BANDWIDTH SCAVENGING FOR
DEVICE COEXISTENCE IN WIRELESS NETWORKS

By

Anthony Tyrone Plummer Jr.

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Electrical Engineering

2011

ABSTRACT

BANDWIDTH SCAVENGING FOR
DEVICE COEXISTENCE IN WIRELESS NETWORKS

By

Anthony Tyrone Plummer Jr.

The objective of this thesis is to develop a wireless channel access framework that allows secondary user (SU) devices to coexist with primary user (PU) devices via utilizing unused spectrum or whitespace found between the PUs' transmissions. Those whitespaces typically last for short durations (i.e. on the order of milliseconds) in between the active data transmissions by the PUs. The key design objectives for an SU access strategy are to "scavenge" the maximum amount of spatio-temporally fragmented whitespace while limiting the amount of disruptions caused to the primary users (PU). These conflicting goals become particularly challenging without deterministic prior knowledge about the future occurrences and durations of the whitespaces. Our approach to address this problem is to develop stochastic whitespace access mechanisms based on previously observed statistical model of the channel whitespace. The key contributions of this thesis are as follows. First, it provides an extensive statistical analysis of the whitespace characteristics using simulations and experiments on a prototype testbed, in the presence of various primary user traffic scenarios. The simulated network allows explorations of the effects of various traffic and topology conditions, and the experimental testbed provides insights into real world whitespace measurements in the presence of various hardware and operating system related limitations. The second contribution is an opportunistic access strategy for the secondary users that is developed based on the measurement and modeling of whitespace resulted from ad hoc mode 802.11 PU traffic. This opportunistic channel access, or scavenging, during ultra-short and non-deterministic 802.11 whitespace is then evaluated for functionality and performance through analytical modeling, network simulation, and testbed experiments. It is demonstrated that the proposed strategy is able to consistently scavenge above 90% of the available whitespace capacity, while keeping the primary users disruption less than 5%. The third major contribution is to generalize the above bandwidth scavenging approach by extending the proposed technique for arbitrary whitespace distributions. This generalization has resulted in a new access strategy that can be applied to non-802.11 primary traffic and is able to handle whitespace durations with multi-modal density functions. Effectiveness under arbitrary whitespace profiles is achieved by introducing a new concept of *transmission opportunities* within a given whitespace, and then developing a knapsack optimization based transmission probability shuffling mechanism across the transmission opportunities with the whitespace. Combining the above three components, the thesis offers a framework for SU support within a PU network with arbitrary topology and traffic profiles.

To My Wife

For All Her

Love And Support

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Subir Biswas for supporting me through my work. His guidance allowed me to flourish during my time graduate school. I would also like to thank my committee Dr. Matt Mutka, Dr. Tongtong Li, and Dr. Percy Pierre for their time and support.

I would also like to especially thank Dr. Percy Pierre and Dr. Barbara O'kelly for their guidance and support in helping me attend and be successful in graduate school.

Thanks must be also given to my lab mates Mahmoud Taghizadeh, Jayanthi Rao, Matthias Schmidt, Muhannad Quwaider, Tao Wu and Fan Yu for all of the brainstorming and implementation discussions. Last but not least, I would like give thanks to my wife Raenita Fenner, my parents Anthony Plummer Sr. and Avis Plummer along with the rest of my friends and family for supporting me through this process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

1.1 Prioritized Device Coexistence in Wireless Networks

Recent research on Dynamic Spectrum Access (DSA) has paved the way for a set of *Secondary* Users (SU) to access underutilized spectrum between *Primary* Users' (PU) transmissions in space, time and frequency. It has been shown [1-3] that such SU access can be feasible through studies of the primary user spectrum usage and the protocols that govern them. While this is particularly true for the license bands, it also applies to the unlicensed bands due to the recent proliferation and the subsequent overcrowding of wireless consumer technologies such as Bluetooth, ZigBee, WLAN, cordless phones, and related applications.

Of particular interest in DSA, is the issue of prioritized coexistence of various devices utilizing unlicensed bands [4-6]. An example of such coexistence is the sharing of the 2.4 GHz Industry Scientific and Medical (ISM) band among different device types including sensors and actuators, Voice-over-IP (VoIP) handsets (e.g. Skype phones), cordless phones, and data terminals such as laptops and data-enabled 3G/4G phones [7]. A suitable primary-secondary relationship among those co-existing devices are imposed based on their respective traffic priorities. For example, consider a VoIP primary user network utilizing a 2.4 GHz band. Along with the high-priority primary user devices there may coexist with low priority secondary user devices (e.g. sensors, laptops etc.) using the same band. The requirement here is to design a strategy to allow the secondary users to access the channel and utilize the bandwidth without disrupting or restricting the high-priority primary user devices.

1.2 General Definition of Primary and Secondary Users

In this work, a solution for multiple networked devices coexisting in a single wireless channel is investigated. Within such a network, a priority structure is defined among the devices, which are grouped into two types namely, Primary Users (PU) and Secondary Users (SU).

A PU is given exclusive access rights to the channel. These rights can be given by a network designer or government agencies such as Federal Communications Commission (FCC). For example, the FCC can determine that only police networks can use a particular wireless channel. The traffic properties of the PU network can vary greatly depending on the topology, traffic pattern, and data rates. As a PU, when there is data to send, channel access is immediate, unless there is another user with equal rights currently using the channel.

SUs are given lower access rights to the channel, which requires SUs to defer channel access when a PU is present. The overall goal of the SU is to minimize disruptions caused by its presence on the channel to a PU network. In opposition to this goal, the SU also wants to maximize its own throughput on the channel. This requires a smart strategy for the SUs to access the channel.

The cooperation between PUs and SUs can vary greatly. On one extreme, PUs and SU work closely together to access the channel. This can manifest in terms of off-line and on-line scenarios. In the off-line case, the PUs and SUs can divide the available bandwidth in terms of frequency or temporal, such as in TDMA protocols. For the online case, PUs can send notification messages to the SUs to indicate their arrival on the channel. Once the PU returns, SUs can just vacate the channel within a short delay.

In general, a secondary device may or may not be of the same type as the primary device. Additionally, the SU device may not understand protocol messages from the primary devices, which limits the amount of possible cooperation between the PUs and SUs. The goal of this thesis is to develop methods for SUs to access various types of PU networks. Therefore, PU and SUs are assumed completely independent of each other in-terms of sending and receiving packets. In other words, the PUs and the SUs cannot communicate with each other. However, the SUs have the ability to detect the primary user signal exclusively through the periodic measurement of the shared channel. This approach allows a number of general access solutions to be developed.

1.3 Role of MAC Layer to support Device Coexistence

Supporting device coexistence in wireless networks can be achieved using the Medium Access Control (MAC) layer through defining a suitable structure to control device access to the available network bandwidth. Coexistence between multiple systems can be achieved through spatial, frequency or temporal separation at the MAC layer, as reported in [7] and [8].

In spatial-based device separation, nodes define their range of communication i.e. transmission range, such that their transmissions do not interfere with other nodes' transmissions. For example, transmission power control can be used to adjust a nodes transmission power to define a suitable non-interfering transmission range [9]. SUs can set their transmission ranges at a minimal range that still allows communication but reduces the amount of disruptions to the PU traffic [10].

Frequency-based separation involves distributing nodes into multiple channels so that their transmissions do not interfere with each other. In this case, lower priority nodes would first identify unoccupied channels, then freely access the unused channels [2, 11]. Additionally adaptive frequency hopping techniques can be used.

Another promising mechanism for traffic prioritization is to utilize the temporal separation between a primary and secondary device accessing the same radio frequency (RF). In this approach, a secondary user models the channel behavior of the primary user network. Then access the channel at an advantageous time that does not interfere with the primary user. In this thesis, a secondary user access strategy is presented to utilize the temporal separation between primary and secondary devices accessing the same RF spectral segments.

1.4 Bandwidth Scavenging for Device Coexistence

To support prioritized device coexistence, a secondary device or user can utilize temporal based access. The secondary users' goal, therefore, is to access the idle times between the transmissions of primary users. These idle times or *whitespaces* typically last for short durations (i.e. order of milliseconds) in between the active data transmissions of PUs. From an SU's standpoint, once a whitespace is identified, the next step is to send a packet in that whitespace, only if the estimated chance of completing the transmission before the whitespace ends is high. The quality of whitespace access, that is bandwidth scavenging in this context, is determined by the resulting secondary user throughput which should be maximized, and the primary user disruption, which should be minimized for a given whitespace profile.

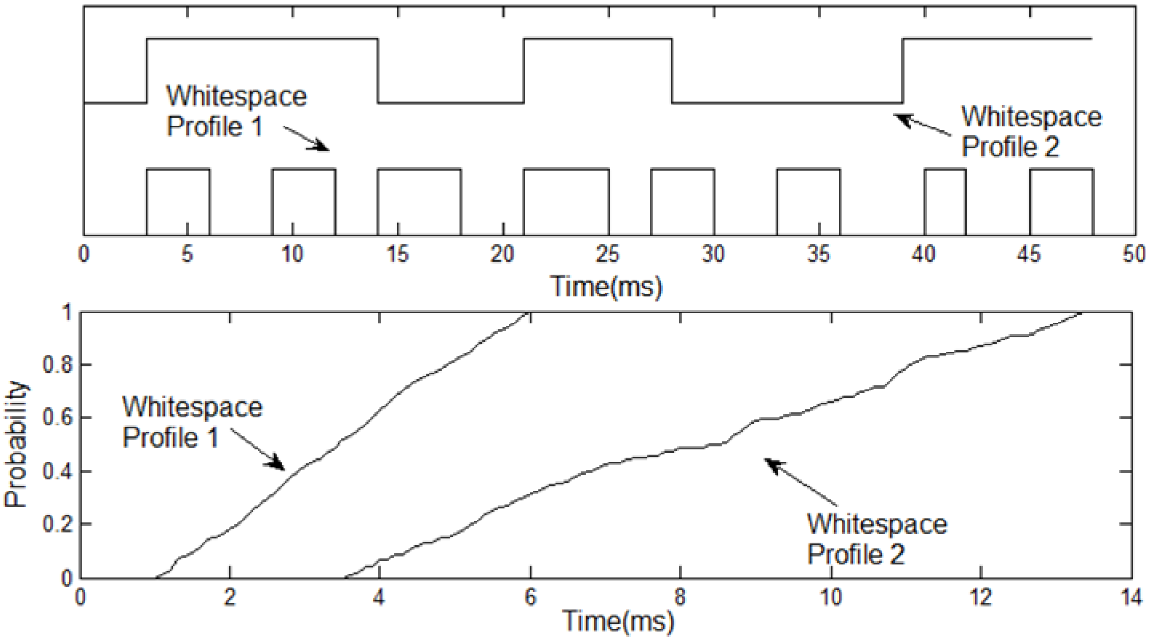


Figure 1-1: Different whitespace profiles and their CDFs

The SU whitespace access strategy depends solely on the statistical profile of the whitespace, which in turn, is determined by the topology, traffic and routing characteristics of the primary user network. Depending on these factors, the available whitespace can have widely varying characteristics. For example, the neighborhood of an SU may experience a large number of whitespace occurrences but each whitespace lasts only for a short duration (see Profile-1 in Figure 1-1.). Conversely, there can be only a few whitespaces, but each with very long durations, as shown in Profile-2 in Figure 1-1. The cumulative distribution functions (CDF) of the two whitespace profiles are also shown in the bottom part of the figure. These two different whitespace profiles indicate that even when the average white space length are the same, for a given target primary user interference, profile-1 is able to support less number of fixed length SU packet transmissions due to a higher level of capacity fragmentation during the ends of the whitespaces. Therefore, the SU throughput for profile-1 is less than that for profile-2, for the same level of target primary user interference.

With non-deterministic primary traffic patterns, an SU is not able to deterministically predict when a detected whitespace will end. As a result, the SU access strategy cannot be deterministic. A reasonable solution is for the SU to access a given whitespace based on a previously observed statistical whitespace profile. Once an SU identifies a whitespace, it sends a packet in that whitespace only if the estimated chance of completing the transmission before the whitespace will have ended is reasonably high. An aggressive access may result in higher secondary user throughput, but can also increase primary user disruptions, which happens when an SU's transmission (the last packet transmission in a whitespace) does not end before the completion of the whitespace in question. A conservative access strategy may produce a reverse effect, meaning a low PU disruption, but at the expense of an SU throughput that is lower than what could be achieved for a given whitespace profile. Therefore, the objective for an SU is to be able to make a transmission decision based on previously observed whitespace statistics, such that by prudently transmitting packets, it is able to maximize the achievable throughput and minimize the disruptions. We term this opportunistic access during ultra-short and non-deterministic whitespaces as *bandwidth scavenging* by the secondary users. In other words, the SUs scavenge the channel capacity left over by the PUs. In this thesis, a solution to prioritized device coexistence is proposed by designing secondary user channel access strategies for efficient bandwidth scavenging.

1.5 State of the Art

In this section, an overview of the current research on device coexistence is presented.

1.5.1 Traffic Models of Primary Users

The traffic profiles of packet based primary users studied in literature have varied greatly [12-15]. The authors in [15] model the primary users' packet arrival intervals and packet durations as two separate Poisson processes. Similarly, [13] uses an alternating ON-OFF pattern, where the duration of the idle and busy states follows an exponential distribution with means v and l respectively. In [12], a Markov chain with known transition probabilities is used to model primary user behavior. Lastly in [14], a Markov model with Exponential, Uniform, Weibull and Generalized Pareto distributions are considered for PUs arrival times. In line with these efforts in the literature, this thesis investigates SU access mechanisms for a variety of PU traffic profiles.

1.5.2 Random Access Based Primary User Networks

Networks with primary users running 802.11 MAC protocol have recently been investigated for possible access by secondary users. The authors in [16, 17] develop a methodology for formally analyzing the whitespace available within 802.11 primary traffic in infrastructure mode. The key idea is to model the whitespace as a semi-Markov process that relies on the underlying 802.11 state model involving DIFS, SIFS, DATA, and ACK transactions. The model describes the whitespace profile in terms of holding times of the idle and busy states of the channel.

The authors in [6] develop an WLAN access strategy for secondary users in which the SUs utilize packet size slots for the channel access. At the beginning of each slot, an SU senses the channel and if the channel is free then it transmits with a specified probability that is calculated from previous measurement. A greedy version also uses probabilistic analysis to decide whether or not to transmit in a given slot. The objective is to minimize PU disruption and maximize SU throughput. The use of time-slotting in their approach requires time synchronization across the secondary users. In the access mechanism proposed in this thesis, the need for such inter-SU time synchronization is avoided via asynchronous whitespace access based on a stochastic whitespace modeling approach.

Prioritized device coexistence within CSMA based WLANs can be achieved [18, 19] by using different *inter frame spacing* (IFS) periods for different user and/or traffic classes. MAC protocol 802.11e [18, 20], for instance, uses different Arbitration IFS (AIFS) periods to provide CSMA based prioritized access among different device and/or traffic classes. When a channel is found free, a node waits for a specific AIFS periods depending on the device or traffic class, before it attempts to send a packet. For higher priority primary traffic, a node waits for a smaller AIFS period. This ensures when multiple nodes contends for the channel, the primary users' nodes (with smallest AIFS) wins.

While providing reasonable access differentiation, these approaches rely only on the instantaneous channel status (i.e. free or busy) for granting access. This leads to undesirable disruptions to the PU traffic as follows. Consider a situation in which an SU intends to transmit a packet and it does so after finding the channel free (i.e. a whitespace) for the AIFS specified for the SUs. Now, in the middle of this SU's packet transmission if an PU in the vicinity intends to send a packet, it needs to wait until the current SU transmission is over. This causes an undesirable delay for the PU traffic, which in turn will affect the PUs' application performance. Since this is mainly a result of the SUs' reliance only on the instantaneous channel state, a more robust approach for the SUs would be to also consider the long term whitespace model. The following DSA approaches attempt to accomplish that.

The topic of device coexistence has also been studied in the context of Bluetooth and WLAN coexistence. The authors in [5] use OLA (OverLap Avoidance) mechanisms, which use simple traffic scheduling techniques at the MAC layer. In the proposed scheme, overlap in time between the Bluetooth traffic and the 802.11 data packets is avoided by performing a proper scheduling of the traffic transmissions at the WLAN stations. In a Bluetooth network, each link occupies the channel slots according to a deterministic pattern. Therefore, an 802.11 station begins transmitting when the Bluetooth channel is idle and adjust the length of the WLAN packet so that it fits between two successive Bluetooth transmissions.

1.5.3 TDMA based Primary User Networks

Time Division Multiple Access (TDMA) protocols have been investigated as a possible method to assign priority to different traffic classes. TDMA works by dividing a channel into fixed sized time slots, which are organized into periodic frames. Each node in a network is assigned a slot or multiple slots to be used during each periodic frame. To institute priority amongst the nodes, higher priority nodes are assigned a larger portion of the available slots in a frame over the lower priority nodes. This process ensures that the higher priority nodes utilizes the channel with higher frequency and allows dedicated bandwidth to the higher priority traffic. Additionally, conflict free access is supported, which means that the high priority traffic should not be disrupted by the lower priority traffic during channel access. However, when the higher priority nodes are not utilizing the allocated bandwidth, lower priority nodes cannot use the unused bandwidth. This can reduce the efficiency of the channel usage.

To overcome this issue, the wireless sensor network protocol presented in [21] divides the TDMA frames into a broadcast period and scheduled access period. The scheduled access sections are further divided into priority groups with multiple transmission slots in each group. Within each frame, a node broadcasts their traffic priority during the broadcast period, which reserves their transmissions for the specified priority groups. During the scheduled period, nodes send packets using one of the available slots within its priority group section. Another protocol named PMAC [22] proposes a TDMA based prioritization MAC utilizing a wide range of prefix reservation and inter-slot intervals.

In [12, 23], the proposed access strategies rely on specific types of PU networks. Protocols are developed in [12] for a slotted PU network where the SUs are synchronized with the PU's slots. In this case, if an PU does not use a slot then the SU can transmit after a small duration of time in the same slot. The PUs in [23] are assumed to use a specific combination of time and frequency multiplexing to prevent inter-PU access collisions.

TDMA based traffic prioritization provides the benefit of absolute priority for different traffic classes. A drawback of the protocol is the requirement of tight time synchronization needed to implement time slotting. Additionally, TDMA protocols can suffer from reduced bandwidth usage due to unused scheduled slots during variable bit rate traffic.

In this thesis, we formulate the PU network more generally by not assuming any specific PU MAC layer such as the slotted ones in [12]. The target here is to develop an SU access mechanism, which can deal with unslotted as well as stochastic PU MAC layers such as 802.11, CSMA and ALOHA protocols. This assumption of a general PU access protocol broadens the applicability of the SU access approach proposed in this thesis.

1.5.4 Other Secondary User Access Strategies

Game theoretic mechanisms have been introduced in [24-32] for spectrum access for the SUs. The PUs in this framework cooperates with the SUs through pricing strategies and subscription fees to facilitate spectrum sharing. These approaches require the PUs to have knowledge of the SUs presence. In contrast, the proposed mechanism in this thesis, no such cooperation is assumed, thus no changes in the PUs behavior is needed. The idea here is to develop an access mechanism for the SUs for scavenging the PUs' leftover bandwidth, without the PUs being aware of such scavenging. For example, data-enabled hand-held devices should be able to scavenge bandwidth in a WLAN without the primary VoIP handsets being aware of the scavenging process. There are also other approaches to access strategies [33-38].

1.5.5 Access Strategies Considering Multiple Secondary Users

Functionality and performance of MAC layer protocols in the presence of multiple secondary users have mostly been studied in the context of multiple channels [6, 39-42]. The authors in [39] utilizes a common control channel between the SUs. When a SU wants to send a packet it first switches to the control channel and sends an RTS. Once a CTS is received from the destination node, the two nodes sense every available PU channel then transmit on a jointly chosen channel. MAC layer protocols for multiple SUs in a single channel context have not received equal attention. The authors in [43] present a random access MAC protocol for secondary users in a single channel. Their proposed protocol utilizes a frame structure where at the beginning of each frame the SUs sense the channel for the presence of PUs. If the channel is sensed free, the SUs continuously send packets until the next frame begins. It is assumed that the whitespace will remain free for the entire duration of the frame. This protocol may not work well when PU traffic pattern are highly dynamic as in packet based primary user networks. The presented solutions in this thesis avoid this limitation through dynamically allocating SU transmissions based on the statistics of the PU traffic. Cooperative sensing have also been investigated as an approach to support multiple secondary users [44, 45].

1.6 Case Study: Coexistence in 802.11 Networks

The proposed bandwidth scavenging concept is applied to scenarios, namely, coexistence 802.11 in networks and traffic protection in CSMA networks.

1.6.1 Motivation and Applications

The recent proliferation and the subsequent overcrowding of wireless consumer technologies such as Bluetooth, ZigBee, WLAN, cordless phones, and related applications in unlicensed bands is leading to a need for prioritized coexistence of various devices [4-6]. An example of such coexistence is the sharing of the 2.4 GHz Industry Scientific and Medical (ISM) band among different device types. The ISM bands are intended for the operation of unlicensed devices, many of which are consumer level devices. Communication devices using the ISM bands must tolerate any interference from other ISM equipment. Therefore, these devices must co-exist within the band. Many home devices such as cordless telephones and car alarms utilize the 2.4 Ghz band for operation. Packet based communication device also share the ISM band. Wireless Personal Area Network (WPAN) and Wireless Local Area Networks (WLAN) applications such as Bluetooth, Zigbee, and 802.11 utilize the ISM bands for communication [7].

A suitable primary-secondary relationship among those co-existing devices are imposed based on their respective traffic priorities. For example, consider a VoIP primary user network utilizing a 2.4 GHz band. Along with the high-priority primary user devices there are coexisting low priority secondary user devices (e.g. sensors, laptops etc.) using the same band. The objective here is to design a strategy to allow the secondary users to access the channel and utilize the bandwidth without disrupting or restricting the high-priority primary user devices.

1.6.2 Problem Definition

The secondary users' goal is to access the idle times between the transmissions of primary users. These idle times or *whitespaces* typically last for short durations (i.e. order of milliseconds) in between the active data transmissions of PUs. The primary challenge in designing an access strategy for the SUs is how to minimize the disruption to the primary users while maximizing the secondary user throughput without deterministic knowledge about the future occurrences of whitespaces and their durations. With non-deterministic primary traffic patterns, an SU is not able to deterministically predict when whitespaces will occur and when a detected whitespace will end. As a result, the SU access strategy cannot be deterministic. A reasonable solution is for the SUs to access a given whitespace based on the statistical profile of the previously observed whitespaces. This requires a study of the whitespace characteristics of an 802.11 primary network so that an appropriate statistical model can be developed. Additionally, efficient access strategies will need to be developed based on the model.

1.7 Case Study: Traffic Protection in CSMA based Sensor Networks

1.7.1 Motivation and Applications

Applications utilizing Wireless Sensor Networks (WSNs) can be categorized into real time and non real time. Real time applications such as event surveillance typically require constant or variable rate data streams to be sent over the sensor network. Depending on the specific application, the rate of this data can be of the order of a few packets per second to tens of packets per second. The objective is to transport such real-time application data packets with minimum delivery delay and packet losses. Non-real time applications such as environmental monitoring need to measure levels of temperature, moisture and other ambient parameters in a time driven or event driven manner. Such applications typically require lower data rates compared to the real-time data streams and have less stringent packet delay and loss requirements.

Applications utilizing WSNs can be categorized into two classes, namely, real time applications and non real time applications. Real time applications typically require some constant rate of data to be sent over the sensor network. Depending on the application, the generation rate of this data can be on the order a few packets per second or hundreds of packets per second. For example, an environmental monitoring WSN may measure the temperature or moisture level every few seconds. Since the temperature and moisture levels do not change with large time scales, sending a few packets per second is a sufficient data rate for real time measurement of the required information. In battlefield monitoring, target surveillance, and audio or video applications data rate requirements may be higher.

Non-real time applications may be also supported by a WSN. Event driven monitoring are typical non real time applications were sensor nodes react when an event occurs within the region that is currently being monitored. Some examples of no-real time applications are structural integrity monitoring, home automation, and animal monitoring. For example, consider a sensor node that is triggered to start sending data when animal movement is detected or when the noise level of an environment reaches a certain threshold. Once the event occurs, the sensor nodes will send data about the event to the collection node or sink.

1.7.2 Problem Definition

A heterogeneous WSN may often require supporting both real time and non real time applications. In this type of networks, nodes that support real time applications must coexist with the nodes that are running non-real time application. In order to protect the real-time traffic from large channel access delay caused due to interruptions from the non real-time traffic, an access priority structure is usually needed.

Chapter 2: Dissertation Scope and Contributions

2.1 Introduction

The principal goal of a secondary user utilizing bandwidth scavenging is to access spectrum resources efficiently while avoiding disruptions to the primary user traffic. Since the SUs and PUs are completely independent of each other, primary traffic patterns are non-deterministic. This means that an SU is not able to deterministically predict when the spectrum will be free. As a result, an SU will need an access strategy that can adjust to the dynamics of the primary user traffics. A reasonable solution is for the SU to access a given idle times or whitespace based on a previously observed statistical whitespace profile. Once an SU identifies a whitespace, it sends a packet in that whitespace only if the estimated chance of completing the transmission before the whitespace will have ended is reasonably high. This opportunistic access during ultra-short and non-deterministic whitespaces is termed as *Bandwidth Scavenging* by the secondary users. In other words, the SUs scavenge the channel capacity left over by the PUs.

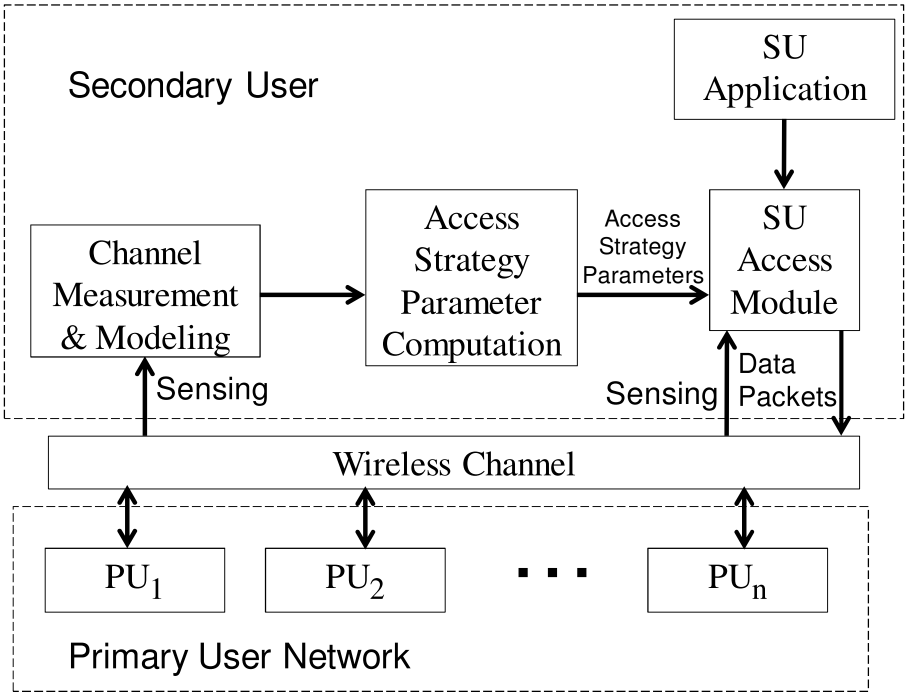


Figure 2-1: Architectural components of Bandwidth Scavenging

2.2 Bandwidth Scavenging Components

Figure 2-1 shows a functional diagram of the architectural components in the Bandwidth Scavenging Concept (BSC). The BSC operates at the Physical (PHY) and Medium Access Control (MAC) layers of the network stack. The SU continuously senses the shared channel using the Channel Measurement and Modeling module. This module creates a statistical model of PU traffic characteristic. Using the resulting model, the Access Strategy Parameter Computation module executes the access strategy algorithm and generates access parameters. The access parameters are then fed into the SU Access Module, which, in addition to channel sensing, transmits SU user packets into the channel using a designated process.

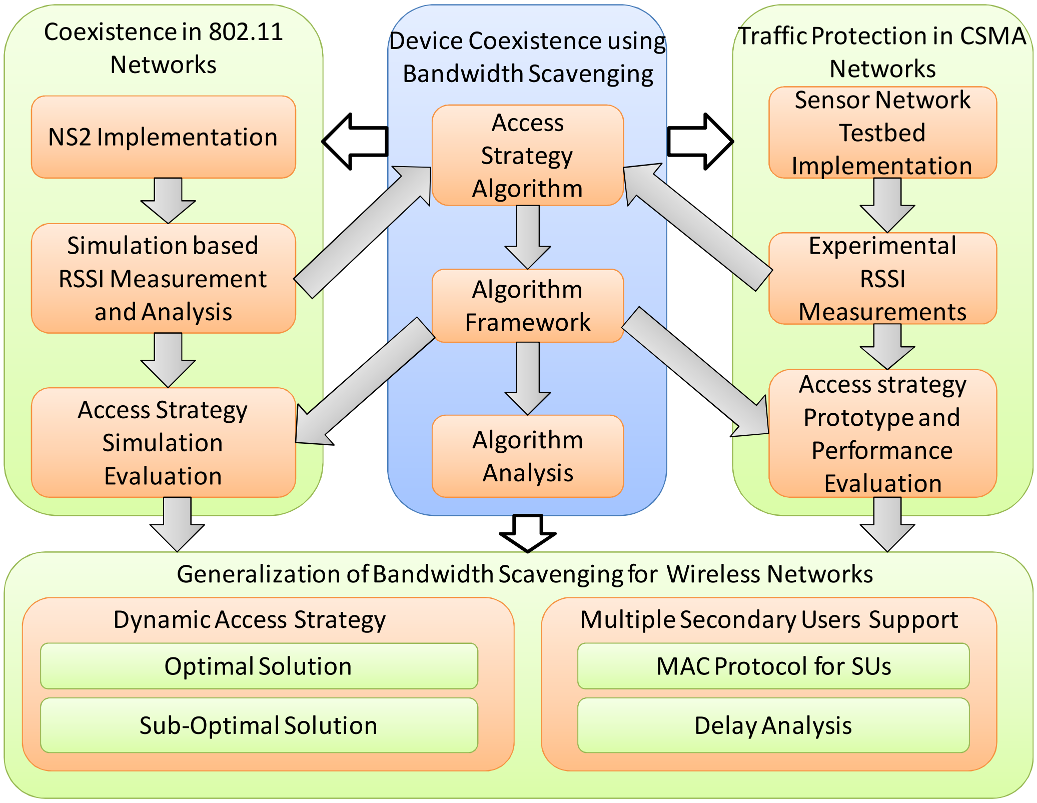

Coexistence in 802.11 Networks
NS2 Implementation
Simulation based RSSI Measurement and Analysis
Access Strategy Simulation Evaluation
Device Coexistence using Bandwidth Scavenging
Access Strategy Algorithm
Algorithm Framework
Algorithm Analysis
Traffic Protection in CSMA Networks
Sensor Network Testbed Implementation
Experimental RSSI Measurements
Access strategy Prototype and Performance Evaluation
Generalization of Bandwidth Scavenging for Wireless Networks
Dynamic Access Strategy
Optimal Solution
Sub-Optimal Solution
Multiple Secondary Users Support
MAC Protocol for SUs
Delay Analysis

Figure 2-2: Application of the Bandwidth Scavenging Concept. For interpretation of the reference to color in this and all other figures, the reader is referred to the electronic version of this dissertation.

2.3 Scope

The Bandwidth Scavenging Concept (BSC) has been applied as a solution to device coexistence in wireless networks. Figure 2-2 shows a visual summary of the application of BSC. The concept has been applied to support prioritized device coexistence in 802.11 networks, traffic protection in CSMA networks and lastly general dynamic spectrum access networks.

2.4 Contributions

2.4.1 Coexistence in 802.11 Networks

The BSC is applied to support device coexistence in 802.11 networks. Three key contributions are made in this case study. First, an extensive ns2 based simulation is performed. In this simulation, a 98 PU and SU node network is developed to investigate the interaction between PUs and SUs. Additionally, a number of PU topologies, traffic patterns, and data rates are implemented to measure the whitespace characteristics of a primary network. Second, an access strategy is developed that facilitates efficient SU channel access in an 802.11 based primary user network. Lastly, an extensive simulation and theoretical evaluation is provided using whitespace traces from the simulation.

2.4.2 Traffic Protection in CSMA based Sensor Networks

The BSC is also utilized to facilitate coexistence between real time nodes and non-real time nodes in a sensor network. Two key contributions are made in this case study. First, an TinyOS based testbed of a priority sensor node network is implemented. The testbed is able to measure the RSSI of a Zigbee channel and report this information. Second, SU access strategies are implemented and evaluated on the testbed. The testbed is able to preliminarily validate BSC on real hardware.

2.4.3 Generalization of Bandwidth Scavenging Concept for Wireless Networks

In this work, the BSC is generalized to support dynamic spectrum access in wireless networks. The BSC is specifically extended to support a generalized access strategy for any type of primary network. Additionally, the protocol is extended to support multiple secondary users.

Chapter 3: 802.11 Whitespace Analysis

3.1 Introduction

Discovery of appropriate transmission opportunities requires the secondary users to first measure and then model the occurrences and durations of whitespace, resulting from the primary users' transmissions. It follows that the whitespace patterns are a function of the physical locations, topology, traffic profile, and routing protocols used by the primary users. This chapter aims to measure, model and analyze the whitespace proprieties in an Ad-hoc 802.11 network. The chapter is organized as follows. First a description of the simulation setup used for analysis is given. Then an analysis of an Ad-hoc 802.11 network channel characteristics is presented. Finally, a whitespace model is developed and is extensively characterized.

3.1.1 Motivation

Fundamentally, the secondary users' goal is to access the idle times between the transmissions of the primary users. In the context of packet based networks, these idle times or *whitespaces* typically last for milliseconds in between the active data transmissions by the PUs. Therefore, the secondary user needs to first analyze and model the whitespace characteristics. A model of the whitespace will provide a framework that will allow the secondary user to develop access strategies that can adapt to the changing characteristics of the channel.

3.1.2 Problem Definition

With 802.11 primary traffic, the whitespaces typically last for milliseconds in between the active data transmissions by the PUs. This is in contrast to long durations of whitespaces experienced in TV bands [46]. In that context, the primary challenge for an SU is to discover unoccupied channels within the frequency domain. When an SU attempts to access TV bands, disruption to the PUs manifests in the form of 'Co-Channel' and 'Adjacent Channel' interference [11, 46]. In contrast, in accessing packet-based networks such as 802.11, the disruption is defined as when an SU transmits during a whitespace, and the time needed to successfully complete the SU transmission is more than the current whitespace duration purely from the PU traffic standpoint. Therefore, the first step is analyzing the whitespace characteristics from a temporal perspective. PU traffics can exhibit many different properties depending on the topology, traffic pattern, and data rates. These different properties require an SU to define a proper model of the whitespace that can adjust to the different characteristics. This chapter aims to present that model.

3.1.3 Related work

Networks with primary users running 802.11 MAC protocol have recently been investigated for possible access by secondary users. The authors in [16, 17] develop a methodology for formally analyzing the whitespace available within 802.11 primary traffic in infrastructure mode. The key idea is to model the whitespace as a semi-Markov process that relies on the underlying 802.11 state model involving DIFS, SIFS, DATA, and ACK transactions. The model describes the whitespace profile in terms of holding times of the idle and busy states of the channel.

The traffic profiles of packet based primary users studied in literature have varied greatly [12-15]. The authors in [15] model the primary users' packet arrival intervals and packet durations as two separate Poisson processes. Similarly, [13] uses an alternating ON-OFF pattern, where the duration of the idle and busy states follows an exponential distribution with means v and l respectively. In [12], a Markov chain with known transition probabilities is used to model primary user behavior. Lastly in [14], a Markov model with Exponential, Uniform, Weibull and Generalized Pareto distributions are considered for PUs arrival times. In line with these efforts in the literature, the proposed SU access mechanism in this thesis is evaluated for a variety of higher layer PU traffic profiles. However, in contrast to these works, we investigate a measurement-and–model based approach for modeling PUs' traffic. This approach captures the impacts of network dynamics, which may not be captured in the model-only (e.g. 2-state Markov [12, 14]) approaches in the literature. Other analysis of PU networks have been investigated [47-54].

3.2 Simulation Setup

The goal of the secondary users is the characterization of the whitespace left over from primary users transmissions. A method for measuring the available whitespace is through detecting the received signal strength (RSSI) at a given channel frequency [16]. We implemented an 802.11 RSSI measurement scheme in the network simulator ns-2 in order to capture the durations of whitespaces between PUs' transmissions.

3.2.1 NS2 Implementation

Ns-2 is an event-driven C++ and TCL language based simulator. It is open source and can be extended to implement many wired and wireless network protocols. Ns2 is used to provide a framework for the PU and SU topology, 802.11 traffic profiles, and the capturing of whitespace durations.

3.2.1.1 Primary and Secondary Users Topologies

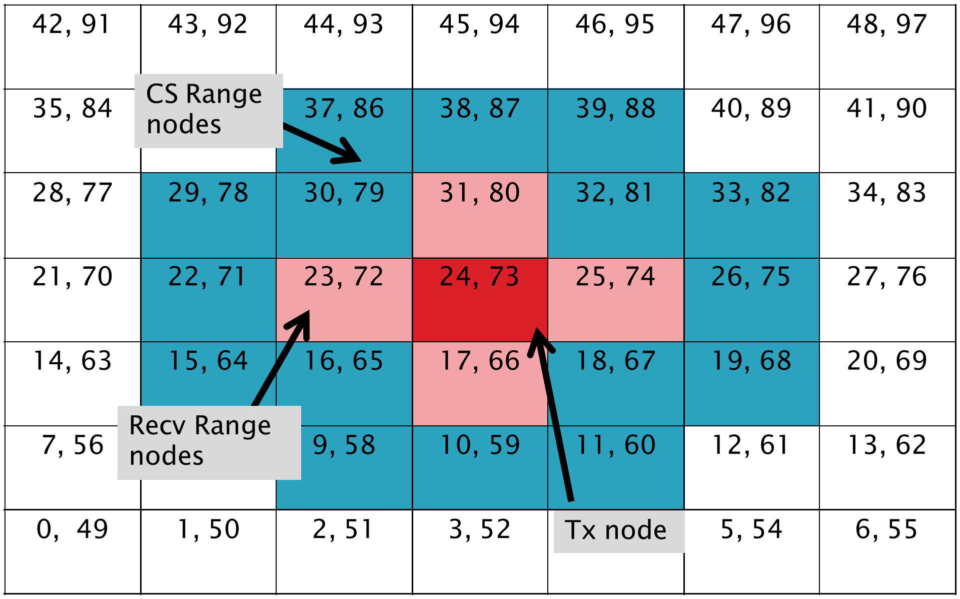
In order to control all the characteristics of the PUs and SUs network a grid topology is used. Figure 3-1 shows topology of the PUs and SUs. There are 98 wireless nodes within an area of 2000 x 2000 meter grid. The nodes designations are split evenly between the secondary users and primary users. A single PU and a single SU occupy a cell in the grid. PUs are numbered from 0 to 48 while the SUs are numbered from 49 to 97. PUs are only able to communicate with other PUs, similarly SUs can only communicate with other SUs.

42, 91	43, 92	44, 93	45, 94	46, 95	47, 96	48, 97
35, 84	36, 85	37, 86	38, 87	39, 88	40, 89	41, 90
28, 77	29, 78	30, 79	31, 80			34, 83
21, 70	22, 71	23, 72	24, 73			27, 76
14, 63	15, 64	16, 65	17, 66	18, 67	19, 68	20, 69
7, 56	8, 57	9, 58	10, 59	11, 60	12, 61	13, 62
0, 49	1, 50	2, 51	3, 52	4, 53	5, 54	6, 55

Primary User node IDs are from 0 to 48

Secondary User node IDs are from 49 to 97

Figure 3-1: Primary and Secondary Users grid topology



42, 91	43, 92	44, 93	45, 94	46, 95	47, 96	48, 97
35, 84		37, 86	38, 87	39, 88	40, 89	41, 90
28, 77	29, 78	30, 79	31, 80	32, 81	33, 82	34, 83
21, 70	22, 71	23, 72	24, 73	25, 74	26, 75	27, 76
14, 63	15, 64	16, 65	17, 66	18, 67	19, 68	20, 69
7, 56		9, 58	10, 59	11, 60	12, 61	13, 62
0, 49	1, 50	2, 51	3, 52		5, 54	6, 55

Figure 3-2: Primary and Secondary Users interference ranges

Transmission ranges of the PUs and SUs are set to 250 meters and the carrier sense range is set to 550 meters. The nodes are located within the grid to control the neighbors of each node in the network. Figure 3-2 shows an example interference range of nodes 24 (or 73) within the grid. All nodes, PUs and SUs, share a single channel. Nodes are only allowed to communicate with their adjacent neighbors in the orthogonal directions (i.e. nodes 17, 23, 25, 31). All other nodes within the nodes two-hop neighborhood are within the carrier sense range and can only interfere with the nodes transmissions as shown in Figure 3-2.

Routing for all messages is implemented on the orthogonal paths between the nodes. Paths can be bi-directional in any orthogonal direction. PUs and SUs routing paths can be completely independent of each other. Various applications can be sent over both the PUs and SUs nodes.

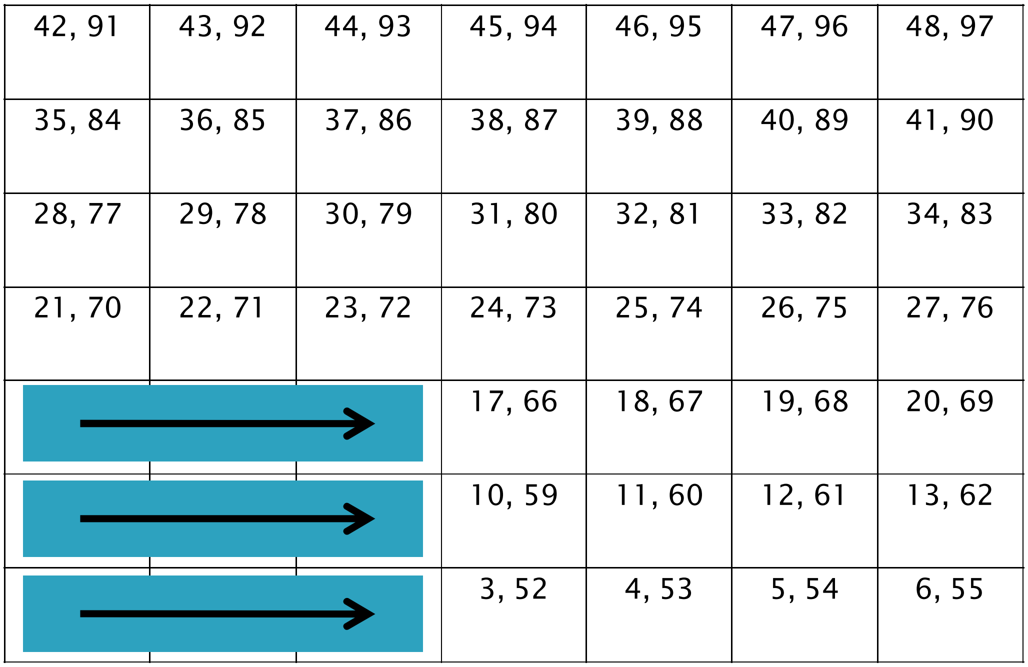
3.2.1.2 Channel Measurement

Each secondary user has the ability to measure the Received Signal Strength Indicator (RSSI) of the channel within its Carrier Sense (CS) range. We assume there is interference on a node if an SU or PU is transmitting within the CS range of the node. The CS range is chosen as opposed to the reception range, since the SUs within up to the CS range of an PU are affected by the PU's signals and vice-versa.

The RSSI measurements are captured within ns-2 in the wireless-phy.cc file. In the file, the signal strength of all messages that a node receives is reported. If the signal strength is above the CS range threshold, the value and duration of the signal is output to a file. The RSSI threshold used in the simulation to determine whitespaces is the RSSI value of -108 dBm, which corresponds to the Carrier Sense (CS) range defined by the 802.11 standard.

3.2.2 Sample Experiments with PUs and SUs topology

In order to validate the simulation setup, few sample topologies were created and whitespace measurements were captured. The traffic flow was set to UDP traffic with a 200 byte packet sizes. The packet arrival interval time for the PU flows was set to 0.07 ms. Whitespace measurements are captured from secondary nodes in the network.



42, 91	43, 92	44, 93	45, 94	46, 95	47, 96	48, 97
35, 84	36, 85	37, 86	38, 87	39, 88	40, 89	41, 90
28, 77	29, 78	30, 79	31, 80	32, 81	33, 82	34, 83
21, 70	22, 71	23, 72	24, 73	25, 74	26, 75	27, 76
→			17, 66	18, 67	19, 68	20, 69
→			10, 59	11, 60	12, 61	13, 62
→			3, 52	4, 53	5, 54	6, 55

Figure 3-3: Primary and Secondary Users interference ranges for three 2-hop PU horizontal connections

Figure 3-4 shows the whitespace measurements from the experiment. In the figure, a 1 value on the y-axis represents the noise floor where there is no signal at that measurement time and a 3 represents a signal on the channel. Observe that the whitespace is captured in between PU transmissions over time by both of the nodes.

Figure 3-5 shows the average whitespace available to all the secondary users in the network. Observe that the SUs close to the PU traffic have reduced available whitespace since the channel is occupied by the PU. With the current PU traffic load, approximately half of the channel is being utilized. This means that 50% of the whitespace is left over in the channel. Figure 3-6 and Figure 3-7 shows the average whitespace available to SUs when there are five 4-hop horizontal connections. Horizontal connections are referred to as connections from the far left nodes in the grid to the right nodes in each row as indicated by the block in the figure. Observe that the larger the amounts of PU connections are affecting a higher amount of SU nodes. SUs that are near the outside PUs nodes transmissions gradually gain more whitespace since they are being affected by a lower amount of PUs. Similarly, Figure 3-8 and Figure 3-9 shows the whitespace for connections spanning the entire PU network. In this case, all SUs are affected with the exception of the SUs near the right edge. Finally, Figure 3-10 and Figure 3-11 shows the connections for two 6-hop PU horizontal connections. Here only the SUs within the transmission ranges of the PUs are affected by the PUs. The presented figures validate the experimental setup ability to capture the whitespace of PUs during the simulation. In addition, the figures show that the SUs proximity to PUs affects the whitespace that is available for usage.

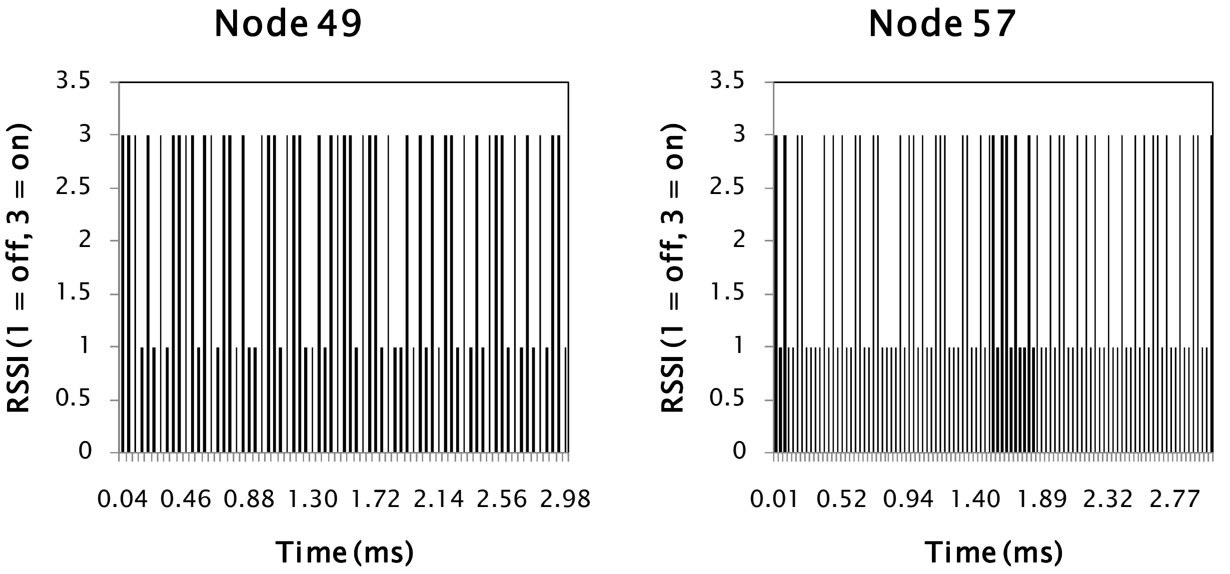


Figure 3-4: Secondary Users whitespace measurements for three 2-hop PU horizontal connections

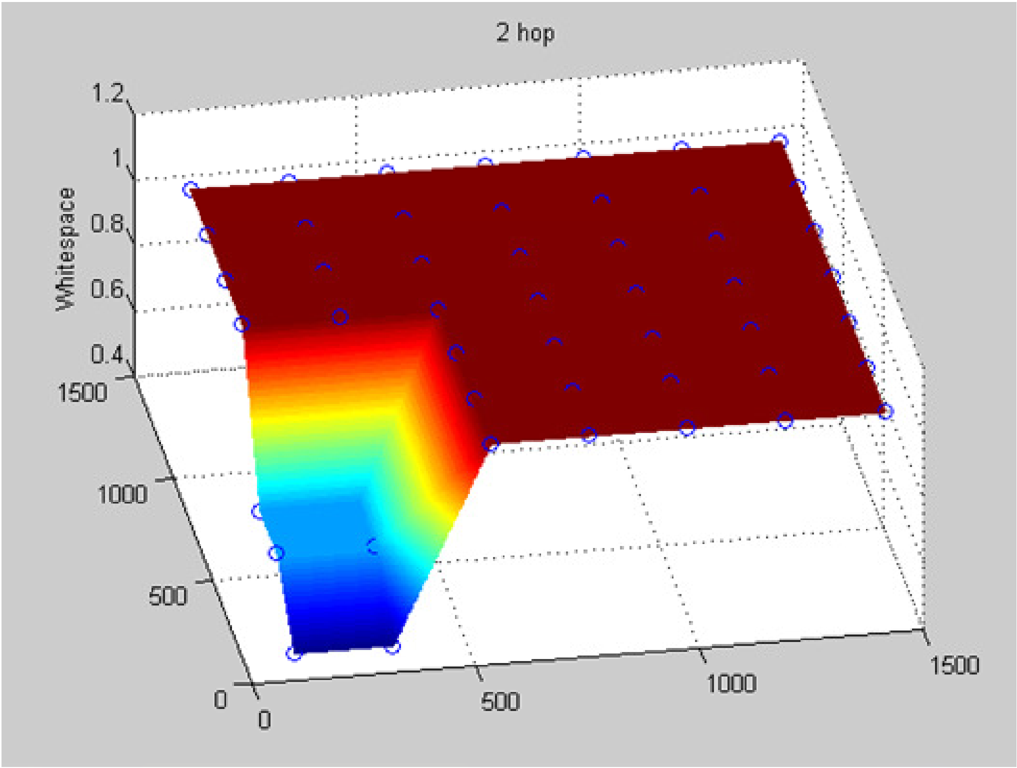
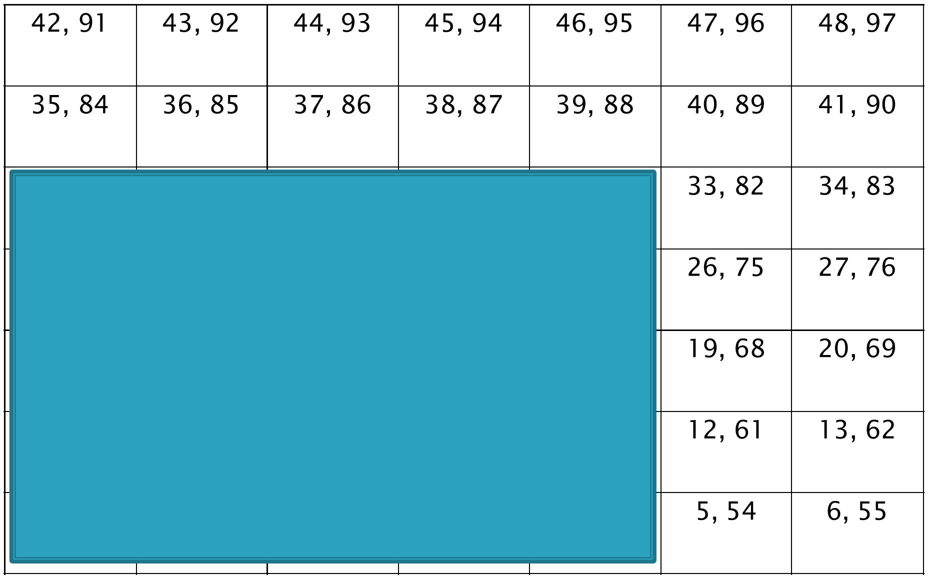


Figure 3-5: Secondary Users whitespace measurements over entire network for three 2-hop PU horizontal connections



42, 91	43, 92	44, 93	45, 94	46, 95	47, 96	48, 97
35, 84	36, 85	37, 86	38, 87	39, 88	40, 89	41, 90
					33, 82	34, 83
					26, 75	27, 76
					19, 68	20, 69
					12, 61	13, 62
					5, 54	6, 55

Figure 3-6: Primary and Secondary Users interference ranges for five 4-hop PU horizontal connections

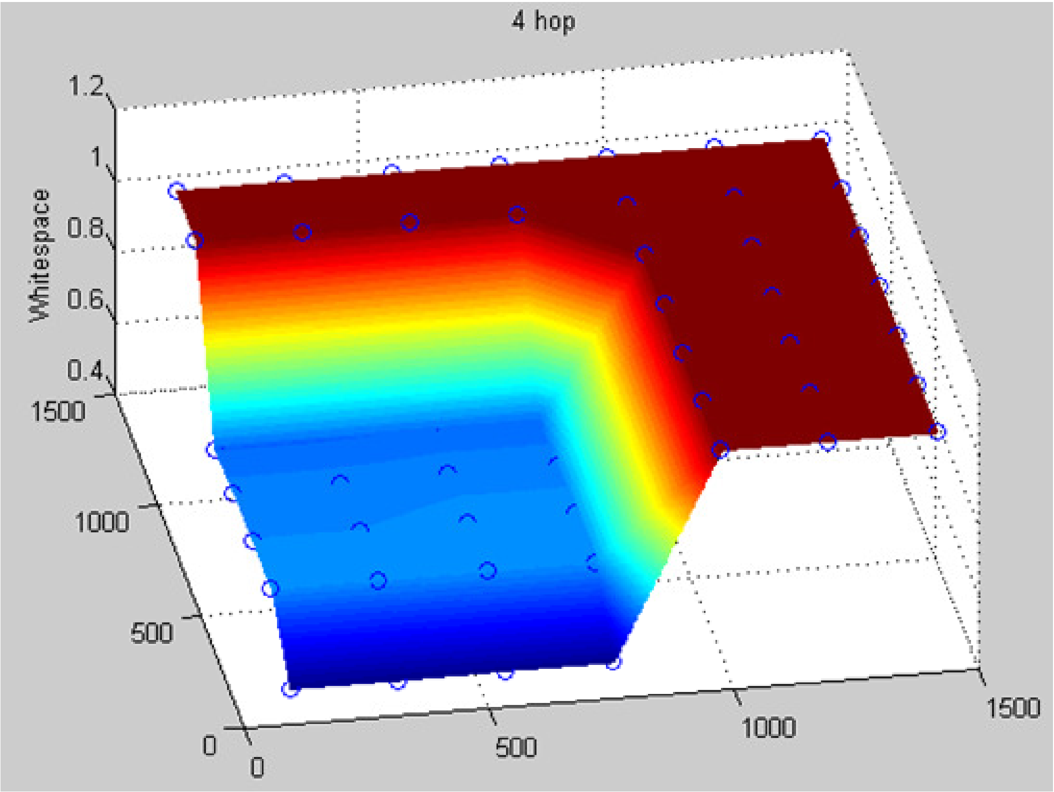


Figure 3-7: Secondary Users whitespace measurements over entire network for five 4-hop PU horizontal connections

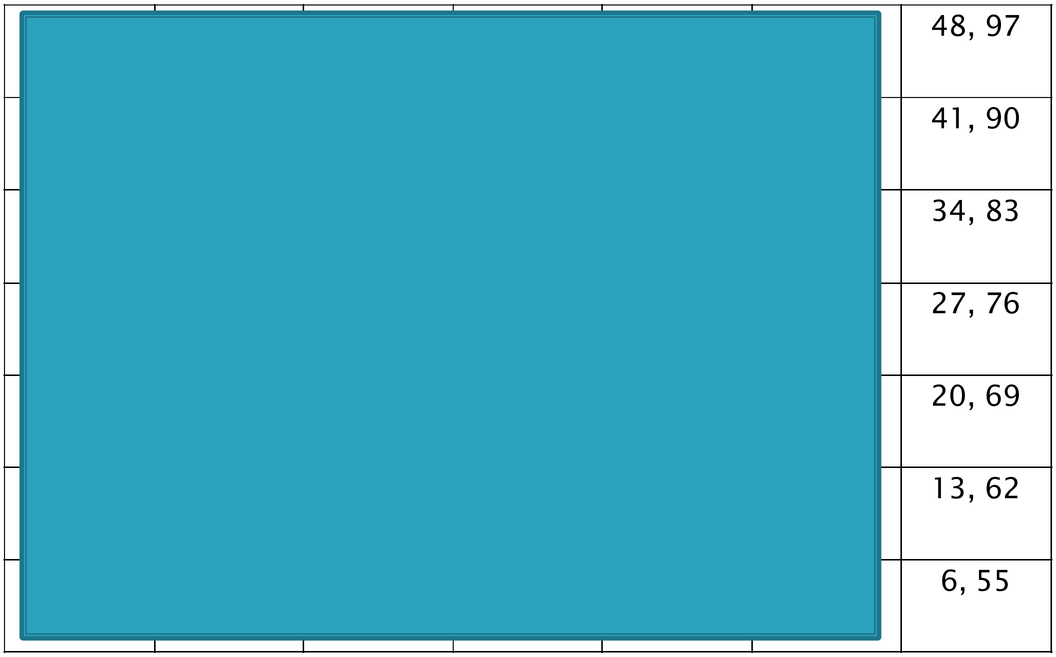


Figure 3-8: Primary and Secondary Users interference ranges for seven 6-hop PU horizontal connections

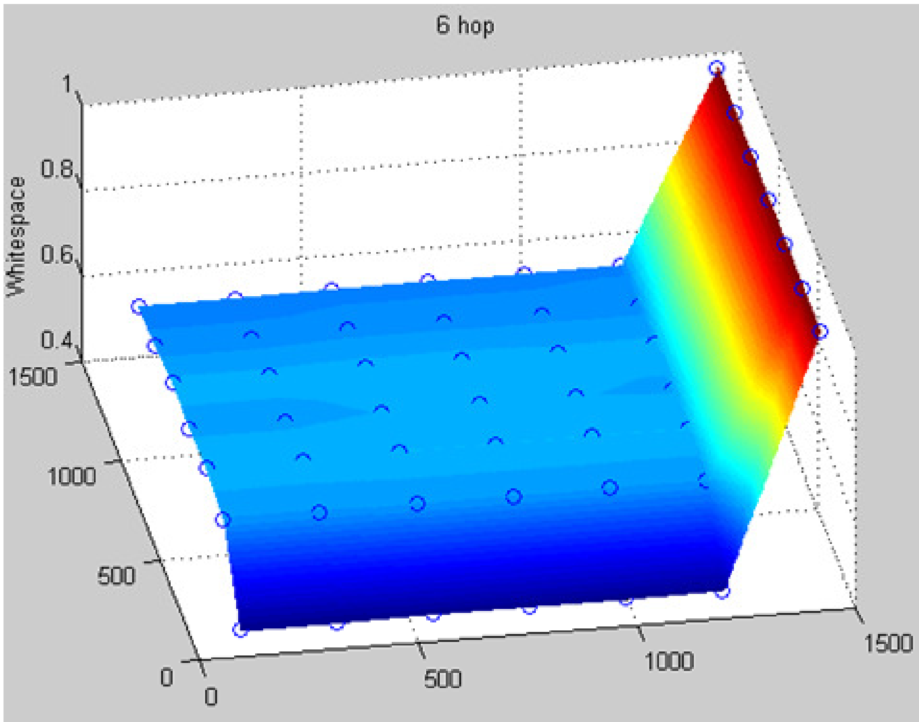
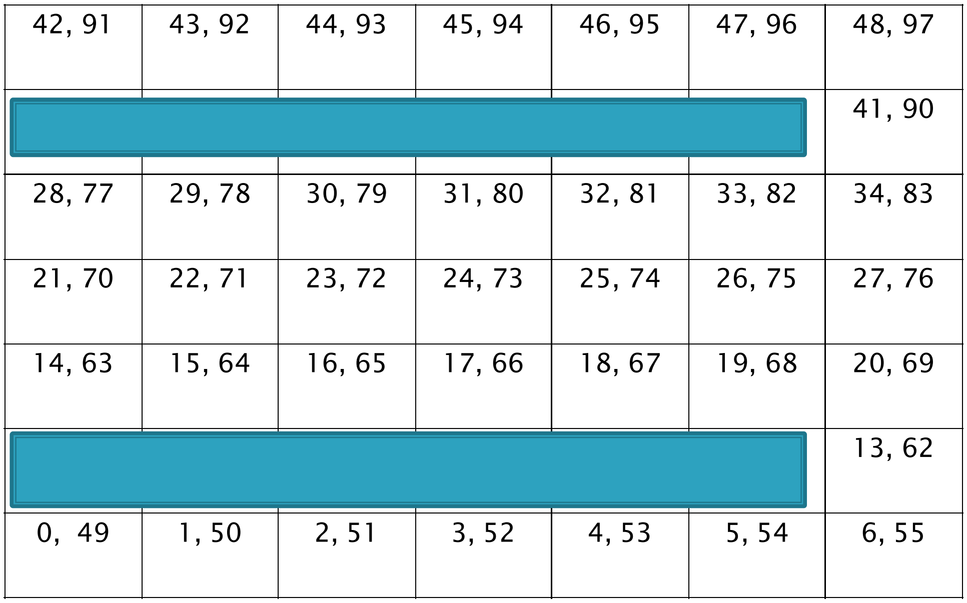


Figure 3-9: Secondary Users whitespace measurements over entire network for seven 6-hop PU horizontal connections



42, 91	43, 92	44, 93	45, 94	46, 95	47, 96	48, 97
						41, 90
28, 77	29, 78	30, 79	31, 80	32, 81	33, 82	34, 83
21, 70	22, 71	23, 72	24, 73	25, 74	26, 75	27, 76
14, 63	15, 64	16, 65	17, 66	18, 67	19, 68	20, 69
						13, 62
0, 49	1, 50	2, 51	3, 52	4, 53	5, 54	6, 55

Figure 3-10: Primary and Secondary Users interference ranges for two 5-hop PU horizontal connections

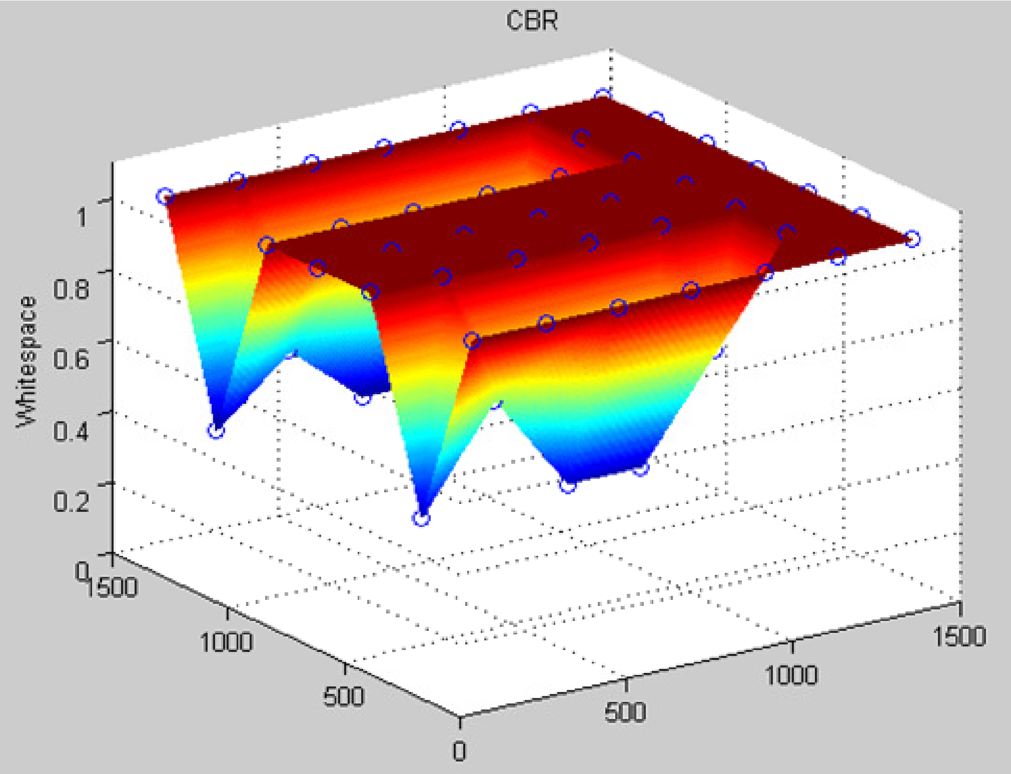


Figure 3-11: Secondary Users whitespace measurements over entire network for two 5-hop PU horizontal connections

3.3 Ad-hoc Primary User Network

In the previous section, ns-2 based PU and SU networks simulation were established. The next step is to apply some specific 802.11 based network topologies and traffic characteristics in order to capture the whitespace characteristics of an Ad-hoc 802.11 channel.

3.3.1 PU and SU Topologies

Using the PU and SU network shown in Figure 3-1, three scenarios are established. First, a linear chain PU topology with a single bi-directional multi-hop primary flow is created. Figure 3-12 shows the topology and traffic direction for the chain topology. This scenario is referred to as TOP1. Note that the arrows show the direction of the traffic. In the second scenario (TOP2) shown in Figure 3-13, there are two multi-hop bi-directional flows in the network on two intersecting linear chains. In the last scenario (TOP3) shown in Figure 3-14, there are two multi-hop flows in a parallel chain network. In each case, the secondary users are within the CS range of all primary users.

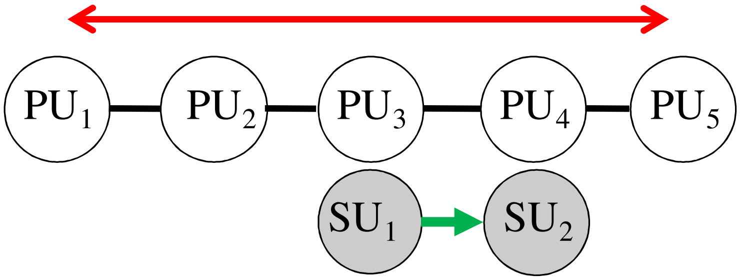


Figure 3-12: PUs and SUs topology and traffic flow for linear chain. (TOP1)

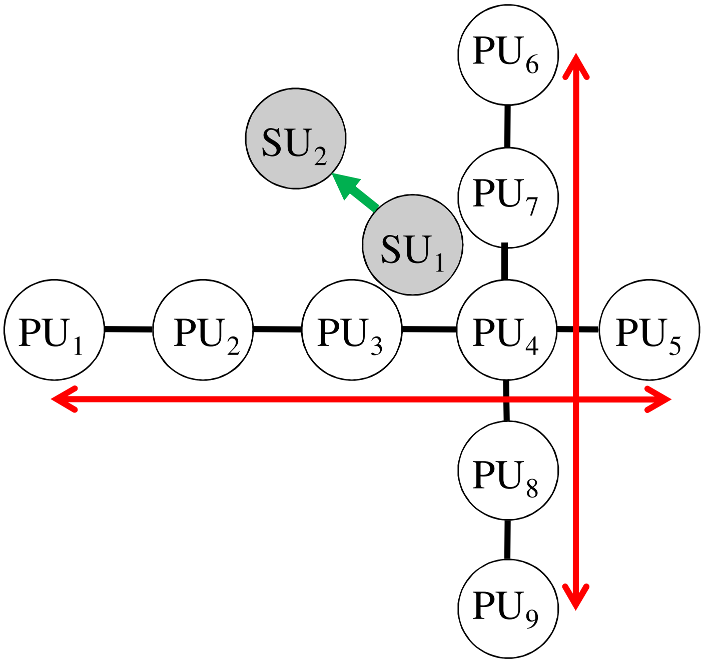


Figure 3-13: PUs and SUs topology and traffic flow for intersecting chain. (TOP2)

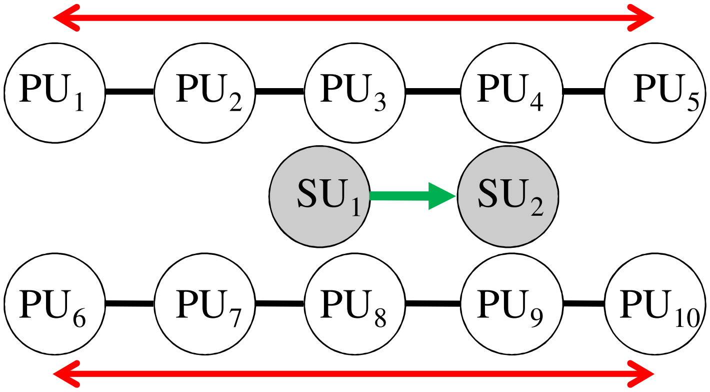


Figure 3-14: PUs and SUs topology and traffic flow for parallel chain. (TOP3)

3.3.2 Traffic Profiles

The traffic profile and data rate of the primary user directly affect the total amount and statistical properties of the available whitespace as viewed by the secondary users. We model three types of PU traffic, namely, bidirectional UDP with a fixed packet size, TCP, and a Video

Stream. For the UDP traffic, the packet arrival intervals are modeled as: 1) Uniformly distributed with a mean of α ms and a variability of $\pm v$ ms around the mean, and 2) a Poisson process with mean packet arrival interval of α ms. The packet arrival intervals for each Uniform and Poisson flow is changed in order to represent varying intensity of the primary traffic.

3.3.3 Channel Characteristics

In this section, the channel characteristics of the previously mentioned traffic profiles are analyzed. The channel measurements of SU_1 is shown in Figure 3-15 for UDP traffic sent for 0.5 inter-packet time (IPT) over 10 seconds. Figure 3-16 shows measurements with the IPT decreased to 0.1, which is twice as fast as the previous case. Notice that there are a larger number of transmissions with the faster PU IPT. Figure 3-17 and Figure 3-18 shows a zoomed in view of the measurements. Observe that each multi-hop primary user transmission is captured by the SU. With these measurements, each whitespace can be identified by the secondary user.

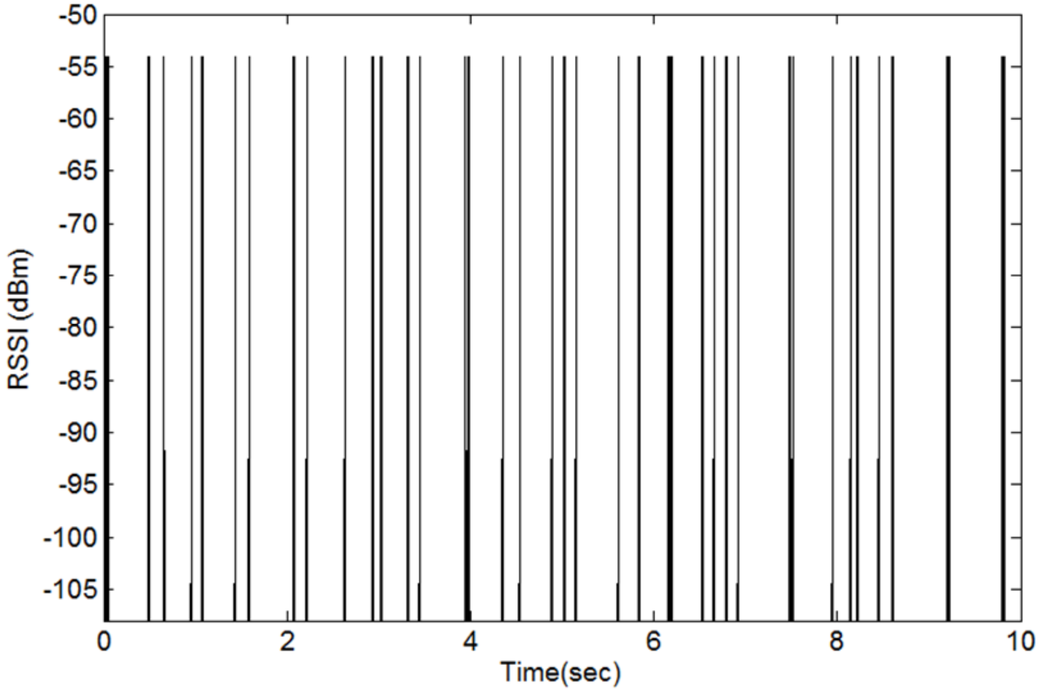


Figure 3-15: Channel measurements for 0.5 inter-packet time over 10 seconds

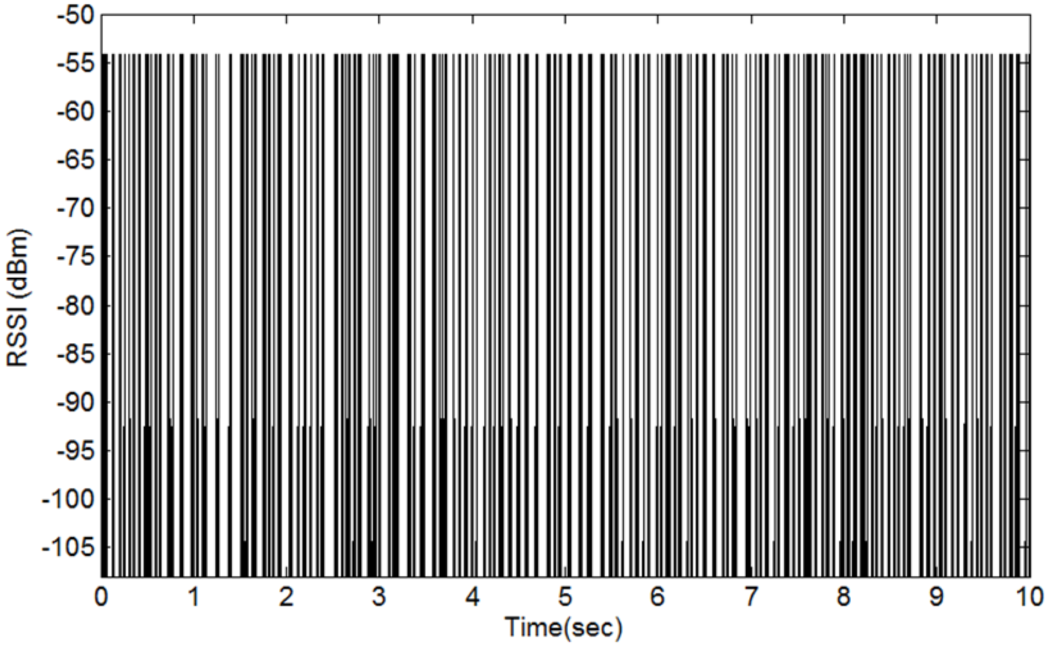


Figure 3-16: Channel measurements for 0.1 inter-packet time over 10 seconds

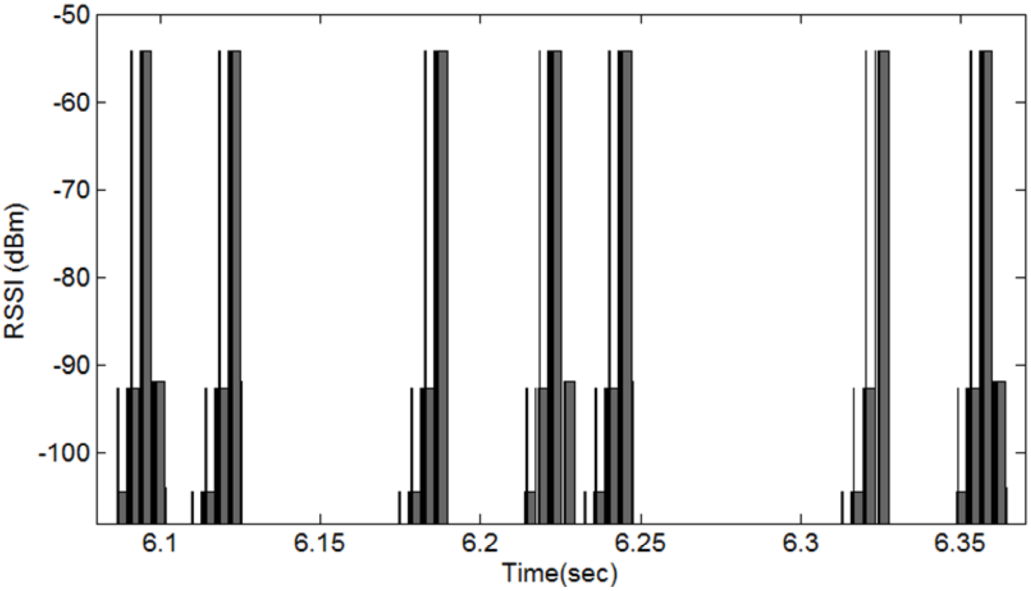


Figure 3-17: Channel measurements for 0.1 inter-packet time over 0.3 seconds

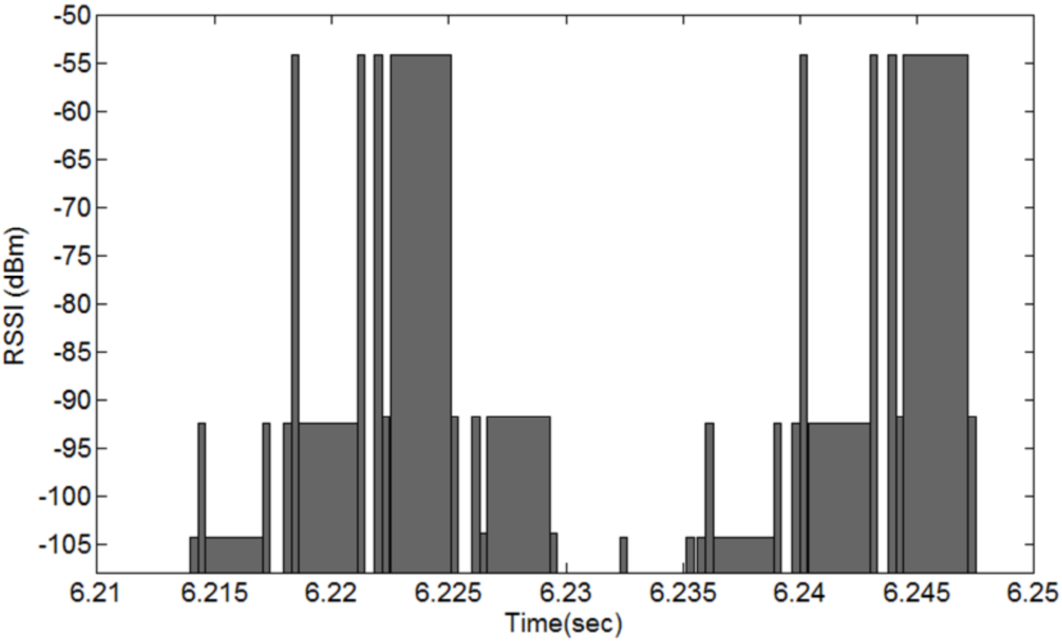


Figure 3-18: Channel measurements for 0.1 inter-packet time over 0.05 seconds

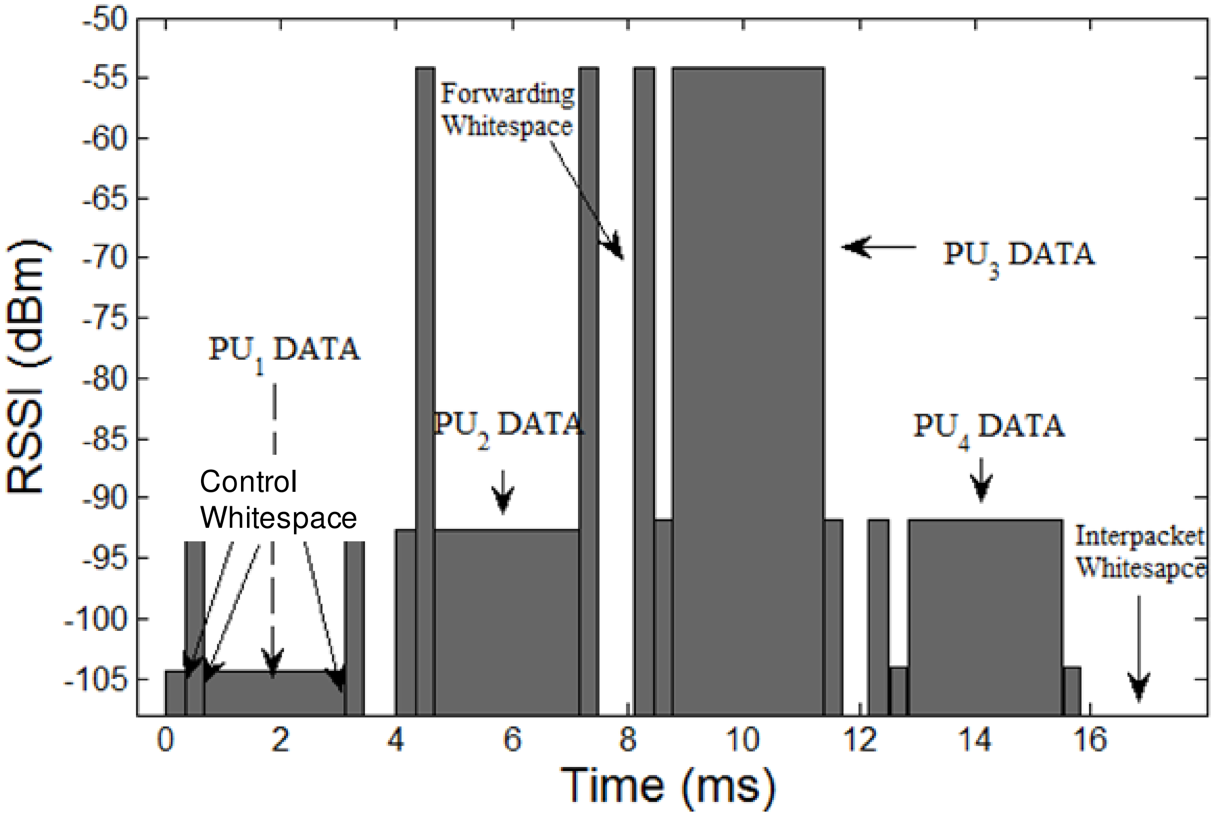


Figure 3-19: Channel measurements for 0.1 inter-packet time

Figure 3-19 shows the RSSI trace, as observed by SU$_1$, as a result of a single packet flow from PU$_1$ to PU$_5$. Observe that since SU$_1$ is physically closest to the primary user PU$_3$, the transmissions from that primary user appears with highest signal strength compared to the other PUs whose signal strength monotonically reduces with increasing distances. Also, since the PUs in this experiment use 802.11 as the MAC protocol, each packet transmission creates an RTS-CTS-DATA-ACK cycle, resulting in white spaces of multiple durations representing different 802.11 inter frame spacing (i.e. SIFS, DIFS, etc.).

Three types of whitespaces can be observed for this example 802.11 primary traffic. As pointed in Figure 3-19, the first is the *control whitespace* caused by the short inter frame spacing (SIFS) periods between the RTS, CTS, DATA, and ACK transmissions. The second whitespace type, termed as *forwarding whitespace*, represents the duration between an PU receiving data and forwarding data. The last whitespace type is the *inter-packet whitespace*, representing the inter packet duration which can vary greatly because it is based on the rate of packet transmission by the PUs.

3.4 Whitespace Characteristics

In the previous section, the channel characteristics were shown for the presented topologies. The next step is to define the whitespace model. Once the model for a measured whitespace profile is computed, the SUs rely solely on this function for channel access without having to worry about the individual PU properties such as topology, traffic pattern, and data rates.

3.4.1 Whitespace Model

An SU periodically senses the channel to detect the whitespaces. Based on the channel sampling report, the channel state is interpreted using an ON-OFF (busy-idle) model as shown in Figure 3-20. The sensing interval of the channel is represented by T_p and the sensing time is represented by T_s. Figure 3-21 shows an example of the periodic sensing process. We assume that the sensing time duration is negligible. After each measurement, a busy or idle report is provided. Consecutive idle measurements are counted (n) and multiplied by T_p to determine the duration of a whitespace. We model the duration of whitespace using its probability density function, *w(n)*, which is the probability of a whitespace being nT_p time long [55]. This *w(n)* is computed periodically to capture any changes in the primary user behavior. Once the *w(n)* for a measured whitespace is computed, the SUs rely solely on this function for channel access without having to worry about the individual PU properties such as topology, traffic pattern, and data rates.

Note that the proposed SU access mechanism in this thesis assumes a certain degree of stationarity of the distribution *w(n)* [12, 14, 15, 56]. Meaning, the PU network topology and traffic behavior remains stationary for long enough duration so that the SUs can make use of *w(n)* to make statistically meaningful access decisions.

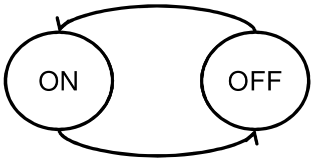


Figure 3-20: On-Off whitespace model

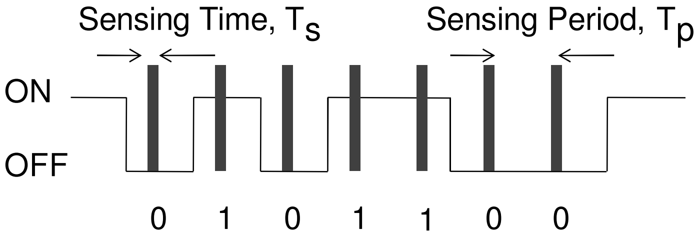


Figure 3-21: Sensing period

3.4.2 Whitespace Statistics

Utilizing the whitespace pdf *w(n)*, varying topologies, traffic rates, and traffic distributions are shown in this section. For these whitespace measurements, the secondary user's channel sensing interval T_p is kept smaller than the shortest duration of possible whitespace in the channel. Considering that for 802.11 [57] the shortest duration whitespace is the SIFS period at 10 µs, the value of T_p was set to 5µs to capture all possible whitespaces. Note that in the legend used in most of the figures, a convention [*topology, traffic profile - packet arrival interval (in ms)*] is used for describing the corresponding scenarios. For example, [TOP1, U90] would represent a whitespace profile obtained when a packet stream with uniformly distributed inter-packet interval (of 90 ms) is sent over a network with TOP1.

3.4.2.1 Uniform and Poisson Distributions

Figure 3-22 shows the *w(n)* for UDP traffic with 90 ms average inter-packet duration that is uniformly distributed across ±20% around the average packet arrival interval for each topology. Observe that the detected whitespace durations from the measured RSSI trace vary a great deal. For example, in the multi-hop TOP2 case, although all the traffic is generated at about 90 ms intervals, the whitespace durations are distributed over the range from 10 μs to 300 ms. This variation results from the dynamic nature of the bidirectional multi-hop primary traffic. Figure 3-23 shows the whitespace for primary traffic with arrival times as an Exponential distribution (i.e. a Poisson process based packet arrival) with 90 ms inter-packet arrival for each topology. Similar to the Uniform case the whitespace distribution varies across a wide range of durations.

Observe that in the pdf figures there is a peak at the on the first millisecond. To highlight this initial peak in the pdf, Figure 3-24 and Figure 3-25 shows the cumulative density function (cdf) of the Uniform and Poisson experiments on the interval from 0 ms 1 ms. From the figures, it is interesting to note that regardless of the traffic profile or topology, the vast majority of the whitespaces (around 95%) last for less than 1 ms.

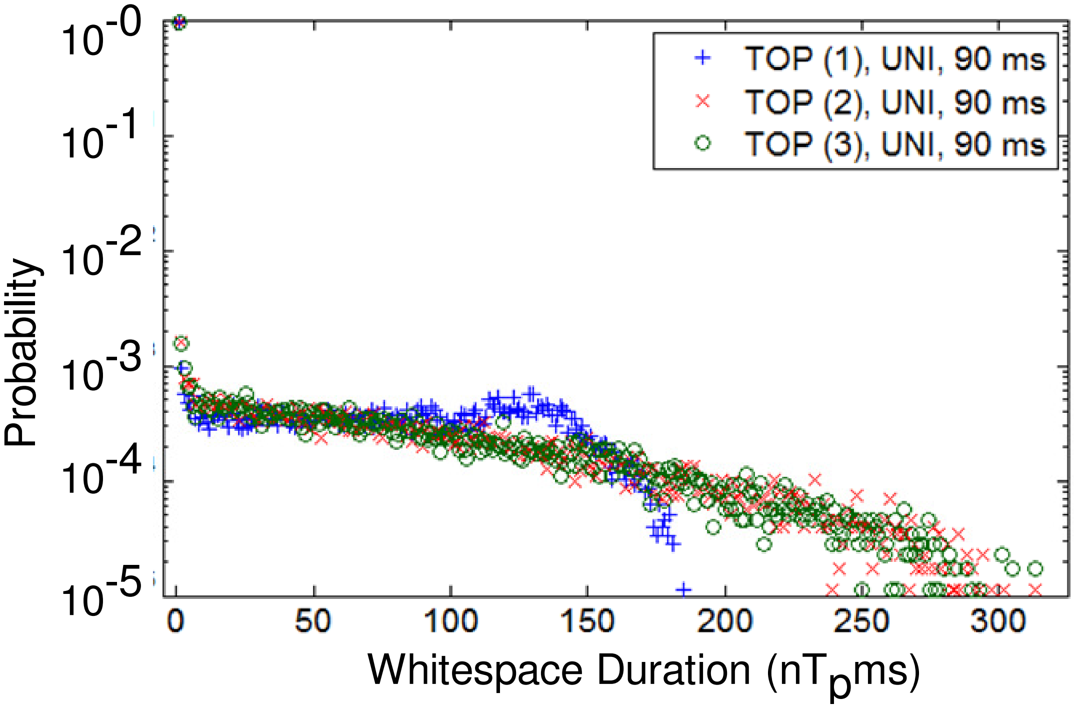


Figure 3-22: Impacts of Uniform whitespace pdf distribution

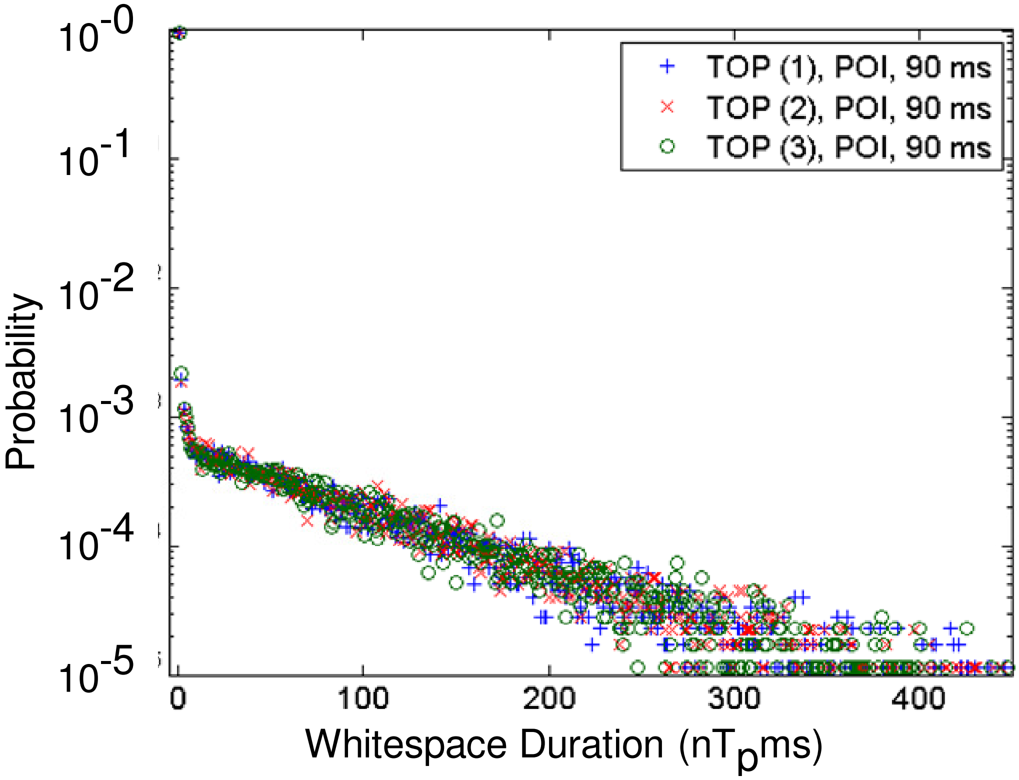


Figure 3-23: Impacts of Poisson whitespace pdf distribution

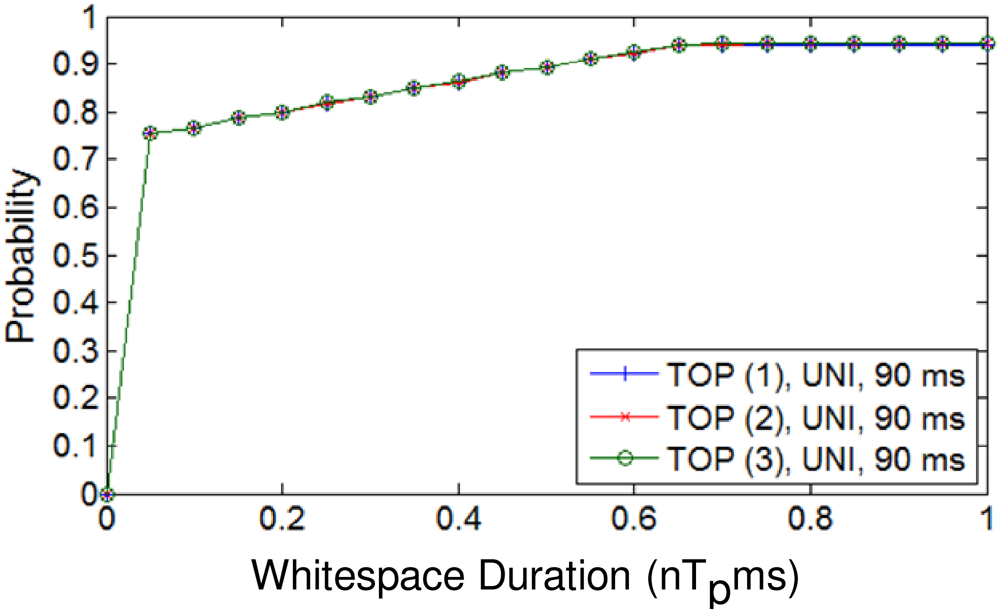


Figure 3-24: Uniform whitespace cdf distribution within the interval 0 ms to 1 ms

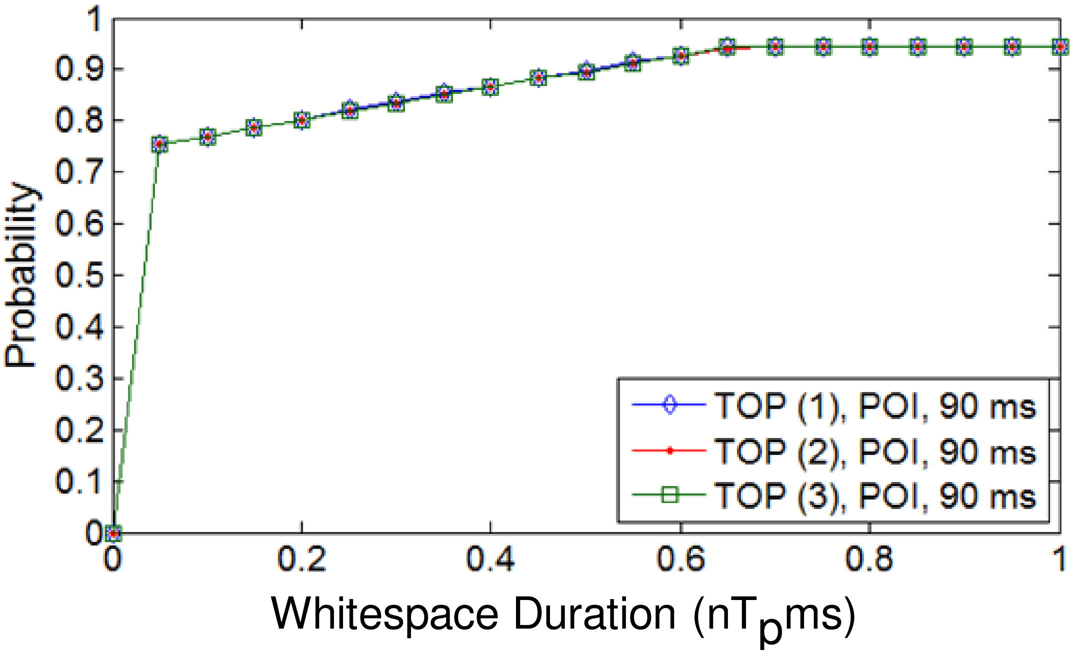


Figure 3-25: Poisson whitespace cdf distribution within the interval 0 ms to 1 msTCP and Video Distributions

Figure 3-26 and Figure 3-27 present the *w(n)* for TCP and Video Streams. In the TCP traffic case, over all the topologies, the whitespace durations are very small. All of the whitespace durations are less than 20 ms long. This would indicate very little opportunity for secondary user access to the spectrum under TCP traffic. This observation is in line with what was reported in studies of the impacts of the TCP protocol in DSA networks [58]. The video stream case shows that there is a larger amount of whitespace available compared to the TCP case. Figure 3-28 and Figure 3-29 shows cdf of the first millisecond for the TCP and video stream cases. Observe again that the whitespace properties are similar to the Uniform and Poisson cases. Where upwards of 95% of the whitespace duration is below 1 ms.

3.4.2.2 Impacts of Data Rates

The impacts of data rates of the primary user traffic on the whitespace distribution are shown in Figure 3-30 and Figure 3-31. The figures present the packet arrival intervals of 60 ms and 90 ms for TOP3 with intersecting multi-hop flows, for both Uniform and Poisson distributions. The figures show that depending on the specific average inter-packet time of the primary traffic, the whitespace durations will be distributed over a larger range. It follows that a faster PU data rate will reduce the amount of whitespace available to the SU. Again, observe that in Figure 3-32, the whitespace trends for all cases from 0 ms to 1 ms have properties that are similar to the previous figures.

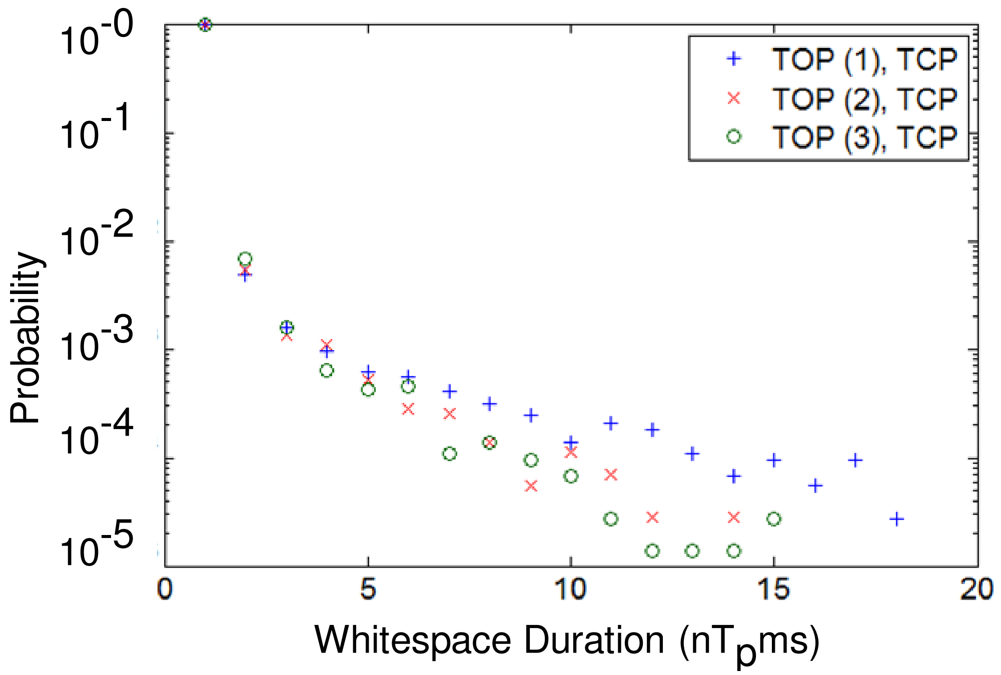


Figure 3-26: Impacts of TCP whitespace pdf distribution

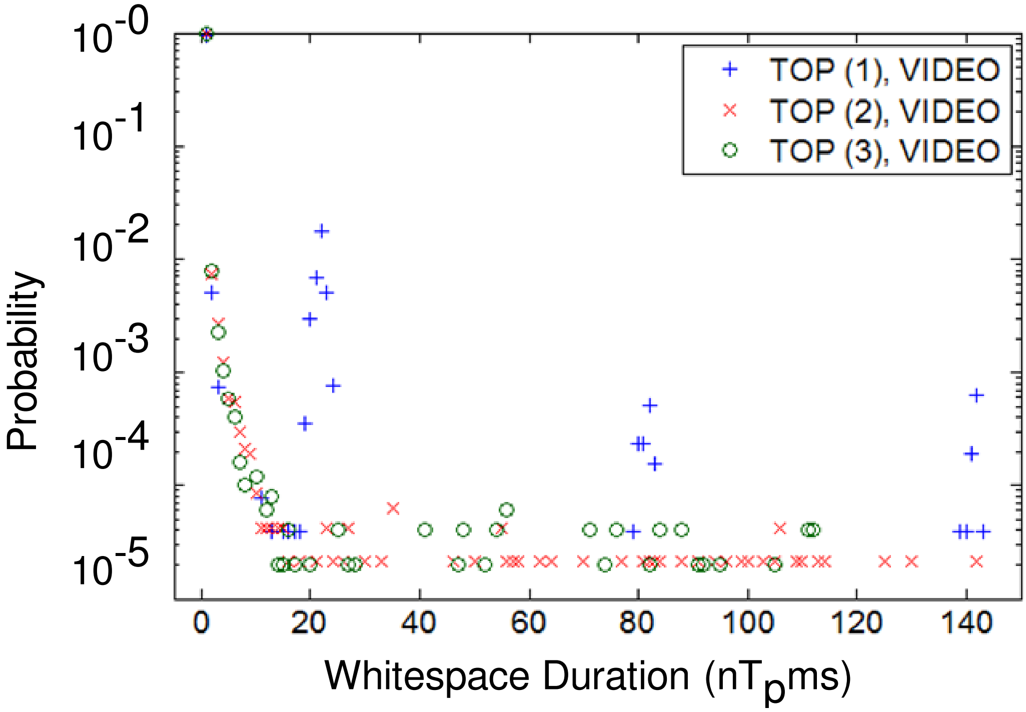


Figure 3-27: Impacts of video stream whitespace pdf distribution

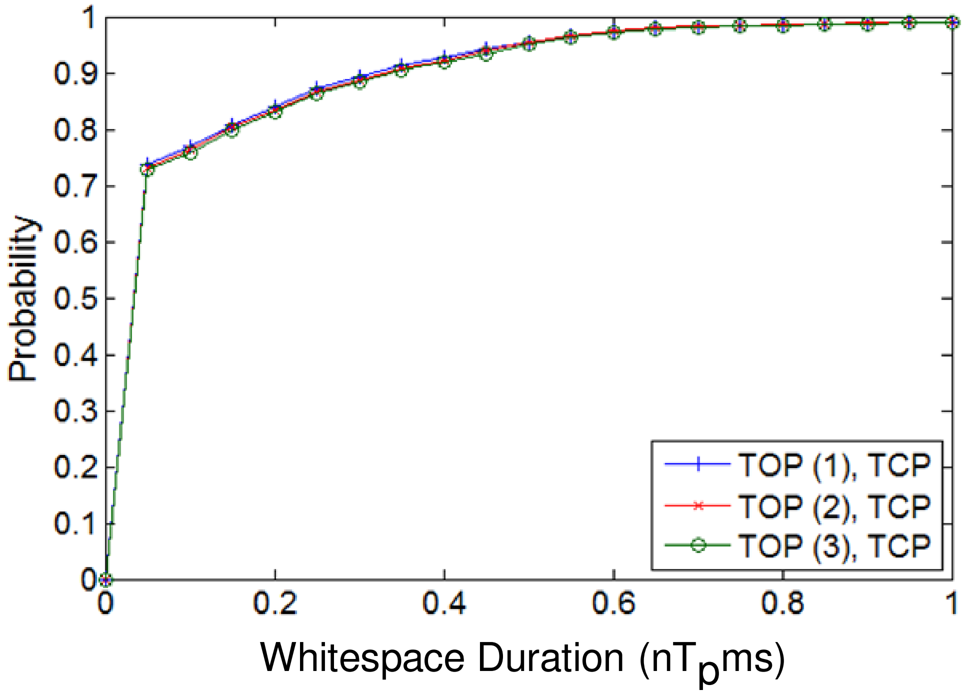


Figure 3-28: TCP whitespace cdf distribution within the interval 0 ms to 1 ms

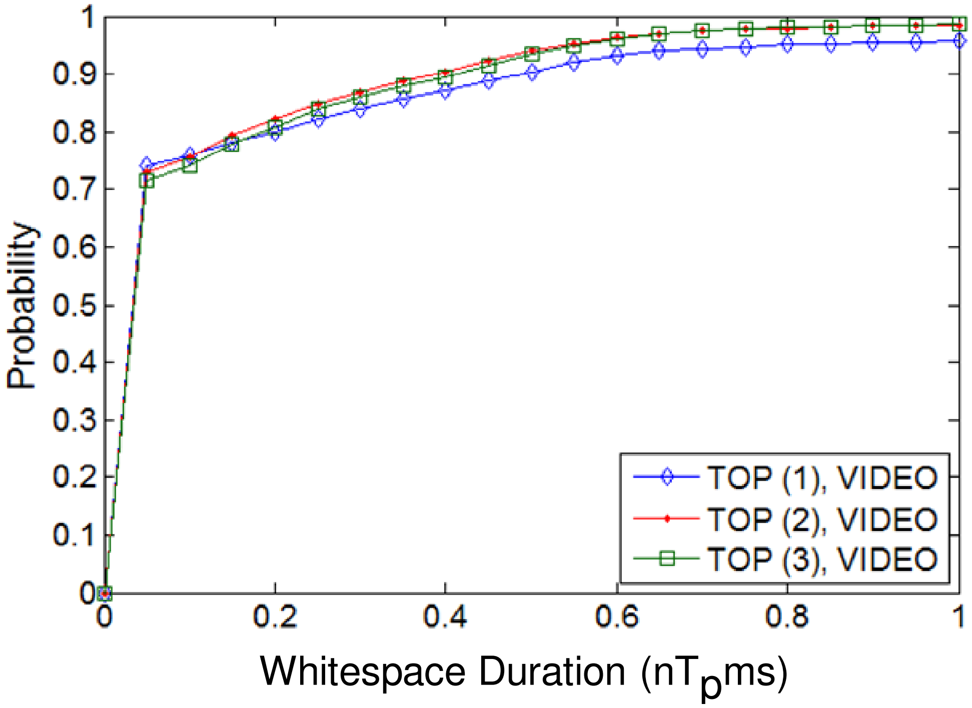


Figure 3-29: Video Stream whitespace cdf within the interval 0 ms to 1 ms

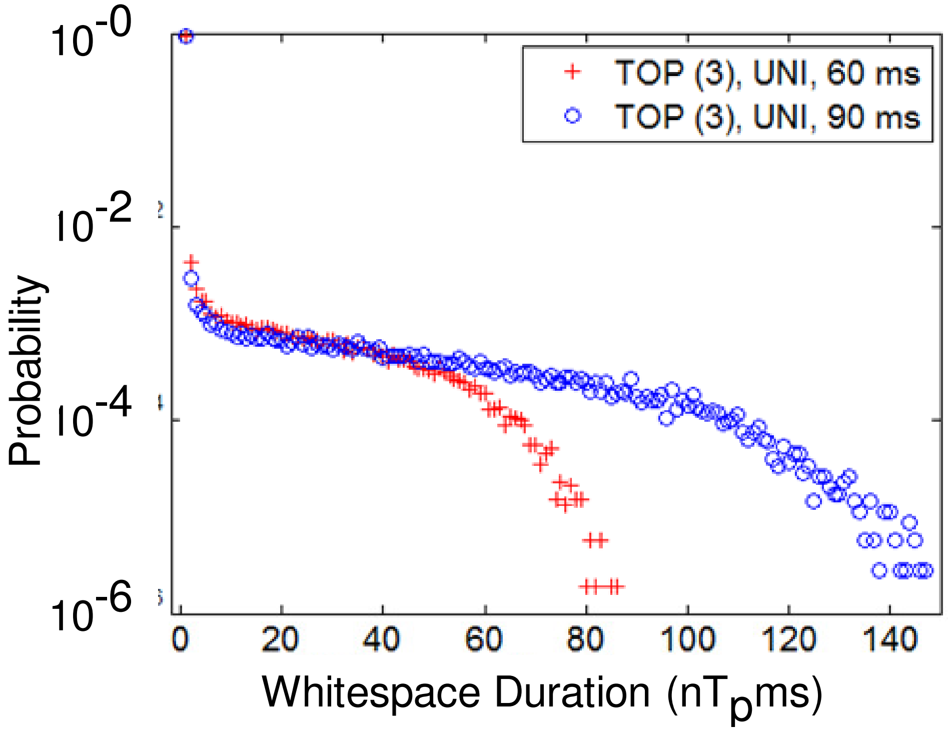


Figure 3-30: Impacts of data rate for Uniform pdf distribution

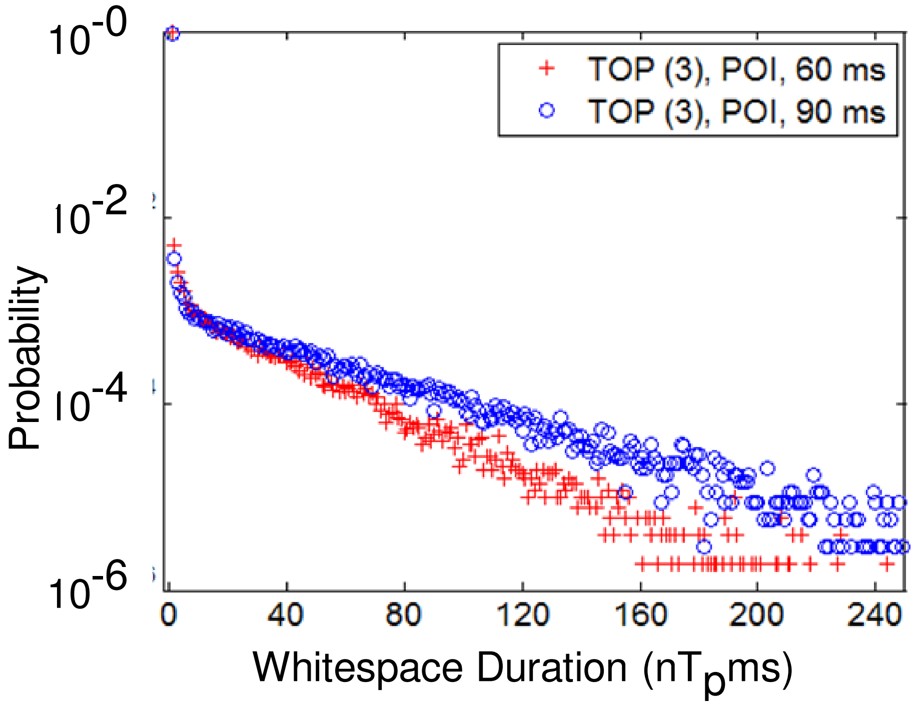


Figure 3-31: Impacts of data rate for Poisson pdf distribution

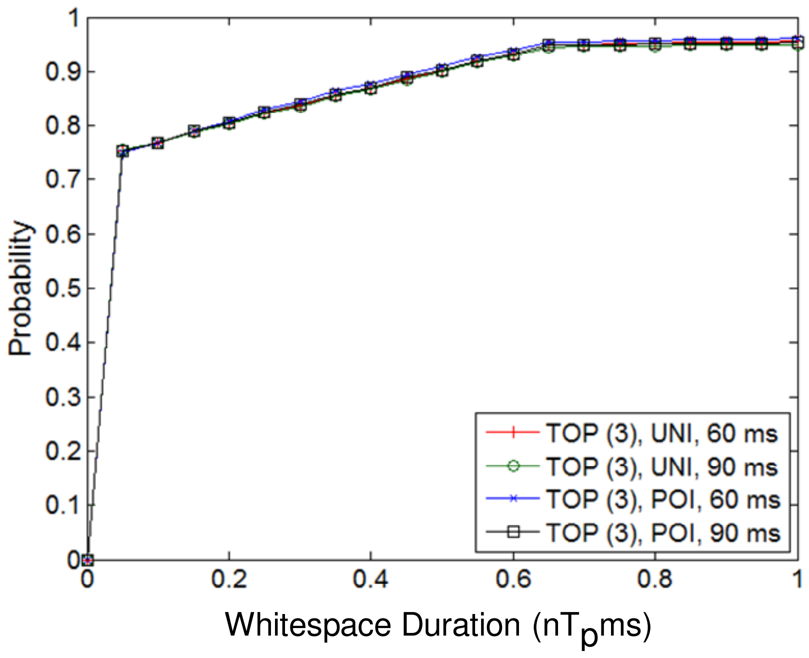


Figure 3-32: Impacts of data rate for Uniform and Poisson cdfs within the interval 0 ms to 1 ms

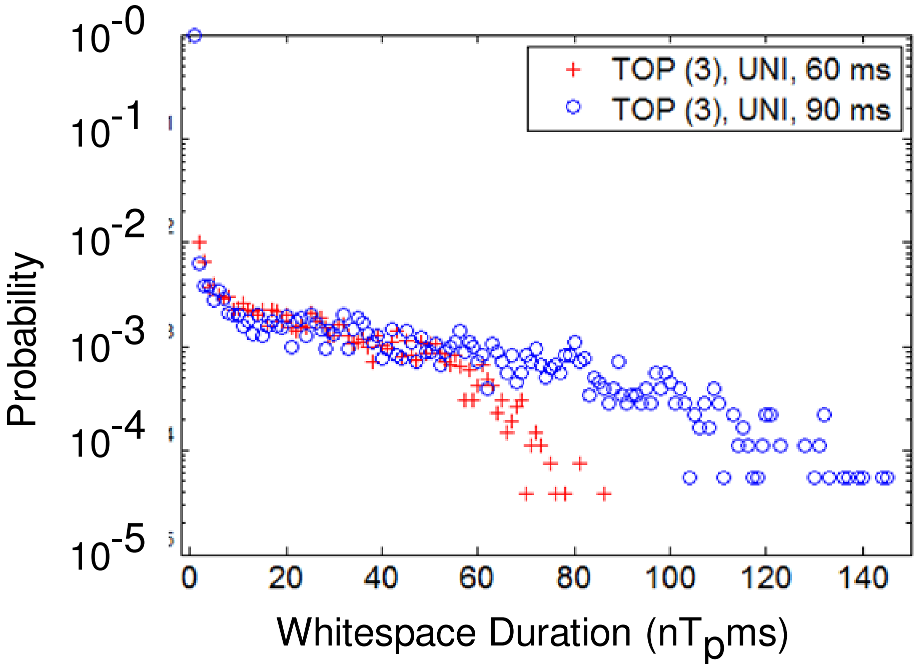


Figure 3-33: Impacts of sensing period on Uniform whitespace pdf distribution

3.4.2.3 Impacts of Finite Sensing Rate

In Figure 3-33 and Figure 3-34, the whitespace pdf *w(n)* is shown with the value of sensing period T_p increased to 50 μs, which is 5 times the duration of the smallest possible whitespace duration for 802.11 primary traffic. This larger sensing period causes some change to the pdf of the whitespace durations. The reason for this change in probability is that the whitespaces with smaller duration than the value of T_p may not be detected. This causes a redistribution (increasing) of the probabilities to whitespace durations greater than T_p. However, as shown in Figure 3-35, although there is a redistribution of whitespace probabilities, the first millisecond still shows that a large portion of the whitespaces are still shorter than 1 ms, which was observed for the faster sensing cases (i.e. T_p set to 5 μs in Figure 3-32).

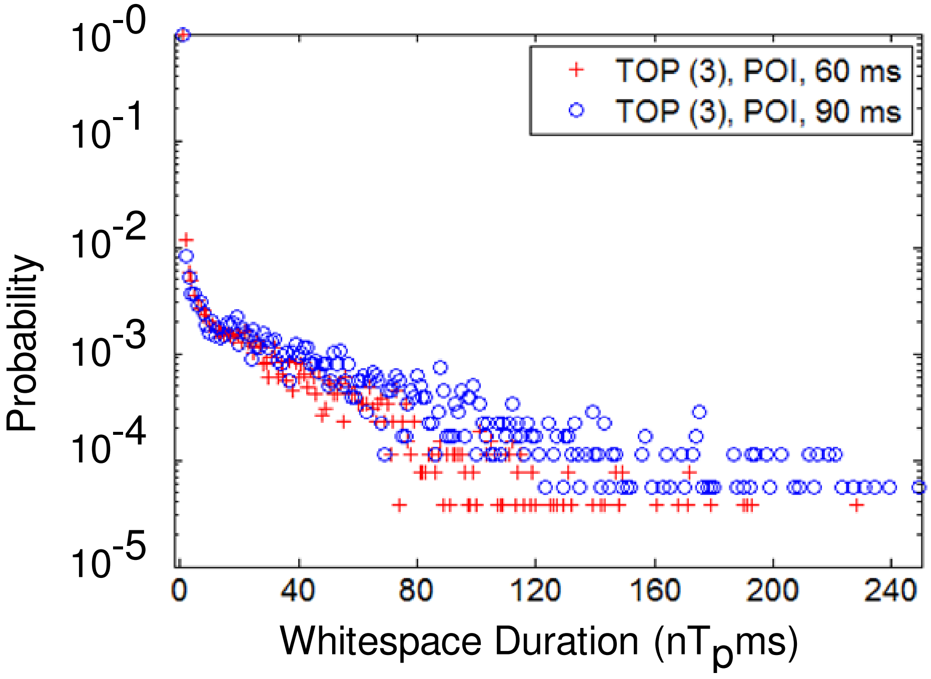


Figure 3-34: Impacts of sensing period on Poisson whitespace pdf distribution

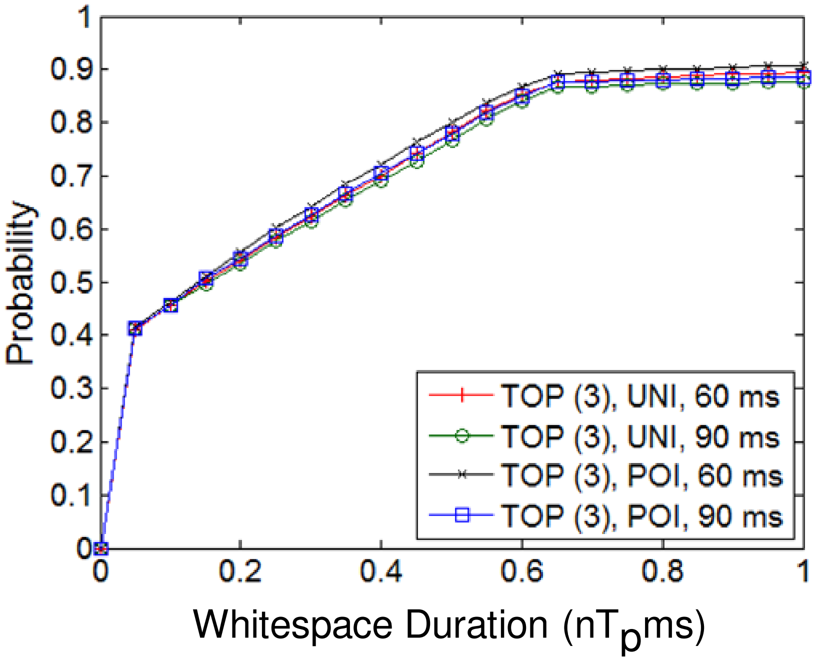


Figure 3-35: Impacts of sensing rate for Uniform and Poisson cdfs within the interval 0 ms to 1 ms

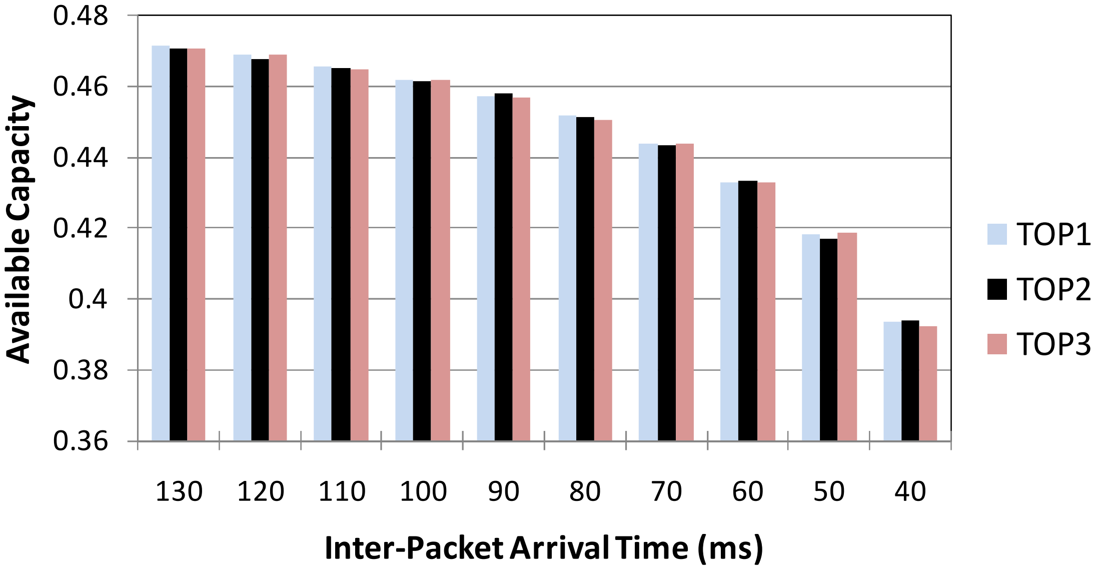


Figure 3-36: Available channel capacity in terms of fraction of whitespace for the Uniform traffic profile with various inter-packet arrival times and topologiesAvailable Whitespace Capacity

The whitespace distributions shown in Section 3.4.2 exhibits very dynamic characteristics depending on the traffic profile and data rate. A question that may be raised is: Is there any capacity available to the SUs to exploit in this type of whitespace? In this section, the available whitespace capacity is investigated. Figure 3-36 shows the available channel capacity in terms of fraction of whitespace for the Uniform traffic profile with various inter-packet arrival times (IPT) and topologies. For the highest IPT case (slowest sending rate), the capacity of the channel is approximately 0.47. This means that the channel is not being utilized nearly half the time. For the lower IPT the capacity decreases to 0.39. It follows that a faster PU sending rate will decrease the available channel capacity for the SUs. Observe that there is little difference between the different topologies. This result occurs because the offered load is similar among all of the topologies.

Figure 3-37 shows the available channel capacity in terms of fraction of whitespace for the Poisson traffic profile. The results for this case are similar to the Uniform experiments. Figure 3-38 shows the available channel capacity for the TCP and Video Stream traffic profiles. The TCP profile has very little available whitespace at approximately 10% for the different topologies. This was also shown in the pdf of the TCP traffic shown in Figure 3-26. For the Video Stream profile, there is a larger capacity for the TOP1 than the other two topologies.

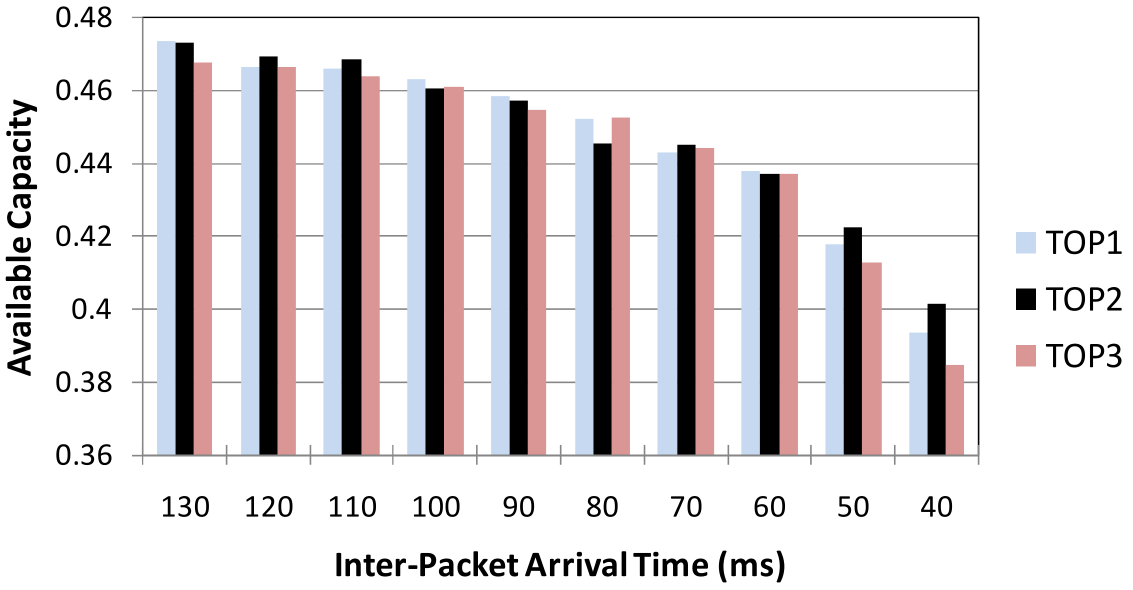


Figure 3-37: Available channel capacity in terms of fraction of whitespace for the Poisson traffic profile with various inter-packet arrival times and topologies

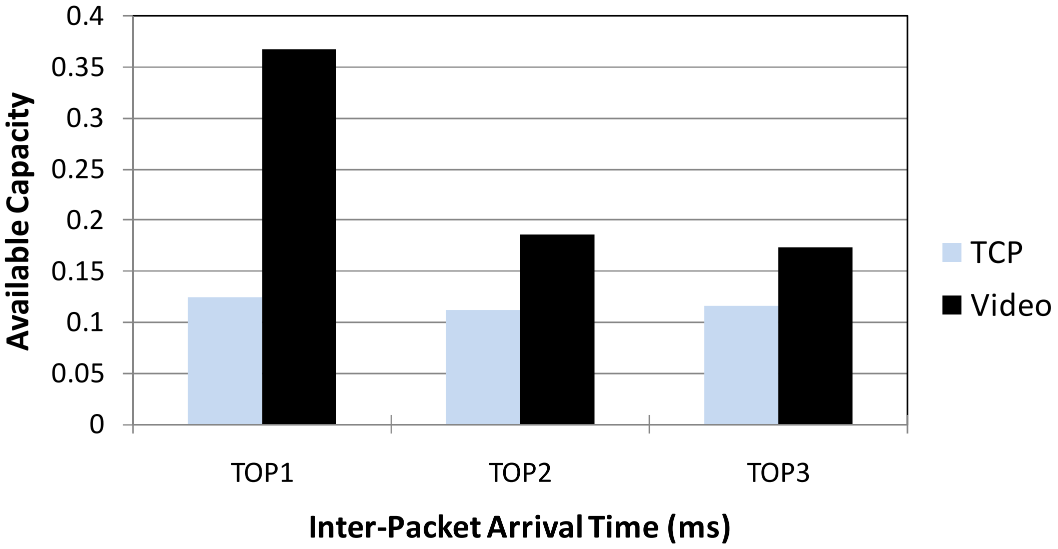


Figure 3-38: Available channel capacity in terms of fraction of whitespace for the TCP and Video Stream traffic profile

3.5 Summary and Conclusions

The distributions presented in Figure 3-22 through Figure 3-38 represent the general whitespace properties for a wide range of topology and traffic variations that were experimented using our ns-2 based simulator. A number of key observations can be made from the whitespace statistics. First, the detected whitespace durations from the measured RSSI trace vary a great deal across different topologies and traffic patterns. This indicates that a secondary user may experience vastly different whitespace environments when attempting to access primary networks. Second, as shown in Figure 3-36 and Figure 3-37, there is available capacity for SUs to scavenge depending on the traffic profile of the PU.

Finally, as shown in Figure 3-24 and Figure 3-25, regardless of the traffic profile or topology, the vast majority of the whitespaces (above 90%) last for less than 1 ms. This results from the large number of *small* whitespaces created during the RTS-CTS-DATA-ACK cycle for each packet transmission. The small whitespaces creating the initial peak in the pdfs are the *control* and the *forwarding* whitespaces as defined in Section 3.3.3. For example, in TOP (3) there are 4 hops for each flow, therefore there will be generally $3 \cdot 4 + 3 = 15$ small whitespaces per transmission representing 12 *control* and 3 *forwarding* whitespaces. While there will be only one inter-packet whitespace per transmission, resulting in a ratio of 15:1 control and forwarding whitespace to inter-packet whitespace. Note that this whitespace property is a feature of 802.11 networks, and can be heavily exploited by SUs while accessing this type of networks. The next chapter presents a secondary user access strategy that exploits this property to maximize SU throughput and minimize interference to the PUs.

Chapter 4: Contiguous SU Transmission Strategy (CSTS)

4.1 Introduction

In this chapter, a secondary user access mechanism, which is termed as Contiguous SU Transmission Strategy (CSTS) is introduced. This strategy is targeted mainly for whitespace environments created by Ad-hoc mode 802.11 PU traffic as discussed in Section 3.4.

4.1.1 Motivation

The key challenge in accessing primary networks is that without any prior knowledge of the starting times and durations of whitespaces, the SUs must access the channel such that the disruption to the PUs is minimized. Additionally, as a conflicting objective, the throughput of the SU should be maximized for a given whitespace profile. These objectives should be accomplished even in the presence of different primary user traffic characteristics.

4.1.2 Problem Definition

From Section 3.4, it is clear that for most 802.11 style primary traffic, due to the inter-frame spacing at the MAC layer, over 90% of the detected whitespaces are too small to accommodate the secondary packets. Additionally, as shown in Section 0, there is capacity available to the SUs in some types of PU profiles. Therefore, in order to facilitate efficient scavenging of the channel capacity left over by the PUs, a secondary user access strategy that can exploit these properties to maximize SU throughput and minimize interference to the PUs is needed.

4.1.3 Assumptions

The key assumptions are as follows. First, no cooperation is assumed between the PUs and SUs. Second, the SUs are assumed to have the ability to differentiate between the PUs' and the SUs' transmissions through physical layer waveform detection, which is feasible through cyclostationary feature detection [59] or waveform signature detection [60]. Third, an SU is able to detect collisions with PUs when an PU returns to the channel (i.e. begins transmission) during a whitespace. Although it is not the focus of this thesis, this capability can be realized by channel sensing after each packet transmission is completed by an SU. If the SUs' packet size is smaller than that of the PUs', a collision can be inferred by an SU by detecting an PU's signal after the SU's transmission is completed [14]. Finally, for the purpose of protocol analysis, it is assumed that the primary user traffic pattern does not change appreciably as a result of secondary transmissions [12, 14, 15, 56].

4.1.4 Related Work

Prioritized device coexistence within CSMA based WLANs can be achieved [18, 19] by using different *inter frame spacing* (IFS) periods for different user and/or traffic classes. MAC protocol 802.11e [18], for instance, uses different Arbitration IFS (AIFS) periods to provide CSMA based prioritized access among different device and/or traffic classes. When a channel is found free, a node waits for a specific AIFS periods depending on the device or traffic class, before it attempts to send a packet. For higher priority primary traffic, a node waits for a smaller AIFS period. This ensures when multiple nodes contends for the channel, the primary users' nodes (with smallest AIFS) wins.

While providing reasonable access differentiation, these approaches rely only on the instantaneous channel status (i.e. free or busy) for granting access. This leads to undesirable disruptions to the PU traffic as follows. Consider a situation in which an SU intends to transmit a packet and it does so after finding the channel free (i.e. a whitespace) for the AIFS specified for the SUs. Now, in the middle of this SU's packet transmission if an PU in the vicinity intends to send a packet, it needs to wait until the current SU transmission is over. This causes an undesirable delay for the PU traffic, which in turn will affect the PUs' application performance. Since this is mainly a result of the SUs' reliance only on the instantaneous channel state, a more robust approach for the SUs would be to also consider the long term whitespace model. The following DSA approaches attempt to accomplish that.

The authors in [6] develop an WLAN access strategy for secondary users in which the SUs utilize packet size slots for the channel access. At the beginning of each slot, an SU senses the channel and if the channel is free then it transmits with a specified probability that is calculated from previous measurement. The objective is to minimize PU disruption and maximize SU throughput. The use of time-slotting in their approach requires time synchronization across the secondary users. In the proposed access mechanism in this thesis, the need for such inter-SU time synchronization is avoided via asynchronous whitespace access based on a stochastic whitespace modeling approach.

In [12, 23], the proposed access strategies rely on specific types of PU networks. Protocols are developed in [12] for a slotted PU network where the SUs are synchronized with the PU's slots. In this case, if an PU does not use a slot then the SU can transmit after a small duration of time in the same slot. The PUs in [23] are assumed to use a specific combination of time and frequency multiplexing to prevent inter-PU access collisions. In this thesis, we formulate the PU network more generally by not assuming any specific PU MAC layer such as the slotted ones in [12]. The target here is to develop an SU access mechanism, which can deal with unslotted as well as stochastic PU MAC layers such as 802.11, CSMA and ALOHA protocols. This assumption of a general PU access protocol broadens the applicability of the SU access approach proposed in this thesis.

The authors in [61, 62] do not restrict PUs to a slotted MAC protocol although their access strategy requires a slotted SU network. During the beginning of each slot the SU senses the channel and if the channel is free then the SU transmits. A greedy version also uses probabilistic analysis to decide whether or not to transmit in a given slot. This framework requires time synchronization across the SUs. In our approach, SUs use a stochastic mechanism and are not required to maintain time synchronization.

Game theoretic mechanisms have been introduced in [24, 25] for spectrum access for the SUs. The PUs in this framework cooperates with the SUs through pricing strategies and subscription fees to facilitate spectrum sharing. These approaches require the PUs to have knowledge of the SUs presence. In contrast, the proposed mechanism in this thesis, no such cooperation is assumed, thus no changes in the PUs behavior is needed. The idea here is to develop an access mechanism for the SUs for scavenging the PUs' leftover bandwidth, without the PUs being aware of such scavenging. For example, data-enabled hand-held devices should be able to scavenge bandwidth in a WLAN without the primary VoIP handsets being aware of the scavenging process.

4.2 CSTS Access Algorithm Overview

From Section 3.4, it is clear that for most 802.11 style primary traffic, due to the inter-frame spacing at the MAC layer, over 90% of the detected whitespaces are too small to accommodate the secondary packets. Since the actual duration of any whitespace is not known a-priori to the SUs, if an SU transmits at the beginning of each whitespace then a vast majority of such transmissions will end up in primary traffic disruptions (PTDs). Primary Traffic Disruption (PTD) is defined as the probability that an SU will cause disruption to PUs as a result of transmitting at the end of a whitespace. If an SU defers transmission attempts, however for a suitably chosen wait-threshold duration μ, such PTDs can be potentially reduced without sacrificing the Effective Secondary User Throughput (EST). In line with this logic, a secondary user access strategy is developed that utilizes a minimum wait-threshold duration μ that is dimensioned based on the statistical properties of the Ad-hoc mode 802.11-based primary networks.

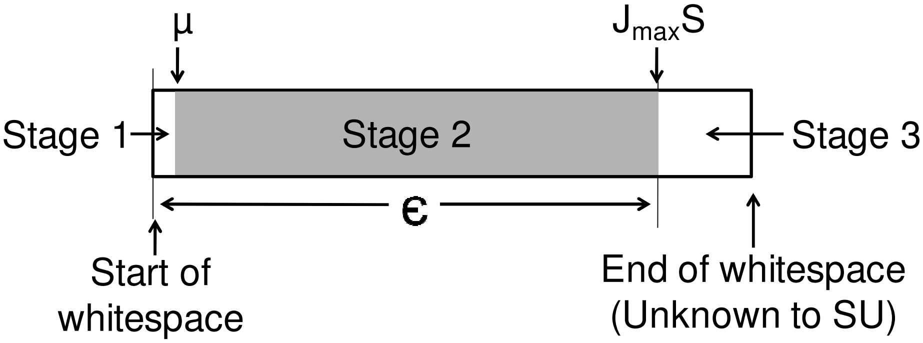
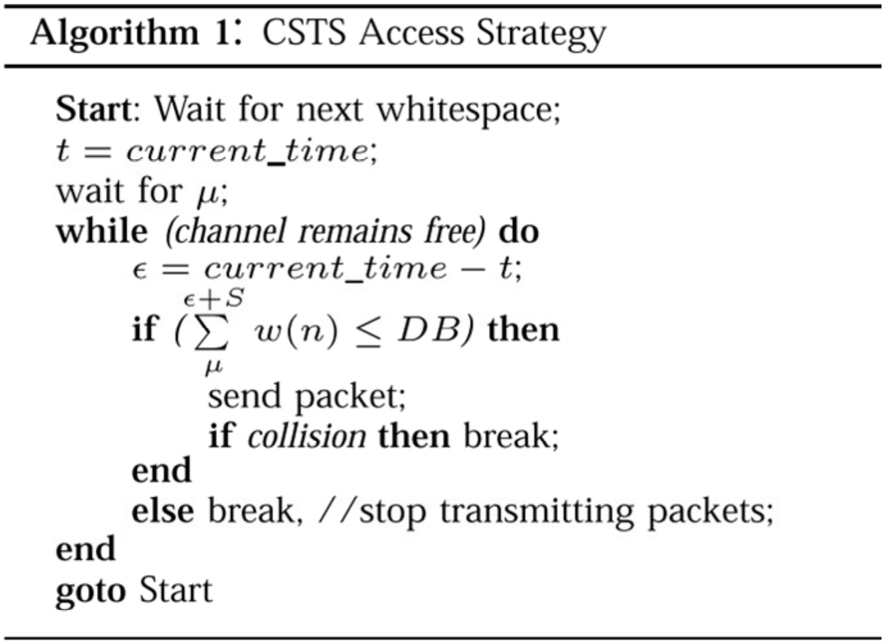


Figure 4-1: Different stages of CSTS access

As shown in Figure 4-1, in CSTS access there are three distinct stages. In the first stage, an SU initially waits for the channel to become idle. Once the channel becomes idle, the SU waits for an additional wait threshold duration μ before it attempts to transmit its first packet of duration S. At this point, the access protocol enters into stage-2. At this stage, if the SU has more packets to send, it continues sending until the measured value of ϵ (time elapsed since the beginning of the current whitespace) becomes so large that the expected PTD for new transmissions become equal to or larger than a pre-determined Disruption Bound (DB). Let us define a term J_{max}, which denotes the maximum number of packets that the secondary user can send before the expected disruption reaches the pre-determined DB. The quantity J_{max} is computed by the SU based on the observed whitespace statistics. After J_{max} packets are sent, the secondary user abandons transmission attempts and starts channel sensing for detecting the next whitespace.

If, however, a whitespace ends before all J_{max} packets are sent by the SU, it vacates or exits the channel after detecting that a primary user has started transmitting. Note that in spite of the SU vacating the channel, disruption to the PUs' traffic can still occur since the PU must defer its transmission until the channel is vacated by the SU. This second stage defines the contiguous block of time that an SU transmits in any whitespace.

Finally, the third stage of CSTS corresponds to the situation in which a whitespace duration is greater than the end time of the ${J_{max}}^{\text{th}}$ packet from the SU. No PTD will be caused in this stage because the SU will exit the whitespace after J_{max} transmissions. The channel access strategy is summarized in the form of the pseudo code in Algorithm 1 in Figure 4-2.



```
Algorithm 1: CSTS Access Strategy

Start: Wait for next whitespace;
$t = current\_time$;
wait for $\mu$;
while (channel remains free) do
    $\epsilon = current\_time - t$;
    if ($\sum_{\mu}^{\epsilon+S} w(n) \leq DB$) then
        send packet;
        if collision then break;
    end
    else break, //stop transmitting packets;
end
goto Start
```

Figure 4-2: CSTS access algorithm

4.3 Functional Overview

Figure 4-3 shows a functional diagram of the architectural components in CSTS. The user tunable control inputs into the system are secondary user packet size (*S*), primary disruption bound (*DB*), and the sensing interval (T_p), which is a property of the secondary user hardware. The SU continuously computes the whitespace pdf *w(n)* with a channel-sensing interval of T_p. The access parameters are then fed into the access module, which, in addition to channel sensing, transmits SU user packets into the channel using the process described in Section *V.B*. The objective is to satisfy a user specified primary disruption bound *DB*.

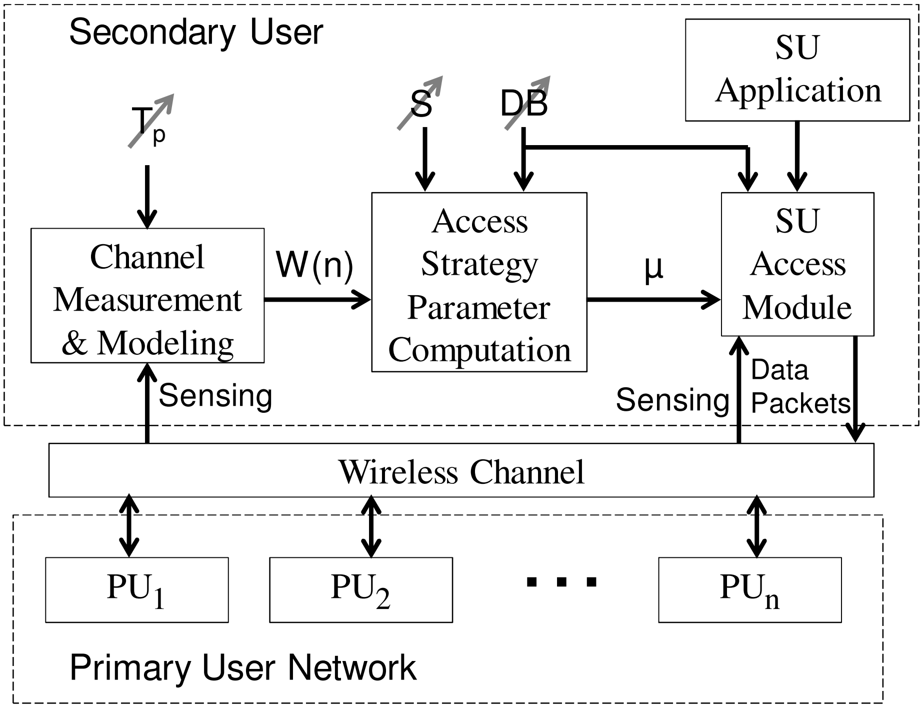


Figure 4-3: Functional diagram of the SU access strategy

4.4 Algorithm Analysis

4.4.1 Primary Traffic Disruption

For a given whitespace profile *w(n)*, the PTD is defined by the probability that the whitespace duration is in the range between μ and $\mu + J_{\max} \cdot S + (J_{\max} - 1) \cdot \gamma$ where *S* is packet size and γ is the inter-packet spacing for the SUs. Therefore, an SU accessing a whitespace characterized by pdf *w(n)* will create primary user disruption (PTD):

$$PTD = \sum_{\mu}^{\mu + J_{\max} \cdot S + (J_{\max} - 1) \cdot \gamma} w(n) \tag{4-1}$$

4.4.2 Effective Secondary Throughput

EST is defined as the average number of SU packets successfully transmitted per whitespace, normalized by the number of packets that could have been transmitted per whitespace for a given *w(n)*. EST represents the efficiency of channel usage by the SU. To model EST, we first define S_j, the probability of sending exactly *j* packets in a whitespace. This corresponds to the event in which the j^{th} packet from an SU in a whitespace has disrupted the PUs, i.e. the whitespace had ended during the transmission of the j^{th} packet. The quantity S_j can be written as:

$$S_j = \sum_{\mu+(j-1)\cdot(S+\gamma)}^{\mu+j\cdot S+(j-1)\cdot\gamma} w(n) \qquad (4\text{-}2)$$

For a whitespace that lasts between μ and $\mu+(J_{\max}-1)\cdot(S+\gamma)$, the number of packets sent by an SU is *j*. For whitespace duration between $\mu + J_{\max}\cdot S + (J_{\max}-1)\cdot\gamma$ and ∞, the number of packets sent by an SU is J_{max}, since the secondary user vacates the whitespace after sending J_{max} packets. Therefore, throughput, the expected number of secondary packets sent for a given whitespace, can be expressed as:

$$Throughput = \sum_{j=1}^{J_{\max}-1} j\cdot S_j + \sum_{j=J_{\max}}^{\infty} J_{\max}\cdot S_j \qquad (4\text{-}3)$$

The throughput equation is valid only when the secondary user captures all available whitespace using perfect sensing (i.e. T_p set to zero). However, when T_p is non-zero, there is a probability that a whitespace smaller than T_p will not be detected and thus will not be accessed by the SU. In that case, the probability of missing a whitespace of length *w* is equal to (T_p -w)/ T_p. We account for this loss of throughput using this equation:

$$Loss = \sum_{j=1}^{\left\lceil T_p/S \right\rceil} \left(j \cdot S_j \frac{\left\lceil T_p/S \right\rceil - j}{\left\lceil T_p/S \right\rceil} \right) \tag{4-4}$$

Although non-zero T_p values can affect the whitespace access, the impacts are alleviated by requiring an SU to wait at least T_p duration before accessing any whitespace.

The effective capacity available to the SU can be expressed as the average number of packets that is sent per whitespace. This quantity can be written as (with μ and γ equal to zero):

$$Capacity = \sum_{j=1}^{\infty} j \cdot S_j \tag{4-5}$$

Therefore, the Effective Secondary user Throughput (EST) can be expressed as:

$$EST = \frac{Throughput - Loss}{Capacity} \tag{4-6}$$

which is computed from Eqns. 3, 4, and 5.

4.4.3 Dimensioning Wait-threshold μ

The selection of wait-threshold μ is critical since it affects both PTD and EST as shown in Equations 4-1 and 4-6. A large μ can reduce primary traffic disruption by preventing the SUs from transmitting during very short whitespaces. A large μ, on the other hand, can bring down the EST, since some portions of large whitespaces will be lost due to this conservatively chosen large wait period. The goal is to pre-dimension the parameter μ, based on the measured pdf function *w(n)*, in order to strike a desirable balance between the PTD and EST. Note that μ is parameterized by T_p and the lowest value of μ is a least one T_p period, since at least one sensing interval is needed to determine if the channel is free. In CSTS, the wait-threshold period determines the PTD caused by the first SU packet during a whitespace. For a wait-threshold t_x, such PTD can be expressed as:

$$f(t_x) = \sum_{t_x}^{t_x+S} w(n) \qquad (4\text{-}7)$$

Now, the minimum and maximum values of t_x, are zero and *2S*, where *S* is the secondary packet duration. The maximum wait-threshold is *2S*, since waiting for a duration that is greater than or equal to *2S* would mean missing out sufficient whitespace that could have been used by the secondary user to send at least one packet. Therefore, for a given whitespace with pdf *w(n)*, the optimal wait-threshold μ is chosen as the smallest t_X, over the range *0* to *2.S*, such that the quantity $f(t_X)$ in Equation 4-7 is minimized.

4.4.4 J_{max} Computation

In CSTS, the number of packets the SU is allowed to transmit in a given whitespace is J_{max}. This quantity is determined by finding the maximum *j* that satisfies the inequality:

$$1-\sum_{\mu+(J_{\max}-1)(S+\gamma)+S}^{\infty} w(n) \leq DB \tag{4-8}$$

The left side of the inequality represents the PTD when the SU sends J_{max} packets in a given whitespace. Therefore, the inequality in Eqn. 8 gives the J_{max} that will reduce disruption to the pre-defined bound *DB*. By plugging in this value of J_{max} in Equation 4-1, we find the overall PTD caused by the CSTS access strategy.

4.4.5 Impacts of wait threshold μ

As discussed in the preceding sections, larger μ values can significantly reduce PTD by avoiding SUs' transmissions in short whitespaces. Figure 4-4 presents theoretical results for the CSTS mechanism when the wait-threshold μ is varied from 0 ms to 2 ms. The results correspond to topologies TOP1 and TOP2 in Fig. 1 with both uniform and Poisson distributed primary traffic. All traffic flows correspond to a mean of 80 ms inter-packet arrival time. The SU packet duration S was chosen to be 1.2 ms and the sensing interval T_p is set to 5 μs.

Observe in Figure 4-4 that as μ increases, the PTD decreases for all combinations of network topology, traffic profile, and data rate. For $\mu = 0$, representing the case when the SU transmits immediately at the start of a whitespace, the amount of PTD is very high for all cases. This is because as shown in Section 3.4, most of the whitespaces (above 90%) are of very short duration, consisting of control and forwarding whitespaces in 802.11 primary traffic. As μ increases, the PTD first decreases abruptly and then relatively moderately, but in a monotonic manner.

The figure also shows the effects of μ on EST. Recall that EST is the efficiency of whitespace usage by the SU. With μ set to zero, since the SU attempts packet transmission in all available whitespaces, the throughput is expected to be the maximum. As μ increases, the throughput decreases since the SU waits for a longer duration before transmitting in a given whitespace. The values of μ, larger than the threshold found in Equation 4-7, cause the EST to reduce drastically by making the access strategy overly conservative.

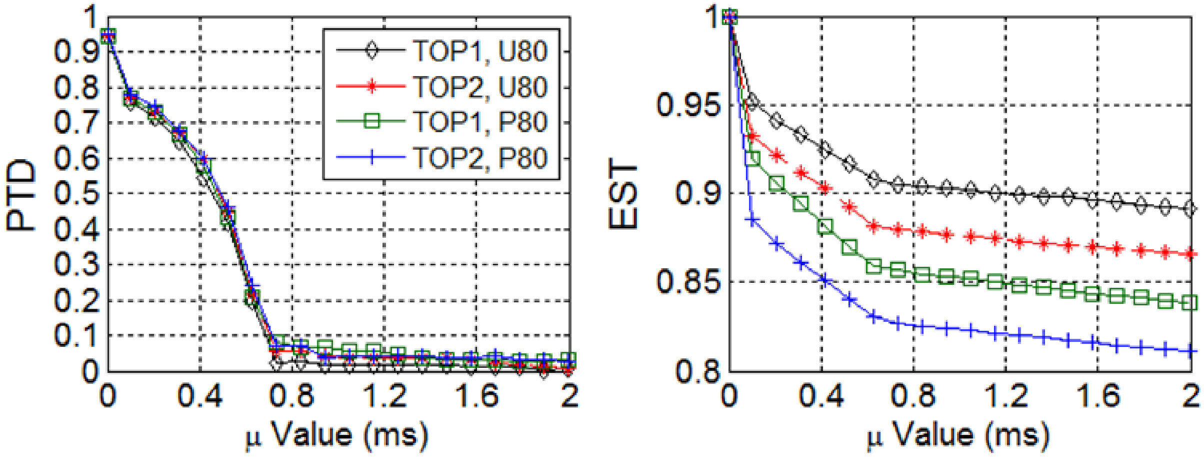


Figure 4-4: Evaluation of the impact of μ in CSTS

4.5 Summary and Conclusion

In this chapter, a secondary users access strategy is presented that access an Ad hoc 802.11 PU whitespace profile. The access strategy exploits the properties of the whitespace by avoiding transmissions during the first millisecond of a whitespace. Once a whitespace is accessed, the parameter J_{max} limits the amount of SU packets sent within that whitespace. Under this process, the secondary users maximize its own throughput while keeping the primary user disruptions to a predefined bound.

Chapter 5: Performance Evaluation for Coexistence in 802.11 Network

5.1 Introduction

In this chapter, the performance of the CSTS strategy is evaluated experimentally and theoretically in terms of the Primary Traffic Disruption (PTD) and the Effective Secondary Throughput (EST).

5.1.1 Comparison Protocols

We compare CSTS with a sense-and-transmit based protocol, called VX scheme [56], and a benchmark scheme (Benchmark).

5.1.1.1 VX Scheme

The VX (Virtual-transmit-if-Busy) Scheme has a similar strategy as CSMA where the SU will sense the channel and transmit when idle [56]. Initially the SU senses the channel, when the channel becomes idle a packet of length L_S is transmitted. Then, the SU starts a vacation from the channel of length V_S. After the vacation, if the channel is busy, the SU starts a virtual transmission stage and then enters into another vacation stage afterwards. Here, a virtual transmission means that the SU does not actually transmit the packet but waits for a time interval which is equal to the packet length L_S. After the vacation, the SU starts the channel sensing again. In contrast to the CSTS scheme, the VX scheme does not use statistical data but relies on short term sensing for its transmission decisions. Algorithmic analysis for this scheme has been presented in [56].

5.1.1.2 Benchmark

The Benchmark protocol assumes full knowledge of the starting and ending times of every whitespace. Therefore, it transmits SU packets in every available whitespace and stops when the next SU transmission will cause disruption to the PUs. The Benchmark protocol can be run only offline to find the best SU performance. This protocol essentially determines the highest throughput an SU can expect without causing any disruption to the PUs.

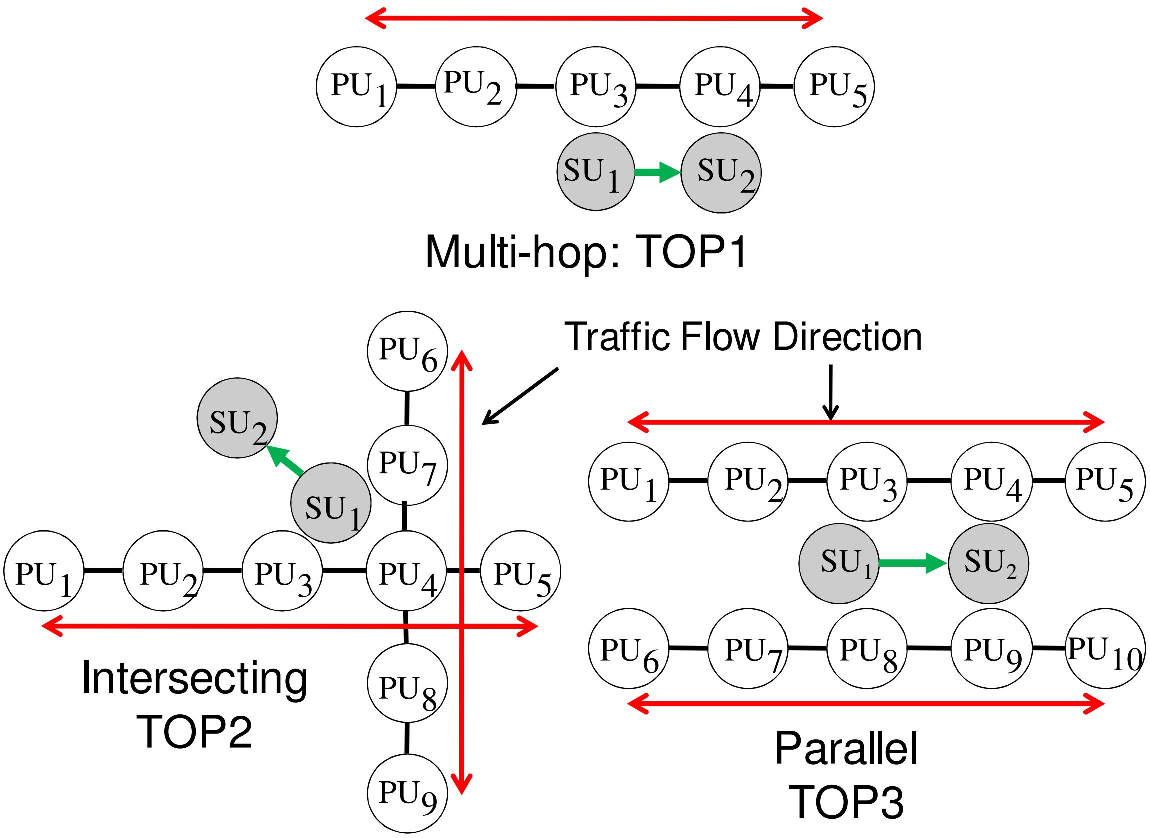


Figure 5-1: Primary user topologies and the traffic flows

5.1.2 Simulation Implementation

The access strategies are evaluated through simulation. There are many steps involved in the creation of the simulator to evaluate the protocols as shown in Table 5-1. First, the network simulator ns-2 was used to create the network topologies described in Section 3.3.1 and shown in Figure 5-1. The primary users use ad hoc mode 802.11 as the MAC layer, and the secondary users in Figure 5-1 observes whitespace profiles similar to those shown in Section 3.4.2. These whitespace profiles including uniform and Poisson distributions, a video stream, and TCP traffics are defined in the ns-2 TCL simulator file in Step 2. Then in Step 3, using AWK based scripts, simulations are created to run ns-2 for the various traffic profiles and topologies. With the combinations of the different topologies, traffic profiles, and data rates there are a large number of simulations that are needed to run.

Step #	Implementation Step
1	Creation of primary and secondary user grid topology
2	Setting of traffic profiles for secondary users
3	Use AWK scripts to create the required large number of simulations
4	Use the wireless-phy.cc file to capture RSSI measurements
5	Use the TCL scripts to parse RSSI outputs to create whitespace durations
6	Import whitespace durations into Matlab
7	Implement access strategies and various performance characteristics
8	Run experiments

Table 5-1: Simulator implementation steps

Next in Step 4, RSSI measurement traces are collected from the SUs for each traffic profile through the modification of the wireless-phy.cc file. The duration and starting times of every whitespace are created in Step 5 by parsing the output RSSI file from Step 4. With the whitespace durations files, the entire primary user transmission behavior is captured. Note that it is assumed that the primary user traffic profile does not change as a result of the activity of the secondary users [12, 14, 15, 56]. Afterwards in Step 6, the whitespace durations files are imported into Matlab. Within Matlab whitespace statistics are gathered such as the whitespace pdf *w(n)*. Lastly, the various access strategies are implemented and evaluated using Matlab and the extracted whitespace durations from the ns-2 simulations.

5.2 CSTS Evaluation

In this section, protocol performance is evaluated using multiple variables including PU loads, SU packet size, and disruption bound. For the presented results, the SU packets are generated back-to-back, meaning packets are always available to send within an SU's buffer. The user-defined disruption bound (*DB*) is set to 0.05 [6], which indicates that only 5% of the PUs' traffic is allowed to experience disruptions from the SUs' transmissions. The sensing interval T_p is set to 5 μs. For the VX scheme, V_s, the channel wait time, is set to 0.5 ms and the *Virtual Xmit* duration is set to 16 ms, which was experimentally shown to deliver the best performance for that access protocol.

5.2.1 Primary User Load

Figure 5-2 demonstrates the impacts of varying PU load on the SU throughput and PU disruption for TOP2 in Figure 5-1. The PU inter-packet arrival duration is varied from 50 ms to 120 ms for simulating different loading conditions. This is represented in the figure as 8 to 20 packets per second (pps). For all the primary packet rates, a ±20% uniformly distributed inter-packet arrival variation was introduced. The SU packet duration was set to 1.2 ms, which is half the PU packet size.

From Figure 5-2, observe that the CSTS access protocol is able to maintain the required primary disruptions within the specified Disruption Bound (*DB*) across a wide range of PU data rates and topologies. This is because the μ and J_{max} values are set according to the observed whitespace statistics. As the PU data rate increases, the duration of usable whitespace in the channel reduces. This occurs because the primary user traffic is utilizing a larger portion of the channel for its transmissions. With increasing PU rate, the number of distinct whitespaces also increases, since the whitespaces become more fragmented. Fragmentation of whitespaces happens because the increase in bidirectional traffic prevents the creation of longer duration whitespaces. In addition, a larger amount of small control and forwarding whitespaces are created in every multi-hop transmission. Since the SU access strategy is executed on a per-whitespace basis, with the increased number of distinct whitespaces, the SU attempts to execute its PU protection mechanisms more often. This causes the SUs' channel access efficiency or EST to decrease due to increased PU load.

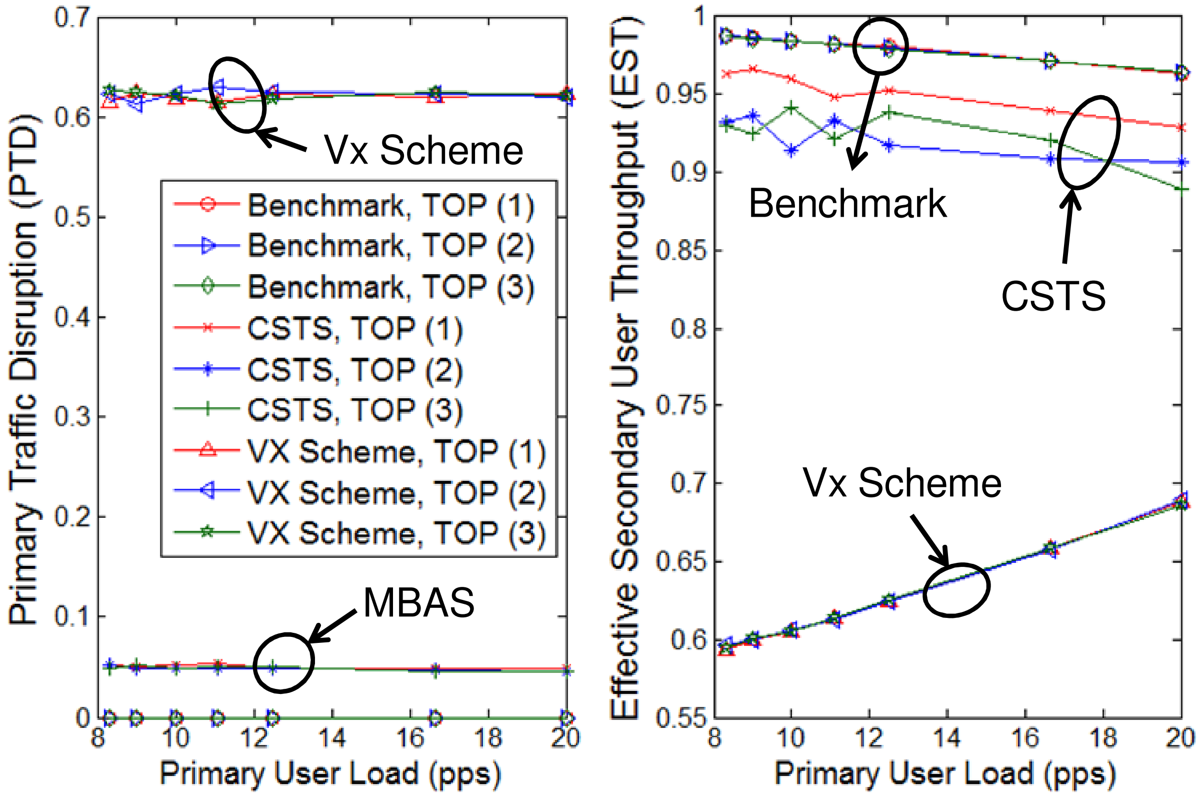


Figure 5-2: Performance for uniformly distributed PU traffic

For all the experimented topologies, the throughput for MBAS remains close to that of the Benchmark protocol. Using the full prior knowledge of the whitespace, the Benchmark protocol is able to evaluate the maximum possible EST with zero PTD. This shows that CSTS is able to maximize throughput while maintain PTD bounds by decreasing the J_{max} as necessary due to the increasing PU traffic load.

For the VX scheme, disruption to the PU traffic is observed to be higher. Utilizing the *Virtual Xmit* wait times, the SU avoids sending during busy times in a channel. However, once the SU starts accessing a whitespace, it stops sending only if an PU returns during a V_s wait time. Thus, there remains a high probability of causing disruption during the SU transmissions. The VX scheme has a relatively lower throughput since it waits for a V_s time after every transmission, which is not the case of CSTS mechanisms. These results suggest that by utilizing the history of the channel, it is indeed possible to improve the SU throughput while maintaining a bound of disruption on the primary networks.

Figure 5-3 shows results for the same setup, but with Poisson distributed PU traffic. Note that the performance trends for this scenario are very similar to those for the Uniform case as shown in Figure 5-2. The throughput is slightly decreased because the Poisson traffic creates a larger amount of fragmented whitespaces, which makes CSTS reduce its transmissions to maintain the required *DB*. Similar results were also observed for video stream PU traffic for which the pdf was shown in Figure 3-27.

We have also experimented with the TCP primary traffic. Since TCP opportunistically utilizes available bandwidth in the channel, the amount of resulting whitespace for TCP traffic is small. As a result, the achievable SU throughput remains severely restricted when the primary disruption bounds are set at low values [17].

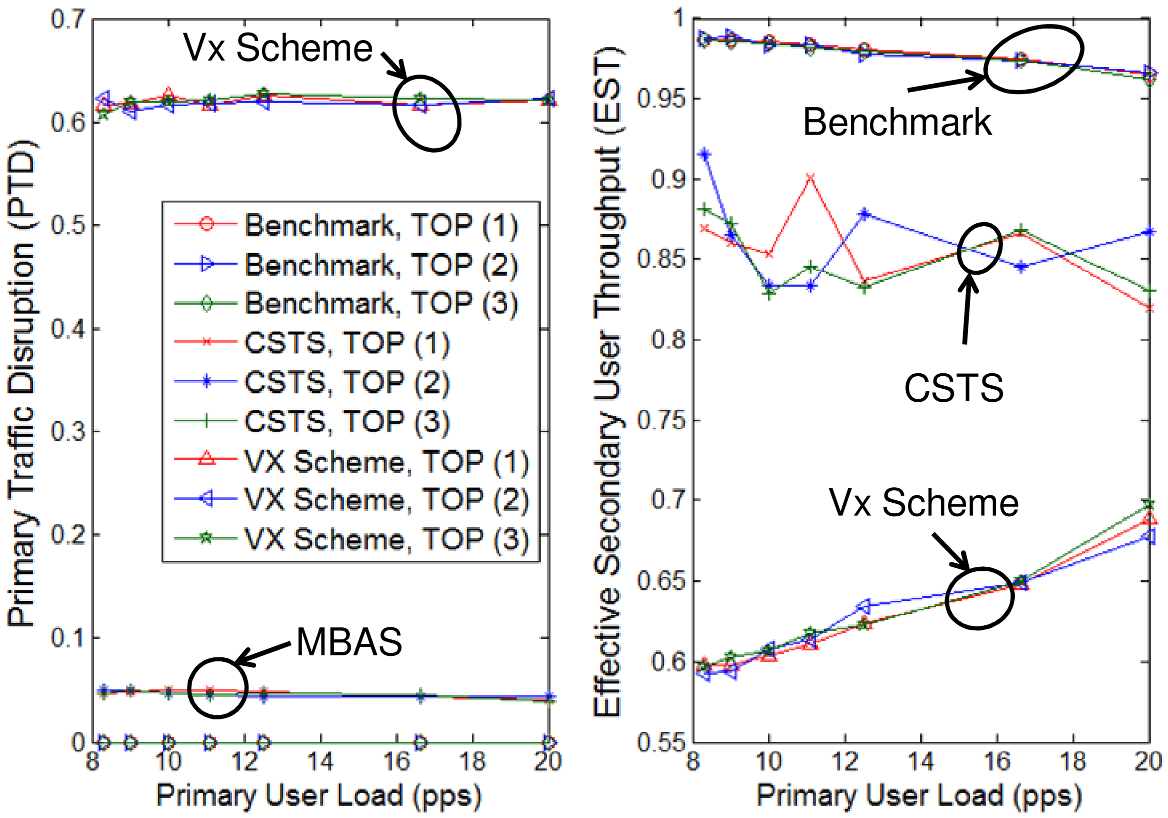


Figure 5-3: Performance for Poisson distributed PU traffic

5.2.2 Secondary User Packet Size

Figure 5-4 shows the performance of CSTS under a varying SU packet size from 625 µs to 2 ms, as done in [14]. The PU rate is set to 10 pps with uniformly distributed traffic on topology TOP1. Observe that the throughput decreases as the SU packet size increases. In order to maintain the *DB* with larger packet sizes, the access strategy must reduce the transmission probabilities since a larger secondary packet has a higher probability of causing disruption to the primary at the end of a whitespace. Similar results are shown in Figure 5-5 and Figure 5-6 for topologies TOP2 and TOP3 respectively. Figure 5-7 shows the performance with the video stream traffic profile. Observe that the EST is lower in this case since the duration of whitespace is generally smaller with the video stream profile as shown in Figure 3-27.

Additionally the whitespace is fragmented more with the larger amount of traffic.

Figure 5-4 through Figure 5-7 presents both experimentally measured (via ns2 simulation) and theoretical values (using Equations 4-1 through 4-8), which are marked as *EXP* and *THY* respectively. It is evident that the experimental results match very well with the theoretical results.

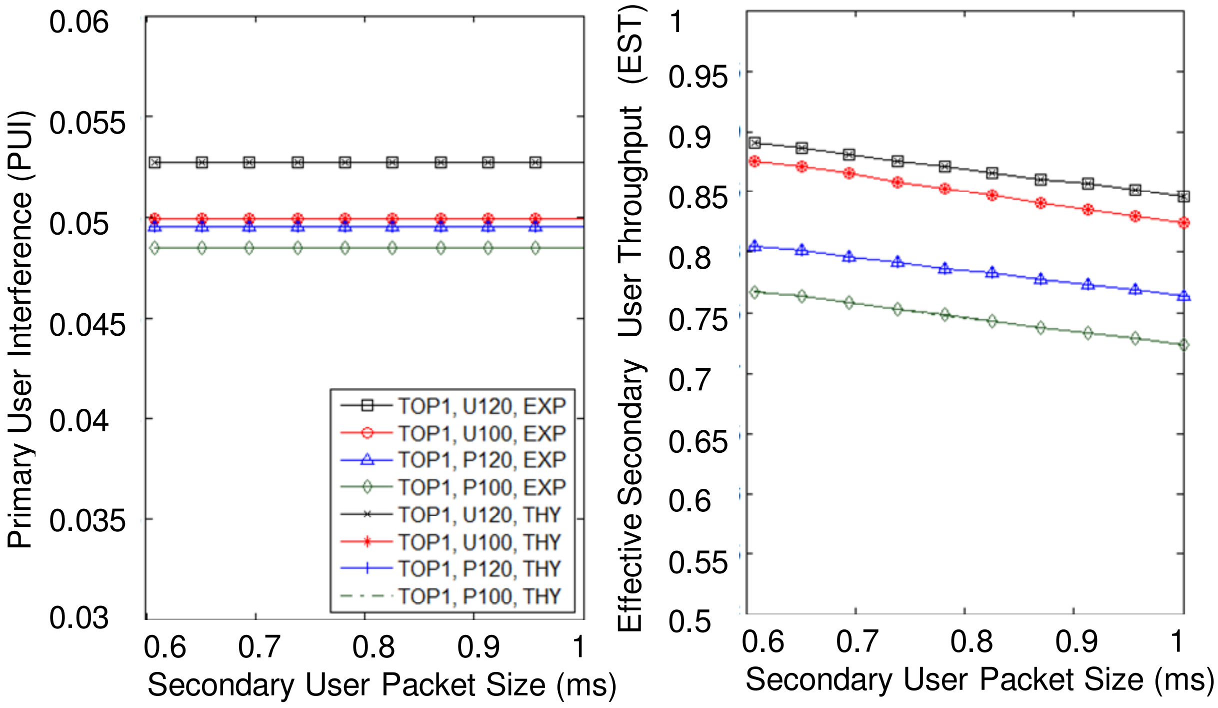


Figure 5-4: Impacts of the SU packet size on EST and PTD on TOP1

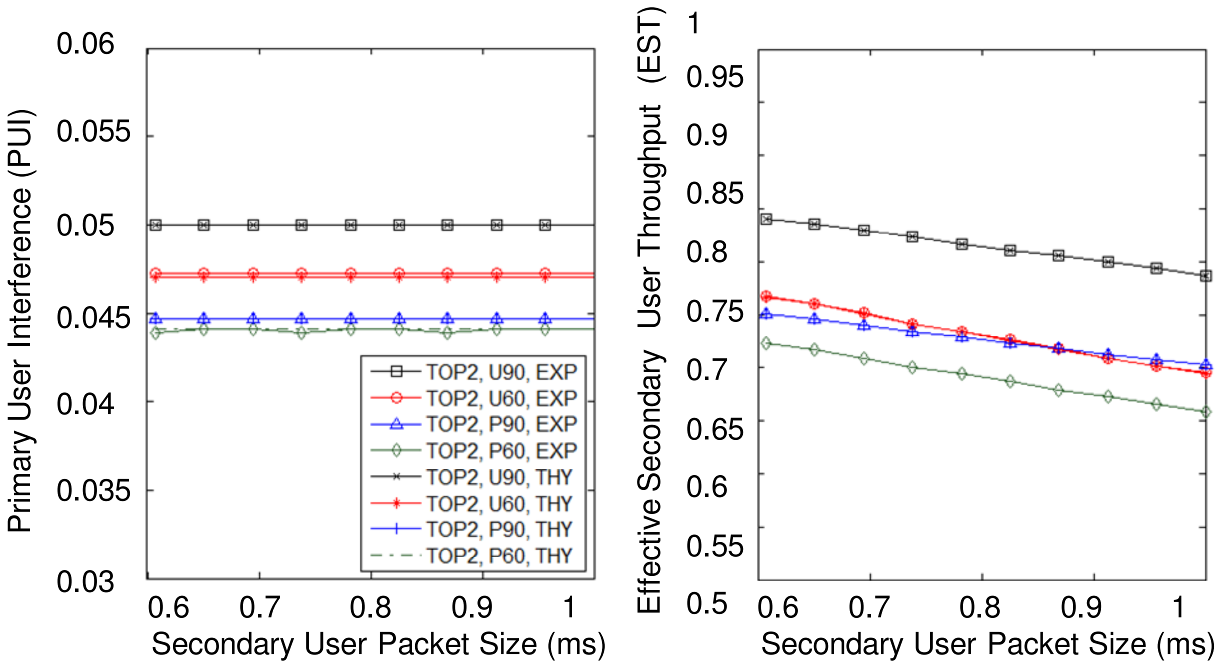


Figure 5-5: Impacts of the SU packet size on EST and PTD on TOP2

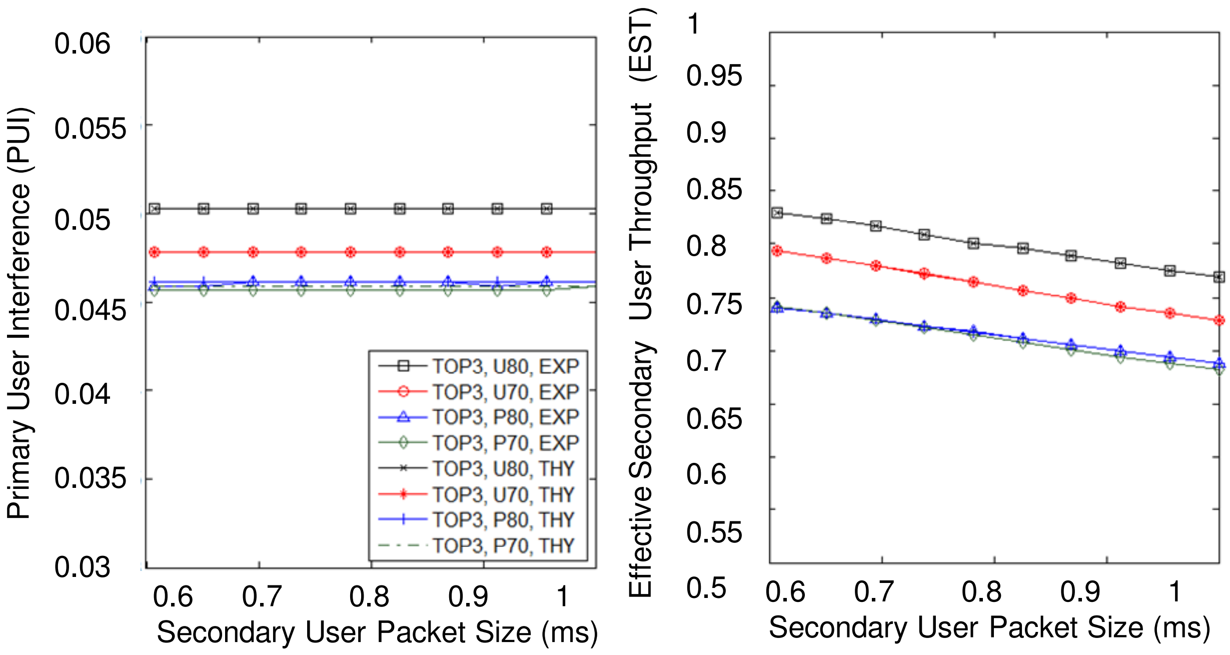


Figure 5-6: Impacts of the SU packet size on EST and PTD on TOP3

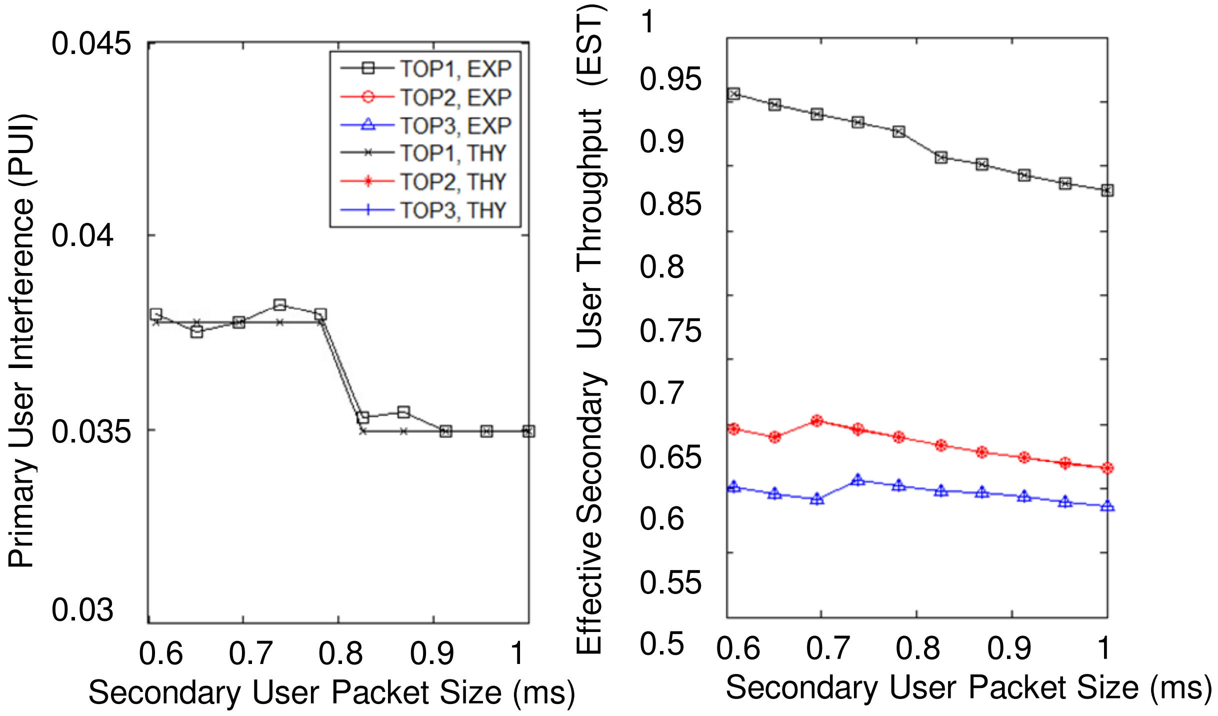


Figure 5-7: Impacts of the SU packet size on EST and PTD for Video Stream

5.2.3 Dynamic PU Traffic Patterns

Figure 5-8 reports the protocol performance from an experiment in which the PU traffic pattern was dynamically changed over time for TOP2 and TOP3. For TOP 2, initially the PU traffic was generated with 60 ms inter-packet time (IPT) with Uniform profile. The pattern was changed after 60 seconds when the IPT was increased to 120 ms. Finally, at 100 seconds the traffic was changed to Poisson distribution with 90 ms IPT. The PUI traffic pattern for TOP 3 was started at 80 ms IPT with Poisson profile, and then after 60 seconds the IPT was changed to 50 ms and finally at 100 seconds it was changed to uniform distribution with 90 ms IPT. The whitespace *w(n)* is continually computed by the access protocol after collecting every 1000 whitespace samples.

Observe in Figure 5-8 that while the primary user traffic pattern changes over time, the PTDs resulting from CSTS stay consistently within the vicinity of the pre-set disruption bound of 0.05. Also, note that there is very little PU traffic disruption during the transitions of *w(n)* as indicated in both the frames in Figure 5-8. This is primarily because the whitespace pdf for ad hoc mode 802.11 traffic maintains very similar properties across different traffic patterns, i.e. above 90% of the whitespaces are below 1 ms. Similar results were shown in Figure 5-9 for different traffic patterns and topologies.

Note that the SU throughput increases after an PU traffic pattern transition when the useable whitespace increases due to the increase in inter-packet interval of the PU traffic. Additionally, the SU dynamically throttles back its transmissions to protect the PUs when the available whitespace reduces during a transition. These results demonstrate the robustness of the CSTS access strategy against varying data rates over time. Additionally, the results suggest that by periodically measuring the channel state the SUs are able to dynamically model the channel whitespace and adjust their access parameters accordingly to minimize the PU disruptions and to maximize the SU throughput.

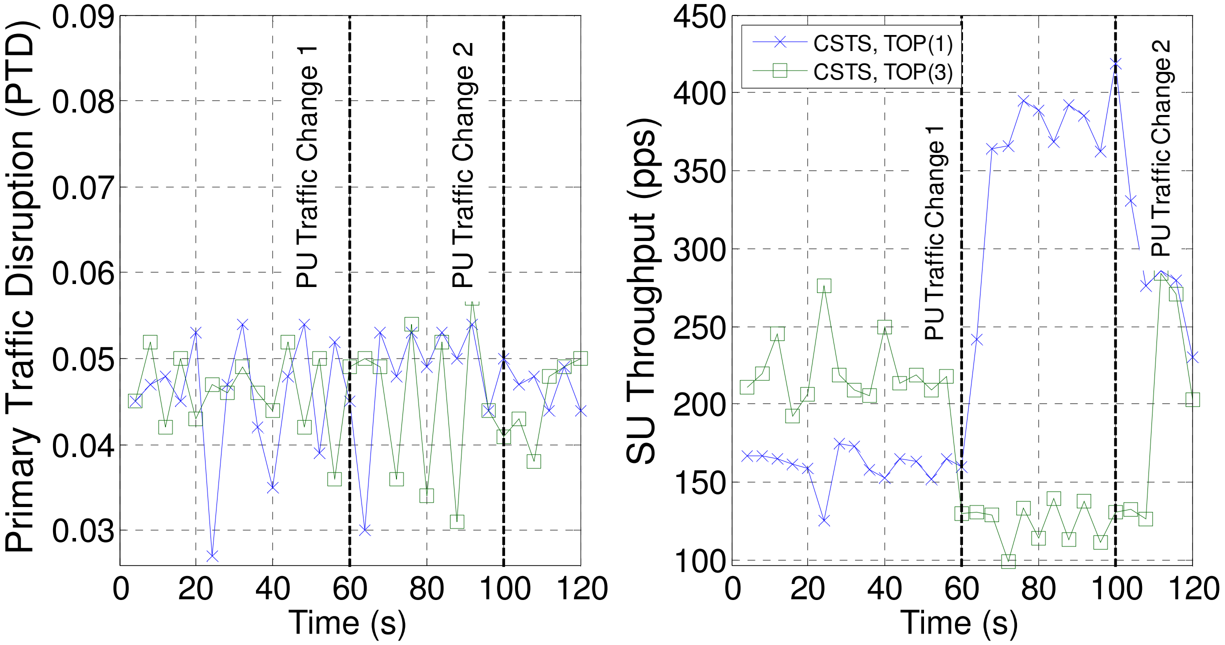


Figure 5-8: Performance with time-varying *w(n)* for TOP2 and TOP3

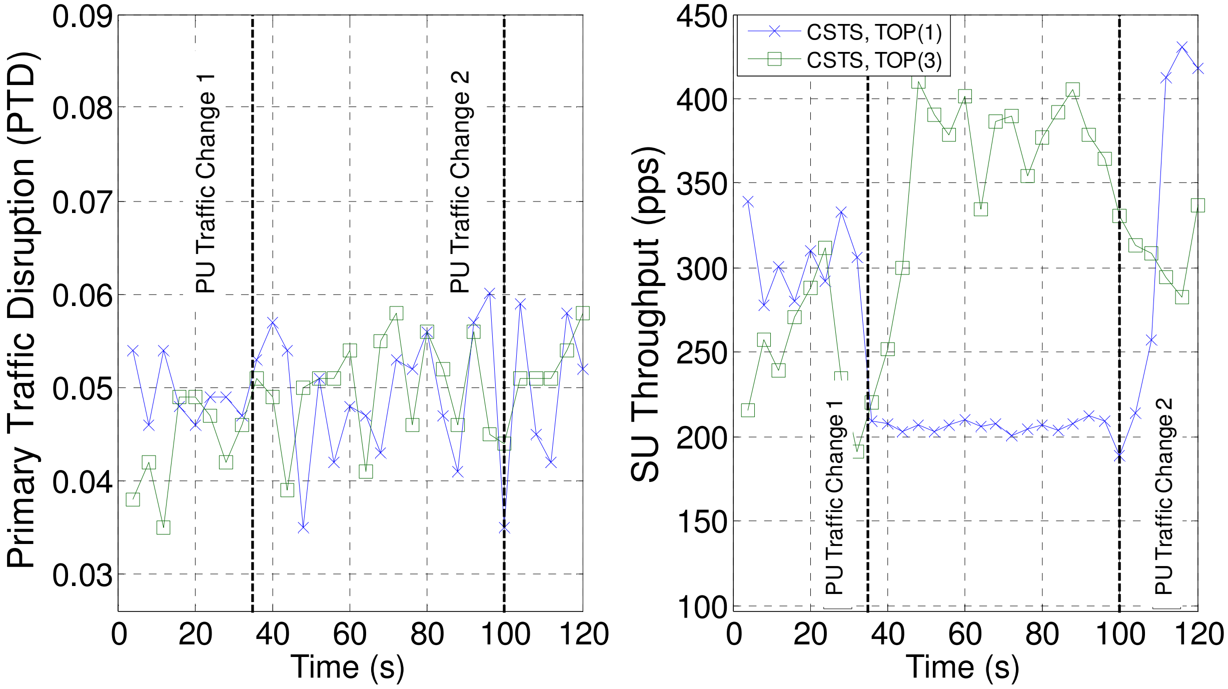


Figure 5-9: Performance with time-varying *w(n)* for TOP1 and TOP3

5.2.4 Channel Sensing Rate

The sensing interval T_p can vary across different hardware and can impact the access protocol performance since it influences the whitespace pdf *w(n)*. In this Section we investigate such performance impacts over a practical sensing interval range from 5 μs to 1 ms [63, 64] for various wireless network interfaces. Figure 5-10 shows such impacts for different topologies with Uniform and Poisson distributed PU traffic and 60 and 70 ms inter-packet arrival times for the CSTS protocol. Both experimentally measured (via ns-2 simulation) and theoretical values (using Equations 4-1 through 4-8), marked as *EXP* and *THY*, are presented. The experimental results match well with the theoretical results, indicating the validity of the experimental observations. Observe that as T_p increases, the EST and PTD remain relatively constant since with T_p within the range of 5 μs to 1 ms, the SUs can detect most of the whitespaces that are smaller than the SUs' packet size, which is 0.6 ms in this case. In the CSTS protocols, an SU does not access a whitespace until the whitespace lasts for at least the T_p duration. In other words, with larger T_p, the SUs behave more conservatively, leading to higher EST and lower PTD. These results indicate that the proposed access strategies can work under varying channel sensing rates that are well within the practical range for the currently available 802.11 hardware [63].

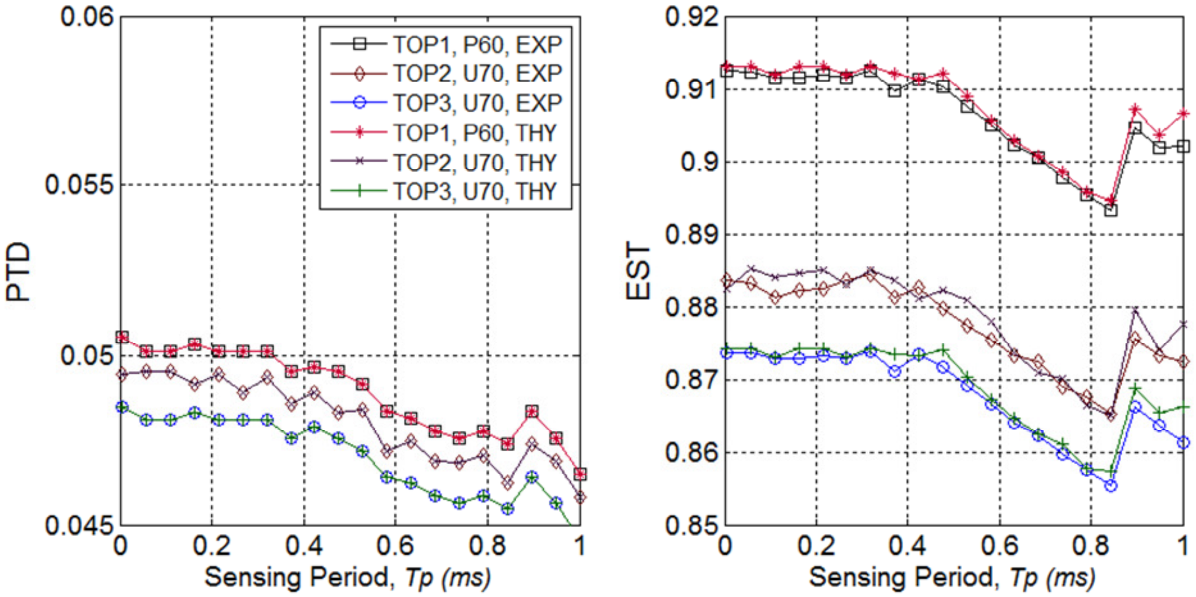


Figure 5-10: Impacts of channel sensing interval T_p

5.2.5 Disruption Bound

Depending on the specific PU applications, the SU access strategies may need to run with different disruption bounds for the PUs. Figure 5-11 and Figure 5-12 presents the performance with varying pre-defined disruption bound or *DB* values for Uniform and Poisson distributions respectively. PU topologies TOP1-3 in Figure 5-1 and inter-packet arrival rates of 60 and 90 ms are evaluated for the CSTS protocol. The left graph in Figure 5-11 validates that the protocol is able to maintain the PTD within the pre-set *DB* for a wide range of *DB* values. The right graph depicts that by allowing larger *DB*s it is possible for the SUs to scavenge more bandwidth from the channel. Another interesting trend in this graph is that the EST rises quite sharply with higher disruption bounds. A practical implication of this large slope is that if the PUs can tolerate a very small amount of disruption (e.g. $\approx 5\%$), reasonably high SU throughput can be achieved. This is in contrast to the very low SU throughputs when the PUs cannot tolerate any disruptions.

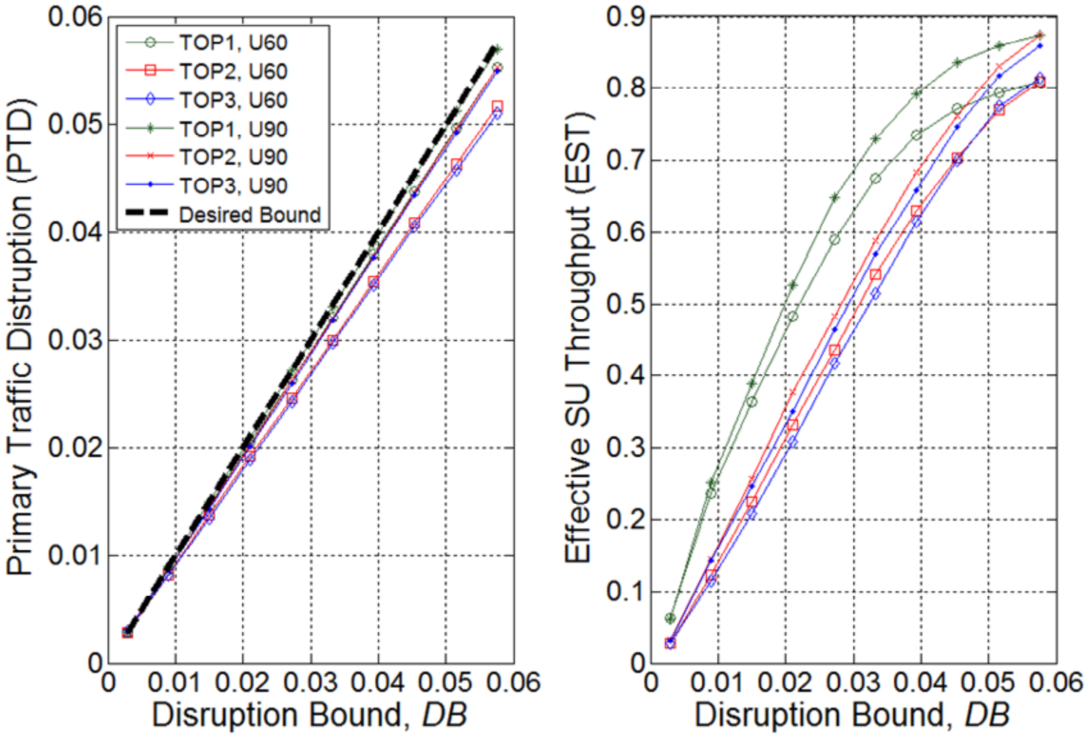


Figure 5-11: Impacts of Disruption Bound for the uniformly distributed PUs

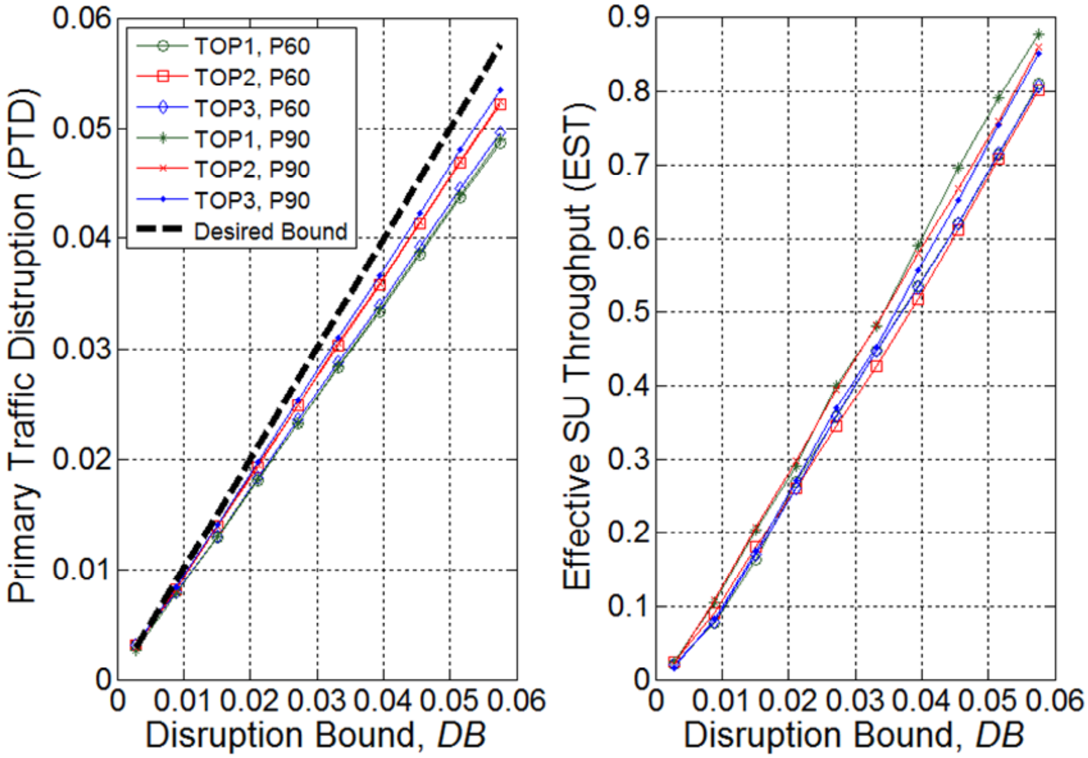


Figure 5-12: Impacts of Disruption Bound for the Poisson distributed PUs

5.3 Summary and Conclusions

This chapter presented an extensive set of results to evaluate the CSTS access strategy. The simulation-based experiments provided a method to explore the impacts of different parameters on CSTS. The primary user load results showed that the protocol is able to maintain high SU throughput under different load conditions. Results from the SU packet size experiments showed that a smaller SU packet provides better throughput for the SUs. Varying PU traffic pattern results showed that CSTS could dynamically adjust to the PU traffic over time. Additionally, periodic sampling of the whitespace provides enough information to maintain good SU throughput while maintain PU disruption bounds. The channel sensing interval results showed that as long as the sensing period is smaller than usable whitespaces, there was enough information to operate the CSTS protocol. Finally, the experiments on the effects of varying disruption bound shows that PUs allowing a small disruption can facilitate more bandwidth scavenging by the SUs.

Chapter 6: Traffic Protection in a Prototype CSMA based Sensor Networks

6.1 Introduction

In this chapter, the Bandwidth Scavenging Concept (BCS) is applied to traffic protection in CSMA based sensor networks. The BSC is implemented in an experimental testbed using the Crossbow TelosB sensor motes [65]. The previously presented 802.11-based simulated network allows explorations of the effects of various traffic and topology conditions while the experimental setup discussed in this chapter provides insights into real world measurement issues. The chapter is organized as follows. First, an overview is given of the TelosB platform. Then the PU network is described in detail. Finally, the SU network is discussed. Software details about the prototype is added in Appendix A.

6.1.1 Motivation

Applications utilizing Wireless Sensor Networks (WSNs) can be categorized into real time and non real time. Real time applications such as event surveillance typically require constant or variable rate data streams to be sent over the sensor network. Depending on the specific application, the rate of this data can be of the order of a few packets per second to tens of packets per second. The objective is to transport such real-time application data packets with minimum delivery delay and packet losses [66]. Non-real time applications such as environmental monitoring need to measure levels of temperature, moisture and other ambient parameters in a time driven or event driven manner. Such applications typically require lower data rates compared to the real-time data streams and have less stringent packet delay and loss requirements.

6.1.2 Problem Definition

A heterogeneous WSN may often require supporting both real time and non real time applications. In this type of networks, nodes that support real time applications must coexist with the nodes that are running non-real time application. In order to protect the real-time traffic from large channel access delay caused due to interruptions from the non real-time traffic, an access priority structure is usually needed.

Now, if a network designer wants to ensure that the requirements of the real time traffic network are met in this heterogeneous WSN, a traffic priority structure can be used. In this case, the real time traffic is given priority access to channel over the non-real time traffic. To address this type of traffic prioritization problem in general, some issues that would need to be addressed are:

- Maintaining a priority structure: Support is needed to facilitate different priority levels for the various sets of network devices
- Supporting higher priority requirements: Data requirements of the higher priority traffic must be met
- Limiting the effects of delaying the higher priority traffic: When lower priority traffic has data to send, the impacts on the higher priority traffic must be minimized
- Supporting different high priority traffic loads: As the high priority traffic changes, the lower priority traffic should adjust its usage of bandwidth accordingly
- Maximizing the throughput of the lower priority traffic: Lower priority traffic throughput need to be maximized given the available channel bandwidth.

Channel history-based mechanisms, namely the Bandwidth Scavenging Concept, may provide a suitable solution due to its ability to dynamically adjust lower priority traffic access to protect higher priority traffic.

6.1.3 Related Work

In baseline 802.11 access [57] when a message arrives from the upper layer, the MAC layer accesses the channel with a DIFS (DCF Inter-Frame Spacing) and Contention Window (CW) duration irrespective of the traffic class. Traffic priority has been introduced in 802.11e [18], by using class specific (referred to as Access Categories or ACs) DIFS and CW values. The quantity DIFS in 802.11e is parameterized as arbitration inter-frame space with specified access classes (AC). Similarly, the CW is redefined with CW_{min}[AC] and CW_{max}[AC] allowing for different priority traffic to be assigned with different overlapping back-off times. The addition of class-specific AIFS and CW values allow for prioritizations since nodes with smaller AIFS periods and lower CW ranges will access the channel first and more often. This core 802.11e prioritization concept has been refined in [19, 67] which propose mechanisms for choosing the class-specific AIFS and CW values dynamically under different network conditions.

As discussed in Section I.B, although 802.11e (based) protocol introduces priority to network traffic, there is no guarantee that a higher priority node's transmission will not be deferred if it arrives at the MAC layer when a lower priority node is in the middle of its packet transmission. Our proposed protocol attempts to reduce such high priority traffic deferrals by allowing low priority nodes to attempt channel access only when it estimates the likelihood of such disruption is low.

A number of papers [21, 22, 68] have proposed traffic prioritization schemes specifically designed for WSNs. The mechanism in [68] is similar to 802.11e in using variable size contention windows and different inter-frame spacing for different types of sensing events. To further differentiate the priority events, the authors improve upon their original design by adding a period where each higher priority event node randomly sends a burst of pulse just after their respective AIFS periods to notify lower priority nodes to backoff until the transmission is finished. While being customized specifically for WSNs, this protocol [11] can suffer from the same 802.11e issues as discussed in Section I.B.

The works in [22] and [21] use time slots similar to TDMA to assign priority to traffic classes. The protocol in [21] divides MAC frames into random access and scheduled access sections. The scheduled access sections are further divided into priority groups with multiple transmission slots in each group. A node broadcasts their traffic priority during the random access period, which reserves their transmissions for specified priority groups in the scheduled section. The protocol PMAC [22] presents another TDMA based prioritization MAC utilizing a wide range of prefix reservation and inter-slot intervals. Both of these TDMA protocol require tight time synchronization to implement priority. In contrast, our approach does not require time synchronization, thus offering a more practical solution that is suitable for ad hoc deployment.

Other dynamic spectrum access testbeds has also been developed [69-76].

6.2 Testbed Setup

In this section, an overview of the experiment testbed is presented. The specific implementation details of the testbed are presented in the Appendix. Figure 6-1 shows an overview on the implementation. In the Bandwidth Scavenging Concept, a secondary user device attempts to access a wireless channel that is occupied by a primary user device. We implement these two separate networks as two independent systems, namely the Primary User Network (PUN) and Secondary User Network (SUN). The purpose of PUN is to generate various traffic patterns on a channel. While the purpose of the SUN is to first perform measurements of the PU channel activity then utilize an access strategy to send messages in between the PU transmissions.

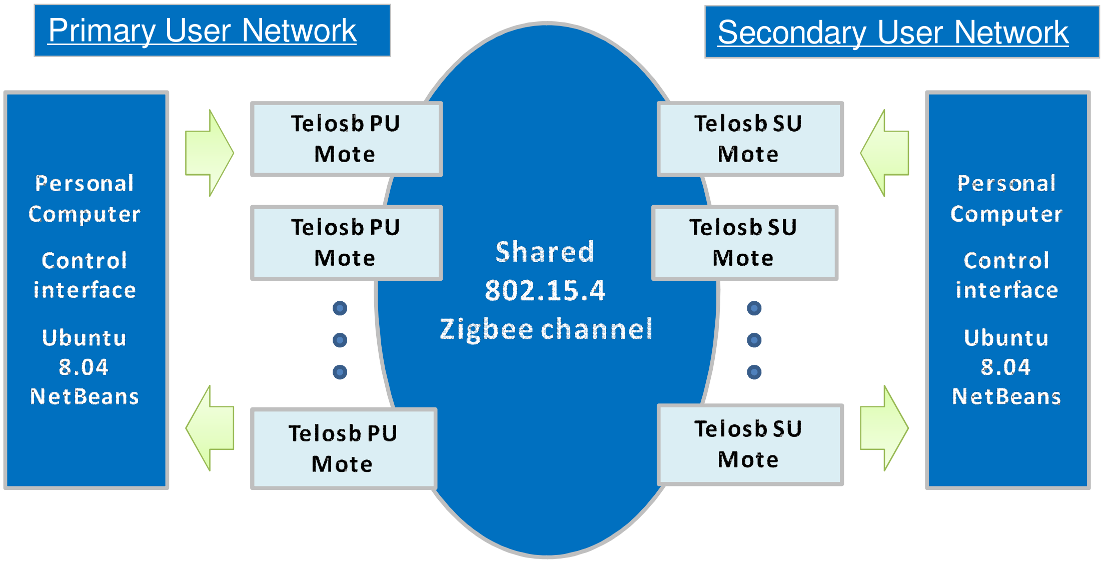


Figure 6-1: System implementation overview

As shown in Figure 6-1, the PUN and SUN is completely independent of each other, although they share an IEEE 802.15.4 Zigbee channel. Crossbox Telosb motes [65] are used as the primary and secondary devices. TelosB was used since the TinyOS port on it provides enough driver level APIs for implementing a custom MAC layer (i.e. CSMA in this case), which is not easily possible for majority of the commercially available 802.11 cards. The communication and programming of the motes are done using the TinyOS 2.1 operating system. For the purposes of traffic generation and statistics gathering, the motes are connected to personal computers running Ubuntu Linux 8.04 LTS. Separate computers are used for the PUN and SUN. The implementation of the computer interfaces is done using java with NetBeans as the IDE.

The communication flow of the experimental network is done using two closed paths. In the PUN path, an PU traffic pattern is generated on the primary user computer and then sent to an PU mote. This mote will send the traffic to another PU mote or multiple motes. When the packets reach the last mote in the path, the packets are forwarded back to the originating PC. The SUN is designed in a similar fashion, by performing a second path using the SU motes and SU computer. Performance metrics for the experimental testbed are gathered on each PUN and SUN computers.

6.2.1 Crossbow TelosB Motes

The Crossbow TelosB Mote hardware consists of a small circuit board with several microcontrollers, three LED's, two physical buttons, a USB connector, connectors for sensors board and a battery pack. This platform was developed by a research community at UC Berkeley and is now sold by several manufacturers.

The motes are running TinyOS as the operating system. TinyOS provides a modular operating system for Wireless Sensor Networks that supports several sensor platforms. TinyOS is maintained and developed by the TinyOS Alliance. This Alliance is structured in working groups consisting of researchers from several universities and companies.

6.2.2 Hardware Components and Network Topology

The testbed is currently setup using two personal computers and 4 TelosB motes. To ensure undistorted results in the measurements, the implementation of the PU and SU networks has been installed on separate computers.

Both computers are running:

- Ubuntu Linux 8.04 LTS, RealTime kernel rt-2.6
- TinyOS 2.1.0 with gedit and gcc programs
- Java 1.6
- NetBeans IDE 6.8

The PUN computer has a dual core Intel processor with 2 GB of ram and the SUN is an Intel Celeron with 768 MB ram.

The topology of the test-bed is kept as simple as possible to keep unwanted effects as low as possible. Figure 6-2 shows the physical layout of the test-bed. On the left side are two Primary User motes. The top left node (PU Sim) is sending the test traffic pattern to the bottom left node. While these motes are sending the bottom right mote (USB0 SUTX) accesses the channel and sends messages to the top right mote. The top right mote is called Basesation because it forwards all messages to the PC to create a full log of the tests.

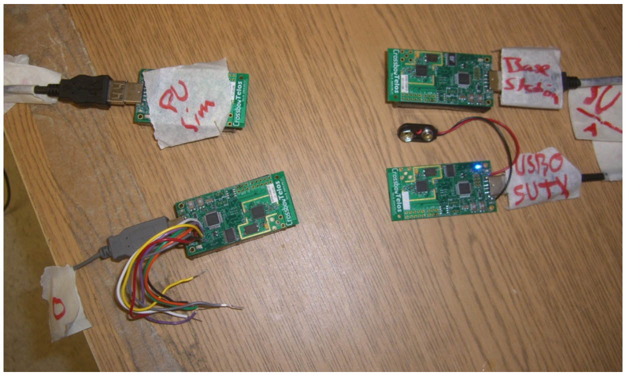


Figure 6-2: Physical topology of the primary and secondary networks

6.3 Primary User Traffic Profiles

To explore the characteristics of different PU traffic, various traffic profiles are sent by a PU and subsequently measured by an SU. A PU transmitter sends messages to a PU receiver according to the given traffic profile. Concurrently, an SU measures the channel using an RSSI sampling program implemented on the motes. The channel is measured at 1 kHz rate with RSSI values being reported at each time step [77]. Consecutive measured RSSI values below a pre-defined threshold (-50 dBm) are considered as whitespaces.

In order to represent PU traffic diversity, three traffic scenarios are established i.e. sent over the PU network:

- A uniformly distributed traffic that is sent at 7 packets per second (pps). Each packet has an added ±20% inter-packet time to vary the arrival time on the channel. This profile corresponds to TelosB based sensor network applications involving constant bit rate sensing.
- A low rate video file streamed over the channel, which represents video surveillance applications.
- A multi-modal traffic with two "peaks" of whitespaces in the pdf *w(n)*. In general, a stream with multiple peaks in its whitespace can be formed by merging multiple constant packet rate streams. This is representative of a multi-application primary traffic at different traffic generation rates.

The packet size of the PU node is set to 120 bytes, which corresponds to approximately 5 ms channel access duration given the bandwidth of the channel (250 kbps data rate). Given that the SU senses the channel at $T_p = 1$ ms intervals, it is able to determine the approximate start and ending times of PU packets (i.e. ON-OFF durations). Figure 6-3a-c shows the whitespace pdfs *w(n)* measured and computed by an SU Transmitter, which is within the transmission range of the PU Transmitter that is generating the traffic profiles as described above. In Figure 6-3a-c, the pdf is plotted as a function of the sensing interval time, i.e. nT_p.

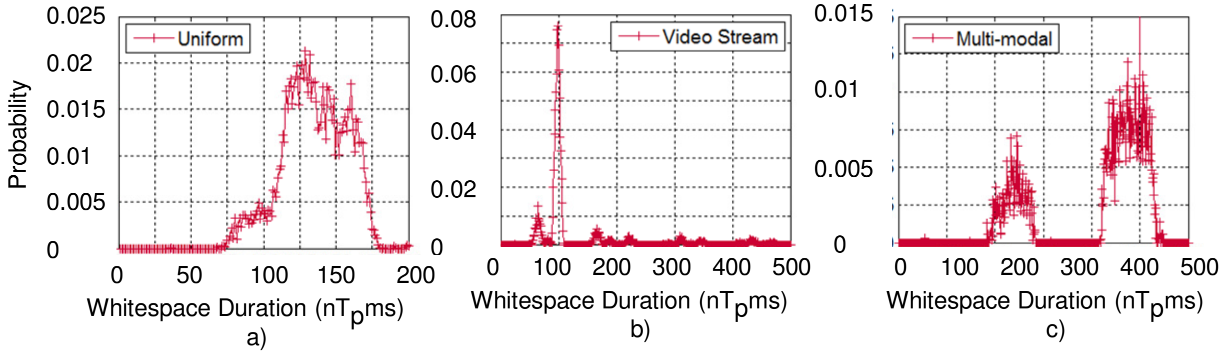


Figure 6-3: Experimental whitespace distributions. (a) Uniform with 20% variation, (b) Low rate video stream, and (c) Multi- modal data stream

The following observations can be made from the whitespace statistics shown in Figure 6-3. In the uniform traffic case of Figure 6-3a, all the whitespaces are clustered between 70 ms and 180 ms, which is the result of the 20% variation on the 7 pps traffic generation rate. Observe that the peak does not form a perfect uniform shape as expected. This result comes from the combined effects of the TinyOS operating system's scheduling jitter and the asymmetry in the hardware delays in the transmission times of the packets. For the pdf of the video stream in Figure 6-3b, the large peak corresponds to fixed rate video packets. The other small peaks correspond to embedded audio packets and uncorrelated video packets, which are transmitted at a slower rate. For the multi-modal pdf case in Figure 6-3c, the whitespace is distributed within two separate ranges. The two peaks correspond to applications that generate a traffic rate of 4 pps with 25% spread, and 2 pps with 75% spread respectively.

The analysis of whitespace in the experimental network highlights the various whitespace environments that an SU may experience when attempting to access unused spectrum.

6.4 Performance Evaluation of CSTS on Prototype Testbed

In this section, the performance of the CSTS strategy is evaluated experimentally using the testbed presented in this chapter and using a Matlab based simulator discussed in Section 5.1.2.

6.4.1 Compared Protocols

We compare CSTS with an inter-frame spacing based protocol (AIFS). The AIFS is based on IEEE 802.11e [18] where priority is given between classes of traffic by waiting class-specific inter-frame spacing (IFS) before sending a packet. When the SU has a packet to send, using AIFS, it waits for the specified IFS time for the channel to remain free before transmitting. In our implementation, a zero IFS was chosen for the PUs and a non-zero IFS was chosen for the SUs. Like in 802.11e, this arrangement gives rise to priority channel access to the PUs. The IFS for the SU is chosen as half the SU packet size, which provided the best performance for the AIFS.

In the simulation results, a benchmark protocol is also plotted. The Benchmark protocol assumes full knowledge of the starting and ending times of every whitespace. Therefore, it transmits SU packets in every available whitespace and stops when the next SU transmission will cause disruption to the PUs. The Benchmark protocol can be run only offline to find the best SU performance. This protocol essentially determines the highest throughput an SU can expect without causing any disruption to the PUs.

6.4.2 Experiment Implementation

The evaluation of the access strategies on different traffic profiles and data rates required a large number of experiments. In contrast to a simulation environment, testbed experiments are run in real time, which causes the runtime to be significantly high. In the experiments, 5000 primary user sample-packets were used. The time necessary to create a single graph point is over

80 min. Hence the creation of a full graph can take up to over 13 hours of runtime ((5000 packets * 5 points in the graph * 2 Access strategies) / (60 packets/min)). This condition required the automation of the experiments. First, all of the Java based GUI elements were removed and replaced with a command line interface. Then AWK batch scripts where used to set all of the necessary experimental parameters. Next, the two Primary and Secondary computers must be synced up together during the running of an experiment. This task was accomplished through networking the computers together with a shared folder. In the folder, files were passed between the computers to signal the starting, ending, and data collection periods of the experiments. This setup allowed experiments to be run over night.

During the experiments, the following values are captured: the Round Trip Time (RTT) of each PU packet, the PU packet losses and a full packet log. From these values, the packets deferred are calculated by counting the packets with a RTT > 100 ms plus the packets lost. The indicator for the secondary user performance is the Measured Secondary User Throughput (MST), which is calculated from the full packet log. In the figures, the AIFS protocol is referred to as priority based and the CSTS is referred to as History based.

6.5 Performance with Uniform PU traffic Profile

Figure 6-4 and Figure 6-5 show the percentage of PU packets that where deferred because of the AIFS and CSTS protocols (referred to as Priority and History based respectively). The PU traffic profile is a uniform distribution at 120 packets per minute (ppm) similar to the distribution shown in Figure 6-3. The traffic load generated by the SU is varied in order to evaluate the impacts of increasing non-priority traffic on the priority traffic. Observe that CSTS has a low percentage of deferred PU packets across all SU rates, while AIFS suffers from greater disruptions. Figure 6-4 shows the actual percentage and Figure 6-5 show the same data on a Log-

Log scale. The disruptions to the PU traffic are quite low for the CSTS strategy. AIFS suffers from greater disruptions since the SU will always transmit after waiting for an IFS. Even after an IFS period, there is some probability that the PU will return to the channel during the SU transmission, causing disruptions for the PU traffic. Figure 6-6 shows similar results through the Round Trip Time.

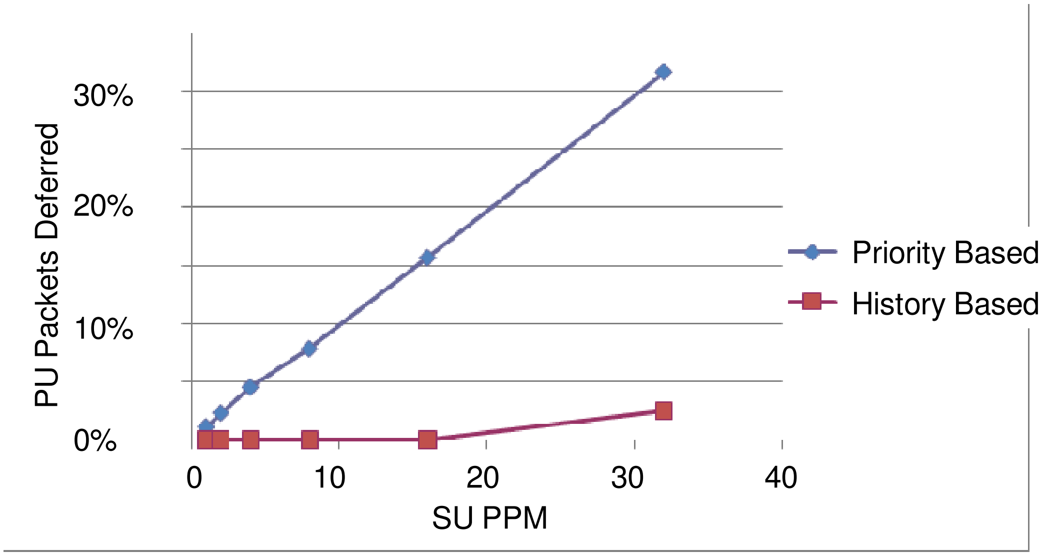


Figure 6-4: PU packets deferred for 120 ppm uniform traffic

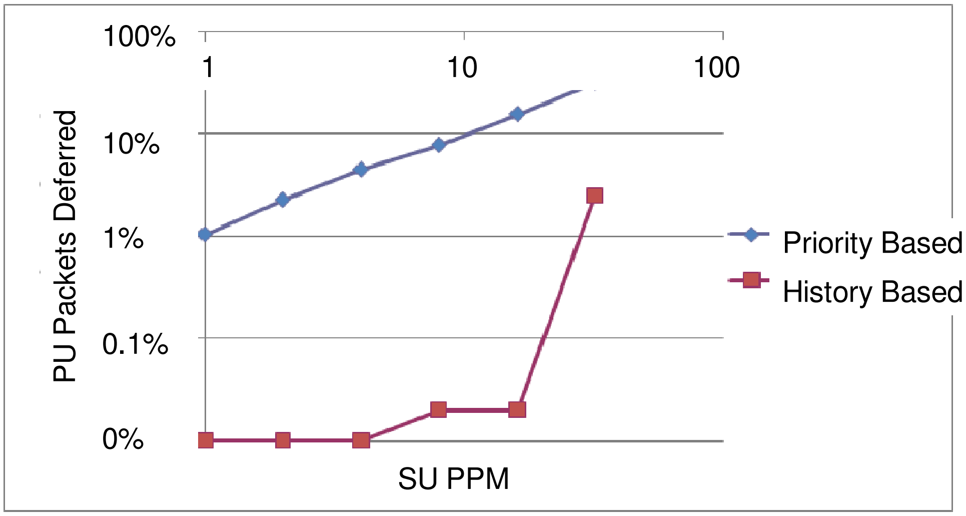


Figure 6-5: PU packets deferred for 120 ppm uniform traffic in Log scale

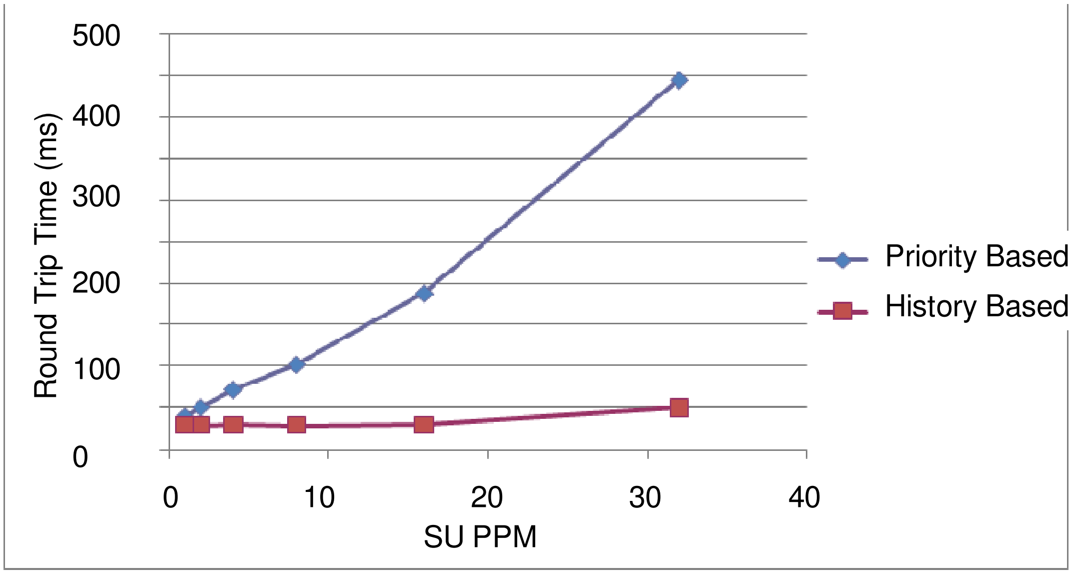


Figure 6-6: RTT for 120 ppm uniform traffic

Figure 6-7 shows the Measured SU throughput (MST) results for the experiment. Observe that the as the SU ppm increases the throughput increases. Note that due to the limitations of the TelosB motes, the channel cannot be completely filled by either primary or secondary user traffic. This limitation was discussed in [78]. Therefore to capture the effects of the completely filling the channel simulation based experiments were performed.

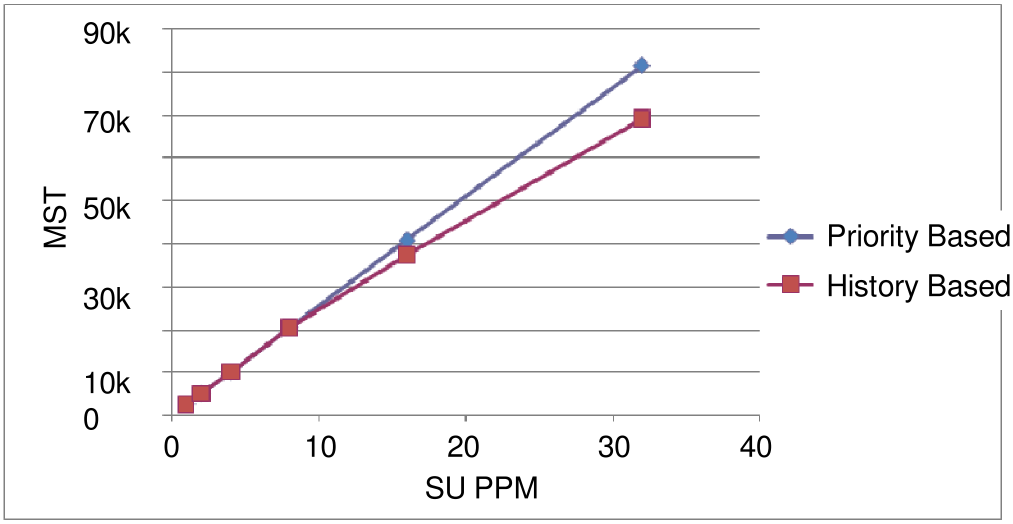


Figure 6-7: MST for 120 ppm uniform traffic

Figure 6-8 show the simulation results for the 120 ppm uniform scenario. This simulation result was created by using the RSSI trace files from the testbed. Observe that CSTS maintain the required disruption bounds (indicated as DBs in the figure) across all SU rates, while AIFS suffers from greater priority traffic disruptions, which is above 10% for the higher SU rates. In terms of throughput, the access strategies are able to support low SU generation rates. However, once the specified DB is reached, CSTS throttle the SU transmissions in order to protect the PU traffic. Notice that AIFS is able to achieve higher SU throughput by always sending after waiting for an IFS, but at the expense of higher disruptions to the PU traffic. This demonstrates how CSTS are able to protect the high priority traffic by limiting the low priority traffic throughput whenever appropriate.

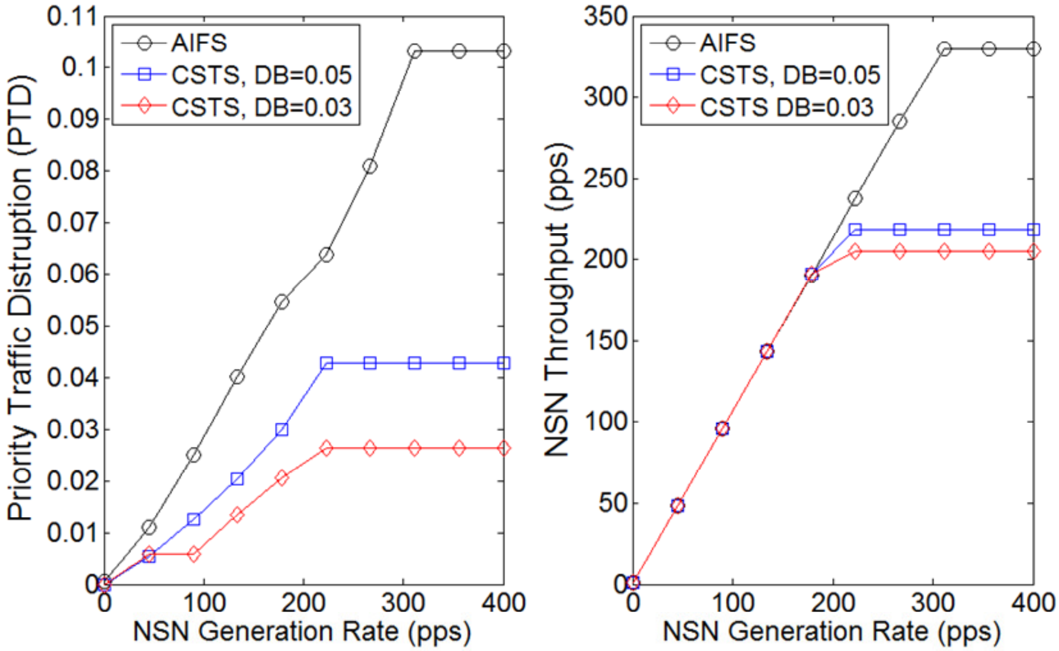


Figure 6-8: Simulation results for 120 ppm uniform traffic

We show the impacts of PU traffic rates in Figure 6-9. PU traffic with the uniform profile is varied from 1 packet per second (pps) to 7 pps. In the presented results, the SU generation rate is set to a high value, which translates to the SU always having packets buffered to send for the entire duration of a whitespace. Observe that as the PU traffic rate increases, CSTS is able to maintain the specified PU disruption bounds while the PTD values for the AIFS protocol are quite high. This is because CSTS create different parameters (J_{max}) for each PU traffic rate to maintain the desired bound. As expected, as the PU rate increases, there are less whitespaces for the SU to access the channel, hence lowering the SU throughput.

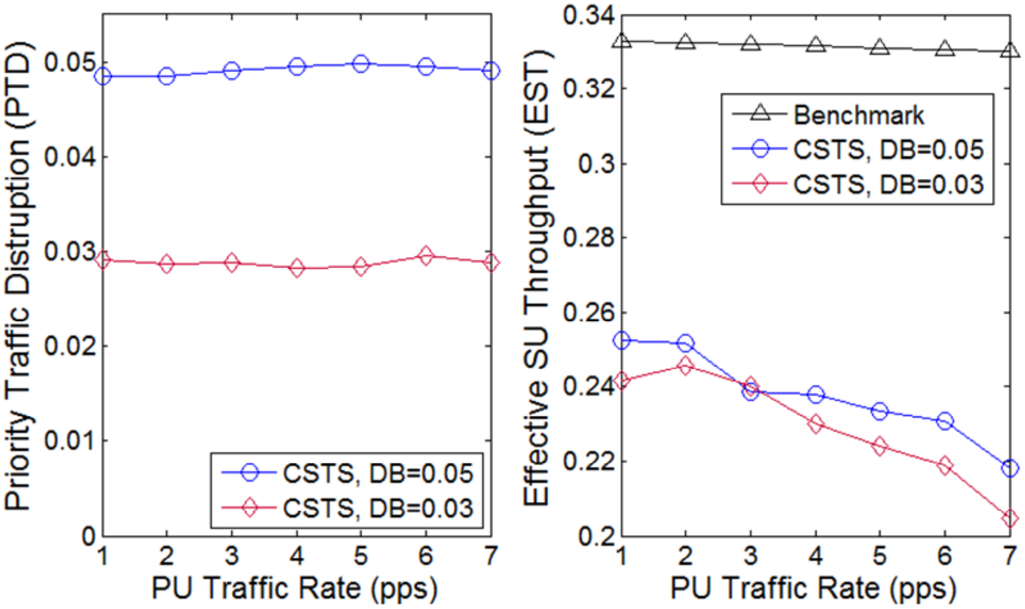


Figure 6-9: Simulation results for various uniform traffic rates

6.6 Performance with PU Traffic with Low-rate Video

The protocol performance with PU traffic from a low-rate video stream (as shown the pdf in Figure 6-3b) is shown in Figure 6-10 through Figure 6-12. Note that the relative performance of the three evaluated protocols in this case is very similar to those observed for the uniform traffic case in the previous section. As in the uniform traffic case, the CSTS protocols are able to maintain the specified DB at the expense of limiting the SU throughput at higher SU load levels. In terms of throughput, the values are smaller that the uniform case. This results from different availability of whitespace in the video stream profile. The simulation results show similar trends.

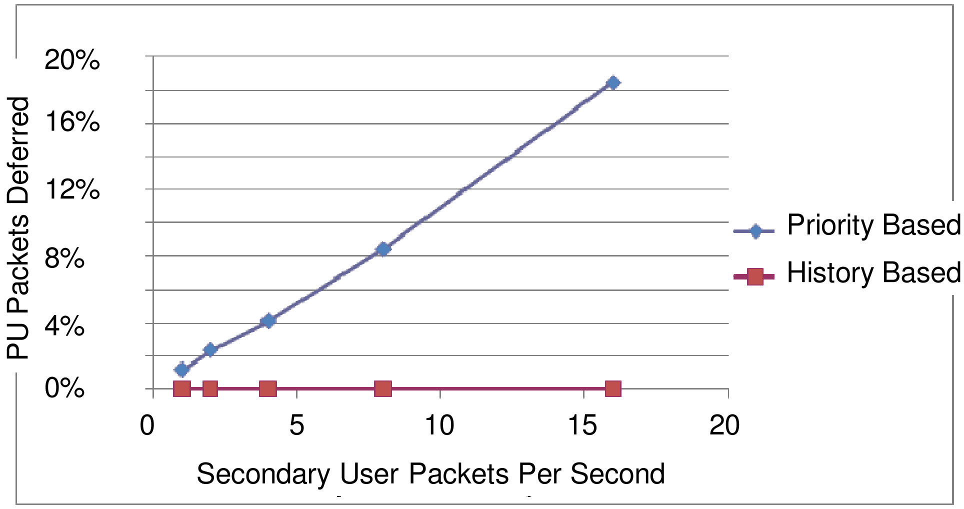


Figure 6-10: PU packets deferred for video stream based traffic

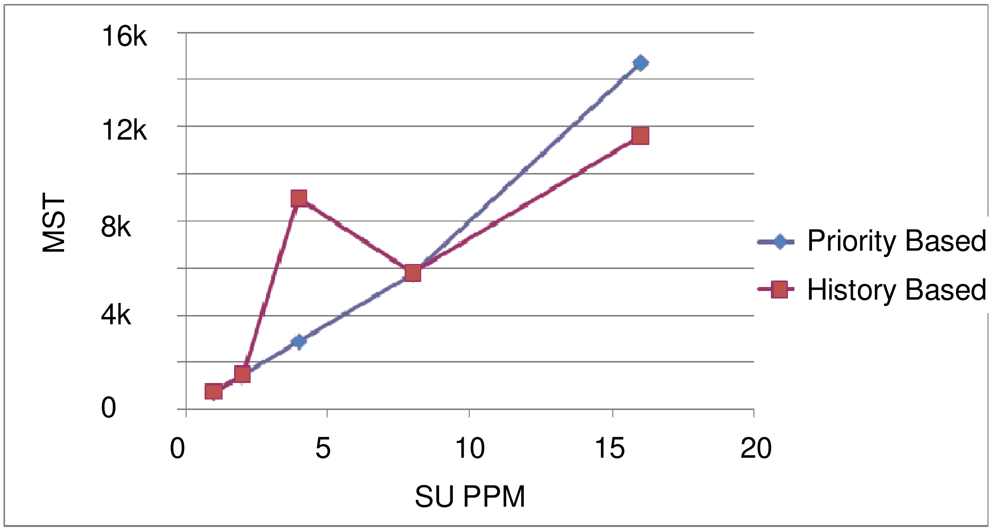


Figure 6-11: MST for video stream based traffic

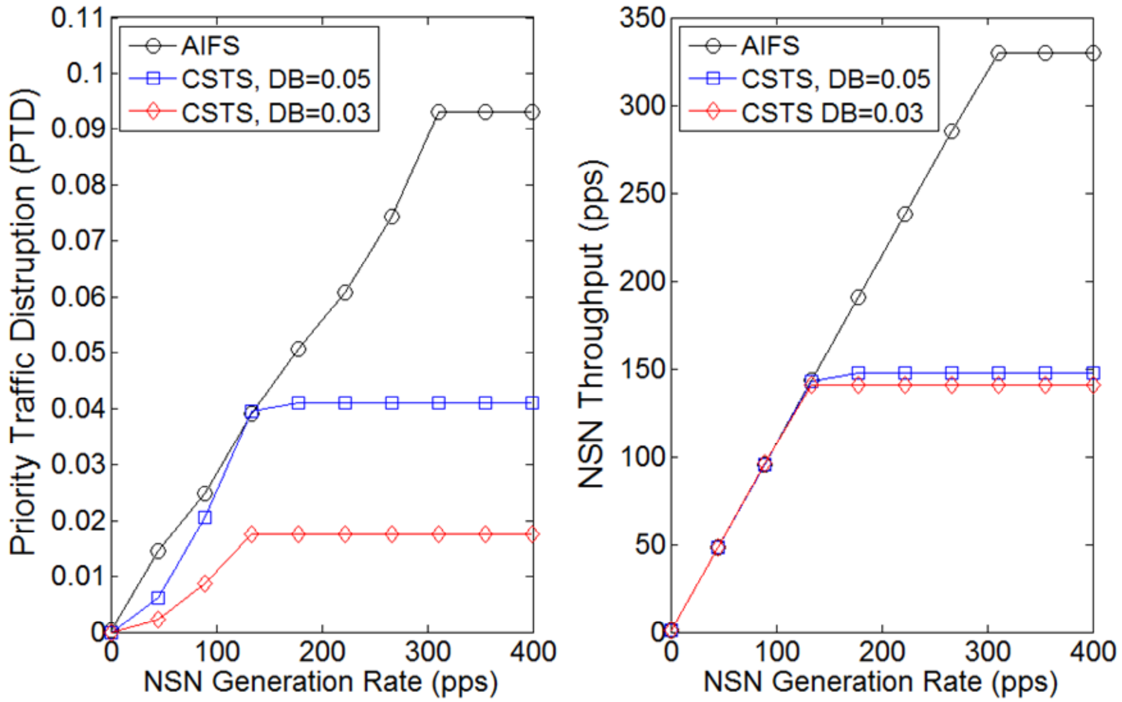


Figure 6-12: Simulation results for video stream based traffic

6.7 Performance with Multi-modal PU Traffic

Similar results to the video stream profile are demonstrated in the multi-modal case (Figure 6-3c) presented in Figure 6-13 through Figure 6-15. Again the CSTS protocol maintains the required PU traffic disruption bounds while the maximizing the SU throughput.

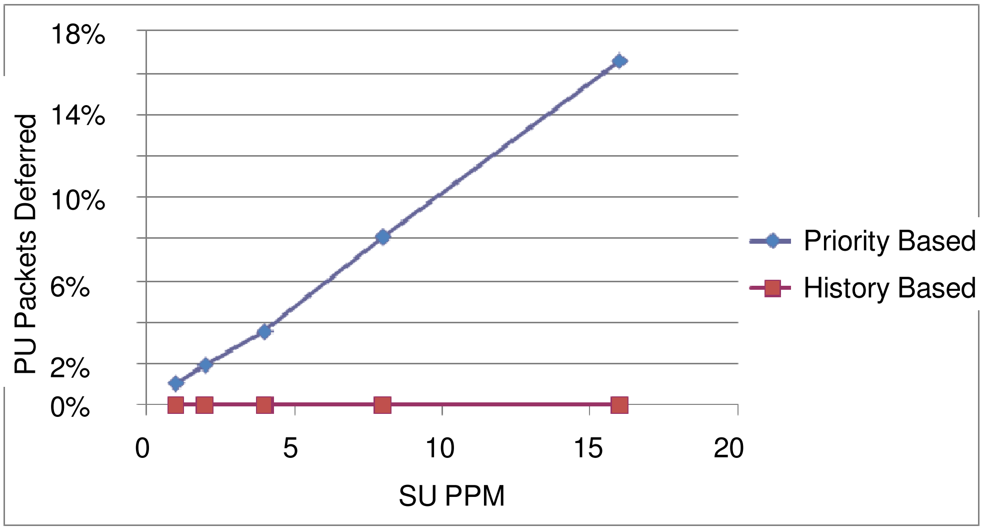


Figure 6-13: PU packets deferred for multi-modal based traffic

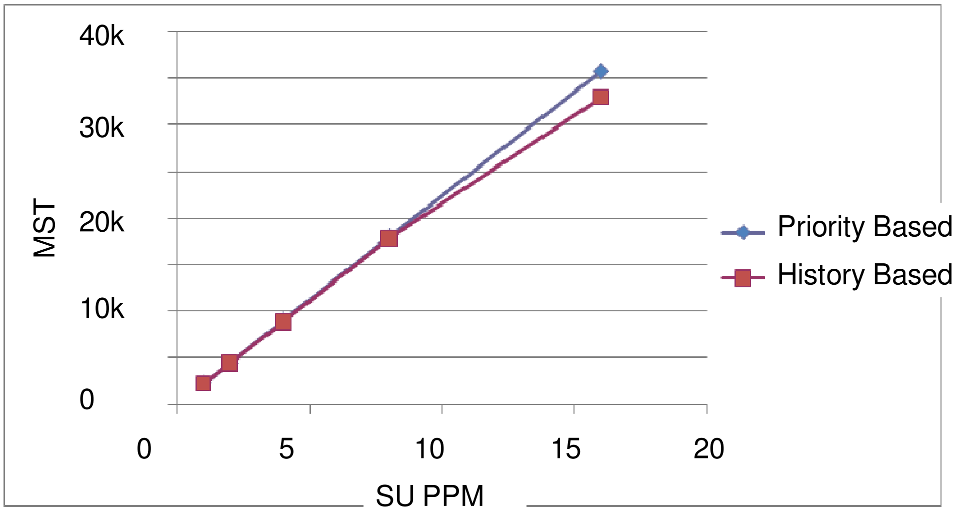


Figure 6-14: MST for multi-modal based traffic

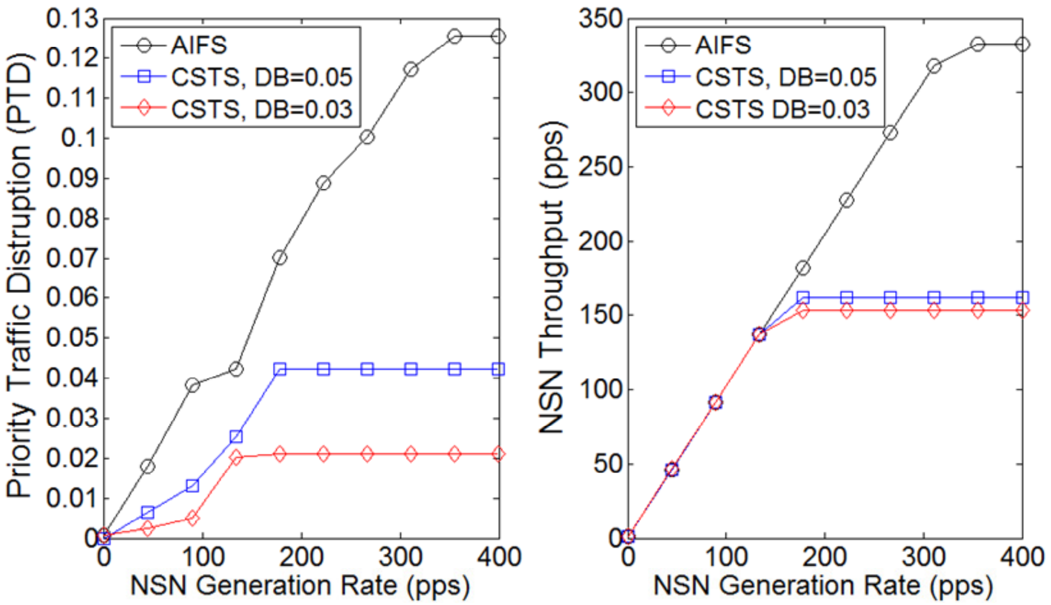


Figure 6-15: Simulation results for multi-modal based traffic

6.8 Summary and Conclusions

This chapter presented a TinyOS based testbed implementation of the Bandwidth Scavenging Concept. A Primary User Network (PUN) was implemented with the purpose of generating various traffic patterns on a channel. Along with a Secondary User Network (SUN) was created to perform measurements of the PU channel activity. Then utilize an access strategy to send messages in between the PU transmissions. The PUN was shown to be able to create multiple traffic patterns using a PC and TelosB mote configurations. The SUN was shown to be able to measure the PU transmissions on the channel. Additionally, multiple SU access strategies were implemented on the motes.

The access strategies were then evaluated using the testbed that was implemented using TelosB motes. From the experimental results, it was shown that the access strategy was able to support the offered SU load. Due to the limitations of the TelosB hardware, additional simulations were run in Matlab. The simulation results showed that the CSTS protocol would throttle back the transmissions of the SUs to protect the PU traffic. Similar to the 802.11 results the Bandwidth Scavenging Concept was shown to minimize PU throughput while maximizing SU throughput.

Chapter 7: Divided SU Transmission Strategy (DSTS)

7.1 Introduction

7.1.1 Motivation

The whitespace profile of a general PU network can be a function of an individual PU properties such as topology, traffic pattern, and data rates. This creates a major challenge for SUs attempting to access the available whitespace. Therefore, an SU access strategy that can adapt to the various whitespace profiles is needed.

7.1.2 Problem Definition

The CSTS protocol that was used in Chapter 4 through Chapter 6 was designed to take advantage of the specific whitespace characteristics in an Ad hoc mode 802.11 network. As a result, in networks with non-802.11 traffic, it may miss *transmission opportunities* because of an over-conservative sending limit that is imposed based on the assumption of small whitespaces corresponding to ad hoc mode 802.11 control packets. The proposed Divided Secondary user Transmission Strategy (DSTS) in this chapter removes this limitation of CSTS.

7.1.3 Related Work

The authors in [16, 17] developed a methodology for formally analyzing the whitespace available within 802.11 primary traffic in infrastructure mode. The key idea in their methodology is to model the whitespace as a semi-Markov process that relies on the underlying 802.11 state model involving DIFS, SIFS, DATA, and ACK transactions. Their semi-Markov model describes the whitespace in terms of holding times of the idle and busy states of the channel. Building on this model, in [6] the authors further develop an WLAN access strategy for secondary users in which the SUs utilize packet-size slots for the channel access. At the beginning of each slot, an SU senses the channel and if the channel is free then it transmits with a specified probability that is calculated from previous measurement. In our proposed access mechanism, a SU utilizes whitespaces based on a stochastic model that minimize PU disruption and maximize SU throughput.

The authors in [14] developed an optimal access strategy for secondary users. This work has certain similarities with ours in that both approaches attempt to maximize the SU throughput while keeping the PU disruption below a pre-determined bound. However, there are a number of key differences as listed below. First, we formulate the problem in a 0-1 knapsack framework and solve it using dynamic programming whereas in [14], a threshold based mechanism is used. Second, we evaluate our access strategy in a realistic scenario where multiple SUs exist within a single collision domain whereas in [14], the authors consider two isolated and independent SUs in separate collision domains. Finally, the characterization of whitespace profiles under a diverse set of traffic profiles using ns-2 simulations and experimental testbed is a unique feature of our work compared to [14].

7.2 Access Based on Transmission Opportunities

A Transmission Opportunity (TO) within a whitespace is defined as a period of time that is large enough to send a single secondary user packet. Clearly, there may be multiple TOs for sending multiple SU packets within a whitespace. However, since the duration of a whitespace is unknown in advance, there is a non-zero probability for an SU transmission to spill over the end-time of the whitespace, causing a PU disruption. The objective is to reduce the occurrences of such transmissions within a whitespace so that the chance of PU traffic disruption is minimized.

Conversely, that reduction needs to be selective so that the SU throughput can be maximized by judiciously scavenging for the legitimate TOs within the whitespace.

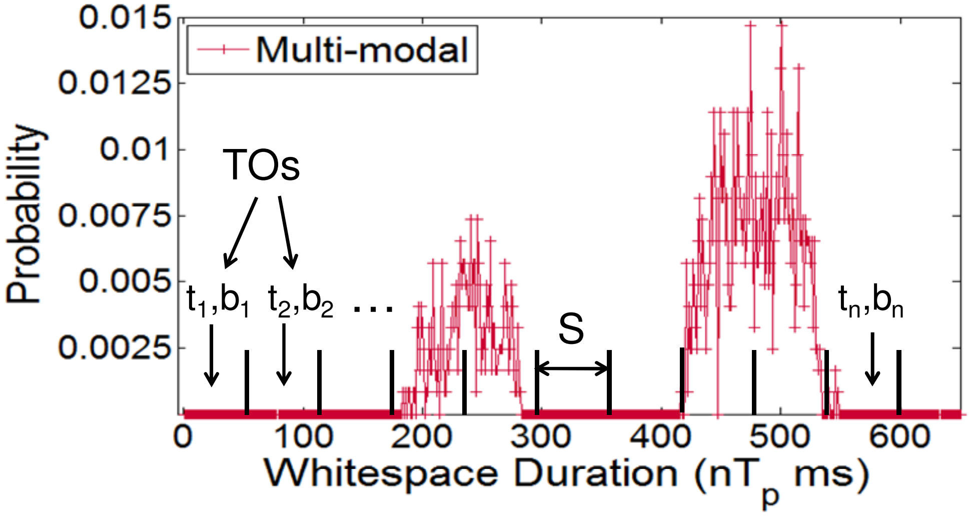


Figure 7-1: DSTS transmission opportunities

Consider the sample whitespace profile shown in Figure 7-1. Observe that the probability that a whitespace ends between [0-200] ms or [300-400] ms is zero. This means that an SU can access these whitespaces without disrupting the PUs. However, utilizing whitespace durations within the ranges of [200-300] *ms* or [400-550] *ms* may cause disruption to the PUs. This is because there is a non-zero probability that a whitespace ends between these ranges. The key idea behind the DSTS strategy for an SU is to determine the best useable TOs based on the recently observed whitespace statistics.

7.3 Transmission Bitmap Vector

In DSTS, a whitespace is divided into *S* duration TOs, where *S* is the duration of an SU packet plus SU control packets. As shown in Figure 7-1, the TOs are referred to as t_1, t_2, …,t_m, where the quantity t_mS represents the largest whitespace observed by the SU. Let the SU packet transmission decision in those TOs be represented by the bits b_1, b_2, ..., b_m, where bit b_0 indicates if the SU should transmit a packet (if available) in TO t_0. These m-bits (1: transmit, and 0: not transmit) form a Transmission Bitmap Vector (TBV). If the bit b_i is set to 1, then the access module in an SU transmits a packet in TO t_i only if a channel-sensing finds the channel to be free during that TO. In other words, a transmission can only occur if the whitespace lasted until the TO t_i. If the channel is found busy (meaning the whitespace has not lasted until the TO t_i) no transmission is made even though the b_i was set to be 1. In addition, if bit b_i was set to be 0 in the TBV, no transmission is made during the TO t_i. Once a whitespace has ended, the procedure resets for the next whitespace. Note that each bit in the TBV is calculated using the *w(n)* resulting from the RSSI measurements similar to those shown in Chapters 3, 6, and 8.

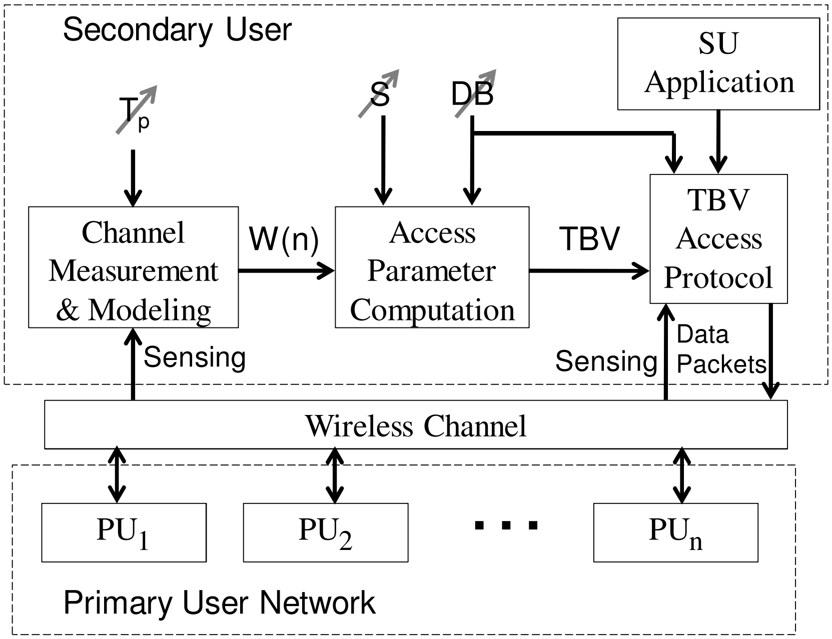


Figure 7-2: Functional diagram of the SU access strategy

7.4 Functional Overview

Figure 7-2 shows a functional diagram of the architectural components in DSTS. The user tunable control inputs into the system are secondary user packet size (*S*), primary disruption bound (*DB*), and the sensing interval (T_p), which is a property of the secondary user hardware. The SU continuously computes the whitespace pdf *w(n)* with a channel-sensing interval of T_p. The access parameters are then fed into the access module, which, in addition to channel sensing, transmits SU user packets into the channel. The objective is to satisfy a user specified primary disruption bound *DB*.

7.5 PU Disruptions and SU Throughput

A method must now be defined to properly select the best TOs that the SU should utilize. Ideally, the SU would like to use every TO to maximize its throughput but disruptions can occur since the duration of a whitespace is unknown. Therefore, each transmission opportunity has a certain gain and loss if it is chosen for use. A gain is represented by throughput for the SU and a loss is represented by a possibility of disruption to the PU. In this section, the throughput and disruptions are formulated for each TO.

7.5.1 Probability of TO Existence

An SU packet transmission during a TO t_i will be made only if the whitespace lasts until that TO. The *probability of the existence* of TO t_i can be expressed as the probability that the whitespace will last at least for *iS* duration, and can be written as:

$$P_i^{existence} = 1 - W\left(\frac{iS}{T_p}\right) \tag{7-1}$$

where *W(n)* is the cumulative distribution function (cdf) of the whitespace pdf *w(n)*, which is expressed as a multiple of the channel sensing interval T_p. When the bit b_i is set to 1 in the TBV, the probability $P_i^{existence}$ will determine if an SU transmission in TO t_i will actually be possible or not. The $P_i^{existence}$ value for a given TO directly represent the throughput an SU will achieve if it sends in that TO. It follows that a TO with high existence probability will occur with greater frequency, thus resulting in higher SU throughput.

7.5.2 Probability of Disruption

An SU packet transmission during a TO t_i will cause a PU traffic disruption if the current whitespace does not last for a full *S* duration since the start of the transmission. The *probability of disruption* is expressed as the probability that a whitespace lasts anywhere between *(i-1)S* to *iS* duration, and can be written as:

$$P_i^{disrupt} = W\left(\frac{iS}{T_p}\right) - W\left(\frac{(i-1)S}{T_p}\right) \tag{7-2}$$

Now consider a situation in which a specific TBV, $\hat{T}$ is selected. Let δ represent a subset (of all *m* bits in $\hat{T}$) of bits that are 1. The rest of the bits are set to 0. During a whitespace, after an SU transmits based on $\hat{T}$, the resulting Primary Traffic Disruption (PTD) can be expressed as:

$$PTD = \sum_{\forall i \in \delta} P_i^{disrupt} \tag{7-3}$$

PTD represents the overall probability that a Primary Traffic Disruption will occur at the end of a whitespace. If each whitespace corresponds to a PU packet transmission (i.e. when no back-to-back PU packets are transmitted), PTD represents the probability that a PU packet will be disrupted due to an SU packet transmission. With a specific subset δ , the Effective Secondary user Throughput (EST) can be expressed as:

$$EST = \sum_{\forall i \in \delta} P_i^{existence} \tag{7-4}$$

7.6 Computing TBV via Solving Knapsack Problem

For a given whitespace pdf *w(n)*, the problem is to compute a Transmission Bitmap Vector $\hat{T}$ such that the EST is maximized while limiting the PTD to a pre-defined bound *DB*. We propose to solve this problem using the 0-1 knapsack problem solution, which is summarized as follows. In the knapsack problem, there are *m* objects (*j=1,2,...,m*) and the j^{th} object has a weight w_j and volume v_j. The goal is to select a subset of the objects in a knapsack so that the total weight is maximized while the volume of the knapsack does not exceed a predefined bound.

The TBV computation is posed as the knapsack problem by mapping the TO t_i to *m* objects. The TO existence probability $P_j^{existence}$ from Equation 8-1 and the disruption probability $P_j^{disrupt}$ from Equation 8-2 are mapped to w_j and v_j respectively. In addition, the PU disruption bound *DB* is mapped as the knapsack volume bound, *V*. The mapping and solution process is formalized as follows.

The problem can be defined as:

$$maximize W = \sum_{j=1}^{m} w_j x_j \tag{7-5}$$

subject to:

$$\sum_{j=1}^{m} v_j x_j \leq V, \quad x_j \in \{0,1\} \tag{7-6}$$

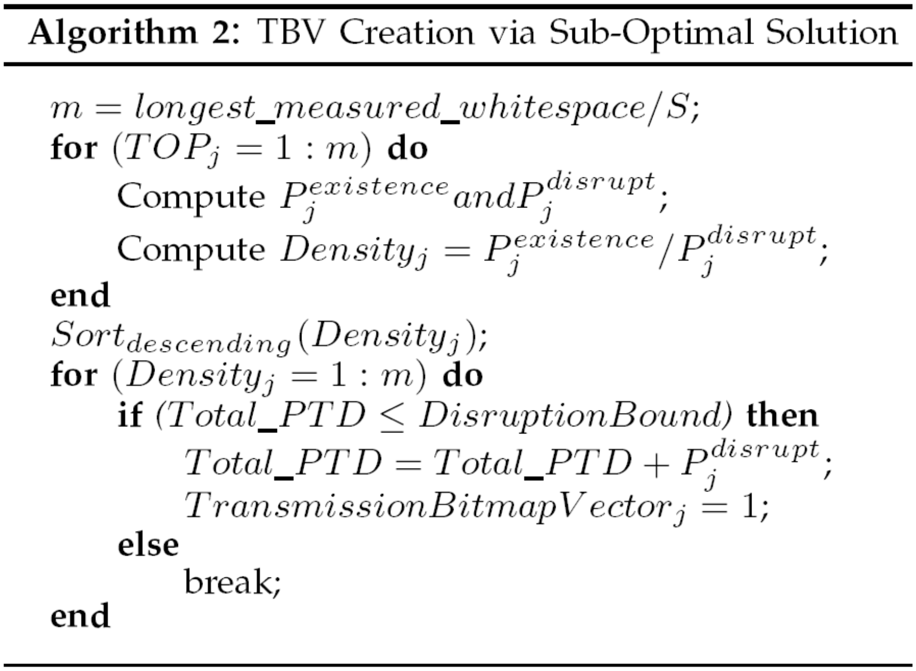
To solve this problem, we can use dynamic programming [79] by defining $s[j,v]$ to be the solution for objects 1,2,…*j* with maximum volume *v*. Then define $s[j,v]$ recursively as:

$$s[j,v] = \begin{cases} 0, & if\ j\ or\ v = 0 \\ s[j-1,v], & if\ v < v_j \\ \max\{w_j + s[j-1, v-v_j], s[j-1,v]\}, & if\ v_j \leq v \end{cases} \tag{7-7}$$

The optimal solution can then be found by calculating $s[j,V]$. Under the assumption that all whitespace can be detected, the $P_j^{existence}$ and $P_j^{disrupt}$ values quantify the SUs' throughput and PU disruption respectively for a given TO t_j. Hence, mapping the knapsack solution to TBV can give an optimal secondary user transmission strategy.

7.7 Sub-optimal TBV Computation

Although providing optimal solutions, dynamic programming can be computationally intensive and may not be feasible for resource-constrained SU hardware. We provide a computationally feasible heuristic solution as shown in Figure 7-3.



Algorithm 2: TBV Creation via Sub-Optimal Solution

```
$m = longest\_measured\_whitespace/S$;
for ($TOP_j = 1 : m$) do
    Compute $P_j^{existence} and P_j^{disrupt}$;
    Compute $Density_j = P_j^{existence}/P_j^{disrupt}$;
end
$Sort_{descending}(Density_j)$;
for ($Density_j = 1 : m$) do
    if ($Total\_PTD \leq DisruptionBound$) then
        $Total\_PTD = Total\_PTD + P_j^{disrupt}$;
        $TransmissionBitmapVector_j = 1$;
    else
        break;
end
```

Figure 7-3: TBV creation via sub-optimal solution

Initially, we compute the Density ($P_j^{existence} / P_j^{disrupt}$) of all *m* objects (i.e. TO) and then sort them in descending order of their density. Afterwards, we assign a 1 in the Transmission Bitmap Vector $\hat{T}$ for the highest density TOs, representing good transmission opportunities. We stop assigning 1s when the Disruption Bound is reached, ensuring that the probability of disruption will meet the pre-defined bound. This scheme provides a relatively fast and simple heuristic solution to the problem. After this algorithm is completed, the $\hat{T}$ for the MS-DSTS access strategy is created. At any given time in a whitespace, if an SU has a packet to send, it will send the packet in the next TO indicated in $\hat{T}$, only if the whitespace lasts until that TO. An SU following this strategy will maintain the specified *DB* while maximizing its throughput.

7.8 TBV Solution Example

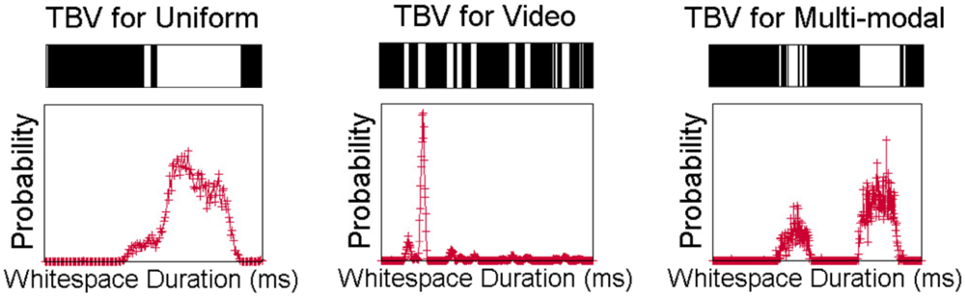


Figure 7-4: TBV of DSTS for different traffic profiles

Figure 7-4 presents an example Transmission Bitmap Vectors (TBV) solution computed by DSTS for three experiments shown in Section 6.3 corresponding to uniform, video stream and multi-modal traffic profiles. The black and while vertical bars in the diagram correspond to the ones and zeroes in the TBV. Observe how the ones in TBV, representing useable transmission opportunities for the SUs, occupy different part of the whitespace spectrum for different traffic profiles.

7.9 Performance Evaluation of the DSTS Protocol

In the following sections, the performance of the DSTS strategy is evaluated experimentally and theoretically in terms of the Primary Traffic Disruption (PTD) and the Effective Secondary Throughput (EST). The whitespace profiles characterized in Sections 3.4.2, and 6.3, i.e. Ad hoc 802.11 and test bed measured CSMA respectively are used to evaluate DSTS.

7.9.1 Comparison Protocols

A comparison of DSTS between multiple protocols is provided. The compared protocols are a sense-and-transmit based protocol, called VX scheme [56], a threshold based access strategy (TBAS) [14], the CSTS protocol presented in Chapter 4 and lastly a benchmark scheme (Benchmark).

7.9.1.1 VX Scheme

The VX (Virtual-transmit-if-Busy) Scheme has a similar strategy as CSMA where the SU will sense the channel and transmit when idle [56]. Initially the SU senses the channel, when the channel becomes idle a packet of length L_S is transmitted. Then, the SU starts a vacation from the channel of length V_S. After the vacation, if the channel is busy, the SU starts a virtual transmission stage and then enters into another vacation stage afterwards. Here, a virtual transmission means that the SU does not actually transmit the packet but waits for a time interval which is equal to the packet length L_S. After the vacation, the SU starts the channel sensing again. In contrast to the CSTS scheme, the VX scheme does not use statistical data but relies on short term sensing for its transmission decisions. Algorithmic analysis for this scheme has been presented in [56].

7.9.1.2 TBAS Protocol

The TBAS scheme determines an optimal transmission strategy for the SUs using an analytical model [14]. In this protocol, first a time related decision metric *g(t)* is introduced. A greater value of *g(t)* represents a smaller probability of disruption. Next, based on a given disruption bound and *g(t),* an optimal threshold γ^* is calculated by evaluating all possible combinations of *g(t)* and *γ*. Finally, using the metric *g(t)* and threshold γ^*, a policy *q(t)* is proposed to determine if the SU should transmit at a given time *t* within a whitespace. Under the assumptions of continuous time and very small SU packet size, it is shown that the policy is optimal. Additionally, for more practical scenarios with finite packet size, the authors propose a heuristic solution. First, the minimum *g(t)* value is found within the length of time of a SU packet duration. Then using the minimum *g(t)* and the corresponding threshold *γ,* a sub-optimal *q(t)* is computed.

7.9.1.3 CSTS Access Strategy

Contiguous SU Transmit Strategy (CSTS) protocol operates based on the channel usage history, and attempts to bound the amount of PU disruption and maximize SU throughput. In STL, there are two distinct stages. In the first stage, an SU initially waits for the channel to become idle. Once the channel becomes idle, the SU attempts to transmit its first packet of duration *S*. If the SU has more packets to send, it continues sending until the measured value of ϵ (time elapsed since the beginning of the current whitespace) becomes so large that the expected PTD for new transmissions become equal to or larger than a pre-determined Disruption Bound (*DB*). A term J_{max} is defined, which denotes the maximum number of packets that the SU can send before the expected disruption reaches the pre-determined *DB*. The quantity J_{max} is computed by the SU based on the observed whitespace statistics. After J_{max} packets are sent, the low priority node abandons transmission attempts and starts channel sensing for detecting the next whitespace.

If, however, a whitespace ends before all J_{max} packets are sent by the SU, it exits the channel after detecting that a PU has started transmitting. Note that in spite of the SU vacating the channel, disruption to the PUs' traffic can still occur since the PU must defer its transmission until the channel is vacated by the SU. This stage essentially specifies a block of time in every whitespace that the SU is allowed to transmit. This block is bounded by the start of a whitespace to the ending of the $J_{max}{}^{th}$ packet.

The second stage of STL corresponds to the situation in which a whitespace duration is greater than the end time of the $J_{max}{}^{\text{th}}$ packet from the SU. No PTD will be caused in this stage because the SU will exit the whitespace after J_{max} transmissions.

7.9.1.4 Benchmark Protocol

The Benchmark protocol assumes full knowledge of the starting and ending times of every whitespace. Therefore, it t0ransmits SU packets in every available whitespace and stops when the next SU transmission will cause disruption to the PUs. The Benchmark protocol can be run only offline to find the best SU performance. This protocol essentially determines the highest throughput an SU can expect without causing any disruption to the PUs.

7.9.2 Ad hoc 802.11 Whitespace

7.9.2.1 802.11 Whitespace profile

In order to represent different 802.11 based Ad-hoc network topologies and traffic characteristics, three scenarios are established (see Figure 7-5). (TOP1) is a linear chain topology with a single multi-hop primary flow representing a simple PU topology. (TOP2) is two multi-hop flows in a network on two intersecting linear chains, which highlights the effects on whitespace of competing primary flows. (TOP3) is two multi-hop flows in a parallel chain network, which emphasize the effects on whitespace of disjoint primary flows. In each case, the secondary users are within the CS range of all primary users.

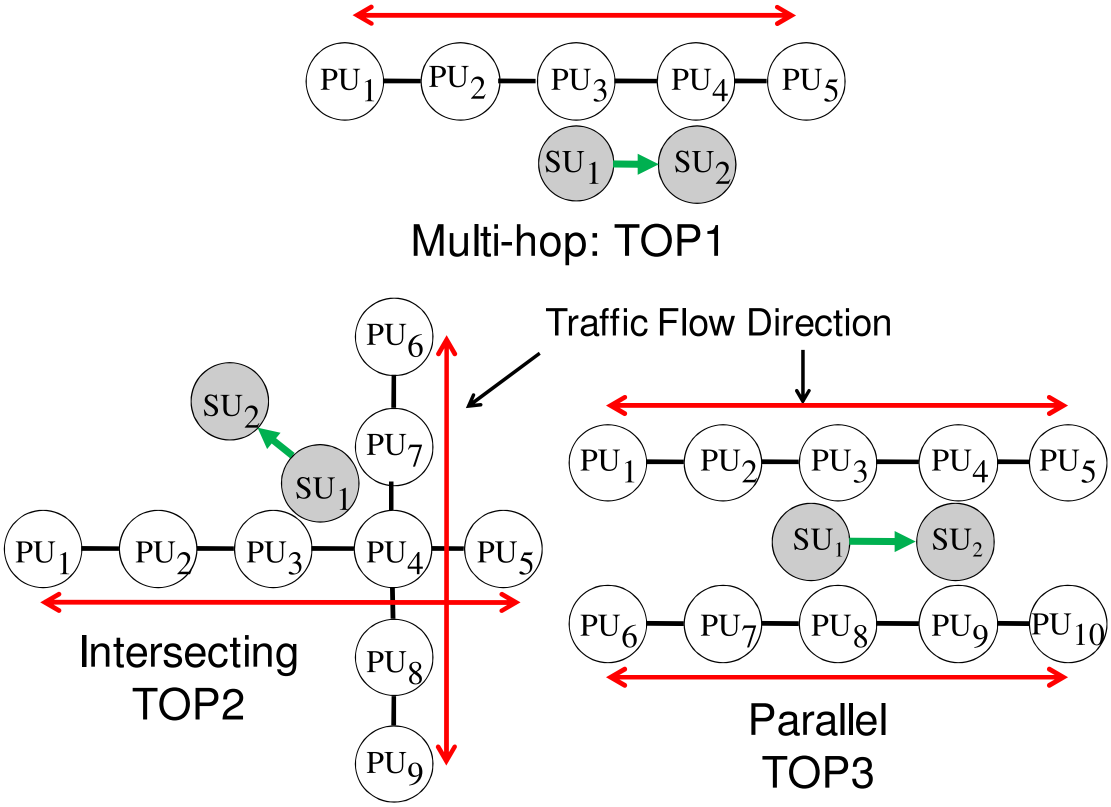


Figure 7-5: Primary user topologies and the traffic flows

Figure 7-6 shows the *w(n)* for UDP traffic with 90 ms average inter-packet duration that is uniformly distributed across ±20% around the average packet arrival interval for each topology. The distributions presented in Figure 7-6 represent the general whitespace properties for a wide range of topology and traffic variations of the Ad hoc 802.11 whitespace.

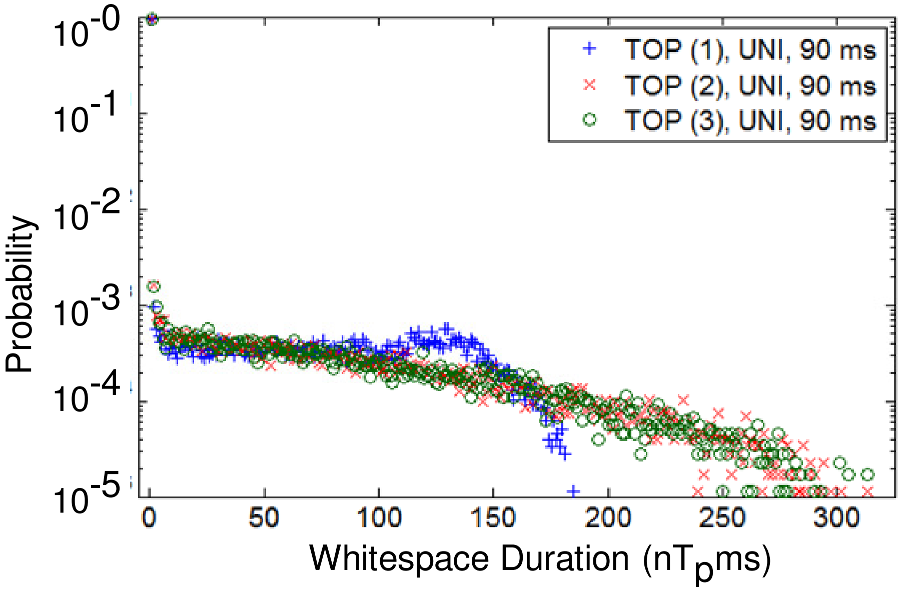


Figure 7-6: Impacts of PU topology with Uniform traffic on whitespace distributions.

7.9.2.2 Impacts of Primary User Load

Figure 7-7 demonstrates the impacts of varying PU load on the SU throughput and PU disruption for TOP2 in Figure 7-5. The PU inter-packet arrival duration is varied from 50 ms to 120 ms for simulating different loading conditions. For all the primary packet rates, a ±20% uniformly distributed inter-packet arrival variation was introduced. The SU packet duration was fixed to 1.2 ms, which is half the PU packet size. For the presented results, the SU packets are generated back-to-back, meaning packets are always available to send within an SU's buffer. The user-defined disruption bound (*DB*) is set to 0.05 [6], which indicates that only 5% of the PUs' traffic is allowed to experience disruptions from the SUs' transmissions. The values of μ, J_{max}, and TBV are calculated dynamically depending on the current PU traffic statistics. The sensing interval T_p is set to 5 μs. For the VX scheme, V_s, the channel wait time, is set to 0.5 ms and the *Virtual Xmit* duration is set to 16 ms, which was shown (via simulation) to deliver the best performance for that access protocol.

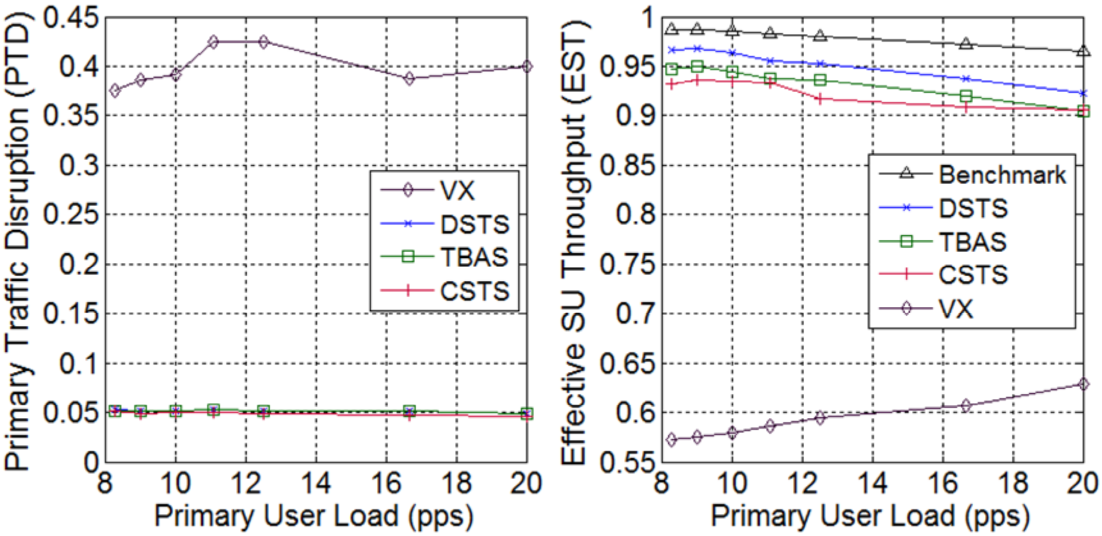


Figure 7-7: Performance for uniformly distributed PU traffic

From Figure 7-7, observe that both CSTS and DSTS access protocols are able to maintain the required primary disruptions within the specified Disruption Bound (*DB*) across a wide range of PU data rates. As the PU data rate increases, the duration of usable whitespace in the channel reduces. Also, the number of distinct whitespaces increases with the PU data rate, since the whitespaces become more fragmented. Since the SU access strategies are executed on a per-whitespace basis, with the increased number of distinct whitespaces, the SU attempts to execute its PU protection mechanisms (e.g. TBV in DSTS) more often. This causes the SUs' channel access efficiency or EST to decrease due to increased PU load.

In Figure 7-7, it should be also observed that the EST for DSTS, CSTS and TBAS access strategies remain close to that of the offline Benchmark protocol, while the target *DB* is maintained. Our simulation experiments comparing the performance gap between the optimal and sub-optimal DSTS solutions showed minimal difference. Additionally, the EST results for sub-optimal DSTS, as shown in Figure 7-7, are only within approximately 3% of the Benchmark performance. Since the Benchmark represents the best possible EST, this 3% difference in fact represents the upper-bound of the performance gap between the optimal and sub-optimal DSTS solutions.

Note that DSTS achieves slightly higher EST over the TBAS. This difference arises in the method used to accommodate non-zero SU packet sizes in DSTS and TBAS. DSTS uses the SU packet size as an input into the knapsack problem while TBAS uses the minimum *g(t)* value over the range of the packet size.

For the VX scheme, disruption to the PU traffic can be observed to be higher. Utilizing the *Virtual Xmit* wait times, the SU avoids sending during busy times in a channel. However, once the SU starts accessing a whitespace, it stops sending only if a PU returns during a V_S wait time. Thus, there remains a high probability of causing disruption during the SU transmissions. The VX scheme has a relatively lower throughput since it waits for a V_S time after every transmission, which is not the case of DSTS, CSTS and TBAS mechanisms. These results suggest that by utilizing the history of the channel, it is indeed possible to improve the SU throughput while maintaining a bound of disruption on the primary networks.

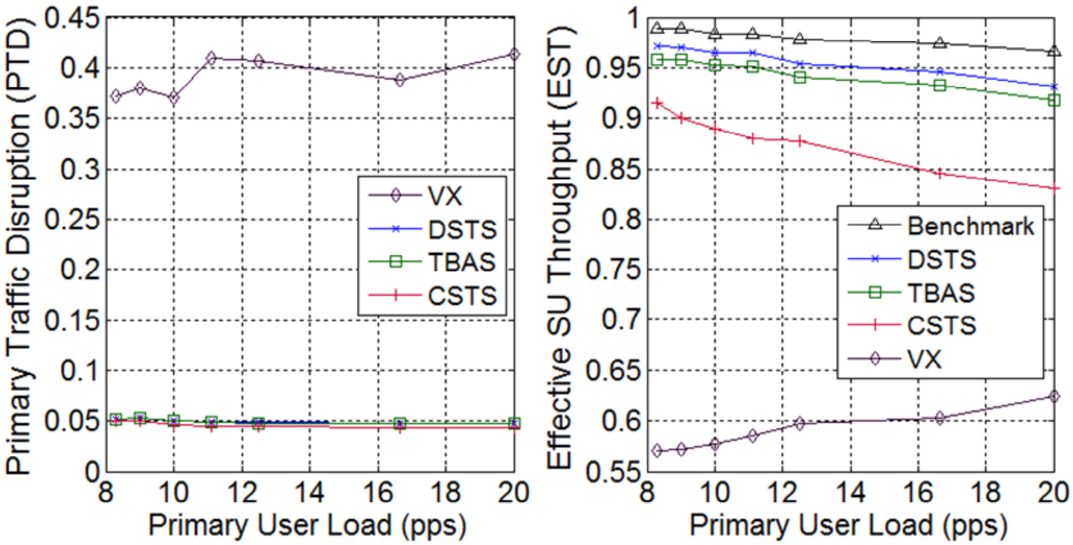


Figure 7-8: Performance for Poisson distributed PU traffic

Figure 7-8 shows results for the same setup, but with Poisson distributed PU traffic. Note that the performance trends for this scenario are very similar to those for the Uniform case as shown in Figure 7-7. Similar results were also observed for video stream PU traffic for which the pdf was shown in Figure 7-6e. We have also simulated TCP primary traffic. Since TCP opportunistically utilizes available bandwidth in the channel, the amount of resulting whitespace for TCP traffic is small. As a result, the achievable SU throughput remains severely restricted when the primary disruption bounds are set at low values [58].

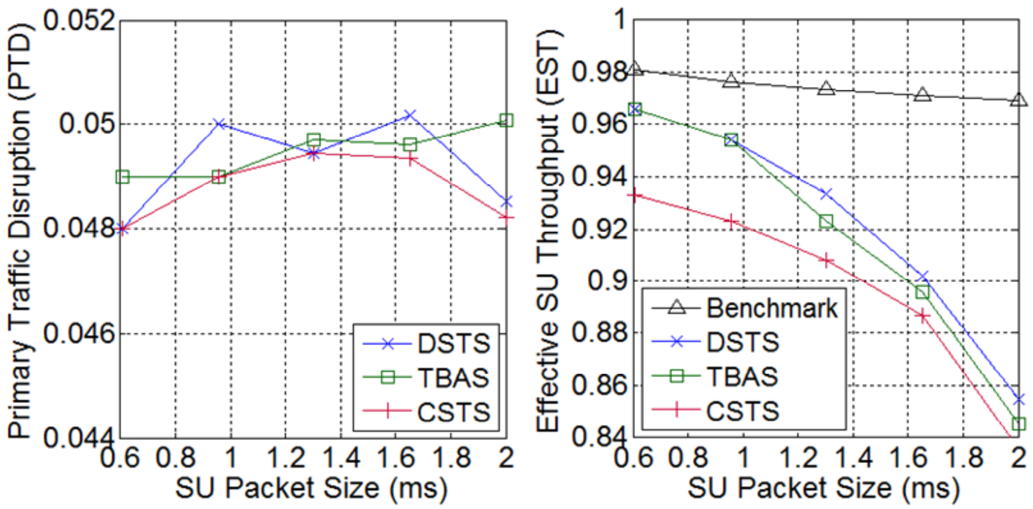


Figure 7-9: Impacts of the SU packet size on EST and PTD

Figure 7-9 shows the impacts of varying the SU packet size from 625 μs to 2 ms, as done in [14]. The PU rate is set to 10 pps with Uniformly distributed Traffic. Observe that the throughput decreases as the SU packet size increases. In order to maintain the *DB* with larger packet sizes, the access strategies must reduce their transmission probabilities since a larger secondary packet has a higher probability of causing disruption to the primary at the end of a whitespace. Additionally note that with increasing packet size, DSTS is able to achieve better performance than the Threshold based scheme TBAS. This is because unlike TBAS, which works with the assumption of very small SU packet size, DSTS takes the actual SU packet length into consideration while setting its parameters.

7.9.2.3 Performance under Dynamic PU Traffic Patterns

Figure 7-10 reports the protocol performance from a simulation experiment in which the PU traffic pattern was dynamically changed over time for TOP3. Initially, the PU traffic is generated with 60 ms inter-packet time (IPT) with the Uniform profile. The pattern changes after 60 seconds when the IPT is increased to 120 ms. Finally, at 100 seconds the IPT is changed to

Poisson distribution with 90 ms average. The whitespace *w(n)* is continually computed by the access protocol after collecting every 1000 whitespace samples.

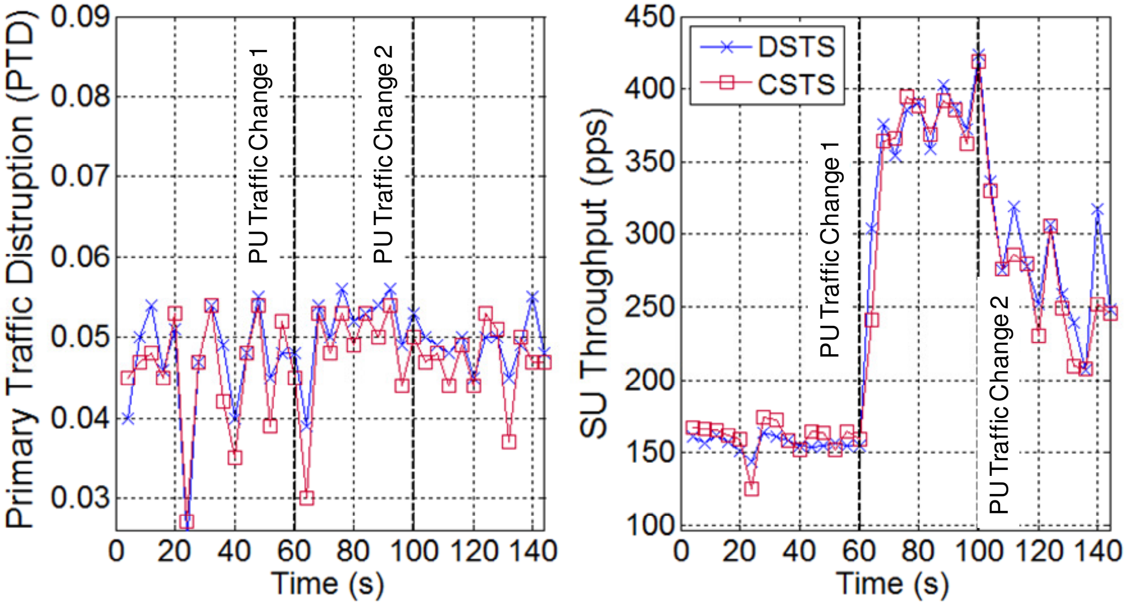


Figure 7-10: Performance with time-varying *w(n)* for 802.11

Observe in Figure 7-10 that while the primary user traffic pattern changes over time, the PTDs resulting from the CSTS and DSTS access strategies stay consistently within the vicinity of the pre-set disruption bound of 0.05. Also, note that there is very little PU traffic disruption during the transitions of *w(n)* as indicated in both the frames in Figure 7-10. This is primarily because the whitespace pdf for ad hoc mode 802.11 traffic maintains very similar properties across different traffic patterns, i.e. above 90% of the whitespaces are below 1 ms (see Figure 7-6).

Note that the SU throughput increases after a PU traffic pattern transition when the useable whitespace increases due to the increase in inter-packet interval of the PU traffic. Additionally, the SU dynamically throttles back its transmissions to protect the PUs when the available whitespace reduces during a transition. These results demonstrate the robustness of the CSTS and DSTS access strategies against varying data rates over time. Additionally, the results suggest that by periodically measuring the channel state the SUs are able to dynamically model the channel whitespace and adjust their access parameters accordingly to minimize the PU disruptions and to maximize the SU throughput.

7.9.3 CSMA Profile Using Test Bed Measured Whitespace

In this section, we evaluate the DSTS strategies using the extracted whitespace pdf *w(n)* from a TelosB based prototype test-bed for a single SU.

7.9.3.1 CSMA Whitespace Profile

In order to represent PU traffic diversity, three traffic scenarios are established i.e. sent over the PU network:

- A uniformly distributed traffic that is sent at 7 packets per second (pps). Each packet has an added ±20% inter-packet time to vary the arrival time on the channel. This profile corresponds to TelosB based sensor network applications involving constant bit rate sensing.
- A low rate video file streamed over the channel, which represents video surveillance applications.
- A multi-modal traffic with two "peaks" of whitespaces in the pdf *w(n)*. In general, a stream with multiple peaks in its whitespace can be formed by merging multiple constant packet rate streams. This is representative of a multi-application primary traffic at different traffic generation rates.

Figure 7-11a-c shows the whitespace pdfs *w(n)* measured and computed by an SU Transmitter, which is within the transmission range of the PU Transmitter that is generating the traffic profiles as described above.

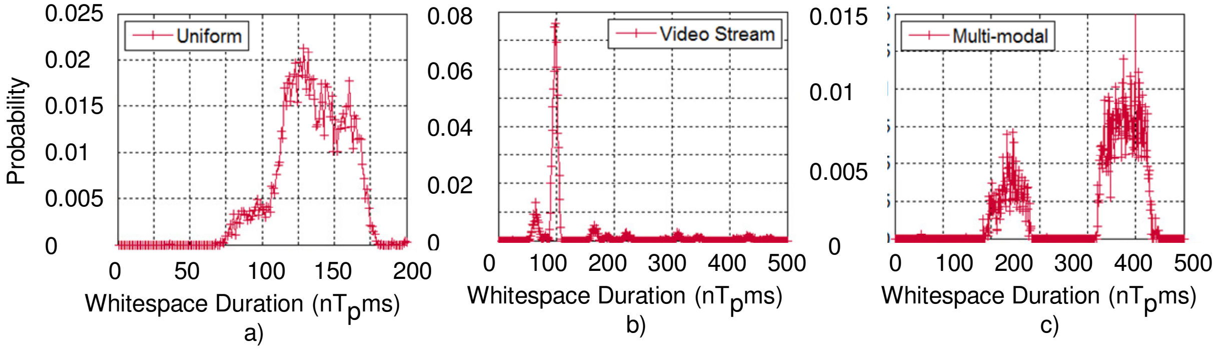


Figure 7-11: Experimental whitespace distributions. (a) Uniform with 20% variation, (b) Low rate video stream, and (c) Multi- modal data stream

7.9.3.2 Uniformly Distributed Primary Traffic

Figure 7-12 shows the PTD and EST with varying SU traffic load when the PU traffic profile is set to be uniformly distributed. The corresponding whitespace pattern observed by the SUs is shown in Figure 7-11a. In Figure 7-12, observe that both CSTS and DSTS are able to maintain the required disruption bounds across all the experimented SU rates. In terms of throughput, the access strategies are able to support low SU generation rates until the primary user disruption bound is reached. However, once the specified *DB* is reached, for both CSTS and DSTS the SU transmissions are throttled back in order to protect the PU traffic.

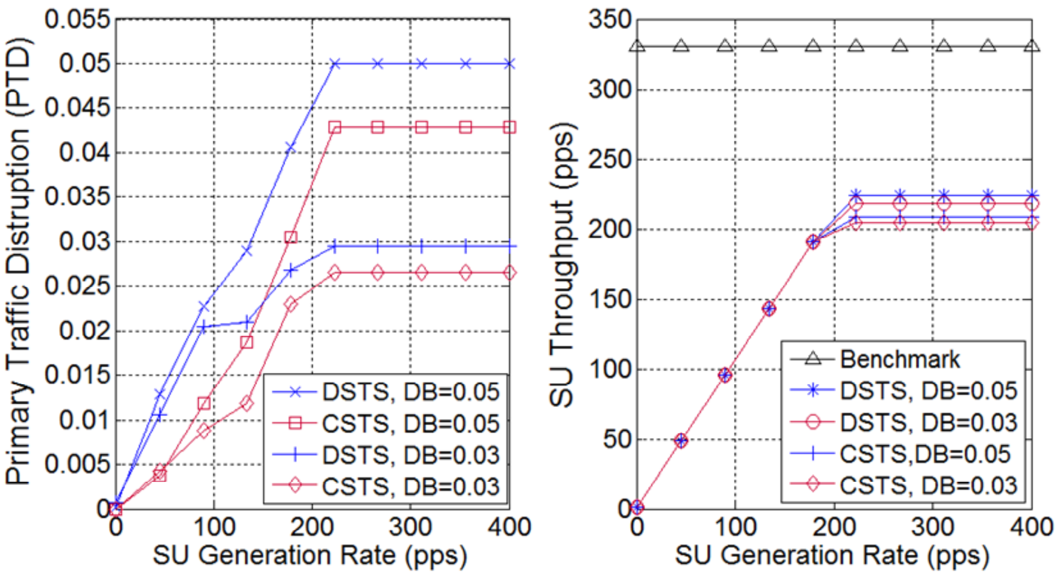


Figure 7-12: Performance with varying SU traffic load under Uniform traffic

Notice that in Figure 7-12 both CSTS and DSTS have very similar maximum SU throughput values. This performance similarity is a direct result of the uniform PU traffic and the resulting whitespace. Since all the whitespace is within a very small range (70 ms and 180 ms), in order to maintain the required *DB,* both access strategies send packets until approximately the same point in a whitespace. This explains the similar SU throughput numbers while guaranteeing the same *DB*.

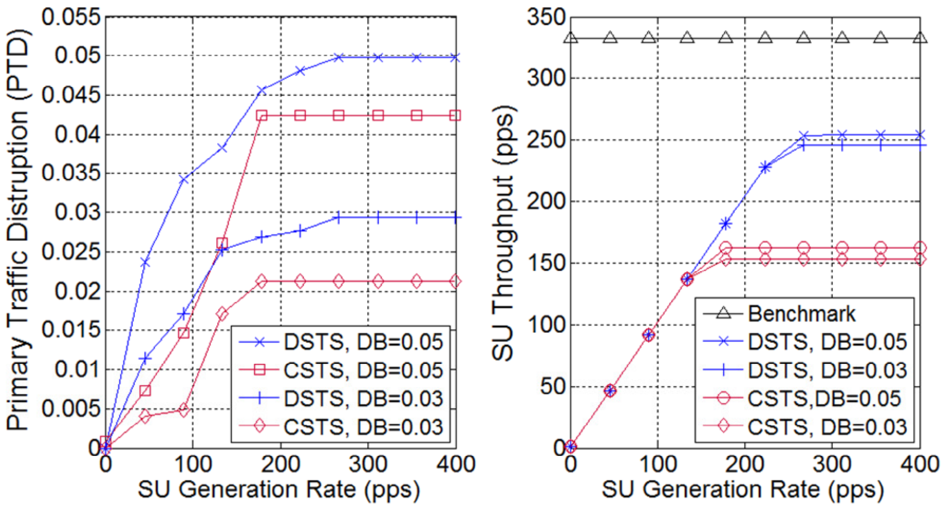


Figure 7-13: Performance with Multi-modal whitespace pdf

7.9.3.3 Multi-modal PU Traffic

Protocol performance for PU traffic with multi-modal whitespace profile (as shown the pdf in Fig. Figure 7-11c) is shown in Figure 7-13. The relative performance of CSTS, DSTS, and Benchmark in this case is very similar to those observed for the uniform traffic case in Figure 7-12. The primary difference here is the ability of the protocol DSTS to support higher sustainable SU throughput compared to that for CSTS. As in the uniform traffic case, both CSTS and DSTS are able to maintain the specified *DB* at the expense of limiting the SU throughput at higher SU load levels. However, observe that DSTS is able to support 56.0% and 60.3% higher throughput compared to CSTS when the DB is specified at 5% and 3% respectively. This large increase in throughput over CSTS results from DSTS's ability in identifying the best transmission opportunities for sending an SU packet with minimal disruption.

Since the first peak of whitespace pdf around 250 ms (in Fig. Figure 7-11c) contains 25% of all whitespaces, the J_{max} value in CSTS is expected to be set near the start of the first peak. This makes CSTS to miss transmission opportunities corresponding to the second peak in the whitespace pdf, which the mechanism DSTS is able to access. Also note that in both DSTS and CSTS, the throughputs corresponding to *DB = 3%* are slightly lower than that in the *DB = 5%* case. This occurs because in order to maintain a lower DB, a lower number of SU packets are sent. As shown in Figure 7-14, results for the video stream case showed similar results to this multi-modal case.

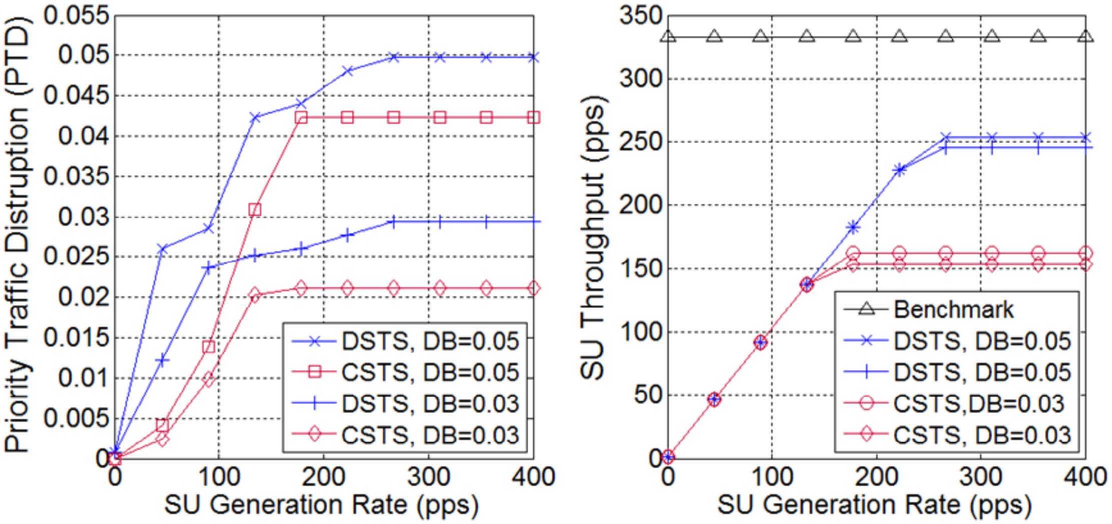


Figure 7-14: Performance with Video Stream whitespace pdf

7.9.3.4 Dynamic PU Traffic Profile

The performance of CSTS and DSTS under a time-varying whitespace profile is shown in Figure 7-15. The disruption bound *DB* is set to 0.05. The PU traffic pattern was changed using the three pdf types in Figure 7-11. Initially, the PU traffic is generated with a uniformly distributed rate of 7 pps. The pattern changes after 60 seconds when the traffic profile is changed to multi-modal. Finally, at 130 seconds, the PU traffic is changed to a video stream.

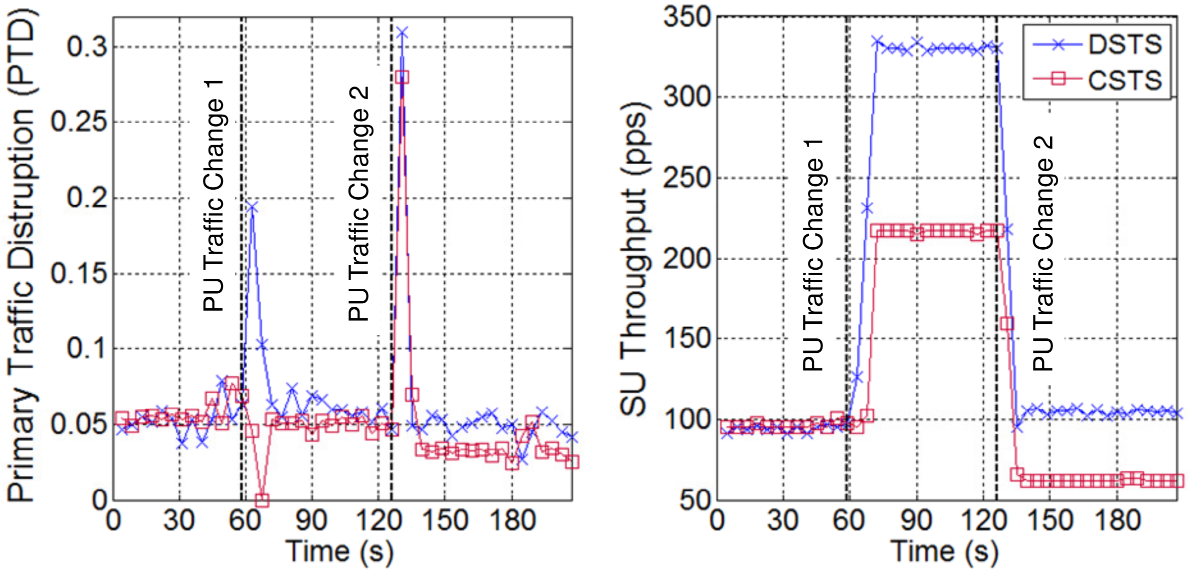


Figure 7-15: Performance with time-varying *w(n)* for CSMA MAC

Observe that at steady state, the PTD remains close to the defined *DB* for both CSTS and DSTS. However, during the *w(n)* transitions, the PTD increases past the disruption bound. In contrast to the dynamic 802.11 based PU traffic case (see Section 7.9.2.3 and Figure 7-10), the changes in *w(n)* during traffic transitions for CSMA traffic is quite significant. This explains the relatively larger (compared to that in Figure 7-10 for the 802.11 case) transient peaks in the PTD during the *w(n)* transitions in Figure 7-15. In terms of throughput, the protocol DSTS is able to achieve higher throughput compared to CSTS when the traffic changes to a video stream and to one that produces multi-modal whitespace.

7.10 Summary and Conclusion

In this chapter, a secondary user access strategy named Divided SU Transmission Strategy (DSTS) is presented that provides a method to access a general PU whitespace profile. The access strategy utilizes multiple transmission opportunities (TO) within a whitespace to find the best times to transmit SU packets. These TOs are consolidated into a Transmission Bitmap Vector (TBV) that defined when SUs should transmit within a whitespace to maximize SU throughput while limiting PU disruptions to a predefined bound. Optimal and Sub-optimal protocols are provided to select the best transmissions opportunities. Evaluation of DSTS under different traffic profiles showed that the protocol outperforms comparison protocols.

Chapter 8: DSTS With Support for Multiple Secondary Users

8.1 Introduction

8.1.1 Motivation

A challenge in prioritized device coexistence is how to efficiently allow multiple SUs to access the channel while minimizing the disruptions to the primary user traffic. The access protocol presented in this chapter aims to provide a solution to that problem.

8.1.2 Problem Definition

An access protocol for multiple secondary users should support efficient SU access while minimizing disruptions to the primary users. The protocol should also support fairness amongst the secondary users. As discussed in Section 7.3, DSTS defines a Transmission Bitmap Vector (TBV) that specifies the best Transmission Opportunities (TOs) within a whitespace. The key idea of the access protocol, termed Transmission Bitmap Vector Access Protocol (TBVAP), is to share a TO among the secondary users. Each SU would independently measure, model, and calculate the TBV. From a functional standpoint, TBVAP operates in the *SU Access Module* in Figure 7-2. Once the identification of the TOs within the whitespaces left by the PUs is done in the Access Parameter Computation Module, TBVAP is used for sending SU data. Utilizing the features of the DSTS access strategy, the proposed TBVAP aims to satisfy these SU access and fairness requirements.

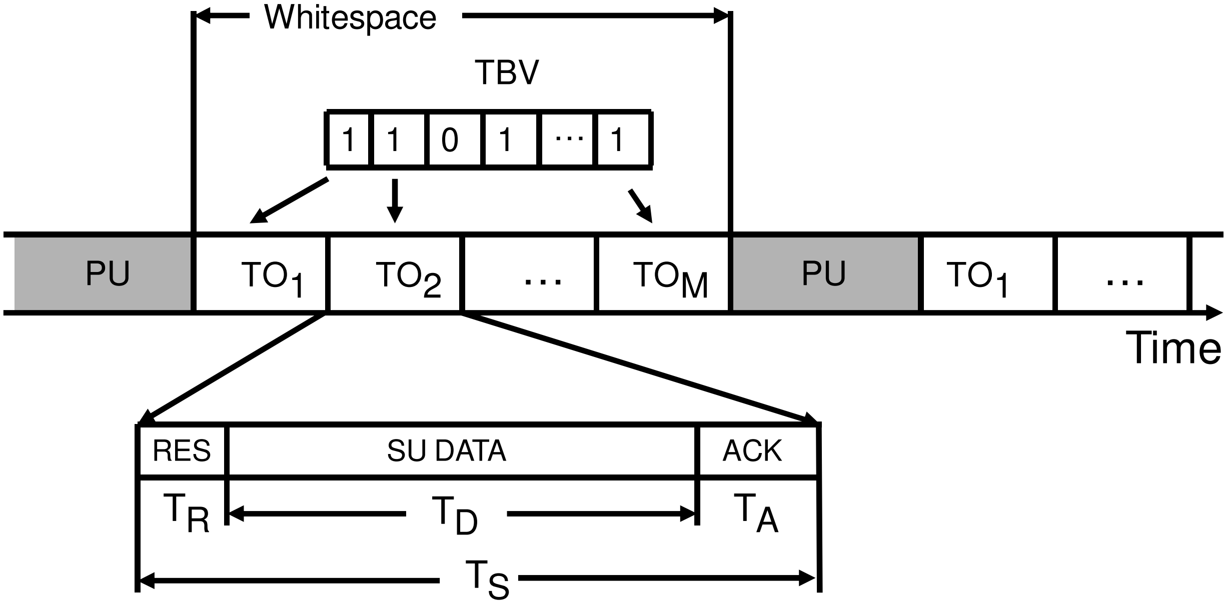
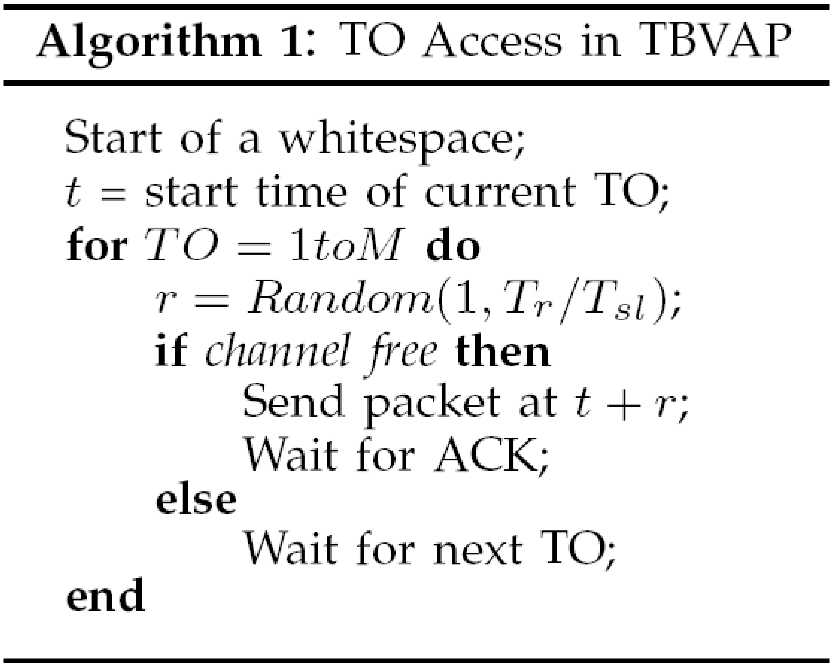


Figure 8-1: TBV with the corresponding TO structure

8.2 TBVAP Structure

An idle duration within the channel (whitespace) can be modeled as a sequence of transmission opportunities for the secondary users. The number of utilized TOs is determined by the duration of the whitespace that the SUs are currently observing. It is clear that the number of TOs varies greatly with the dynamic nature of the PU traffic. In TBV calculation, the number of TOs is set to the length of the longest duration observed whitespace divided by the size of a TO. This ensures that with varying whitespace duration, a portion of the TBV will be utilized. Figure 8-1 shows the structure of the TBVAP. Depending on the length of a whitespace, each whitespace consists of *j* number of TOs. Since the SUs are within the same whitespace domain, the beginning of a whitespace is common to all SUs. Therefore, at the start of a whitespace, each SU dynamically creates *1, 2,…j* TOs of length T_S until the whitespace ends.



Algorithm 1: TO Access in TBVAP

```
Start of a whitespace;
$t$ = start time of current TO;
for $TO = 1toM$ do
    $r = Random(1, T_r/T_{sl})$;
    if channel free then
        Send packet at $t + r$;
        Wait for ACK;
    else
        Wait for next TO;
end
```

Figure 8-2: TO Access in TBVAP

Within a TO, there are three different time periods, namely *Reservation*, *Data*, and *Acknowledgment*. The Reservation period is used to support fairness among the SUs. As shown in Figure 8-2, first, at the start of the period, each SU selects a uniformly distributed number between [1, T_R/T_{SL}] where T_R is the length of the reservation period and T_{SL} is the minimum duration needed to detect a nodes transmission. The node with the lowest value gains access to the TO. This process reduces inter-SU collisions and supports fairness among the SU nodes. Next, the successful node sends a packet within T_D seconds during the Data period. Note that the length of T_R+T_D is fixed and the lengths of T_R and T_D can be adjusted depending on the number of the SUs and the desired size of a SU packet. Lastly, a positive acknowledgment packet is sent to the sender SU during the Acknowledgment period within T_A seconds. Let T_{PD} be the propagation delay, thus the TO length, T_S is given by T_R+T_D+T_A+2T_{PD}. Note that regardless of the number of SUs, the access protocol guarantees the disruption bound on the primary user traffic.

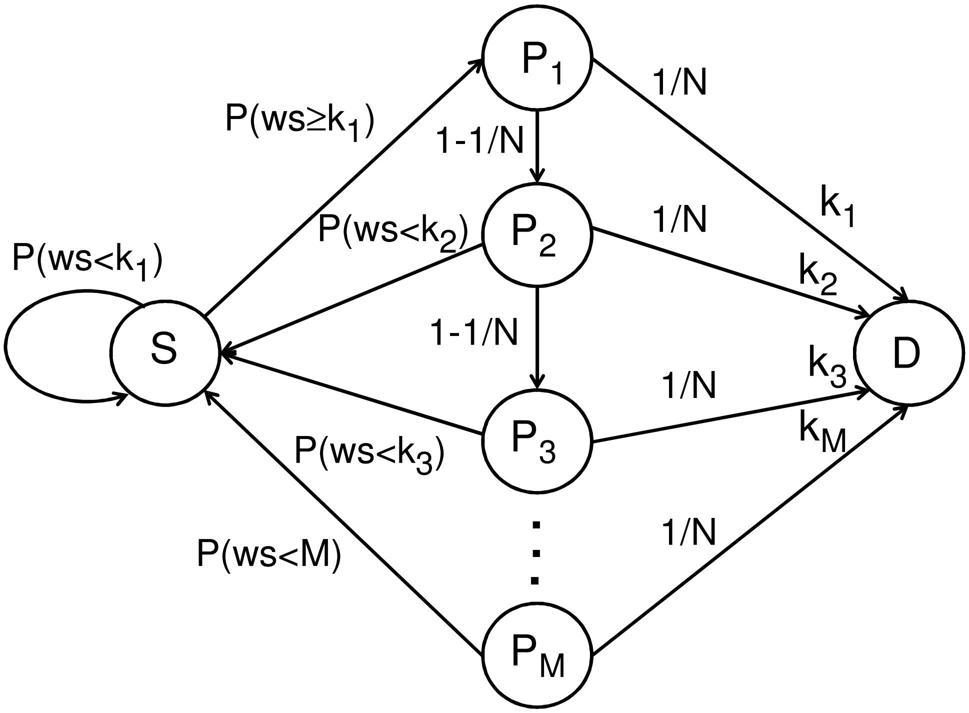


Figure 8-3: Delay analysis for TBVAP

8.3 Delay Analysis

The average delay of the proposed access protocol is computed in this section. As shown in Figure 8-3, we first define a state *S* where a SU will attempt to send a packet in a whitespace. Let k_i be the index of the i^{th} 1 in the TBV (i.e. locations where packet transmissions actually happens). With probability P_1, the whitespace will last for k_1 duration to allow a SU packet to be successfully transmitted. Assuming that there is *N* SUs attempting to access TO_1, there is a *1/N* probability of success for a SU to deliver its packet. A successfully delivered packet, denoted as state *D*, would then have a delay of k_1. If the SU did not gain access to the TO slot, it will attempt to transmit in the next slot with probability P_2. If the whitespace does not last long enough to send in the second TO, the SU will go back to state S. This process continues until the packet is sent.

We denote the expected delay for a SU packet sent in a whitespace as E_{ws}. Thus, we have:

$$E_{ws} = \sum_{k=1}^{M} kx_k P_k \tfrac{1}{N}\left(1-\tfrac{1}{N}\right)^{f(k)} \tag{8-1}$$

where P_k is the $P_k^{existence}$, M is the maximum number of TOs, and *f(k)* is a function that gives the number of 1's in a TBV$_{1\ldots k}$. The term x_k value is 1 when there is a 1 in TBV$_k$ and 0 otherwise.

Note that within a whitespace, at most M transmissions can be accommodated. Therefore, sometimes a SU packet may be deferred until the next whitespace. To calculate this, we define ρ as the probability that a packet can be sent in a whitespace. ρ is given by:

$$\rho = \sum_{k=1}^{M} x_k P_k \tfrac{1}{N}\left(1-\tfrac{1}{N}\right)^{f(k)} \tag{8-2}$$

Thus, the expected number of whitespaces before a secondary user packet can be successfully sent is *1/ρ*. Using Eqns. (9) and (10) we denote the average delay as:

$$Delay = E_{ws} + \frac{L}{\rho} \tag{8-3}$$

where *L* is the average whitespace length.

8.4 Whitespace Analysis for Multiple PUs

When an SU attempts to measure the whitespace in a general network, the number of PUs can affect the whitespace characteristics. In this section, an investigation of the effects of multiple primary users on the whitespace characteristics is presented.

8.4.1 Network Setup

To investigate a general wireless network, we consider a network with SUs existing within a single collision domain with the PUs. Thus, each secondary user's whitespace is equally affected by transmissions of the primary users. Figure 8-4 shows the setup of the PUs and SUs. In this network, each secondary user can hear the transmissions of all of the primary users.

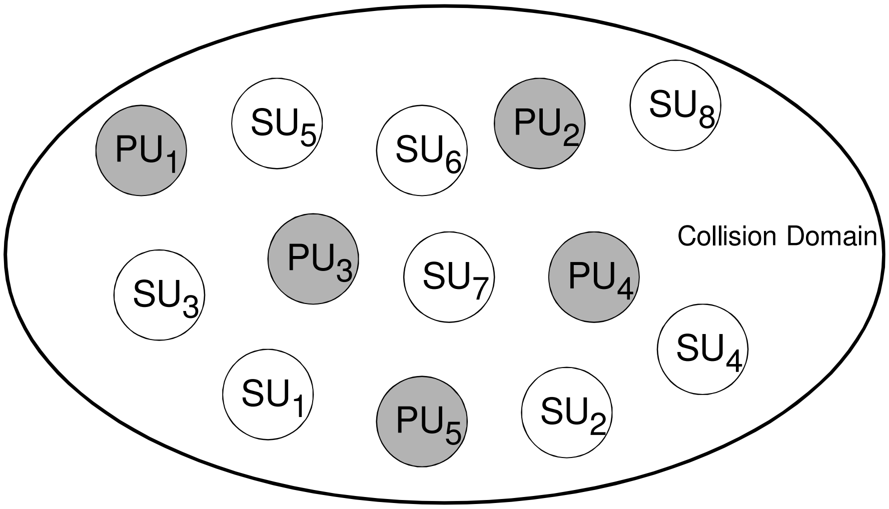


Figure 8-4: Multiple SUs and PUs within the same collision domain

The above network scenario is chosen to gain controlled insight to the interactions between multiple primary traffic flows, and their effects on the resulting whitespace as perceived by the secondary users.

8.4.2 Primary User Traffic Profile

The number of primary users and traffic data rate directly affect the total amount and statistical properties of the available whitespace as viewed by the secondary users. We model two PU traffic profiles, namely, Poisson and Uniform. The Poisson profile is modeled as a Poisson process with mean packet arrival interval of α ms. The Uniform profile consists of uniformly distributed packet arrival intervals with a mean of α ms and a variability of $\pm v$ ms around the mean. Each PU independently transmits according to the specified traffic profile. The packet arrival intervals for each Uniform and Poisson along with the number of PUs flow are changed in order to represent varying intensity of the primary traffic.

8.4.3 Whitespace Analysis

The whitespace pdf $w(n)$, with varying number of PUs in a fully connected network are shown in Figure 8-5 through Figure 8-8. Figure 8-5 shows the $w(n)$ for Poisson traffic with α = 200 ms average inter-packet duration. Observe that with a single PU, the whitespace follows an exponential distribution with an average whitespace duration equal to 197 ms $\approx \alpha$. This is expected, as the PU represents a single Poisson process. When the number of PUs is increased to two, the whitespace duration is exponentially distributed with the average reduced to 98.3 ms $\approx$ $\alpha/2$. Note that the average whitespace durations are slightly reduced by the PU packet time and collisions between the PU traffic. The above trend for distribution of whitespace holds for a higher number of PUs shown in Figure 8-6. In general, with n PUs, the average whitespace duration is $\approx \alpha/n$.

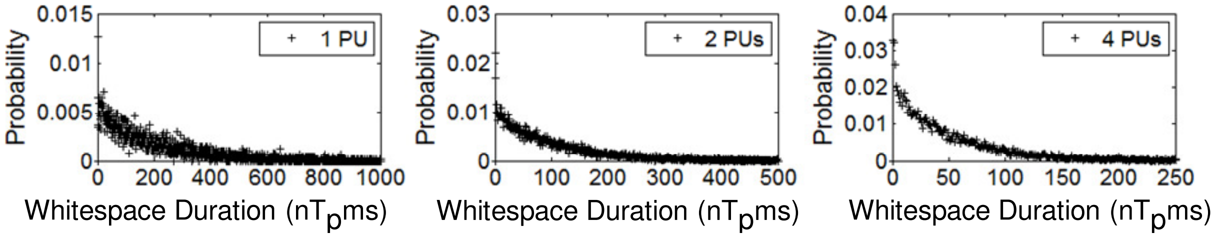


Figure 8-5: Impacts of 1, 2, and 4 Poisson primary users on whitespace distributions.

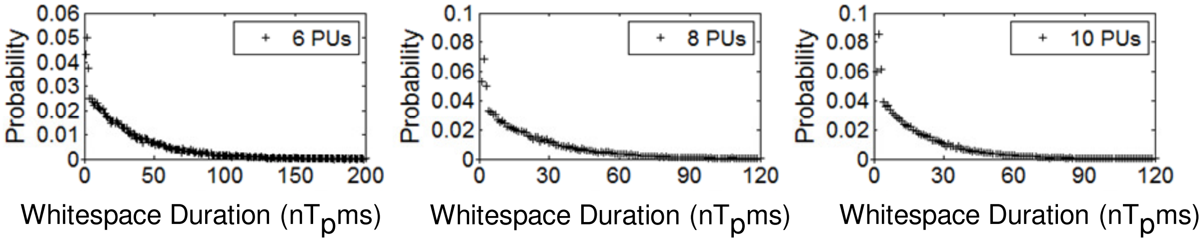


Figure 8-6: Impacts of 6, 8, and 10 Poisson primary users on whitespace distributions.

Figure 8-7 shows the whitespace for primary user Uniform traffic with arrival times uniformly distributed across ±20% around a 200 ms average packet arrival interval for each PU. With a single PU, the whitespace follows a uniform distribution between [160, 240] ms. As the number of PUs increases as shown in Figure 8-8, the shape of *w(n)* approaches that of an exponential distribution similar to the Poisson Case.

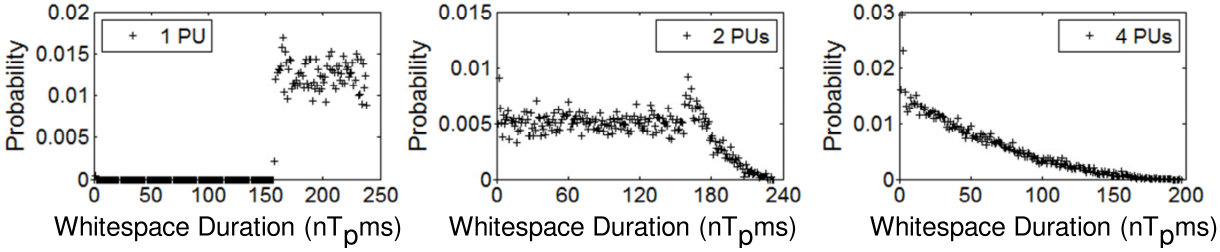


Figure 8-7: Impacts of 1, 2, and 4 Uniform primary users on whitespace distributions.

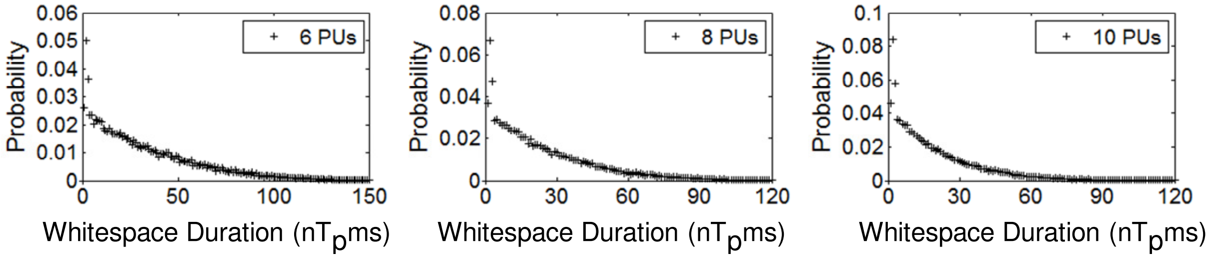


Figure 8-8: Impacts of 6, 8, and 10 Uniform primary users on whitespace distributions.

With the uniformly random traffic generation rate, if we quantize time in terms of transmission slots, at any given slot there is a probability γ that a packet is sent by one of the primary users. With a larger number of primary users in the network, this probability will be higher. Given that a whitespace is a sequence of slots without any packet, the probability that there is a whitespace with length ***l*** slot is equal to:

$$P(ws = l) = (1-\gamma)^l \gamma \tag{8-4}$$

From Equation 9-4, it can be seen that with the discrete uniform traffic of multiple PUs, the number of empty slots before next transmission is geometrically distributed. With the large number of traffic generated by multiple PUs, the geometric distribution approaches the exponential distribution. This is an interesting result given that each primary user is sending at fixed interval with uniformly distributed variance. This leads to an important claim namely, *the whitespace profile of the channel follows an exponential distribution if the primary users have a sufficiently large traffic generation rate.*

8.4.4 Whitespace Capacity

Figure 8-9 shows the fraction of channel idle time that is left over from the PUs traffic for various numbers of primary users and packet arrival rates for Poisson traffic. Observe that the fraction of idle time is large when there is a smaller number of PUs. As the number of PUs increase the available capacity to the SUs decreases. Moreover, with faster transmission times, i.e. smaller packet arrival intervals, the capacity reduces at a higher rate. For example, with 200 ms inter-packet arrival time and 10 PUs the fraction of idle time is 0.89 compared to 0.73 for the 80 ms case. These capacity trends are also similar for the Uniform case shown in Figure 8-10.

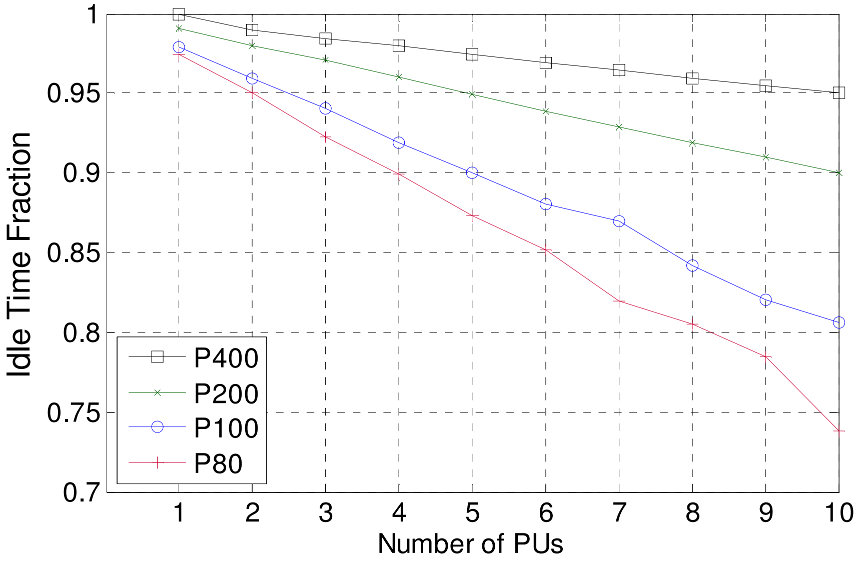


Figure 8-9: Idle time fraction for various Poisson PU inter-packet arrivals

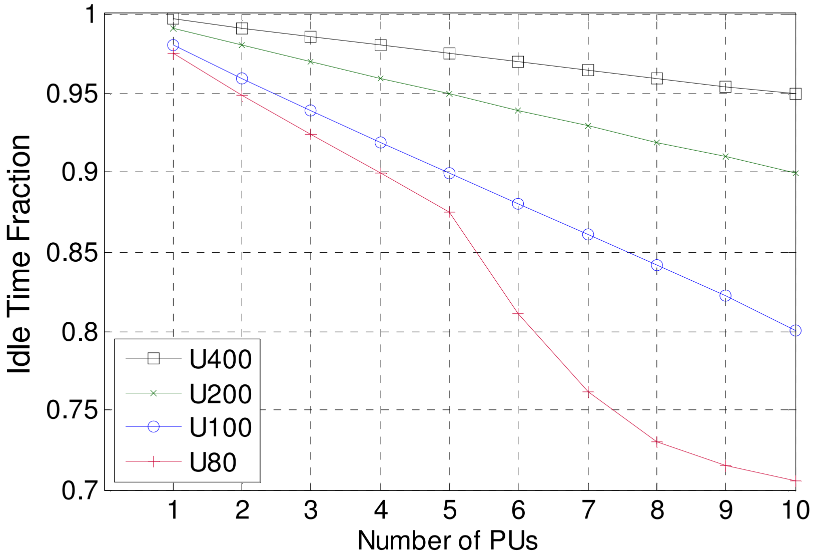


Figure 8-10: Idle time fraction for various Uniform PU inter-packet arrivals

8.5 Performance Evaluation of TBVAP

8.5.1 Impacts of Primary User Load

Figure 8-11 demonstrates the impacts of varying PU load on the SU throughput and PU disruption. The PU inter-packet arrival duration is varied from 80 ms to 400 ms for simulating different loading conditions. The SU packet duration i.e. TO slot, was fixed to 1 ms. The number of SUs is set to 15, with each SU generating packets with a 100 ms Poisson inter-arrival time. The sensing interval T_p is set to 5 µs. The user-defined disruption bound (*DB*) is set to 0.05 [6], which indicates that only 5% of the PUs' traffic is allowed to experience disruptions from the SUs' transmissions. The value of TBV is calculated dynamically depending on the current PU traffic statistics.

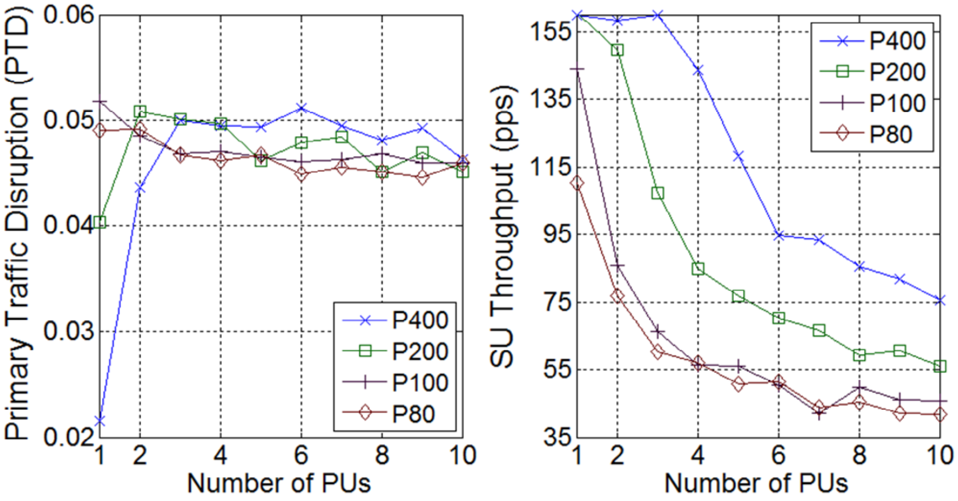


Figure 8-11: Performance for Poisson distributed PU traffic

From Figure 8-11, observe that TBVAP is able to maintain the required primary disruptions within the specified Disruption Bound (*DB*) across a wide range of PU numbers and data rates. As the number of PUs increases, the duration of usable whitespace in the channel reduces. Moreover, the number of distinct whitespaces increases with the PU data rate, since the whitespaces become more fragmented. Given that the SU access strategy is executed on a per-whitespace basis, with the increased number of distinct whitespaces, the SU attempts to execute its PU protection mechanisms (e.g. TBV in DSTS) more often. This causes the SUs' throughput, presented in terms of packets per second (pps), to decrease due to increased number of PUs. When the PU data rate (i.e. inter-packet arrival time) is decreased from 400 ms to 80 ms, the resulting throughput also decreases.

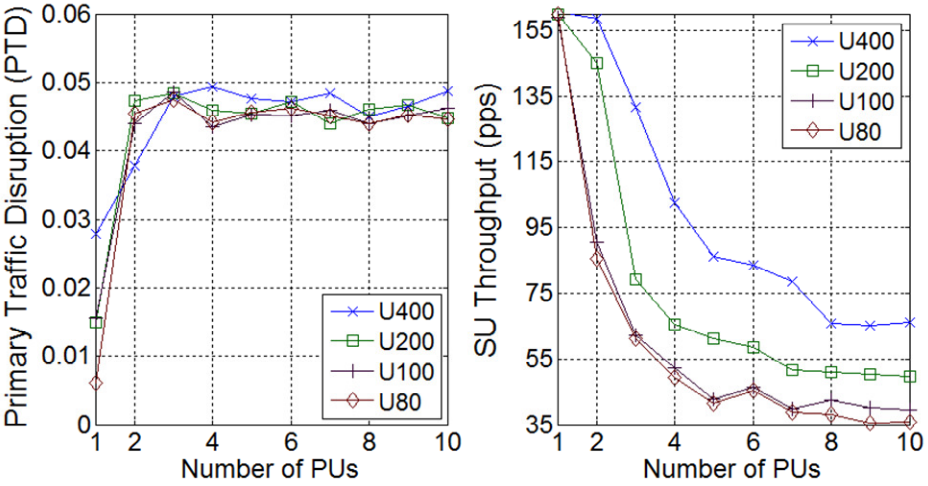


Figure 8-12: Performance for uniformly distributed PU traffic

Figure 8-12 shows results for the same setup, but with a ±20% uniformly distributed inter-packet arrival variation PU traffic. Note that the performance trends for this scenario are very similar to those for the Poisson case as shown in Figure 8-11.

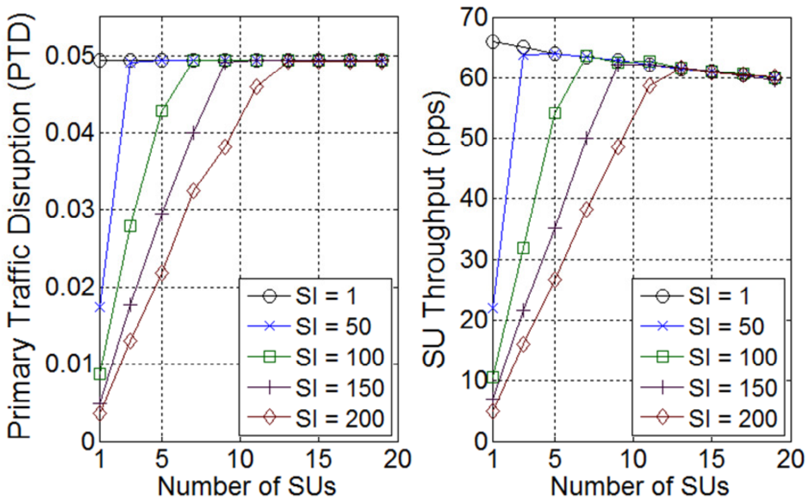


Figure 8-13: Performance for multiple SUs under Poisson PU traffic and 4 PUs

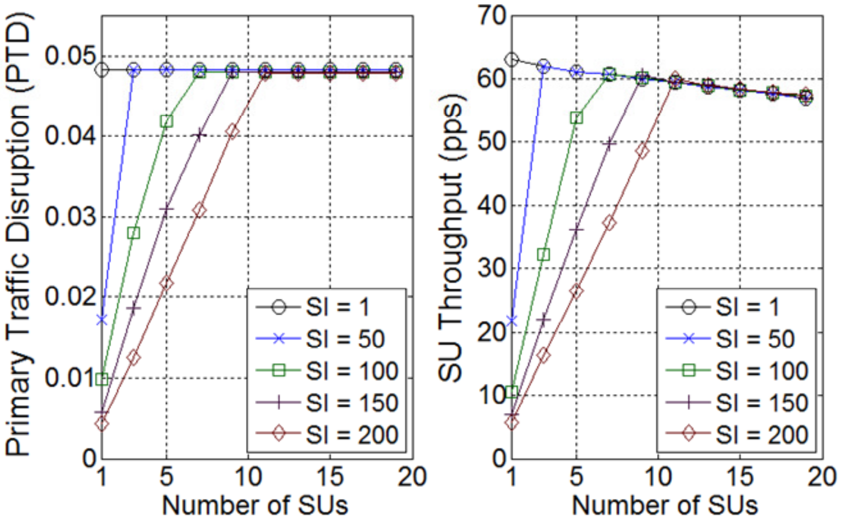


Figure 8-14: Performance for multiple SUs under Uniform PU traffic and 4 PUs

8.5.2 Impacts of Secondary User Load

Figure 8-13 and Figure 8-14 show the impacts of varying number of SUs for both Poisson and Uniform traffic respectively. In the Poisson and Uniform case, the number of PUs is set to 4 and the inter-arrival time is 100 ms. The Secondary user Inter-packet time (SI) is varied from 200 ms to 1 ms, to represent increasing SU load. Each SU independently generate packets according to the specified data rate. Observe that with a single SU and high data rate, the maximum allowable throughput for the SUs is reached. Moreover, the disruption bound is reached. As the number of SUs increases, the throughput decreases. This occurs, because of the increased number of collisions between the larger numbers of SUs. It follows that the collisions within assigned TOs are increasing with larger numbers of SUs. When the SU traffic rate is lowered (i.e. 100 ms) the amount of PTD increases until it reaches the disruption bound, DB. This occurs because the channel is able to support the SU load. Furthermore, the maximum network throughput is achieved with different numbers of SUs depending on the SU traffic load. For example, with SI = 50 ms, the maximum throughput is reached around 5 SUs. In contrast, with SI = 20 ms, it is reached around 13 SUs. After the maximum throughput is reached, increasing the number of SUs reduces the SU throughput due to increased collisions.

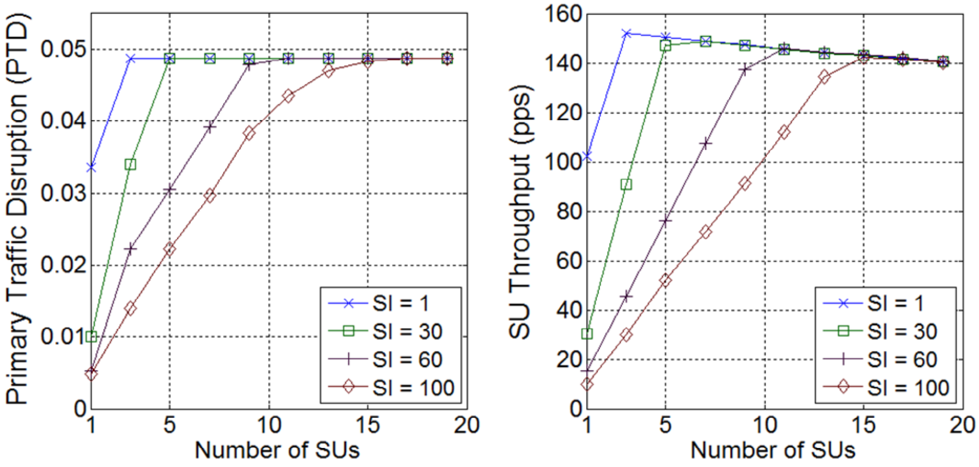


Figure 8-15: Performance for multiple SUs under Poisson PU traffic and 1 PUs

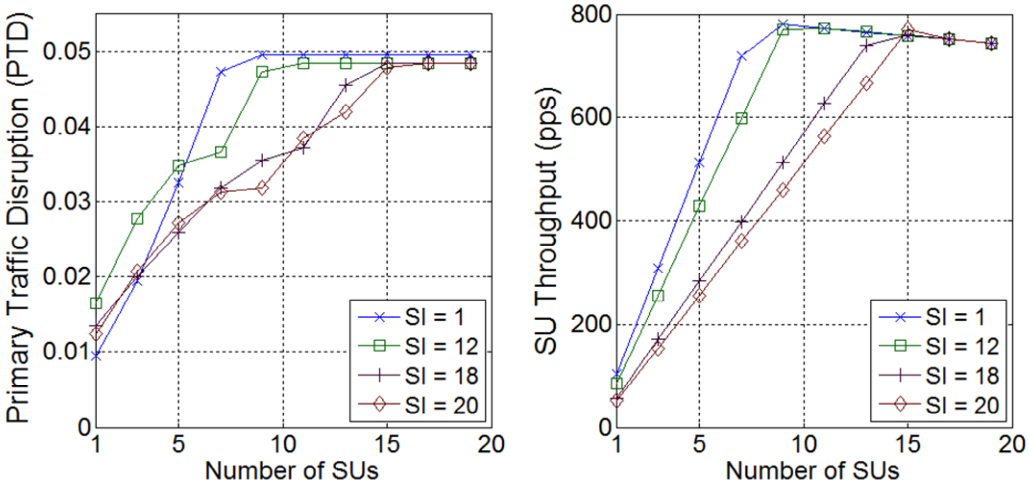


Figure 8-16: Performance for multiple SUs under Uniform PU traffic and 1 PUs

Figure 8-15 and Figure 8-16 present the results for a single PU with Poisson and Uniform distributions respectively. Both traffic profiles have a 100 ms inter-packet arrival time. Observe that the channel with Uniform traffic can support over 5 times the amount of SU throughput with respect to the Poisson traffic. This interesting result comes from the properties of the whitespace. Notice the whitespace pdf for 1 PU in Figure 8-7, the uniform distribution creates a large whitespace duration for lasting for ≈ 160 ms. This type of whitespace provides SUs with a large amount of channel time to utilize. In contrast, the Poisson distribution with 1 PU in Figure 8-5, creates highly dynamic whitespace durations. Therefore, the SUs must be more conservative to protect the PU traffic. As shown in Figure 8-15 and Figure 8-16, when the number of PUs is increased the whitespace characteristics are similar between the Uniform and Poisson profiles. Therefore, the SU performance is similar. These results follow from the discussion in Section 8.4.3.

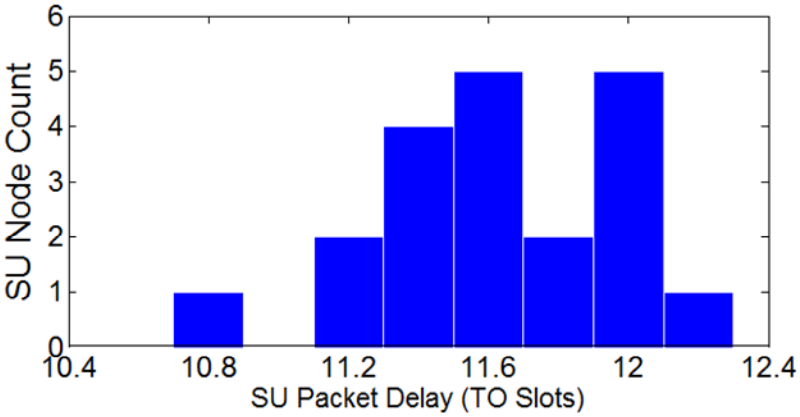


Figure 8-17: Delay performance for 20 SUs under Poisson traffic with 5 PUs

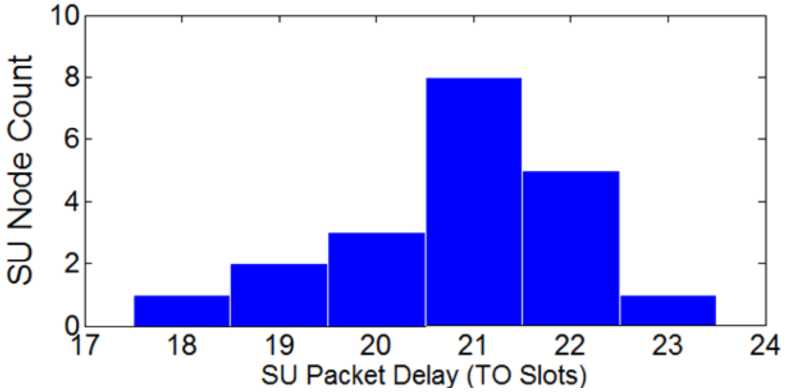


Figure 8-18: Delay performance for 20 SUs under Uniform traffic with 10 PUs

8.5.3 Delay Performance

As discussed is Section 8.2, a feature of the TBVAP is supporting fairness amongst the SUs. Figure 8-17 shows the packet delay in terms of TO slots for 20 secondary users. The primary user traffic is Poisson with 80 ms inter-packet time and 5 PUs. Observe that the delay range for all SUs is clustered around 11 slots signifying that each SU is given equal access to the channel. This trend is also shown in Figure 8-18 where the primary user traffic is uniformly distributed with 80 ms inter-packet time and 10 PUs. These results show that with the dynamic nature of the PU traffic the DSTS protocol is able to provide fairness amongst the SUs.

8.6 Summary and Conclusion

In this chapter, the Transmission Bitmap Vector Access Protocol (TBVAP) was introduced to support access for multiple SUs in a wireless channel. The protocol provides fairness in that each SU is given an equal chance to utilize a Transmission Opportunity. These features were demonstrated using results from a multiple PU CSMA based whitespace profile.

Chapter 9: Summary and Conclusions

9.1 Contributions

In this thesis, we have developed access strategies that allow secondary user (SU) devices to coexist with primary user (PU) devices through utilizing unused spectrum or whitespace found between PUs' transmissions. A secondary user continually monitors its surrounding whitespace, models it, and then attempts to access the available spectrum so that the effective secondary user throughput is maximized while the resulting disruption to the primary users is kept limited to a pre-defined bound. This opportunistic access during ultra-short and non-deterministic whitespaces is termed as *bandwidth scavenging* by the secondary users. In other words, the SUs scavenge the channel capacity left over by the PUs.

Chapter 3 presented the whitespace properties for an Ad hoc 802.11 PU network under a wide range of topology and traffic variations that were experimented with using an ns-2 based simulator. A number of key observations were made from the whitespace statistics. First, the detected whitespace durations from the measured Received Signal Strength Indicator (RSSI) trace vary a great deal across different topologies and traffic patterns. This indicates that a secondary user may experience vastly different whitespace environments when attempting to access primary networks. Second, it was shown that there is available capacity for SUs to scavenge depending on the traffic profile of the PU. Third, regardless of the traffic profile or topology, the vast majority of the whitespaces (above 90%) last for less than 1 ms. This results from the large number of *small* whitespaces created during the RTS-CTS-DATA-ACK cycle for each packet transmission.

In Chapter 4, a secondary user access strategy called the Contiguous SU Transmission Strategy (CSTS), is described. CSTS is used to access an Ad hoc 802.11 PU whitespace profile. CSTS exploits the properties of the whitespace, analyzed in Chapter 3, by avoiding transmissions during the first millisecond of a whitespace. Once a whitespace is accessed, the parameter J_{max} limits the amount of SU packets sent within that whitespace. Under this process, the secondary users maximize its own throughput while keeping the primary user disruptions to a predefined bound.

In Chapter 5, an extensive set of results to evaluate the CSTS access strategy is presented. Simulation-based experiments provided a method to explore the impacts of different parameters on CSTS. The primary user load results showed that the protocol is able to maintain high SU throughput under different load conditions. Results from the SU packet size experiments showed that a smaller SU packet provides better throughput for the SUs. Varying PU traffic pattern results showed that CSTS could dynamically adjust to the PU traffic over time. Additionally, periodic sampling of the whitespace provides enough information to maintain good SU throughput while maintaining PU disruption bounds. The channel sensing interval results showed that as long as the sensing period is smaller than usable whitespaces, there is enough information to operate the CSTS protocol. Analytical expressions for the secondary throughput and primary disruption resulting from the proposed schemes were also developed for validating the simulation results. Finally, the results from the effects of varying disruption bound showed that PUs allowing a small disruption could facilitate more bandwidth scavenging by the SUs.

Chapter 6 presented a TinyOS based testbed implementation of the Bandwidth Scavenging Concept (BSC). The BSC was applied to traffic protection in heterogeneous networks. A Primary User Network (PUN) was implemented with the purpose of generating various traffic patterns on a channel. Along with a Secondary User Network (SUN) was created to perform measurements of the PU channel activity. Then access strategies is utilized to send messages in between the PU transmissions. The PUN was shown to be able to create multiple traffic patterns using a PC and TelosB mote configurations. The SUN was shown to be able to measure the PU transmissions on the channel. Additionally, multiple SU access strategies were implemented on the motes.

The access strategies were then evaluated using the testbed that was implemented using TelosB motes. From the experimental results, it was shown that the access strategy was able to support the offered SU load. Due to the limitations of the TelosB hardware, additional simulations were run in Matlab. The simulation results showed that the CSTS protocol would throttle back the transmissions of the SUs to protect the PU traffic. Similar to the Ad hoc 802.11 results the BSC was shown to minimize PU throughput while maximizing SU throughput.

In Chapter 7, a new and novel secondary user access strategy named Divided SU Transmission Strategy (DSTS) was presented that provided a method to access a general PU whitespace profile. The access strategy utilizes multiple transmission opportunities within a whitespace to find the best times to transmit SU packets. These TOs are consolidated into a Transmission Bitmap Vector (TBV) that defined when SUs should transmit within a whitespace to maximize SU throughput while limiting PU disruptions to a predefined bound. Optimal and Sub-optimal protocols are developed to select the best transmissions opportunities. Evaluation of DSTS under different traffic profiles showed that the protocol outperforms comparison protocols.

Chapter 8 a protocol named the Transmission Bitmap Vector Access Protocol (TBVAP) was introduced to support access for multiple SUs in a wireless channel. The protocol provides fairness in that each SU is given an equal chance to utilize a Transmission Opportunity identified by the DSTS protocol. These features were demonstrated using results from a multiple PU CSMA based whitespace profile.

9.2 Future Work

The channel analysis and access strategies presented in this thesis were developed for a single wireless channel. The single channel case presented many challenges as highlighted throughout the thesis. Another challenge that arises in this paradigm is the multiple whitespace domains (MWD) protection problem. The MWD problem arises when there are multiple PUs and multiple SUs spatially distributed within a given area. Research in this area has been mostly focused in the context of multiple channels [40-42, 80-100]. In this approach, SUs have access to two or more channels, which may or may not have PUs using the channels. Typically, the SU solution to avoid PU disruptions is to switch channels when a PU is occupying the channel. With PUs occupying different channels, separate whitespace domains may exist on each channel.

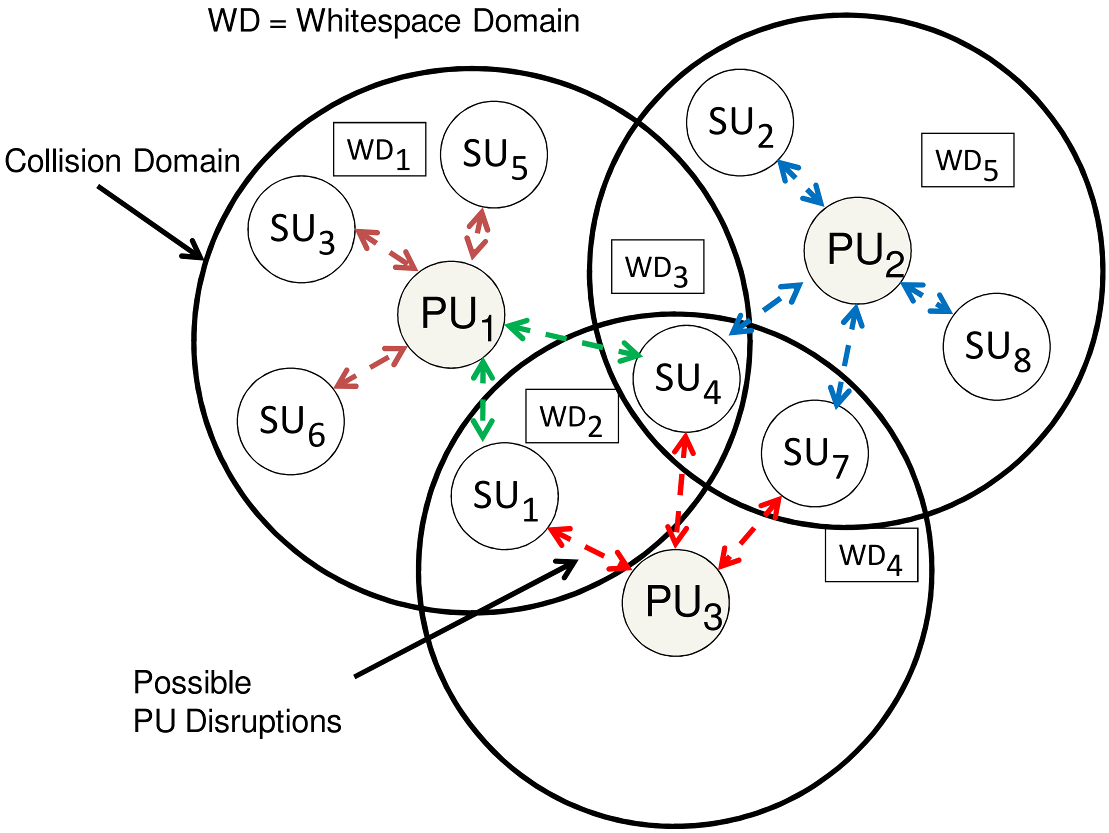


Figure 9-1: Example of Multiple Whitespace Domains

Even in a single channel environment, multiple protection zones can be created for each PU. As shown in Figure 9-1, PU_1, PU_2 and PU_3 have individual collision domains that must be protected from SU traffic. Consequently, this means that protecting the PUs becomes more difficult and some additional issues arise. SUs affected by a single primary user, such as SU_3, SU_5, and SU_6 in whitespace domain (WD_1), would have a single access strategy (i.e. TBV) to protect the PU. When a SU is within the collision domains of multiple PUs, such as SU_1, SU_4, or SU_7, (WD_2, WD_3, and WD_4 respectively) there is a different solution for each domain. Given that SUs may be separated by multiple hops, they may not be able to hear the transmission of other SUs. Thus, the combined effects of these two solutions can create more disruptions on a PU than allowed. The objective is to develop a secondary user access strategy that allows the protection of primary users within multiple whitespace domains.

APPENDIX

Appendix A: TelosB Based TestBed Implementation

A.1 Introduction

With Bandwidth Scavenging, a secondary user device attempts to access a wireless channel that is occupied by a primary user device. In this appendix, the details of a TelosB based tested is presented. The testbed is implemented as a Primary User Network (PUN) and a Secondary User Network (SUN). The purpose of the PUN is to generate various traffic patterns on a channel. While the purpose of the SUN is to first perform measurements of the PU channel activity then utilize an access strategy to send messages in between the PU transmissions.

A.2 Primary User Network (PUN)

This section will discuss the details of the Primary User Network (PUN). The PUN system components are implemented on a PC and on the motes. Recall that the purpose of PUN is to generate various traffic patterns on a channel. The implementation provides a flexible creation of different traffic patterns on the channel:

- Uniform Traffic Distributions
- Multi-Modal Traffic Patterns
- Trace based Traffic Patterns

These traffic patterns can be created at various data rates with different variances. Figure A-1 shows an overview of the communication flow of the PUN system. On the PC side, the Traffic Generator creates packets according to the desired traffic pattern and sends the packets to the measurement class. The measurement class keeps track of the timing of each generated packet and sends them to the communication interface. This interface takes care of the communication between the PC and motes. Once a mote receives the packet from the communication interface, the packet is sent on the channel. Then it is received by another mote and sent back to through the PC-mote communication interface. The communication interface then forwards the message back to the measurement class. From there an output will be generated detailing the statistics of the packets. In the example figure, there are 6 traffic flows with three motes.

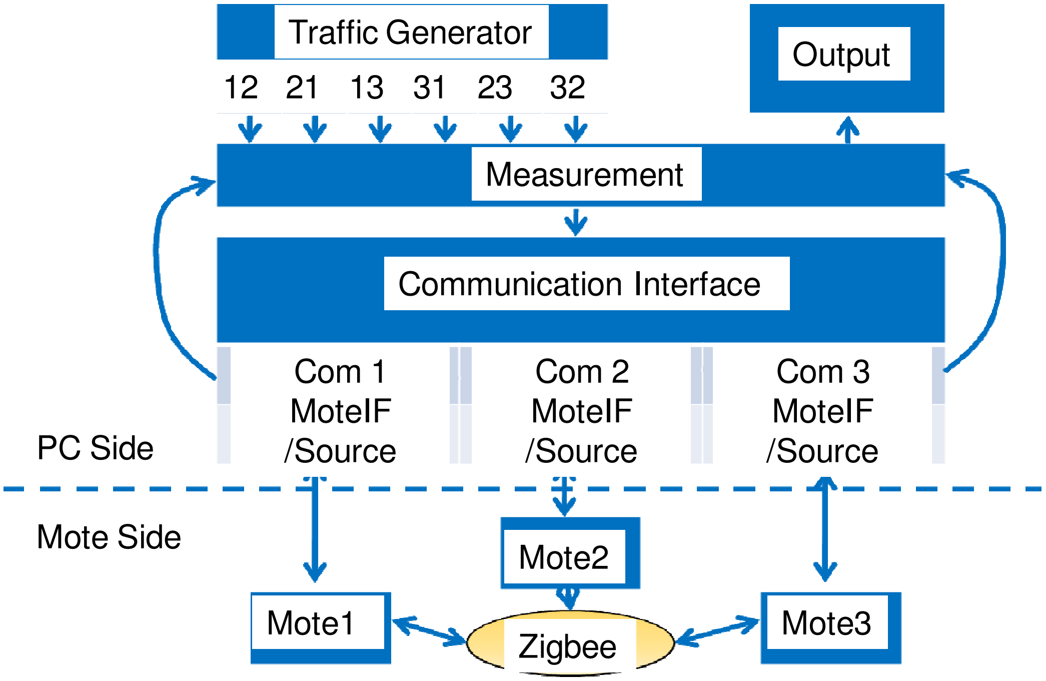


Figure A-1: Overview of PUN implementation

A.2.1 Mote Side Implementation

The TinyOS implementation on the mote is mostly used for forwarding messages from the PC to the mote radio and vice versa. First, the PC sends a command to the mote, which includes the destination mote and the payload. From there the message will be put into a queue on the mote and dispatched as soon as possible. On the return path, the mote will receive a message, check whether it has the correct destination and then put the message into a second queue. From there it will be sent to the pc as soon as possible.

A.2.1.1 Application Layer

As discussed in the previous section, the program on the mote bridges messages from the PC to the radio and vice versa. This bidirectional bridging behaves similar to a base station application, where one mote receives messages in order to forward them to the PC.

To minimize the workload generated by the PC-mote communication only message type, source, destination, id and a randomly generated payload are transferred from the pc. Figure A-2 shows the message structure on the mote. Each of these variables uses two bytes. The message type currently not used and is kept in the implementation for potential future use. The mote can store the message data in a queue and start dispatching from the queue if it is not already active. The queue is implemented as a ring buffer, which has a FIFO behavior.

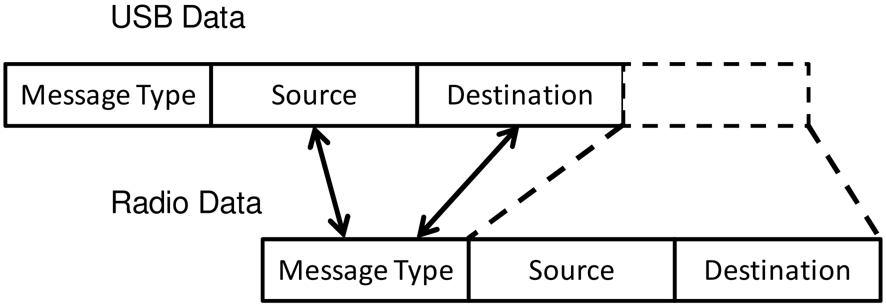


Figure A-2: Message structure of mote

The ringbuffer consists of an array of data structures and two pointers. The data structs contain the actual data of the message, while the pointers point data-in and data-out positions in the array. The pointers increment and point to the different structs on the array and start again from the beginning when they reach the end of the array.

As soon as data sending is started, it takes the first data set from the queue and generate a new AM-TYPE message. The message contains the desired data and a dummy payload. The length of the dummy payload can be set on compile time. In the experiments, the total payload length was set to 60 bytes. The message id and the random generated payload use 4 bytes of these 60 bytes, while the source and destination are found in the message header. The AM-TYPE message adds one more additional byte of payload to the message after being sent to the lower layer.

If the sending to the lower layers returns anything except “success”, the application schedules a resend. This improves the sending reliability but does not guarantee a successful transmission because it only handles internal resource failures but does not have a mechanism like ACK packets.

When the lower layer signals the application layer that a message has been received from the radio, the application layer extracts the same variables from the message that were stored in the message previously. These variables are stored in a queue to be sent to the pc. Similar to the path from the mote, the dispatch of the messages to the pc are activated if it is not active yet.

A.2.1.2 MAC Layer

There are two MAC layer implementations for the PUN, which take care of the dispatch of the messages to the Physical layer. These two protocols are non-persistent CSMA with backoffs and a simplified CSMA-Mechanism with backoffs turned off.

The non-persistent CSMA is the standard MAC-Layer implementation of the TinyOS with the low power listening turned off. In this implementation, first the mote attempts to access the channel and senses if it is free. To sense whether the channel is free or not, the mote takes two clear channel assessment samples separated by a 250 µs waiting period. This waiting period is to make sure an ACK-Turnaround period is not sampled. If the channel is busy the mote performs a congestion back off to allow the other mote to finish their transmissions. After that time, the mote checks again, if the channel is free. The congestion is randomized and increased for each retry. If the mote cannot detect a free channel for several retries, it returns fail.

The simplified CSMA is a modified version of the non-persistent CSMA with the back offs turned off. The simplified CSMA is implemented in an independent application platform called telsobPU. Since the backoffs are disabled, the mote waits until 2 consecutive samples to indicate that the channel is free. The number of required consecutive samples can be set on compile time. If the channel is found to be free, the mote sends its message.

A.2.2 PC Side Implementation

The PC side implementation is done using Java. The implementation is done across several functional classes as show in Figure A-3. As shown in the figure, there is a Traffic Generator Class, a Measurement Class, a Communication Interface Class and an Output Class. Additionally, there are classes for a graphical interface and batch processing of measurements.

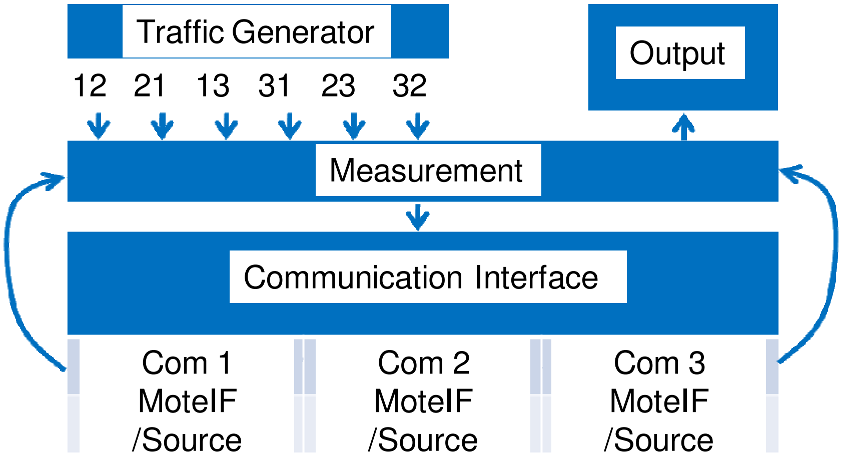
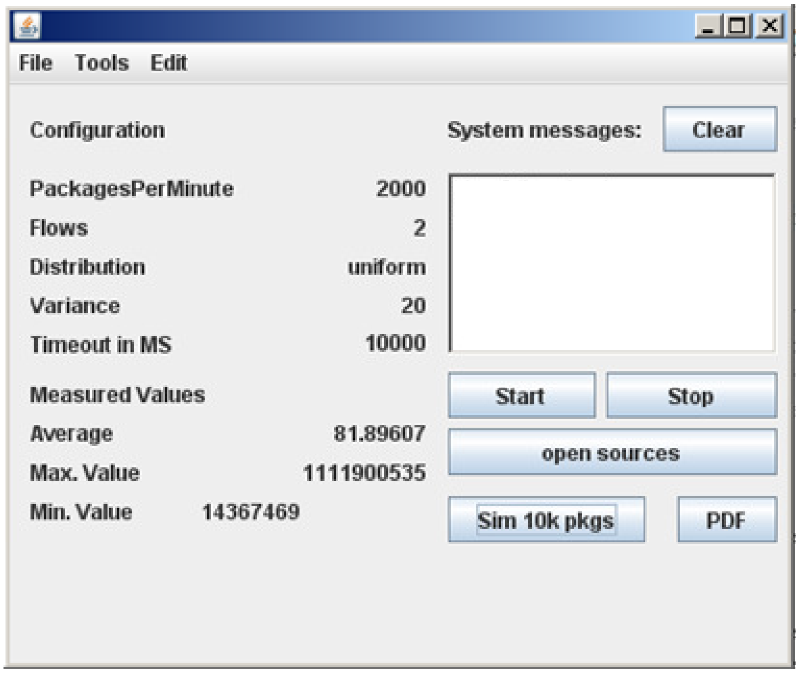


Figure A-3: PC side overview

The Control Interface is shown in Figure A-4. The screen is split up into four major logical parts. On the top left corner, there is a listing of the current configuration of the traffic profile. In the example case shown, there are two flows sending 2000 packets per minute with a uniform distribution and 20% variance. One flow is from mote A to B and the other one is from mote B to A. The number of packets sent is split evenly between the two flows, which leave each of them with 1000 packets per minute.

In the lower left hand corner, the current measurement statistics are shown. In this example, the average round trip time (RTT) for a message was 81 ms and the minimum RTT has been 14 ms while the maximum has been 1 second. On the top right field, system messages are displayed.



Configuration

PackagesPerMinute	2000
Flows	2
Distribution	uniform
Variance	20
Timeout in MS	10000

Measured Values

Average	81.89607
Max. Value	1111900535
Min. Value	14367469

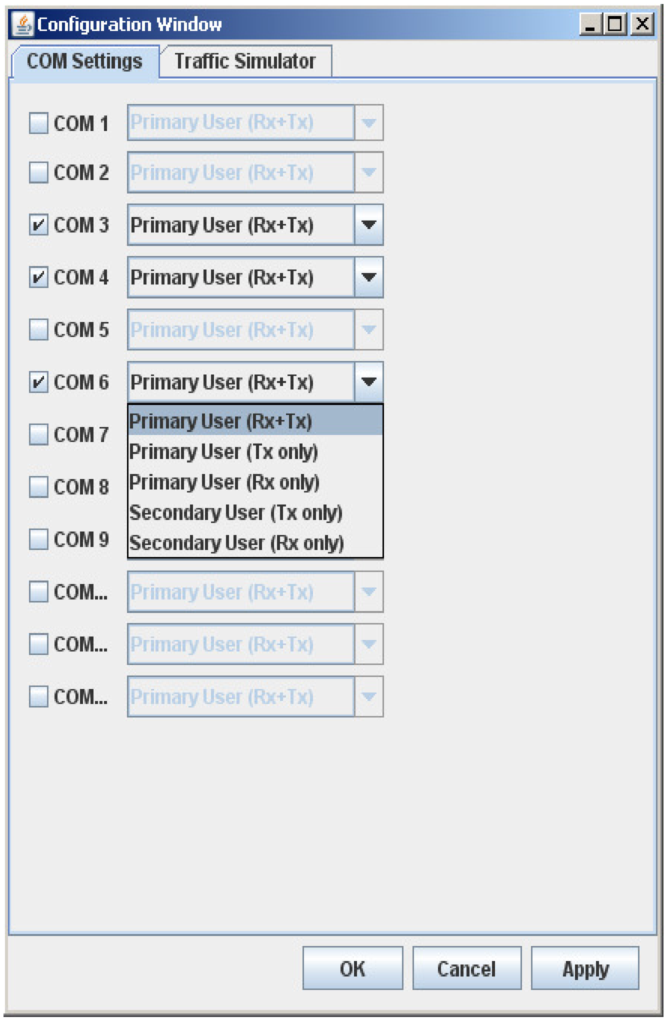
Figure A-4: Primary user control interface

The lower right hand corner allows the user to control the other classes. The button “open sources” tells the communication interface to set up the mote communication and prepare the necessary configurations for the test run. The Start and Stop buttons control the packet generation in the traffic generator. “Sim 10k pkgs” schedules 10,000 packets to be sent by the traffic simulator. After a test has run, clicking the PDF button prints out the measured data. The menu bar allows the user to access further settings like com-port setup or traffic profiles. The modular design with several functional classes provides a flexible and reliable implementation. In the following sections, a detailed description of the implemented classes are given.

A.2.2.1 Communication Interface

The Communication Interface (CI) is the central access point for data to and from the motes. It provides functions for other classes to send data to the mote and sends received data from the motes to the desired destination classes.

The CI provides a flexible implementation, which can access up to 12 motes. Figure A-5 shows the CI configuration screen for the Primary User. The inputs given to the CI are the currently used COM-Ports and the flow directions that should be set. The possible options are sending only, receiving only, and sending and receiving. From the given information, the program calculates which flows the Traffic Generator shall set up and access the necessary COM-ports. A list of flows and accessible COM-ports can be accessed from other classes. To allow all the motes to be accessed concurrently and independently, a thread is generated for each mote. Each thread accesses the mote through a mote interface provided by the TinyOS tool-chain. The mote interface itself accesses a “phoenix source” which accesses a “packet source”. The packet source access the actual comport. The reason for this layering is that with each layer more functionality is implemented. Filtering for specific messages and signaling that a specific message type is received are some of the features implemented by these layers.



☐ COM 1 Primary User (Rx+Tx)
☐ COM 2 Primary User (Rx+Tx)
☑ COM 3 Primary User (Rx+Tx)
☑ COM 4 Primary User (Rx+Tx)
☐ COM 5 Primary User (Rx+Tx)
☑ COM 6 Primary User (Rx+Tx)
☐ COM 7
☐ COM 8
☐ COM 9
☐ COM... Primary User (Rx+Tx)
☐ COM... Primary User (Rx+Tx)
☐ COM... Primary User (Rx+Tx)

Figure A-5: Primary User communication interface configuration screen

A.2.2.2 Measurement Class Implementation

The Measurement Class (MC) is responsible for keeping track of each packet and for generating a log of the measurement runs. To provide this functionality, it keeps track of the messages that are being sent. To keep track of these messages, the MC provides a send and receive interface for other classes.

The MC is above the Communication Interface. The idea here is to capture the timestamps for each packet as close to the actual dispatch time as possible. In the MC, a list of all messages that have been sent and not returned yet is maintained. Figure A-6 shows the MC data struct. The start timestamp, message type, source, destination, message ID and a two-byte random payload are saved in the struct. The message id works as a unique identifier for the message.

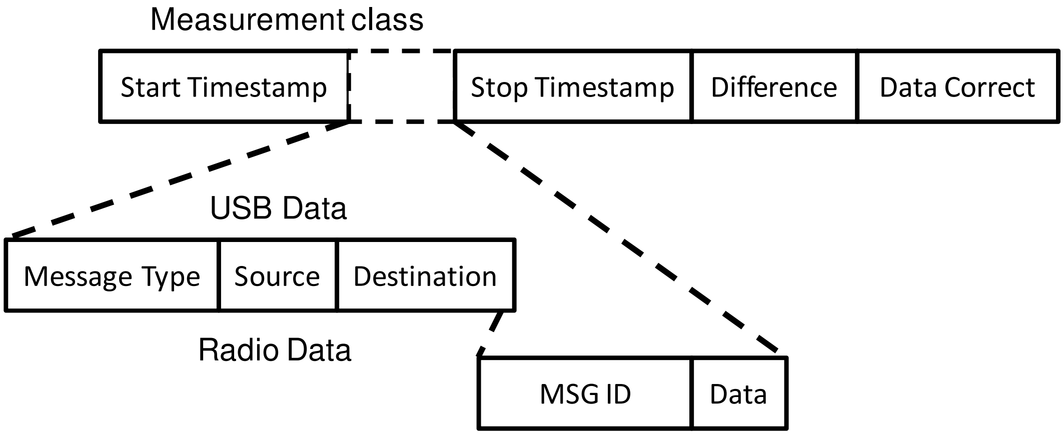
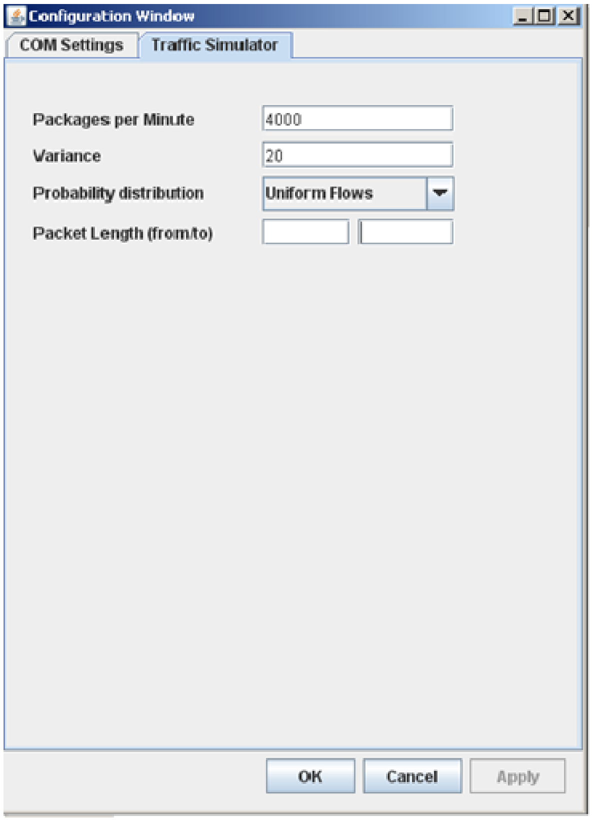


Figure A-6: Measurement data structure

The figure also shows that a larger amount of data is stored on the PC side than what is passed down to the mote. The mote handles the messages as discussed in Section A.2.1.1. Upon reception in the mote, the message is passed on to the output handling. A java timer task is scheduled which regularly checks for messages. If a message is overdue, the timer task marks them as timed out.

A.2.2.3 Traffic Generator Class

The Traffic Generator Class (TGC) creates messages according to a desired traffic profile. The Traffic Generator is implemented in two classes, a control class and a message creation class. The message generation can be controlled by the input form, which is displayed in Figure A-7. Different rates, variance and probability distributions can be set. In the current implementation, the probability distributions are uniform distribution, two multi modal versions and trace based traffic.



Packages per Minute: 4000

Variance: 20

Probability distribution: Uniform Flows

Packet Length (from/to):

Figure A-7: Traffic Simulator control screen

After the variables are set, the control class reads the configuration and creates a thread for each flow. The thread is in a loop while a flag called MessageGenerationON is set to true. The underlying control class controls this flag. Within this loop, it checks whether it is time to send by comparing the timestamp for the next send to the current system time. If it is time to send, a packet is generated and sent to the Measurement Class described above. After sending the packet, the time for the next packet is calculated. Then the loop waits again until the next packet is ready. To wait the thread is sent to sleep for one nanosecond. This is shown to be the most efficient implementation for the java and Linux scheduler. The calculation of the next times to send is based on the used access strategy. For the uniform traffic probability distribution, the next sending time is calculated by calculating the average inter-packet time plus or minus the variance percentage and adding this value to the last timestamp.

In the multi-modal case, there are two different implementations. The first implementation works like the uniform probability distribution, though only a certain percentage of messages are actually sent. The corresponding Probability Density Distribution can be seen in Figure A-8. The other implementation is a dual version of the multi modal protocol. It works like the first version though an “else” statement is added to the sending condition. The else statement calculates the next value like in the uniform case and then send the message immediately. This creates a two-peaked PDF as shown in Figure A-9. This peak has a uniform distribution while the second peak has the shape of a triangle.

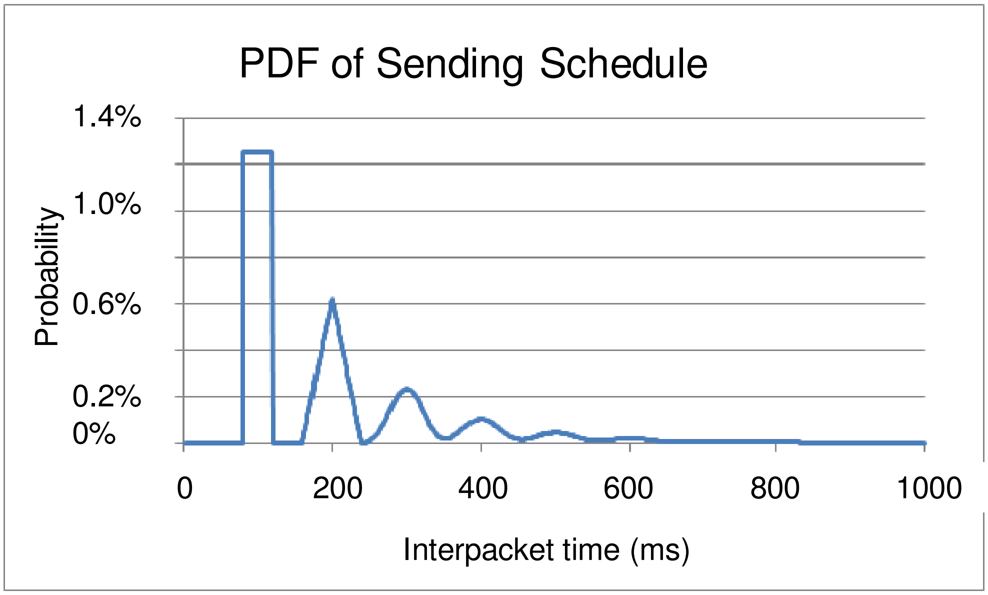


Figure A-8: PDF of the first multimodal scheme

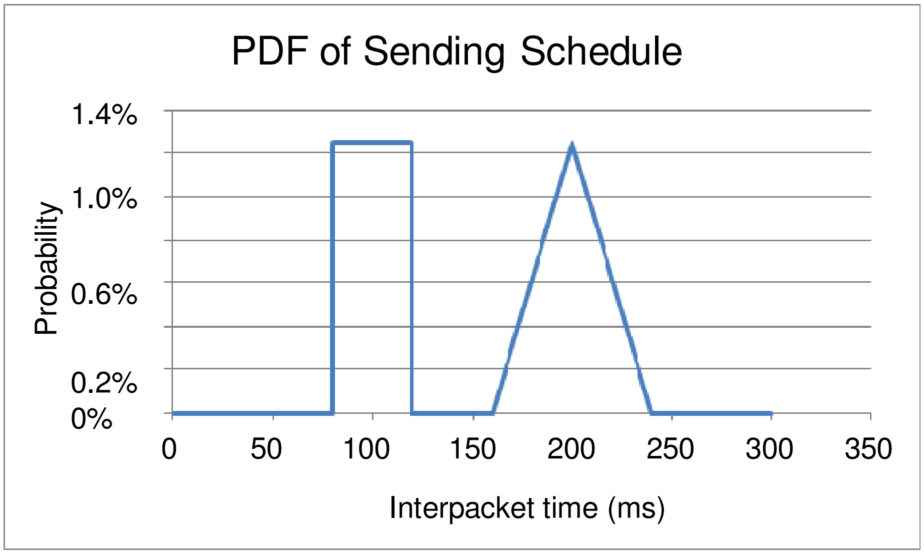


Figure A-9: PDF of the second multimodal scheme

The last implemented traffic profile is a trace based packet generation. This approach takes an actual traffic pattern as an input. This input can be derived from a video or audio stream or any other application. These traces are then post processed and converted into inter-packet times. These times are stored in a file, which is imported by the java program. Next, a thread is created and the next time to send is calculated according to the next value in the tracefile. This implementation provides a flexible and accurate execution for the generation of different traffic patterns.

A.2.2.4 Output Class

The output class is designed to capture and pre-process the output date. The data is sent to the output class in the format described in Section A.2.1.1. From this data, the following indictors are created:

- A full log of the messages for debugging and verifying purposes
- The minimum value
- The maximum value

- The average Round Trip Time
- Number of messages that timed out
- Total Messages received
- Total Messages sent
- Messages with a Round Trip Time over 100 ms. This indicator is chosen as most messages are within the range of 20 to 40 ms.
- Probability Density Function for the Round Trip Time from 0 to 100 ms.

A.2.3 Traffic Generation Evaluation

In the evaluation of the system, it has proven to work reliably under most scenarios. Figure A-10 shows a typical example of a Round Trip Time (RTT) PDF. The PDF was created with an average inter packet time of 100 ms and a variance of 20%.

Packet loss and values over 100ms are extremely rare cases so this implementation builds a solid base line for measurements with the secondary user. However, some issues found during the evaluation of the implementation have to be addressed.

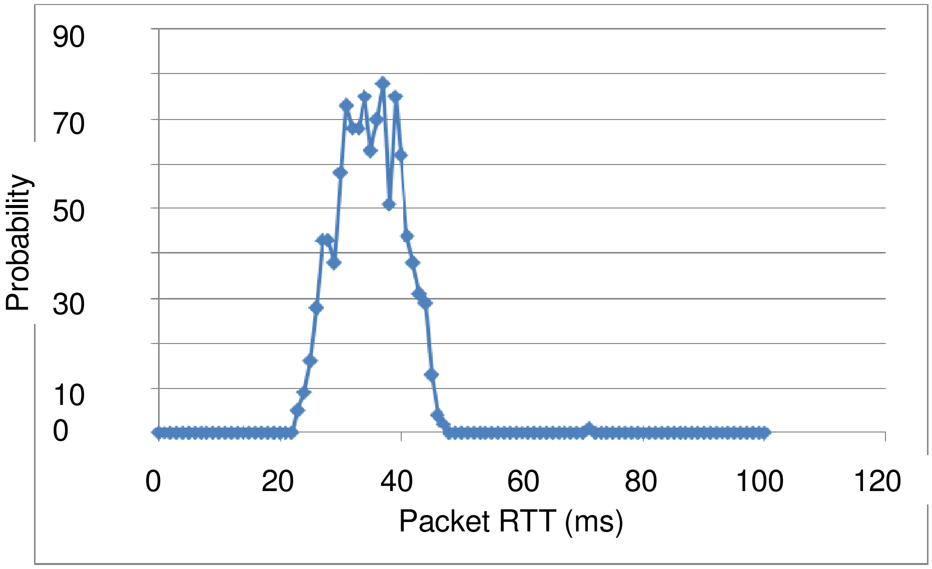
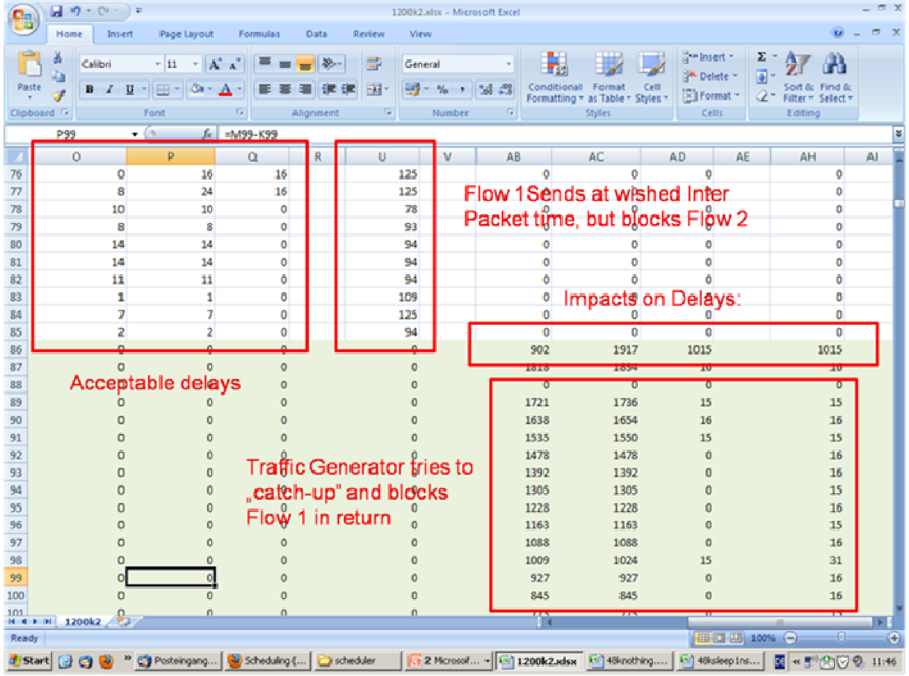


Figure A-10: PDF of the RTT distribution of packets

While stressing the system and using multiple flows, the system shows some unwanted behavior. Figure A-11 shows that the actual dispatch differs from the desired timings. The different flows seem to block each other. On the top left, the scheduled messages of the first flow are listed. The delays shown in the top right red box are all within an acceptable limits. The first column shows the time difference between the scheduled time and the time when it is to be sent to the mote. The second column shows the time difference between the scheduled time and the time the java has completed the dispatch to the mote. The third column shows the delay from before the sending to the mote to the time when the dispatch is completed. The second box on the top shows the inter packet time of the first flow. While all these values look good, the first flow blocks the second flow, which should be sending in between. This causes the delays to add up to one second inter-packet time. Afterwards the second flow tries to catch up and send its scheduled packets, so it is sending the messages to the mote as fast as it can. This leads to an inter packet time of 15 ms instead of 100 ms +-20% variance. Additionally, the first flow gets blocked and stops sending adding up messages in its queue.



P99 = =M99-K99

	O	P	Q	R	U	V	AB	AC	AD	AE	AH	AI
76	0	16	16		125		0	0	0		0	
77	8	24	16		125		[illegible]	[illegible]	[illegible]		0	
78	10	10	0		78		0	0	0		0	
79	8	8	0		93		0	0	0		0	
80	14	14	0		94		0	0	0		0	
81	14	14	0		94		0	0	0		0	
82	11	11	0		94		0	0	0		0	
83	1	1	0		109		0	0	0		0	
84	7	7	0		125		0	0	0		0	
85	2	2	0		94		0	0	0		0	
86	0	0	0		0		902	1917	1015		1015	
87	0	0	0		0		1818	1834	10		10	
88	0	0	0		0		0	0	0		0	
89	0	0	0		0		1721	1736	15		15	
90	0	0	0		0		1638	1654	16		16	
91	0	0	0		0		1535	1550	15		15	
92	0	0	0		0		1478	1478	0		16	
93	0	0	0		0		1392	1392	0		16	
94	0	0	0		0		1305	1305	0		15	
95	0	0	0		0		1228	1228	0		16	
96	0	0	0		0		1163	1163	0		15	
97	0	0	0		0		1088	1088	0		16	
98	0	0	0		0		1009	1024	15		31	
99	0	0	0		0		927	927	0		16	
100	0	0	0		0		845	845	0		16	
101	0	0	0		0		[illegible]	[illegible]	[illegible]		[illegible]	

Flow 1Sends at wished Inter Packet time, but blocks Flow 2

Impacts on Delays:

Acceptable delays

Traffic Generator tries to „catch-up" and blocks Flow 1 in return

Figure A-11: Example of Traffic Generator delays

The displayed case is a worst-case scenario, though it is a good example to address the problem of the scheduler in windows and the Java Virtual Machine. Both of these schedulers together determine the system response time. Unfortunately, these schedulers are optimized for throughput instead of real-time applications. Several tweaks in the program code have significantly improved the programs behavior however there is no guarantee that this does not happen in multi flow cases.

Another issue is that the total precision of actual sending times. Figure A-12 and Figure A-13 show that the scheduler has some kind of grid, which gets laid over the originally scheduled messages. The comparison shows that the messages are only dispatched at distinct peaks with a constant interval of roughly 10-15 ms. Figure A-14 and Figure A-15 shows the pdf of scheduled and sent packets with a variance of 20% respectively.

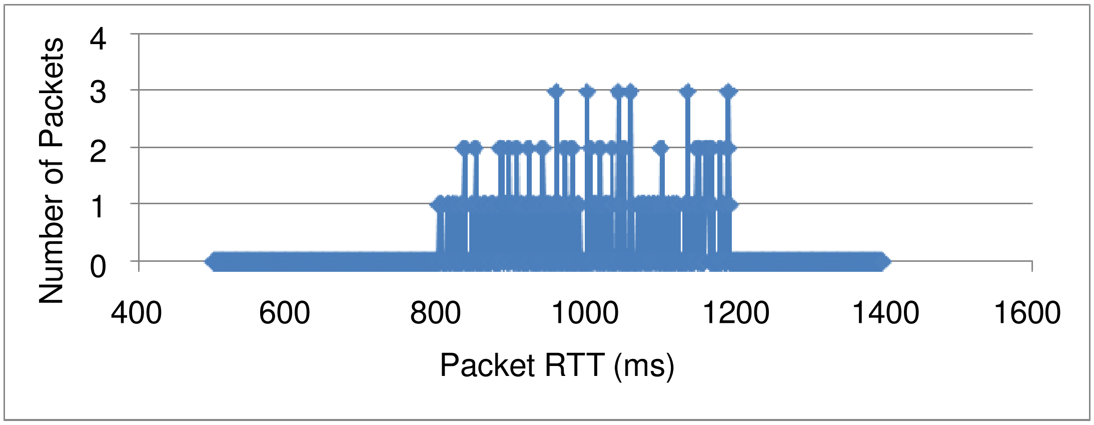


Figure A-12: PDF of scheduled packets at 1 pps with a variance of 20%

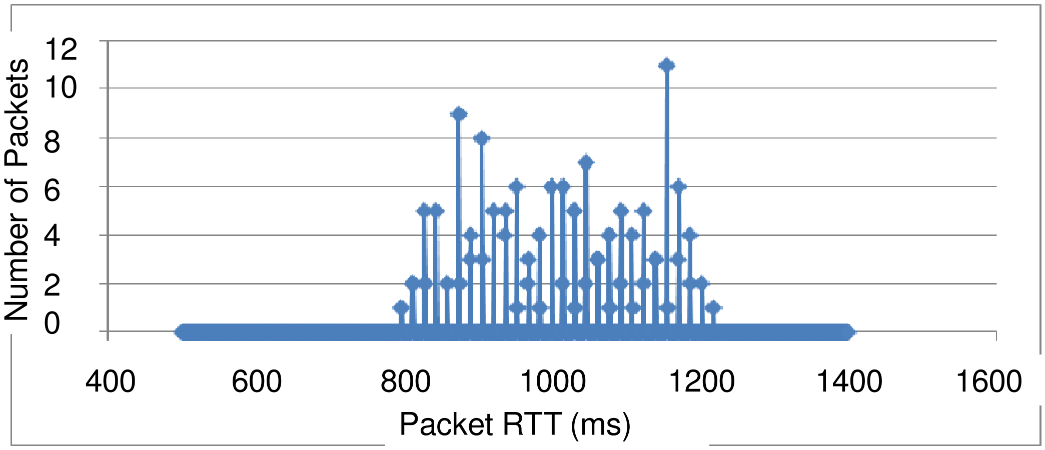


Figure A-13: PDF of sent packets at 1 pps with a variance of 20%

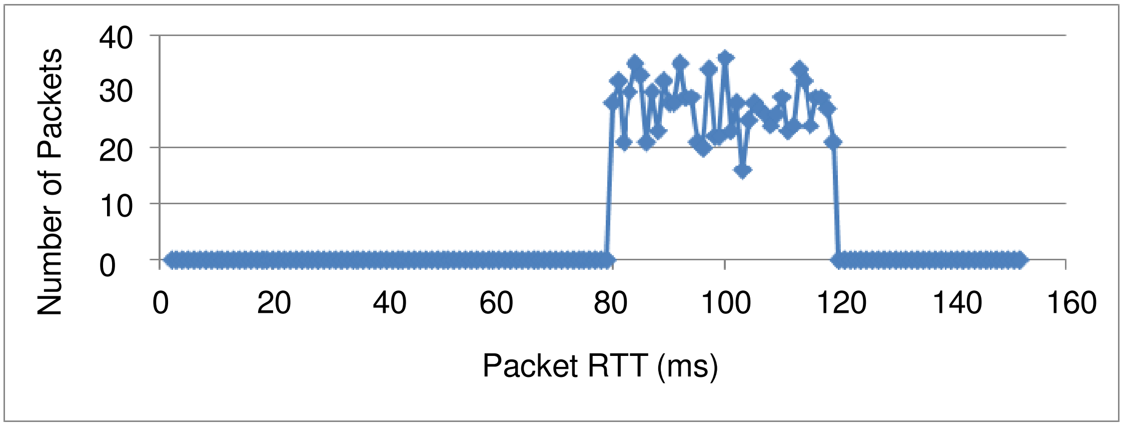


Figure A-14: PDF of scheduled packets at 10 pps with a variance of 20%

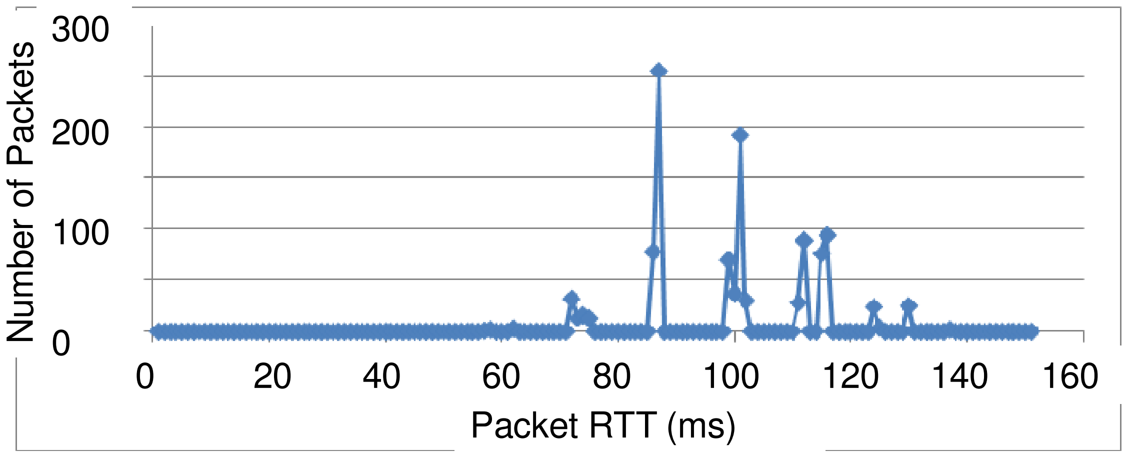


Figure A-15: PDF of sent packets at 10 pps with a variance of 20%

A.3 Secondary User Network (SUN)

This section discusses the details of the Secondary User Network (SUN). The SUN is implemented on the PC and on the motes. Recall that the purpose of the SUN is to perform measurements of the PU channel activity. Then utilize an access strategy to send messages in between the PU transmissions.

A.3.1 RSSI Measurement

A crucial part of the proposed secondary user access strategy is the gathering of information about the channel occupancy. In this work, the method used to collect PU channel activity information is done via Received Signal Strength Indicator (RSSI) based sampling. The channel is sampled with a TelosB mote running an RSSI-Sample application. The application was contributed by Stanford University to the TinyOS contribution repositories [77]. This application is capable of sensing the channel at a rate of up to 1.3 kHz.

Figure A-16 shows the main part of the RSSI-Sample application. When a timer interrupt fires, it calls a “Resource.request();” command and then schedules a new alarm at

ALARM_PERIOD (sampling interval). This procedure causes a loop, which is repeated until the desired number of samples is gathered.

```
async event void Alarm0.fired()
{
        call Resource.request();
        call Alarm0.start(ALARM_PERIOD);
}
```

Figure A-16: Pseudo-code of RSSI-Sampling

The Resource.request command produces a callback that accesses the CC2420 registers. When this happens, the application reads the RSSI value from the register and store it in the flash memory. The RSSI value register is being continuously updated and always provides the average RSSI value over the last 8 symbol periods (125µs) [101]. The application sends all samples to the PC when the sampling is done. In the proposed implementation, 1,048,576 samples can be collected at rates up to 1.3 kHz. Table A-1 provides a brief overview about the key stats of two proposed sampling rates. Given that, the payload length is 60 or 110 bytes while sampling the channel, both sampling rates prove to give us valid results. The payloads correspond to 2.5 ms and 4 ms channel occupation.

Sampling frequency	**1.0 kHz**	**1.3 kHz**
Test runtime	17:30 min	13:06 min
Sample duration	8 Symbols / 125 µs	8 Symbols / 125 µs
Inter sample time	1000 µs / 875 µs	750 µs / 525 µs

Table A-1 RSSI sampling specifications

The collected data gets post processed in Matlab, to produce a PDF of the whitespace durations. First, each sample is determined as a blackspace or a whitespace by comparing the

RSSI value to a reference threshold. Then the time between each blackspace is being calculated. Finally, the PDF and CDF of the whitespace are plotted.

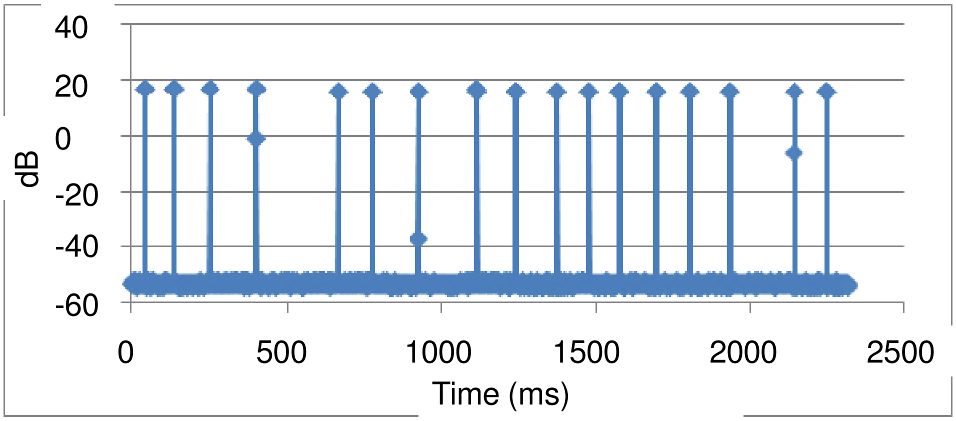


Figure A-17: RSSI Measurement example

An example of a whitespace trace is shown in Figure A-17. The peaks at +20 dB represent the measured RSSI values for transmitted packets while the lower values at around -50 dB represent the noise floor. Note that the values of the samples are misleading as the register values offered by the Mote have a linear offset. Figure 36 shows the relation between the RSSI-register value and the RF Level on the channel. The RSSI offset is approximately -45 dB, which was found empirically during the development of the system [101]. For the whitespace calculations, a comparison threshold of -40 dB has lead to a reliable detection of packets without interpreting the noise floor or interference as packets.

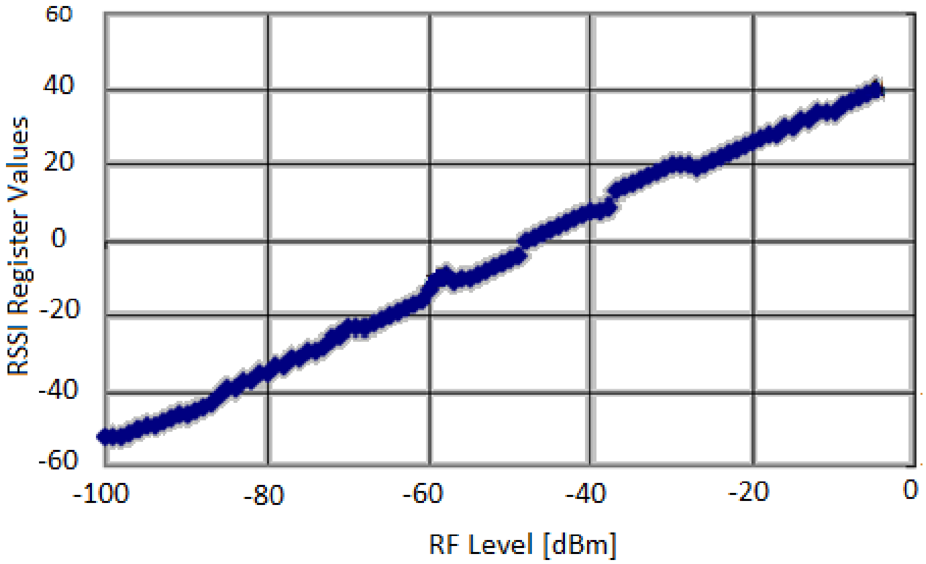


Figure A-18: RSSI register values and actual RF level [101]

A.3.2 Evaluation of the RSSI Sampling

During the evaluation of the RSSI-sampling some minor flaws were detected. Figure A-18 shows the sampling results from packets sent at a constant inter packet time (IPT) of 500 ms. As the IPT is constant, the single spike is consistent with the traffic we generated. To get the whitespace duration from an IPT we have to deduct it by the time the packet occupies the channel. In this case, the packet is 128 Bytes long and occupies the channel for 4ms. This creates a mismatch in the timing of roughly 10%.

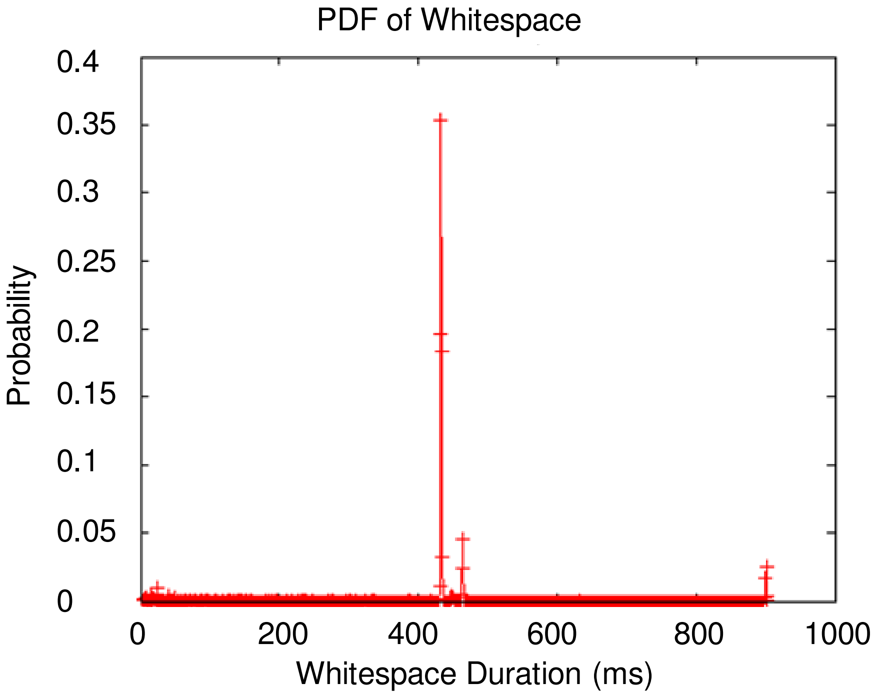


Figure A-19: RSSI measurement of 2 pps and a 1 kHz sampling rate

Figure A-19 shows the sampling result of a PU uniform traffic pattern that sends 2 packets per second with an variance of 20%. Similar to the previous example, an offset of 10% can be detected. The starting and endpoint of the big block should be from 400 to 600 minus the packet time instead of ~360ms to ~540ms. The measurement values of 800 ms to 1000 ms suggest that there are long whitespaces, though these whitespaces had never been on the channel. Knowing that these whitespaces never happened leads to the conclusion, that these whitespaces was not detected by the sensing application (a similar effect can be seen in Figure A-20 with a tiny spike at 900ms). In theory, every packet lasts 4 ms on the channel and therefore should generate at least 3 positive samples, yet we have these results.

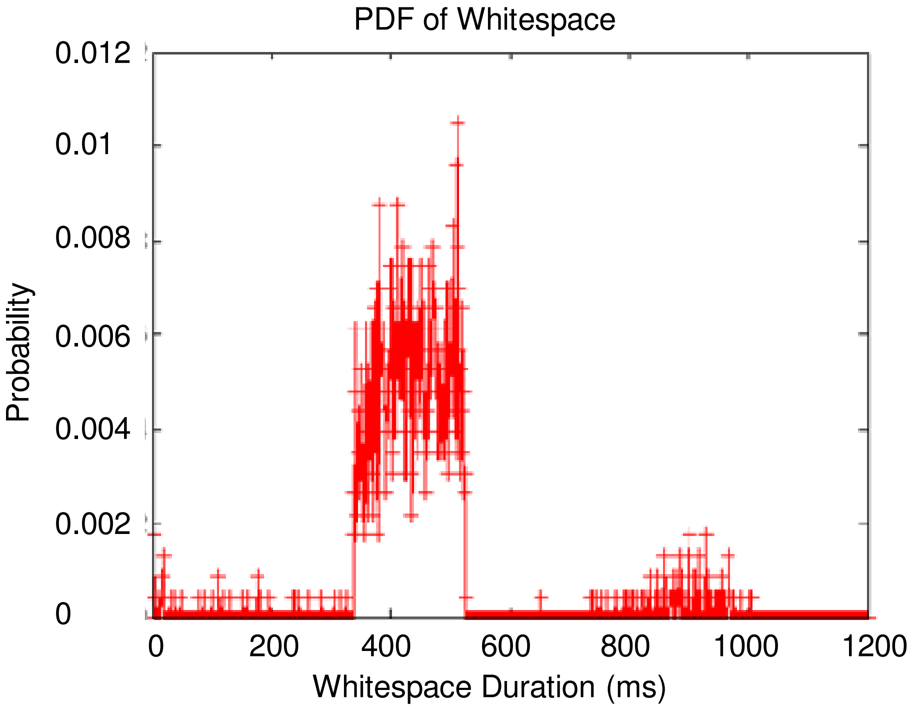


Figure A-20: RSSI measurement of 2 pps with 20% variation and a 1 kHz sampling rate

Figure A-21 through Figure A-23 shows examples of different traffic patterns, including uniform, video stream, and multi-modal that were measured by the RSSI-Sampling application.

In conclusion, the sampling application verifies the scheduling issues that were detected during the development of the PUN. Additionally extensive test were made to verify the behavior of the RSSI-Sampling application. This implementation provides reliable data about the channel occupation characteristics and the identified flaws can be taken into account while processing the data.

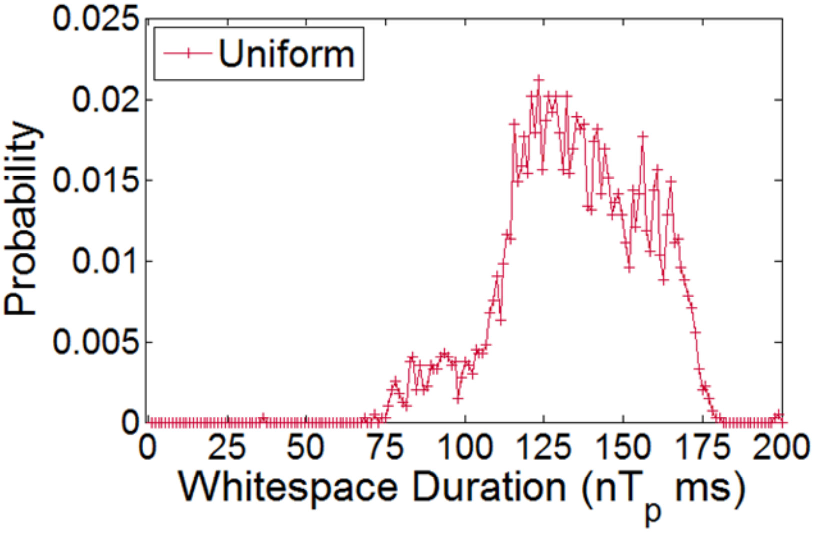


Figure A-21: RSSI measurement of Uniform distribution with 7 pps with 20% variation

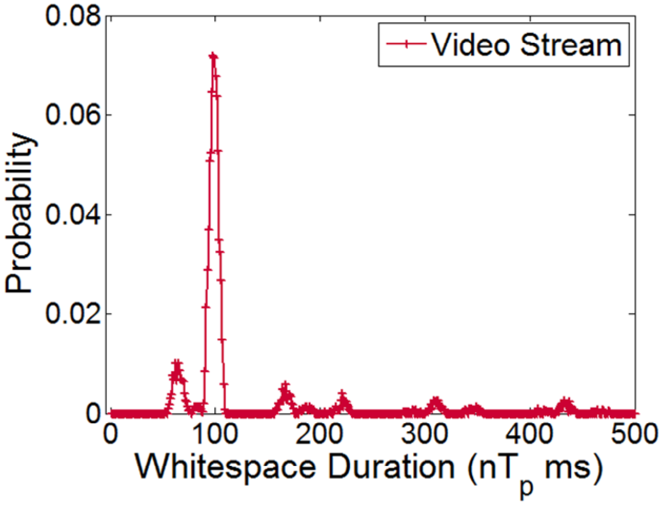


Figure A-22: RSSI measurement of Video Stream profile

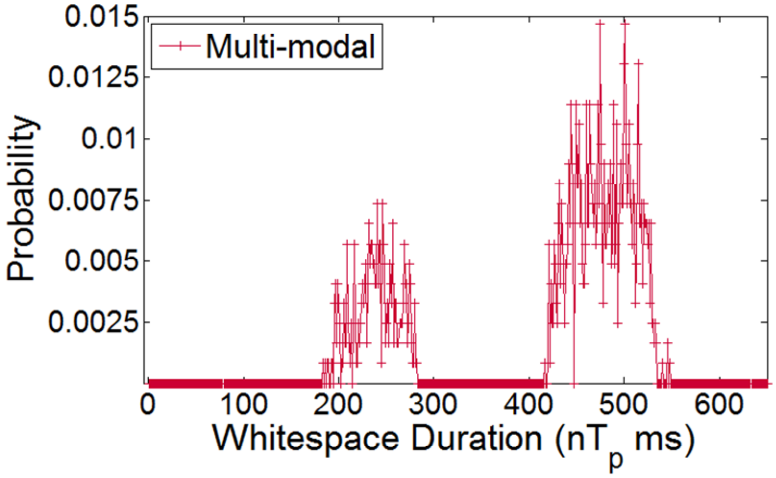


Figure A-23: RSSI measurement of Multi-model traffic profile

A.3.3 Mote Side Implementation

In this section, the SUN implementation is provided. The SUN implementation on the motes requires heavy modifications in the TinyOS platform. Flexible control of the motes for multiple access strategies requires tight synchronization between the application and network layers.

The application layer processes four main tasks:

- Control-Interface
- Packet generation
- Packet queuing
- Access strategy (Packet scheduling according to desired MAC-Layer)

The network layer of the TinyOS platform performs the actual sending of packets. This network layer is kept very simple as many MAC layer functionalities were moved up to the application layer within the mote.

Figure A-24 shows the overall information flow in the mote. The packet generator schedules packets into the queue. Then the access strategy takes the messages from the queue and sends it. All three steps are configured and controlled by the mote control interface, which communicates with the Java Control Interface on the PC. Observe that this process is different from the process used in the PUN. The purpose of this configuration is because the SUN must support and access strategy, while the PUN only needs to send messages.

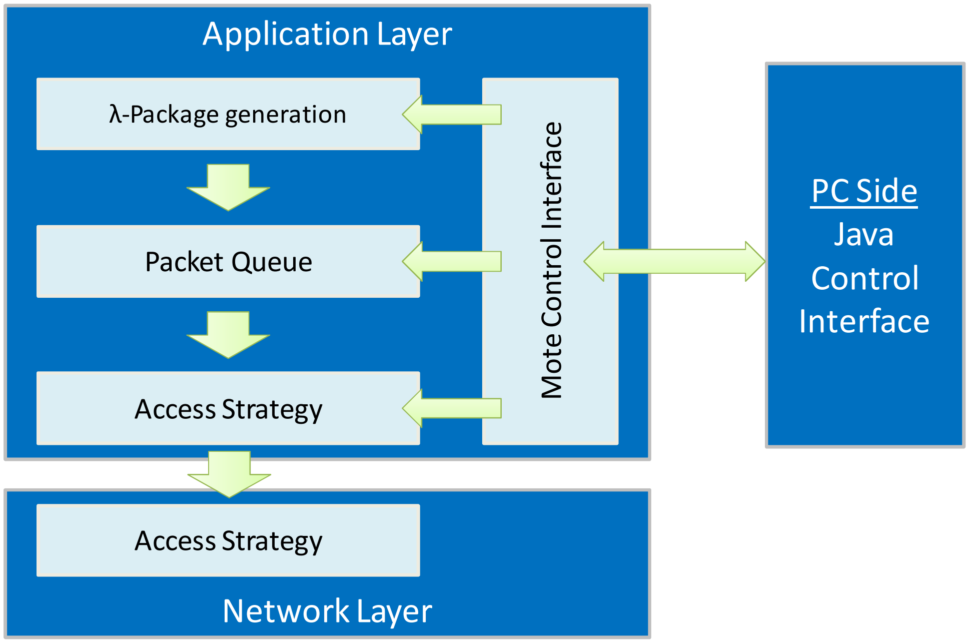
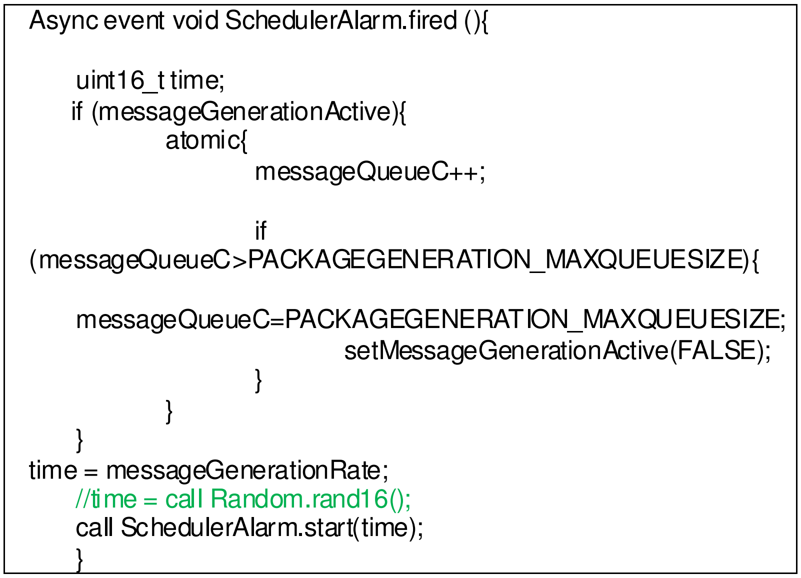


Figure A-24: Overview of SUN implementation

A.3.3.1 Application Layer

The procedures on the application layer are handled by two-timer loops. The first timer loop is the packet generation. During the startup of the mote, the SchedulerAlarm is started while messageGenerationActive is set to false. Figure A-25 shows the code. The Alarm fires although nothing happens until messageGenerationActive is set to true by the control interface.



```
Async event void SchedulerAlarm.fired (){

    uint16_t time;
    if (messageGenerationActive){
            atomic{
                    messageQueueC++;

                    if
(messageQueueC>PACKAGEGENERATION_MAXQUEUESIZE){

    messageQueueC=PACKAGEGENERATION_MAXQUEUESIZE;
                            setMessageGenerationActive(FALSE);
                    }
            }
    }
time = messageGenerationRate;
    //time = call Random.rand16();
    call SchedulerAlarm.start(time);
    }
```

Figure A-25: Pseudo code of packet generation

Once messageGenerationActive is set, the program starts adding messages to the packet queue. Unlike regular queues, this queue does not store actual messages, but the number of messages in the queue. This slim implementation is sufficient, as actual data is not transmitted. In case of too many packets are being scheduled, there is a queue overflow check comparing the message queue messageQueueC to a maximum queue size PACKETGENERATION_MAXQUEUESIZE.

The queue, which is filled by the packet generator, is the common interface between the packet generation and the access strategy that is being tested. The queue is implemented as a simple variable integer, representing the number of messages in the queue. Each time the mote sends a message it decreases the message queue counter. The second timer loop takes care of the scheduling of the message on the channel. The timer loop continuously checks whether there are messages in the queue that should be sent. Two core principles handle the behavior of the mote's access strategy. One is the timer mentioned and the other is the m_state variable. The m_state variable defines what has to be done when the second timer is fired.

Table A-2 shows an overview of the states and the access strategies. Currently the Arbitrary InterFrame Spacing (AIFS) and the CSTS protocols are implemented. The state of the mote is also the interface for the control interface. By changing the state, the desired access strategy can be activated. The actual change does not happen immediately, but on the next call of the handleStates procedure.

Access strategy	**States**
Idle, do not Access	S_READY
AIFS	S_SEND_EPSILON
CSTS	S_SEND_MU, S_SEND_J, S_SENSE_END_OF_WHITESPACE_HISTORY_ONE

Table A-2: Access strategies and states

A.3.3.1.1 AIFS Access Strategy

The Arbitrary Inter-Frame Spacing (AIFS) access strategy is inspired by the 802.11e prioritization scheme [18]. The prioritization of the traffic is implemented through using different AIFS. In the implementation, the waiting period is called epsilon and it is assumed that there is always a message to send. If there is no message to send, the mote waits for a short period and check again if a message has been generated. Before the sending of each message, the mote checks whether the channel is free for a period of epsilon. As soon as this occurs, it transmits the message.

A.3.3.1.2 CSTS Access Strategy

The CSTS access strategy accesses the channel according to the primary user traffic pattern. Therefore, the access strategy starts when a whitespace is detected. Figure A-26 shows a flow chart of CSTS with a filled queue. The code also checks if there is a message to send before each sending command.

In CSTS, at the start of a whitespace, transmissions are delayed for a μ period. The duration of μ is based on the statistics of the channel. After the channel is free for a μ duration the first packet is sent. Afterwards, the next packet is sent if the duration in the whitespace, *t* is smaller than t_{max}. t_{max} is calculated using the process as J_{max} detailed in Chapter 4 and then converted into packet numbers. Therefore, the actual check is done against the number of packets that have been sent into that whitespace rather than comparing actual time values. Before the packet is sent, the mote checks whether the channel is free for an epsilon period. Once this message is sent the program returns to the t<tmax check as shown in Figure A-26.

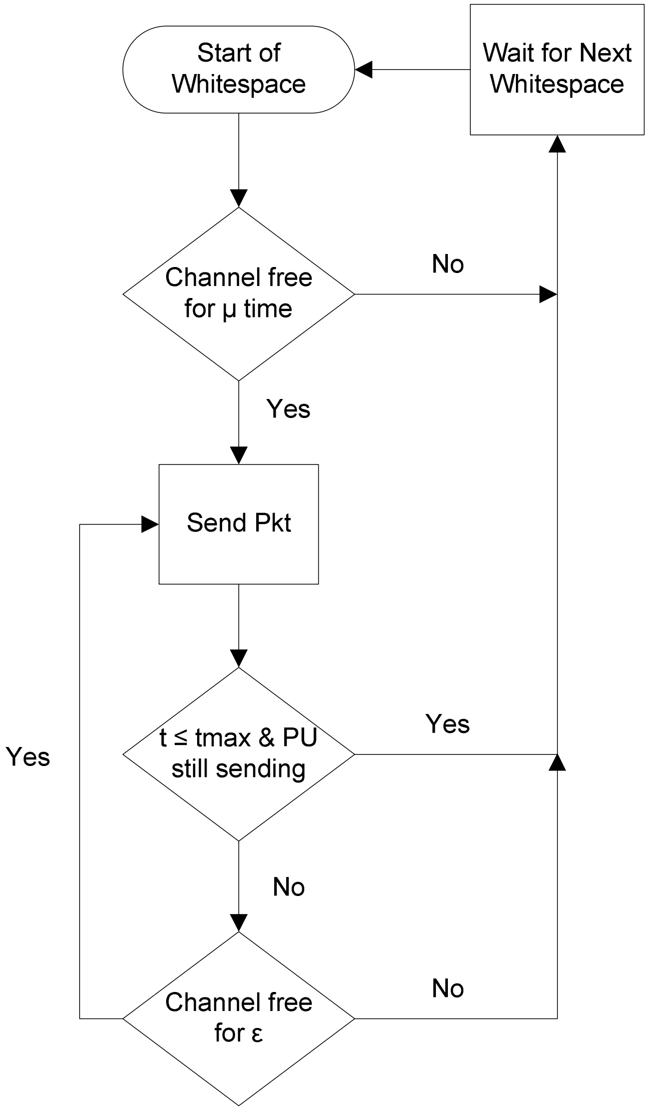


Figure A-26: Flow chart of CSTS implementation

Both the AIFS and CSTS access strategies require sensing of the channel to detect when it is free for an μ or epsilon period. The implementation of the waiting for the beginning of the next whitespace is done by a procedure, which is sampling the Clear Channel Assessment pin of the CC2420 chip. If the CCA-pin indicates that the channel is empty, the mote continues waiting until the PU returns. If the CCA-pin indicates that, the channel is busy and has indicated that it was busy for the last X samples, the primary user is assumed to have returned. Therefore, the beginning of a new whitespace is expected. The requirement for having multiple positive samples is because the CCA-pin detects any energy on the channel and interference shall not trigger the Secondary User to send his messages. These functions are implemented in the Network Layer and is described in the next section.

A.3.3.1.3 DSTS Access Strategy

The DSTS access strategy is able to access more complex whitespace traces as discussed in Chapter 7:. Similar to the CSTS access strategy, DSTS starts at the beginning of a whitespace. Since time is divided into timeslots, DSTS first checks if the current whitespace-slot is a transmit slot based on the Transmission Bitmap Vector (TBV). The mote calculates whether to send in the current timeslot by accessing file that contains the TBV assignments for each slot. If the current slot is not used for sending a packet, the mote waits for the next timeslot. After the program has sent a packet or not, it monitors the channel in a similar fashion as the sensing for the end of a whitespace protocol described above. The duration of a timeslot can be set on runtime, though a slot time of 20ms was used for all test runs as it works reliably and is sufficiently slow.

If the primary user returns to the channel, the mote waits for the beginning of the next whitespace. Otherwise, it checks if the current timeslot is larger than the maximum timeslot. Figure A-27 shows the flowchart of the DSTS implementation.

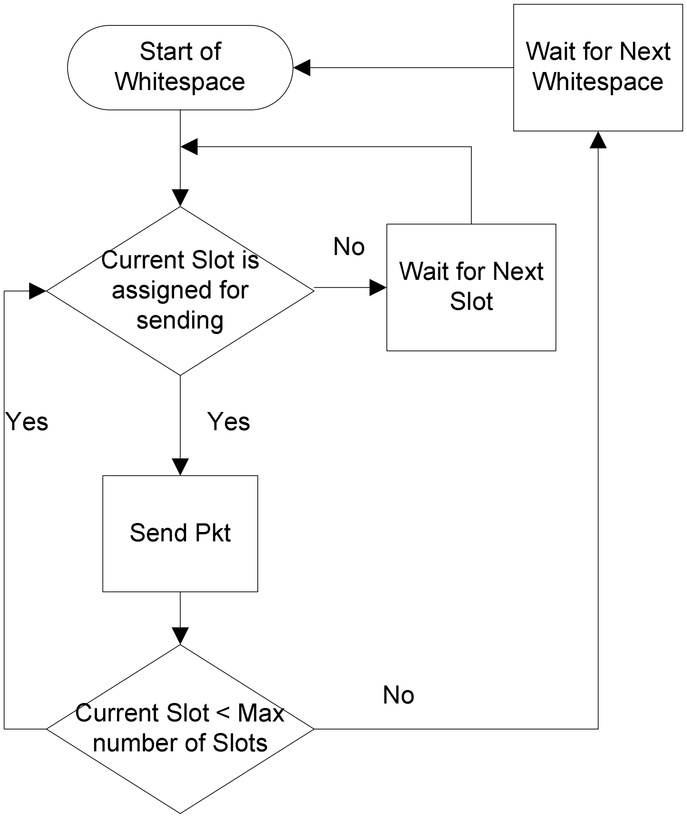


Figure A-27: Flow chart of DSTS implementation

A.3.3.2 Network Layer

The modifications to the Network layer are implemented on the two lowest layers on top of the Hardware Abstraction Architecture (HAA). The application layer sends the Network Layer arguments as to how it should process a message. Network access has been simplified from its original version to grant the upper layers as fast access to the channel as possible. The original non-persistent CSMA-CA network access has been deactivated and two CSMA algorithms have been implemented. The original CSMA-CA code was the original file for the network access. Many parts of the original code have been kept, especially the parts that handle concurrency, write into the CC2420 mote's registers and the mote's HAA interfaces.

The first CSMA version is used by the μ and epsilon packets of the previously described access strategies. Initially, it writes the message on to the mote and prepares it for sending. After the message is on the mote and ready to send it samples the channel to ensure, that it is free for a certain amount of time (epsilon or μ). If one of samples returns busy, the counter is reset. This procedure continues until the message is sent on the channel by strobing a pin connected to the CC2420.

The j-packets of CSTS also use the second version of the CSMA algorithm. It is identical to the first one except for the sampling procedure. If a sample returns busy the sending process is stopped and a fail signal is returned to the upper layers to signal that the primary user has returned.

A.3.4 PC Side Implementation

The destination SU mote receives all messages from the PU and the SU motes. All messages are then forwarded to the PC. A modified version of the BaseStation application contributed to the TinyOS is used [102]. The application has been modified to have a longer queue and to support a longer payload. The reason for forwarding all messages instead of just SU messages is to create a full log of all messages. This log is maintained by the Java control interface described below along with the computation of the packets loss and the SU throughput.

A.3.4.1 SU Control Interface

The Secondary User Control interface provides a convenient way of scheduling, measuring and reporting test runs of the access strategies. Figure A-28 shows the Secondary User Control Interface. In the top left corner, the buttons are the buttons to control the access strategy that is used by the mote. Right below that, the report controls are grouped. On the top right corner is a text field, which displays the report. In the bottom left corner are the control buttons for the packet scheduling and in the bottom right corner are the controls for the access strategy settings.

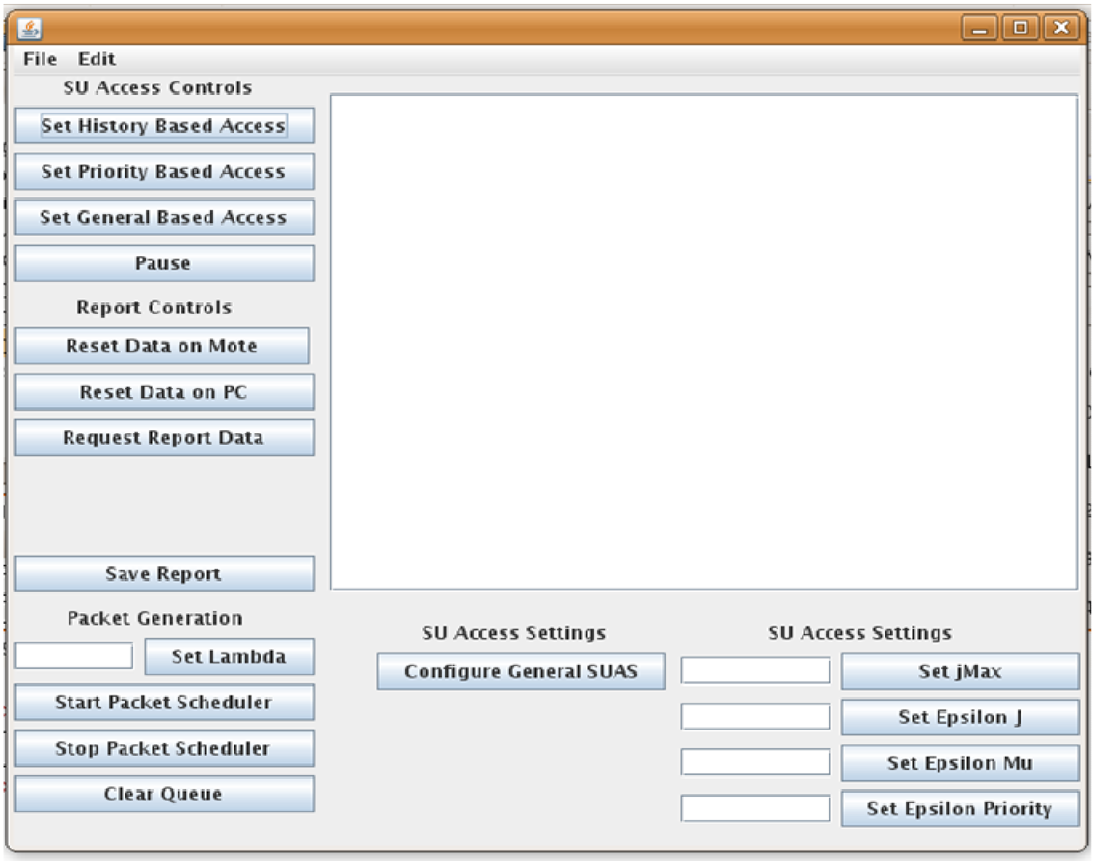


Figure A-28: Secondary user control interface

All command buttons, which changes the behavior of the mote, send a command message over the serial port to the SUTX mote. The changes is set on the mote runtime as soon as the message is received. The access strategy controls send a command message to change the m_state variable of the mote. The change of m_state forces the mote to go into the loop of the desired access strategy. The commands for the packet scheduling accesses the loop timing of the access strategy and the Boolean value, which decides whether packets are being scheduled or not. The controls on the bottom allow changing the parameters of the access strategies. The button Set Jmax is the corresponding variable for the calculated time t_{max} in subsequently sent packets.

The report functions are only processed on the mote. A click on the “Request Report” button generates a full report from the stored data and display it in the textbox. Indicators like SU throughput, PU throughput, scheduled SU messages and a full log of all messages sent during the experiments are part of this report. The full report in the window and manually added comments can be saved to a file by clicking the Save report button.

A.3.4.2 Data Collection

The data collection is implemented by directly accessing the packet source. The SURX forwards all messages to the PC and the PC stores all these messages are in an ArrayList during an experiment. Every time the Java interface receives a message, it updates the internal counters for the measured data.

The application distinguishes between Primary User Network and Secondary User Network by their AM_TYPE. The measured Secondary User throughput is calculated by counting the messages and analyzing the messageDispatchCounter. This counter increments for each message that is scheduled to be sent by the SUTX. The number of missing packages is derived from missing “increments” in the messageDispatchCounter. After each experiment, the results can be validated by comparing the number of received and missing packages to the number of packets that had been scheduled by the traffic generator.

Appendix B: LIST OF PUBLICATIONS

Peer Reviewed Journals:

1. **A. Plummer**, M. Taghizadeh, and S. Biswas, *"Measurement based Bandwidth Scavenging in Wireless Networks"*, In Press, IEEE Transactions on Mobile Computing, 2011.

2. **A. Plummer**, and S. Biswas, *"Distributed Spectrum Assignment for Cognitive Networks with Heterogeneous Spectrum Opportunities"*, In Press, Wiley Journal of Wireless Communications and Mobile Computing, 2011.

3. **A. Plummer**, M. Taghizadeh, and S. Biswas, *"Model Based Bandwidth Scavenging for Device Coexistence in Wireless LANs"*, Submitted, Elsevier Pervasive and Mobile Computing Journal, Invited based on the paper presented in ICDCN 2011.

4. **A. Plummer**, M. Taghizadeh and S. Biswas, *"Supporting Prioritized Device Coexistence in the Presence of Multiple Secondary Users"*, under preparation, IEEE Communications.

Peer Reviewed Book Chapter:

5. **A. Plummer**, M. Taghizadeh, and S. Biswas, *"Traffic Protection in Heterogeneous Sensor Networks"*, In Press, Wireless Multi-Access Environments and Quality of Service Provisioning: Solutions and Application. IGI Global.

Peer Reviewed Conferences:

6. **A. Plummer**, M. Taghizadeh, and S. Biswas, *"Model Based Bandwidth Scavenging for Device Coexistence in Wireless LANs"*, In proceedings of the 11th International Conference on Distributed Computing and Networking (ICDCN), January 2011, Kolkata, India.

7. **A. Plummer**, M. Taghizadeh and S. Biswas, *"Statistical Bandwidth Scavenging for Prioritized Device Coexistence"*, In proceedings of the 29th IEEE International Performance Computing and Communications Conference (IPCCC), December 2010, Albuquerque, New Mexico.

8. **A. Plummer** M. Schmidt, M. Taghizadeh, and S. Biswas, *"Traffic Protection via Bandwidth Scavenging in Heterogeneous Sensor Networks"*, In proceedings of the 53rd IEEE Global Communications Conference (Globecom), December 2010, Miami, Florida.

9. M. Taghizadeh, **A. Plummer**, and S. Biswas, *"Towards Optimal Cooperative Caching in Social Wireless Networks"*, In proceedings of the 53rd IEEE Global Communications Conference (Globecom), December 2010, Miami, Florida.

10. M. Taghizadeh, **A. Plummer**, and S. Biswas, *"Cooperative Caching for Improving Availability in Social Wireless Networks"*, In proceedings of the 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), November 2010, San Francisco, California.

11. **A. Plummer**, M. Taghizadeh, and S. Biswas, *"Measurement based Capacity Scavenging via Whitespace Modeling in Wireless Networks"* In proceedings of the 52nd IEEE Global Communications Conference (Globecom), December 2009, Honolulu, Hawaii, (Won Best Paper Award).

12. **A. Plummer**, and S. Biswas, *"Capacity Scavenging in Wireless Networks: A Comparative Study"*, In proceedings of the 28th IEEE Military Communications Conference (MILCOM), October 2009, Boston, Massachusetts.

13. M. Quwaider, **A. Plummer**, J. Rao, M. Taghizadeh, and S. Biswas, *"Real-time Posture Detection using Body Area Sensor Networks"*, In proceedings of the 13th International Symposium on Wearable Computers (ISWC), September 2009, Linz, Austria.

14. **A. Plummer**, T. Wu, and S. Biswas, *"A Localized and Distributed Channel Assignment Framework for Cognitive Radio Networks",* In proceedings of the ACM International Workshop on Cognitive Wireless Networks (CWNets), August 2007, Vancouver, Canada.

15. **A. Plummer**, T. Wu, and S. Biswas, *"A Cognitive Spectrum Assignment Protocol using Distributed Conflict Graph Construction"*, In proceedings of the 26th IEEE Military Communications Conference (MILCOM), October 2007, Orlando, Florida.

16. Q. Huo, S. Biswas and **A. Plummer**, *"Towards a Pulse Switching Protocol for Event and Target Tracking using Ultra Wide Band Impulse Radio"*, under review, 8th IEEE Sensor, Mesh and Ad Hoc Communications and Networks (SECON), June 2011, Salt Lake City, Utah.

BIBLIOGRAPHY

Bibliography

[1] S. Geirhofer, L. Tong, and B. M. Sadler, "Dynamic Spectrum Access in WLAN Channels: Empirical Model and Its Stochastic Analysis," in *ACM TAPAS*, 2006.

[2] V. R. Petty, R. Rajbanshi, D. Datla, W. Frederick, and et al., "Feasibility of Dynamic Spectrum Access in Underutilized Television Bands," in *Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[3] I. F. Akyildiz, W. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Computer Networks Journal (Elsevier),* 2006.

[4] N. Golmie, N. Chevrollier, and O. Rebala, "Bluetooth and WLAN Coexistence: Challenges and Solutions," *IEEE Trans. Wireless Communication,* vol. 10, pp. 22-29, Dec. 2003.

[5] C. F. Chiasserini and R. R. Rao, "Coexistence mechanisms for interference mitigation between IEEE 802.11 WLANs and Bluetooth," in *INFOCOM*, 2002.

[6] S. Geirhofer, L. Tong, and B. M. Sadler, "Cognitive Medium Access: Constraining Interference Based on Experimental Models," *IEEE Jrnl on Selected Areas in Communications,* vol. 26, no. 1, Jan. 2008.

[7] C. F. Chiasserini and R. R. Rao, "Coexistence mechanisms for interference mitigation in the 2.4-GHz ISM band " *IEEE Transactions on Wireless Communications,* vol. 2, no. 5, 2003.

[8] L. Jingli, L. Xiangqian, and A. Swami, "Collision analysis for coexistence of multiple bluetooth piconets and WLAN with dual channel transmission," *IEEE Transactions on Communications,* vol. 57, pp. 1129-1139, 2009.

[9] I. Stojmenovic, S. Giordano, M. Conti, and S. Basagni, "Mobile Ad Hoc Networking," IEEE, 2005, pp. 175 -203.

[10] J. Mietzner, L. Lampe, and R. Schober, "Distributed transmit power allocation for multihop cognitive-radio systems," *IEEE Transactions on Wireless Communications,* vol. 8, pp. 5187-5201, 2009.

[11] R. Ahuja, R. Corke, and A. Bok, "Cognitive Radio System using IEEE 802.11a over UHF TVWS," in *Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2008, pp. 1-9.

[12] Y. Chen, Z. Qing, and A. Swami, "Joint Design and Separation Principle for Opportunistic Spectrum Access in the Presence of Sensing Errors," *IEEE Transactions on Information Theory,* vol. 54, no. 5, May 2008.

[13] S. Huang, X. Liu, and Z. Ding, "On Optimal Sensing and Transmission Strategies for Dynamic Spectrum Access," in *New Frontiers in Dynamic Spectrum Access Networks. DySPAN 2008.*, Oct. 2008.

[14] S. Huang, X. Liu, and Z. Ding, "Optimal Transmission Strategies for Dynamic Spectrum Access in Cognitive Radio Networks," *IEEE Transactions on Mobile Computing,* vol. 8, pp. 1636-1648, 2009.

[15] B. Wang, Z. Ji, and K. J. R. Liu, "Primary-Prioritized Markov Approach for Dynamic Spectrum Access," in *Dynamic Spectrum Access Networks (DySPAN)*, 2007, pp. 507-515.

[16] S. Geirhofer, L. Tong, and B. M. Sadler, "Dynamic Spectrum Access in WLAN Channels: Empirical Model and Its Stochastic Analysis," in *ACM Workshop on Technology and Policy for Accessing Spectrum (TAPAS)*, August 2006.

[17] S. Geirhofer, T. Lang, and B. M. Sadler, "A Measurement-Based Model for Dynamic Spectrum Access in WLAN Channels," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-7.

[18] S. Mangold, C. Sunghyun, G. R. Hiertz, O. Klein, and B. Walke, "Analysis of IEEE 802.11e for QoS support in wireless LANs," *Wireless Communications, IEEE,* vol. 10, pp. 40-50, 2003.

[19] M. Mishra and A. Sahoo, "A Contention Window Based Differentiation Mechanism for providing QoS in Wireless LANs," in *9th International Conference on Information Technology, (ICIT '06)*, 2006, pp. 72-76.

[20] C. R. Stevenson, W. K. C. Wireless, G. Chouinard, W. Hu, S. J. Shellhammer, W. Caldwell, and F. T. Group, "IEEE 802.22: The first cognitive radio wireless regional area network standard," *IEEE Communications Magazine,* vol. Vol. 47, 1, , 2009.

[21] L. Hongjun, L. Xun, J. Shen, and M. Hongxu, "A Fair Multi-priority MAC Protocol Design of Wireless Sensor Networks," in *Networks Security, Wireless Communications and Trusted Computing, (NSWCTC '09)*, 2009, pp. 455-458.

[22] P. Lei, W. Hongyi, and T. Nian-Feng, "An Efficient and Scalable Prioritized MAC Protocol (PMAC) for Backbone Communication in Wireless Sensor Networks," in *Sensor Technologies and Applications (SENSORCOMM '09.)*, 2009, pp. 508-513.

[23] E. Jung and L. Xin, "Opportunistic Spectrum Access in Heterogeneous User Environments," in *New Frontiers in Dynamic Spectrum Access Networks. DySPAN 2008.*, Oct. 2008.

[24] I. Stanojev, O. Simeone, Y. Bar-Ness, and T. Yu, "Spectrum Leasing via Distributed Cooperation in Cognitive Radio," in *IEEE International Conference on Communications. ICC '08.* , May 2008.

[25] A. O. Ercan, L. Jiwoong, S. Pollin, and J. M. Rabaey, "A Revenue Enhancing Stackelberg Game for Owners in Opportunistic Spectrum Access," in *New Frontiers in Dynamic Spectrum Access Networks, (DySPAN 2008)*, Oct. 2008.

[26] Z. Qu, Z. Qin, J. Wang, L. Luo, and Z. Wei, "A cooperative game theory approach to resource allocation in cognitive radio networks," in *Conference on Information Management and Engineering (ICIME)*, 2010.

[27] L. Yi-Bing, Y. Rui, and Y. Fang, "Non-cooperative spectrum allocation based on game theory in cognitive radio networks," in *Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, 2010, pp. 1134-1137.

[28] E. Del Re, G. Gorni, L. Ronga, and R. Suffritti, "A power allocation strategy using Game Theory in Cognitive Radio networks," in *Game Theory for Networks*, 2009, pp. 117-123.

[29] Y. B. Reddy and C. Bullmaster, "Application of Game Theory for Cross-layer Design in Cognitive Wireless Networks," in *Information Technology: New Generations*, 2009, pp. 510-515.

[30] L. Yun, P. Qi-Cong, S. Huai-Zong, C. Xing-Feng, and W. Ling, "Power control algorithm based on game theory in cognitive radio networks," in *Apperceiving Computing and Intelligence Analysis (ICACIA)*, 2010, pp. 164-168.

[31] C.-g. Yang, J.-d. Li, and W.-y. Li, "Joint rate and power control based on game theory in cognitive radio networks," in *Communications and Networking in China (ChinaCOM)*, 2009, pp. 1-5.

[32] L. You-En, L. Kun-Hsing, and H. Hung-Yun, "Design of Power Control Protocols for Spectrum Sharing in Cognitive Radio Networks: A Game-Theoretic Perspective," in *International Conference on Communications*, 2010, pp. 1-6.

[33] Y. Ge, Y. Sun, S. Lu, and E. Dutkiewicz, "ADSD: An Automatic Distributed Spectrum Decision method in Cognitive Radio networks," in *First International Conference on Future Information Networks*, 2009.

[34] D. Teo, K. Zhong, and B. C. Ng, "An Iterative Threshold Selection Algorithm for Cooperative Sensing in a Cognitive Radio Network," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-8.

[35] R. Urgaonkar, S. Member, M. J. Neely, and S. Member, "Opportunistic Scheduling with Reliability Guarantees in Cognitive Radio Networks," *Optimization,* vol. 8, pp. 766-777, 2009.

[36] Y. C. Liang, Y. Zeng, E. Peh, and A. T. Hoang, "Sensing-Throughput Tradeoff for Cognitive Radio Networks," in *IEEE International Conference on Communications*, 2007, pp. 5330-5335.

[37] S. Huang, X. Liu, and Z. Ding, "Short Paper: On Optimal Sensing and Transmission Strategies for Dynamic Spectrum Access," in *Dynamic Spectrum Access Networks (DySPAN)*, 2008, pp. 8-12.

[38] L. Cao, H. Zheng, and S. Barbara, "Stable and Efficient Spectrum Access in Next Generation Dynamic Spectrum Networks," in *INFOCOM*, 2008, pp. 1543-1551.

[39] J. Jia, Q. Zhang, and X. Shen, "HC-MAC: A Hardware-Constrained Cognitive MAC for Efficient Spectrum Management," *IEEE Journal on Selected Areas in Communications,* vol. 26, 2008.

[40] M. Thoppian, S. Venkatesan, R. Prakash, and R. Chandrasekaran, "MAC-Layer Scheduling in Cognitive Radio based Multi-Hop Wireless Networks," in *WoWMoM '06*, 2006.

[41] S. Eljack, A. Igbal, and W. Furong, "A Multi Channel Cognitive MAC Protocol with Efficient Channel Reservation and Collision Avoidance Method," in *Conference on Multimedia Information Networking and Security (MINES)*, 2009.

[42] N. Baldo, A. Asterjadhi, and M. Zorzi, "Multi-channel medium access using a virtual network coded control channel," in *International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2009.

[43] Q. Chen, Y. Liang, M. Motani, and W. Wong, "CR-CSMA: A Random Access MAC Protocol for Cognitive Radio Networks," in *IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2009.

[44] E. C. Y. Peh, Y.-C. Liang, Y. L. Guan, and Y. Zeng, "Optimization of Cooperative Sensing in Cognitive Radio Networks: A Sensing-Throughput Tradeoff View," *IEEE Transactions on Vehicular Technology,* vol. 58, pp. 5294-5299, 2009.

[45] O. Fatemieh, R. Chandra, and C. a. Gunter, "Secure Collaborative Sensing for Crowd Sourcing Spectrum Data in White Space Networks," *Dynamic Spectrum Access Networks (DySPAN),* pp. 1-12, 2010.

[46] V. R. Petty, R. Rajbanshi, D. Datla, W. Frederick, D. Daniel, and et al., "Feasibility of Dynamic Spectrum Access in Underutilized Television Bands," in *New Frontiers in Dynamic Spectrum Access Networks. DySPAN 2007.*, Apr. 2007.

[47] G. J. Minden, J. B. Evans, L. S. Searl, D. Depardo, R. Rajbanshi, J. Guffey, T. R. Newman, V. R. Petty, F. Weidling, M. Peck, B. Cordill, and D. Datla, "An Agile Radio for Wireless Innovation," *IEEE Communications Magazine,* pp. 113-121, 2007.

[48] H. Harada, H. Fujii, T. Furuno, S. Miura, and T. Ohya, "Iterative Cyclostationarity-Based Feature Detection of Multiple Primary Signals for Spectrum Sharing Scenarios," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-8.

[49] H. Li, "Learning the Spectrum via Collaborative Filtering in Cognitive Radio Networks," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-12.

[50] J. Nasreddine, J. Riihijarvi, and P. Mahonen, "Location-Based Adaptive Detection Threshold for Dynamic Spectrum Access," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-10.

[51] L. Yu, W. Zhang, S. J. Shellhammer, and B. D. Rao, "Noncoherent Diversity Combining for Spectrum Sensing," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-7.

[52] O. In, "Primary User Behavior in Cellular Networks and Implications for Dynamic Spectrum Access," *IEEE Communications Magazine,* vol. 47, 3,, pp. 88-95, 2009.

[53] T. V. Nguyen and F. Baccelli, "A Probabilistic Model of Carrier Sensing Based Cognitive Radio," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-12.

[54] R. Tandra, A. Sahai, and V. V. Veeravalli, "Space-Time Metrics for Spectrum Sensing," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-12.

[55] H. Kim and K. G. Shin, "Fast Discovery of Spectrum Opportunities in Cognitive Radio Networks," in *New Frontiers in Dynamic Spectrum Access Networks. DySPAN 2008.*, Oct. 2008.

[56] S. Huang, X. Liu, and Z. Ding, "Opportunistic Spectrum Access in Cognitive Radio Networks," in *IEEE INFOCOM 2008*, April 2008.

[57] "IEEE std. 802.11 - 2007: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," http://standards.ieee.org/getieee802/download/802.11-2007.pdf.

[58] A. M. R. Slingerland, P. Pawelczak, R. V. Prasad, A. Lo, and R. Hekmat, "Performance of Transport Control Protocol Over Dynamic Spectrum Access Links," in *DySPAN*, 2007, pp. 486-495.

[59] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation Issues in Spectrum Sensing for Cognitive Radios," in *Conference on Signals, Systems and Computers*, Nov. 2004.

[60] P. D. Sutton, B. Ozgul, K. E. Nolan, and L. E. Doyle, "Bandwidth-Adaptive Waveforms for Dynamic Spectrum Access Networks," in *IEEE Dynamic Spectrum Access Networks (DySPAN)*, 2008, pp. 1-7.

[61] Q. Zhao, S. Geirhofer, L. Tong, and B. M. Sadler, "Optimal Spectrum Access via Periodic Channel Sensing," in *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2007.

[62] S. Geirhofer, L. Tong, and B. M. Sadler, "Cognitive Medium Access: Constraining Interference Based on Experimental Models," *IEEE Jrnl on Selected Areas in Communications,* vol. 26, no. 1, Jan. 2007.

[63] Maxim, "2.4GHz 802.11b Zero-IF Transceivers.," http://pdfserv.maxim-ic.com/en/ds/MAX2820-MAX2821.pdf.

[64] Intersil, "Direct Sequence Spread Spectrum Baseband Processor," http://www.datasheetarchive.com/datasheet-pdf/012/DSA00211974.html.

[65] Crossbox, "TelosB mote platform," http://www.xbow.com/Products/productdetails.aspx?sid=252.

[66] Z. Karakehayov, "Adaptive Medium Access for Wireless Sensor Networks," in *IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, 2007.

[67] M. Jims, N. Sarma, and S. Nandi, "Priority Based Fairness Provisioning QoS-Aware MAC Protocol," in *International Conference on Advanced Computing and Communications*, 2007, pp. 593-598.

[68] A. M. Firoze, L. Y. Ju, and L. M. Kwong, "PR-MAC A Priority Reservation MAC Protocol For Wireless Sensor Networks," in *International Conference on Electrical Engineering, (ICEE '07)*, 2007, pp. 1-6.

[69] G. J. Minden, J. B. Evans, L. Searl, D. DePardo, V. R. Petty, R. Rajbanshi, T. Newman, Q. Chen, F. Weidling, J. Guffey, D. Datla, B. Barker, M. Peck, B. Cordill, a. M. Wyglinski, and a. Agah, "KUAR: A Flexible Software-Defined Radio Development Platform," in *Dynamic Spectrum Access Networks (DySPAN)*: Ieee, 2007, pp. 428-439.

[70] M. B. H. Weiss, S. Delaere, and W. H. Lehr, "Sensing as a Service: An Exploration into Practical Implementations of DSA," in *Dynamic Spectrum Access Networks (DySPAN).* vol. 15260, 2010, pp. 1-8.

[71] H. Harada, "A small-size software defined cognitive radio prototype," in *Symposium on Personal, Indoor and Mobile Radio Communications*, 2008, pp. 1-5.

[72] H. Harada, "Software defined radio prototype toward cognitive radio communication systems," in *Dynamic Spectrum Access Networks (DySPAN)*, 2005, pp. 539-547.

[73] Z. Yan, Z. Ma, H. Cao, G. Li, and W. Wang, "Spectrum Sensing, Access and Coexistence Testbed for Cognitive Radio using USRP," in *Conference on Circuits and Systems for Communications*, 2008, pp. 270-274.

[74] R. Dhar, G. George, A. Malani, and P. Steenkiste, "Supporting Integrated MAC and PHY Software Development for the USRP SDR," in *IEEE Workshop on Networking Technologies for Software Defined Radio Networks*, 2006, pp. 68-77.

[75] Z. Miljanic, I. Seskar, K. Le, and D. Raychaudhuri, "The WINLAB Network Centric Cognitive Radio Hardware Platform, WiNC2R," in *Conference on Cognitive Radio Oriented Wireless Networks and Communications*, 2007, pp. 155-160.

[76] Z. Tong, M. S. Arifianto, and C. F. Liau, "Wireless transmission using universal software radio peripheral," in *Conference on Space Science and Communication*, 2009, pp. 19-23.

[77] Stanford SING, "SourceForge.net Repository, Retrieved October 21st, 2009, from http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/stanford-sing/," 2009.

[78] F. Österlind and A. Dunkels, "Approaching the maximum 802.15.4 multi-hop throughput," in *HotEmNets*, 2008.

[79] P. Toth, "Dynamic programming algorithms for the Zero-One Knapsack Problem," *Computing,* vol. 25, pp. 29-45, 1980.

[80] X. Liu, B. Krishnamachari, and H. Liu, "Channel Selection in Multi-Channel Opportunistic Spectrum Access Networks with Perfect Sensing," in *Dynamic Spectrum Access (DySPAN)*, 2010.

[81] K. Ghaboosi, A. B. MacKenzie, L. a. DaSilva, A. S. Abdallah, and M. Latva-Aho, "A Channel Selection Mechanism based on Incumbent Appearance Expectation for Cognitive Networks," in *IEEE Wireless Communications and Networking Conference*, 2009.

[82] C. Cordeiro and K. Challapali, "C-MAC: A Cognitive MAC Protocol for Multi-Channel Wireless Networks," in *Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[83] Y. R. Kondareddy, P. Agrawal, and K. Sivalingam, "Cognitive Radio Network setup without a Common Control Channel," in *Military Communications Conference (MILCOM)*, 2008.

[84] K.-l. A. Yau, P. Komisarczuk, and P. D. Teal, "A Context-aware and Intelligent Dynamic Channel Selection Scheme for Cognitive Radio Networks," in *Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, 2009.

[85] H. Su and X. Zhang, "CREAM-MAC: An Efficient Cognitive Radio-EnAbled Multi-ChannelMAC Protocol forWireless Networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2008.

[86] H. Su and X. Zhang, "Design and analysis of a multi-channel cognitive MAC protocol for dynamic access spectrum networks," in *Military Communications Conference (MILCOM)*, 2008.

[87] S. Zheng, Y. Liang, C. Tham, and P. Kam, "Design of MAC with cooperative spectrum sensing in ad hoc cognitive radio networks," in *IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2009.

[88] M. Wellens, J. Riihijarvi, and P. Mahonen, "Evaluation of Adaptive MAC-Layer Sensing in Realistic Spectrum Occupancy Scenarios," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010.

[89] L. Cao, L. Yang, and H. Zheng, "The Impact of Frequency-Agility on Dynamic Spectrum Sharing," in *Dynamic Spectrum Access Network (DySPAN)*, 2010.

[90] Y. Gai, B. Krishnamachari, and R. Jain, "Learning Multiuser Channel Allocations in Cognitive Radio Networks: A Combinatorial Multi-Armed Bandit Formulation," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-9.

[91] Y. Ke, "MCR-MAC: Multi-channel cognitive radio MAC protocol for cooperative incumbent system protection in wireless ad-hoc network," in *Ubiquitous and Future Networks Conference*, 2009.

[92] T. Shu, S. Cui, and M. Krunz, "Medium Access Control for Multi-Channel Parallel Transmission in Cognitive Radio Networks," in *Globecom*, 2006, pp. 1-5.

[93] F. Wang and M. Krunz, "Multi-channel spectrum-agile MAC protocol with adaptive load control," in *World of Wireless, Mobile and Multimedia Networks*, 2009.

[94] M. Sun, V. Le, and Z. Feng, "A Multi-channel Transmission Scheme in Cognitive Radio Networks," in *Communications and Networking in China (ChinaCOM)*, 2009.

[95] B. R. Tamma, N. Baldo, B. S. Manoj, and R. R. Rao, "Multi-Channel Wireless Traffic Sensing and Characterization for Cognitive Networking," in *IEEE Communications Conference*, 2009.

[96] A. K.-L. Yau, P. Komisarczuk, and P. D. Teal, "On Multi-Channel MAC Protocols in Cognitive Radio Networks," in *Australasian Telecommunication Networks and Applications Conference*, 2008.

[97] N. B. Chang and M. Liu, "Optimal Channel Probing and Transmission Scheduling for Opportunistic Spectrum Access," *IEEE/ACM Transactions on Networking,* vol. 17, pp. 1805-1818, 2009.

[98] H. M. Almasaeid and A. E. Kamal, "Receiver-Based Channel Allocation for Wireless Cognitive Radio Mesh Networks," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010.

[99] S. Eljack, B. Huang, L. Tu, and P. Zhang, "Synchronized Multi-Channel Cognitive MAC Protocol with Efficient Solutions for Second Spectrum Access," in *Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing*, 2009.

[100] J. Park, P. Pawelczak, and D. Cabric, "To Buffer or to Switch: Design of Multichannel MAC for OSA Ad Hoc Networks," in *Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1-10.

[101] "CC2420: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver ", http://focus.ti.com/docs/prod/folders/print/cc2420.html.

[102] TinyOS, "Index of /tinyos-2.x/apps/BaseStation," http://www.tinyos.net/tinyos-2.x/apps/BaseStation/.