Database-assisted end-to-end theoretical and simulation framework for cognitive radio networks

A Dissertation Presented
by
Abdulla K. Al-Ali
to
The Department of Electrical and Computer Engineering
in partial fulfilment of the requirements
for the degree of
Doctor of Philosophy
in
Computer Engineering
Northeastern University

Boston, Massachusetts

April 1, 2014
Abstract

Cognitive radios are envisaged to address the problem of spectrum scarcity by opportunistically using licensed bands, without interfering with the higher priority transmissions in them. Recent FCC rulings mandate the use of spectrum databases for obtaining the spectrum usage information, and how to integrate the database coordination with the higher layer networking protocol stack remains an open challenge. This dissertation addresses this concern through the following contributions: (a) An end-to-end transport layer called TFRC-CR is devised that uses equation based transmission rate control, and relies on database updates rather than intermediate node information. (b) A framework for vehicular cognitive radio is created that uses cross-correlation between 2G signals obtained from an Android device and signals from the TV white space to reduce the number of database queries. Moreover, this framework also involves a practical demonstration of interference alignment to optimally use the channel during the querying process. (c) An extension for the network simulator-3 that provides cognitive abilities, such as spectrum sensing, primary user detection, and spectrum hand-off to the research community which allows them to simulate these complex radios in a virtual environment.
Dedication

This thesis is dedicated to my wife, Marie. Thank you for your support and patience. For your insights and influence that paved the way for me to see and be; to see difficulty as ease, and to be who I am today.
Acknowledgements

I want to express my sincere gratitude and appreciation to my mentor and advisor, Dr. Kaushik Chowdhury, for his advice, courage and guidance in my research and for his support through the bad and good times. I know I would not be where I am without his help and continuous mentoring. I have learned a great deal working under his supervision, and for that will be eternally grateful.

To my family, who, even being so far away for so long, continue to provide their love and support. To my father and mother; you have always been a role model to me. A role model not just for my career but for the way you show and communicate your enormous capacity of love.

My lab mates; Rahman, Farima, Prusayon, Yousuf, Yufuk, Meenu, Bruna, Yifan and Jinghzi. Your assistance and friendship has meant more than what I can express. Thanks for all the discussions, help and fun time we had in the lab. I know I will miss that.

And finally, I owe a debt of gratitude to Qatar University, for their continuous support throughout my graduate career.
## Contents

1 **Introduction** .............................................................. 8  
   1.1 Cognitive radio, the motivation and definition ............................. 8  
      1.1.1 Cognitive Radio .................................................. 9  
   1.2 Main contributions of this thesis ......................................... 10  
   1.3 Cognitive radio transport layer design ................................... 11  
   1.4 Improve sensing accuracy and optimize the database access policy .... 12  
      1.4.1 Interference temperature .......................................... 12  
      1.4.2 Local and cooperative spectrum sensing .......................... 13  
      1.4.3 The FCC database ................................................ 14  
   1.5 A cognitive radio module for the network simulator 3 .................. 18  

2 **Cognitive radio transport layer design** ................................ 20  
   2.1 Problem overview .......................................................... 20  
   2.2 Related Work .................................................................. 22  
   2.3 Discussion on Rate Control in TFRC ....................................... 24  
   2.4 TFRC-CR Design Goals .................................................... 25  
      2.4.1 Low utilization of available bandwidth ............................ 26  
      2.4.2 Low transmission rate after PU departure ......................... 26  
      2.4.3 Slow recovery and ramp up ......................................... 27  
      2.4.4 Buffer overload and interference ................................... 28  
   2.5 Design and Implementation of TFRC-CR .................................. 28  
      2.5.1 TFRC-CR Spectrum Management ...................................... 29  
      2.5.2 Sending rate adaptation ............................................. 34  
   2.6 Performance evaluation ..................................................... 36  
      2.6.1 Spectrum Management ................................................. 38  
      2.6.2 TFRC sending rate adjustment ....................................... 39  
      2.6.3 High density nodes and PUs ......................................... 44  
   2.7 Conclusion .................................................................. 44
3 Accessing spectrum databases using interference alignment in vehicular cognitive radio networks

3.1 Problem overview ......................................................... 46
  3.1.1 Exploring signal correlation between 2G and TV channels .... 47
  3.1.2 Practical demonstration of interference alignment (IA) ....... 47

3.2 Related Work ............................................................... 48
  3.2.1 Cognitive radios and VANETs .................................. 48
  3.2.2 Interference Alignment and Full-duplex wireless communication . 48

3.3 Network Architecture and Overview .................................. 49

3.4 Experimental study of Correlation between 2G and TV Channels . 51
  3.4.1 Experimental Setup .............................................. 51
  3.4.2 Methodology and Observations ................................ 52
  3.4.3 Motivation for Proposed Research ................................ 53

3.5 Optimal Database Querying Strategy .................................... 54
  3.5.1 Exploiting Correlation between 2G and TV Whitespace ....... 56
  3.5.2 Database Query or Local Sensing Decision .................... 57
  3.5.3 Spectrum Updates using Interference Alignment .............. 58

3.6 Performance Evaluation .................................................. 62
  3.6.1 Improvements due to spectrum correlation exploitation ....... 63
  3.6.2 Improvements through interference alignment ............... 65

3.7 Conclusion ................................................................. 68

4 A cognitive radio module for the network simulator 3

4.1 Problem overview ......................................................... 70

4.2 Related Work and Background .......................................... 72

4.3 Ns-3 Simulator Model for CR ........................................... 73
  4.3.1 Building blocks for the simulator ................................ 74
  4.3.2 Layer-specific modifications to ns-3 ......................... 76
  4.3.3 RX interface cognitive cycle .................................... 78

4.4 Performance Evaluation .................................................. 79
  4.4.1 Validation ............................................................ 80
  4.4.2 CRE-NS3 overhead ................................................ 81
  4.4.3 CRE-NS3 vs. CRAHN ............................................. 81

4.5 Conclusion ................................................................. 82

Conclusion ................................................................. 84

Publications ................................................................. 86
List of Figures

1.1 Cognitive cycle [1] ................................................................. 9
1.2 Interference temperature threshold above the Noise Floor ............ 13

2.1 Throughput comparison between TCP and TFRC in a 3-hop chain ad-hoc network ................................................................. 21
2.2 Method of sample collection in TFRC: the first dropped packet after an RTT concludes a sample. A received packet is denoted by an arrow and a dropped packet by x ......................................................... 24
2.3 Wired vs. wireless most recent sample ($x_i$) size over a two-hop network topology ................................................................. 27
2.4 TFRC-CR finite state machine. Shaded states involve rate modifications discussed in Section 2.5.2 ..................................................... 29
2.5 Throughput (kbps) vs Time for 3-hop (long PU activity) ............... 33
2.6 Method of sample collection in TFRC-CR; instead of relying on the first dropped packet, sample collection is based on a static time interval ............ 34
2.7 Optimal multiplier M value for a 3 and 4-hop topologies ............... 36
2.8 3-Hop chain and PU region .................................................. 38
2.9 Sending rate $X_{bps}$ during PU activity region ............................ 39
2.10 Throughput comparison as PU on time increases for different number of hops ................................................................. 40
2.11 Throughput comparison as PU on time increases for 2 and 3 hop scenarios (TCP & TFRC) ..................................................... 41
2.12 The interference, throughput and goodput comparison are provided in (a), (b) and (c), respectively .................................................. 42
2.13 Throughput and queue length (for node 3) comparisons under different stress events: none, congestion, bandwidth increase and bandwidth decrease 43
2.14 Throughput as the number of streams and PUs change, respectively . 44
3.1 Network architecture with two BSs A and B that have spectrum database access, and two vehicles C and D moving from left-right in a horizontal plane ................................................................. 50
3.2 The traversed path in our measurement experiment with markers indicating locations of high RSSI correlation between TV channels and 2G spectrum (a), and explanation for the correlation (b)  ................................................................. 52
3.3 Spectrum correlation between one TV channel band and one 2G radio tower RSSI values .................................................................................................................. 54
3.4 Four different types of RSSI readings performed by sensing ................................................. 55
3.5 Decision tree of the cognitive radio node. Solid lines indicate a possible outcome while the dotted lines indicate the impossible ones ................................................. 55
3.6 Sensing vs. Querying decision flow chart .............................................................................. 59
3.7 Interference alignment with three transmitters (A, B, C) and two intended receivers (C, D) .................................................................................................................. 60
3.8 Number of spectrum-correlation points as the time of the experiment increases ..................... 63
3.9 Number of queries for various spectrum correlation percentages ........................................... 64
3.10 Accuracy of sensing .............................................................................................................. 65
3.11 Channel utilization per vehicle for no IA allowed, IA allowed and pure query scenarios in (a) light density and fast moving vehicles, (b) light density and slow moving vehicles, (c) high density and fast moving vehicles and (d) high density with slow moving vehicles ................................................................................. 66
3.12 Channel utilization per vehicle (a) density and (b) velocity changes, respectively ........................................... 68
3.13 Number of queries as the SNR increases .............................................................................. 69
4.1 Cognitive cycle [1] ............................................................................................................... 74
4.2 The main building blocks of the proposed extension ............................................................ 75
4.3 New layered architecture of an ns-3 CR node ........................................................................ 78
4.4 State machine of the RX interface ....................................................................................... 79
4.5 UDP flow’s throughput vs. time in a CR network. Gray areas indicate PU activity ................. 80
4.6 Simulated CR network topologies (a) and (b) ..................................................................... 81
4.7 Memory utilization vs. total CPU execution time (kernel + userspace) for both NS3 and CRE-NS3 .................................................................................................................. 82
4.8 Total CPU execution time (kernel + userspace) for both CRE-NS3 and CRAHN ................... 83
4.9 Memory utilization (MB) for both CRE-NS3 and CRAHN .................................................... 83

6
List of Tables

1.1 Example results from different database administrators at the Northeastern University campus .................................................. 16
2.1 Comparison between TFRC-CR and related works .......................... 24
3.1 Comparison between our architecture and related works ...................... 49
3.2 Interference alignment equations for node C .................................. 61
3.3 Interference alignment equations for node D .................................. 61
3.4 Interference alignment equations for node A .................................. 62
3.5 Interference alignment equations for node B .................................. 62
4.1 Comparison between CRE-NS3 and related works ............................ 72
Chapter 1

Introduction

1.1 Cognitive radio, the motivation and definition

Data spectrum is a finite natural resource. Ever since the introduction of wireless devices, the demand for faster data and larger bandwidth has been increasing at a rapid rate. For example, global mobile data traffic grew 70 percent in 2012 and mobile network speeds more than doubled in 2012 [2]. The number of wireless subscribers world-wide is also exploding. The number of mobile subscribers grew 45 percent from 2012 to 2013 [3]. New mobile devices and tablets with higher resolution displays are on the rise and so is the demand for high definition content streaming, which require higher wireless data rates on these congested airwaves. There is also a new generation of connected wireless services that are being increasingly adopted, such as sensors, data streaming TVs, and satellite connected phones, among others, which contribute to the spectrum scarcity problem.

There are two possible ways to handle this new demand; one is to increase the number of vacant spectrum that mobile users are allowed to use and the other is to improve the way in which we access what is currently available. In November 2002, the Federal Communications Commission (FCC), which is the government body that manages and licenses spectrum access in the United States, has published a report that aims at improving the way in which this precious resource is used [4]. One of the major findings in that report is that the effective utilization of the current spectrum access poses problem comparable to the actual physical scarcity of the spectrum. In other words, the available physical spectrum cannot be neglected; efficient ways to utilize this existing spectrum, and new principles of sharing spectrum are also needed.

Studies have revealed that while unlicensed spectrum utilization is high at most urban places, some other bands remain largely or partially unoccupied [5–7]. These temporal or spatial unoccupied channels provide for an opportunity to relief the spectrum congestion
problem. An example of such channels are the digital broadcast television spectrum. We will refer to such channels of spectrum as “white space” where Secondary Users (SUs) may use the channel if it is left vacant by the TV broadcaster, or Primary Users (PUs). This new complex nature of when and how to access these white spaces necessitates a new type of RF radio that is aware of its surroundings and can act based on observations it finds from the environment in which it is deployed in. All the above concerns motivate the use of the emerging technology called cognitive radio, which we describe next in detail.

1.1.1 Cognitive Radio

The term cognitive radio (CR) was first coined by Mitola III et. al. in 1999 [8] to describe a radio node that is able to observe the environment surrounding it, learn from the parameters obtained, plan and orient itself for possible actions, decide the set of actions to perform next and finally act on them. Later in 2005, Haykin [9] officially defined the cognitive radio as:

“Cognitive radio is an intelligent wireless communication system that is aware of its surrounding environment (i.e., outside world), and uses the methodology of understanding-by-building to learn from the environment and adapt its internal states to statistical variations in the incoming RF stimuli by making corresponding changes in certain operating parameters (e.g., transmit-power, carrier-frequency, and modulation strategy) in real-time, with
two primary objectives in mind:

- highly reliable communications whenever and wherever needed;
- efficient utilization of the radio spectrum.”

A cognitive radio adhoc network (CRAHN) [1] is a network where the nodes communicate with each other directly without a central infrastructure or base station. Each node in CRAHN performs a cognition cycle (See Figure 4.1 [1, 8, 9]). The four main cognitive management functions are:

- **Spectrum sensing**: The cognitive radio nodes sense the medium they are currently tuned to to determine whether a primary user is occupying the channel. Based on the outcomes of the sensing process, the spectrum holes (if available) are determined. When the nodes perform sensing, no transmission may be possible, and hence, sensing causes disruption to ongoing data connections.

- **Spectrum decision**: The radio must select one of the vacant spectrum holes to transmit in based on different metrics such as Quality Of Service (QoS). For example, a node may select a spectrum hole that is least occupied by other SUs.

- **Spectrum sharing**: given that multiple SUs might be concurrently occupying the same spectrum hole, Medium Access Control (MAC) implementations are necessary. Through this function block, nodes may coordinate their sensing, transmission and idle times to reduce interference with the PU and the vacant time of the channel time.

- **Spectrum mobility**: Once a node determines the next channel to transmit on, it needs to be able to handoff the current channel and switch to it. This incurs an additional spectrum switching delay.

The contributions of this thesis towards enhancing spectrum efficiency in the current unlicensed band, and identifying the licensed frequencies for opportunistic use are stated below.

### 1.2 Main contributions of this thesis

- **Chapter 2 - Cognitive radio transport layer design**: We design an equation-based transport protocol that relies on the FCC database to improve the data transmission rate, reduce congestion and enable fast response time in the specialized scenarios of multi hop CRs.
Chapter 3 - Improve sensing accuracy and optimize the database access policy:

We devise a novel method to improve the accuracy of sensing by leveraging cross correlation information between 2G cellular and TV whitespace spectrum. We also propose an efficient spectrum access policy that uses Interference Alignment and Full-Duplex technologies to shorten the time it takes to transmit CR control messages. Both these techniques are demonstrated for a practical case of vehicular CR networks.

Chapter 4 - A cognitive radio module for the network simulator 3:

We develop a comprehensive simulation framework for ns-3, which provides ready access to several advanced CR features, such spectrum sensing, spectrum decision and mobility, allows multiple link layers and interfaces and also enables studies on coexisting CR and non-CR nodes. This new module is compared against existing work in ns-2 and the code modules are released for further development by the research community.

1.3 Cognitive radio transport layer design

While the main functional blocks of spectrum sensing, switching, and sharing have experienced rapid strides over the past decade [1], work on higher layers of the protocol stack, such as the transport layer that is essential for realizing large scale practical deployments, remains in a nascent stage.

To date, the work on CR transport protocols has been based on the TCP window-behavior, where the acknowledgement packets (ACKs) sent by the received determine the state of congestion within the network [10]. This self-clocking mechanism of TCP is highly susceptible to the observed round trip time. With periodic interruptions caused by primary user appearance, or large scale bandwidth fluctuations, this mechanism by itself is unable to distinguish true congestion from PU induced spectrum changes. These works that directly adapt TCP for CR networks, rely on comprehensive information from the underlying layers, as well as the intermediate nodes of the route. While there are distinct merits in a cross-layer approach, such a design violates the traditional end-to-end paradigm associated with the transport layer.

Chapter 2 of this thesis presents a fresh perspective on the design of CR-specific protocol using an equation-based approach, wherein the concept of the congestion window in classical TCP is eliminated, and instead, an equation is devised as a function of the effective packet loss rate. This equation is not dependent on the time variance of the returning ACKs, and hence, the source transmission rate is less impacted by temporary disruptions in the flow. This problem of reliance on ACK timing is exacerbated in CR networks because nodes pause their transmission when they are engaged in sensing or channel switching.
This, in turn, results in varying round-trip time estimates (in the case of TCP) rendering the self-clocking nature ineffective. The frequency and reliance on the ACKs for window based transmissions also lead to reverse path performance impact on the forward DATA path. In TCP, this can amount to 10%-20% of the data stream rate, as demonstrated in [11].

Our aim is to design a transport protocol that is rate based (instead of window based), does not use information from underlying layers, operates through the end-to-end paradigm, and can work efficiently over multiple hops. These features allow the protocol to operate only at the source and destination. To the best of our knowledge, this is the first attempt at transport protocol design for cognitive radios with these specific goals that will help in future practical deployment.

1.4 Improve sensing accuracy and optimize the database access policy

The FCC realized the need for new spectrum for the emerging market and went through a succession of orders and policy making to regulate the way in which the white space or spectrum holes are determined and accessed by secondary users. The following gives a historical background and an overview of the different policies that span the last decade and we follow it by explaining the contribution of this thesis in more details.

1.4.1 Interference temperature

In November 2003, the FCC released an order to seek comment from researchers on the feasibility of implementing interference temperature to detect PU activity [12]. Interference temperature is a method of measuring and containing interference to primary user receivers by setting an interference temperature limit or threshold that is above the Noise Floor (See Figure 1.2). Secondary Users may transmit alongside the Primary Users as long as their transmission does not exceed this threshold at the receivers. This would mean that CR networks transmit at low power which should not pose a problem given that the distances between CRAHN nodes is shorter than that which is between PUs and SUs.

Researchers have explored the viability of such technique [13–16]. However, in May 2007, the FCC abandoned this path as a mean for reliable spectrum access policy due to “increased interference in the frequency bands where it would be used” [17]. They also sighted that the resulting performance is low compared to the amount of interference.
1.4.2 Local and cooperative spectrum sensing

In December 2003, the FCC released an order requesting a comment on the feasibility of implementing a cognitive radio network that may use cooperative and local sensing to detect PU activity and use the sensed primary channel when it is detected to be idle [18].

Since this release, researchers have proposed frameworks and algorithms that exploit temporal [19,20] and spatial [21–23] white space access opportunities using collaboration. Most of the work in these papers focused on the optimization of dividing spectrum across nodes but assumed either perfect knowledge of the spectrum opportunities available to the SUs or perfect sensing. For example, [22] proposed a collaborative scheme to optimize spectrum allocation using a color-sensitive graph coloring model to characterize the spectrum access problem and provides rules for SUs to use to avoid interference with the PU. In [24], the authors make a comparison between centralized strategy where a server allocates spectrum for CRs and a distributed system where CRs collaborate to negotiate the local spectrum assignment. In [25], a framework is devised that pits nodes that organize themselves into clusters. These clusters of nodes then choose spectrum based on the global optimal assignment.

Some work was also done on optimizing local sensing. For example, in [26], Chen et al. integrated the MAC spectrum access policy with physical layer sensing to optimize the throughput for SUs under the constraint of a collision threshold as perceived by the PU.

In 2010 [27], the FCC withdrew the sensing requirement: “However, at this juncture, we do not believe that a mandatory spectrum sensing requirement best serves the public interest. As petitioners and responding parties indicate, the geo-location and database access method and other provisions of the rules will provide adequate and reliable protec-
tion for television and low power broadcast auxiliary services, so that spectrum sensing is not necessary”. However, this order did not completely eliminate the need for spectrum sensing: “We anticipate that some form of spectrum sensing may very well be included in TVBDs (TV Band Devices) on a voluntary basis for purposes such as determining the quality of each channel relative to real and potential interference sources and enhancing spectrum sharing among TVBDs” [27].

This database mandate will be used throughout Chapters 2 and 3 of this thesis. The next section will give an overview of the database structure and of currently deployed databases.

1.4.3 The FCC database

In November 2011, the FCC has released a landmark ruling [27] that mandated the use of spectrum database access. These rulings contained specifics for mobile and stationary cognitive radio devices to follow before accessing any free white band TV spectrum in their vicinity.

The FCC ruling categorized the types of devices that may access the database into the following:

- Fixed devices
- Mode I
- Mode II
- Sensing only devices

Fixed devices

Fixed devices operate from a stationary location. Uses for such devices range from WiFi Hotspots, rural broadband distribution to cellular-style installations. These devices typically operate with high power and are installed in high terrain locations or on top of buildings. As such, the database returns different results for these types of devices given that the signals from such devices propagate over longer distances and can affect multiple channels due to the relatively high signal strength.

Portable devices

Mode I, II and Sensing only devices can be all categorized as portable devices. The discussion on the difference between these types of devices will follow later. These types of
devices operate at lower transmission power relative to fixed devices. Due to the shorter propagation paths for these devices, the spectrum database returns more channels or bands for these devices to utilize. The transmission power limit on these devices is 40 mW or 16dBm. Some examples of such devices: laptops, WiFi Access Points (AP), tablets and smartphones.

The databases

As of this writing, there are currently ten FCC database managers or administrators. In January 26, 2011, the FCC approved the following companies to manage the spectrum databases: Comsearch, Frequency Finder Inc., Google Inc., KB Enterprises LLC and LS Telcom, Key Bridge Global LLC, Neustar Inc., Spectrum Bridge Inc., Telcordia Technologies, and WSdb LLC [28]. However, in April 18, 2011, Microsoft applied to join the group [29] and got approved to do so in July 29, 2011 [30].

The database structure

To date, only four of the ten approved database managers have websites that allow for free public testing [31]. These companies are Key Bridge Global, Google Inc., Spectrum Bridge and Telcordia. The database for these sites is similar: they categorize the list of devices into Fixed, Portable and Microphones. One can put the coordinates of a specific location, select the type of device and search for the results. The results highlight the list of available channels that this specific type of device may use.

For fixed devices, the Height Above Average Terrain (HAAT) is returned along with the allowed height for this fixed device installation. In all sites, the allowed antenna height above ground level has to be less than 30 meters.

Microphones and portable devices operate in the same frequencies due to the low power allowed to transmit, which is 16 dBm. The only difference between the two is that Microphone devices are allowed to reserve a channel for exclusive use for a certain period. The databases also let portable and microphone devices that query it know which of these free channels are currently reserved by a microphone, and therefore, may not be used.

We conducted a study in March 2011 between all available public databases. The location in question was Forsyth St, in the Northeastern University campus. The results are shown in Table 1.1.
<table>
<thead>
<tr>
<th>Database</th>
<th>Fixed devices</th>
<th>Height (AGL)</th>
<th>Power</th>
<th>Portable Devices</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key bridge global [32]</td>
<td>2, 5, 6, 7</td>
<td>3 m (default)</td>
<td>N/A</td>
<td>23, 24, 26, 28, 33, 34, 44, 46, 48, 50</td>
<td>low</td>
</tr>
<tr>
<td>Google, Inc. [33]</td>
<td>2, 5, 6, 7</td>
<td>10 m (default)</td>
<td>36 dBm</td>
<td>23, 24, 26, 28, 44, 46, 48, 50</td>
<td>16 dBm</td>
</tr>
<tr>
<td>Spectrum Bridge [34].</td>
<td>2, 5, 6, 7</td>
<td>30 m</td>
<td>N/A</td>
<td>23, 24, 26, 28, 44, 46, 48, 50</td>
<td>16 dBm</td>
</tr>
<tr>
<td>Frequency Finder, Inc. [35]</td>
<td></td>
<td></td>
<td></td>
<td>No public database</td>
<td></td>
</tr>
<tr>
<td>KB Enterprises, Inc. [36]</td>
<td></td>
<td></td>
<td></td>
<td>No public database</td>
<td></td>
</tr>
<tr>
<td>NeuStar, Inc. [37]</td>
<td></td>
<td></td>
<td></td>
<td>No public database</td>
<td></td>
</tr>
<tr>
<td>Telecordia Tech [38].</td>
<td>2, 5, 6, 7</td>
<td>N/A</td>
<td>N/A</td>
<td>23, 24, 26, 28, 44, 46, 48, 50</td>
<td>16 dBm</td>
</tr>
<tr>
<td>WSdb, Inc. [39]</td>
<td></td>
<td></td>
<td></td>
<td>No public database</td>
<td></td>
</tr>
<tr>
<td>Microsoft, Inc. [40]</td>
<td></td>
<td></td>
<td></td>
<td>No public database for USA</td>
<td></td>
</tr>
<tr>
<td>Comsearch, [41]</td>
<td></td>
<td></td>
<td></td>
<td>No public database</td>
<td></td>
</tr>
</tbody>
</table>

Table 1.1: Example results from different database administrators at the Northeastern University campus
Frequency of spectrum and database access for all approved devices

The FCC ruling distinguishes between the frequency that these different types of devices (fixed, mode I, mode II, sensing) retrieve the free spectrum information from one of the approved database managers.

- **Fixed devices: Towers and Base Stations**
  
  Each fixed device needs to register its height above ground and its precise coordinates with one of the FCC database managers by a professional when installed. Once installed and registered, fixed devices must query the database once a day. The database returns the available spectrum that may be used by fixed devices for the next 48-hour window. If these devices are unable to contact the FCC database after 48 hours, they must cease operation immediately.

- **Portable devices: Mode I, II and sensing**
  
  In the FCC ruling, portable devices are further categorized into three subcategories. These are Mode I, II and sensing only devices. They all share the same maximum power limitation, which is 16 dBm, and therefore, see the same set of allowed spectrum channels from the database. However, they differ in the way they are allowed to access the free spectrum, and how they retrieve spectrum information from the database.

  **Mode II:** These types of devices are equipped with means to geolocate themselves with the help of, for example, a GPS unit. These types of devices must query the FCC database at least once every 24 hours if stationary. If the devices are mobile, then they must issue these checks every time they traverse more than 100m from the previous database retrieval location, or after 60s have elapsed.

  **Mode I:** These types of devices are not equipped with means to geolocate themselves but are able to connect to Mode II and Fixed devices via ad-hoc means. Mode I devices are required to query Mode II or Fixed devices at least once every 60 seconds regardless of position. When they request a database query, they must also submit a unique identifier that the Mode II or Fixed device will forward to the database manager for verification purposes. Mode I devices may not share this spectrum information with any other device.

  **Sensing only:** These types of devices need to pre register and be approved by the FCC before deployment. Once approved, they may sense and utilize channels that they perceive as unoccupied by any PU. Similarly to Mode I devices, sensing devices may not share their spectrum occupancy findings with any other device.
In Chapter 3 of this thesis, we show how the sensing accuracy can be optimized by leveraging cross-correlation information from different spectrum channels. We then leverage this increase of accuracy to improve on the database spectrum access policy by relying on a combination of database accesses and spectrum sensing to reduce the cost and amount of required database querying in a vehicular cognitive ad-hoc network.

In the second half of Chapter 3, we also make use of interference alignment [42]. Interference alignment is an emerging technology that is able to separate useful signals from interfering ones into orthogonal planes. This allows a combination of messages to be sent simultaneously into one time slot, leading to better utilization of the channel and power efficiency. We provide to the best of our knowledge, the first real-world application of interference alignment to allow for shortened channel utilization and better efficiency in the control channel in the cognitive vehicular network that we propose.

1.5 A cognitive radio module for the network simulator 3

Networking engineers and researchers contributing in the emerging area of cognitive radio researchers face uphill challenges when testing a new concept, given the challenging environment in which these networks operate. The CRs must opportunistically determine which licensed channels are available and make use of this spectrum before the licensed user reclaims it. Accurate protocol operation is critical, as any prolonged use of the channel raises concerns of interfering with the activities of the licensed users. This concern directly translates to meticulous testing of the protocol or networking concept in a controlled environment. Given the uphill costs of purchasing multiple software defined radios that serve as the hardware building blocks of the CR network, and the time investment is writing and deploying code in them, computer simulation often becomes the methodology of choice.

While several commercial simulators exist, such as OPNET [43], which can capably simulate heterogenous networks, our focus in this work remains on affecting improvements to open-source simulators. The main challenges in such environment are as follows:

- CR protocols are generally cross-layered. Any change in one layer, such as spectrum sensing duration at the physical layer (PHY), has direct impact on the decisions made in the upper layers of the protocol stack, requiring extensive code changes throughout the stack.

- The testing time increases dramatically with the complexity of the protocol. Owing to the large number of variables that can be controlled, identifying the dominating factor that impacts the environment to the greatest possible extent may force large number of trial runs.
The inter-dependence of the protocol layers requires network architects to implement more than one layer. For e.g., spectrum sensing at the PHY can impact the TCP throughout and its interpretation of congestion. Thus, the expertise required to simulate CR networks effectively is more than conventional wireless networks.

New functions unique to the area of CR, such as spectrum sensing, spectrum handoff and licensed or primary user (PU) detection need to be embedded in the simulator.

The focus of Chapter 4 is on providing the first cognitive radio extension to the network simulator 3 [44] or ns-3, which is a discrete event driven simulator. This simulator is poised to replace the widely popular predecessor, network simulator 2 or ns-2. Ns-3 offers several advantages over ns-2, including: (i) it has a new core written in C++, (ii) it is geared for wireless communications, (iii) it has an organized modular architecture that is expandable, (iv) it includes intuitive and extensive documentation via the html Doxygen [45] interface, and (v) the same ns-3 code can be easily adapted to work in real devices [44].
Chapter 2

Cognitive radio transport layer design

2.1 Problem overview

Cognitive radio networks enable opportunistic use of available licensed spectrum to reduce the pressure on the unlicensed ISM bands in the 2.4 GHz and 5 GHz range. While the main functional blocks of spectrum sensing, switching, and sharing have experienced rapid strides over the past decade [1], work on higher layers of the protocol stack, such as the transport layer that is essential for realizing large scale practical deployments, remains in a nascent stage.

To date, the work on CR transport protocols has been based on the TCP window behavior, where the acknowledgment packets (ACKs) sent by the receiver determine the state of congestion within the network [10, 46, 47]. This self-clocking mechanism of TCP is highly susceptible to the observed round trip time. With periodic interruptions caused by the primary user’s appearance or large scale bandwidth fluctuations, this mechanism by itself is unable to distinguish true congestion from PU induced spectrum changes. These works that directly adapt TCP for CR networks rely on comprehensive information from the underlying layers, as well as the intermediate nodes of the data path route. While there are distinct merits in a cross-layer approach, such a design violates the traditional end-to-end paradigm associated with the transport layer.

In window-based transport protocols, the problem of reliance on ACK timing is exacerbated in CR networks because nodes pause their transmission when they are engaged in sensing or channel switching. This, in turn, results in varying round-trip time estimates (in the case of TCP) rendering the self-clocking nature ineffective. The frequency and reliance on the ACKs for window based transmissions also lead to reverse path performance impact on the forward DATA path. In TCP, this can amount to 10%-20% of the data stream rate as demonstrated in [11]. This work presents a fresh perspective on the
design of CR-specific protocols using an equation-based approach, wherein the concept of the congestion window in classical TCP is eliminated, and instead, an equation is devised as a function of the effective packet loss rate. This equation is not dependent on the time variance of the returning ACKs, and hence, the source transmission rate is less impacted by temporary disruptions in the flow.

The authors of [11] also report that CSMA/CA at the link layer results in bursty end-to-end flows when coupled with TCP at the transport layer. We independently verify this in Figure 2.1 for a three node network where no congestion is introduced. The observed increase in TCP throughput may not only cause a potential adverse impact to the CR network through congestion, but also to the PUs by interfering with their packet delivery performance. Instead, the equation based TCP Friendly Rate Control (TFRC), a representative of the broader class of equation based transport protocol [48], remains stable, and in the absence of any other external stimulus, avoids the bursty transmissions seen in TCP.

Our approach towards transport protocol design for CR follows a new direction of using an equation based control, hitherto unexplored in the current literature. For this, we use the TFRC as the departure point. We not only adapt the state-action behavior of TFRC, but also modify the actual rate control equation leading to our new design for CR that we name as TFRC-CR. The main features of this new protocol are as follows:

- It allows the TCP source to integrate with designated spectrum databases, as mandated by the FCC in a recent ruling [27]. This limited (and required) interaction with the database totally removes any need of feedback from the intermediate nodes or
from the underlying layers. Thus, TFRC-CR reverts back to the classical end-to-end paradigm associated with the transport layer.

- It intelligently polls the spectrum database only when needed, by identifying a possible PU arrival event based on the observed trend in packet losses, i.e., it does not consume the back-end system resources used for interacting with the database. Current regulations from FCC specify database polling at least once every 60 seconds for Mode I devices (more on that in Section 2.5), and our aim is to increase the access frequency only when a critical need is detected.

- It enhances the speed of response by distinguishing between spectrum change and true congestion. Hence, the transmission rate is almost never penalized unless the need is justified. Likewise, the rate of increase in the transmitted segments when new spectrum becomes available is much higher than that possible in the classical window based TCP, owing to the immediate effect of the rate equation.

- It modifies the TFRC rate control equation by changing the definition of the loss-event interval. This change allows the protocol to utilize the bandwidth more efficiently by having a higher and more accurate sending rate and throughput.

The rest of this chapter is organized as follows: Section 4.2 gives the related works in the area of transport protocol design for CR. The preliminary background of TFRC and the motivation for adapting it for CRs is described in Sections 2.3 and 2.4. In Section 2.5, we describe the proposed protocol (TFRC-CR) in detail. Section 2.6 gives results from our comprehensive simulation study, and finally, we conclude our chapter in Section 3.7 with pointers to future research.

2.2 Related Work

While transport layer research in wireless networks has received considerable attention over the past decade, protocols focused specifically on CR networks are still in a nascent stage.

By minor modifications of the information contained in the feedback acknowledgments (ACKs) sent by the destination, such as by falsely advertising a receive window of 0 in Freeze TCP [49] when an impending hand-off is detected, the TCP source can be prevented from transmitting. The single end-to-end connection can be split into the wired (sender to base station or BS, when such an infrastructure support exists) and wireless (BS to the wireless node) planes, as shown in WTCP [50]. In Addition, some protocols explore tuning the sender’s transmission rate through explicit notifications (TCP EFLN) [51]
and via selective retransmissions of lost packets (TCP SACK) [52]. While each of these approaches have merits, they were not originally designed with the aim of licensed or primary user protection, sudden large-scale bandwidth fluctuations, and periodic interruptions caused by spectrum sensing and channel switching.

More specific to cognitive radio, various measurement studies have demonstrated the need for a new transport protocol for cognitive radio networks (CRNs) [46, 53, 54]. In particular, the suitability of TCP for CR networks, given its widespread use, has been explored in [46, 53–56]. The work in [46] proposes modifications to TCP and introduces three different protocols: cogTCP, cogTCPE and cogTCPW. The knowledge module common to all of the above is linked to the transport protocol that leverages information from the link and physical layer such as sensing times and estimated bandwidth. This family of protocols is designed for single hop scenarios.

Other protocols that leverage cross-layer information [57–60] also exist in literature. DSAsync [59] is a framework that modifies the base station’s link layer that connects the outer wired network to the inner cognitive radio environment. It uses information from the link layer to explicitly pause the source and destinations’ TCP streams, and is focused on a centralized (1-hop) topology between the CR nodes and the BS. Similarly to DSAsync, [60] proposes modifications to the Base Station that connects TCP over the internet to a Cognitive Radio network by modifying the Base Station by two proposed methods: a) Local loss recovery by base station and b) Split TCP connection. The CR network is a one-hop network and the proposal restricts its modifications to the base station and not the transport protocol layer in the cognitive radio nodes. TP-CRAHN uses a window based approach similar to TCP, relies on intermediate node feedback and uses a cross-layer approach in each node involved [10]. TCP-CReno [47] modifies TCP Reno so that it pauses and resumes the data when the node is performing sensing. This information is retrieved from the MAC layer.

Protocols that change lower network layers to improve throughput at the transport layers were also investigated [57, 58, 61]. In [57], Luo et al. optimize the throughput of TCP by using an algorithm to decide which channel to use. Optimization at the physical, MAC, and link layer such as sensing times, access decision, modulation and coding scheme, and link layer frame size were also done in [58]. In [61], changes to the sensing and transmission times of the CR nodes were done to improve the throughput at TCP. These frameworks do not propose a new transport protocol, but improve the throughput by changing lower network layer parameters.

Different from all of the above, our aim is to design a transport protocol that is rate based, does not use information from underlying layers, operates through the end-to-end paradigm, and can work efficiently over multiple hops. These features allow the protocol to operate only at the source and destination. To the best of our knowledge, this is the
<table>
<thead>
<tr>
<th>Protocol</th>
<th>CR</th>
<th>Eq. Based</th>
<th>Multihop</th>
<th>No cross layer</th>
<th>End-to-End</th>
<th>No mod. to low layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freeze TCP [49]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WTCP [50]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TCP EFLN [51]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TCP SACK [52]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[57–61]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TCP-Creno [47]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CogTCP/E/W [46]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TP-CRAHN [10]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TFRC-CR</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2.1: Comparison between TFRC-CR and related works

\[ x_i \]

\[ x_{i-1} \]

Figure 2.2: Method of sample collection in TFRC: the first dropped packet after an RTT concludes a sample. A received packet is denoted by an arrow and a dropped packet by x

first attempt at transport protocol design for cognitive radios with these specific goals. We are hopeful that will help in future practical deployment. Table 2.1 shows a summary and comparison between our proposed protocol and the related ones discussed above.

### 2.3 Discussion on Rate Control in TFRC

TFRC employs equation based congestion control in unicast traffic. We use this as the platform to build our protocol because it aims at providing a stable throughput, as opposed to the sudden fluctuations caused by the additive increase multiplicative decrease behavior of TCP. Given that TFRC is rate based, we also have finer grain control over the sending rate. We begin by describing the classical rate control equation in TFRC.

Let \( x_i \) be the number of consecutive packets delivered to the destination in the \( i^{th} \) sample. The counting of these packets for the calculation of \( x_i \) continues till the first loss
occurs, after the completion of the round trip time (RTT). Figure 2.2 shows this procedure for two samples $i$ and $i-1$. Next, this step is repeated $n$ times, obtaining $x_1, \ldots, x_n$ in this process. Now, the loss event rate ($p$) at the receiver is defined as the inverse of the weighted average of all $x_i$, $i = 1, \ldots, n$. TFRC, by default, averages the last 8 samples (i.e., $n = 8$) which gives an approximate snapshot of how the traffic flow changed over the last $n$ RTTs. The weights $w_i$ that are used to scale the $x_i$ values are obtained as follows:

$$w_i = \begin{cases} 1, & i < \frac{n}{2} \\ 1 - \frac{i-(\frac{n}{2}-1)}{\frac{n}{2}+1}, & \text{otherwise} \end{cases} \quad (2.1)$$

As an example, if $n = 8$, then the values of the weights are: $\{1, 1, 1, 1, 0.8, 0.6, 0.4, 0.2\}$. These weights are then used to calculate the weighted average loss interval ($I_{mean}$) of the last $n$ samples $x_1, \ldots, x_n$ as:

$$I_{mean} = \frac{\sum_{i=0}^{n} x_i \cdot w_i}{\sum_{i=0}^{n} w_i} \quad (2.2)$$

Finally, the loss event rate value ($p$) is obtained as $\frac{1}{I_{mean}}$ and informed to the sender through ACK packets. The sender changes the transmission rate through the equation (2.3) that estimates TCP’s average sending rate (to achieve fairness with TCP):

$$X_{bps} = \frac{s}{RTT \cdot \sqrt{\frac{2p}{3}} + \left\{ t_{RTO} \cdot (3\sqrt{\frac{3p}{8}} \cdot p \cdot (1 + 32p^2) \right\}}, \quad (2.3)$$

Where $X_{bps}$ is TCP’s average transmit rate in bytes per second, $s$ is the packet size in bytes, $RTT$ is the round-trip time in seconds, $p \in [0, 1]$, $t_{RTO}$ is TCP’s retransmission timeout value in seconds, and $b$ is the maximum number of packets acknowledged by a single TCP ACK. By default, $b$ is set to 1 and $t_{RTO}$ is set to $4 \times RTT$. This choice is beyond the scope of our discussion and is discussed in detail in [48].

### 2.4 TFRC-CR Design Goals

In this section, we shall identify the specific features of classical TFRC that we target in our design of the transport layer protocol for CR networks.
2.4.1 Low utilization of available bandwidth

TFRC reduces the sending rate at the source whenever packets are dropped in the sending sequence. This simplistic approach is sufficient for wired networks, where packets are only dropped due to congestion, but not in CR networks where drops can happen due to multiple factors. We explain the mechanism as follows, and identify possible ways to adapt this situation.

Equation 2.3 can be reduced to:

\[ X_{\text{bps}} = \frac{s}{\text{RTT} \cdot f(s)} \]

where

\[ f(s) = \sqrt{\frac{2p}{3}} + 12 \cdot \sqrt{\frac{3p}{8}} \cdot p \cdot (1 + 32 + p^2) \] (2.4)

When \( t_{\text{RTO}} = 4 \times \text{RTT} \) and \( b = 1 \), these are approximated to equal TCP’s sending rate [48].

From equation 2.4, the two contributing factors to the sending rate are \( \text{RTT} \) and \( p \), the loss event rate. As explained earlier \( p \) is the inverse of the weighted average of the samples \( x_i \) (see Equation 2.2). In wireless networks, the sample \( x_i \) values tend to be smaller than in wired networks. This is because TFRC assumes that dropped packets occur due to congestion only (when intermittent losses are possible owing to channel errors). As soon as a single dropped packet is encountered at the receiver after an RTT has elapsed, the sample round \( i \) is completed, the value of \( x_i \) is logged and computation for \( x_{i+1} \) begins immediately. Figure 2.2 describes this situation. The overall effect of this behavior is that for wired networks, this leads to relatively equal sample lengths, i.e., \( x_1 \approx \ldots \approx x_n \) values. However, and particularly in CR networks, the disparity in the samples \( x_i \) is wider, as PU activity, spectrum availability changes, among other factors contribute to occasional packet losses. Figure 2.3 shows this divergent behavior for wired and wireless cases. Because of this discrepancy in sample lengths in cognitive radio networks, TFRC’s sending rate is fluctuating even when no stress events are introduced in the network. Our aim is to equalize the sample lengths by introducing a time-based sample collection method. This will produce equal lengths when the network is in normal condition and will also adapt in size as the network’s available bandwidth changes.

2.4.2 Low transmission rate after PU departure

Classical TFRC is unable to use the maximum allocated bandwidth after a PU vacates the spectrum due to the transmission source’s low rate that is caused by timeout events when a CR node ceases transmission or due to interference caused by PU activity. Instead, TFRC-CR is designed to neglect the last \( n \) loss event rates after a prolonged idle state due to PU
activity. This prevents the pitfall of resuming the transmission with a false $p$ which leads to less than optimal throughput. Note that classical TFRC will recover to the maximum transmission rate after at least $n$ samples have been recorded. This takes at least $n$ RTTs to achieve.

### 2.4.3 Slow recovery and ramp up

After an extended period of packet losses that limit TFRC’s sending rate to the minimum rate, the protocol starts polling for changes in bandwidth over large intervals of time. This polling interval is a function of the current transmission rate. For example, during PU Activity, classical TFRC’s nofeedback timer expires multiple times which leads to reduction in the effective rate by half each time until the minimum rate of $\frac{s}{t_{mbi}}$ is reached. Here, $s$ is the packet size and $t_{mbi}$ is set to 64 seconds in the default implementation. This means that TFRC will poll the network once every 64 seconds, which can cause an equivalent delay for the CR network to resume transmission again after a PU departure. Thus vital spectrum opportunity is wasted in this extended downtime for the CR network. Our goal is to speed the resurgence of the data flow by integrating the transport protocol with the FCC database which can inform the sending node in advance the time at which the current PU activity ends.
2.4.4 Buffer overload and interference

TFRC may send multiple packets during the duration of the PU activity as part of its regular rate control which causes additional interference with the PU. This serious problem multiplies at the link layer, where typically the medium access control protocol attempts several rounds of transmission per packet before reaching the maximum retry limit. Moreover, the added processing tasks as well as the consumption of buffer space by these additional packets that cannot be transmitted on the same channel contribute to the overhead. While the CR network can be designed to seek out and immediately leverage alternate channels, this feature is not known at the source in advance (owing to lack of inter-layer communication). By relying on the FCC database to inform the sender of possible disruptions due to PU activity, this problem can be alleviated; the sender will pause the data flow based on that information which will lessen the stress on the intermediate nodes’ queues.

2.5 Design and Implementation of TFRC-CR

We first present the modified finite state machine for TFRC-CR which we will refer to as we describe the overall functions related to spectrum management and PU avoidance. Then, we discuss how to change the rate equation in TFRC-CR such that the connections utilize the maximum allocated bandwidth.

Note about FCC database and connectivity: In [27], the FCC has recently allowed different devices to use a centralized database to infer PU activity. Devices that are allowed to use the channels are categorized in three modes: Mode I, Mode II and sensing mode. Mode II are geolocation capable devices that are required to access the database directly at least once per day before utilizing any channel, Mode I devices needs to connect to Mode II devices directly or a fixed-based station, and are required to refresh their local database information once every minutes. Sensing mode nodes need to be certified by the FCC and can then sense and use what they perceive as vacant spectrum. In the design of TFRC-CR, we are only interested in spectrum awareness at end locations and do not involve the intermediate nodes. This still maintains the end-to-end sense of the transport protocol. In any case, we do not rely on the knowledge of specifically “where” or ”at which node” PU activity occurred in the connection, but that it occurred somewhere in the connection. Similar assumptions on rough estimations of a region of interest have been made in earlier works on routing (see reference [62] for example).
2.5.1 TFRC-CR Spectrum Management

This section covers the following key features of our proposed protocol: (i) ensuring that connection immediately becomes active after the PU leaves the impacted region, (ii) striking a balance between polling the network too frequently by the source, an action that may itself cause interference with the PU, and conversely, reacting too slowly to spectrum change, and (iii) estimating the available bandwidth as soon as possible after the PU vacates the spectrum. These changes are explained below using a finite state machine diagram, shown in Figure 2.4.

**Normal state**

This is the default state of TFRC-CR. The protocol returns to this state whenever the source infers the connection to be free of spectrum outages. We describe in detail how the equation governing the sending rate (and hence, congestion control) in this state is modified
in Section 2.5.2. Recall that when a packet is dropped, the no feedback timer expires in the absence of the ACK. From here onward, our protocol’s operation diverges from classical TFRC: To differentiate congestion from possible PU activity at this timer expiry, the source queries the FCC mandated spectrum database to check if a PU arrived on any of the feasible channels. Note that the source has no knowledge of the location of the nodes in the connection except the destination, nor the specific channels used by any of these nodes. However, a sudden arrival event of the PU (as indicated by the database) and the resulting timeout are treated as correlated events. If the database affirms the PU presence, the protocol enters into the PU detected state, and if not, the ACK loss is interpreted as a case of network congestion and is handled by the standard TFRC rate control which cuts the sending rate in half. In the normal state, the protocol continues to calculate the average ACK inter-arrival time (denoted as $I_n$), and the standard deviation of the round-trip time (denoted as $RTT_{stddev}$). These values are used in the subsequent states to influence the rate control mechanism.

Please note that our reliance on the ACK timer expiry to query the FCC database and infer PU activity here is different than congestion inference that is typically used in window-based transport protocols; TFRC-CR’s end-destination periodically generates ACKs for the sender, whether or not (i) there is true network congestion, and (ii) packet arrival to the destination. Thus, the rate control is decoupled from the frequency of returning ACKs, which is the typical method used in window-based protocols. In TFRC-CR, if a sender does not receive an ACK, then it signals a larger event, such as a total disruption on the link. In this state, we use this scenario to map a PU appearing which renders the link unusable.

**PU detected state**

This intermediate state is implemented as an additional measure to verify that the last ACK timeout in the normal state was in fact due to PU activity. Upon entering this state, the source waits for a period (the average inter-arrival time during normal state or $I_n$) for any incoming ACK, while continuously polling the spectrum database. If no subsequent ACKs are received, and the database reveals that the PU is still present, then the protocol transitions into the paused state where the source must wait for the PU activity to get completed. On the other hand, if an ACK does arrive in that time period, then the protocol returns to the normal state as this implies that the intermediate nodes are not affected by the PU activity. In such a case, the ACK timer expiry was due to random channel errors or congestion.
**Paused state**

In this state, TFRC-CR determined that the PU is present and assumed that it is responsible for disrupting the continuous data stream. The challenge now is to identify when the transmission rate can revert back to a higher value and this is obtained by polling the connection with an occasional packet. When a portion of the spectrum is occupied by a PU, the link layer algorithms on the node pair on the affected link may either pause the transmissions altogether, or immediately try and identify an alternate spectrum for that link. Note that the source has no idea which of these options is selected as no intermediate node feedback is allowed. Additionally, simple monitoring of the spectrum database, even if it indicates the presence of a PU, does not reveal any corrective action by the nodes in the connection. Thus, determining when the connection is active again is a non-trivial task.

By increasing the transmission rate too early, the source risks added interference to the PU before it vacates the spectrum. Also, by delaying the rate increase, the source is unable to efficiently use the available bandwidth of the connection if the PU vacates the spectrum earlier or the nodes in the connection transition to a new channel. We have undertaken a substantial set of simulations and empirically identify the optimal polling rate as

\[ X_{\text{bps}} = \frac{s}{RTT_{\text{avg}} + 4 \times RTT_{\text{std}}} \]  

(Section 2.3), i.e., the source will send a packet every time the Retransmission Timeout (RTO) value of TCP [63] expires whether or not the nodes have switched the channel. In comparison, TFRC reduces the rate after each ACK timer expiry in half, until it reaches a rate of \( \frac{s}{64} \) (see Section 2.4.3), which sends out a packet every 64 seconds. This leads to slow reaction to both the sudden reduction in bandwidth when the PU starts affecting the connection, and to the higher available bandwidth once the PU is out of the vicinity.

If an ACK is received in the **paused state** due to the polling packets that the sender transmitted, the protocol enters into the **resumed state**. TFRC-CR perceives this ACK as an indication that the intermediate nodes have moved to a vacant spectrum and allows the rate to adapt accordingly. If no feedback packets are received during this period, indicating that the nodes have not switched the spectrum, TFRC-CR will enter the **slow start state** immediately after the PU leaves the spectrum. The PU exit time is known by querying the aforementioned spectrum database.

**Resumed state**

During PU activity and while the protocol is in the **paused state**, the intermediate nodes may either switch to a vacant spectrum or remain in the occupied channels. If the intermediate nodes have switched spectrum and the link is no longer disrupted, the sender will
receive an ACK from the destination. TFRC-CR will then enter the *resumed state* and initiate slow-start (see *Slow start state* below for details). This is done to adjust the sending rate based on the new channel characteristics that the nodes have switched to. Please note that the rate control at this state is modified according to the discussion that will follow in Section 2.5.2.

The protocol stays in this state until the PU exits, at which time it runs an algorithm to see whether another slow-start is required. Notice that TFRC-CR does not yet return to the *normal state* because the ACK that was received in the *paused state* could be due to an intermediate node falsely misdetecting the PU presence; a realistic outcome considering the existing state-of-the-art sensing algorithms.

If the nodes never misdetect the PU presence and have not switched to a vacant spectrum, then the protocol will not enter this state because it will remain in the *paused state*. The protocol runs Algorithm 1 when the current active PU exits the vicinity. This time is scheduled based on the query results from the integrated FCC spectrum database which is known at the sender. Finally, the average ACK inter-arrival time $I_{PU}$ is calculated in the duration of this state for use in Algorithm 1.

**Slow start decision box**

The goal in this algorithm is to determine whether a slow-start is required or not. TFRC-CR slow-starts if the rate at the time of the PU exit is relatively low in comparison to the rate recorded during the last *normal state*. In other words, the ACK received during the *paused state* was a result of a sensing error and a slow-start to probe for new bandwidth is required. Otherwise, the protocol immediately returns to the *normal state* because the intermediate nodes have found a vacant spectrum and resumed transmission. The decision whether to slow-start is made based on the results obtained in Algorithm 1.

**Algorithm 1** : is slow start required

\[
\begin{align*}
& \text{let } I_n \text{ be the average inter-arrival of ACKs at the sender in the *normal state*} \nonumber \\
& \text{let } I_{PU} \text{ be the average inter-arrival of ACKs at the sender during the *paused state*} \nonumber \\
& \text{let current time } = t \nonumber \\
& \text{let } t_{PU} \text{ time last ACK received during *paused state*} \\

1: & \text{ if } I_{PU} > (2 \times I_n) \text{ OR } t - t_{PU} > (3 \times I_n) \text{ then} \\
2: & \quad \text{return } true \\
3: & \quad \text{else} \\
4: & \quad \text{return } false \\
5: & \text{end if}
\end{align*}
\]
In summary, Algorithm 1 checks if either of the following is true based on empirical observations to correctly determine whether the packet received during slow-start was in error and TFRC-CR should therefore slow-start:

- **Case I**: If the average inter-arrival of ACKs during the paused state ($I_{PU}$) is larger than twice the average inter-arrival of ACKs during normal state ($I_n$).

- **Case II**: If the time elapsed ($t - t_{PU}$) since the last ACK received during an ongoing PU activity is larger than 3 times the average inter-arrival time ($I_n$) of ACKs during normal state. This is necessary because the average ACK inter-arrival time ($I_n$) is calculated online, and if there are two consecutive ACKs that arrive due to a PU misdetection, $I_n$ will be small. The second condition is designed to catch these exceptions.

**Slow start state**

TFRC-CR enters slow-start if the rate during resumed state was slow according to Algorithm 1 or if the previous state was the paused state, i.e., no ACKs were received in the paused state. Slow-start is used to quickly probe the new vacant spectrum for the maximum available bandwidth. TFRC-CR slow-starts by resetting the weights and variables of TFRC. This is done by having the source flag the next packet as a slow-start request packet (SSREQ). When the destination receives this packet, it resets its own loss rate $p$ calculations (see Section 2.3) and sends back a slow-start acknowledgement packet (SSACK) immediately. During slow start, the nofeedback timer is set to $RTT_{avg} + 4 \times RTT_{stddev}$ [63] where $RTT_{avg}$ and $RTT_{stddev}$ are the average and standard deviation of the round-trip-time. We use this as a more accurate result than TFRC’s default static $2 \times \text{packetsize}$ / 300.

Once the SSACK packet is received at the source, TFRC-CR returns back to the normal state, thus completing the cycle.
2.5.2 Sending rate adaptation:

As explained in Section 2.3, TFRC ends the collection of each sample whenever it encounters a dropped packet. However, in wireless networks susceptible to random errors [11], this leads to sub-optimal and small sample ($x_i$) values that are far from the correct ones that allow for full utilization of the available bandwidth in the wireless network.

Due to the random nature of these dropped packets, our protocol cannot rely on them to determine the correct size of the samples. Instead, we propose to look at a time-based window for incoming packets (See Figure 2.6). This way, our protocol ends sample collection after a given period of time instead of when it encounters the first dropped packet. This method is further explained by identifying how to select the interval for collecting the samples, and how to scale the observed samples in that interval as a function of the connection length.

![Figure 2.6: Method of sample collection in TFRC-CR; instead of relying on the first dropped packet, sample collection is based on a static time interval](image)

**Collection time interval:**

The collection time interval needs to strike a balance between being prohibitively high, and therefore reacting slowly to network condition changes (e.g. bandwidth increase/decrease, PU activity, congestion), and too low such that the collected samples do not represent the correct network condition in terms of the amount of packets that arrived and what percentage of those were dropped. In our empirically obtained results for centralized networks (where the source and destination are directly connected), setting the collection time interval to 1.0 s displayed a sound balance between having a good reaction speed to condition changes and having enough packets in the samples ($x_i$) to meet the transmission rate that would fully utilize the spectrum’s available bandwidth.

Furthermore, we observed for larger connections, i.e., for 3-hops and more, the packets received in the 1.0 s duration were too few to correctly represent the correct sending rate at the sender. Simply increasing the collection time interval is not possible due to the adverse effect on the speed of the response to traffic congestion or spectrum related changes. Thus, we add an initialization phase that precedes the slow-start phase of the connection,
wherein the source-destination pair send test packets to identify a static multiplier $M$. This multiplier is a function of the length of the connection that is unknown to the source and must be empirically decided. In the actual operation, the source scales the number of correctly transmitted samples in the 1.0 s duration with this value $M$ before weighting their average and determining the effective loss rate $p$.

Initialization phase for choice of multiplier:

The choice of a multiplier is critical to have the best balance between having a very high sending rate, which can adversely lead to higher RTT -This is because the increase in queuing delay at the intermediate nodes may eventually result in higher dropped packet rates when the sending rate is significantly higher than the capacity of the connection- and conversely, having a very small multiplier value leads to low throughput and under-utilization of the available bandwidth. Equation 2.2 therefore becomes:

$$I_{\text{mean}} = \frac{\sum_{i=0}^{n} x_i \times w_i \times M}{\sum_{i=0}^{n} w_i} \quad (2.6)$$

where $M$ is the multiplier value.

Figure 2.7 shows the effect of increasing the multiplier value (The exact network parameters are discussed in details in Section 2.6). As $M$ increases, the RTT increases and eventually, the dropped packet rate increases when the buffers overflow in the intermediate nodes. For the 3-hop scenario, $\approx 190$ is the best multiplier value, and any higher leads to unnecessary increase in RTT (queuing delay), higher dropped packet rate without any significant throughput gain. Likewise, for a 4-hop topology, the correct value is $\approx 390$. Note the high dropped rate when the multiplier is very low which is due to the very low throughput.

When TFRC-CR slow starts, we find the optimum $M$ by using Algorithm 2. This algorithm increments $M$ until the change in throughput increase is less than $\%10$ (condition $\frac{TP_{\text{now}}}{TP_{\text{prev}}} > 1.1$) while making sure that the dropped rate remains below $\%10$ (condition $\text{dropped rate} < 0.10$). This will give us the highest possible throughput while maintaining a low dropped rate. The while loop (lines 5-8) are traversed once every time a new ACK is received at the source.

This optimal value of $M$ is retained for all future scaling during the protocol operation.
**Algorithm 2**: Finding optimum $M$

1: Let $TP_{now}$ be throughput at current time
2: Let $TP_{prev}$ be the previous throughput
3: Let dropped rate be the dropped packets’ rate
4: $M = 1$ and $TP_{prev} = 1$
5: while dropped rate $< 0.10$ and $\frac{TP_{now}}{TP_{prev}} > 1.1$ do
6: \hspace{1em} $M += 1$
7: \hspace{1em} $TP_{prev} = TP_{now}$
8: end while

![Optimal M](image)

**Figure 2.7**: Optimal multiplier M value for a 3 and 4-hop topologies

## 2.6 Performance evaluation

In our simulation, we use the Cognitive Radio Ad-Hoc Network framework from [53], and expand it significantly to support the transport layer operations. The revised simulator
which incorporates TFRC-CR can be downloaded from the link in [64] where instructions to compile and integrate it with existing ns-2 installations can be found. In Sections 2.5.1 and 2.5.2, we simulate TFRC-CR over a multihop chain in ns-2 as depicted in Figure 2.8. This topology is best suited to evaluate our protocol under the following different scenarios: a) single and multihop simulations where node 4 is the sink and the sending node can vary from node 1 to node 3, b) create a bottleneck at node 3 either due to spectrum bandwidth change, or c) due to congestion (when node 5 has a second active connection to node 6).

In the simulation setup, nodes ignore incoming RTS packets if they sense any PU activity within their immediate vicinity. This leads to an increase in queued packets at the node immediately preceding the nodes in the active PU region and eventually dropped due to retries or timeouts. Each node has a 0.1 probability of misdetecting the PU activity. In these cases, the nodes that misdetect, transmit data concurrently with the PU causing interference. We set our sensing period to 0.1s and transmission period to 3s [65]. The transmission rate at the link layer is set to 11 Mbps using 802.11b specifications. The nodes in this simulation do not switch to another spectrum when the PU is detected; they wait until the PU exits the spectrum to start transmitting again. This is done to clearly demonstrate how the source correctly detects, pauses and resumes the transmission. All nodes pick from 10 available spectrum bands at random at the beginning of the simulation. Each node will have two different interfaces: one for receiving packets and one for sending. The authors of classical TFRC in [48] recommend setting $b$, the number of packets that are acknowledged by a single ACK, to 1 and $n$, the number of weights to average, to 8. The authors advise against setting $n$ to a higher number owing to the slow reaction to congestion conditions that this would ensue. We follow these recommendations throughout this section.

We will first illustrate our spectrum management changes (Section 2.5.1) followed by our rate adjustment evaluation (Section 2.5.2). Lastly, we evaluate our protocol in high density node and PU scenarios in Section 2.6.3. Our simulation compares TFRC-CR against TCP and default TFRC as the baseline window and equation based transport protocols, respectively. To the best of our knowledge, there are very limited existing transport protocols specifically targeting cognitive radios (for e.g., TP-CRAHN [10], cogTCP [46] and TCP-CReno [47]). Unfortunately, comparisons with them are not viable due to (i) completely different window-based and rate-based implementations, (ii) extensive feedback from the underlying layers and intermediate nodes that is required in them, but specifically avoided in our approach, and (iii) the fact that some of proposed protocols (e.g. cogTCP [46]) are designed for a centralized wireless transmission topologies while our protocol is designed to work in multihop scenarios as well.
2.6.1 Spectrum Management

In this section, we omit our changes to the rate control in order to showcase the state machine control algorithm (Section 2.5.1). Figure 2.5 shows us the throughput difference between these two protocols in one simulation run. Areas in gray denote PU activity regions. Throughput regions $A$, $B$ and $C$ are of interest. This simulation is run over a 3-hop chain (i.e. Node 1 sends to node 4 while nodes 5 and 6 are deactivated in Figure 2.8).

- **Region A**: In this region, TFRC-CR is in the Normal state. We observe that the protocol’s throughput matches that of TFRC.

- **Region B**: We see that immediately following the PU activity region that ends at time 150 s and 200 s, TFRC-CR slow-starts immediately because it utilizes the information received from the spectrum database. Due to the rate reduction that happens in TFRC during the PU activity time, the data stream resumes later. The longer the PU activity, the slower the rate which leads to longer delays.

- **Region C**: In this region, the PU is active but we notice the spikes in throughput. The spikes occur more frequently at the beginning of the PU activity region due to the long time it takes TFRC to reduce the rate (i.e. with every ACK timeout, it reduces the rate by half). This problem is exacerbated when the nodes surrounding the PU region (nodes 3 and 4 in Figure 2.8) misdetect the PU activity, which occurs...
with a probability of 0.1. To clearly demonstrate the effects of our *paused state* adjustment, however, one must look at the sending rate during this time, and not the throughput at the receiver. This is discussed in the next section.

![TFRC vs TFRC-CR Sending rate](image)

**Figure 2.9:** Sending rate $X_{bps}$ during PU activity region

**Sending rate during Paused state**

Figure 2.9 plots the sending rate in the first PU region from Figure 2.5. We observe that a) TFRC-CR reduces the rate to the *paused state* polling rate (i.e., $\frac{s}{RTT_{avg} + (4 \times RTT_{std})}$) when the PU is affecting the region, and b) how the sending rate slow-starts immediately after. This is in contrast with the slow decay that happens in TFRC, followed by the slow resumption of the rate after the PU exits.

**2.6.2 TFRC sending rate adjustment**

In this section, we illustrate the advantages of our rate adjustment (Section 2.5.2) by comparing the throughput, interference with PU and queue lengths of the affected nodes.

**Throughput with variable PU on time and number of hops**

First, we look at how the throughput changes as we modify the PU on time in centralized and in multihop (2 and 3 hop) scenarios. For each data point, we run the simulator using
Figure 2.10: Throughput comparison as PU on time increases for different number of hops

20 different seeds and compute the average throughput. The PU on and off time follows an exponential distribution and the average value are denoted by the x-axis. The topology follows that of Figure 2.8, where node 4 is always the receiver and the role of the sender alternates from nodes 1 to 3, depending on the hop count.

Centralized system: Figure 2.10 compares TCP, TFRC and TFRC-CR in a centralized (1-hop) network. Interestingly, TCP and TFRC perform well in such topologies due to less random channel errors and dropped packets. We can see that TFRC-CR’s performance is relatively close to that of TCP and TFRC. When the PU on activity is small, TFRC-CR performs worse due to the spectrum management control where TFRC-CR pause the rate every time it detects PU activity and resumes afterwards. Meanwhile, TFRC and TCP don’t pause during these periods, and because the PU activity is small, the nodes that are immediately affected by the PU queue the incoming packets at the MAC layer and send them out once PU activity ceases. We do notice also that TFRC-CR outperforms TCP and TFRC as the PU on time increases.

Multiple (2 and 3) hops: Figure 2.11 gives us the average throughput for both TCP and TFRC in 2 and 3 hop scenarios. We notice the low throughput for both protocols, which further degrades to 0 as PU on time increases. TFRC-CR’s 2-hop and 3-hop implementations are plotted in Figure 2.10, due to the large difference in scale. We can clearly see the advantages of using a time-based window to compute the loss event rate \( p \). In the 2-hop scenario with an average PU on time of 1 s, we see that TFRC-CR’s average throughput is 100 KBps while TFRC averages at 8 KBps and TCP at 12 KBps. This is about a 10
times improvement over both traditional protocols. This example clearly demonstrates the ineffectiveness of existing transport protocols for multihop end-to-end CR scenarios.

**Throughput performance under different stress events**

Figures 2.13(a) and 2.13(b) compare the three transport protocols under different stress events. The three stress events are a) congestion: node 5 sends to 6 in our topology (Figure 2.8) using the same transport protocol, b) bandwidth increase: we set the hop from node 3 to 4 to be a bottle neck by setting its bandwidth to 5 Mbps, and c) bandwidth decrease: we set the bandwidth of the aforementioned hop to 1 Mbps.

This simulation is run for a 2-hop scenario (i.e. node 2 sends to node 4 in our topology). 20 different simulations are run and averaged for each bar. The three stress scenarios are introduced at second 100 in the simulation run and the network returns to normal at second 200. The results are averaged in that stress interval (second 100 to 200).

Though TFRC-CR’s throughput outperforms TFRC and TCP in all scenarios, however this comes at a cost: the queue length at node 3 (the node immediately before the sink node) is higher for TFRC-CR than TFRC or TCP (Figure 2.13(b)) with the exception of the bandwidth increase scenario where it compares similarly to TCP. The increase in queue lengths is expected as this is a direct result of having a larger throughput at the bottleneck node leading to accumulated packets at the queue when these stress events are introduced.
Interference with the PU

This performance analysis compares the three transport protocols in regards to interference with the PU. We calculate the interference percentage with the PU activity by measuring the sum of time it takes to transmit RTS, CTS, ACK and DATA packets during PU activity divided by the total PU activity time period. For each transport protocol, a 2-hop scenario is simulated 20 times and averaged with the variable PU on rate indicated by the x-axis. TFRC-CR’s interference is notably higher than TFRC and TCP (Figure 2.12(a)). This is attributed to the much higher throughput for TFRC-CR (Figure 2.12(b)) in the same simulation runs: when the sending rate is high, the node immediately before the PU vicinity

Figure 2.12: The interference, throughput and goodput comparison are provided in (a), (b) and (c), respectively

Interference with the PU

This performance analysis compares the three transport protocols in regards to interference with the PU. We calculate the interference percentage with the PU activity by measuring the sum of time it takes to transmit RTS, CTS, ACK and DATA packets during PU activity divided by the total PU activity time period. For each transport protocol, a 2-hop scenario is simulated 20 times and averaged with the variable PU on rate indicated by the x-axis. TFRC-CR’s interference is notably higher than TFRC and TCP (Figure 2.12(a)). This is attributed to the much higher throughput for TFRC-CR (Figure 2.12(b)) in the same simulation runs: when the sending rate is high, the node immediately before the PU vicinity
Figure 2.13: Throughput and queue length (for node 3) comparisons under different stress events: none, congestion, bandwidth increase and bandwidth decrease

(node 3) queues packets that are on route and then sends them out when it misdetects PU activity which we set at a probability of 0.1. This happens regardless of how the sender pauses the sending rate. A solution to this problem is to lessen the probability of mis-detection by having more accurate sensors or to immediately empty the queue if a PU is encountered. This involves changing the MAC layer protocol which is beyond the scope of this thesis.

Note that although the interference bars look high for TFRC-CR, the percentage is well below 3% even for the worse case (low PU on time).

**Goodput**

Goodput is measured as \( \frac{\text{total data packets received}}{\text{total data packets sent}} \times 100 \). We can see from Figure 2.12(c) that for the same simulation runs as those of Figures 2.12(a) and 2.12(b), the goodput of TFRC-CR is 95% when PU activity is small (PU time is 2 and 20 for on and off times) and better than TFRC and TCP when PU activity is high (10 and 5 seconds for PU on and off time, respectively). This is an improvement especially considering that TFRC and TCP had little to no throughput in the high PU activity scenario. TFRC-CR’s higher goodput rates at high PU activity scenarios is a direct result of having TFRC-CR slow the sending
rate in the paused state (i.e., when PU is active). By slowing down immediately, it avoids the excess, and eventually lost, packets that are sent in TCP and TFRC.

Figure 2.14: Throughput as the number of streams and PUs change, respectively

2.6.3 High density nodes and PUs

In the following simulation, we place 25 nodes in a $5 \times 5$ grid and we vary the number of end-to-end streams and PUs. Our purpose is to simulate a high density node and PU environment. The on and off times for the PUs is set as an exponential distribution with averages 10 and 5 seconds, respectively. The $x$ and $y$ coordinates of the PUs are selected at random, and so is the source and destination node for each data streams. We study the impact of the number of active streams in Figure 2.14(a), with the number of PUs fixed at 15. In the second study, the number of streams is fixed at 10 in Figure 2.14(b), while the number of PUs is progressively increased. The rest of the parameters are identical to those discussed in the beginning of Section 2.6. In Figures 2.14(a) and 2.14(b), we can see that TFRC-CR provides higher throughput even as the density of the nodes and PUs increase.

2.7 Conclusion

We presented an equation-based transport protocol, TFRC-CR, which is geared to meet the demands of CR networks and presents a fundamentally different control mechanism compared to the typically used TCP-based schemes. TFRC-CR was demonstrated to perform
significantly better than its classical counterparts TFRC and TCP with respect to both PU protection and transmission efficiency in a dynamically changing spectrum environment. Our protocol does not assume any cross-layer feedback or input from intermediate nodes, which aligns it with the traditional end-to-end paradigm in the evolving space of transport layer research for multihop CR networks.
Chapter 3

Accessing spectrum databases using interference alignment in vehicular cognitive radio networks

3.1 Problem overview

The landmark FCC ruling in November 2011 in the US mandated the use of spectrum databases, with rules of access for stationary and mobile cognitive radio nodes, as well as the consideration of specific capabilities such as geo-location [28]. These databases release information on the spectrum usage in the vicinity of the requesting node, which must be periodically refreshed to maintain updated information. However, the FCC also allows for local spectrum sensing, though such unassisted and unilateral sensing by a node must adhere to strict performance metrics. Identifying spatial regions or durations in which (i) the local sensing is likely to yield reliable and repeatable results (i.e., less random fluctuations on the signal imposed by the channel) and when (ii) accessing spectrum databases is a must, remains an open challenge.

In CR vehicular networks, where high bandwidth communication is required by fast moving nodes, the exchange of spectrum information must be undertaken in the shortest possible extent of time [66]. This serves two important purposes: First, the spectrum knowledge remains current and relevant to the specific location a vehicle is currently situated in, and second, given the large number of vehicles that traverse the road, it reduces the channel usage time per vehicle. Thus, the wireless spectrum is more efficiently utilized even during the spectrum querying process, mitigating the risk of congesting this channel used for disseminating the spectrum queries.

The work described in this chapter is focused on answering the following two ques-
tions: How and when must a node rely on spectrum databases vs. relying on self-generated measurements? When spectrum database query is the only solution, how do we ensure that the exchange of a large number of information and control feedback between the neighboring base stations (BSs) and the querying nodes is accomplished in minimum possible time? To address the above concerns, we make the following contributions in this chapter:

3.1.1 Exploring signal correlation between 2G and TV channels

Signals from transmitters that are located in nearby areas are likely to experience a strong similarity in coarse channel behavior, owing to the common set of reflection, absorption causing objects and the large-scale path loss. We explore an interesting concept through an experimental study, where we establish that at certain locations, the behavior of the received signal strength in cellular channels can be used to predict the accuracy of spectrum sensing for certain channels in the TV whitespace. In some cases, the TV transmitter and one or more cellular transmitters (operating on totally different bands) are located near each other, and when the signals from these two are received at a common location, we detect a strong correlation in the behavior of the Received Signal Strength Indication (RSSI). Thus, any sudden change in the TV spectrum usage in such locations can be verified by comparing with the corresponding fluctuations in the cellular channels. This provides an additional layer of check and reduces mis-detection and false alarm. Of course, such a close correlation occurs only for limited durations and at specific locations (based on the relative distances and the presence of these two different transmitter types), but it can potentially offset costly spectrum database queries at these times.

3.1.2 Practical demonstration of interference alignment (IA)

The method of interference alignment allows separation of the useful signals and the interference signals arriving at a node into orthogonal planes. In recent formulations, non-intuitive results were presented for two transmitters and two receivers, each of which is enabled with two antenna modes, wherein four messages could be transmitted in three time slots [67]. We further enhance this scenario with three concurrent transmitters, and extend the basic formulations for a full duplex case with simultaneous transmission and reception. Specifically, we demonstrate how the resulting four transmissions and seven meaningful message receptions (some of them at the intended node, some of them through overhearing) in three time slots are mapped into the case of multiple mobile CR vehicles seeking spectrum updates. To the best of our knowledge, we make the first scenario-specific use of the emerging technique of IA.
The rest of this chapter is organized as follows. We describe the related work and provide background information on FCC database ruling in Section 4.2. Section 3.3 presents the network architecture and overview of the approach, followed by experimental findings that motivate our work in Section 3.4. The database querying strategy exploiting the correlation among TV whitespace and 2G spectrum, as well as the IA setup for efficient channel access, is given in Section 3.5. We undertake comprehensive performance evaluation in Section 3.6, and finally, conclude in Section 3.7.

3.2 Related Work

3.2.1 Cognitive radios and VANETs

In [68], the authors devise a spectrum sensing framework for CR enabled vehicles that send the data to roadside units or a base station (BS), which in turn forwards the data to a processing unit. The processing unit then infers which channels the vehicles are allowed to use based on the aggregate sensing information, and finally schedules this information to be broadcasted to passing vehicles alongside. Belief propagation techniques are used in [69], where vehicles combine different observations from surrounding vehicles, and spatial correlation is used to decide on channel availability. In [70], a framework for coordinated spectrum sensing method is proposed in the absence of any BS or a roadside static base station. Instead, some vehicles are temporarily assigned the role of a “master” vehicle that coordinates the sensing and schedules the transmission activity of surrounding vehicles. In [71], a cooperative sensing framework called Cog-V2V is devised, where each node aggregates information it receives from surrounding vehicles to determine which channel to use in the current and future locations alongside the vehicles’ path. In a study of a different nature, [72] analyzes a major interstate freeway in the state of Massachusetts (I-90) for free spectrum availability along its length. The authors argue for a use of a centralized database that vehicles can use to access the free spectrum information. An algorithm is also proposed on how to reliably populate such a database based on the sensing measurements obtained along the I-90 interstate.

3.2.2 Interference Alignment and Full-duplex wireless communication

Few works exist that demonstrate real-time implementation of IA ever since the authors of [42] wrote the first theoretical and practical scheme in IA. In [73], the authors develop an improvement to the IA scheme called Blind IA where no previous Channel State Infor-
Table 3.1: Comparison between our architecture and related works

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Use DB</th>
<th>Distributed</th>
<th>Use IA</th>
<th>Improve sensing</th>
<th>No precise chan. assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref [68]</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref [69]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Ref [70]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref [71]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Ref [72]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Proposed work</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Different from above, our work aims at sharing spectrum sensing correlation results between two different TV and cellular channels among nodes to improve the accuracy of local sensing. This is in contrast to previous works where nodes cooperate to choose a certain channel for communication. We also offer, to the best of our knowledge, the first real world scenario usage of IA in combination with full-duplex technologies by extending the scheme to allocate three transmitters that exchange four messages in three time slots. The combination of both technologies allow for an accurate, cost effective and efficient utilization of TV whitespace channels using both database management and local sensing.

Table 3.1 shows a comparison between our proposed architecture in relation to the aforementioned works.

### 3.3 Network Architecture and Overview

The overall network architecture and usage scenario of our approach is shown in Figure 3.1. A and B are two fixed BSs that have access to the spectrum database. Each BS
can only provide reliable readings in a limited extent of space, shown by the boundary lines separating them. Each query to the database incurs a finite cost. CR enabled vehicles move along a straight line path from left-right, and two such nodes are indicated by C and D.

- **New incoming vehicle D:** Node D enters a new region serviced by A. At this junction, BS A sends message $x^d_a$ which includes BS A’s ID, coordinates and transmission power. Node D becomes aware of this BS and may use it for future database querying requests. This beacon message $x^d_a$ is also overheard by BS B (more on that in the next section).

- **Fixed BS A and B:** BS A sends its beacon information to D ($x^d_B$) which D may use to query for the upcoming journey through the region served by A. $x^d_a$ includes A’s ID, coordinates and transmission power, which is also overhead by BS B. This information can be used, for example, so that the BSs minimize the overlap region of their coverage area by dynamically adapting and coordinating their transmission power. As per the FCC ruling, a mobile node operating in Mode II must request a new set of spectrum availability readings after moving 100 m from its initial query point, or within a minute, whichever occurs earlier. Thus, a vehicle that has moved further into A’s territory (shown by vehicle C) beyond the first 100 m, must re-obtain the spectrum usage information. Vehicle C sends a preamble to synchronize the BSs in range and to request a new spectrum availability information. BS A sends this information ($x^c_a$) which contains spectrum information for the remainder of the journey in A’s region. At the same time, BS B, which is also synchronized due to hearing the message preamble from C, provides the spectrum usage
(x_C^d) to vehicle C, as it is likely to enter B’s new service area next.

- **Vehicle C within BS A’s service region:** The decisions taken by vehicle C, and the subsequent messaging that follows forms the core contribution of our work. C first determines if there exists correlation among the 2G spectrum and TV whitespace using experimental data (Section 3.4), and if so, sends out a broadcast (x_C^d) mainly intended for the incoming vehicle D. A high degree of correlation implies that the cellular channel can be a reliable metric to quantify the likelihood of sensing errors. This message (x_C^d) is also overheard by both BSs A and B. This overheard message indicates which channels are correlated and therefore can be used for future spectrum query replies. For example, a BS can prioritize the usage of vacant channels that it knows vehicles on the road are not using (due to the absence of any overheard correlation information by the BS). In turn, when D reaches the location currently occupied by C, it undertakes local spectrum sensing using the cellular and TV signal correlation (Section 3.5.1). If vehicle C’s measurements indicate that a spectrum update is necessary, then it issues a preamble similarly to vehicle C previously and both the BSs send the updated query information to vehicle D. The IA approach (Section 3.5.3) ensures that all the message exchanges involving vehicle C and the BSs are completed in fewer slots that the number of messages.

This cycle repeats as vehicles move in a straight line path, ensuring reduced spectrum update overhead and improved local sensing capability.

### 3.4 Experimental study of Correlation between 2G and TV Channels

To study the correlation between the RSSI in the cellular 2G frequencies and the TV channels, we gathered a comprehensive set of measurements in the city of Boston over the span of one month.

#### 3.4.1 Experimental Setup

We used two devices to collect data simultaneously: a Universal Software Radio Platform (USRP) was used to sense the digital TV channels 21 – 51 [77], and an Android Samsung Galaxy S3 smartphone was used to gather the following information: (i) RSSI values of nearby 2G cellular towers, (ii) current GPS coordinates, and (iii) the true TV spectrum availability queried at 60 s intervals. We used Spectrum Bridge [34] by writing an Android application that directly accessed their proprietary APIs to return the available/occupied channels and the signal strengths in the area of study. These devices were placed in a car, and the actual path traversed is shown in Figure 3.2(a). The path progresses along a
counter-clockwise direction, starting and ending at Northeastern University campus (at the bottom).

### 3.4.2 Methodology and Observations

The RSSI samples gathered by the moving car were stored and analyzed offline to detect whether a level of spectrum cross-correlation exists in any $40 - 220$ s moving window among the 2G spectrum and the TV channels. The points in Figure 3.2(a) show the locations where the spectrum correlation were between $85 - 95\%$. Understandably, these correlations were detected at low building density areas: (a) bridges, (b) suburban low building density areas or (c) in broad street intersections, where the random multipath effects resulting from the neighboring structures was comparatively small.

In Figure 3.2(b), we see that as the vehicle moves along a direction from the bottom to the top, there are specific regions for which the TV transmitters (or PUs) in certain
channels and the 2G spectrum towers are nearby, and their respective signals follow a somewhat similar propagation path. This scenario exists for PU1 and the 2G tower. At this
time, the signal from PU2, on a different TV channel, encounters an altogether different
multipath environment. Thus, the signal fluctuations between 2G and PU1 towers are
likely to be correlated in a small window of the traversal path, shown by $L$, and totally
uncorrelated at other times when the structural neighborhood changes. Thus, on observing
sudden changes in the subset of highly correlated TV channels and matching these changes
with the channel behavior of the 2G spectrum, the CR vehicle can identify if these are
carried by multipath effects or due to PU activity.

Figure 3.3 shows a randomly selected location point with two RSSI measurements
plotted against time, one from a 2G cellular tower and the other from channel number
46 in the TV band, with intervals where strong correlation exist. The upper plot is for
measurements through the Android phone (i.e., measuring signals in the 2G spectrum)
and the bottom one is from the USRP (i.e., in the TV channel). A moving average
filter with a set span 11 was applied to both RSSI values for smoothening before calculating the
correlation. The correlation is obtained using the crosscorr function which is explained
by Equation 3.1. In this equation, $f^*$ denotes the complex conjugate of $f$ and $n$ is the lag
of one set relative to the other. We use a default lag of $n = 20$ (i.e. from $-10$ to $10$) and
evaluate $max(f \star g)$ as our maximum spectrum-correlation value. We see in Figure 3.3
that the shaded time interval has 87% correlation while intervals immediately next to this
segment are highly uncorrelated: the spectrum-correlation yielded $-22\%$ and $-44\%$ for
these time durations before and after the shaded region, respectively.

$$
(f \star g)[n] = \sum_{m=-\infty}^{\infty} f^*[m] g[n + m]
$$

(3.1)

3.4.3 Motivation for Proposed Research

In our proposed method, for certain locations and TV channels for which a strong corre-
lation exists with the 2G spectrum, the CR vehicle may not require additional spectrum
database updates to infer PU activity in these specific TV channels. Instead it may rely on
local spectrum sensing alone, as for these channels it can track and attribute the cause of
rapid signal fluctuations, thereby lowering the chance of false alarm and missed detection.

A vehicle that already has access to 2G RSSI values (effectively operating in Mode II)
may be able query the spectrum database directly without the need of a roadside base
station or to perform local sensing. However, this is not always possible or optimal due
to multiple factors: (a) the phone observes the RSSI values but does not access the Internet (e.g. when roaming). Moreover, a phone’s service provider represents only a subset
of the total RSSI values obtained, while a significant portion of the measurement data is gathered from towers that the phone cannot legitimately connect to (i.e. different HSPA mobile carriers), (b) having access to roadside base stations or local sensing, as opposed to querying the database directly, will save in the total overhead over the Internet given that a BS needs to query the database once every 24 hour period. The BS can then mediate and share that information with passing vehicles saving on total traffic directed to the centralized database, (c) the latency to retrieve such spectrum information is reduced because the BS is much closer to the vehicles than an online database, and (d) given that the information is retrieved from the BS which queries the database once and then shares it at a fraction of a cost, the total cost per query incurred by each vehicle is also reduced.

### 3.5 Optimal Database Querying Strategy

Using the network architecture described in Fig 3.1, we discuss in this section how a node decides whether spectrum database or local sensing needs to be undertaken, how to improve the local sensing capability, and undertake message transfers with reduced control overhead.
Figure 3.4: Four different types of RSSI readings performed by sensing

Figure 3.5: Decision tree of the cognitive radio node. Solid lines indicate a possible outcome while the dotted lines indicate the impossible ones
3.5.1 Exploiting Correlation between 2G and TV Whitespace

Node $C$ collects the RSSI values of the 2G and whitespace towers spectrum, and analyzes it to determine if a high correlation exists between any pair of TV channels with those in the 2G band. We describe the four cases below that demonstrate the types of readings that $C$ may obtain at any given point. The four cases are shown in Figure 3.4.

- **Case I** is when a vehicle reads RSSI values that all surpass a certain sensing threshold $\lambda$ [78] on average.
- **Case II** is when the average drops below that threshold due to significant fading.
- **Case III** represents very low RSSI readings, which is the case for vacant channels. These RSSI readings are well below the threshold.
- **Case IV** occurs when the average RSSI value is above the threshold, but the channel is really vacant. Case IV’s outcome, however, can be minimized because contrary to current detection schemes that have a probability of false alarm rate [78–80] due to uncertain noise power knowledge, in our proposed architecture, nodes can have perfect noise power knowledge by studying the RSSI values of channels that are known to be vacant by the database and setting the threshold to a value above the highest observed floor noise RSSI value. After minimizing the probability of false alarm in Case IV, vehicle $C$ will have to make a decision based on which of the remaining three cases of RSSI readings it is currently observing.

Vehicle $C$ uses the observed RSSI values (in the 2G spectrum and the TV whitespace), as well as the energy detection based sensing threshold $\lambda$ to infer whether a PU exists or not, or whether the correlation information leads to indeterministic results. We use Figure 3.5 to explain the possible scenarios using the decision pathways indicated by numbers at the end points in the figure. The following numerical list correspond to these numbers. Note that this decision logic is used after correlation is detected by $C$ between the 2G and the TV whitespace (WS) RSSI signals. The dotted lines indicate a state that is not feasible, while the solid pathways indicate a possible outcome:

1. **WS RSSI > $\lambda$ and 2G RSSI > $\lambda$**: this indicates that the PU is currently occupying the channel because a correlation exists, and both RSSI values are above the threshold $\lambda$.

2. **WS RSSI > $\lambda$ and 2G RSSI < $\lambda$**: this scenario is not feasible because the 2G towers don’t have an on and off time and should remain constantly on at this location. Therefore if a correlation existed between this pair previously, this outcome is not possible. If the 2G network is not present, then we expect the WS channel to also be vacant due to their previous correlation but this isn’t the case. The node must query the database if it wishes to use this whitespace channel.
3. $WS RSSI < \lambda$ and $2G RSSI > \lambda$: this outcome indicates that a PU is not present; the two signals were previously correlated (and correlation only happens when both channels are not vacant), if the 2G signal is still occupied while the WS signal is not, then this indicates that the PU is not present and the node may use this channel to transmit data. These two channels have a high correlation therefore if fading, shadowing or multipath effects exists, it will affect both of the channels and not just one of them. This is not a possible outcome if the PU is present.

4. $WS RSSI < \lambda$ and $2G RSSI < \lambda$: this is a possible outcome if the PU is present. The two correlated channels are below the threshold $\lambda$ which indicates a possible fading/shadowing/multipath effect that is affecting both channels. The read RSSI values are below the threshold but we know from previous correlation that the two channels exhibit the same effects.

5. $WS RSSI > \lambda$: From our experiment results, if WS RSSI values are above the threshold $\lambda$, then this indicates that the PU is present. Therefore this is not a possible outcome.

6. $WS RSSI < \lambda$ and $2G RSSI > \lambda$: this is a possible outcome, see (3) above.

7. $WS RSSI < \lambda$ and $2G RSSI < \lambda$: this is not a possible outcome because if the two channels are correlated, then the absence of the 2G RSSI network indicates the presence of channel effects (fading/shadowing/multipath) that is exhibited on both channels, and therefore the PU is still occupying the channel.

### 3.5.2 Database Query or Local Sensing Decision

In this section, we explain the querying process for a snapshot of the vehicles currently in the network, and shown in Figure 3.1. We show how the database querying is accomplished by vehicle $C$, resulting with low traffic on the channel. We then extend the discussion for vehicle $D$, and then finally, for any other general vehicle along the road.

- **Case for Vehicle $C$:** In this case, vehicle $C$ follows the steps outlined in Figure 3.6 to determine if it is able to come to a conclusion about the PU activity using the correlation and RSSI measurements. As vehicle $C$ enters a new region, it checks to see if there is any prior knowledge of correlation between TV whitespace and 2G spectrum at its present location. If no (the decision flow moves along the right arm of the flowchart), it issues a spectrum query to the nearest BS. At this stage it is in a position to measure the signal correlation itself, and inform other vehicles in the downstream chain. If a spectrum-correlation is found, it broadcasts that information to these vehicles, including node $D$. If no correlation
is detected, it simply uses spectrum information from the database, and concludes the algorithm. Please note that the Query Database and the Find Correlation blocks can happen simultaneously; this key fact will be used in the subsequent section when querying the database and transmitting to the downstream vehicle happens simultaneously via IA.

- **Case for Vehicle D:** We describe this case for vehicle D using the left arm traversal of the decision logic in Figure 3.6. We assume that it received prior confirmation of spectrum correlation at its present location (say, by vehicle C). It then uses the reasoning described in Section 3.5.1 to determine PU presence. First, it checks if the TV whitespace (WS) RSSI readings are below the threshold. If so, it checks whether the 2G RSSI readings are also below the threshold. If both conditions are true, then it concludes that the PU is present. It arrives at this somewhat nonintuitive result as (i) the prior vehicle verified correlation exists, (ii) the 2G spectrum has beacon signals that are always present at set intervals in time, and (iii) it does not detect the 2G beacons, possibly lost due to multipath and fading, which indicates similar trends in the TV whitespace. On the other hand, if the 2G RSSI values are above the threshold, then (see path 6 in Figure 3.5) we determine that the PU is absent, and D may use that channel for data communication.

If, however, the whitespace RSSI is above the threshold \( \lambda \) which indicates the presence of the PU, vehicle D completes the algorithm by moving to the right arm of the decision flow, i.e., it issues a query to the database. The rest of the steps follow exactly the procedure for vehicle C described above.

The description of the steps for this network snapshot are applicable for any vehicle in the network. The overall actions of the vehicles along the entire length of the road allows each vehicle to extrapolate correlation information and hence, serve as the source of spectrum correlation information for the remaining downstream vehicles.

### 3.5.3 Spectrum Updates using Interference Alignment

The overall aim of the database access phase, when it occurs, is to allow all the transmissions and packet exchanges to be completed in exactly three time slots. This results in a remarkable efficiency of the channel usage. The modification to the classical IA scheme that is leveraged for this purpose is described next.

Interference alignment places the interfering signals along vectors oriented orthogonal to those of the useful signal [42]. It achieves this by allowing the receiver to switch antenna modes according to a pre-set pattern. The antenna mode can be altered by changing the antenna response parameters, reducing the height, or as we suggest in our case, simply use antenna 1 or 2, when the receiver needs to be in mode 1 or 2, respectively. Thus, IA demonstrates much improved resilience to signal losses by not attempting to cancel or limit the interference level at its location, which requires extensive transmitter side information.
Figure 3.6: Sensing vs. Querying decision flow chart
Our extension to the 2x2 blind model of IA proposed in [67] is based on two important considerations. First, we use a full duplex radio that can simultaneously transmit and receive, though it has only one transmitter and receive chain, respectively. Off the shelf USRP radios can be adapted to support this full duplex operation [75, 76]. Secondly, we depart from the traditional model of a cleanly separated set of transmitters and receivers. Instead, in our case, one of the receivers (vehicle C) switches its role into a transmitter to send a message to the second receiver (i.e., the downstream vehicle D). The individual alignment equations based on the received packets are described below for each node, and the channel gains for the unidirectional transmissions are depicted in Fig 3.7.

Transmitter/Receiver- Vehicle C

CR Vehicle C changes its mode in the pattern 1 – 2 – 1, as indicated by switching between the antennae 1 and 2. It subtracts the third equation from the first to get two unknowns in two variables, by eliminating $x^d_a$, which is the message destined for D. The resulting set of linear equations can be solved to obtain $x^c_a$ and $x^c_b$, respectively. These unknowns represent the spectrum database information C requested from A, and B for the subsequent journey towards B’s control region. In the last transmission slot, C transmits its assessment of the feasibility of spectrum sensing to D, using the transmit antenna 2. This message, $x^d_c$, is also overheard by the BSs A and B, which as we noted previously in Section 3.3, can be used for different quality of service channel prioritization in all subsequent queries.

Receiver- Vehicle D

CR Vehicle D subtracts the second equation from the first, for which it was in the same mode 1. This leaves the only unknown $x^d_a$, which is the beacon information that A intends to supply to D as it enters for the first time in its control region. This value is then sub-
Antenna

<table>
<thead>
<tr>
<th></th>
<th>Rx</th>
<th>Tx</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: Interference alignment equations for node C

Table 3.3: Interference alignment equations for node D

Transmitter- BS A

BS A hears the transmissions from the neighboring BS B in the first two slots, and the transmission from vehicle C to D in the final slot \( x_c^d \). This last message also informs BS A what channels vehicle C found to be correlated and may use that information for subsequent queries in, for example, optimizing quality of service. While A does not switch operational modes, it does rely on the full duplex ability to continuously transmit on antenna 1 and receive on antenna 2.

In the first slot, it transmits a summation of the messages \( x_a^c + x_a^d \) to both the vehicles, while in the second slot, it only transmits \( x_a^c \) to vehicle C. These messages are received by the adjoining vehicles and BS B in the first two slots, respectively, subject to the channel gains from A to them.
Table 3.4: Interference alignment equations for node A

<table>
<thead>
<tr>
<th>Antenna</th>
<th>Rx</th>
<th>Tx</th>
<th>Node A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>$z^{[a]}(1) = (x_b^c)h_{ba}^{12}$</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>$z^{[a]}(2) = (x_b^c)h_{ba}^{12}$</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>$z^{[a]}(3) = (x_c^d)h_{ca}^{22}$</td>
</tr>
</tbody>
</table>

Transmitter- BS B

The operation of BS B is very similar to that of C. Subtracting the second equation from the first returns the message $x_a^d$ sent to D from A. This is, in turn, substituted in equation 3, to obtain $x_c^d$, which informs B what channels vehicle C found to be correlated and may use that to improve subsequent queries’ quality of service.

Table 3.5: Interference alignment equations for node B

<table>
<thead>
<tr>
<th>Antenna</th>
<th>Rx</th>
<th>Tx</th>
<th>Node B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>$z^{[b]}(1) = (x_a^c + x_a^d)h_{ab}^{12}$</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>$z^{[b]}(2) = (x_a^c)h_{ab}^{12}$</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td>$z^{[b]}(3) = (x_a^d)h_{ab}^{12} + (x_c^d)h_{cb}^{22}$</td>
</tr>
</tbody>
</table>

In summary, the information exchange equations are carefully constructed using IA, while ensuring that the preference for local sensing or database updates is correctly communicated to the network entities.

3.6 Performance Evaluation

In this section, we first provide trace-driven simulation studies to demonstrate the feasibility of exploiting the spectrum correlation approach described in Section 3.6. Then, we provide quantitative results on the efficiency of the channel utilization by using the IA scheme from Section 3.5.3 through a simulation study in MATLAB.
3.6.1 Improvements due to spectrum correlation exploitation

We use traces from the experimental setup described in Section 3.4, where a USRP radio is placed inside a moving vehicle. The Android phone accesses the 2G spectrum, maintains a history of the path traveled via the in-built GPS capability, as well as queries the Spectrum Bridge Inc. database through software APIs every 100m traversal, or when 60s elapse, whichever is earlier. As a subsequent check, all measured sensing results obtained via energy detection through the USRP are validated with offline verification through the database, at the end of the experiment. A subsequent trailing vehicle may schedule its own query, unless the vehicle ahead transmits a beacon that informs it to cancel the database query and rely on local sensing, using the procedure adapted from Section 3.5. The observed signal fluctuations in the 2G spectrum as well as the TV whitespace are saved in a laptop computer and correlations are calculated via a continuously running MATLAB program.

Number of spectrum correlation points

Figure 3.8 shows the number of spectrum-correlation points as the vehicle moves along the map shown in Fig 3.2(a). In this figure, we can see that the number of these points is higher when the correlation level is set at 70%, as compared to 85%, by about 4 times. This demonstrates the trade-off that exists between higher accuracy (imposed by higher correlation demands) between the 2G spectrum and TV whitespace, and total number of correlation points where this approach can actually be used.
Figure 3.9: Number of queries for various spectrum correlation percentages

Number of queries

We list the total number of database queries that a vehicle engages in the path that was traversed to collect the spectrum information in Figure 3.2(a). As the distance traversed increases, the number of queries that have to be performed also increase. In Figure 3.9, we plot the number of queries that are undertaken by the CR vehicle if no spectrum correlation information is leveraged, as opposed to when it utilizes the spectrum correlation information to reduce the number of direct queries. Lower the allowed spectrum correlation percentage, the less the number of queries that the vehicle will undertake. In the case of correlation level set to 70\%, the vehicle only needs to query the database 108 times, instead of 140 without our proposed scheme. This results in a saving of up to 23\%. This number can be easily scaled with appropriate multipliers, such as the monetary cost of spectrum usage time imposed by carriers, the channel utilization, resulting data congestion, among others to obtain tangible impacts of reducing the number of database queries.

Accuracy

In Figure 3.10, we plot the probability of error with \( P_{err}^x \) and without \( P_{err}^c \) our proposed scheme where

\[
P_{err} = \frac{\text{Number of times the query resulted in mis-detection}}{\text{total number of queries}} \quad (3.2)
\]

We first calculate \( P_{err}^c \) by assuming Rayleigh fading over a period of time for the gathered RSSI readings, to check whether the average RSSI values obtained over the sensing time \( t_s \) are above or below the RSSI threshold \( \lambda \). Here, we chose \( t_s = 0.022s \) and \( \lambda \) was
Figure 3.10: Accuracy of sensing

empirically observed over the total simulation time [78] to be $-0.5$ (amplitude dB). We then calculate the fraction of times that the average RSSI value is below the threshold over the total number of sensing intervals to give us the classical energy detection probability of error $P_{err}^c$. This is in turn plotted against $P_{err}^x$:

$$P_{err}^x = (1 - \frac{x_j}{X}) \times P_{err}^c$$  \hspace{1cm} (3.3)

where $x_j$ is the number of spectrum correlation matches that occur at speed $j$, and $X$ is the total number of such correlations obtained over the entire experimental duration. One can observe that the probability of error decreases as the speed of the moving vehicle increases for both the classical energy detection and the correlation based approach. However, the error in the latter is always less at all the measured speeds. This is because the RSSI may stay below the threshold for longer periods of time when the speed is slower, resulting in the sensing errors when $t_s = 0.022s$ [78].

3.6.2 Improvements through interference alignment

In this section, we simulate the scenario in Figure 3.1 where the entire distance traversed is $2 \text{ km}$ and the BSs $A$ and $B$ are located at distance $500 \text{ m}$ and $1500 \text{ m}$, respectively with the coverage radius of each BS set to $500 \text{ m}$. We vary the inter-arrival rate of vehicles that
Figure 3.11: Channel utilization per vehicle for no IA allowed, IA allowed and pure query scenarios in (a) light density and fast moving vehicles, (b) light density and slow moving vehicles, (c) high density and fast moving vehicles and (d) high density with slow moving vehicles
arrive from left to right in the figure (read vehicle density) and the vehicle speed and determine the performance gains in channel utilization and number of queries. We compare the performance of different scenarios: (a) Pure query: in this case, vehicles do not leverage any spectrum correlation information. Vehicles entering will query the database whenever 60 s has elapsed or 100 m was traversed. No carrier-sensing algorithm is implemented here, so the queries may collide causing interference. (b) CSMA: in CSMA, vehicles back-off when a collision is detected based on 802.11 standards. (c) Without IA (w/o IA). This scenario has vehicles using spectrum correlation information to save database query costs but does not exploit IA for channel utilization gains and (d) with IA (w/ IA) where vehicles send data using IA after exploiting any spectrum-correlation information. In all simulations, we use a real-world segment of the RSSI readings that we obtained in our March 2011 experiment (See section 3.4).

**Channel utilization**

Figure 3.11 shows a comparison of channel utilization per vehicle (time slots) in various speed and density scenarios as the simulation is run for a duration of 1000 s. The results are averaged over 20 simulation runs. There are a number of interesting facts that can be derived from the four figures: (a) When speed is increased, the channel utilization is improved through IA because a vehicle can obtain future spectrum information from tower B at the same instance it is also obtaining it from A for the rest of the duration in A (See Section 3.5.3 and 3.6), while in the slower scenarios, the vehicles have expired spectrum information which requires them to re-query the database and hence incur higher channel utilization. (b) When density increases, without IA (w/o IA) scheme is significantly improved over pure query. This is because having more cars in the scenario leads to better chances of sharing (and exploiting) spectrum-correlation information and hence, less channel utilization. (c) In all scenarios, IA outperforms both pure query and without IA.

In Figure 3.12(a), we vary the vehicle density (inter-arrival rate) while keeping the velocity constant. Conversely, in Figure 3.12(b), we fix the vehicle density and vary the speed. We compare CSMA with pure query, no IA information leveraged, and full IA in the global utilization of the channel. In both cases, the CSMA reaches channel saturation early. This is because an increase in vehicle density and speed causes more collisions, leading to the channel being sensed for prolonged periods of time. We also see that when we increase the vehicle velocity in Figure 3.12(b), the IA scheme results in slower increase in total channel utilization than that of pure query and without IA. Since we keep the average distance between vehicles unchanged, increasing the speed results in an increase of the number of cars entering the region at any given time and thus, in all cases total channel utilization increases. When we increase the vehicle density in Figure 3.12(a), we
see that CSMA attains maximum channel utilization before the three remaining schemes. Additionally, these competing schemes result in a faster increase in channel utilization as the vehicle density grows, as compared to the scheme with IA.

Number of queries as the noise level changes

In Figure 3.13, we apply White Gaussian noise to the signals on the 2 km stretch of the simulation with increasing SNRs. The plot shows that as the SNR increases, the number of queries decreases. This is due to having more spectrum-correlation between the TV and 2G signals as the SNR increases leading to less reliance on the database to query for spectrum information.

3.7 Conclusion

In this chapter, we describe a new paradigm for spectrum database access, which allows querying the database only when needed. The resulting method exploits the correlation that exists among two entirely different spectrum bands at specific locations, thereby improving the performance of local sensing and reducing the costs associated with repeated database queries. Results reveal about 23% reduction of queries, making it attractive for practical spectrum database deployments. In addition, we also explored a real-world interference alignment application that can reduce the control channel utilization. We followed this by simulation runs to verify the performance gains using this approach. This
non-trivial approach can potentially open up a hitherto unexplored direction in spectrum sensing, and future work will be focused on building the protocol suite that enables quick and efficient exchange of spectrum data between the vehicles and the BSs.
Chapter 4

A cognitive radio module for the network simulator 3

4.1 Problem overview

Networking engineers and researchers contributing in the emerging area of cognitive radio researchers face uphill challenges when testing a new concept, given the challenging environment in which these networks operate. The CRs must opportunistically determine which licensed channels are available and make use of this spectrum before the licensed user reclaims it. Accurate protocol operation is critical, as any prolonged use of the channel raises concerns of interfering with the activities of the licensed users. This concern directly translates to meticulous testing of the protocol or networking concept in a controlled environment. Given the uphill costs of purchasing multiple software defined radios that serve as the hardware building blocks of the CR network, and the time investment is writing and deploying code in them, computer simulation often becomes the methodology of choice.

While several commercial simulators exist, such as OPNET [43], which can capably simulate heterogenous networks, our focus in this work remains on affecting improvements to open-source simulators. The main challenges in such environment are as follows:

- CR protocols are generally cross-layered. Any change in one layer, such as spectrum sensing duration at the physical layer (PHY), has direct impact on the decisions made in the upper layers of the protocol stack, requiring extensive code changes throughout the stack.

- The testing time increases dramatically with the complexity of the protocol. Owing to the large number of variables that can be controlled, identifying the dominating
factor that impacts the environment to the greatest possible extent may force large number of trial runs.

• The inter-dependence of the protocol layers requires network architects to implement more than one layer. For e.g., spectrum sensing at the PHY can impact the TCP throughout and its interpretation of congestion. Thus, the expertise required to simulate CR networks effectively is more than conventional wireless networks.

• New functions unique to the area of CR, such as spectrum sensing, spectrum handoff and licensed or primary user detection need to be embedded in the simulator.

This chapter is focused on providing the first cognitive radio extension to the network simulator 3 or ns-3, which is a discrete event driven simulator. This simulator is poised to replace the widely popular predecessor, network simulator 2 or ns-2. Ns-3 offers several advantages over ns-2, including: (i) it has a new core written in C++, (ii) it is geared for wireless communications, (iii) it has an organized modular architecture that is expandable, (iv) it includes intuitive and extensive documentation via the html Doxygen interface, and (v) the same ns-3 code can be easily adapted to work in real devices.

Despite the clear superiority of this new simulation platform, ns-3 lacks implementation support for CR networks. To introduce this capability, several changes are required in various network layers in ns-3. For example, a device needs multiple wireless interfaces to transmit, receive and negotiate with neighboring nodes. The PHY need to be able to sense and detect Pus, the medium access control (MAC) needs to decide and initiate handoff to another available channel when a PU is detected, and the routing protocol needs to exchange the neighboring nodes’ current listening channel. Based on the recent FCC mandate that requires CRs to use a centralized spectrum database, the simulator will have to also incorporate such querying capabilities to identify the available channels using these databases.

Our main contribution is to fully realize the first CR extension for ns-3 that has the following features:

• It provides CR capabilities at the different network layers such as sensing, PU detection, channel handoff and decision making.

• It incorporates the ability to query a database to obtain PU activity results.

• It has the ability to simulate cognitive with non-cognitive legacy wireless nodes in one test environment.

• It includes seamless support for multi-channel and multi-radio node architectures.
**Simulator** | **Multi-interface** | **Hand-off** | **Cog and Non-Cog** | **Query DB** | **Extensible policies** | **Ns-3**
---|---|---|---|---|---|---
CRCN [81] | ✓ | ✓ | | | | |
CogNS [82] | | ✓ | | | | |
[83] | ✓ | ✓ | ✓ | | ✓ | |
[84] | ✓ | ✓ | ✓ | | ✓ | |
CRAHN [85] | ✓ | | ✓ | | ✓ | |
CRE-NS3 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓

Table 4.1: Comparison between CRE-NS3 and related works

- It provides new application programming interfaces (APIs) to expose the creation of these node-level and network-level features without advanced code changes. It comes with extensive documentation through Doxygen.

- It allows studying the overhead, performance, and comparison with an existing extension of CR in ns-2.

- It is accompanied by the release of the full source code, with additional guides on how to compile and run trial examples.

The rest of this chapter is organized as following: Section 4.2 discusses the existing network simulators, and their utility in implementing work for CR networks. We describe the model of our proposed approach in detail in Section 4.3, followed by the proposed changes to the networking layers in ns-3. Performance evaluation studies are presented in Section 4.4, and finally Section 4.5 concludes the chapter.

### 4.2 Related Work and Background

To the best of our knowledge, no previous CR implementation exists for ns-3. There are, however, few modules for CR that has been built for other simulators. Cognitive Radio Cognitive Network (CRCN) [81] provides a radio extension for ns-2. This simulation framework provides multi/single radio and multi-channel support per node. It provides APIs that return information, such as the current noise or traffic conditions at a given channel, and provides a mechanism for channel hand-off. CogNS [82] provides another extension to ns-2. This module allows one network interface per node that is able to sense PU activity and defer to another free channel based on a proposed spectrum decision algorithm. Nodes created in this environment cannot incorporate multiple radios per node. [83] provides a CR simulation extension for OMNeT++ [86]. It provides support for multiple interfaces per node, and is focused on evaluating CR MAC layer protocols.
[84] is a simulator written in C++ for CR networks. The proposed simulator offers a modular approach that provides a full network layer stack. Because this work is not based on a more popular platform that has been tested by the general networking community, this simulator involves some overhead in porting the already known protocol stacks in changing environments. For e.g., for a researcher interested in simulating CR networks with an alternate MAC, such as a different 802.11 standard, the effort required is prohibitive. As compared to this, other popular simulators such as OMNet++, ns-2 and ns-3 already provide a wider user-base and support community.

The work described in this chapter uses the ns-2 CRAHN extension in [85] as the starting point. This earlier CR extension for ns-2 provides three network interfaces per node: a control interface to exchange control information between neighboring nodes, a receiving and transmitting interface. The CR module also incorporates hand-off and sensing capabilities at the PHY and MAC layers. It provides for PU detection mechanisms based on a PU activity model table that is loaded in the simulator a-priori. Our work, while based on this extension, has the following key differences: all the work that is proposed in this chapter is based on the new ns-3 simulator, which (i) requires extensive code change, (ii) provides dynamically adjustable sensing/hand-off times that can be configured from the command line, (iii) is able to include cognitive and non-cognitive interfaces in one node, and (iv) is able to mix cognitive and non-cognitive nodes in one test environment. The simulation extension we propose can also incorporate a larger number of cognitive interfaces and/or nodes compared with ns-2 (in the order of thousands). We also design a different simulation architecture, composed of a Spectrum Manager block that hides or masks the inner CR API calls, leading to an organized modular approach.

Overall, our proposed extension provides easy access to the specialized CR functionalities compared to other existing simulators through APIs, offers flexible CR and non-CR nodes to coexist, and incorporates advanced abilities, such as querying the recently mandated FCC database to infer PU activity. Our work aims at realizing a complete CR simulation platform to save time and resources for CR researchers who are either forced to write the entire CR software stack or embark on the costly enterprise of purchasing multiple real devices.

Table 4.1 provides an overview of the comparison between our proposed ns-3 extension, and the other approaches discussed in this section.

### 4.3 Ns-3 Simulator Model for CR

In this section, we present the architectural model of our proposed ns-3 simulator. We first describe the building blocks of the extension, followed by an explanation of the needed
changes at various network layers for a given node.

4.3.1 Building blocks for the simulator

The typical functions of the CR node, follow the essential stages of Spectrum Mobility, Spectrum Sharing, Spectrum Sensing and Spectrum Decision, as described in the cognitive cycle described in [1]. As seen in Figure 4.1, these operations may occur one after the other, moving the CR node from one operational stage to the next. We use a similar approach to design the ns-3 simulator, by implementing each of these constituent blocks of the cognitive cycle, as shown in Figure 4.2. The spectrum management global block serves as a black box to the other modules in ns-3. Different layers in the network simulator keep a reference to the Spectrum Manager instance and tie their cognitive functionality via exposed APIs and hooked listeners. An example of such APIs are startSensing(), startHandoff(channel), isSpectrumFree(channel), alertNeighbors().

Internally, the Spectrum Manager block contains several submodules that map to the cognitive cycle, which are described next.

- **Spectrum Sensing/Database Query**. This block is responsible for checking whether a PU exists in a given channel within a specified period of time. It infers the PU activity from a static PU Database that is loaded before the simulation starts. This submodule can be used to either (a) mimic sensing with a given probability of sensing error $P_{err}$, or (b) query an FCC approved database to determine the PU activity. The PU database that is loaded into the simulator is a text file that defines the number of PUs, their current occupied channel, the transmission power to determine the range, and a list of on ($PU_{on}$) and off ($PU_{off}$) times. A researcher can create and
Figure 4.2: The main building blocks of the proposed extension

load this text file with any given $PU_{on}$ and $PU_{off}$ distributions such as normal or exponential.

- **Spectrum Decision.** In this block, several policies are implemented. First, a policy is incorporated to determine whether a hand-off should be performed based on sensing/querying results. Second, a policy that determines to which channel a hand-off should happen is written. Note that these policies are extensible: One can implement different and independent policies, or extend the existing ones. A global repository is linked to this submodule which hosts current occupied channels by all CR nodes in the simulator. This can be used, for example, to determine the least occupied channel that a node should switch to to guarantee desired Quality-of-Service (QoS).

- **Spectrum Mobility.** This submodule initiates the hand-off protocol in the current node. It is tied directly to the PHY layer in the ns-3 node which will be discussed later in Section 4.3.2.

- **Spectrum Sharing.** This submodule uses the built-in carrier sensing MAC 802.11 standards in ns-3 to make sure that the available spectrum is shared in a collision-free manner between the CR nodes that chose to transmit on this same channel.

These sub-blocks are linked internally whenever a given API in the Spectrum Manager
block is invoked. For example, to initiate sensing, a researcher may invoke a call to the `startSensingAndSwitchIfNecessary()` API from any arbitrary network layer. The Spectrum Sensing block will then initiate a look-up via the PU Database, call the Spectrum Decision block once sensing is performed. If the Spectrum Decision block decides to switch to a new free spectrum, it initiates a hand-off via a call to the Spectrum Mobility block. The same network layer that invoked the call will then be notified via a listener once this cycle is completed and transmission can be resumed. This can be used, for instance, in a transport layer protocol where the transport protocol needs to determine when to resume the data flow [87].

### 4.3.2 Layer-specific modifications to ns-3

In this section, we detail the needed changes to each layer of the protocol stack for a given CR node in ns-3. Figure 4.3 depicts an overview of these changes. As can be seen, the proposed CR extension exposes several APIs and listeners to all the networking layers. We also make use of ns-3 tagging feature. The method to ‘tag’ a packet with some information helps to determine that packet’s internal routing in a given node, thereby avoiding the costly overhead that would ensue if said information was to be integrated into the packet’s header instead. More details on this feature and how it is used will be discussed, as we explain the changes to each of network layers.

**All layers up to the transport layer**

No changes are proposed to these layers. However, all the Spectrum Manager’s APIs and listeners are exposed to these layers so a network researcher working on a CR application, for example, can make use of the CR features of the node by calling the respective APIs in the Spectrum Manager.

**Transport layer**

Our framework modifies this layer so that any packet that is generated here will be tagged as a `DATA` packet. This information will be processed by the lower layers to determine the correct routing of such packets. This change affects all transport layer protocols defined in the simulator such as TCP, UDP, and potentially any new transport protocol that a researcher might be interested in implementing.
Network layer

For CRs to work in an ad-hoc topology, some information must be exchanged between neighboring nodes to determine listening channel of each member of the network. We extend the information carried in the packets of the AODV protocol [88] to include the current listening channel of each node. This information will be passed along with every HELLO, RREQ and RREP messages. Every packet that is generated by AODV is tagged asCTRL or control packet. This tag will be used by the lower layers to route the packet to the correct MAC interface. Moreover, the address resolution protocol (ARP) [89] is extended to route the packets depending on their tag to the correct MAC interface on the destination node. A detailed discussion on the MAC at the link layer and physical layer changes is given next.

Link and physical layers

We have undertaken substantial changes in both these layers. First, we define a new concept of Cognitive Interface (See Cognitive Interface block in Figure 4.3). A CR node may define any number of these cognitive interfaces. Each interface constitutes of three separate MAC-PHY layers; The first is for communicating control packet information on a common control channel. For e.g., AODV and ARP messages will be communicated over such an interface. We call this interface theCTRL interface. The second is used to transmit data messages to neighboring nodes (TX). This interface is switchable; i.e., it switches between different channels to transmit queued data packets that are destined to different nodes (and possibly, different listening channels). The switching time and how long each channel transmission duration is can be defined using the ns-3 attribute system. This system is a mechanism to pass parameters on the command line without the need to recompile the core of the simulator to change the value of various exposed parameters. There are several TX interface switching policies that we defined in our simulator, such as round robin, random, and switch between ‘active’ channels that implies channels that currently have packets awaiting transmission in the MAC queue. Finally, a switchable receiving interface RX is present, which senses for PU activity, hand-offs when PU is detected and alerts the neighbors about its new channel selection via an AODV HELLO packet. The transmission, sensing times, and probability of detection error can all be defined using the ns-3 attribute system. We emphasize that the Cognitive Interface makes all these new calls through the Spectrum Manager block. This provides a cleaner and easier cross-layer referencing as opposed to having each layer hold references to several other network layers. The tagging mechanism that was discussed earlier in the transport and network layers are used here to determine which interface a packet should be sent on.

The TX Distributed Coordination Function (DCF) is also modified to store enqueued
packets into different MAC queues based on the channel that they should be transmitted on. This will help the $TX$ interface select which packets to transmit when it switches spectrum.

At the physical layer (PHY), a new sensing state is added. The functionality of the sensing state is similar to that of the hand-off state where the PHY layer instructs the DCF to halt dequeueing from the respective MAC queue, while the sensing or hand-off operation is ongoing. The sensing and hand-off times can be defined using the ns-3 attribute system. The sensing state in the PHY layer uses the Spectrum Manager APIs which query the PU Database (See Figure 4.2) to determine PU activity. Note that the PHY layer can switch between any number of defined channels. These channels can have a different frequency, propagation path loss and delay models, as defined by the default ns-3 simulation environment.

### 4.3.3 RX interface cognitive cycle

When the RX component in the Cognitive Interface starts sensing, hand-off or transmit data, it transitions along the cognitive cycle depicted in the state machine in Figure 4.4. The cycle is first triggered by the *Sense* state. If no PU activity is detected, the state moves to the *Transmit* state, and after a predefined period of data transmission, returns back to
the Sense state. If a PU is found, the state machine moves to the Decision state. Based on the policies executed, if the decision is to stay on the PU occupied channel, then no transmission will happen and the state immediately returns back to the Sense state. If a hand-off is decided, the decision block also decides which channel the hand-off should occur to. After this, the state machine moves to the Handoff state. Once the hand-off is completed, sensing is triggered again before confirming the PU’s vacancy and the data transmission resumption.

4.4 Performance Evaluation

In this section, we first validate the proposed module by running a single UDP flow from one CR node to another. We study the performance overhead when comparing ns3 with our approach (abbreviated as CRE-NS3), and finally evaluate CRE-NS3 versus the previous iteration of this module for ns-2 (CRAHN [85]).

The environment where the next set of evaluations are conducted is an Arch Linux 64-bit distribution with Linux kernel v3.13.5. The CPU is an Intel Core i7 860 clocked at 2.80 GHz. All simulations were performed in a single thread/core. The installed RAM has a total capacity of 16 GB.

In the simulations below, the nodes perform sensing and data transmission in intervals
Figure 4.5: UDP flow’s throughput vs. time in a CR network. Gray areas indicate PU activity of 100 ms and 1 s, respectively. The CR interface channel switching delay is set to 25\(\mu\)s. The wifi MAC standard is set to 802.11g with a rate of 54 Mbps.

### 4.4.1 Validation

The purpose of this evaluation is validate the CR extension by showing the throughput of a UDP flow with a constant data rate of 512 kbps or 62.5 KB/s from one CR to another. Both nodes are subject to PU activity that follows an on-off exponential distribution with average duration of 2 s and 10 s, respectively. Figure 4.5 shows the stream’s throughput at the sink. The gray areas indicate the times when the PU is active. In this simulation, the switching policy is set to never (i.e. nodes do not switch to a vacant channel when a PU is detected, instead, they continue sensing until the channel is vacant). This choice was made to emphasize the disruption of data whenever a PU is active. We can clearly see that the flow is disrupted every time a PU is active. We also observe that the throughput receives a spike whenever (a) the PU vacates the channel, and (b), whenever the nodes perform sensing for 100 ms. Both of these spikes are explained by the flushing of the MAC queue that accumulates packets while the nodes wait for the PU to vacate or when the nodes perform sensing.
4.4.2 CRE-NS3 overhead

In this evaluation, the nodes are placed in a topology as indicated by Figure 4.6(a). Each node sends constant rate data at 512 kbps to the immediate neighboring nodes as indicated in the figure by the arrows for a total of 250 s. A set of 20 simulation runs are executed for a non-CR network, and then again for a CR network. The total execution time (kernel and userspace in Linux’s time tool) and the maximum memory consumption using Valgrind’s massif tool [90] are then averaged over the 20 runs. Figure 4.7 shows the increase in both CPU utilization and memory consumption by our proposed module. The memory consumption increases by about 0.8 MB and the CPU execution time increases by about 10 s. The increase in both the execution and the memory consumption is mainly attributed to the fact that every CR node in the simulation has a combination of three total MAC-PHY interfaces instead of one (See Figure 4.3).

4.4.3 CRE-NS3 vs. CRAHN

Due to its similar architecture, in this section, CRAHN [85] in ns-2 is evaluated against CRE-NS3 for the same network topology as depicted by Figure 4.6(b). The number of nodes \( n \) increases with every simulation run until \( n = 13 \) or a total number of nodes of \( 13 \times 13 = 169 \). If \( i \) and \( j \) are a node’s row and column index, then each node sends a stream to the nodes located at \( i + 1, j \) and \( i, j + 1 \) as depicted in the figure by the arrows. The stream that is sent from each node is a constant UDP flow with a bitrate of 512 kbps.
While the total CPU execution time is improved by a small margin in CRE-NS3 as shown in Figure 4.8, Figure 4.9 shows a substantial improvement (by an order of magnitude of 1) in the memory consumption for the same network topology.

4.5 Conclusion

In this chapter, we introduce the first cognitive radio extension for ns-3. This simulation environment is able to perform the major spectrum-related cognitive functions as well as offers flexible policy-based decision making. Substantial changes were also implemented to the transport, network, link and physical layers in ns-3 to incorporate the aforementioned CR features. Our evaluations also shows minimal processing and memory overhead when running the extension as opposed to the base ns-3, and lower memory and processing time when compared to that of the previous CR extensions developed for ns-2.
Figure 4.8: Total CPU execution time (kernel + userspace) for both CRE-NS3 and CRAHN

Figure 4.9: Memory utilization (MB) for both CRE-NS3 and CRAHN
Conclusion

This thesis has covered considerable breadth in terms of the protocol stack layers that were studied (e.g., transport, link and PHY), functionalities that were explored in these layers (e.g., rate control, interference alignment), as well as network architectural features (e.g., spectrum databases, multi hop ad hoc networks, multi-transceiver nodes). Research on CR is necessarily cross-layered, and as this dissertation reveals, changes in any one aspect of the protocol stack has far reaching consequences on the ultimate end-to-end operation.

The thesis presents protocol design, experimentation, analysis, as well incorporates devising new simulation frameworks. We anticipate that the utility of this thesis will extend beyond the included research results, allowing an active participation of students and researchers in this exciting area through simulation studies made possible by our ns-3 extension.

In summary, the thesis has focused on enhancing reliability, by both efficiently using the spectrum as well as identifying new spectrum that could be effectively used, without degradation in the quality of service of licensed users. The main results are included below.

In Chapter 2, we presented an equation-based transport protocol, TFRC-CR, which presented a fundamentally different control mechanism compared to the typically used TCP-based schemes. TFRC-CR was demonstrated to perform significantly better than its classical counterparts TFRC and TCP with respect to both PU protection and transmission efficiency (in 2 hop scenarios, throughput for TFRC-CR is 100 KB/s while TFRC is 8 KB/s and TCP is 12 KB/s). Our protocol does not assume any cross-layer feedback or input from intermediate nodes, which aligns it with the traditional end-to-end paradigm in the evolving space of transport layer research for multihop CR networks.

Chapter 3 describes a new paradigm for spectrum database access, which exploits the correlation that exists among two entirely different spectrum bands at specific locations, thereby improving the performance of local sensing. Results reveal about 23% reduction of queries, making it attractive for practical spectrum database deployments. In addition, we also explored a real-world interference alignment application that can reduce the control channel utilization (5 time slots versus 35 for without IA). This non-trivial approach can potentially open up a hitherto unexplored direction in spectrum sensing, and future
work will be focused on building the protocol suite that enables quick and efficient exchange of spectrum data between the vehicles and the BSs.

Finally, in Chapter 4, we introduce the first cognitive radio extension for ns-3. This simulation environment is able to perform the major spectrum-related cognitive functions as well as offers flexible policy-based decision making. Substantial changes were also implemented to the transport, network, link and physical layers in ns-3 to incorporate the aforementioned CR features. Our evaluations also shows minimal processing and memory overhead (an increase of 0.8 MB only) when running the extension as opposed to the base ns-3, and lower memory and processing time (memory is lower by an order of magnitude of 1) when compared to that of the previous CR extensions developed for ns-2.
Publications


Bibliography


[12] “FCC, establishment of interference temperature metric to quantify and manage interference and to expand available unlicensed operation in certain fixed mobile and satellite frequency bands,” ET Docket No. 03-237, DA 03-289, November 2003.


89


