.11 transmission being succeed, and Lp_{STW} is packet payload size of 802.11 in baseline time slot.

4.2.1.2 Throughput for BoX-MAC

From the perspective of the channel, the throughput of BoX-MAC is simply the time spent in a successful BoX-MAC transmission, thus we have

$$S_B = b_0 P_{0STB} 3Lp_{\mathrm{TXB}}$$

where b_0 is the stationary probability for state 0 in the channel MC (Section 4.1.4), P_{0STB} is the transition probability for a BoX-MAC transmission being succeed, and $3Lp_{\mathrm{TXB}}$ is packet payload size of BoX-MAC in baseline time slot.

4.2.2 Delay Analysis

By Little's Law, delay $D = L/\lambda$, where L is the steady state queue length and λ is the arrival rate. Consequently, to compute the delay we need to obtain the steady state queue length of the BoX-MAC and 802.11. For a $M/G/1$ queue, the expression for the average queue length is $L = \rho + \frac{\lambda^2(\sigma^2+1/\mu^2)}{2(1-\rho)}$ [94], where σ^2 is the variance of the service time, and $\rho = \lambda/\mu$.

Since we obtain the PGF's of the MAC service time of BoX-MAC and 802.11 in Section 4.1.5.2 and 4.1.5.3, the corresponding variances are

$$\sigma_B^2 = \mathrm{Var}(T_B) = BG''(1) + BG'(1) - (BG'(1))^2$$
$$\sigma_W^2 = \mathrm{Var}(T_W) = WG''(1) + WG'(1) - (WG'(1))^2$$

Then the expressions for the queue length are

$$L_B = \rho + \frac{\lambda_B^2(\sigma_B^2 + 1/\mu_B^2)}{2(1-\rho_B)}$$
$$L_W = \rho + \frac{\lambda_W^2(\sigma_W^2 + 1/\mu_W^2)}{2(1-\rho_W)}$$

With average MAC service time, we can simply write the expressions for delay (by Little's Law) as $D_W = L_W/\lambda_W$ and,

$$D_B = L_B/\lambda_B \tag{4.10}$$

4.3 Performance Tuning for UNSAT

4.3.1 Parameters Tuning for Delay Constraints

In this section we present a protocol tuning method that maximizes 802.11 throughput while satisfying delay constraints of 802.15.4, by varying the CW size of individual nodes. CW affects throughput since it directly controls the transmission probability (i.e. aggressiveness) [82]. As previously demonstrated in previous chapter, the congested CW size of BoX-MAC (W_1') and the minimum CW size of 802.11 (W_0) have a significant impact on the achievable throughput in coexisting networks, while the initial CW size of BoX-MAC (W_0') and the maximum CW size of 802.11 (W_m) do not. Thus, we only consider tuning W_1' and W_0, and fix the other two. Our tuning method also adjusts the WiFi data arrival rate λ_W since it is the only way to tune the throughput under unsaturated condition.

Since we want to ensure that the performance (i.e. throughput) of WiFi devices be maximized under the delay constraint of BoX-MAC, we formulate this as a non-linear optimization problem,

$$\begin{aligned} \underset{W_0, W_1', \lambda_W}{\arg\max} \quad & S_W \\ \text{subject to} \quad & D_B \leq \text{Delay Threshold}, \\ & W_1', W_0, L_B, L_W \geq 0 \end{aligned}$$

where the throughput of 802.11 S_W (see Equation (4.9)) is the objective function, while the delay threshold of BoX-MAC (see Equation (4.10) for D_B) and Equations (4.4)-(4.8) serve as the main constraints. More importantly, the CWs and queue lengths (i.e. L_W, L_B) for all nodes must be greater than or equal to 0, so that the queue of each node is stable (i.e. arrival rate $<$ service rate). By solving this problem numerically (e.g. *fmincon* toolbox in MATLAB), we obtain the optimal CW sizes for BoX-MAC and optimal CW's and arrival rates λ_W for WiFi. This tuning method is centralized, and due to the homogeneity of devices of same type, the optimal parameters for same type are equal. Thus this method does not exhibit fairness issues for the same type of devices. We note that ensuring fairness between BoX-MAC and 802.11 is not appropriate because of the very large difference in their maximal flow rates.

4.4 ns-3 based Simulator for UNSAT

4.4.1 Simulator Design

We implemented our simulator for single-cell coexistence based on the well-known ns-3 simulator, which is a discrete-event network simulator. ns-3 has been developed to provide an open, extensible network simulation platform, which is easy for developers to implement their own protocols, especially when the protocols are on the high IOS layer (e.g. application layer).

The spectrum PHY module in ns-3 is used as the common operating channel for both 802.11 and BoX-MAC devices. Since our simulator is focused on MAC layer, the implementation involved significant modifications at MAC layer of the WiFi and LrWPAN modules of ns-3, to handle homogeneous and heterogeneous collisions. In addition, the network and transportation layers are removed to accelerate the simulation speed. Furthermore, an unsaturated traffic generators are added to the application layers of WiFI and ZigBee to enable the simulation for saturation.

4.5 Model and Tuning Evaluation

4.5.1 Validation of Model for UNSAT

To valide the proposed model for UNSAT, we compare the throughput and delay obtained from the model with those obtained from the ns-3 simulator.

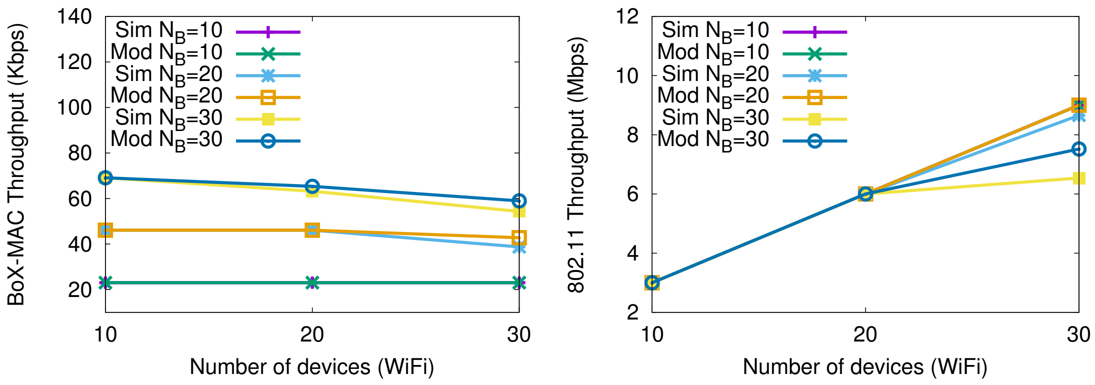


Figure 4.5: Throughput of model and simulator vs # devices.

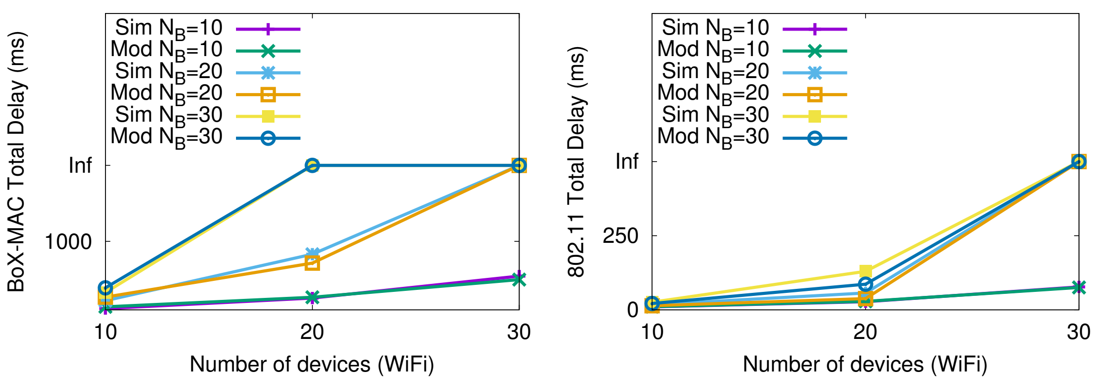


Figure 4.6: Total delay of model and simulator vs # devices.

Extensive simulations were performed under varying configuration parameters in order to characterize the effects of CW size tuning on aggregate throughput and total delay. The parameters we vary are: number of nodes, the minimum contention window size W_0 of 802.11, the congested contention window size W_1' of BoX-MAC and the corresponding per-node offered load (data arrival rate) λ_W and λ_B. The default values of these parameters are: $W_0 = 16$, $W_m = 1024$, $P_W = 1500$, $\lambda_W = 20$ for WiFi, and $W_0' = 310$, $W_1' = 70$, $P_B = 48$, $\lambda_B = 4$ for BoX-MAC.

Typically, the time spent for the model to numerically converge is less than one minute on a fast PC (with i7 4790K CPU and 16GB ram), while solving the optimization problem takes a little longer than that. The simulation time for all tests is 10000s, which, on average, takes more than ten hours on the same machine.

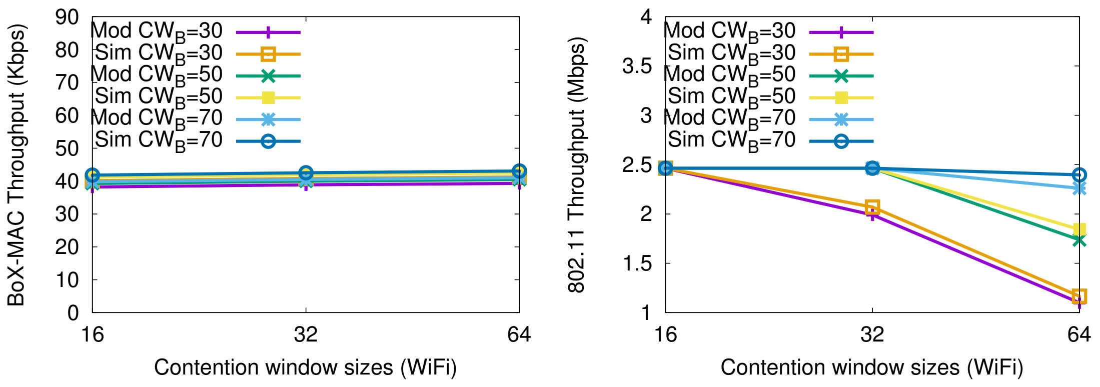


Figure 4.7: Throughput of model and simulator vs CW sizes.

4.5.1.1 Effects of number of devices

Analyzing the coexistence of WiFi and WSN by varying the number of devices in a real implementation is tedious and time consuming. The same can be done by

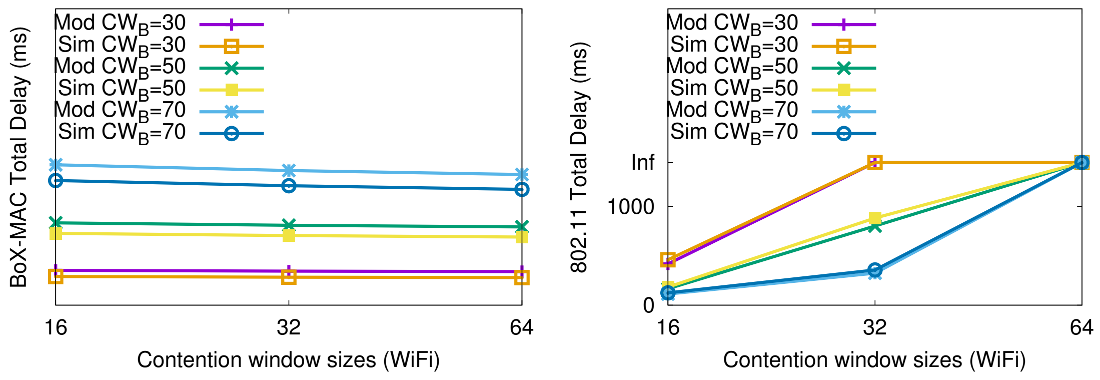


Figure 4.8: Total delay of model and simulator vs CW sizes.

merely varying the number of devices in the analytical model and simulator. Figure 4.5 and Figure 4.6 depict the effect on throughput and delay when the number of WiFi (i.e., $\{5, 10, 20, 40\}$) and BoX-MAC (i.e., $\{10, 20, 40, 80\}$) devices are varied, while setting other parameters to their defaults. We surprisingly observe that the results for the simulator and analysis match each other closely. Specifically, when the network density is small (e.g. 20 BoX-MAC vs 10 WiFi), the throughput of both types increase linearly with the increase of the number of devices due to unsaturated queueing condition. For a crowded network (e.g. 80 BoX-MAC vs 40 WiFi), the BoX-MAC reach near-saturated condition (we have chosen $\lambda_B = 4$ on purpose to demonstrate), and the 802.11 devices reach saturated condition. Thus, we observe that the throughput of BoX-MAC still increase linearly, however, the one for 802.11 increases slowly due to saturation. The delay exhibits similar behavior as the throughput. Especially when the number of devices becomes large, the delay of BoX-MAC becomes large due to near-saturated condition, while the queue for 802.11 tends to infinity (denoted by “Inf” in the figure) due to saturation.

4.5.1.2 Effects of contention window sizes

To analyze the impact of contention window size on throughput and delay, we consider a scenario with 10 802.11 and 20 BoX-MAC devices. All default parameters are used expect the CW size, which are selected within $\{30, 50, 70\}$ and $\{16, 32, 64\}$ for BoX-MAC and 802.11, respectively. The results are depicted in Figure 4.7 and Figure 4.8 for throughput and delay, respectively. Remarkably, the results obtained from the model agree with the simulation quite well. In the case of throughput, since the CW's for BoX-MAC always lead to unsaturated condition, its throughput does not vary much. While for 802.11, since a smaller CW (such as $W_0 = 16$) implies higher aggressiveness, the queue is unsaturated, and the throughput of 802.11 equals to its input. However, when the CW of 802.11 increases, due to the decrease of the aggressiveness, its queue becomes sensitive to the aggressiveness of BoX-MAC devices. For example, if CW of BoX-MAC equals 70, $W_0 = 64$ lead to saturation, but if CW of BoX-MAC equals 30 or 50, $W_0 = 32$ tends towards saturation. For the delay analysis, since bigger CW results in longer MAC service time, the delay is higher for bigger CW. As mentioned before, the queue of 802.11 becomes saturated for some cases, hence the corresponding delay is infinite.

4.5.1.3 Effects of per-node offered load

Due to the significant impact of traffic arrival rate, i.e per-node offered load, on the network performance, we study its effect on the throughput. As earlier, we consider a scenario with 40 WiFi, 40 BoX-MAC devices and default parameters. We vary the per-node offered load in $\{2, 4, 6\}$ packets per second for Box-MAC and $\{1, 10, 20, 30\}$ packets per second for WiFi. The results are depicted in Figure 4.9 and Figure 4.10 for throughput and total delay, respectively. Amazingly, the simulation results are in close agreement with those obtained from the model. In particular,

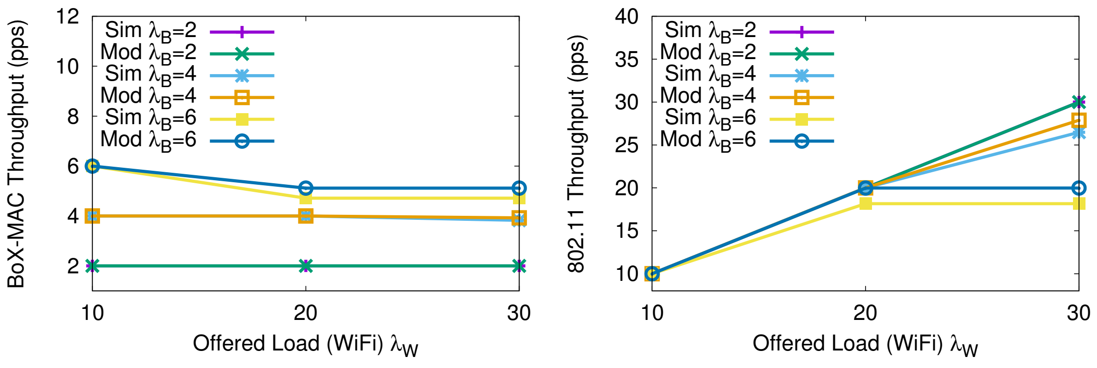


Figure 4.9: Throughput of model and simulator vs offered load.

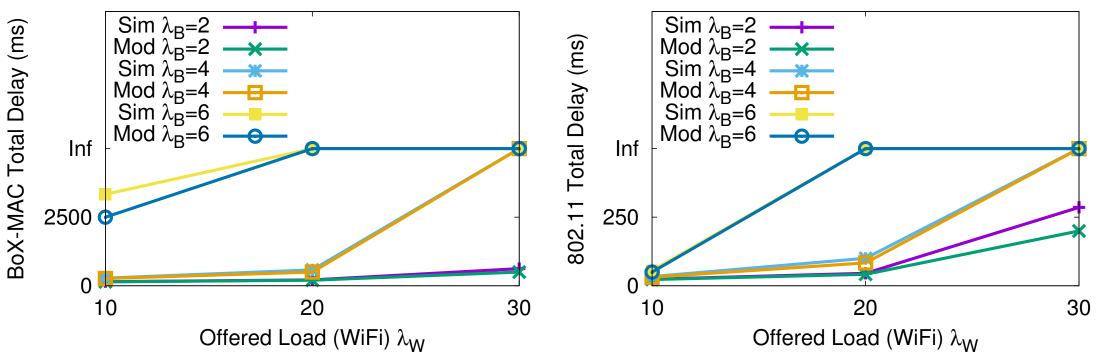


Figure 4.10: Total delay of model and simulator vs offered load.

when the offered load of WiFi increases, until saturation, the throughput of both BoX-MAC and WiFi increase linearly. Then, their throughput gradually and slowly decreases due to the increased collision rate. Similar results can be observed for the delay. As the queue becomes saturated, the delay tends toward infinity.

4.5.2 Parameters Tuning Evaluation

We evaluate the tuning method proposed in Section 4.3.1 using our ns-3 simulator. We demonstrate that our approach guarantees the delay constrains of BoX-MAC

Table 4.1: Optimal parameter values obtained under $\lambda_B = 2$

Delay constraint	λ_W (**802.11**)	W_0 (**802.11**)	W_1' (**BoX-MAC**)
50 ms	53	8.1748	13.2673
100 ms	78	5.1457	21.3654

Table 4.2: Optimal parameter values obtained under $\lambda_B = 4$

Delay constraint	λ_W (**802.11**)	W_0 (**802.11**)	W_1' (**BoX-MAC**)
50 ms	42	14.6472	6.7848
100 ms	60	9.2516	10.8520

while maximizing 802.11 throughput. The evaluation of the tuning method is performed upon 20 BoX-MAC and 10 802.11 nodes, with different BoX-MAC delay constraints among $\{50, 100\}$ms.

We inspect three cases of the arrival rates λ_B for BoX-MAC from $\{2, 4, 6\}$, and obtain from the model the optimal CW sizes for BoX-MAC (i.e. W_1') and 802.11 (i.e. W_0) as well as the optimal arrival rates λ_W for 802.11. As shown in Table 4.1, when the delay constraint for BoX-MAC is 100ms, the optimal CWs obtained from the tuning method are significantly different from their default values (i.e. $W_0 = 16$ and $W_1' = 70$). And since λ_B is only 2, it is easy to satisfy the delay constraint, which offers WiFi a very good available throughput (which is = 78). However when delay constraint becomes 50 ms, since it is harder to satisfy the constraint, the BoX-MAC becomes more aggressive by reducing its W_1', and WiFi increase the W_0 to further give space for BoX-MAC. Therefore the maximal throughput of WiFi λ_W decreases to 53. Similarly, when $\lambda_B = 4$ or 6, the same trends of W_1', W_0 and λ_W can be observed in Table 4.2 and 4.3. Especially, for an extreme case where $\lambda_B = 6$ and delay constraint equals 50, the BoX-MAC becomes very aggressive (i.e. $W_1' = 3.1837$), while WiFi

Table 4.3: Optimal parameter values obtained under $\lambda_B = 6$

Delay constraint	λ_W (**802.11**)	W_0 (**802.11**)	W_1' (**BoX-MAC**)
50 ms	27	20.3774	3.1837
100 ms	46	13.2743	7.2562

reduce its aggressiveness by increasing its W_0 to 20.3774 and thus has a throughput of only 27.

In practice, this tuning method can be used on a dual radio powerful master device which first computes the optimal parameters and then informs both WiFi and BoX-MAC devices about the values obtained.

5. COEXISTENCE OF DUTY-CYCLING WIFI AND ZIGBEE WITH UNSATURATED TRAFFIC

In this chapter a mathematical model and a simulator for symmetric coexistence of duty-cycling WiFi and ZigBee with unsaturated traffic (i.e. DC-UNSAT) are discussed. Since the difficulty of modeling the DC-UNSAT problem is quite high, we actually tackle it through two steps, i.e. first modeling the WiFi power saving mode (PSM) to get an idea, and then solve the whole problem of DC-UNSAT. It is worth emphasizing that, after modeling the PSM, we notice that it is too complicated to solve the DC-UNSAT problem by using the same manner for modeling PSM, a different type of approach is employed eventually, i.e. an approximation based approach. However, we think it is still valuable to present the modeling for PSM, which is in Section 5.2.

5.1 Background

5.1.1 WiFi 802.11 PSM Protocol

The Ad-Hoc PSM (namely IBSS) is defined in IEEE 802.11 standard [88]. Similar to infrastructural WiFi network, all devices in IBSS PSM are time synchronized with a centralized beacon node. As shown in Figure 5.1, each cycle is called a Beacon Interval (BI), and each BI has two fixed length windows, i.e. Announcement Traffic Indication Message (ATIM) window and DATA window. Every node wakes up at the beginning of a cycle, and after hearing a beacon, it stays awake for ATIM long to listen to or transmit an ATIM packet. The ATIM packet is a control frame exchanged by devices to determine the behavior in the following DATA window. Specifically, when a device has data for a receiver, it transmits an ATIM packet to the receiver during ATIM using the 802.11 DCF mechanism (i.e. randomly backoff).

In response to an ATIM packet, the receiver will reply an ATIM-ACK. After a successful ATIM handshake, both devices will be in power-on mode in the following DATA window, where the actual packet transmission takes place. If a device fails to send an ATIM frame within the ATIM window, the data frame is buffered and another attempt will be made during the next BI cycle. A device enters the power save mode at the end of the ATIM window if it has no data to transmit/receive, or the transmission/reception in ATIM window is failed. *To make the modeling tractable, we assume that a transmitter which succeeds in ATIM window can only send one packet in the following DATA window.*

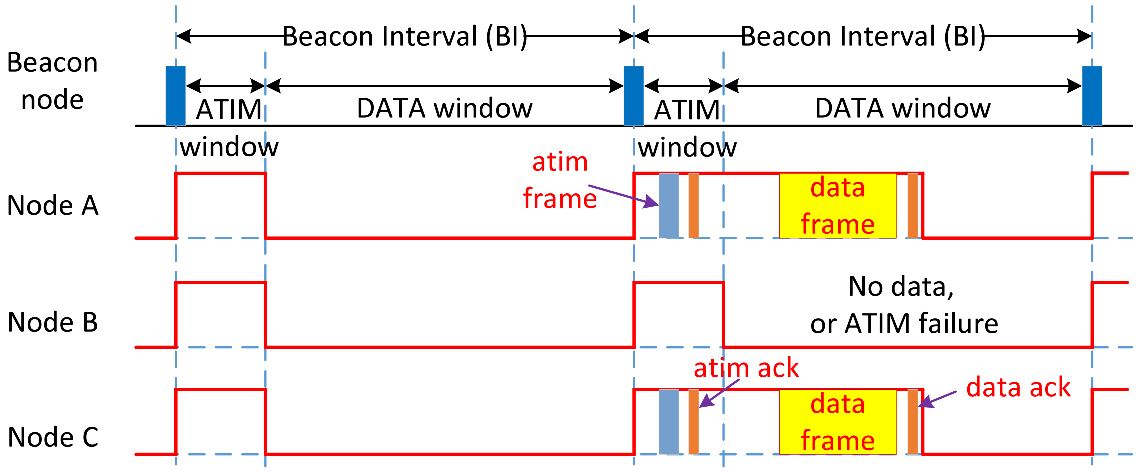


Figure 5.1: 802.11 IBSS PSM Protocol

5.1.2 ZigBee BoX-MAC LPL Protocol

BoX-MAC low power listening (LPL) is considered in this dissertation because it is the most widely used duty cycling protocol for WSNs. Figure 5.1 demonstrate how LPL works. Specifically, a receiver (e.g. node B or C here) wakes up periodically to check if there is a data for it, and the duty cycle ratio determines its awake duration in each cycle. Regardless of whether the reception is successful or not, the receiver immediately goes back to sleep whenever its awake time is exhausted. While for a sender (e.g. node A), it wakes up right away if a packet needs to be transmitted, and then it repeatedly sends the packet (i.e. retries) and wait for the reply from the corresponding receiver, until it succeeds or the entire cycle is used up. Since a receiver has same cycle length as the sender, it always has chance to receive that packet. If the transmission succeed, the sender goes to sleep when it receives the reply; otherwise the packet is dropped after it is retried for one cycle. Note that each retrial is still based on 802.15.4 (but for simplicity, it has only two backoff stages), i.e. a sender backs off first, then checks channel (namely clear channel assessment (CCA)) and sends data only if the channel is idle.

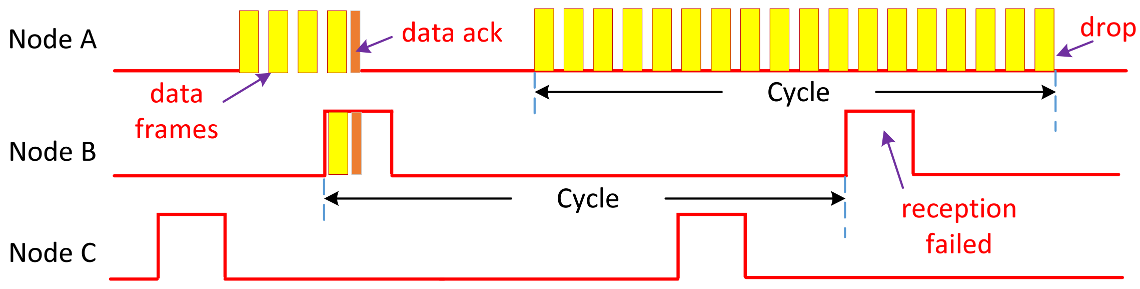


Figure 5.2: LPL Protocol

5.2 Mathematical Model for 802.11 PSM

In this section, we introduce a four dimensional Markov Chain to model the 802.11 IBSS PSM standard.

5.2.1 Key Assumptions

We assume that all devices employ *duty cycling mechanism* to save energy, and are within a single wireless cell (i.e. all devices are mutually reachable). For all devices, the *packets sizes d are the same*, and the *data arrival rates λ are equal.* For simplicity, we also consider an *ideal channel* (i.e. no shadowing, fading and capture effect), implying communication failure is only due to collisions. Moreover, we make two fundamental, yet widely used assumptions: a) the probability τ of each transmission attempt is constant and independent of other attempts; and b) the conditional collision probability given an attempt is constant and independent of the collision history. These assumptions were proven to be accurate for both saturated and unsaturated scenarios [48]- [93].

5.2.2 Markov Chain Model

Now we describe the Markov Chain model and its analysis.

5.2.2.1 The Per-node Markov Chain

In the original Discrete Time Markov Chain model of the reference node (or MC for simplicity) for 802.11 DCF [48], each state is two dimensional $(s(t), c(t))$, where $s(t)$ is a stochastic process representing the current backoff stage ($0 \leq s(t) \leq m$, where m is the maximum stage), and $c(t)$ is a process representing the current backoff counter in the corresponding backoff stage ($0 \leq c(t) \leq CW(i)$, where $CW(i)$ is the contention window size of stage i). This Markov Chain maintains the Markovian property because given a state $(s(t), c(t))$, the following state i.e. $(s(t+1), c(t+1))$ is independent of the state before t. *Notably, henceforth the terms "backoff stage" and "stage" are used interchangeably.*

However, for the 802.11 IBSS PSM protocol, because of the two windows (i.e.

ATIM window and DATA window), both $s(t)$ and $c(t)$ will be reset to 0 whenever the window is used up. Thus given a state $(s(t), c(t))$, it is not sufficient to determine whether the next state is $(s(t+1), c(t+1))$ or $(0, 0)$ (i.e. when window is used up), which implies that the Markovian property cannot be maintained with only two dimensions. To solve this problem, we introduce a new stochastic process $w(t)$ to denote the time elapsed in the corresponding ATIM/DATA window. Then we have $0 \leq w(t) \leq \overline{A}$ (or $\overline{D}$), where $\overline{A}$ and $\overline{D}$ are the lengths of a ATIM and a DATA window, respectively. Particularly, $w(t) = 0$ in the beginning of a window, and increments at the end of each time slot until $\overline{A}$ (or $\overline{D}$) (i.e. when the window is exhausted).

According to the IBSS PSM protocol, a node can send at most one ATIM packet in an ATIM window, and based on our simplification of the protocol, within one DATA window, a node can send at most one data packet. This property is important because after a successful transmission (within a window) a node will not contend for the channel in that window, which implies that the number of contending (active) nodes may reduce. Since the probability that a node senses a busy channel depends on the number of active devices in a window (which is a variable), we bring in another stochastic process $n(t)$ which represents the number of active nodes in an ATIM/DATA window (thus $0 \leq n(t) \leq N$, where N is the total number of devices).

Therefore, the entire Markov Chain is four dimensional, and each state can be denoted as $(s(t), c(t), w(t), n(t))$. *For simplicity, we name each state as **(stage counter, backoff counter, window counter, node counter)***. Since this 4-D Markov Chain is fairly complex, we divide the entire chain into several components, which are shown in Figure 5.3.

Big picture of the MC: Each BI includes an ATIM window (represented by the dashed-line box in Figure 5.3) and a DATA window (represented below the dashed-line box). The reference node enters an $ATIM$ component at the beginning of the

BI if it has data to transmit (with probability $1-p_0$, where p_0 is the probability of an empty queue). If the handshake in the ATIM is successful, the node enters the SA component (stands for success in the ATIM window) to wait until the ATIM window ends, and then enters a $DATA$ component. While if it fails in the ATIM, the node enters the EW component (stands for waiting till BI ends). If there is success in the DATA window (i.e. the $DATA$ component), the node enters the SD component (stands for success in the DATA window). If the transmission fails in DATA, the node immediately returns to an $ATIM$ component to resend the data. The $ATIM$ (along with the corresponding SA and EW) and $DATA$ (with the corresponding SD) components have N similar regions representing different number of active devices at the beginning of the corresponding window and are indexed with the symbol X. For example, $DATA_X$ means that the node is in a DATA window with X active nodes at the beginning of the window.

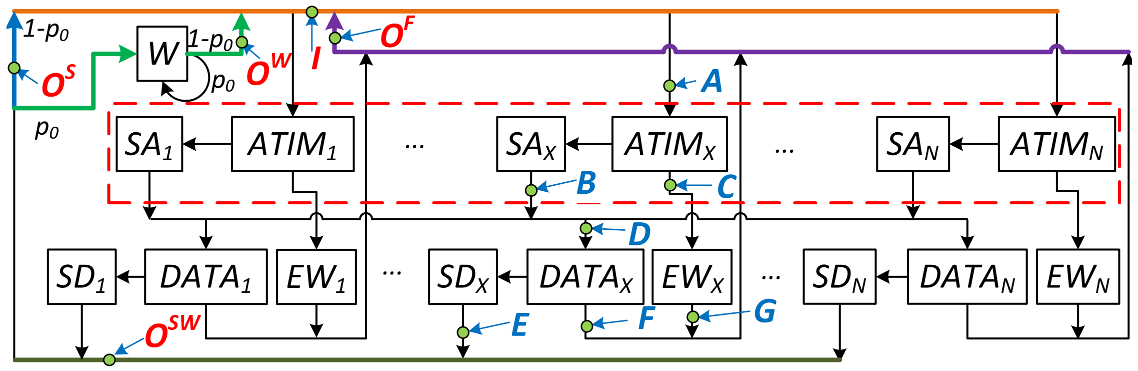


Figure 5.3: Big picture of Markov Chain for IBSS PSM.

Before exploring each component in detail, we first introduce our main idea of solving the MC, i.e., finding the stationary probability of each state. Due to the complexity of the MC, there is no simple closed form expression, we therefore adopt a numerical method. As we know, a numerical method usually employs an iterative approach, until the objective converges. We initially make a guess of the stationary probabilities (denoted as an input vector $\mathbf{I} = [I_1, I_2, \cdots, I_N]$) for the input of each component $ATIM_X$. Then, by traversing the MC, we get a high level transition probability matrix (denoted by $\mathbf{P}$) from an old input vector to a new one (due to the closed loop property), thus we have $\mathbf{I}' = \mathbf{I} \cdot \mathbf{P}$. Finally, by iteratively updating $\mathbf{I}$, we will obtain a stationary vector (i.e., $\mathbf{I} \cdot \mathbf{P} \sim \mathbf{I}$). We therefore need to find $\mathbf{P}$, which we will briefly discuss here. For ease of explanation, in Figure 5.3, we mark probabilities of some states such as points A (i.e. the input probability of $ATIM_X$), B (i.e. the output probability of SA_X) etc. Through detailed analysis of the MC (discussed later), the transition probability P_{AB} between A and B are obtained (P_{AB} is a vector, as shown later, and is denoted as $P_{AB} = \mathbf{PSA_X}$). Similarly, P_{AC} (as $\mathbf{PFA_X}$), P_{AG} (as $\mathbf{PEW_X}$), P_{DE} (as $\mathbf{PSD_X}$) and P_{DF} (as $\mathbf{PFD_X}$) are derived accordingly. In the following sections, we will cover the details of the MC to compute these high level transition probability vectors, and later, we will discuss the method to compute $\mathbf{P}$.

The details of $ATIM_X/DATA_X$: Figure 5.4 shows the $ATIM_X/DATA_X$ component, where the backing off procedure takes place (recall that X represents the number of active nodes at the beginning of ATIM/DATA window). We present the $ATIM_X$ and $DATA_X$ components together because of their similarity in behavior. To make the presentation concise, we only show the first stage (i.e. stage 0) of the total m stages, and we assume that its input probability is i_{Y_X} (i.e. i_{A_X} or i_{D_X} for ATIM window or DATA window). The part above the big down-arrow shows the high-level concept of the MC for stage 0, which is very similar to the one for 802.11 DCF [48], but with two major differences: (i) each state in this MC has two outputs, i.e. one (the horizontal left-arrow) to its normal next state (same as the MC in [48]),

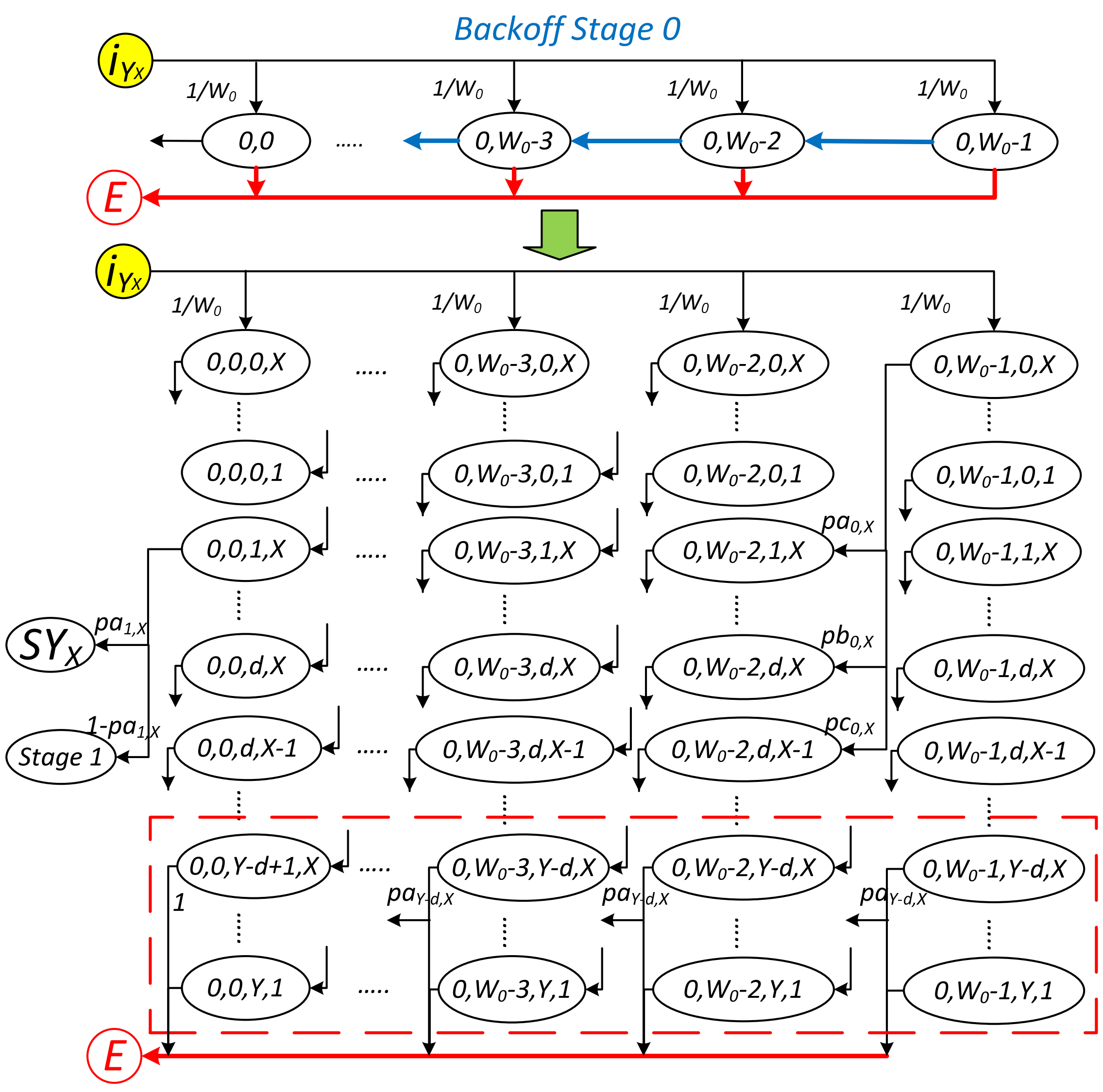


Figure 5.4: Detailed MC for $ATIM_X/DATA_X$.

and the other one (i.e. the vertical down-arrow) to component E ($E = EW_X$ if the MC is for $ATIM_X$; if for $DATA_X$, E corresponds to the end of DATA window, i.e. the end of current BI cycle); (ii) in this MC, each state (i, j) is a set of $\overline{Y} \cdot X$ states, i.e. $(i, j, 0, X), (i, j, 0, X - 1), \cdots, (i, j, \overline{Y}, X), \ldots, (i, j, \overline{Y}, 1)$, where $\overline{Y} = \overline{A}$ or $\overline{D}$, which are the lengths of the ATIM or DATA window.

The detailed MC of stage 0 is shown below the big down-arrow of Figure 5.4, and for each state (i, j, k, n) (*not including the states on the leftmost column and the states in the dashed-line box*), there are three possible transitions: when the channel is idle, the chain transits to state $(i, j-1, k+1, n)$ w.p. $pa_{k,n}$ because with the passage of one time slot, the backoff counter will decrement and the window counter will increment (i.e. $j \to j-1$ and $k \to k+1$); when channel contains a successful transmission, the backoff counter has to freeze during the transmission, the window counter will increases by d since the transmission takes d time slots, and the node counter will decrement because the node which has succeeded in the transmission will stay silent in this window, thus the MC has to enter state $(i, j-1, k+d, n-1)$ w.p. $pb_{k,n}$; and when there is a failed transmission due to collision, the backoff counter and the window counter have same results as the previous case, but because of the failure, the node counter will not change (i.e. to state $(i, j-1, k+d, n)$ w.p. $pc_{k,n}$). To represent a MC for stage i, we define a $((\overline{Y} \cdot X) \cdot W_i)$ matrix $\mathbf{Y_{X_i}}$ (i.e. $\mathbf{A_{X_i}}$ or $\mathbf{D_{X_i}}$), where W_i is the contention window size for backoff stage i. To clarify the idea behind $\mathbf{Y_{X_i}}$, we have collapsed the last two dimensions (i.e. window counter $w(t)$ and node counter $n(t)$) of the MC into one. Thus the element $\mathbf{Y_{X_i}}(k \cdot X + n, j)$ corresponds to the state (i, j, k, n) in the MC.

According to the behavior described above, the three conditional probabilities of the reference node (i.e. $pa_{k,n}$, $pb_{k,n}$ and $pc_{k,n}$) can simply be defined as $pa_{k,n} = (1-\tau_{k,n})^{n-1}$, $pb_{k,n} = (n-1)\tau_{k,n}(1-\tau_{k,n})^{n-2}$ and $pc_{k,n} = 1 - pa_{k,n} - pb_{k,n}$ where $\tau_{k,n}$ is the transmission attempt probability of any other node. *From these definitions, we can easily see that $pa_{k,n}$, $pb_{k,n}$ and $pc_{k,n}$ depend on the node counter n and $\tau_{k,n}$, and since $\tau_{k,n}$ depends on the window counter k and node counter n (which we will explain later), $pa_{k,n}$, $pb_{k,n}$ and $pc_{k,n}$ depend on k and n.* To compute $pa_{k,n}$, $pb_{k,n}$ and $pc_{k,n}$, we need to obtain $\tau_{k,n}$ first. Because of the complexity of the MC, there is no simple close form expression for $\tau_{k,n}$, we therefore resort to a constructive manner method.

As before, we have defined $\mathbf{Y_{X_i}}$, thus $\mathbf{Y_{X_0}}$ of backoff stage 0 can be initialized as,

$$\mathbf{Y_{X_0}} = \begin{bmatrix} \frac{i_{Y_X}}{W_0} & \frac{i_{Y_X}}{W_0} & \cdots & \frac{i_{Y_X}}{W_0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}, \mathbf{Y} \text{ is } \mathbf{A} \text{ or } \mathbf{D} \tag{5.1}$$

where the first row (i.e., state (0, j, 0, X)) is initialized by the input i_{Y_X}/W_0, and all other elements are 0.

As mentioned before, since each state (i, j, k, n) has three outcomes (when $k < \overline{Y} - d$), the corresponding matrix entry $\mathbf{Y_{X_i}}(k \cdot X + n, j)$ will be used to update three other elements as follows (recall that d is the packet size in time slots),

$$\begin{cases} \mathbf{Y_{X_i}}(k \cdot X + n + X, j-1) += \; pa_{k,n} \cdot \mathbf{Y_{X_i}}(k \cdot X + n, j) \\ \mathbf{Y_{X_i}}(k \cdot X + n + d \cdot X, j-1) += \; pb_{k,n} \cdot \mathbf{Y_{X_i}}(k \cdot X + n, j) \\ \mathbf{Y_{X_i}}(k \cdot X + n + d \cdot X + 1, j-1) += \; pc_{k,n} \cdot \mathbf{Y_{X_i}}(k \cdot X + n, j) \end{cases} \tag{5.2}$$

We now consider the states on the leftmost column corresponds to the transmission attempt of the reference node. If the attempt fails (w.p. $1 - pa_{k,n}$), the chain will go to the next backoff stage, thus the update rule between stage i and $i+1$ (i.e., transition from state $(i, 0, k, n)$ to states $(i+1, 0 : W_{i+1} - 1, k+d, n)$ is as,

$$\begin{aligned} &\mathbf{Y_{X_{i+1}}}(k \cdot X + n + d \cdot X, 1 : W_{i+1}) = \\ &\qquad \frac{1 - pa_{k,n}}{W_{i+1}} \mathbf{Y_{X_i}}(k \cdot X + n, 1), \quad k \leq \overline{Y} - d, \mathbf{Y} \text{ is } \mathbf{A} \text{ or } \mathbf{D} \end{aligned} \tag{5.3}$$

where the symbol "$1 : W_{i+1}$" denotes all columns in $\mathbf{Y_{X_{i+1}}}$.

All devices in the network are time synchronized, and their per-node MC's have knowledge of the current number of active devices. Thus given a specific window counter and node counter (e.g. (k, n)), i.e. when specifying a row (denoted as

$R(k,n)$) in the MC, the state of any other node must be on the same row. Therefore, the way to compute $\tau_{k,n}$ becomes simple because according to the definition, given a state (e.g. (i,j,k,n)) of a reference node, the transmission attempt probability of any other node equals the probability that such node lies in state $(i,0,k,n)$, i.e. the leftmost elements of all m stages on the row $R(k,n)$. From this, we get $\tau_{k,n}$ as,

$$\tau_{k,n} = \frac{\sum_{u=0}^{m} \mathbf{Y_{X_u}}(k \cdot X + n, 1)}{\sum_{u=0}^{m} \sum_{v=1}^{W_u} \mathbf{Y_{X_u}}(k \cdot X + n, v)}, \ \mathbf{Y} \text{ is } \mathbf{A} \text{ or } \mathbf{D} \tag{5.4}$$

Now we have a method to calculate $\tau_{k,n}$ in terms of $\mathbf{Y_X}$'s. However, Equations (5.2) and (5.3) reveal that generating $\mathbf{Y_X}$ depends on $\tau_{k,n}$ (because $pa_{k,n}$, $pb_{k,n}$ and $pc_{k,n}$ are needed). By Equation (5.1), we observe that the first row of the matrix $\mathbf{Y_{X_0}}$ is known, thus by Equation (5.4) we can compute $\tau_{0,X}$ for the first row. Then by using Equations (5.2) and (5.3), we obtain three other rows for $\mathbf{Y_{X_0}}$ and two rows for the matrix of $\mathbf{Y_{X_1}}$. Since the rows in the matrix are ordered by the window counter (i.e. time elapsed from 0 to Y), by iteratively updating rows from top down (i.e. row in $\mathbf{Y_{X_i}}$ updates another three rows in $\mathbf{Y_{X_i}}$ and two rows in $\mathbf{Y_{X_{i+1}}}$), all $\tau_{k,n}$'s and all entries of the matrices $\mathbf{Y_X}$'s can be attained.

Since we have modeled $ATIM_X$ and $DATA_X$, we are ready to derive the output of them (i.e. the inputs for SY_X and EW_X). As we know, if the reference node succeeds in its transmission (w.p. $pa_{k,n}$), the MC transits to component SY_X (i.e. SA_X or SD_X). We define a $(\overline{Y}+1) \cdot X$ vector $\mathbf{SY_X}$ for component SY_X (We will discuss the details of SY_X later in this section), and then similar to Equation (5.3) the input for $\mathbf{SY_X}$ (i.e. the output of $\mathbf{Y_{X_i}}$ when the reference node succeeds) can be written as,

$$\begin{aligned}&\mathbf{SY_X}(k \cdot X + n + d \cdot X + 1) = \\ &\qquad pa_{k,n} \sum_i \mathbf{Y_{X_i}}(k \cdot X + n, 1), \quad k \le \overline{Y} - d, \mathbf{Y} \text{ is } \mathbf{A} \text{ or } \mathbf{D} \end{aligned} \tag{5.5}$$

While for the output of $ATIM_X/DATA_X$ when the reference node fails, we notice that the dashed line box in Figure 5.4 correspond to the time instances close to the end of a ATIM/DATA window. A node in these states may use up its window and exit $ATIM_X/DATA_X$. For example, for states after $(i, j, Y - d, n)$ of the reference node, if it sees a transmission of other nodes, its window counter increases by d, thus the window is exhausted (since $Y - d + d = Y$), it has to leave $ATIM_X/DATA_X$. Since the bottom $d{\cdot}X$ rows (except the last X row) of $\mathbf{Y_{X_i}}$ will exit $ATIM_X/DATA_X$ w.p. $1 - pa_{k,n}$ (w.p. 1 for the last X rows where the time left in the window is too short for even one time slot of waiting), the output $\mathbf{FY_X}$ of $\mathbf{Y_{X_i}}$ when the reference node fails can be as,

$$\begin{aligned}\mathbf{FY_X}(e) = \sum_{i=1}^{m} \Bigg(&\sum_{j=1}^{d-1} (1 - p_a) \cdot \mathbf{Y_{X_i}}((\overline{Y} - j) \cdot X - e + 1, 1 : W_i) \\ &+ \mathbf{Y_{X_i}}(\overline{Y} \cdot X - e + 1, 1 : W_i) \Bigg), \quad 1 \le e \le X, \mathbf{Y} \text{ is } \mathbf{A} \text{ or } \mathbf{D}\end{aligned} \tag{5.6}$$

For the ATIM window, since the output of $\mathbf{A_{X_i}}$ (i.e. $\mathbf{FA_X}$) is the input for EW_X, we define another $(\overline{Y}+1) \cdot X$ dimensional vector $\mathbf{EW_X}$ for component EW_X (we will cover the details of EW_X later in this section). Then we have $\mathbf{EW_X}(e) = \mathbf{FA_X}(e)$. The high level transition probability vector for this case is $\mathbf{PFA_X} = \mathbf{FA_X}/i_{A_X}$ (recall that $\mathbf{PFA_X}$ corresponds to P_{AC} in Figure 5.3). While for the DATA window, since there is no EW_X for it, we only need to express the corresponding high level transition probability vector as $\mathbf{PFD_X} = \mathbf{FD_X}/i_{D_X}$ (note that $\mathbf{PFD_X}$ is P_{DF} in Figure 5.3).

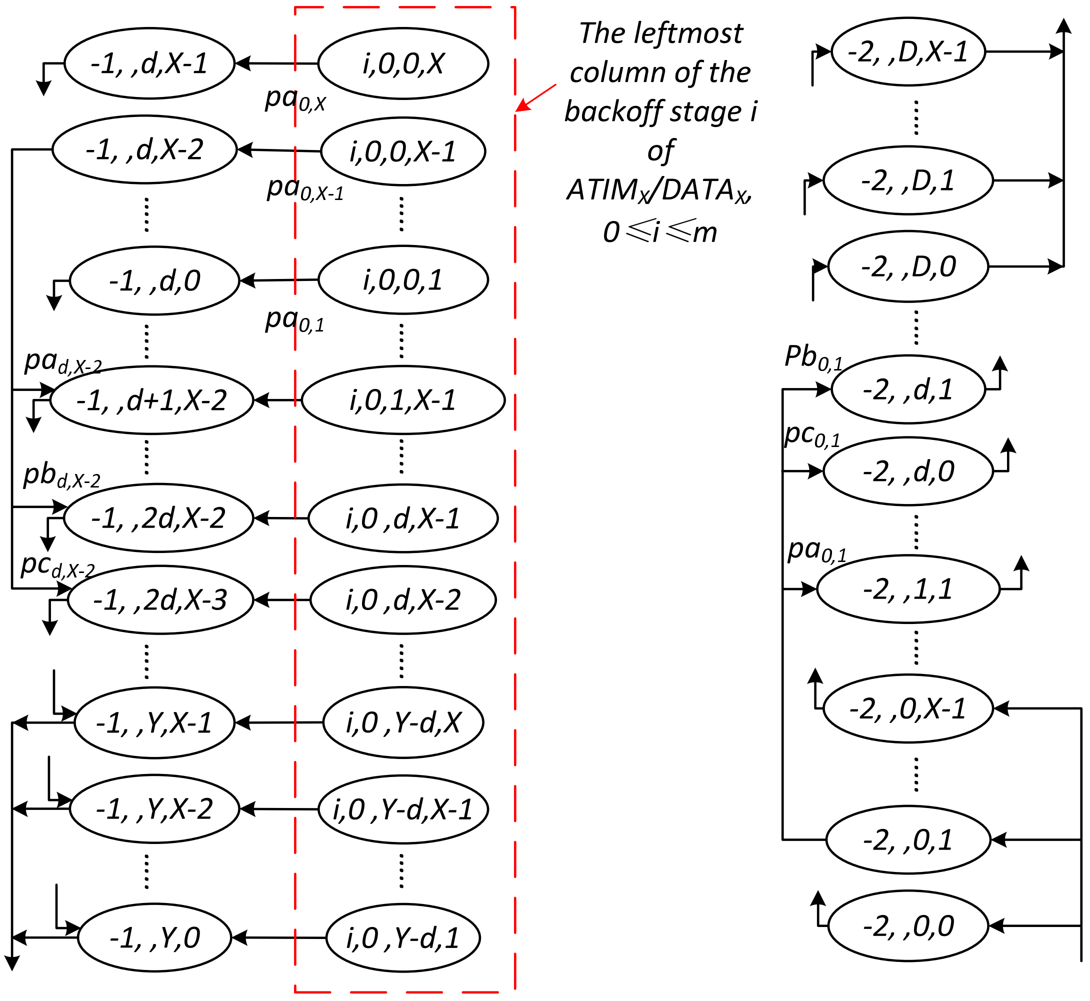


Figure 5.5: Detailed MC for SY_X (left) and EW_X (right).

The details of SY_X: As mentioned before, component SY_X (i.e. SA_X and SD_X) occurs in the ATIM or DATA window when the reference node succeeds in the corresponding window. Recall that the fourth dimension is the number of active nodes (i.e. $n(t)$), even though in SY_X the reference node is not active (within the current window), the remaining node are still active, thus the MC of the reference node still needs to track $n(t)$. This property is important because the knowledge of the number of active nodes is used to determine which region the MC to enter in the following window. The detailed MC is depicted on the left side of Figure 5.5, where we can see two columns. The right column is the leftmost column of the MC of stage i ($i \in [0, m]$) of $ATIM_X/DATA_X$. By Equation (5.5), we know that the inputs of S_X are from the leftmost column of each stage of $ATIM_X/DATA_X$. Since the packet transmission takes d slots of time, a state $(i, 0, k, n)$ will transit to $(-1, , k+d, n-1)$ w.p. $pa_{k,n}$. The left column is the main part of SY_X. Similar to the update rule of $ATIM_X/DATA_X$, depending on whether the channel is free, contains a collision, or contains a successful transmission, each state $(-1, , k, n)$ has three outcomes, i.e. to state $(-1, , k+1, n)$, to $(-1, , k+d, n)$ or to $(-1, , k+d, n-1)$, respectively. It is worth noting that, when $k \geq \overline{Y} - d$ (so $k + d \geq \overline{Y}$, i.e. the window is used up), each state will only transit to $(-1, , k+1, n)$ (w.p. $pa_{k,n}$) and $(-1, , \overline{Y}, n)$ (w.p. $1 - pa_{k,n}$). Earlier, we have defined a vector (i.e. $\mathbf{SY_X}$) for SY_X, to systematize the update rule for a vector, we define a vector operator $\mathbf{UPDT}(Vec) = \sum_s \mathbf{UPDT}(Vec(s))$ as,

if $s < (\overline{Y} - d) \cdot X$

$$\mathbf{UPDT}(Vec(s)) : \begin{cases} Vec(s+X) += pa_{k,n} \cdot Vec(s) \\ Vec(s+d\cdot X) += pb_{k,n} \cdot Vec(s) \\ Vec(s+d\cdot X+1) += pc_{k,n} \cdot Vec(s) \end{cases}$$

if $(\overline{Y} - d) \cdot X \le s < (Y-1) \cdot X$

$$\mathbf{UPDT}(Vec(s)) : \begin{cases} Vec(s+X) += pa_{k,n} \cdot Vec(s) \\ Vec(s+d\cdot X) += (1 - pa_{k,n}) \cdot Vec(s) \end{cases}$$

if $s = (\overline{Y} - 1) \cdot X \quad \mathbf{UPDT}(Vec(s)) : \quad Vec(s+X) \mathrel{+}= Vec(s)$

where $s = k \cdot X + n$.

By updating using $\mathbf{UPDT(SY_X)}$ $(1 \le X \le N)$, the last X elements of $\mathbf{SY_X}$ (i.e. $\mathbf{SY_X}(\overline{Y} \cdot X : (Y+1) \cdot X)$, note that $\overline{Y} = \overline{A}$ or $\overline{D}$ for ATIM or DATA window) store the probabilities of different number of successful nodes given X active nodes at the beginning of the corresponding window. For example, the last element represents the probability that all nodes are successful given X active nodes, while the last but one represents the probability that exactly $X - 1$ nodes succeed. The high level transition probability vector $\mathbf{PSY_X}$ (i.e. $\mathbf{PSA_X}$ and $\mathbf{PSD_X}$) equals $\mathbf{SY_X}(Y \cdot X : (Y+1) \cdot X)/i_{Y_X}$ (recall that $\mathbf{PSA_X}$ corresponds to the P_{AB}, and $\mathbf{PSD_X}$ corresponds to P_{DE} in Figure 5.3).

The details of EW_X: In terms of component EW_X which represents the time waiting for the DATA window to end when the reference node fails in the ATIM window. Same as SY_X discussed before, although the reference node is not active, it still needs to track the number of active nodes in the network. The detailed MC for EW_X is shown on the right side of Figure 5.5. From before we know that the inputs of EW_X is from the bottom $d \cdot X$ rows of each stage of $ATIM_X/DATA_X$ (i.e., Equation (5.6)). Same as SY_X, we update EW_X using $\mathbf{UPDT(EW_X)}$ $(1 \leq X \leq N)$, and the last X elements of $\mathbf{EW_X}$ (i.e., $\mathbf{EW_X}(D \cdot X : (D+1) \cdot X)$) store the probabilities of number of successful nodes (although here the reference node fails) given X active nodes at the beginning of an ATIM window. We obtain the high level transition probability vector $\mathbf{PEW_X}$, which equals $\mathbf{EW_X}(D \cdot X : (D+1) \cdot X)/i_{A_X}$ (note that $\mathbf{PEW_X}$ is P_{AG} in Figure 5.3).

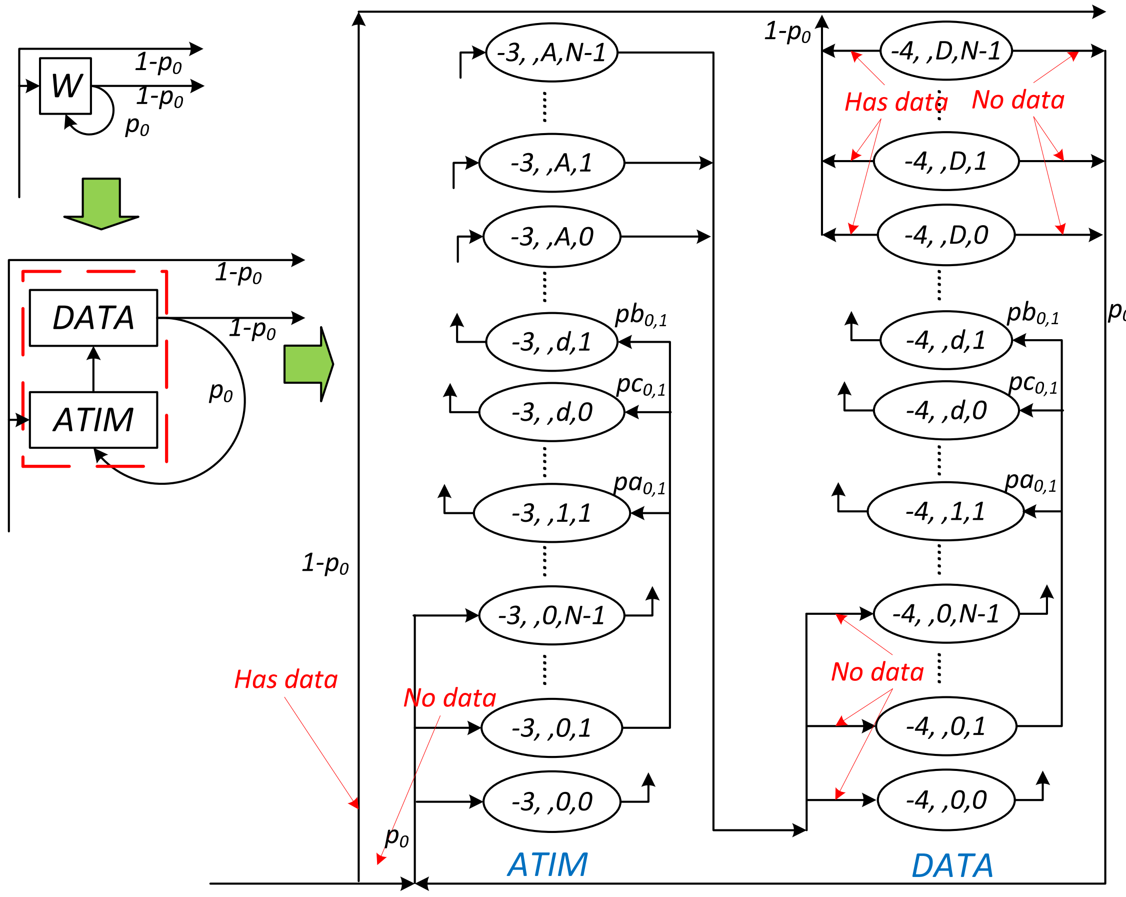


Figure 5.6: Detailed MC for component W.

The details of W: W represents time spent waiting for the queue to become non-empty. The detailed MC is shown in Figure 5.6. When entering W the reference node is at the beginning of a BI cycle, and if its queue has data (w.p. p_0) it exits W immediately to send the data. While if the queue is empty, it has to wait for an entire BI until it has the chance to check the status of its queue again. Thus the main body of W includes an $ATIM$ and a $DATA$ component to reflect an entire cycle. Generally, each MC for $ATIM$ and $DATA$ is similar to that of EW_X, and the outputs of $ATIM$ components directly lead to the input of $DATA$ component. Notably, at the output of $DATA$, besides the outputs that exit the W component (w.p. p_0), there is another output that leads to the input owing to an empty queue (w.p. $1-p_0$), and this loop corresponds to the one on the top-left corner in Figure 5.6.

Similar to vector $\mathbf{EW_X}$, we define two $(Y+1)\cdot N$ vectors $\mathbf{WA}$ and $\mathbf{WD}$ for component $ATIM$ and $DATA$ (in W), respectively. Assuming that at the beginning of a cycle the reference node enters W due to empty queue, the number of active nodes is X, and the corresponding input for W is i_{W_X}. Then we can apply $\mathbf{UPDT}(\mathbf{WA})$ and $\mathbf{UPDT}(\mathbf{WD})$ to derive the output vector $\mathbf{WD_X}(D\cdot N:(D+1)\cdot N)$, and the high level transition probability vector $\mathbf{PW_X}$ (i.e. from the input from $ATIM$ to the end of $DATA$ in W) is $\mathbf{WD_X}(D\cdot N:(D+1)\cdot N)/i_{W_X}$. We have not yet discussed the inputs for W, which will be covered later.

5.2.2.2 Solving the Markov Chain

Now we derive the high level transition probability matrix $\mathbf{P}$. Observe that in Figure 5.3, from one ATIM region, the MC can enter any one of the N DATA regions, i.e., it forms a one-to-many network, which can be represented by the product of a vector (i.e. PSA_X) and a matrix. Based on this idea, we define a high level transition probability vector $\mathbf{PS_X}$, which stores the transitions from the input of $ATIM_X$ to the point O^{SW} in Figure 5.3 (i.e. all possible outputs when the reference node succeeds in an BI cycle). Then, we have $\mathbf{PS_X} = \mathbf{PSA_X} \cdot \mathbf{PSD}$, where PSD is defined as $\mathbf{PSD} = [\mathbf{PSD_1}, \mathbf{PSD_2}, \cdots, \mathbf{PSD_N}]^T$. Similarly, for the case that the reference node fails in a BI cycle, we define a high level transition probability vector $\mathbf{PF_X}$ containing the transitions from the input of $ATIM_X$ to the point O^F in Figure 5.3. $\mathbf{PF_X}$ is expressed as $\mathbf{PF_X} = \mathbf{PEW_X} + \mathbf{PSA_X} \cdot \mathbf{PFD}$ because when the reference node fails in a BI cycle, it either fails in the ATIM window or succeeds in the ATIM but fails in the DATA window (note $\mathbf{PFD} = [\mathbf{PFD_1}, \mathbf{PFD_2}, \cdots, \mathbf{PFD_N}]^T$). Since $\mathbf{PS_X}$ and $\mathbf{PF_X}$ give us the transitions from the input (point I in Figure 5.3) to the overall output (O^{SW} and O^F), we still need the transitions from point O^{SW} to points O^S (trivial) and O^W, and the transitions from the overall output (points O^S, O^F and O^W) to the overall input I to complete the loop.

We now discuss the transitions from point O^S to the overall input. As we know, after the current cycle, if a node fails, it will resend the packet in the following cycle. Whereas if it succeeds, it may or may not send a new packet depending on whether it has one pending in its queue. This implies that the active number of nodes at the beginning of the new cycle depends on the result of previous cycle, i.e. the numbers of failed, successful and idle nodes. To model this property, we consider a scenario where there are X nodes initially, and at the end (of the cycle), i nodes (including the reference node) are successful. Then we know that the number of failed nodes is $X - i$, all of which will resend their packets in next cycle. While the remaining $N - (X - i)$ devices may or may not have data to send. If we assume the reference node has data in its queue, then the probability of j active nodes at the beginning of the new cycle is expressed as $\binom{N-X+i-1}{i+j-X-1} p_0^{N-j}(1-p_0)^{i+j-X}$ (recall that each node has same probability p_0 of queue being not empty). Now we define a $N \cdot N$ transition probability matrix, which represents that the reference node is successful and has data in the queue to the input of $ATIM_X$, as follows,

$$\text{if i<X, j<N-(X-1+i)} \quad \text{or} \quad \text{i>X ,} \quad \mathbf{PSI_X}_{i,j} = 0;$$
$$\text{otherwise,} \quad \mathbf{PSI_X}_{i,j} = \binom{N-X+i-1}{i+j-X-1} p_0^{N-j}(1-p_0)^{i+j-X} \tag{5.7}$$

Consequently, we express one term for $\mathbf{I'}$ at point O^S as,

$$\mathbf{I'^S} = \mathbf{I} \cdot [\mathbf{PS_1} \cdot \mathbf{PSI_1}, \cdots, \mathbf{PS_N} \cdot \mathbf{PSI_N}]^T$$

Since $\mathbf{PSI_X}$ is defined under the assumption that the reference node is successful and has data in the queue, for the cases when the queue is empty or the packet transmission failed, this matrix needs to be modified. We first define $\mathbf{PFI_X}$ to be the transition probability matrix from point O^F to point I. Then, from expression (5.7), we derive $\mathbf{PFI_X} = \frac{1}{1-p_0}\mathbf{PSI_X}$. Thus $\mathbf{I'}$ from point O^F is expressed as,

$$\mathbf{I'^F} = \mathbf{I} \cdot [\mathbf{PF_1} \cdot \mathbf{PFI_1}, \cdots, \mathbf{PF_N} \cdot \mathbf{PFI_N}]^T$$

The last case is that the reference node succeeds but has no data in the queue. Similar to the previous case where the reference node fails, we define $\mathbf{PSW_X}$ to be the transition probability matrix from point O^{SW} to the input of W, and we have $\mathbf{PSW_X} = \frac{p_0}{1-p_0}\mathbf{PSI_X}$. According to the structure of the MC, we express the input vector $\mathbf{b_0}$ of W as below,

$$\mathbf{b_0} = \mathbf{I} \cdot [\mathbf{PS_1} \cdot \mathbf{PSW_1}, \cdots, \mathbf{PS_N} \cdot \mathbf{PSW_N}]^T$$

Since W has a self-loop, the input $\mathbf{b_0}$ cannot be directly used to derive the output of W. We therefore derive the stationary probabilities of the input first. Observe that the loop back to W (w.p. p_0) is the same as the transition from SD_X to W. Thus, we define its transition probability matrix as $\mathbf{PWW_X} = \mathbf{PSW_X}$. Similarly, the transition probability matrix to exit W (i.e. to enter I, w.p. $1 - p_0$) is same as $\mathbf{PSI_X}$. Therefore, we obtain $\mathbf{PWI_X} = \mathbf{PSI_X}$.

Since we already have $\mathbf{PW_X}$ for W, we have,

$$\mathbf{b_0} + \mathbf{b} \cdot [\mathbf{PW_1} \cdot \mathbf{PWW_1}, \cdots, \mathbf{PW_N} \cdot \mathbf{PWW_N}]^T = \mathbf{b}$$

from which we solve for $\mathbf{b}$. In MATLAB, the solution is obtained using the "\" operator.

Then by using $\mathbf{b}$, we obtain $\mathbf{I'}$ from point O^W, as:

$$\mathbf{I'^W} = \mathbf{b} \cdot [\mathbf{PW_1} \cdot \mathbf{PWI_1}, \cdots, \mathbf{PW_N} \cdot \mathbf{PWI_N}]^T$$

Thus, the overall $\mathbf{I}$ is updated to $\mathbf{I'}$ at every iteration (cycle) until it converges, as follows ($\mathbf{P}$ can be derived from it),

$$\begin{aligned}
\mathbf{I'} =& \mathbf{I'^S} + \mathbf{I'^F} + \mathbf{I'^W} = \mathbf{I} \cdot \mathbf{P} \\
=& \mathbf{I} \cdot \Big([\mathbf{PS_1} \cdot \mathbf{PSI_1}, \cdots, \mathbf{PS_N} \cdot \mathbf{PSI_N}]^T + [\mathbf{PF_1} \cdot \mathbf{PFI_1}, \cdots, \mathbf{PF_N} \cdot \mathbf{PFI_N}]^T \Big) \\
& + \mathbf{b} \cdot [\mathbf{PW_1} \cdot \mathbf{PWI_1}, \cdots, \mathbf{PW_N} \cdot \mathbf{PWI_N}]^T
\end{aligned}$$

We also require the normalized condition (i.e. the summation of the stationary probabilities of all states is one) of the MC for this numerical method to work. Since we have the expression for the stationary input probability vector from which the stationary probability of each state can be attained, the normalized condition is applied easily.

5.2.3 Queueing Analysis

We now discuss how p_0 is determined using queueing theory concepts. First, we assume that the data arrival process is Poisson distributed. The service time is considered to be a general distribution, we apply $M/G/1$ queueing analysis to estimate the queue empty probability, p_0. For a $M/G/1$ queue, the queue empty probability, $p_0 = 1 - \lambda/\mu$, where λ is the average packet arrival rate and μ is the average service rate. Since the arrival rate λ is known, we only need to determine the mean of the service time, i.e., $1/\mu$.

The computation of the average service time for IBSS PSM protocol is simple because of the cyclic behavior. We use a method based on probability generation function (PGF). The PGF method is feasible because the length of the cycle $\overline{C}$ (i.e. $\overline{C} = \overline{A} + \overline{D}$) is fixed, thus can be discretized (i.e., treated as a unit). Consider a cycle with X active devices at the beginning of a cycle, we define a random variable T_X representing the service time of the reference node, and let $\phi_{T_X}(s)$ be the PGF of T_X. If the reference node fails in its transmission, at the beginning of the next cycle, the service time will be "renewed" to T_Y, where Y is the active nodes number for the new cycle. This renewal property for a random variable is precious since it greatly simplifies the analysis. Let z_1 be the number of active devices after one cycle (i.e. at the beginning of next cycle), based on the renewal property of T_X, we have the following expression for the PGF of T_X [95],

$$\begin{aligned}\phi_{T_X}(s) &= s^{\overline{C}} \sum_{i=1}^{N} E(s^{T_i} \mathbf{1}_{z_1=i}) \\ &= s^{\overline{C}} \sum_{i=1}^{N} (\mathbf{PF_X} \cdot \mathbf{PFI_X})(i) \cdot \phi_{T_i}(s)\end{aligned}$$

where $\mathbf{1}_{()}$ is the indicator random variable, and $(\mathbf{PF_X} \cdot \mathbf{PFI_X})(i)$ represents the transition probability from X active nodes at the beginning of current cycle to i active nodes at the beginning of next cycle, given that the reference node failed in current cycle.

Having obtained the PGF expression of T_X, we take its derivative to compute the expectation as $E(T_X) = \overline{C} + \sum_{i=1}^{N} (\mathbf{PF_X} \cdot \mathbf{PFI_X})(i) \cdot E(T_i)$. Since there are N such linear equations, we can solve for each $E(T_X)$, where $X = 1, \cdots, N$. After we have the mean of T_X, we compute the mean of the overall service time as:

$$E(T) = \sum_{k=1}^{N} \frac{I_k}{\sum_{j=1}^{N} I_j} \cdot E(T_X)$$

where $\frac{I_k}{\sum_{j=1}^{N} I_j}$ is the ratio of the stationary probabilities of different number of active nodes at the beginning of a cycle. Finally, by plugging $E(T) = 1/\mu$ into $p_0 = 1 - \lambda/\mu$, p_0 is obtained.

5.2.4 Performance Predication

After we have obtained all information about the MC, i.e., the stationary probability of each state and the probability of queue being empty, we are ready to derive the expressions for aggregate throughput, total delay, and energy consumption.

5.2.4.1 Aggregate Throughput

The definition of aggregate throughput is the average time used in successful transmission in DATA window for all devices. As we know, for the reference node to be successful in DATA window, it must succeed in the ATIM window first, and then transmit a DATA without collision before the DATA window is exhausted. Thus, we can express its behavior in the DATA window as below:

$$\mathbf{Th}(X) = d\sum_{i=0}^{m}\sum_{k=0}^{\overline{D}}\sum_{n=1}^{X}\mathbf{D_{X_i}}(k\cdot X+n,1)\cdot pa_{k,n}, \quad 1\le X\le N$$

where $\mathbf{Th}$ is a column vector, and $\mathbf{D_{X_i}}(k\cdot X+n,1)\cdot pa_{k,n}$ represents a successful transmission attempt in the DATA window (recall that $\mathbf{D_{X_i}}(k\cdot X+n,1)$ corresponds to the state $(i,0,k,n)$ in the MC for $DATA_X$).

To get the overall throughput, we plug in the stationary probabilities vector $\mathbf{I}$ and the transition probability matrix $\mathbf{PSA} = [\mathbf{PSA_1}, \mathbf{PSA_2}, \cdots, \mathbf{PSA_N}]^{\mathbf{T}}$ for the reference node being successful in the ATIM window. Thus the expression of the aggregate throughput is $N\cdot\mathbf{I}\cdot\mathbf{PSA}\cdot\mathbf{Th}$.

5.2.4.2 Total Delay

By Little's law, the total delay can be computed as $Delay = L/\lambda$, where L is the average length of the queue. For a $M/G/1$ queue, the expression for the average queue length is $L=\rho+\frac{\lambda^2(\sigma^2+1/\mu^2)}{2(1-\rho)}$ [94], where σ^2 is the variance of the service time, and $\rho=\lambda/\mu$.

Thus, we need to compute the variance of the service time. As we have derived earlier, we have the PGF of T_X. By the property of PGF, we know that $Var(T_X) = \phi''_{T_X}(1)+E(T_X)-E^2(T_X)$. The second derivative of $\phi_{T_X}(s)$ is expressed below, and it gives us N linear equations, solving which we can get the variance of each T_X, where $X=1,\cdots,N$.

$$\begin{aligned}\phi''_{T_X}(s) =& \overline{C}(\overline{C}-1)s^{\overline{C}-2}\sum_{i=1}^{N}(\mathbf{PF_X}\cdot\mathbf{PFI_X})(i)\cdot\phi_{T_i}(s)\\&+2\overline{C}s^{\overline{C}-1}\sum_{i=1}^{N}(\mathbf{PF_X}\cdot\mathbf{PFI_X})(i)\cdot\phi'_{T_i}(s)\\&+s^{\overline{C}}\sum_{i=1}^{N}(\mathbf{PF_X}\cdot\mathbf{PFI_X})(i)\cdot\phi''_{T_i}(s)\end{aligned}$$

To compute the overall variance of T, we use:

$$Var(T) = \left(\sum_{k=1}^{N}\frac{i_k}{\sum_{j=1}^{N} i_j}\right)^2 \cdot Var(T_X)$$

Finally, by Little's law, we get the per-node total delay.

5.2.4.3 Energy Consumption

Energy consumption is defined as the average awake time per cycle per node. The computation for energy consumption is similar to the one for throughput, except that when a node fails in the ATIM window, its awake time is only A; while if a node fails in the DATA window, the awake time is $\overline{C} = \overline{A} + \overline{D}$; and if a node successfully sends its data in DATA window, the awake time is then $\overline{A} + current\_time + d$, thus we have,

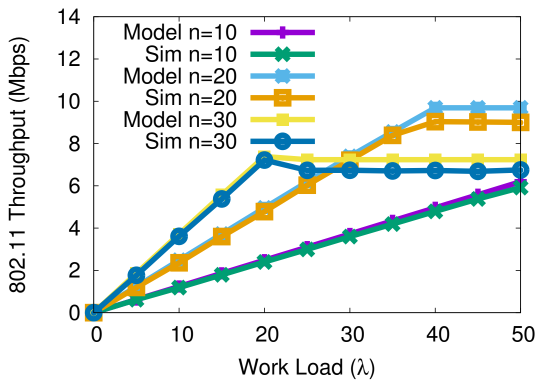


Figure 5.7: Throughput vs Load.

$$\mathbf{En}(X) = \overline{A}\sum \mathbf{PFA}_X + (\overline{A}+\overline{D})\sum \mathbf{PSA}_X \cdot \mathbf{PFD}$$
$$+\sum_{i=0}^{m}\sum_{k=0}^{\overline{D}}\sum_{n=1}^{X}(\frac{k\cdot X+n}{X} + d + \overline{A}) \cdot \mathbf{D_{X_i}}(k\cdot X+n, 1)\cdot pa_{k,n}$$

where $\frac{k\cdot X+n}{X}$ represent the current time in the DATA window.

Similar to throughput, the per-node energy consumption is computed as $\mathbf{I}\cdot\mathbf{PSA}\cdot\mathbf{En}$.

5.2.5 Model Validation

Our model has been realized in MATLAB using the *fsolve* function. Since $\tau_{k,n}$ does not depend on the input i_{Y_X}, given the number of devices X, ATIM/DATA window size $\overline{Y}$ (i.e. $\overline{A}$ or $\overline{D}$) and the minimal contention window size W_0, the matrix $\mathbf{Y_{X_i}}$ can be generated beforehand to save time. We have implemented our IBSS 802.11 PSM simulations based on the well-known ns-3 Simulator. The implementation involves significant modifications to ns-3, due to the lack of IBSS PSM support in the 802.11 module in ns-3. We compare the throughput, total delay, and energy

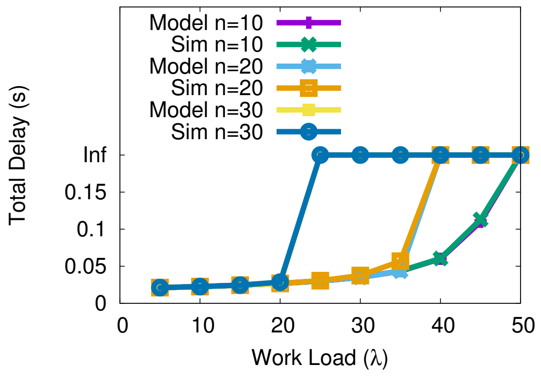


Figure 5.8: Delay vs Load.

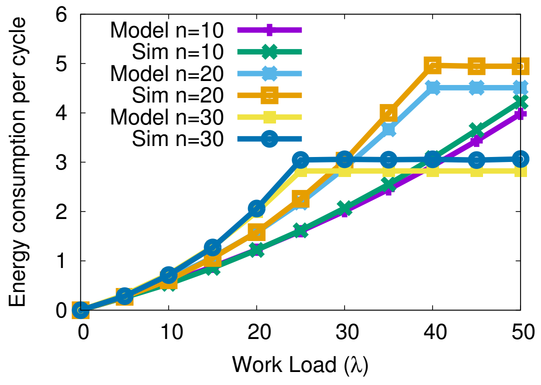


Figure 5.9: Energy vs Load.

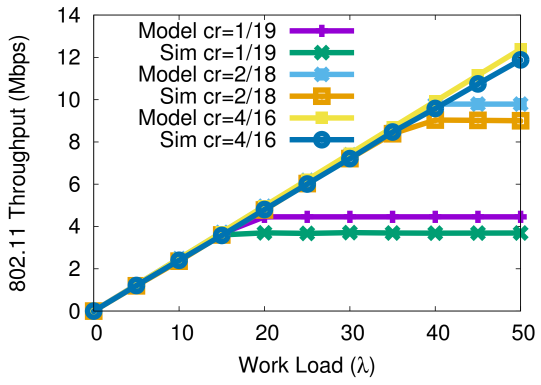


Figure 5.10: Throughput vs Load.

consumption obtained from our analytical model with those from the ns-3 simulator.

We performed extensive experiments by varying parameters in order to characterize the effects of nodes density, per-node workload (i.e. data arrival rate) and duty cycling ratio (i.e. cr =ATIM window size/DATA window size) on aggregate throughput, total delay and energy consumption. The parameters we vary are: number of nodes N, per-node offered load λ and the duty cycling ratio cr (we change the ratio of ATIM window to DATA window, while fixing the length of a cycle to 20ms). The default contention window sizes are $CW_{\min} = 16$, $CW_{\max} = 512$, the default packet size is $P = 1500B$, and the default values of the other parameters are: $N = 20$, $\lambda = 20$ and $cr = 2/18$ (i.e. ATIM window=2 ms, DATA window=18 ms).

The typical time used for the analytical model to converge is less than ten minutes on a fast PC (with i7 4790K CPU and 16GB ram). The simulation time for all tests is 10000s, which, on average, takes more than ten hours on the same machine.

5.2.5.1 Effects of per-node workload

Due to the significant impact of traffic arrival rate, i.e., per-node workload on the network performance, we study its effect on the throughput, delay as well as energy consumption. We vary the per-node workload from 5 to 50 packets per second. The results are depicted in Figure 5.7, Figure 5.8 and Figure 5.9 for throughput, total delay and energy consumption, respectively. As shown, all simulation results are in close agreement with those obtained from the model. In particular, for a given settings for other parameters (e.g. $N = 20$, $cr = 2/18$), when the workload λ increases until saturation, the throughput increases linearly. Then, due to the increased collision rate the throughput under saturation slowly decreases as the workload increases. As for the total delay, shown in Figure 5.8, a slight increase is observed when the workload increases. Especially when the queue becomes saturated, the delay tends toward infinity. In Figure 5.9, we see that the energy consumption increases (almost linearly) until saturation due to the increase in collisions. When the traffic is saturated, the energy consumption stays nearly constant because the nodes that failed in the ATIM window are forced to sleep in the DATA window of that cycle. From above we conclude that PSM saves energy even under saturated traffic scenario.

5.2.5.2 Effects of number of devices

The number of devices is another important factor that affects the performance of a network. Figure 5.7, Figure 5.8 and Figure 5.9 depict the effect on throughput, total delay and energy consumption when the number of WiFi devices are varied ($\{10, 20, 30\}$, we also vary workload as before and set $cr = 2/18$). As can be seen in the figures, the results from the simulator and analysis match each other closely. In Figure 5.7, when the traffic is not saturated, the throughput achieved for larger network is higher than that of a smaller network. However, due to the larger number of devices, a larger network reaches saturation faster when the workload increases. For example, for a network with 30 nodes, the traffic becomes saturated when $\lambda > 25$, while for a network of 10 nodes, $\lambda > 50$ causes saturation. In terms of total delay and energy consumption, Figure 5.8 and Figure 5.9 tell us similar results as the throughput, i.e. larger network causes saturation more easier. Especially when saturation is reached, the total delay becomes infinite because of the unlimited increase of the queue length, and the energy consumption does not change due to the "force sleep" mechanism of the PSM protocol.

5.2.5.3 Effects of duty cycling ratio

To understand the impact of duty cycling ratio on throughput, total delay and energy consumption, three different duty cycling ratio $cr \in \{1/19, 2/18, 4/16\}$ are examined (we still vary workload as before and set number of devices N to 20). The

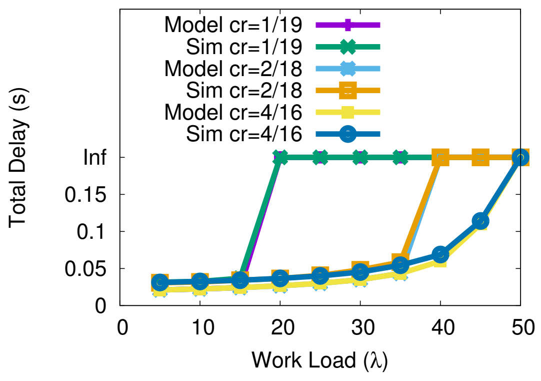


Figure 5.11: Delay vs Load.

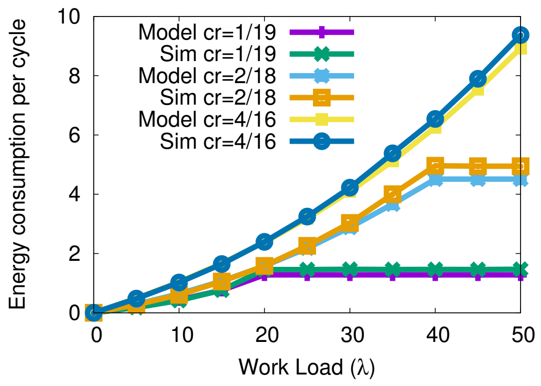


Figure 5.12: Energy vs Load.

results are depicted in Figure 5.10, Figure 5.11 and Figure 5.12 for throughput, total delay and energy consumption, respectively. As expected, the results obtained from the model agree with the simulation quite well. As shown in Figure 5.10, when the traffic is not saturated, as N is the same for all three cases, we observe the same throughput. Moreover, since there is a smaller chance that all devices are successful in a shorter ATIM window, with the increase of the workload, lower duty cycling ratio leads to saturation faster. For example, for $cr = 1/19$, the traffic becomes saturated when $\lambda > 20$, while for $cr = 4/16$, $\lambda > 50$ causes saturation. In terms of total delay and energy consumption, Figure 5.11 and Figure 5.12 reveal similar results as in Figure 5.10, i.e., shorter ATIM window causes saturation easier. In particular, under saturation, the total delay approaches infinity and the energy consumption stays constant due to the same reason as described before.

5.3 Approximation based Mathematical Model for DC-UNSAT

In this section, we first of all briefly describe the Ad-hoc 802.11 PSM and LPL protocols. Then we introduce the methodology employed in the analysis of the co-existing duty cycling networks. Finally an approximation analytical model is thoroughly discussed in subsections 5.3.2, 5.3.3 and 5.3.4.

5.3.1 Methodology

5.3.1.1 Challenges

Analyzing the performance of coexisting 802.11 PSM and LPL devices is very challenging due to following reasons.

First, in PSM, the number of active node (i.e. the node having data to send) in the network decreases in the ATIM window and DATA window because a node can only send one notification and one data packet (an assumption for simplicity). Thus, the classical method which based on a two dimensional Markov Chain is infeasible for PSM because it assumes that the number of active nodes is constant (i.e. saturation). Moreover, since the ATIM window and DATA window is finite, it is possible that a node gets reset before it succeeds, thus within a transmission cycle PSM cannot actually reach the stationary state (i.e. it is not independent of time). Therefore, in order to model PSM using a Markov Chain, two new dimensions are needed, one for the number of current active nodes and the other for the residual time in the ATIM/DATA window, as shown in Section 5.2. Since the time factor becomes the fourth dimension, Markov property is still maintained. This method gives high accuracy, however, becomes intractable very easily due to the immense states incurred by the large size of the DATA window (100ms/10μs=10000, where 10μs is the length of one timeslot).

Second, since LPL is asynchronous, it is even more complicated than the synchronous PSM. Similar to PSM, due to the duty cycling property, a LPL node may be reset when its cycle is exhausted, thus a dimension for the residual time of a cycle is necessary for a Markov Chain based model. Nevertheless, to save energy to its best, LPL usually has a very long cycle (e.g. 500ms) which simply renders the Markov Chain based method infeasible.

5.3.1.2 Main idea

To overcome the intractability of the high dimensional Markov Chain, this dissertation employs an approximation approach which is simple, computationally efficient and reasonably accurate.

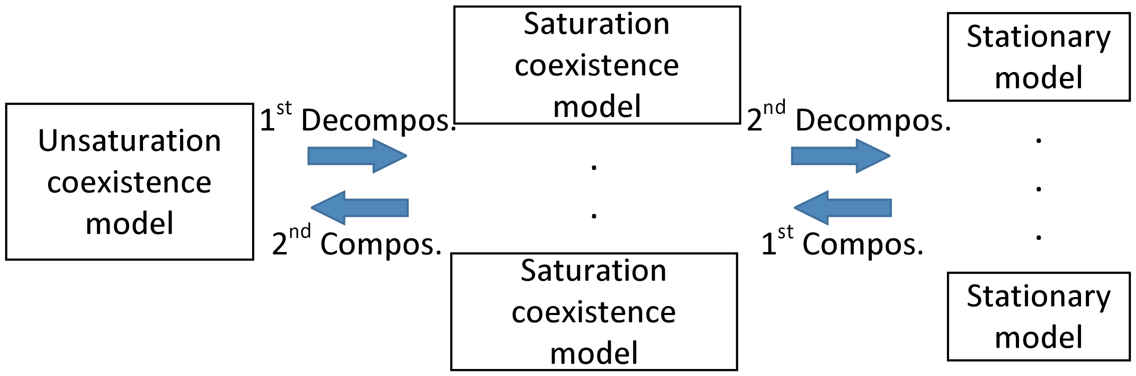

Unsaturation coexistence model
1st Decompos.
2nd Compos.
Saturation coexistence model
Saturation coexistence model
2nd Decompos.
1st Compos.
Stationary model
Stationary model

Figure 5.13: Main idea illustration

Step 1: As we know, the seminal 2-D Markov Chain model is for saturation problem which is easier to study due to the independence of the the number of active nodes. Thus our main idea is decomposing the entire problem of modeling the coexistence of PSM and LPL nodes with unsaturated traffic into several smaller yet tractable ones, i.e. the modelings of the coexistence of different numbers of saturated PSM and LPL nodes [74]. Then if the distribution of the number of saturated nodes (i.e. $Pr(n_W = w, n_B = b), w \in [0, W], b \in [0, B]$ where W and B are the number of PSM and LPL transmitters, respectively) can be obtained, the saturation models can be applied to approximate the unsaturated network effortlessly. These ideas are illustrated as the "1^{st} Decompos." and the "2^{nd} Compos." in Figure 5.13. The model for the saturated coexistence is discussed in 5.3.3, and the 2^{nd} composition is described in 5.3.4.

Step 2: However, saturation problem itself is still very challenging due to the aforementioned "reset" mechanism, thus it is necessary for us to further decompose it (i.e. "2^{nd} Decompos." in Figure 5.13). Basically, because of the "reset", a PSM or a LPL node cannot become stationary in a cycle, but we also notice that the length of a cycle is several orders of magnitude greater than a transmission of node (because an active node can only send one packet). In other words, although the stationary state cannot be reached, the saturation 2-D MC models for 802.11 and 802.15.4 (named *stationary model* here) should yield reasonably good approximation (i.e. "1^{st} Compos."). Specifically, when modeling the self-behavior of a node, the "reset" probability is ignored and the classical 2-D MC model is used (but notice that such "reset" probability takes effect (i.e. exists) for all other cases). The stationary models are discussed in detail in Section 5.3.2.

5.3.1.3 Key Assumptions

We have following key assumptions: a) All devices are within a single wireless cell, and each transmitter has one dedicated receiver (which is named "PSM pair" for short), i.e. there are $2W$ PSM nodes and $2B$ LPL nodes in total; b) Same type of devices are homogeneous (i.e. same Poisson traffic pattern, same data packet size); c) Channel is ideal, implying communication fails only due to collisions; d) For the stationary 2-D MC model, the "independent and constant" assumption for transmission attempt probability τ and the conditional collision probability p is applied as in [48]; e) Single CCA instead of double for LPL is assumed to simplify the coexistence analysis; f) Given w active PSM and b active LPL nodes, the corresponding reset probability (see $Pe_{W_{w,b}}$ and $Pe_{B_{w,b}}$ in Section 5.3.2.4 and 5.3.2.5, respectively) are independent and are constant for same type of devices.

5.3.2 Stationary Model

There are three stationary models that we consider for PSM, LPL and the channel, respectively, where PSM and LPL ones describe the self-behaviors of each, while the channel model analyzes their interactions. We assume that for each model, there are w ($w \in [0, W]$) and b ($b \in [0, B]$) active PSM and LPL nodes, respectively. *Note that this information is represented by the subscript w, b in the following context.*

5.3.2.1 PSM Model

Let us denote by $\tau_{W_{w,b}}$ the probability that a PSM node attempts to send, and $p_{W_{w,b}}$ the conditional collision probability. Then by the 2-D Markov Chain model [48] for saturation 802.11 network, we have

$$\tau_{W_{w,b}} = \frac{2(1 - 2p_{W_{w,b}})}{(1 - 2p_{W_{w,b}})(CW_0 + 1) + p_{CW_{w,b}} \cdot CW_0 \cdot (1 - (2p_{W_{w,b}})^m)} \tag{5.8}$$

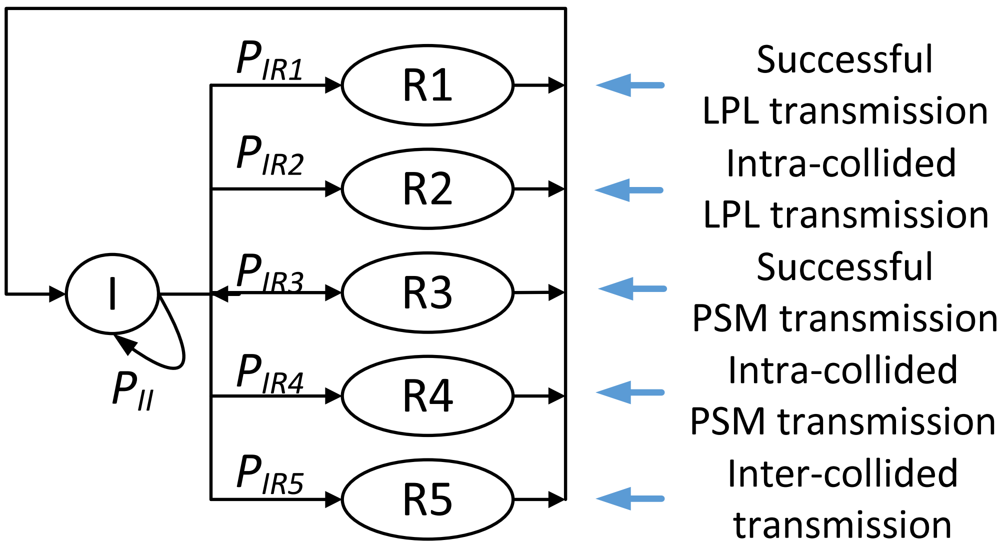
where CW_0 is the minimum contention window size and m is the maximum backoff stage.

5.3.2.2 LPL Model

Similar to the stationary model for PSM, we denote by $\tau_{B_{w,b}}$ the probability that a LPL node attempts to sense the channel, and $P_{I_{w,b}}$ the probability of the channel being idle, then by same approach as the 802.11 Markov Chain model [48], we have the following expression for $\tau_{B_{w,b}}$,

$$\tau_{B_{w,b}} = \frac{2}{3(P_{I_{w,b}}(CW'_0 + 1) + (1 - P_{I_{w,b}})(CW'_1 + 1))} \tag{5.9}$$

where CW'_0 and CW'_1 are the contention window sizes for the two backoff stages of 802.15.4. Since $P_{I_{w,b}}$ represents the state of the channel, it will be derived in next section.



P_{IR1}
P_{IR2}
P_{IR3}
P_{IR4}
P_{IR5}
P_{II}
I
R1
R2
R3
R4
R5
Successful LPL transmission
Intra-collided LPL transmission
Successful PSM transmission
Intra-collided PSM transmission
Inter-collided transmission

Figure 5.14: Markov Chain for the channel model

5.3.2.3 Model for the channel

The channel model is Markov Chain based, which describes the state of the channel. As shown in Figure 5.14, I stands for idleness of the channel and the R's represent different types of transmissions. Specifically, $R1$ and $R3$ represents a non-collided transmission of any LPL node and any PSM node, respectively; while $R2$ and $R4$ represents an intra-collision between LPL nodes only and between PSM nodes only, respectively; $R5$ denotes an inter-collision between LPL nodes and PSM nodes.

Let us first define a shorthand $SH_{B_{w,b}}$ for the probability that a LPL node keeps silent, i.e. it either gets reset (w.p. $Pe_{B_{w,b}}$) or is still backing off (w.p. $(1-Pe_{B_{w,b}})(1-\tau_{B_{w,b}})$), thus $SH_{B_{w,b}} \triangleq Pe_{B_{w,b}} + (1-Pe_{B_{w,b}})(1-\tau_{B_{w,b}})$. We also define another shorthand $SH_{W_{w,b}}$ for the probability that all PSM nodes keep silent, i.e. $SH_{W_{w,b}} \triangleq Pe_W + (1-Pe_W)(1-\tau_{W_{w,b}})^w$. Then the transition probabilities for the channel Markov Chain can be derived as,

$$\begin{aligned}
P_{IR1_{w,b}} &= \left(b \cdot \tau_{B_{w,b}} \cdot (1-Pe_{B_{w,b}}) \cdot SH_{B_{w,b}}^{b-1}\right) \cdot SH_{W_{w,b}} \\
P_{IR2_{w,b}} &= \left(\sum_{i=2}^{b} \binom{b}{i} \tau_{B_{w,b}}^{i} (1-Pe_{B_{w,b}})^{i} \cdot SH_{B_{w,b}}^{b-i}\right) \cdot SH_{W_{w,b}} \\
P_{IR3_{w,b}} &= \left((1-Pe_W) \cdot w \cdot \tau_{W_{w,b}} (1-\tau_{W_{w,b}})^{w-1}\right) \cdot SH_{B_{w,b}}^{b} \\
P_{IR4_{w,b}} &= \left((1-Pe_W) \sum_{i=2}^{w} \binom{w}{i} \tau_{W_{w,b}}^{i} (1-\tau_{W_{w,b}})^{w-i}\right) \cdot SH_{B_{w,b}}^{b} \\
P_{IR5_{w,b}} &= \left(1-SH_{B_{w,b}}^{b}\right) \cdot (1-Pe_W) \cdot (1-(1-\tau_{W_{w,b}})^{w}) \\
P_{II_{w,b}} &= 1 - \sum_{k=1}^{5} P_{IRk} \\
P_{I_{w,b}} &= \frac{1}{1+\sum_{k=1}^{5} L_{R_k} P_{IRk_{w,b}}}
\end{aligned} \tag{5.10}$$

where $P_{I_{w,b}}$ represents the stationary probability for the channel being idle, which can be obtained by solving this Markov Chain; L_{R_k} is the length of state $Rk, 1 \leq k \leq 5$; $P_{IR1_{w,b}}$ is the transition probability from I to $R1$, i.e. only one LPL node attempts to transmit (w.p. $b \cdot \tau_{B_{w,b}} \cdot (1 - Pe_{B_{w,b}}) \cdot SH^{b-1}_{B_{w,b}}$) and all PSM nodes do not send (w.p. $SH_{W_{w,b}}$); other transition probabilities are very similar to $P_{IR1_{w,b}}$, which are ignored here for concision.

Based on how the conditional collision probability for PSM (i.e. $p_{W_{w,b}}$) is defined, we can also derive $p_{W_{w,b}}$ by using $SH_{B_{w,b}}$ from the channel model, as,

$$p_{W_{w,b}} = 1 - (1 - \tau_{W_{w,b}})^{w-1} \cdot SH^{b}_{B_{w,b}} \tag{5.11}$$

So far we have four expressions (5.8)-(5.11) for six unknowns $\tau_{W_{w,b}}$, $\tau_{B_{w,b}}$, $P_{I_{w,b}}$, $Pe_{B_{w,b}}$, $Pe_{W_{w,b}}$ and $p_{W_{w,b}}$, another two equatoins are necessary for these simultaneous equations to be solvable. Now we begin to discuss how to derive $Pe_{W_{w,b}}$ and $Pe_{B_{w,b}}$.

5.3.2.4 $Pe_{W_{w,b}}$

According to the definition of $Pe_{W_{w,b}}$, since all PSM nodes are synchronized with each other, the probability of a PSM node being reset is simply the probability that a window is exhausted. Therefore for ATIM and DATA window, we have two different values for it, which are,

$$Pe_{W_{w,b}} \triangleq Pe_W = \frac{1}{T_{W_A}} \text{ (for ATIM) or } \frac{1}{T_{W_D}} \text{ (for DATA)} \tag{5.12}$$

where T_{W_A} and T_{W_D} are the lengths of the ATIM and DATA window, respectively.

5.3.2.5 $Pe_{B_{w,b}}$

For $Pe_{B_{w,b}}$, since whether or not a LPL node gets reset depends on if the corresponding receiver receives successfully, thus the reset probability is actually the probability that a node fails to transmit within the awake period of its receiver. Let us denote by T_B the cycle length of LPL, and by Pa the ratio of a receiver being awake in a cycle, thus the awake time of a receiver is $Ta_B = T_B \cdot Pa$. Given that the receiver is awake, there are three possible status for a transmitter, which are mute (due to backing off or busy channel), failed transmission and successful transmission, respectively. The probabilities for a node to enter one of these three status are as follows,

$$\begin{aligned}
Pi_{B_{w,b}} &= 1 - P_{I_{w,b}} \cdot \tau_{B_{w,b}} \\
Pf_{B_{w,b}} &= P_{I_{w,b}} \cdot \tau_{B_{w,b}} \left(1 - SH_{B_{w,b}}^{b-1} \cdot SH_{W_{w,b}}\right) \\
Ps_{B_{w,b}} &= P_{I_{w,b}} \cdot \tau_{B_{w,b}} \cdot SH_{B_{w,b}}^{b-1} \cdot SH_{W_{w,b}}
\end{aligned}$$

The expressions are self-explanatory and the description is thus ignored for brevity.

To compute the probability that a node failed to send a packet within Ta_B of its receiver (i.e. $Pe_{B_{w,b}}$), we note that a sender succeeds only if there is a successful transmission before the end of Ta_B. Since a node can only be in one of the aforementioned three status at any given point of time, and if we name both the mute and failed transmission statuses as a waiting status, there must be zero or more waitings before a success. Thus we can use Geometric distribution to derive the probability of the number of waitings (i.e. random variable X) before the first successful transmission as below,

$$Pr(X = k) = (Pi_{B_{w,b}} + Pf_{B_{w,b}})^k Ps_{B_{w,b}} = (1 - Ps_{B_{w,b}})^k Ps_{B_{w,b}}$$

where $1 - Ps_{B_{w,b}}$ is the probability for a waiting status.

Thus $Pe_{B_{w,b}}$ is simply the corresponding CDF for r.v. X, which can be expressed as,

$$Pe_{B_{w,b}} = Pr(X \geq K) = (1 - Ps_{B_{w,b}})^{K+1} \tag{5.13}$$

where K is the number of waitings a Ta_B can accommodate. The question now becomes how to compute K. Since we already have Ta_B, if we know the average time cost of each waiting $Du_{B_{w,b}}$, then K is imply $\frac{Ta_B - Ds_B}{Du_{B_{w,b}}}$, where Ds_B is the length of a successful transmission.

However, since a waiting can either be a mute (which lasts three timeslots) or a failed transmission (uses Df_B timeslots), we thus compute the length of a waiting as the average length of a mute and a failed transmission, i.e.

$$Du_{B_{w,b}} = \frac{Pf_{w,b}}{1 - Ps_{w,b}} Df_B + \frac{Pi_{w,b}}{1 - Ps_{B_{w,b}}} \cdot 3$$

Then K is obtained, and $Pe_{B_{w,b}}$ can be derived by using Equation (5.13). With (5.8)-(5.13), $\tau_{W_{w,b}}$, $\tau_{B_{w,b}}$, $P_{I_{w,b}}$, $Pe_{B_{w,b}}$, $Pe_{W_{w,b}}$ and $p_{W_{w,b}}$ can be obtained numerically. *It is worth emphasizing that the model here are only for a specific pair of w and b (and we name it **(w, b)-stationary-model**), and $W \times B$ rounds of computations are needed to get all required values for the approximation approach to work.*

As we know, the stationary model is used in the 1st composition process, i.e. approximating the saturation coexistence model, which is discussed in the next section.

5.3.3 Model of Coexistence of Saturated PSM and LPL

Since the objective is to derive the expressions of throughput and energy consumption for unsaturated duty-cycling coexistence problem, some higher level metrics rather than the fundamental probabilities obtained in the previous section are needed to achieve our goal more efficiently. The higher level metric we consider is the per-packet average MAC service time, which is the duration from a packet becomes the head of the queue until it gets transmitted successfully or dropped (for LPL only). As shown later, this metric is proved to be not only easy to obtain, but also convenient in deriving throughput and energy consumption. Therefore, the service time for PSM (denoted by $S_{W_{w,b}}$) and LPL (by $S_{B_{w,b}}$) are our focus in the following few subsections.

5.3.3.1 PSM

To obtain $S_{W_{w,b}}$, we need to know the probability that a packet gets transmitted successfully in a cycle (by $P_{W_{w,b}}$). Since each PSM cycle has ATIM and DATA windows, for a node to succeed in a cycle, it has to be successful in both windows. We denote by $P_{WA_{w,b}}$ and by $P_{WD_{w,b}}$ the probabilities that a packet gets transmitted successfully in ATIM and in DATA, respectively, and then we have $P_{W_{w,b}} = P_{WA_{w,b}} \cdot P_{WD_{w,b}}$. Due to the reset event, each cycle is independent from the one experienced before, thus $S_{W_{w,b}}$ can be derived by using the results of the Geometric distribution with respect to $P_{W_{w,b}}$ as,

$$S_{W_{w,b}} = T_W \cdot \sum_{i=0}^{\infty}(i+1)(1-P_{W_{w,b}})^i P_{W_{w,b}} = \frac{T_W}{P_{W_{w,b}}} \tag{5.14}$$

As $P_{W_{w,b}} = P_{WA_{w,b}} \cdot P_{WD_{w,b}}$, next we discuss how to obtain each for ATIM and DATA, respectively.

ATIM window: Here we notice that the probability $P_{WA_{w,b}}$ is actually the ratio of nodes that finish their transmissions in ATIM, i.e.

$$P_{WA_{w,b}} = \frac{\text{\# successful nodes in ATIM}}{w}$$

thus the number of successful nodes for ATIM window is needed. Obviously, since the length of ATIM window is fixed, if the average time usage of each successful transmission is known, the number of successful nodes is obtained simply by dividing the length of ATIM by the average time. Then the problem becomes how to compute the average length of each successful transmission.

As known, we assume the PSM and LPL nodes are saturated. Thus for PSM, at the beginning of each cycle (also the start point of the ATIM window), there are w active PSM and b active LPL nodes, implying that it can be approximated by the aforementioned (w, b)-stationary-model.

To compute the average length of the first successful transmission for case (w, b) in ATIM (denoted by $s_{WA_{w,b}}$), we notice that before the first successful transmission, there are three possible cases, i.e. idle, LPL transmission or failed PSM transmission. Since the transition probability to each case can be easily get from the stationary channel model, we can first compute the average length of them and then utilize

Geometric distribution again to compute $s_{WA_{w,b}}$, as

$$\begin{aligned}
P_{IB_{w,b}} &= P_{IR1_{w,b}} + P_{IR2_{w,b}} + P_{IR5_{w,b}} \\
P_{IWc_{w,b}} &= P_{IR4_{w,b}} \\
Du_{w,b} &= \frac{P_{II_{w,b}} \cdot 1 + P_{IB_{w,b}} \cdot Ds_B + P_{IWc_{w,b}} \cdot Df_W}{P_{II_{w,b}} + P_{IB_{w,b}} + P_{IWc_{w,b}}} \\
P_{IWs_{w,b}} &= P_{IR3_{w,b}} \\
s_{WA_{w,b}} &= (1 - P_{I_{w,b}}) \cdot \frac{Ds_B}{2} + \frac{1 - P_{IWs_{w,b}}}{P_{IWs_{w,b}}} \cdot Du_{w,b} + Ds_W
\end{aligned}$$

where $P_{IB_{w,b}}$ is the transition prob. from idle to a LPL transmission, $P_{IWc_{w,b}}$ is the one to a failed PSM transmission and $Du_{w,b}$ is the average length of the three cases. In the last equation, $((1 - P_{I_{w,b}}) \cdot \frac{Ds_B}{2})$ is the length of possible ongoing LPL transmissions at the beginning of the PSM cycle, $\frac{1-P_{IWs_{w,b}}}{P_{IWs_{w,b}}}$ is the number of failures (including the three cases) before the first success, which is obtained by Geo-Dist. Ds_W is the size of a successful PSM transmission.

Then after one PSM node succeeds, it quits the contention and thus the number of active PSM decrements (i.e. becomes $w-1$), then the new status can be approximated by the $(w-1, b)$-stationary-model, and so on until the ATIM window is used up. Since $s_{WA_{w,b}}$ is obtained, we can compute $s_{WA_{w-1,b}}, s_{WA_{w-2,b}}, \cdots, s_{WA_{1,b}}$ using similar approach. Finally, the average number of packet transmitted successfully within T_{W_A} can be computed as follows,

$$\max_{\sum_{i=0}^{x_{w,b}} s_{WA_{w-i,b}} \leq T_{W_A}} \qquad x_{w,b} \leq w$$

where $x_{b,w}$ is the average number of nodes that is successful in the ATIM window, thus $P_{WA_{w,b}} = \frac{x_{w,b}}{w}$.

DATA window: Inside DATA window, since the number of remaining active PSM is only $x_{w,b}$, there are at most $t_{w,b}+1$ values for $s_{WD_{j,b}}, j \in [0, x_{w,b}]$ (corresponds to $s_{WA_{j,b}}$). Assume there are $y_{w,b}$ nodes succeed in the DATA window, then $P_{WD_{w,b}} = \frac{y_{w,b}}{x_{w,b}}$. The detailed derivation for $P_{WD_{w,b}}$ and $s_{WD_{j,b}}$ are ignored for conciseness due to their high similarities with $P_{WA_{w,b}}$ and $s_{WA_{j,b}}$. *It is worth noting that since the length of DATA window is very long compare to the length of a successful transmission, almost all $x_{w,b}$ nodes succeed (i.e. $y_{w,b} \approx x_{w,b}$, thus $P_{WD_{w,b}} \approx 1$).*

With $P_{WA_{w,b}}$ and $P_{WD_{w,b}}$ (thus $P_{W_{w,b}}$), $S_{W_{w,b}}$ is obtained by Equation (5.14).

5.3.3.2 LPL

As described before, the number of active PSM nodes decreases in the ATIM/DATA window, which also affects the performance of LPL. Thus our idea to tackle this problem is still utilizing the stationary model, i.e. we first compute the MAC service time of LPL packet under different number of PSM nodes $j \in [0, w]$ by the (j, b)-stationary model as

$$s_{B_{j,b}} = Pe_{B_{j,b}} \cdot T_B + (1 - Pe_{B_{j,b}}) \cdot \frac{T_B}{2}$$

where $\frac{T_B}{2}$ comes from the observation that when a LPL succeeds in its cycle (w.p. $1 - Pe_{B_{j,b}}$), the average time cost is $\frac{T_B}{2}$ because the receiver averagely wakes up in the middle of a cycle (i.e. uniform distribution).

Then the ratio (i.e. distribution) of j active PSM nodes within T_W (denoted by $\gamma_{W_{j,b}}$) can be obtained by the following equation,

$$\gamma_{W_{j,b}} = \frac{s_{WA_{j,b}} \cdot \mathbb{1}_{[w-x_{w,b},w]}(j) + s_{WD_{j,b}} \cdot \mathbb{1}_{[x_{w,b}-y_{w,b},x_{w,b}]}(j)}{T_W}$$

where $\mathbb{1}_{[range]}(j)$ is an indicator function (namely the function equals to 1 if $j \in$

$[range]$, otherwise it is 0), $s_{WA_{j,b}}$ and $s_{WD_{j,b}}$ are described in the previous section.

Finally the actual MAC service time of a LPL packet under w saturated PSM and b saturated LPL $S_{B_{w,b}}$ is computed by taking the expectation of $s_{B_{j,b}}$ over the distribution of different active PSM nodes in T_W, i.e.

$$S_{B_{w,b}} = \sum_{j=0}^{w} \gamma_{W_{j,b}} \cdot s_{B_{j,b}} \tag{5.15}$$

So far, the key metrics for w saturated PSM and b saturated LPL have been obtained in expressions (5.14) and (5.15), we are now ready to apply it to our objective, i.e. the unsaturated coexistence case.

5.3.4 Model of Coexistence of Unsaturated LPL and PSM

As mentioned in Section 5.3.1.2, the main idea is obtaining the distribution of the number of saturated PSM and LPL nodes from the traffic pattern, and then approximating the unsaturated coexistence using the results from the analysis of saturated case. Since per-packet average MAC service time is important in throughput and energy consumption derivation, we thus compute them for unsaturated PSM and LPL (denoted by S_W and S_B) through $S_{W_{w,b}}$ and $S_{B_{w,b}}$.

5.3.4.1 M/G/1 model

To compute the distribution of the number of saturated PSM and LPL nodes from the Poisson traffic pattern, we employ two M/G/1 queueing models, which are described below for PSM and LPL, respectively.

PSM: M/G/1 queueing theory is useful to derive the queue idle probability $Q0_W$ for PSM as

$$Q0_W = 1 - \lambda_W \cdot S_W \tag{5.16}$$

where λ_W is the data arrival rate and S_W is the per-PSM-packet average MAC service time for unsaturated case (derived later).

Then with $Q0_W$, the distribution of the number of saturated PSM nodes (denoted by β_{W_w} here, w represents # saturated PSM nodes) can be obtained through

$$\beta_{W_w} = \binom{W}{w}(1 - Q0_W)^w \cdot Q0_W^{W-w}, w = 0, 1, \cdots, W \tag{5.17}$$

LPL: Similar to the analysis for PSM, we have the following two expressions for LPL,

$$Q0_B = 1 - \lambda_B \cdot S_B \tag{5.18}$$

$$\beta_{B_b} = \binom{B}{b}(1 - Q0_B)^b \cdot Q0_B^{B-b}, b = 0, 1, \cdots, B \tag{5.19}$$

where λ_B, S_B and β_{B_b} have similar meanings as these defined for PSM above.

5.3.4.2 Per-packet average MAC service time

The last steps are deriving S_W and S_B. Since both the distributions for the number of saturated nodes (i.e. β_{W_w} and β_{B_b}) and the per-packet average MAC service times for saturated case (i.e. $S_{W_{w,b}}$ and $S_{B_{w,b}}$) are get, the 2nd composition processes are straightforward, as follows,

$$S_W = \frac{\sum_{w=1}^{W} \beta_{W_w} \cdot \sum_{b=0}^{B} \beta_{B_b} \cdot S_{W_{w,b}}}{1 - \beta_{W_0}} \tag{5.20}$$

$$S_B = \frac{\sum_{b=1}^{B} \beta_{B_b} \cdot \sum_{w=0}^{W} \beta_{W_w} \cdot S_{B_{w,b}}}{1 - \beta_{B_0}} \tag{5.21}$$

By solving the simultaneous equations (5.16)-(5.21), all unknowns are attained. Note that equation (5.17) (or (5.19)) represents $W + 1$ (or $B + 1$) equations, but since the number of unknowns and equations are the same, they are solvable.

5.4 Performance Prediction for DC-UNSAT

With S_W, S_B and all other necessary metrics such as β_{W_w}, β_{B_b} etc, the expressions for throughput and energy consumption are derived in this section.

5.4.1 Throughput

Since LPL sender simply drops a packet if it fails in the cycle, while PSM retries until the packet succeeds, their throughput are defined differently.

5.4.1.1 PSM

For PSM, the maximum throughput is simply the reciprocal of per-packet MAC service time. If current data arrival rate is less than the maximum throughput, namely the traffic is unsaturated, the actual throughput equals to λ_W, otherwise (i.e. the traffic becomes saturated) the actual throughput is the maximum. Thus the expression for the actual throughput of a PSM sender is,

$$Tr_W = \lambda_W \cdot \mathbb{1}_{[0,\frac{1}{S_W})}(\lambda_W) + \frac{1}{S_W} \cdot \mathbb{1}_{[\frac{1}{S_W},+\infty)}(\lambda_W) \tag{5.22}$$

5.4.1.2 LPL

For LPL, the throughput is defined as the probability that a packet succeeds in a cycle, thus:

$$Tr_B = \sum_{b=1}^{B} \beta_{B_b} \cdot \sum_{w=0}^{W} \beta_{W_w} \cdot (1 - F_{B_{w,b}}) \tag{5.23}$$

where $F_{B_{w,b}}$ is the probability of an LPL packet being dropped (i.e. an LPL getting reset) in a cycle under saturated case, which can be obtained through $F_{B_{w,b}} = \sum_{j=0}^{w} \gamma_{W_{j,b}} \cdot Pe_{B_{j,b}}$, where $Pe_{B_{j,b}}$ is the probability that an LPL node gets reset in the **stationary model** (see equation (5.13)).

5.4.2 Energy consumption

Energy consumption is defined as the average percentage of time that a node stays awake within a cycle.

5.4.2.1 PSM

For PSM, there are three different cases, i.e. failure due to ATIM failure w.p. $1 - P_{WA_{w,b}}$, failure due to DATA failure w.p. $P_{WA_{w,b}} \cdot (1 - P_{WD_{w,b}})$, success w.p. $P_{W_{w,b}} = P_{WA_{w,b}} \cdot P_{WD_{w,b}}$. These cases correspond to different energy consumption, i.e. T_{W_A}, T_W and $L_{W_{w,b}}$, respectively, where L_W is the average energy consumption when a node successfully finishes its transmission in a cycle. Thus

$$L_{W_{w,b}} = T_{W_A} + \frac{1}{y_{w,b}} \sum_{i=0}^{y_{w,b}} \sum_{j=0}^{i} s_{WD_{(y_{w,b}-j),b}}$$

where $y_{w,b}$ and $s_{WD_{k,b}}$ are defined in Section 5.3.3.1.

Therefore the expression for energy consumption under the case (w, b) is,

$$En_{W_{w,b}} = \sum_{r=0}^{\infty} \sum_{i=0}^{r} \binom{r}{i} (1 - P_{WA_{w,b}})^i \left(P_{WA_{w,b}} \cdot (1 - P_{WD_{w,b}})\right)^{r-i} \cdot (P_{WA_{w,b}} \cdot P_{WD_{w,b}}) \frac{i \cdot T_{W_A} + (r-i) \cdot T_W + L_W}{r \cdot T_W}$$

where r is the number of retransmissions.

The equation above is difficult to solve because r can be infinity and $\binom{r}{i}$ is prohibitive to compute. However, as mentioned before, $P_{WD_{w,b}} \approx 1$, hence it can be simplified as,

$$En_{W_{w,b}} = \frac{T_{W_A}}{T_W} + \frac{(L_{W_{w,b}} - T_{W_A}) \cdot P_{WA_{w,b}}}{T_W \cdot (1 - P_{WA_{w,b}})} \cdot \ln(\frac{1}{P_{WA_{w,b}}})$$

Finally, since a PSM receiver consumes same energy with its transmitter due to their synchronization, the overall energy consumption of a PSM pair is expressed as,

$$En_W = \sum_{w=1}^{W} \beta_{W_w} \cdot \sum_{b=0}^{B} \beta_{B_b} \cdot En_{W_{w,b}} \tag{5.24}$$

5.4.2.2 LPL

The expression for LPL is extremely simple because according to the LPL protocol, a LPL node stays awake whenever it has data in its queue, thus for a transmitter $En_{B_T} = 1 - Q0_B$. While for a receiver, En_{B_R} is simply the awake probability Pa, i.e. $En_{B_R} = Pa$.

Thus the overall energy consumption of a LPL pair is,

$$En_B = \frac{(1 - Q0_B + Pa)}{2} \tag{5.25}$$

With Equations (5.22)-(5.25), performance can be optimized by formulating the requirement as an optimization problem with constraints, which is the focus of the following section.

5.5 Performance Tuning for DC-UNSAT

5.5.1 Motivation

As we know, a network usually has a fairly stable long term traffic pattern, however, since the default configurations for the duty cycling parameters are not optimized for coexistence scenarios, some energy is excessively wasted.

5.5.2 Parameters Tuning for Energy Consumption

To tackle this problem, we utilize the proposed model to optimize these parameters such that the total energy consumption is minimized while the traffic requirement is still respected. The corresponding optimization formulation is as follows,

$$
\begin{aligned}
\underset{R_W, R_B, T_W, T_B}{\arg\min} \quad & En \triangleq \frac{En_W \cdot W + En_B \cdot B}{W + B} \\
\text{subject to} \quad & Tr_W == \lambda_W, \\
& Tr_B > 85\%, \\
& R_W, R_B, T_W, T_B > 0.
\end{aligned}
$$

where En is the total energy consumption of both PSM and LPL (note that En is a percentage), R_W and R_W are the duty cycle ratios for PSM and LPL, respectively. The constraint $Tr_W == \lambda_W$ means that all input traffic is guaranteed to be served by a PSM transmitter, while for a LPL sender, since it has no retransmission mechanism, we ensure 85% of the traffic be served.

To solve this optimization problem, we notice that the solution space is not continuous because the unit of T_W and T_B is timeslot ($= 10\mu s$), we thus employ the "pattern search" [96] which is a powerful numerical optimization method for noncontinuous or non-differentiable problems. Specifically, we input the model to MATLAB and then formulate and solve the problems with the "patternserach toolbox" [97]. Usually, the convergence speed depends on the choice of the initial guessing, but the patternserach method always converges, which proves its effectiveness.

5.6 Monte Carlo based Simulator for DC-UNSAT

5.6.1 Simulator Design

A coexistence simulator for duty cycling network was implemented in the well-known ns-3 simulation framework. Specifically, we implemented in ns-3.25 the PSM protocol based on the WiFi module, the LPL protocol based on the LrWPAN module and also replaced their default channel modules with a common one such that PSM and LPL nodes were sensible to each other. Then all PSM and LPL nodes are placed in single cell to form a symmetric coexistence scenario, and we compare the throughput and energy consumption obtained from it against those from the proposed analytical model.

5.7 Model and Tuning Evaluation

5.7.1 Model for DC-UNSAT Validation

To validate our model, we compare the throughput and energy consumption obtained from our analytical model with those obtained from the proposed simulator.

We performed extensive simulations by varying following parameters: number of senders W and B, the per-node offered load (packet arrival rate) λ_W and λ_B, and the duty cycling settings (i.e. lengths T_W and T_B, and ratios R_W and R_B). The default settings are: $W = 20, CW_0 = 16$, $m = 6$, $P_W = 1000$ bytes, $\lambda_W = 5p/s$, $T_W = 100ms$, $R_W = 10\%$ for PSM, and $B = 20, CW'_0 = 120$, $CW'_1 = 70$, $P_B = 30$ bytes, $\lambda_B = 1p/s$, $T_B = 400ms$, $R_B = 5\%$ for LPL.

Typically, the time spent for solving the model is about two minutes on a fast PC (with Intel Core i7 4790K CPU, 16GB of RAM and 256GB SSD HD), while it takes around 20 minutes for the optimization problem to numerically converge (with reasonable initial guessing). The simulation times for all tests are 10000s, which take about one hour if the traffic is light and more than ten hours if otherwise.

5.7.1.1 Effects of number of nodes

Measuring the performance of coexisting PSM and LPL by varying the number of devices in hardware experiment is prohibitive in terms of deployment, especially when tens of devices are involved. However, it is very economic to do so using the analytical model and simulator. Figure 5.15 and Figure 5.16 depict the effect on throughput and energy consumption when the number of PSM (i.e., $\{5, 10, 15, 20, 25, 30\}$) and

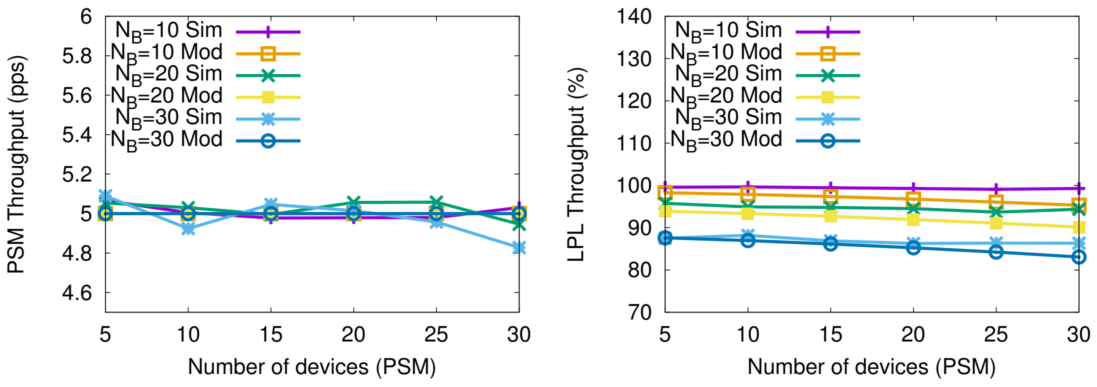


Figure 5.15: Throughput of model and simulator vs # devices.

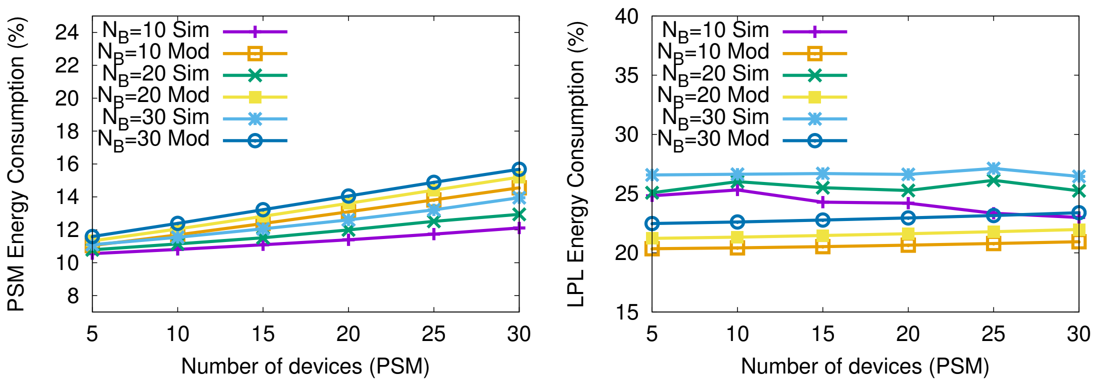


Figure 5.16: Energy cost of model and simulator vs # devices.

LPL (i.e., $\{10, 20, 30\}$) devices are varied. Note that all other settings are set to their defaults. Surprisingly, we observe that the results from the simulator match with the ones from the analysis closely. To be more specific, since the default $\lambda_W = 5p/s$, if the number of PSM is small (e.g. 15), PSM is unsaturated even there are 30 LPL, thus its throughput equals 5. While if $W = 30$ and $B \geq 20$, Tr_W is little less than 5 due to saturation. For LPL, with the increase of # nodes, Tr_B decreases due to more intensive contention. Note that the variation of W does not affect Tr_B significantly because PSM nodes are in active state only for a short time in a PSM cycle. In terms of energy consumption, En_W increases with W and B, but not hugely due to the property of PSM protocol (i.e. the number of successful PSM nodes in ATIM does not change much with # active nodes). Similar results apply to En_B as well.

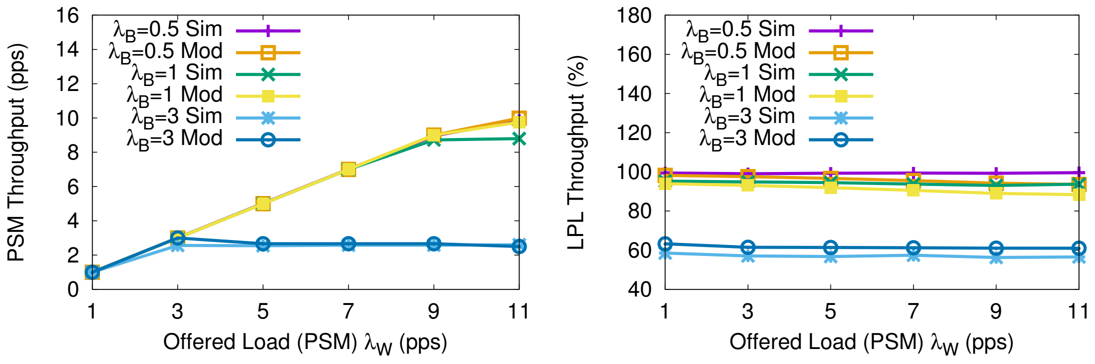


Figure 5.17: Throughput of model and simulator vs offered load.

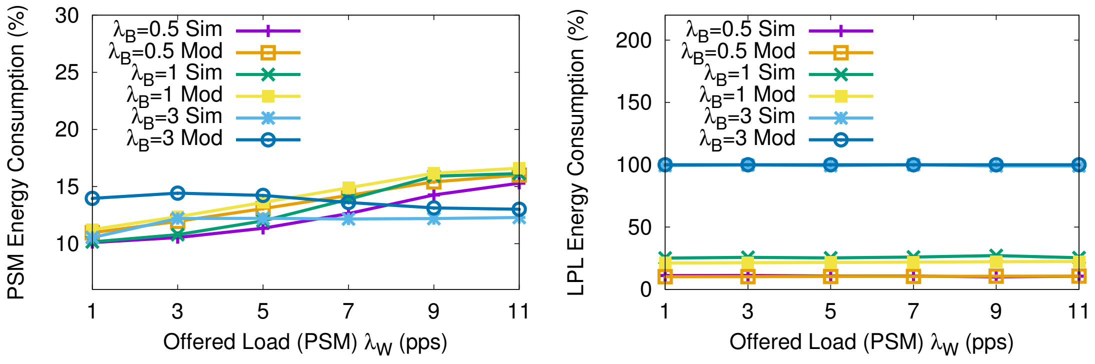


Figure 5.18: Energy cost of model and simulator vs offered load.

5.7.1.2 Effects of per-node offered load

Due to the significance of per-node offered load (i.e. traffic arrival rate) to the network performance, we study its effect on the throughput and energy cost. Similar as before, a scenario with 20 PSM and 20 LPL nodes with default parameters (except $\lambda_{W/B}$) are considered. We vary λ_W in $\{1, 3, 5, 7, 9, 11\}$ and λ_B in $\{0.5, 1, 3\}$ packet per second. The results obtained are depicted in Figure 5.17 and Figure 5.18 for throughput and energy consumption, respectively. Remarkably, the simulation results are in close agreement with those from the model. More precisely, when λ_W increases, if LPL is not saturated (i.e. $\lambda_B < 3$), since PSM is not saturated, its throughput increases linearly until λ_W is near 10 (where saturation is reached). While if LPL is saturated, PSM becomes saturated when λ_W reaches 3 (because the channel is too busy with LPL transmissions), thus Tr_W is almost constant for $\lambda_W \geq 3$. For LPL, as long as $\lambda_B < 3$ (i.e. unsaturated), $Tr_B \approx 100\%$, but if the saturation is attained, Tr_B can only get 60%. Due to the same reason as described in 5.7.1.1, the variation of the offered load of PSM has very limited impact on Tr_B. As for energy consumption, due to the properties of PSM and LPL protocols, En_W only increases slightly with λ_W, while En_B increases greatly with λ_B, especially when saturation is reached, En_B becomes 100% because the senders are always busy with the transmissions.

5.7.1.3 Effects of duty cycle settings

To analyze the impact of duty cycling length and ratio on throughput and energy usage, we investigate few different combinations of $(T_{W/B}, R_{W/B})$ with other default settings for the nodes. The combinations we use are $\{(50, 0.1)$, $(50, 0.2)$, $(100, 0.05)$, $(100, 0.1)$, $(200, 0.025)$, $(200, 0.05)\}$ for PSM, and $\{(200, 0.1)$, $(400, 0.04)$, $(800, 0.0125)\}$ for LPL. The results are depicted in Figure 5.19 and Figure 5.20 for

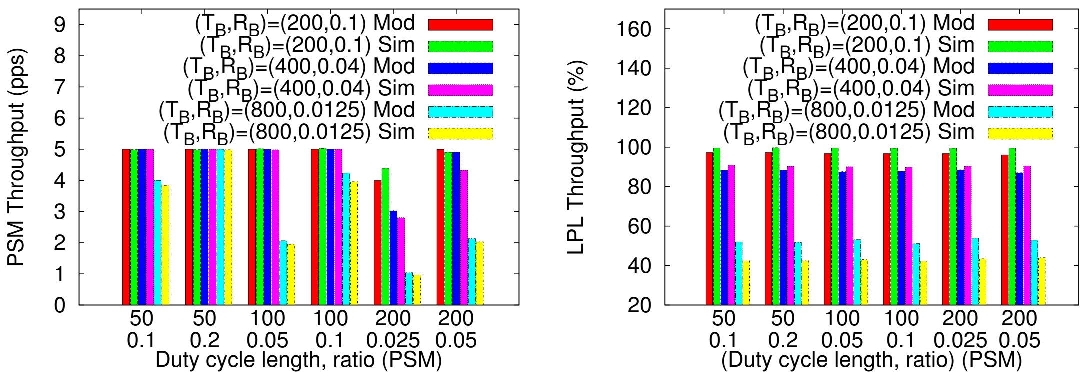


Figure 5.19: Throughput of model and simulator vs duty cycle.

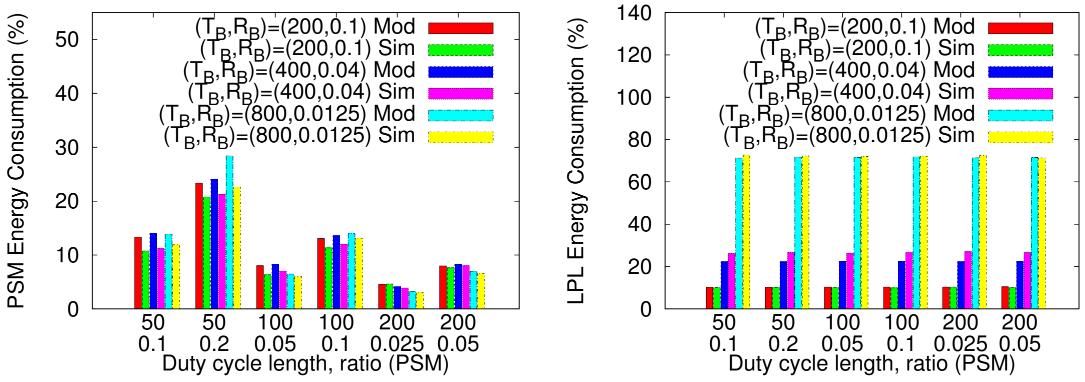


Figure 5.20: Energy cost of model and simulator vs duty cycle.

throughput and energy cost, respectively. Amazingly, the results obtained from the model agree with the simulation quite closely. In the case of throughput, when T_W is small, $Tr_W = \lambda_W$ due to the unsaturation; while when $T_W = 200ms$, if R_W is small (such as 0.025 here), since the ATIM is too short to send out enough packet, PSM becomes saturated (namely $Tr_W < \lambda_W$), otherwise (like $R_W = 0.05$), it is just enough to handle all incoming traffic, thus $Tr_W = \lambda_W$. As saturated LPL affects PSM significantly, we can thus observe that if $T_B = 800ms, R_B = 0.0125$, only a small $T_W = 50ms$ and a big $R_W = 0.2$ can ensure a unsaturated condition for PSM. The case for LPL is similar to what we discussed before, Tr_B is only around 50% when it is near saturation. In terms of energy cost, as before, for PSM, regardless of the saturation level, En_W is only slightly greater than R_W; while for LPL, its energy cost highly depends the degree of saturation (higher value implies bigger En_B).

5.7.2 Parameters Tuning Evaluation

Table 5.1: Results for the performance optimization, part 1

Data arrival settings	Default values						
	$\boldsymbol{En}$	$\boldsymbol{Tr_W}$	$\boldsymbol{Tr_B}$	T_W	R_W	T_B	R_B
$\lambda_W = 2, \lambda_B = 0.5$	**9.5%**	**2**	**98%**	100	0.1	400	0.05
$\lambda_W = 5, \lambda_B = 1$	**13.5%**	**5**	**92%**	100	0.1	400	0.05
$\lambda_W = 8, \lambda_B = 2$	**25.5%**	**5.25**	**69%**	100	0.1	400	0.05

Table 5.2: Results for the performance optimization part 2

Data arrival settings	Optimal values						
	$\boldsymbol{En}$	$\boldsymbol{Tr_W}$	$\boldsymbol{Tr_B}$	T_W	R_W	T_B	R_B
$\lambda_W = 2, \lambda_B = 0.5$	**5.4%**	**2**	**92%**	300	0.02	50	0.15
$\lambda_W = 5, \lambda_B = 1$	**6.9%**	**5**	**89%**	102	0.03	50	0.15
$\lambda_W = 8, \lambda_B = 2$	**12.1%**	**8**	**85%**	50	0.1	50	0.15

In this section, we evaluate the performance optimization method proposed in Section 5.5. We demonstrate that this optimization minimize the total energy consumption of PSM and LPL, while satisfying their throughput constraints. This evaluation is performed on 20 PSM and 20 LPL nodes, firstly with default values for T_W, R_W, T_B and R_B, and then with the corresponding optimal values obtained from the optimization method. We inspect three different scenarios of the data arrival rate, which are $\{\lambda_W = 2, \lambda_B = 0.5\}$, $\{\lambda_W = 5, \lambda_B = 1\}$ and $\{\lambda_W = 8, \lambda_B = 2\}$.

As shown in Table 5.1 and 5.2, when the work load is light (e.g. $\lambda_W = 2, \lambda_B = 0.5$), the default duty cycling settings (namely $T_W = 100ms$, $R_W = 0.1$, $T_B = 400ms$ and $R_B = 0.05$) can offer satisfactory results. Specifically, under such condition, the total energy consumption is only 9.5% and both throughput constraints are met, however it is still not optimal. By our optimization method, the optimal settings for the four parameters are $T_W = 300ms$, $R_W = 0.02$, $T_B = 50ms$ and $R_B = 0.15$, and the corresponding energy cost is 5.4%, thus 4% more energy are saved than the default one. A more interesting case is when the traffic load becomes high (for instance, $\lambda_W = 8, \lambda_B = 2$), where the default duty cycling setting consumes 25.5% of the energy, and especially, the throughput constraints cannot be satisfied for both PSM (i.e. $Tr_W = 5.25 < 8$) and LPL ($Tr_B = 69\% < 85\%$). However, the optimal settings (i.e. $T_W = 50ms$, $R_W = 0.1$, $T_B = 50ms$ and $R_B = 0.15$) generated by the optimization method yield amazingly well result, i.e. $En = 12.1\%$, which is only half of the default. At the same time, both throughput constraints are satisfied as well. Therefore, the proposed approach is very effective in optimizing energy efficiency.

6. CONCLUSIONS AND FUTURE WORKS

In this section, we conclude this dissertation and present some idea for future works.

6.1 Conclusions

Due to the ubiquity of WiFi and ZigBee networks, the demand for predicting and tuning the performance of their coexistence networks is becoming more and more intense. However, there are no simulators or mathematical models available for this purpose in the market, thus this research is devoted to solve this problem.

Specifically, based on the energy efficiency and traffic pattern of three practical coexistence scenarios i.e. disaster rescue cite, smart hospital and home automation, we first of all classify them into three classes, which are non-sleeping devices with saturated traffic (SAT), non-sleeping devices with unsaturated traffic (UNSAT) and duty-cycling devices with unsaturated traffic (DC-UNSAT). Then three simulator are proposed, one for each class, where each simulator is verified by using simple hardware based experiment. Similarly, we present three analytical models to address the corresponding performance prediction and tuning problems. More precisely, for UNSAT, our analysis is anchored in solid theoretical results based on modeling 802.11 DCF and BoX-MAC as Markov Chains, and a channel model that is able to accurately estimate channel busy probabilities; while for UNSAT, more delicately designed Markov Chain models are combined with M/G/1 queueing analysis to depict the unsaturated coexistence network; due to the big complexity for duty-cycling protocols, a divide-and-conquer strategy as well as an approximation method are employed to realize the modeling for DC-UNSAT. Next, for each class we derive the expressions for performance metrics like throughput, delay, energy consumption etc., and predict them using both the proposed simulator and the model. Due to the higher accuracy of the simulator, the results from them are used as ground truth to validate the accuracy of the model. Last, according to some common performance tuning requirements for each class (e.g. priority, satisfying delay constraint, minimizing energy consumption etc.), we formulate them into optimization problems and propose the corresponding solving methods.

The results show that the proposed simulators have high prediction accuracy, while the models, although are less accurate than the former, can be used in fast prediction. In particular, the models can be easily used in optimization problems for performance tuning, and the results prove its high efficiency.

6.2 Future Works

6.2.1 Improving model

Improving the model is an eternal theme. In this dissertation, we mainly use Markov Chain, which, although is convenient to use, is often too detailed, thus is cumbersome to computationally solve. So if we use a more abstract approach, such as Renew process, Point process, etc., the computation overhead can usually be reduced significantly. Certainly the side effect is that the modeling becomes far more difficult, which is a trade-off.

In addition to reducing the computational difficulty, it is possible to increase the practicality of the model. For example, here we assume that the traffic pattern is Poisson distribution, which simplifies the analysis, but does not reflect the reality very well. Thus for future works, we can take into account more practical traffic patterns such as Pareto distribution.

6.2.2 Hidden terminal problem

WiFi and ZigBee use the RTS/CTS mechanism to solve the famous Hidden terminal problem. But for coexistence networks, RTS/CTS does not work properly because WiFi and ZigBee do not understand each other. In this dissertation, the existences of terminal devices are not ignored for simplicity, i.e. it is assumed that every device is hearable from each other, and the basic (i.e. non-RTS/CTS) protocols are used.

Since coexistence of WiFi and ZigBee devices diminishes the efficacy of the RTS/CTS mechanism, the simulation and modeling for RTS/CTS protocols should be considered.

6.2.3 Coexistence of heterogeneous WiFi/ZigBee nodes

In this dissertation, we assume that all devices of the same type have the same traffic arrival rate, thus they will have same parameters when fairness is guaranteed. However, in reality, even the nodes of the same type may have different traffic arrival rate based on their actual needs. So it is meaningful to study the coexistence of heterogeneous WiFi/ZigBee nodes.

Obviously the simulator directly supports the corresponding simulation, but the situation for the model becomes complicated. Since the parameters of each node are likely to be different due to the heterogeneity, the variable space becomes much larger, and if the original model is used for performance tuning, it will be extremely difficult to solve the optimization problem. Thus for future works, a more powerful model should be found.

6.2.4 Packet size as a tunable parameter

For the sake of simplicity, only CW is used as a variable parameter in this research. In fact, if packet size is also tunable, the results from performance tuning may be even better. However, because the variable space increases, the optimization problem become very difficult to solve by using the original model. Therefore a better model is necessary, which can be a possible path for future work.

6.2.5 Real-time tuning and real-time distributed tuning

Performance tuning is currently performed either offline (i.e., at design time), or on-line on the central node, but neither is in real time (because one result often takes several minutes to obtain). Certainly, real-time performance tuning is more desirable, that is, to tune the parameter instantly according to the change of the environmental to optimally reach the performance objective. To do so, it is necessary to find approximation methods for both the model and the approach to solve the optimization problem, such that the optimal parameters can be get quickly. This is very challenging because modeling a complex network is always quite complicated, however, it can be a good direction for the future work.

In addition, distributed performance tuning is always preferred than centralize manner, but is far more difficult. Since each node only know their own information, they must infer the condition of the entire network based on the state of the channel, which is difficult. Furthermore, the computational power of each node is weak, so the aforementioned approximation methods must be even easier to solve, which, undoubtedly, is extremely challenging.

REFERENCES

[1] C. Systems, “Five reasons to go wireless.” http://www.cisco.com/cisco/web/solutions/small_business/resource_center/articles/work_from_anywhere/why_go_wireless/index.html.

[2] G. Miao, J. Zander, K. W. Sung, , and S. B. Slimane, *Fundamentals of Mobile Data Networks*. Cambridge University Press, 2016.

[3] W. Alliance, “WiFi.” http://www.wi-fi.org/.

[4] Z. Allicance, “ZigBee.” http://www.zigbee.org/.

[5] A. Research, “ABI Research Anticipates More Than 20 Billion Cumulative Wi-Fi Chipset Shipments by 2021 While Increased Use of 5GHz Spectrum Raises Coexistence Issues with LTE-U.” https://www.abiresearch.com/press/abi-research-anticipates-more-20-billion-cumulativ/.

[6] O. World, “IEEE 802.15.4 markets.” http://onworld.com/zigbee/.

[7] T. W. G. for WLAN Standards, “802.11.” http://www.ieee802.org/11/.

[8] I. . W. G. for WPAN, “802.15.4.” http://www.ieee802.org/15/.

[9] J.-S. Han, H.-S. Kim, J.-S. Bang, and Y.-H. Lee, “Interference mitigation in IEEE 802.15.4 networks,” in *GLOBECOM*, 2011.

[10] R. Xu, G. Shi, J. Luo, Z. Zhao, and Y. Shu, “Muzi: Multi-channel ZigBee networks for avoiding WiFi interference,” in *iThings/CPSCom*, 2011.

[11] W. Yuan, X. Cui, and I. G. Niemegeers, “Distributed adaptive interference-avoidance multi-channel MAC protocol for ZigBee networks,” in *CIT*, 2010.

[12] J. Huang, G. Xing, G. Zhou, and R. Zhou, "Beyond co-existence: Exploiting WiFi white space for ZigBee performance assurance," in *ICNP*, 2010.

[13] C. Liang, N. Priyantha, J. Liu, and A. Terzis, "Surviving WiFi interference in low power ZigBee networks.," in *SenSys*, 2010.

[14] Y. Yan, P. Yang, X. Li, Z. Yafei, L. Jianjiang, L. You, J. Wang, J. Han, and Y. Xiong, "Wizbee: Wise ZigBee coexistence via interference cancelation in single antenna," *Mobile Computing, IEEE Transactions on*, vol. 14, no. 12, pp. 2590–2603, 2015.

[15] X. Zhang and K. Shin, "Cooperative carrier signaling: Harmonizing coexisting WPAN and WLAN devices," *Networking, IEEE/ACM Trans.*, vol. 21, no. 2, pp. 426–439, 2013.

[16] NSNAM, "The Network Simulator - ns-2." http://www.isi.edu/nsnam/ns/.

[17] Riverbed, "OPNET." https://www.riverbed.com/products/steelcentral/opnet.html?redirect=opnet.

[18] A. Sikora and V. Groza, "Coexistence of IEEE 802.15.4 with other systems in the 2.4 GHz-ISM-band," in *IMTC*, 2005.

[19] S. Pollin, B. H. I. Tan, C. Chun, and A. Bahai, "Harmful coexistence between 802.15.4 and 802.11: a measurement-based study," in *CrownCom*, 2008.

[20] B. Radunović, R. Chandra, and D. Gunawardena, "Weeble: Enabling low-power nodes to coexist with high-power nodes in white space networks," in *CoNext*, 2012.

[21] G. P. Joshi, S. Y. Nam, and S. W. Kim, "Cognitive radio wireless sensor networks: Applications, challenges and research trends.," *Sensors (14248220)*, vol. 13, no. 9, 2013.

[22] Z. Chen, C.-X. Wang, X. Hong, J. Thompson, S. Vorobyov, and X. Ge, "Interference modeling for cognitive radio networks with power or contention control," in *2010 IEEE Wireless Communication and Networking Conference*, IEEE, 2010.

[23] T. M. Chiwewe, C. F. Mbuya, and G. P. Hancke, "Using cognitive radio for interference-resistant industrial wireless sensor networks: an overview," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1466–1481, 2015.

[24] C. Jiang, Y. Chen, K. R. Liu, and Y. Ren, "Analysis of interference in cognitive radio networks with unknown primary behavior," in *2012 IEEE International Conference on Communications (ICC)*, pp. 1721–1725, IEEE, 2012.

[25] B. H. Ellingsæter, "Cognitive radio: Interference management and resource allocation," 2010.

[26] A. Rabbachin, T. Q. Quek, H. Shin, and M. Z. Win, "Cognitive network interference," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 2, pp. 480–493, 2011.

[27] K. Lee, C.-B. Chae, T.-K. Sung, and J. Kang, "Cognitive beamforming based smart metering for coexistence with wireless local area networks," *Communications and Networks, Journal of*, vol. 14, no. 6, pp. 619–628, 2012.

[28] Y. Wang, Q. Wang, Z. Zeng, G. Zheng, and R. Zheng, "Wicop: Engineering WiFi temporal white-spaces for safe operations of wireless body area networks in medical applications," in *RTSS*, 2011.

[29] Y. Jeong, J. Kim, and S.-J. Han, "Interference mitigation in wireless sensor networks using dual heterogeneous radios," *Wireless Networks*, vol. 17, no. 7, pp. 1699–1713, 2011.

[30] S. M. Kim and T. He, "Freebee: Cross-technology communication via free side-channel," in *MobiCom*, 2015.

[31] I. Howitt, J. Gutierrez, *et al.*, "IEEE 802.15.4 low rate-wireless personal area network coexistence issues," in *WCNC*, 2003.

[32] J. W. Chong, H. Y. Hwang, C. Y. Jung, and D. K. Sung, "Analysis of throughput in a ZigBee network under the presence of WLAN interference," in *ISCIT*, 2007.

[33] W. Yuan, X. Wang, and J. Linnartz, "A coexistence model of IEEE 802.15.4 and IEEE 802.11b/g.," in *SCVT*, 2007.

[34] NSNAM, "The Network Simulator - ns-3." https://www.nsnam.org/.

[35] OMNeT++, "OMNeT++." https://omnetpp.org/.

[36] SCALABLE NETWORK TECHNOLOGIES, "QualNet." http://web.scalable-networks.com/qualnet.

[37] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 126–137, ACM, 2003.

[38] R. Malhotra, V. Gupta, and R. Bansal, "Simulation & performance analysis of wired and wireless computer networks," *Global Journal of Computer Science and Technology*, vol. 11, no. 3, 2011.

[39] S. Pandey and V. Tyagi, "Performance analysis of wired and wireless network using ns2 simulator," *International Journal of Computer Applications*, vol. 72, no. 21, 2013.

[40] R. Azizi, "Performance study and simulation of an anycast protocol for wireless mobile Ad-Hoc networks," *arXiv preprint arXiv:1307.6976*, 2013.

[41] C. Mallanda, A. Suri, V. Kunchakarra, and S. Iyengar, "Simulating wireless sensor networks with OMNeT++,"

[42] E. Egea-Lopez, J. Vales-Alonso, A. S. Martinez-Sala, P. Pavon-Marino, and J. García-Haro, "Simulation tools for wireless sensor networks,"

[43] A. Kostin and L. Ilushechkina, "Winsim: a tool for performance evaluation of parallel and distributed systems," in *International Conference on Advances in Information Systems*, pp. 312–321, Springer, 2004.

[44] S. Pollin, M. Ergen, S. Ergen, and B. Bougard, "Performance analysis of slotted carrier sense IEEE 802.15.4 medium access layer," *Wireless Comm., IEEE Trans.*, vol. 7, no. 9, pp. 3359–3371, 2008.

[45] Y. Xiao, X. Shan, and Y. Ren, "Game theory models for ieee 802.11 dcf in wireless Ad-Hoc networks," *IEEE Communications Magazine*, vol. 43, no. 3, pp. S22–S26, 2005.

[46] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New insights from a fixed point analysis of single cell IEEE 802.11 WLANs," in *IEEE/ACM Transaction on Network*, IEEE/ACM, 2007.

[47] P. Rathod, O. Dabeer, A. Karandikar, and A. Sahoo, "Characterizing the exit process of a non-saturated IEEE 802.11 wireless network," in *MobiHoc*, ACM, 2009.

[48] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *Selected Areas in Comm., IEEE Journal*, vol. 18, no. 3, pp. 535–547, 2000.

[49] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: throughput and delay analysis," *Computer communications*, vol. 25,

no. 3, pp. 313–321, 2002.

[50] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of reliable transport protocol over IEEE 802.11 wireless LAN: analysis and enhancement," in *INFOCOM*, 2002.

[51] C. H. Foh, J. W. Tantra, *et al.*, "Comments on IEEE 802.11 saturation throughput analysis with freezing of backoff counters," *IEEE Communications Letters*, vol. 9, no. 2, pp. 130–132, 2005.

[52] J. S. Vardakas, M. K. Sidiropoulos, and M. D. Logothetis, "Performance behaviour of IEEE 802.11 distributed coordination function," *IET circuits, devices & systems*, vol. 2, no. 1, pp. 50–59, 2008.

[53] I. Tinnirello, G. Bianchi, and Y. Xiao, "Refinements on IEEE 802.11 distributed coordination function modeling approaches," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 3, pp. 1055–1067, 2010.

[54] J. Choi, J. Yoo, and C.-k. Kim, "A novel performance analysis model for an IEEE 802.11 wireless LAN," *IEEE Communications Letters*, vol. 10, no. 5, pp. 335–337, 2006.

[55] Z. Hadzi-Velkov and B. Spasenovski, "Saturation throughput-delay analysis of IEEE 802.11 DCF in fading channel," in *ICC*, IEEE, 2003.

[56] P. Chatzimisios, A. Boucouvalas, and V. Vitsas, "Influence of channel BER on IEEE 802.11 DCF," *Electronics letters*, vol. 39, no. 23, pp. 1687–9, 2003.

[57] X. J. Dong and P. Varaiya, "Saturation throughput analysis of IEEE 802.11 wireless LANs for a lossy channel," *IEEE Communications Letters*, vol. 9, no. 2, pp. 100–102, 2005.

[58] R. Zheng, J. Hou, and L. Sha, "Performance analysis of power management policies in wireless networks," *Wireless Comm., IEEE Trans.*, vol. 5, no. 6, pp. 1351–1361, 2006.

[59] A. Zanella and F. De Pellegrini, "Statistical characterization of the service time in saturated IEEE 802.11 networks," *IEEE Communications Letters*, vol. 9, no. 3, pp. 225–227, 2005.

[60] T. Sakurai and H. L. Vu, "MAC access delay of IEEE 802.11 DCF," *Wireless Comm, IEEE Trans.*, vol. 6, no. 5, pp. 1702–1710, 2007.

[61] K. Duffy, D. Malone, and D. J. Leith, "Modeling the 802.11 distributed coordination function in non-saturated conditions," *IEEE Communications Letters*, vol. 9, no. 8, pp. 715–717, 2005.

[62] E. M. Winands, T. Denteneer, J. Resing, and R. Rietman, "A finite-source queuing model for the IEEE 802.11 DCF," *European transactions on telecommunications*, vol. 16, no. 1, pp. 77–89, 2005.

[63] C. H. Foh, M. Zukerman, and J. W. Tantra, "A Markovian framework for performance evaluation of IEEE 802.11," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1276–1265, 2007.

[64] D. Malone, K. Duffy, and D. Leith, "Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions," *Networking, IEEE/ACM Trans.*, vol. 15, no. 1, pp. 159–172, 2007.

[65] H. Zhai, Y. Kwon, and Y. Fang, "Performance analysis of IEEE 802.11 MAC protocols in wireless LANs," *Wireless comm. and mobile computing*, vol. 4, no. 8, pp. 917–931, 2004.

[66] R. P. Liu, G. J. Sutton, and I. B. Collings, "A new queueing model for QoS analysis of IEEE 802.11 DCF with finite buffer and load," *IEEE Transactions on Wireless Communications*, vol. 9, no. 8, pp. 2664–2675, 2010.

[67] B. Li and R. Battiti, "Analysis of the IEEE 802.11 DCF with service differentiation support in non-saturation conditions," in *Quality of Service in the Emerging Networking Panorama*, Springer, 2004.

[68] H. Zhai, X. Chen, and Y. Fang, "How well can the IEEE 802.11 wireless LAN support quality of service?," *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 3084–3094, 2005.

[69] O. Tickoo and B. Sikdar, "Modeling queueing and channel access delay in unsaturated IEEE 802.11 random access MAC based wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 4, pp. 878–891, 2008.

[70] Y. H. Bae, K. J. Kim, M.-N. Moon, and B. D. Choi, "Analysis of IEEE 802.11 non-saturated DCF by matrix analytic methods," *Annals of Operations Research*, vol. 162, no. 1, pp. 3–18, 2008.

[71] K. Ghaboosi, B. H. Khalaj, Y. Xiao, and M. Latva-aho, "Modeling IEEE 802.11 DCF using parallel space–time Markov chain," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 4, pp. 2404–2413, 2008.

[72] E. Karamad and F. Ashtiani, "Performance analysis of IEEE 802.11 DCF and 802.11 e EDCA based on queueing networks," *IET communications*, vol. 3, no. 5, pp. 871–881, 2009.

[73] Q. Zhao, D. H. Tsang, and T. Sakurai, "A simple model for nonsaturated IEEE 802.11 DCF networks," *IEEE communications letters*, vol. 12, no. 8, pp. 563–565, 2008.

[74] E. Felemban and E. Ekici, "Single hop IEEE 802.11 DCF analysis revisited: Accurate modeling of channel access delay and throughput for saturated and unsaturated traffic," *Wireless Comm., IEEE Trans.*, vol. 10, no. 10, pp. 3256–3266, 2011.

[75] P. Swain, S. Chakraborty, S. Nandi, and P. Bhaduri, "Performance modeling and evaluation of IEEE 802.11 IBSS power save mode," *Ad Hoc Networks*, vol. 13, pp. 336–350, 2014.

[76] O. Yang and W. Heinzelman, "Modeling and performance analysis for duty-cycled MAC protocols with applications to S-MAC and X-MAC," *Mobile Computing, IEEE Trans.*, vol. 11, no. 6, pp. 905–921, 2012.

[77] P. Park, P. Di Marco, P. Soldati, C. Fischione, and K. H. Johansson, "A generalized Markov chain model for effective analysis of slotted IEEE 802.15.4," in *MASS*, 2009.

[78] P. Park, C. Fischione, and K. Johansson, "Adaptive IEEE 802.15.4 protocol for energy efficient, reliable and timely communications," in *IPSN*, 2010.

[79] P. Park, P. Di Marco, C. Fischione, and K. H. Johansson, "Modeling and optimization of the IEEE 802.15.4 protocol for reliable and timely communications," *Para. and Dist. Sys., IEEE Trans.*, vol. 24, no. 3, pp. 550–564, 2013.

[80] S.-C. Wang, Y.-M. Chen, T.-H. Lee, and A. Helmy, "Performance evaluations for hybrid IEEE 802.11b and 802.11g wireless networks," in *IPCCC*, 2005.

[81] Z.-n. Kong, D. H. Tsang, B. Bensaou, and D. Gao, "Performance analysis of IEEE 802.11e contention-based channel access," *Selected Areas in Comm., IEEE Journal on*, vol. 22, no. 10, pp. 2095–2106, 2004.

[82] D.-J. Deng, C.-H. Ke, H.-H. Chen, and Y.-M. Huang, "Contention window optimization for IEEE 802.11 DCF access control," *Wireless Comm., IEEE Trans.*, vol. 7, no. 12, pp. 5129–5135, 2008.

[83] R. Pries, S. Menth, D. Staehle, M. Menth, and P. Tran-Gia, "Dynamic contention window adaptation (DCWA) in IEEE 802.11e wireless local area networks," in *ICCE*, 2008.

[84] Y. Yang, J. Wang, and R. Kravets, "Distributed optimal contention window control for elastic traffic in single-cell wireless LANs," *Networking, IEEE/ACM Trans.*, vol. 15, no. 6, pp. 1373–1386, 2007.

[85] L. Gannoune, S. Robert, N. Tomar, and T. Agarwal, "Dynamic tuning of the maximum contention window (CWmax) for enhanced service differentiation in IEEE 802.11 wireless ad-hoc networks," in *VTC*, 2004.

[86] Y. Jin and G. Kesidis, "Distributed contention window control for selfish users in IEEE 802.11 wireless LANs," *Selected Areas in Comm., IEEE Journal on*, vol. 25, no. 6, pp. 1113–1123, 2007.

[87] T. Cui, L. Chen, and S. H. Low, "A game-theoretic framework for medium access control," *Selected Areas in Comm., IEEE Journal on*, vol. 26, no. 7, pp. 1116–1127, 2008.

[88] T. W. G. for WLAN Standards, "Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *IEEE Std 802.11*, 2007.

[89] I. . W. T. G. 4, "Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area network (WPAN)," *IEEE Std 802.15.4*, 2006.

[90] D. Moss and P. Levis, "BoX-MAC: Exploiting physical and link layer boundaries in low-power networking," *Tech. Report SING-08-00*, 2008.

[91] D. Qiao and K. Shin, "Achieving efficient channel utilization and weighted fairness for data communications in IEEE 802.11 WLAN under the DCF," in *QoS, Tenth IEEE International Workshop on*, 2002.

[92] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica: Journal of the Econometric Society*, pp. 520–534, 1965.

[93] F. Daneshgaran, M. Laddomada, F. Mesiti, and M. Mondin, "Unsaturated throughput analysis of IEEE 802.11 in presence of non ideal transmission channel and capture effects," *Wireless Comm., IEEE Trans.*, vol. 7, no. 12, pp. 5129–5135, 2008.

[94] D. Gross, *Fundamentals of queueing theory.* John Wiley & Sons, 1998.

[95] S. I. Resnick, *Adventures in stochastic processes.* Springer Science & Business Media, 1992.

[96] R. Hooke and T. A. Jeeves, ""Direct search" solution of numerical and statistical problems," *Journal of the ACM (JACM)*, vol. 8, no. 2, pp. 212–229, 1961.

[97] Mathworks Matlab Help, "Patternsearch toolbox reference."