

RICE UNIVERSITY

**A Protocol Class for Stealing Residual Bandwidth in
Uncoordinated Distributed Wireless Networks**

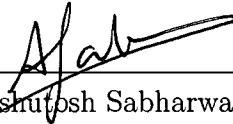
by

Scott David Novich

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

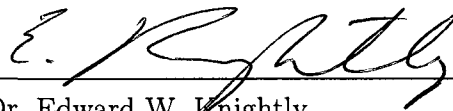
APPROVED, THESIS COMMITTEE:



Dr. Ashutosh Sabharwal, *Chair*
Assistant Professor of Electrical and
Computer Engineering



Dr. Behnaam Aazhang
J.S. Abercrombie Professor and Chair of
Electrical and Computer Engineering



Dr. Edward W. Knightly
Professor of Electrical and Computer
Engineering

HOUSTON, TEXAS

APRIL 2010

UMI Number: 1486066

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI 1486066

Copyright 2010 by ProQuest LLC.

All rights reserved. This edition of the work is protected against unauthorized copying under Title 17, United States Code.



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

ABSTRACT

A Protocol Class for Stealing Residual Bandwidth in Uncoordinated Distributed
Wireless Networks

by

Scott David Novich

The need for finding effective means of recycling spectrum is becoming increasingly apparent as the world becomes more crowded with wireless devices. While finding a policy solution to this problem will require years, “cognitive radio” is an immediately applicable technology-based solution. Our attention is focused on how a distributed uncoordinated cognitive group of “secondary” users (those with lower priority access to the spectrum) can push data through its network on a single band and in the presence of non-cognitive “primary” users (those with priority access to the spectrum). The main contribution is a novel class of cognitive radio protocols that accomplish this through feedback, where secondaries estimate residual bandwidth and adapt a performance-based parameter. This class of solutions is presented, its parameters are explored and a specific implementation is demonstrated with insights gained.

ACKNOWLEDGEMENTS

The author would like to thank the following people for all their input and support: my advisor Dr. Ashutosh Sabharwal, my colleagues and friends Nicholas Berthaume, Debashis Dash, Siddharth Gupta, David Kao, Christopher Mills, Pedro Santacruz and of course my amazingly supportive friends and family.

Contents

Abstract	ii
Acknowledgements	iii
1 Introduction	1
2 Problem Formulation and Related Work	3
2.1 Dealing with Lack of Knowledge	3
2.2 Primary Objective	6
2.3 Related Work	8
3 From Concept to Implementation	9
3.1 Determining F_p	10
3.1.1 Exploring F_p in a Primary Flow Without a Secondary Flow	11
3.1.2 Exploring F_p in a Primary Flow in the Presence a Secondary Flow	13
3.2 How to Sense and Determine T_{sense}	15
3.3 Finding the Genie-Aided Rate Region	20
3.4 Putting it all Together: Bit Burglar Dillinger	21
3.4.1 Protocol Implementation Overview	22
3.4.2 Implementation and Simulation	25
3.4.3 Analysis	27
4 Future Work and Conclusion	33
4.1 Future Work	33
4.1.1 The Rate Step Algorithm in Bit Burglar Dillinger	33

4.1.2	Bit Burglars That Learn	33
4.1.3	Sharing Knowledge Between Secondary Nodes	34
4.1.4	More Than One Secondary Flow or Primary Flow	34
4.2	Conclusion	35
A	WARP Implementation	36
	References	40

List of Figures

2.1	The Primary-Secondary Two-Link Topology	5
2.2	An overlay of 1) An approximate characterization of the achievable rate region (convex hull) 2) A characterization of the rate region for a coordinated centralized protocol - TDMA 3) A characterization of the rate region for an uncoordinated and distributed protocol - IEEE 802.11	5
2.3	A Sketch of the Protocol for a Generic Bit Burglar Implementation	7
3.1	Traffic Profile of Rate Growth for Packet Sizes of 100 Bytes	12
3.2	Traffic Profile of Rate Growth for Packet Sizes of 1 kilobyte	13
3.3	Jitter Versus Primary Rate (Without Secondary) for Packet Sizes of 100 Bytes	14
3.4	Jitter Versus Primary Rate (Without Secondary) for Packet Sizes of 1 kilobyte	15
3.5	Average Primary Throughput vs Secondary CBR Input Rate - 1 kilobyte Packets	16
3.6	Average Primary Delay vs Secondary CBR Input Rate - 1 kilobyte Packets	17
3.7	Sense Error Mean Versus Sense Length for CBR Traffic	18
3.8	Sense Error Standard Deviation Versus Sense Length for CBR Traffic	19
3.9	Queue Stability Operating Above Channel Capacity	21
3.10	Queue Stability Operating Below Channel Capacity	22
3.11	Empirical Rate-Region of Topology Based on Long-term Queue Stability	23
3.12	Protocol Example for an Unknown Rate Region	23
3.13	Protocol Run for Static Primary Rate	26
3.14	Protocol Run for Simple Primary Rate Changes	27

3.15 Protocol Run for Over Rate Region of Primary Rate Changes	28
3.16 Simple Traffic Profile: Rate Step Size versus Secondary Throughput .	29
3.17 Simple Traffic Profile: Rate Step Size versus Primary Jitter	30
3.18 Throughput versus $T_{coherence}$	31
3.19 Jitter versus $T_{coherence}$	32
A.1 Secondary Rate Adapting in BBP on WARP	37
A.2 Primary Throughput with Conservative BBP on WARP	38
A.3 Primary Throughput with Aggressive BBP on WARP	39

List of Tables

3.1 A Comparison of Statistics Between Protocol Runs with Varying Primary Rate Volatility	26
---	----

Introduction

We address the spectrum sharing problem in cognitive radio, where there exist networks of users that have priority access to the spectrum (“primary users”) as well as other networks that have lower-priority access to the spectrum (“secondary users”). These secondary users must somehow be “polite” towards the primary networks’ flow.

This problem is an important one, as it raises the issue of the rapid growth in the proliferation of wireless devices [1] (thanks to economies of scale and penetration in developing markets) juxtaposed with the way that spectrum is currently regulated. The applications for a solution to this problem are far-reaching - from tiered access networks (e.g., granting public use of a private band, but required a guaranteed QoS for certain services) to spectrum sharing between dense pockets of sensor networks. This is underscored by the fact that spectrum is incredibly expensive to lease, as much of it is allocated in a static manner (certain entities being granted full rights to certain bands) with little room granted for public use [2]. As a result, this places a severe damper on innovation in the wireless domain and especially so when placed in the context of small-to-medium-sized businesses.

To date, some technology-level (rather than public policy) solutions have been proposed that attempt to solve the problem. Many of these solutions are either

architecture-level [3, 4] or information theoretic [5] solutions that do not provide implementation details, or they fall into a class of implementable solutions called “multiple channel MAC protocols.” [6–11]

We propose a class of protocols, called “Bit Burglar Protocols” (BBPs), which seek to push secondary user data through an entire band - without the need for breaking it up into sub-channels where there exists a primary network. Secondary users attempt to push as much data through the network constrained to being allowed to affect the primary traffic or flow within some defined “reasonable” degree. This is accomplished by a feedback-based system, which takes inspiration from the ubiquitously-adopted Transmission Control Protocol (TCP), where secondary users estimate the residual bandwidth left over from primary users, through perturbing the channel, and then attempt to utilize what is left over. Through our exploration of this space, we come to an obvious but fundamental realization: **the more information the secondary has about the channel and the topology, the better the decisions it can make.**

The essential requirements for BBPs are few. First, secondary users must be able to sense the energy use on the channel and be able to differentiate between what is channel usage by the primary and the secondary. Second, secondary users must be able to adapt their traffic profile in some way (e.g., transmission rate, power, coding rate, etc.).

The primary contributions of this thesis are to present this new class of protocols, identify key parameters of the class that should be explored, explore some of the fundamental trade-offs of these parameters, implement a specific protocol case, and subsequently present new insights on how effective these protocols are as well as what can be done to improve and extend them.

Problem Formulation and Related Work

2.1 Dealing with Lack of Knowledge

We assume a legacy system. A primary network should not have to adapt its infrastructure to work with the secondary network. Our protocol will also be distributed and hence coordinated protocols like TDMA are not considered. There is no central controller or a dedicated sub-channel that the secondary network itself may use for keeping track of its constituents. Therefore, the protocol must be both distributed and uncoordinated. It is the goal of the secondary network to remain as invisible as possible to the primary while being able to transmit and receive its own information. As a result of to using uncoordinated flows, our capacity region resorts to a collision/interference channel, where the primary sends packets assuming the secondary does not exist. At the same time, there exists a challenge in that the secondary tries to send packets in such a way that the primary suffers no additional packet losses (and therefore rate loss) or take a hit in some other performance parameter such as jitter.

Next, assume that the primary channel has a capacity of C (without any interference) and the primary is sending at a rate $Rate_p < C$. If C , $Rate_p$, and the rate region

between the aggregate primary and secondary networks are known to the secondary flows, then the secondary can choose rate $Rate_s$ without causing any degradation in the primary flow's performance. This would be an example of a *genie-aided* protocol, where significant headway has been made by the likes of Devroye et al. [5]. However, in our situation, neither C nor $Rate_p$ are assumed known to the secondary network. As a result, the secondary flow does not know $Rate_s$, which under our protocol class is its maximum safe rate for transmission. Hence, our objective is to devise a protocol-based solution that tries to achieve $Rate_s$ by learning about $Rate_p$ and C .

Without genie-aided knowledge (or centralized coordination) in this scenario, there must then be overheads in obtaining this knowledge. To use a well-known colloquialism: "knowledge is power" (or performance). To illustrate this concept, consider a simple two-flow topology comprised of a primary and a secondary flow, where all nodes may hear one another and the channels are all additive white Gaussian noise (AWGN) (see Figure 2.1). It is known that for this topology, the 2-user interference channel, an achievable rate region is the Han-Kobayashi rate region [12] (see Figure 2.2), which is within 1-bit of optimal [13]. In contrast, the coordinated Time-Division-Multiple-Access (TDMA) protocol always underperforms. If coordination is not possible, one might use the uncoordinated and decentralized protocol (such as in the case of this work) IEEE 802.11. As the amount of initial knowledge by a user decreases, the achievable rate region becomes worse and worse. Within the context of our problem, the goal is to find the operating point $Rate_p$ on a rate region similar to IEEE 802.11. Further, without a genie, the performance of the secondary (at least in the short term) can at least be expected to be worse than the maximum achievable rate with a genie. Hence, the more knowledge that one is armed with at the outset, the better the performance that is achieved.

Without the aid of a genie and operating under decentralized and uncoordinated

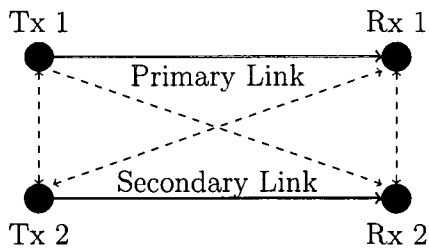


Figure 2.1: The Primary-Secondary Two-Link Topology

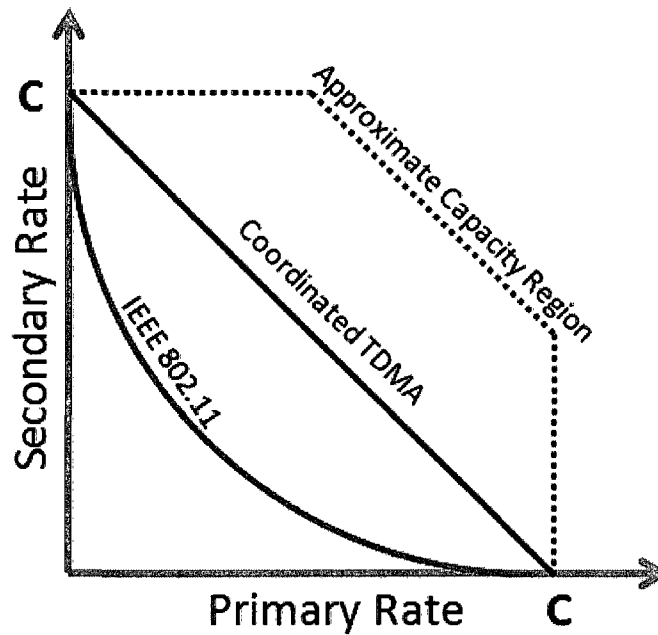


Figure 2.2: An overlay of 1) An approximate characterization of the achievable rate region (convex hull) 2) A characterization of the rate region for a coordinated centralized protocol - TDMA 3) A characterization of the rate region for an uncoordinated and distributed protocol - IEEE 802.11

restraints, we take inspiration from a higher-layer solution that also seeks to overcome the challenge of dealing with a lack of knowledge: Transmission Control Protocol (TCP). With a wireless link abstracted away, there can still exist an inherently unknown bottleneck in a network. If too much data is being transmitted across the data-layer channel, a sink may become backlogged and be forced to drop packets. There then exists a problem of maximizing transmission rate to match the bottleneck,

which is where TCP comes in.

TCP utilizes a feedback-based system through perturbation [14]. TCP adapts transmission rate in the form of a window-size: the amount of data a source sends consecutively without the need to receive an acknowledgment (ACK) back from a sink. Very roughly, TCP always attempts to increase a source's window size until it misses an ACK, in which case the window size is decreased. Similarly, in our scenario, as a primary link's characteristics and operating point are initially unknown, a BBP operating at the secondary should try to increase some performance parameter in its system until it affects the primary flow by some amount deemed unacceptable.

2.2 Primary Objective

The objective in BBPs is to achieve a system where the secondary attempts to maximize its own adaptable performance parameter(s) constrained to minimizing its effect on the primary flow with some known degree of accuracy,

$$\max_{\text{Prob}(|F_p - F'_p| \geq \epsilon) \leq \delta} R_s \quad (2.1)$$

We define R_s as the adaptable performance parameter of the secondary. F_p is some statistic (or possibly a set of statistics) of the aggregate primary flow where F'_p is this statistic (or set of statistics) as *sensed* by a secondary user. F_p can represent statistics such as (but not necessarily limited to) the average aggregate primary rate, average primary packet delay, or average primary jitter. The constraint can be understood as an outage constraint, as we are trying to control how often we end up decreasing the primary rate by more than ϵ . In the scope of this work, this optimization is not performed analytically, but rather an example protocol is presented that characterizes the optimization.

Thus there are two key steps involved in this class of protocols as are illustrated

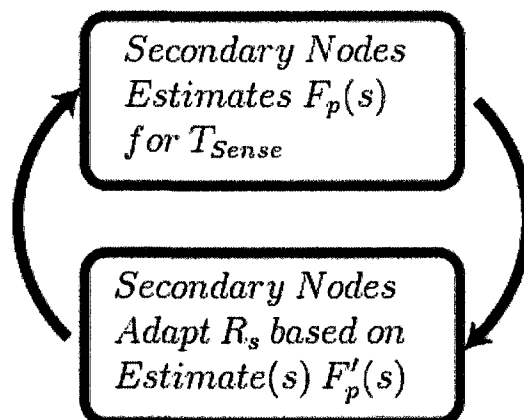


Figure 2.3: A Sketch of the Protocol for a Generic Bit Burglar Implementation in Figure 2.3.

1. Sensing F_p
2. Choosing R_s

This idea generates “ground zero” for our class. It provides us with some key questions we must answer in order to create an implementation:

- What is the most effective parameter to use for F_p ?
- When and how long do we sense the primary flow to obtain an estimate, F'_p ?
- How can we use this estimate F'_p to choose our Rate?

From the above questions, we are able to develop a specific protocol implementation of Bit Burglar class, where the answer to these questions will depend on our system’s model and network topologies, which in turn will provide a more specific protocol solution that one can implement. There are a myriad of directions (or sub-protocols) that can be taken with respect to how sensing occurs (e.g., how long the secondary should sense for) and how the secondary goes about choosing its new rates

based on F_p (e.g, the secondary may want to choose an initial rate lower than where it thinks it could operate).

2.3 Related Work

After extensive literature review, at the time of writing this thesis and as far as the Author knows, there has yet to be a protocol solution developed and implemented for a distributed, uncoordinated, and single-band scenario. As mentioned earlier, much work has been accomplished in the realm of multiple-channel-based protocol solutions [6–11], but within the context of single-band cognitive radio protocols, very little work has been done. The most similar work has been performed by Fan et al. [15] where an 802.11-based single-band rate-adaptive scheme called Limiting Transmission Probability (LTP) is proposed, modeled, and simulated. There are two key differences between this implementation and our work, however. First, it is assumed that the primary may relay requested throughput to the secondary (so the primary must adapt its protocol to work with the secondary). Second, a sensing scheme is not discussed, proposed, or implemented.

Similar to the work by Fan et al., Popovski et al. [16] presents a single-band solution where there is assumed to be a primary base-station that broadcasts the allowed data-rate for transmission to its users, which may be heard by the secondary network. Again, there exists an issue of the secondary being provided with information that it would normally need to sense.

Drifting further from our proposed solution, a Nash-equilibrium rate-adaptive transmission protocol has been developed by Huang and Krishnamurthy [17], which relies on a coordinated TDMA-solution between secondary users.

From Concept to Implementation

We focus on a simple case study within the scope of this thesis: a 2-Flow interference channel (see Figure 2.1) comprised of one primary flow and one secondary flow.

All nodes utilize an IEEE 802.11 MAC layer running TCP for the transport layer and a Constant-Bit-Rate (CBR) traffic source (unless otherwise noted). The channel is AWGN (unless otherwise noted). We choose TCP for its elastic properties, and IEEE 802.11 as it is a common protocol in coordination-free wireless networks. As discussed in the previous chapter, there are several questions that must be answered to go from concept to implementation:

1. Determination of an adaptable performance parameter R_s for the secondary (assumed to be transmission rate in this work).
2. Determination of a good statistic(s) for the primary flow F_p to sense.
3. Determination of an algorithm or formula for sensing F_p .
4. Determination of how long to sense - T_{sense} .
5. Determine of *when* and *how* to sense F_p and adapt R_s based on F'_p .

As such, we attack each issue sequentially, with exception of the first question,

where we choose R_s to be our transmission rate as controlled by the CBR source at the secondary.

3.1 Determining F_p

The first task is to determine what to sense. There is a subtle underlying implication: the statistic that one senses will ultimately represent the aspect of the primary flow that needs to remain as unaffected as possible, and is something that should ultimately be known *a priori*. In other words, it is assumed that the secondary must understand the requirements or application running at the primary before-entering the network. If our decisions are to be made based on sensing throughput, then one is in effect trying to guarantee not perturbing primary throughput beyond what is acceptable. The same could go for a statistic such as jitter, where some effect in variation of the inter-arrival time of packets must be kept to a minimum. As an example, if the primaries were to be running Voice-over-IP (VoIP) applications, the secondary may then need to guarantee both throughput and jitter for the primary.

For this work, we will attack the problem of determining F_p from a more empirical standpoint: the goal being to choose a single F_p that is easiest to sense and understand what occurs when the secondary flow begins to infringe on the primary flow. Two sets of simulations of the given topology and assumptions are performed in NS2, with no modifications made to the underlying architecture, under two conditions. First, without the presence of the secondary flow, the effects CBR rate on the primary flow's jitter (see Equation 3.2) and throughput (see Equation 3.1) will be explored. It is important to note that throughput and jitter need not be the only statistic that can be observed. Second, these effects will again be explored but for a static primary CBR rate and in the presence of an active secondary flow that is changing its CBR rate. The effect of end-to-end delay is not addressed as it is not possible for the secondary

to sense this statistic without having to communicate with the primary. Additionally, the effects of packet-size and the difference between a Gaussian and Rayleigh channel are addressed as part of the preliminary investigation.

$$T_p = \frac{\sum_{k=1}^{Number\ of\ Packets\ Received} Bytes\ in\ Packet\ k}{Length\ of\ Window} \quad (3.1)$$

$$J = \frac{\sum_{k=2}^{Packets\ Received-1} |IAT_k - IAT_{k-1}|}{Packets\ Received\ in\ Window} \quad (3.2)$$

3.1.1 Exploring F_p in a Primary Flow Without a Secondary Flow

Each of the following simulations (see Figures 3.1, 3.2, 3.3, and 3.4) are performed in NS2 with no modifications to the underlying architecture. The secondary source is turned off completely, and the primary source sweeps across its CBR rate changing every second (e.g., the CBR source rate at time t is $t * 1000$. So at $t = 50$ seconds, the CBR source rate is $50k$ packets/second). Each simulation is comprised of a single trial, such that the following presented results are not averaged.

It is immediately clear that smaller CBR packet sizes translate into greater overhead in the system, such that channel saturation occurs faster. For example, the observed throughput is the aggregate throughput for all data frames in the IEEE 802.11 MAC. For transmitting some set amount of data in the long-term, by decreasing the data frame payload size, there must be more control frames sent across the channel (such as response acknowledgments from the receiver) to achieve sending this total amount of data. Beyond this, there is no observable difference in the effect of packet-size in the system. Also as expected, the Rayleigh channel adds more variance to the throughput samples, although not very much until the channel becomes saturated.

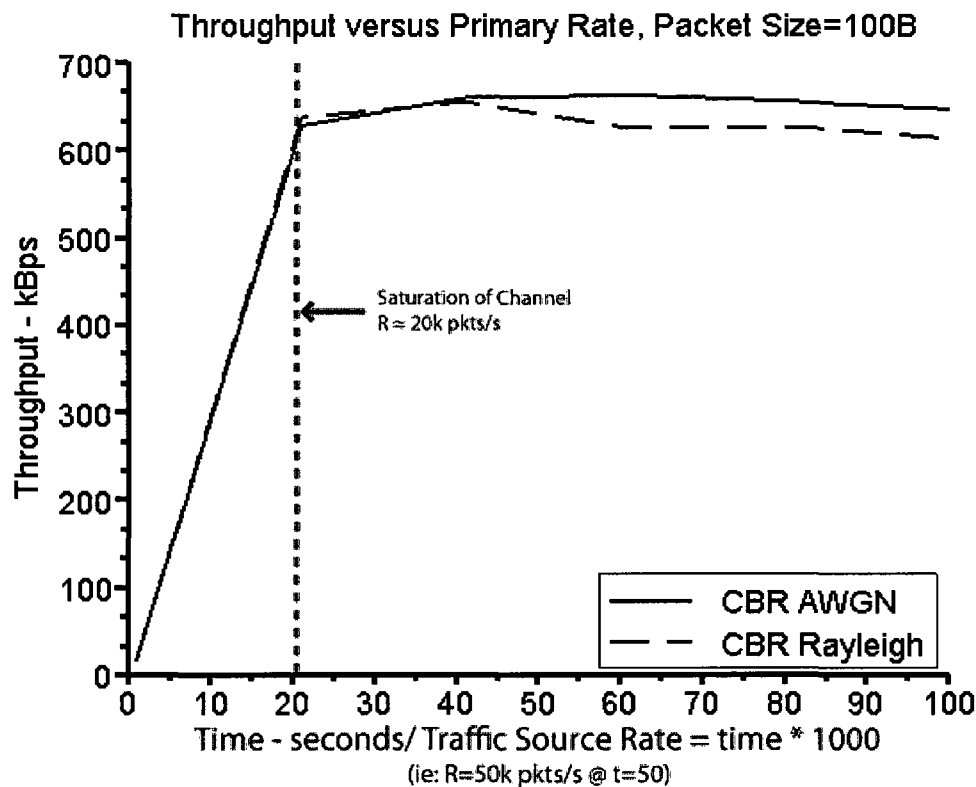


Figure 3.1: Traffic Profile of Rate Growth for Packet Sizes of 100 Bytes

Within the context of a BBP implementation, there is one key result: A directly proportional relationship exists between CBR source rate and (unaffected) throughput. This does not apply to jitter. The implication of this is seemingly simple, but important nonetheless: By choosing R_s to be the CBR rate of the traffic source, the secondary's "performance" is based solely on the amount of data it can push through the system. From an application standpoint, by only having control over the amount of data sent (and not the periodicity of the data), the secondary is limiting itself to the sorts of applications it can support (e.g., a secondary running FTP could reliably work under these conditions, but not necessarily VoIP).

Regardless, these results still do not provide a good intuition for which sense statistic F_p should be chosen. For that, we look to what happens to the primary flow

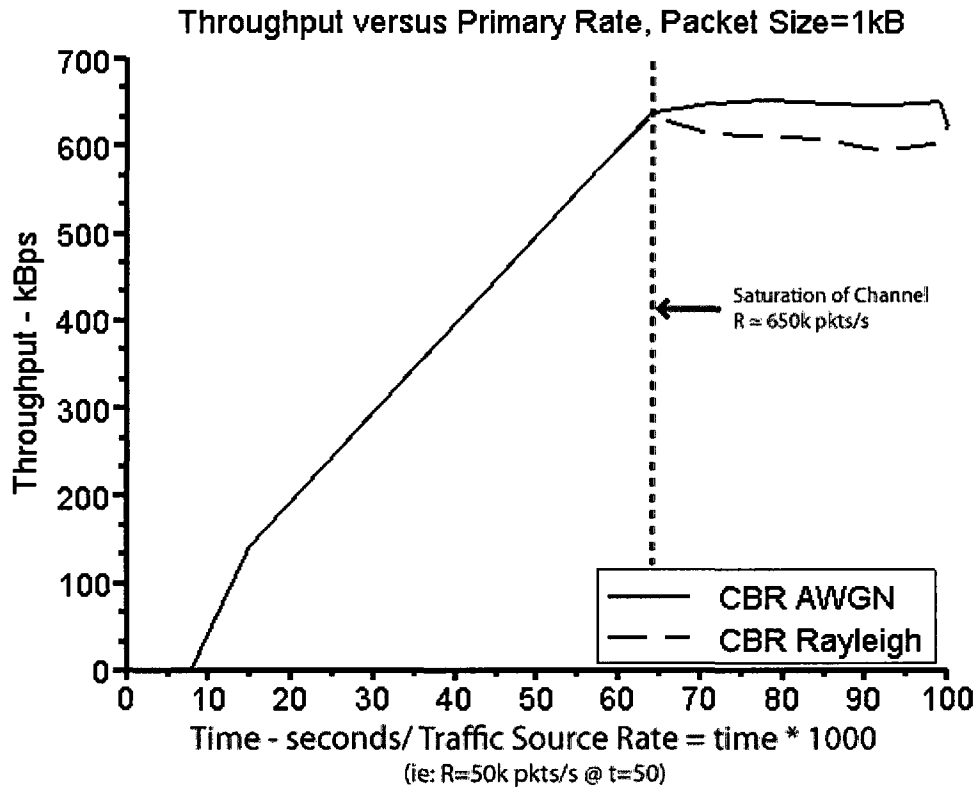


Figure 3.2: Traffic Profile of Rate Growth for Packet Sizes of 1 kilobyte

when it is affected by the secondary.

3.1.2 Exploring F_p in a Primary Flow in the Presence a Secondary Flow

Upon introducing the presence of a secondary flow, we perform a similar set of experiments in NS2. The results are found in Figures 3.5 and 3.6. In each of these simulations, the primary CBR rate is held static as the the rate at the secondary is swept over the rate region in a similar manner. Also, the channel is assumed to be AWGN and packet sizes are fixed to 1 kilobyte. The throughput and jitter calculation is performed on the *primary* flow. Due to the nature of random-access interference, the simulations are each run for 20 trials and averaged in an attempt to clean the

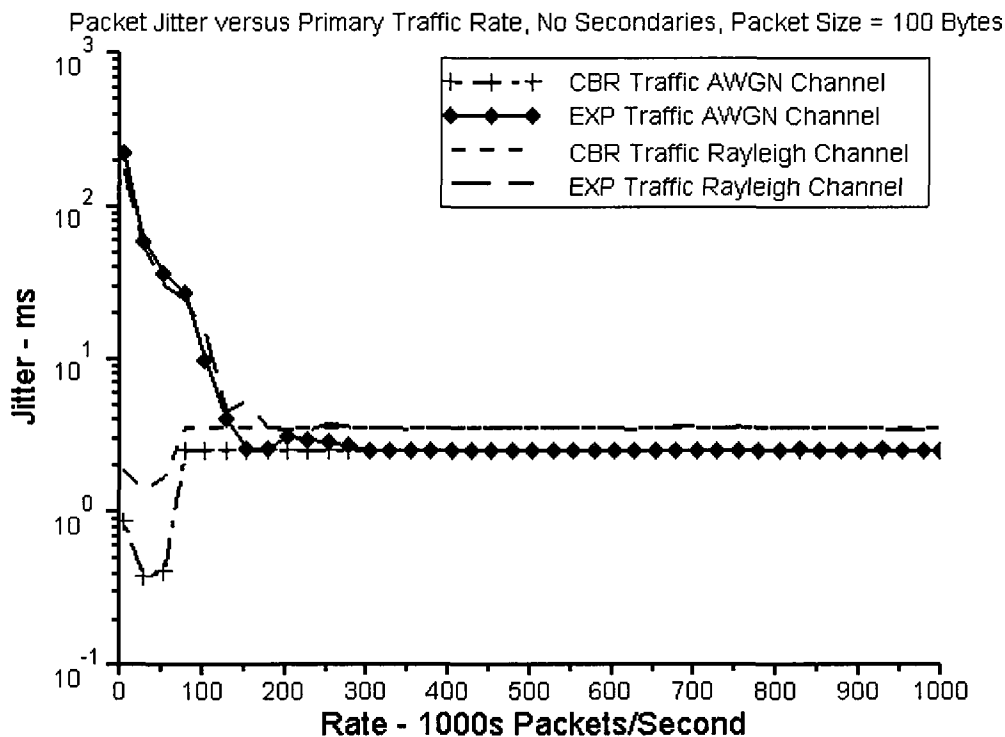


Figure 3.3: Jitter Versus Primary Rate (Without Secondary) for Packet Sizes of 100 Bytes

trends.

This set of simulations yields two key results. First, in the case of the secondary flow's effect on throughput (see Figure 3.5), the clean fairly monotonic trending indicates that throughput is a good candidate for F_p . Setting the secondary CBR rate to be greater than the residual bandwidth only causes a decrease in the primary's throughput. No increases in the primary throughput are witnessed under these conditions.

Second, the effect on primary jitter has a clear trend but yields obvious non-monotonic results, which may indicate that jitter by itself may be a challenging statistic to use in determining whether or not the primary flow has been affected. In other words, the secondary can not differentiate between an increase or decrease

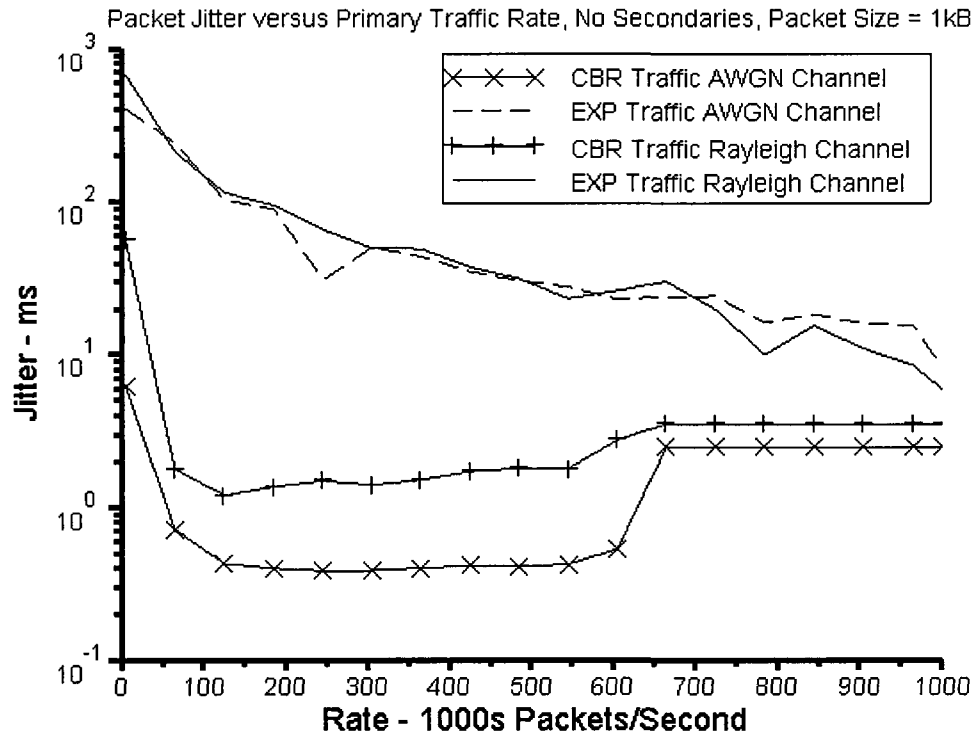


Figure 3.4: Jitter Versus Primary Rate (Without Secondary) for Packet Sizes of 1 kilobyte

in jitter for its decision making, in effect losing a degree of freedom over control. For example, the secondary may not be able to assume it can increase R_s based on witnessing a “temporary” decrease in jitter; there is a lack of correlation between R_s and its effect on the primary jitter. Subsequently, based on these two results, **throughput is selected as the statistic F_p for sensing.**

3.2 How to Sense and Determine T_{sense}

The algorithm for sensing F_p must reflect the true throughput (Equation 3.3) and is chosen in the spirit of being as unassuming as possible (Equation 3.4). This formula does not need to take into account the type of traffic being sent by the primary. For

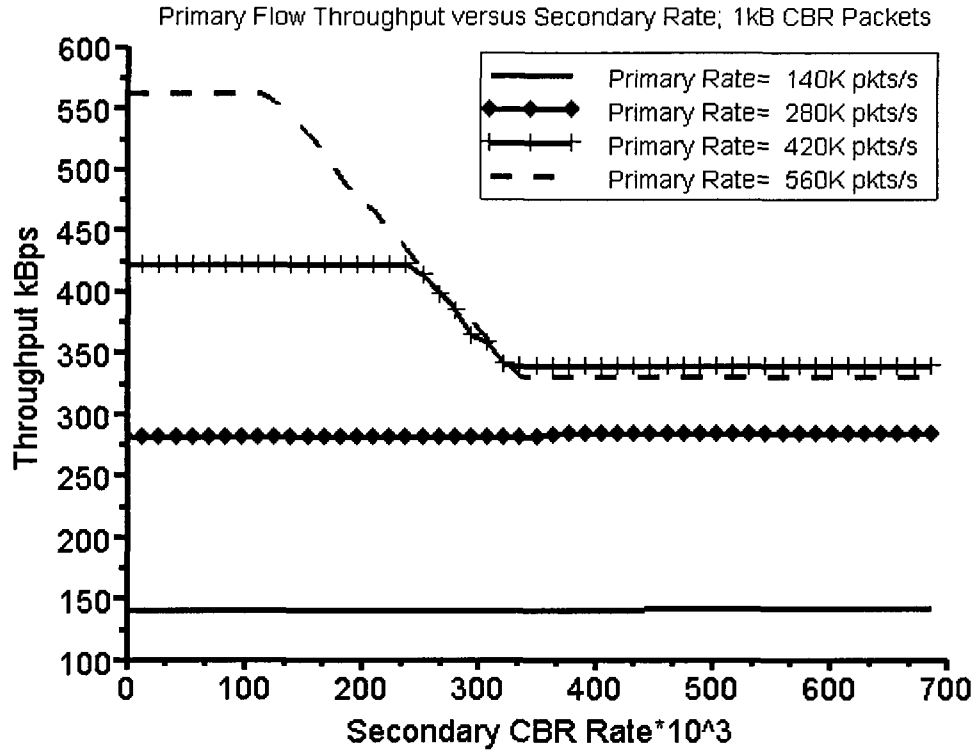


Figure 3.5: Average Primary Throughput vs Secondary CBR Input Rate - 1 kilobyte Packets

instance, the formula could be written in a manner such that the primary CBR rate could be extrapolated from knowing that the source traffic is perfectly periodic, yet we don't allow for this.

$$F_p = \lim_{t \rightarrow \infty} \frac{\sum_{k=1}^{\text{Number of Packets Received}} \text{Bytes in Packet } k}{t} \quad (3.3)$$

$$F'_p = \frac{\sum_{k=1}^{\text{Number of Packets Heard}} \text{Bytes in Packet } k}{T_{\text{sense}}} \quad (3.4)$$

With definitions for F_p and F'_p , one can then devise a metric for the error in the

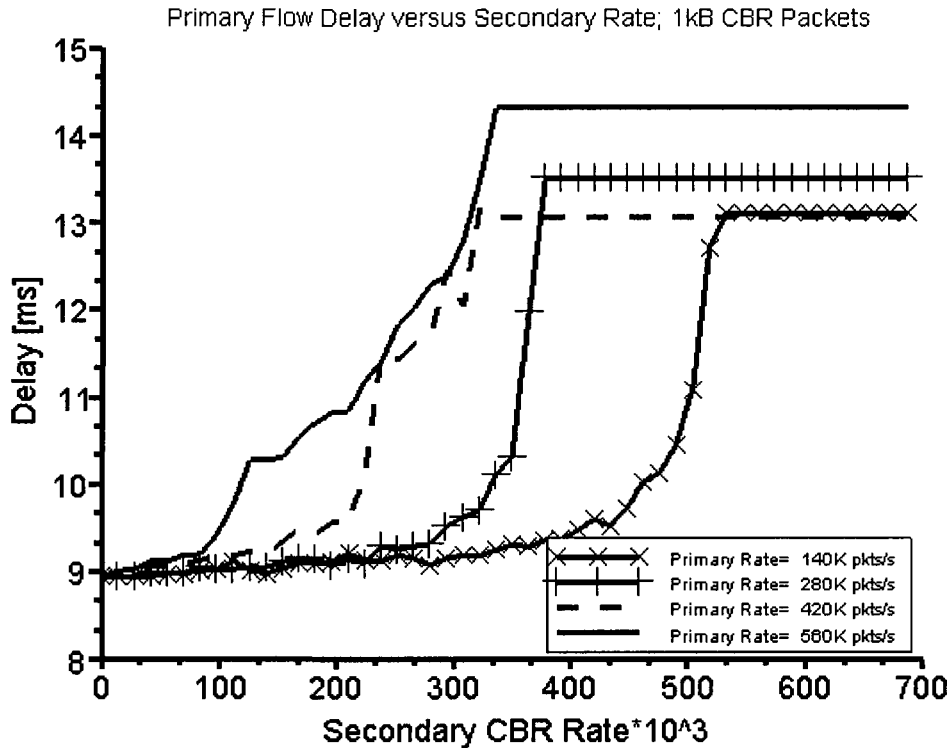


Figure 3.6: Average Primary Delay vs Secondary CBR Input Rate - 1 kilobyte Packets

sensing in (3.5):

$$Error = |F_p - F'_p| \quad (3.5)$$

Now that *how* to sense has been determined, the question of *how long* to sense (what is T_{sense}) must be addressed. To accomplish this, the effect of T_{sense} on the sense error is explored in NS2 as seen in figures 3.7 (the mean error versus T_{sense}) and 3.8 (the standard deviation in error versus T_{sense}). For each figure, the error metric is represented on the Y-axis, with T_{sense} on the X-axis. Each figure is comprised of a set of 1000 trials. Each trial is 800-seconds long and has a fixed CBR rate and T_{sense} with the secondary flow turned off. Within each trial, all results from each sensed F'_p are averaged. F_p for the error calculation is approximated by setting a large T_{sense} of

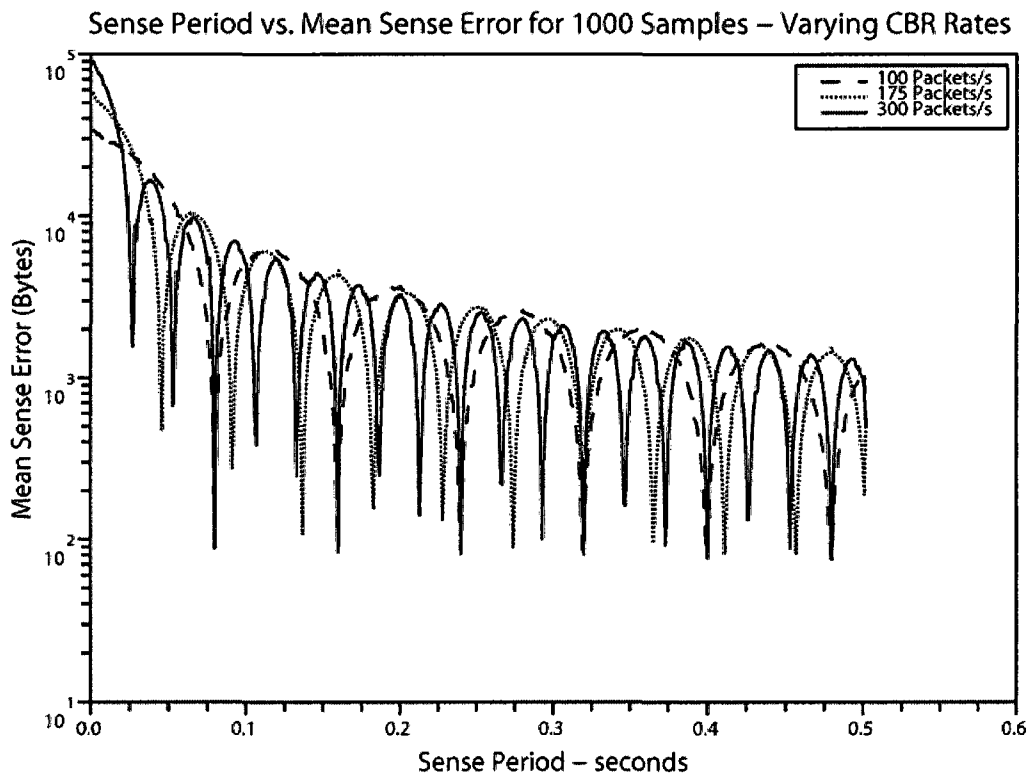


Figure 3.7: Sense Error Mean Versus Sense Length for CBR Traffic

20,000 seconds and simulating a trial that is at least that in length of time.

The results are what one would expect as seen in Figures 3.7 and 3.8. First, the mean error and its variance decrease faster than linearly, with the *envelope* decreasing monotonically. Second, as expected, there is a strong periodicity to the mean and standard deviation of error, which is due to our sense Equation 3.5 and the CBR traffic source. If the sense window size lines up with the period of outgoing traffic periodicity perfectly, the error drops to zero. Otherwise, there is an induced error due to the extra time being factored into the error formula. With respect to how the rate plays a role, the error mean and standard deviation is all comparable, but the periodicity of the error in sensing is of course different.

Within the context of the protocol, this scheme has promise referring back to

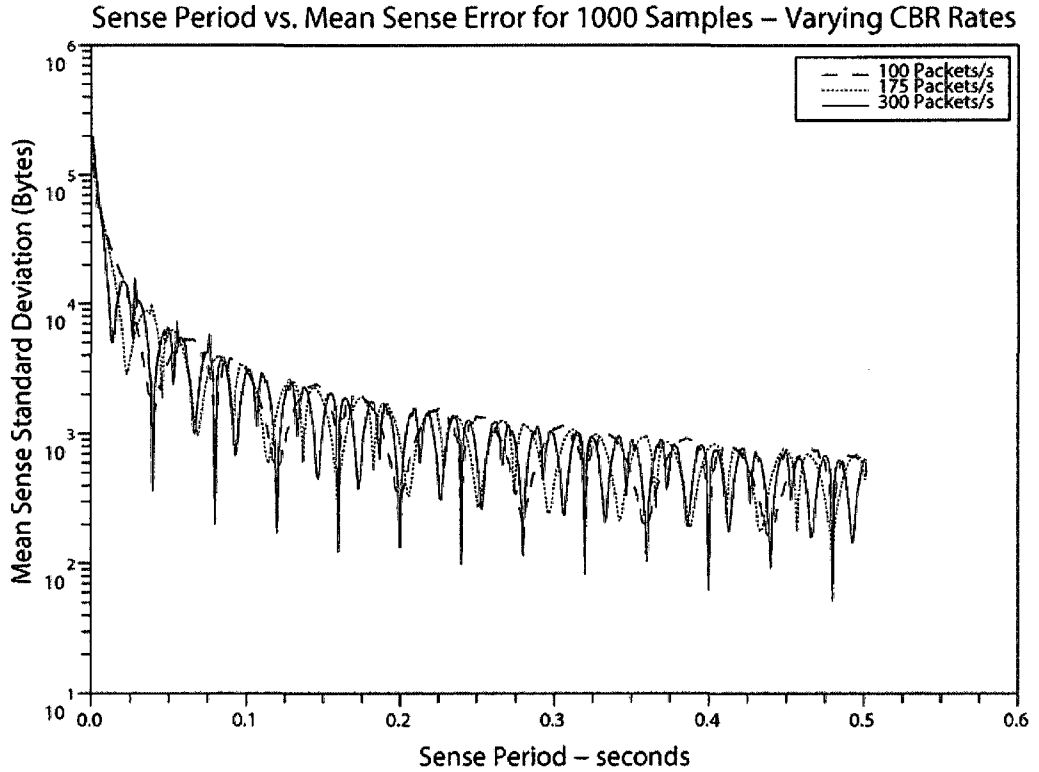


Figure 3.8: Sense Error Standard Deviation Versus Sense Length for CBR Traffic

Equation 2.1. If an error versus T_{sense} curve can be obtained offline, one may set a static T_{sense} that will bound this sensed error. Nonetheless, a trade off exists between the length of T_{sense} and the performance of the protocol. This trade off will ultimately be how quickly the secondary may adapt R_s , which can lead to the secondary having an adverse effect on the primary if T_{sense} is both too large or too small. If T_{sense} is too small, the secondary may miss out on hearing primary traffic, which translates into the secondary believing that it has more spectral opportunity than it really does. Conversely, if T_{sense} is too large, the primary may change its traffic profile too quickly for the secondary to appropriately adapt, which may either lead to lost opportunity or affecting the primary flow by more than what is allowed. One may be able to combat poor estimates by having the secondary make conservative choices in setting its rate.

In the end, we assume that the sense period is shorter than some coherence interval where the primary flow is not supposed to change.

3.3 Finding the Genie-Aided Rate Region

For purposes of comparing how well the implemented protocol holds up to a genie-aided case, the genie-aided rate region must be determined for the given system parameters and assumptions. Since F_p has been chosen to be throughput and R_s the CBR rate, the rate region must be a function of CBR rate region pairs that can achieve maximal combined throughput. As it is outside of the scope of the paper to analytically model this rate region as a function of the CBR rate pairs and other system parameters present, the rate region is obtained empirically. This is accomplished by sweeping across all possible rate pairs (for the primary and secondary CBR sources) in the system, and viewing their long-term effects on queue stability at each of the nodes in the two flows. If the combined flows are operating above the channel capacity, the queues at each node should become filled to a point where packets are being dropped in a near 1:1 ratio between total packets blocked by the queue to the total packets sent (see figure 3.10). Otherwise, this ratio should tend towards zero (see figure 3.9).

With this idea, a simulation that sweeps across all rate pairs is performed in NS2 to obtain each point for the empirical rate region: for a fixed primary rate, the maximum achieved secondary rate is chosen to be the rate simulated before which the long-term ratio of blocked-to-sent-packets tends towards one. A preliminary simulation of this averaged over 20 trials is presented in figure 3.11 overlaid with an expected curve fit. When implemented as a genie-aided Bit-Burglar protocol, a look-up table is used in the system, where given a primary rate, the acceptable secondary rate is returned. As the table is not continuous, if a given primary rate lies between two points on the

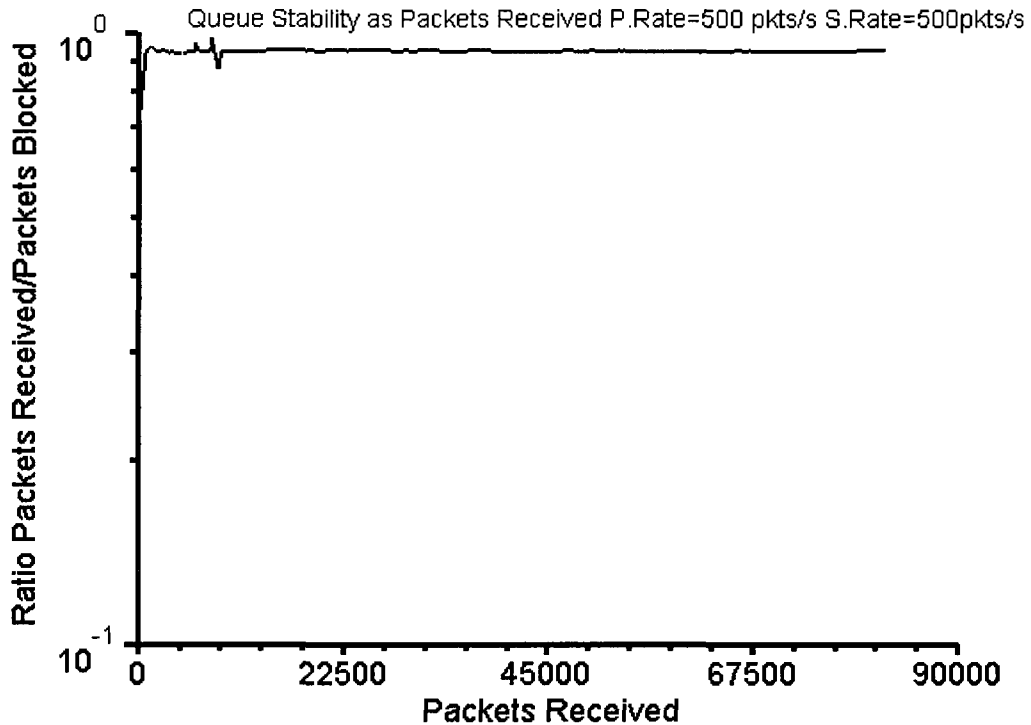


Figure 3.9: Queue Stability Operating Above Channel Capacity

table, then the closest found primary rate is used.

3.4 Putting it all Together: Bit Burglar Dillinger

Now that F_p , R_s , how to sense, and how to choose T_{sense} have been determined, the final step towards implementation is to determine how and when to perform the actions of sensing and adapting. Within the scope of this work, a possible example solution, “Bit Burglar Dillinger” (BBD), is provided and implemented, but no claims of the optimality of the solution are made.

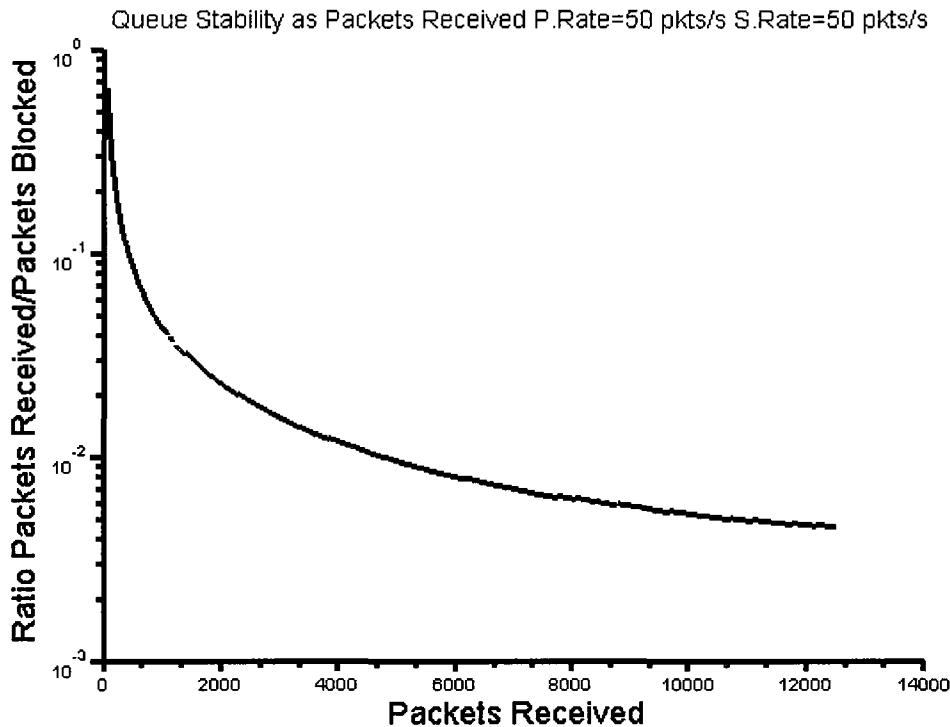


Figure 3.10: Queue Stability Operating Below Channel Capacity

3.4.1 Protocol Implementation Overview

For the implementation of BBD in general, it is worth noting that the secondary does not necessarily need to know anything about the primary flow other than being able to differentiate between detected energy caused by its own flow and the primary. The secondary must also introduce some heuristic parameters into its protocol design to be able to account for when it has affected the primary flow versus when the primary flow has changed its rate on its own accord. We also introduce the idea of a coherence time $T_{coherence}$, the length of time that the primary flow maintains a constant rate, and P_{step} , the minimum step size that the primary adjusts its rate. These parameters are introduced for purposes of preliminary analysis and thought experiments. Thus, Bit Burglar Dillinger is presented in Figure 3.12:

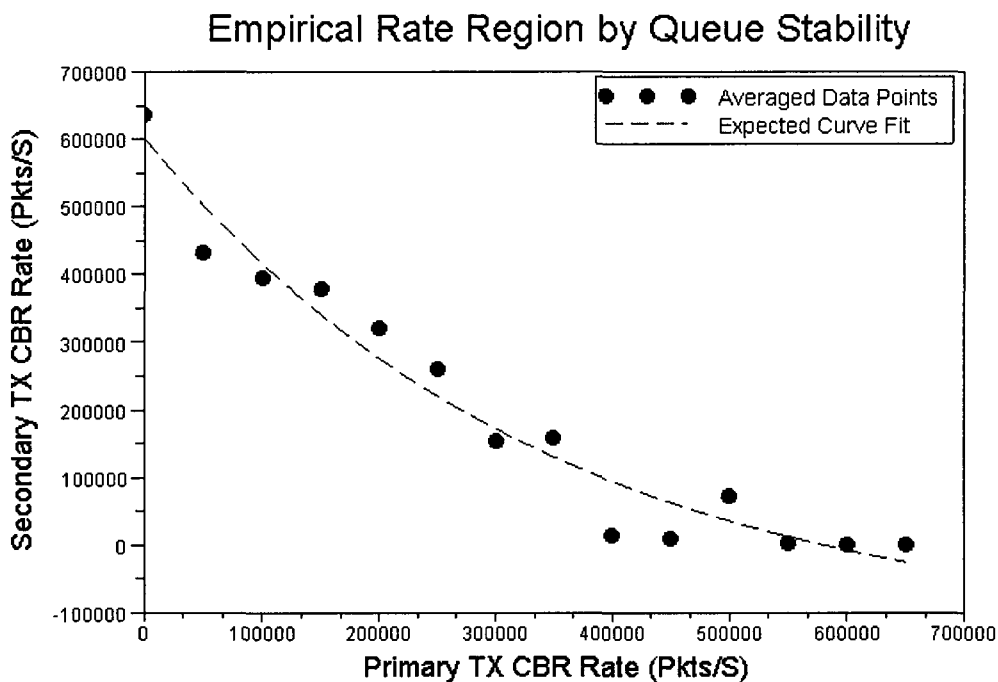


Figure 3.11: Empirical Rate-Region of Topology Based on Long-term Queue Stability

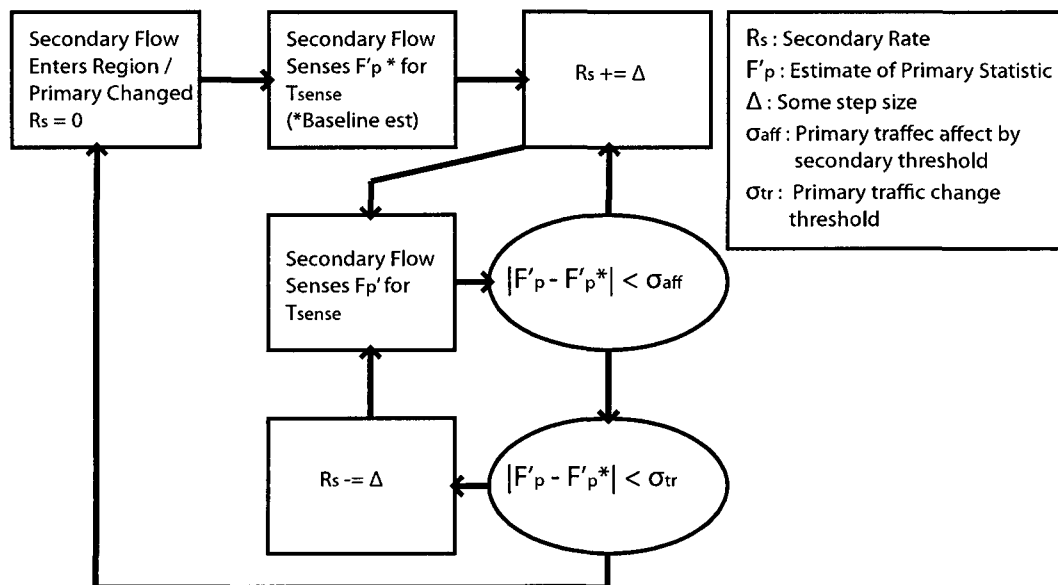


Figure 3.12: Protocol Example for an Unknown Rate Region

1. The secondary flow enters the network, and immediately sets R_s to 0.
2. The secondary flow then obtains a baseline estimate of F'_p* (this could be mean throughput, jitter, the amount of time some energy is present on the channel etc), by sensing the primary for T_{sense}
3. The secondary flow then attempts to increase R_s by some step size Δ .
4. The secondary flow obtains a new estimate of F'_p by sensing for T_{sense} .
5. If the difference in F'_p* and F'_p is lower than a threshold σ_{aff} , then the secondary increases R_s again by some step size δ .
6. If this difference crosses some threshold σ_{aff} , but is still under some greater threshold σ_{traff} , the secondary decreases R_s by some step size Δ .
7. If this difference surpasses σ_{traff} , the primary is considered to have changed on its own accord (or has been affected too much), and so the secondary resets R_s back to 0, and the protocol repeats.

As the rate region is not known, R_s must reset to zero if an intrinsic change in the primary flow is detected. As such, if $T_{coherence} < T_{converge}$, the secondary simply misses out on potential opportunity. On the other hand, if $T_{coherence} > T_{converge}$, the secondary may then begin to build a table of a converged R_s measurements for a given R_p . As mentioned earlier in the section on sensing, if $T_{coherence} < T_{sense}$, it is possible for the secondary to either affect the primary flow, if the primary rate increases, or for the secondary to miss out on potential opportunity if the primary rate decreases.

Another dimension that must be thought out it is the effect of P_{Step} on the static threshold σ_{aff} when $P_{Step} < \sigma_{aff}$. Again a similar issue arises of either missed opportunity or affecting the primary. In the situation where the sensed rate increases but still remains under σ_{aff} , there is an implication that F'_p* can be higher than it

actually is. However, if the secondary is still broadcasting, this can also imply that, although there is still additional capacity available, the primary may not be getting as much throughput as it intends. In the converse, where the sensed rate decreases, the secondary may have either affected the primary flow, or the primary has changed its rate intrinsically in which case the secondary missing out on additional transmission opportunities. With this understanding, it is assumed that T_{sense} and σ_{aff} are chosen in such a way that the potential lost transmission opportunities or effects on the primary flow are acceptable.

3.4.2 Implementation and Simulation

The protocol is implemented in NS2 by creating modified versions of the IEEE 802.11 MAC and CBR Traffic modules for the secondary nodes. A timer-interrupt is used at the IEEE 802.11 MAC layer for periodically recording F'_p in every T_{sense} interval. Similarly, a synchronized timer-interrupt is used in the modified CBR Traffic module for adapting the transmission rate as a function of F'_p and static thresholds (like σ_{aff}). A global vector of structs is used for the “memory” storage at each node. Each node is assigned an ID, which represents the address of the vector it is allowed to access. Each struct contains any sort of statistics (such as F_p) that each node tracks. For a non-genie-aided protocol, a node may only access its own struct. Three sets of simulations are performed and analyzed (with their genie-aided counterparts) to see how the protocol holds up to varying degrees of $T_{coherence}$. Regions are shaded to denote the lost potential throughput between BBD and the Genie-Aided case with perfect sensing. Results are presented in Figures 3.13, 3.14, and 3.15 and statistics on the results are presented in Table 3.1.

From this set of protocol runs in Figures 3.13, 3.14, and 3.15, two results are immediately apparent. First and most importantly, the protocol is successful in that it can push data through in an adaptive manner and within a known degree of accurate

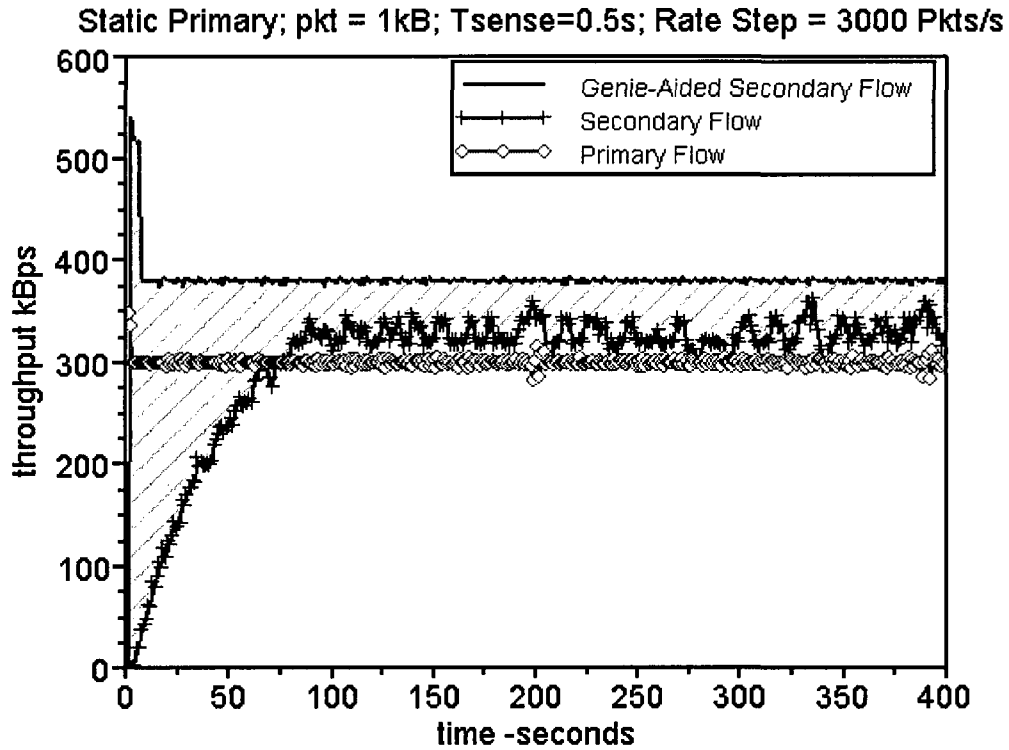


Figure 3.13: Protocol Run for Static Primary Rate

	Static Primary	Simple Primary Run	Complex Primary Run
Lost Sec. TP-kBps	81	179	281
Pri. TP Diff. (BBD)-kBps	0.00	0.24	0.00
Pri. TP Diff. (Genie)-kBps	0.00	0.14	0.00
Pri. Jitter Diff. (BBD)-ms	2.46	1.91	2.26
Pri. Jitter Diff. (Genie)-ms	2.70	4.96	6.52

Table 3.1: A Comparison of Statistics Between Protocol Runs with Varying Primary Rate Volatility

sensing (based on an exploration of T_{sense}). Second, from Figure 3.1 there appears to be a trend between the amount of lost potential throughput and the volatility of the primary traffic that will be explored further.

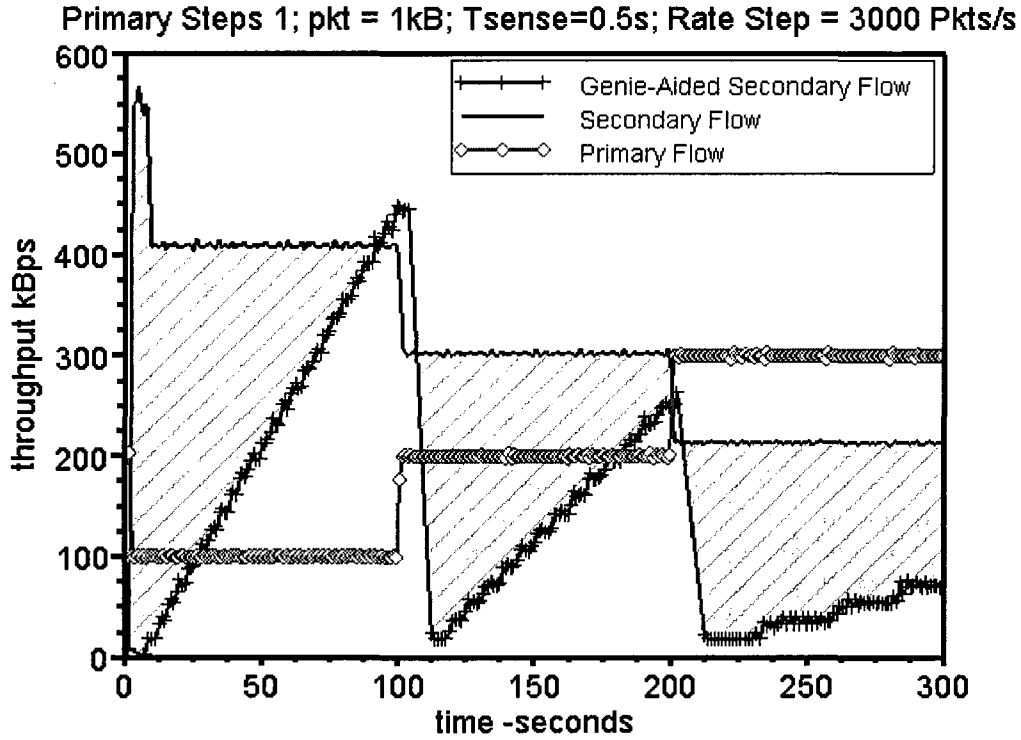


Figure 3.14: Protocol Run for Simple Primary Rate Changes

3.4.3 Analysis

There are two parameters worth exploring for better understanding the protocol and the system: the rate step size Δ (used by the secondary in BBD) and $T_{coherence}$, which measures how quickly the primary adapts its rate. For each set of simulations, a simulation trial is run for a given parameter (either Δ or $T_{coherence}$) with the results subsequently graphed.

First, we study Δ with respect to how it affects primary jitter, primary throughput, and secondary throughput. For the analysis, the simple primary traffic profile as seen in Figure 3.13 is used. Figure 3.16 demonstrates the relationship between Δ and both the primary and secondary throughput. This result demonstrates that, at least in the sense of throughput, the protocol is quite effective in ensuring that the

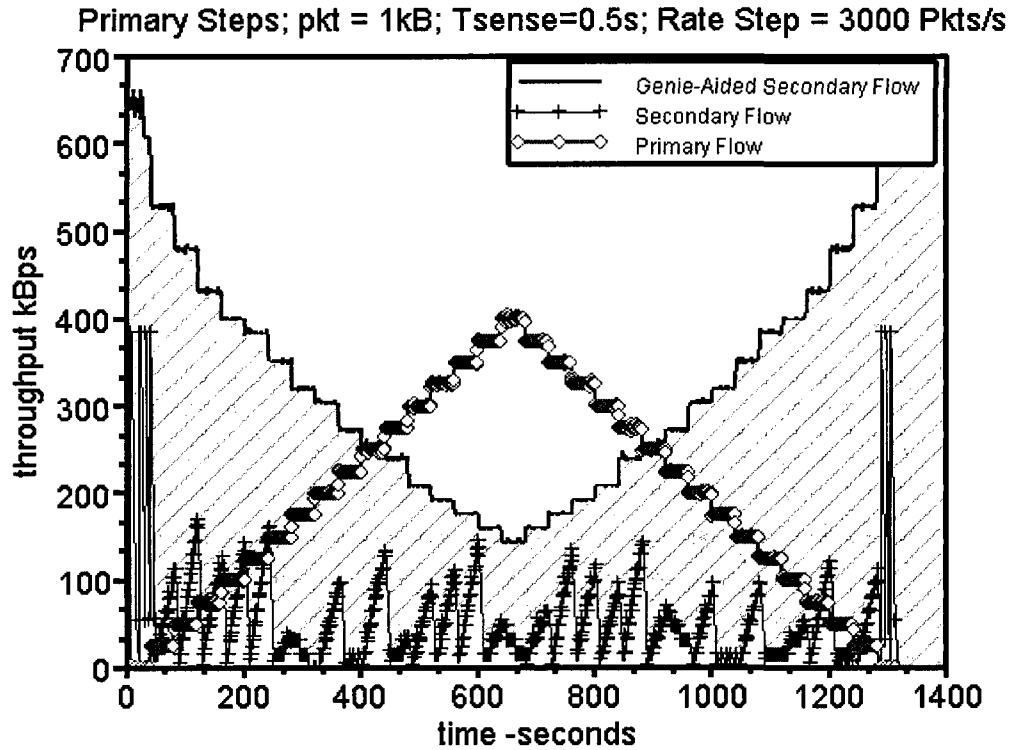


Figure 3.15: Protocol Run for Over Rate Region of Primary Rate Changes

primary flow can still push roughly the same amount of data through without getting crushed while the secondary seemingly is able to push a significant amount of data through its network as well. This is not without trade off, however: a jitter analysis in Figure 3.17 indicates that the primary flow's profile very rapidly becomes wildly affected.

Having explored the effect of Δ , we turn our attention to understanding the effects of $T_{coherence}$. In this scenario, a random-number generator that is scaled to channel capacity is introduced into the traffic generator used by the primary transmitter. For a total simulation time of 500 seconds, a random primary rate is chosen at each $T_{coherence}$ interval. Each trial is run 20 times such that the results are averaged. After 20 trials, a new value for $T_{coherence}$ is selected. The results for throughput and jitter can be seen in Figures 3.18 and 3.19.

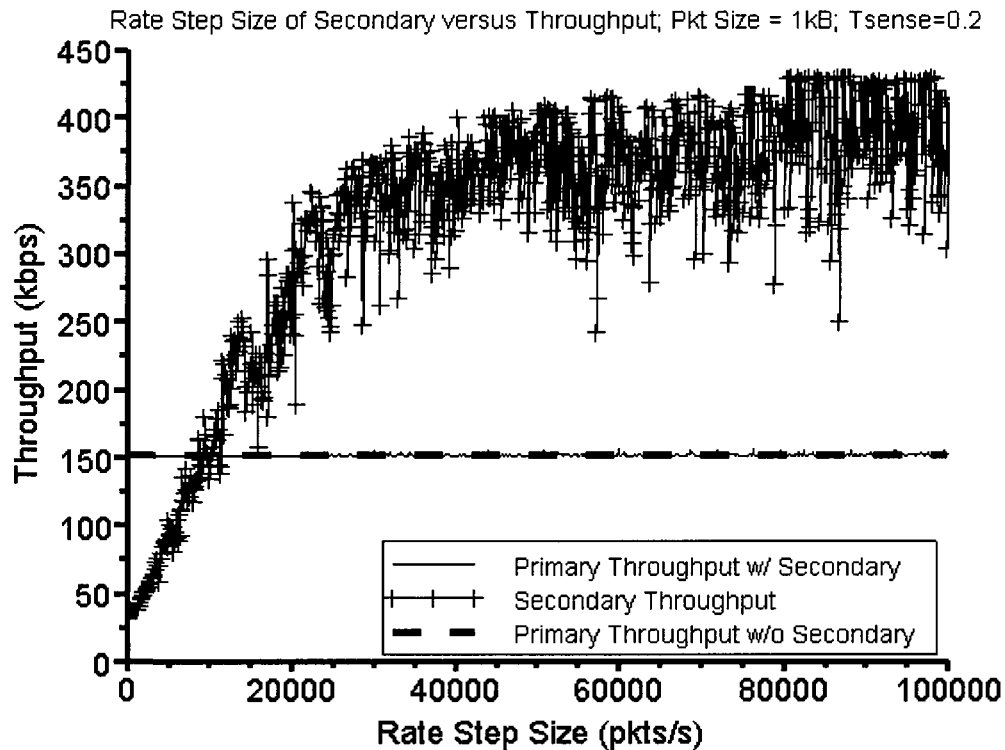


Figure 3.16: Simple Traffic Profile: Rate Step Size versus Secondary Throughput

As expected, from the results witnessed in Table 3.1, the larger that $T_{coherence}$ is, the more data the secondary is able to push through the system. Unsurprisingly, the secondary jitter also decreases due to it having less intrinsic rate changes and less instances of the system settling in 802.11. The implication from these results is that there is a fundamental trade off with Bit Burglar Dillinger, where it can only perform as well (obtain more throughput in its network) as the primary flow's volatility allows. Simply put, if the primary flow is volatile, performance in the amount of data the secondary can push through the network under BBD is greatly hindered. Nonetheless, there are potential ways to overcome this that will be addressed in Chapter 4.

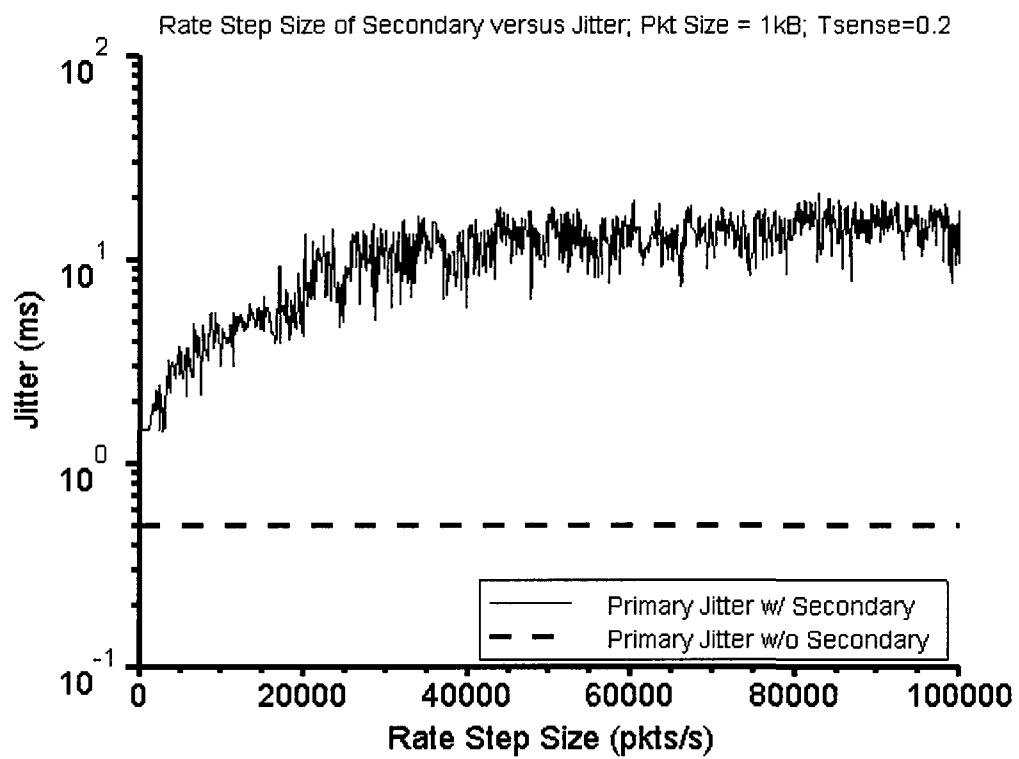


Figure 3.17: Simple Traffic Profile: Rate Step Size versus Primary Jitter

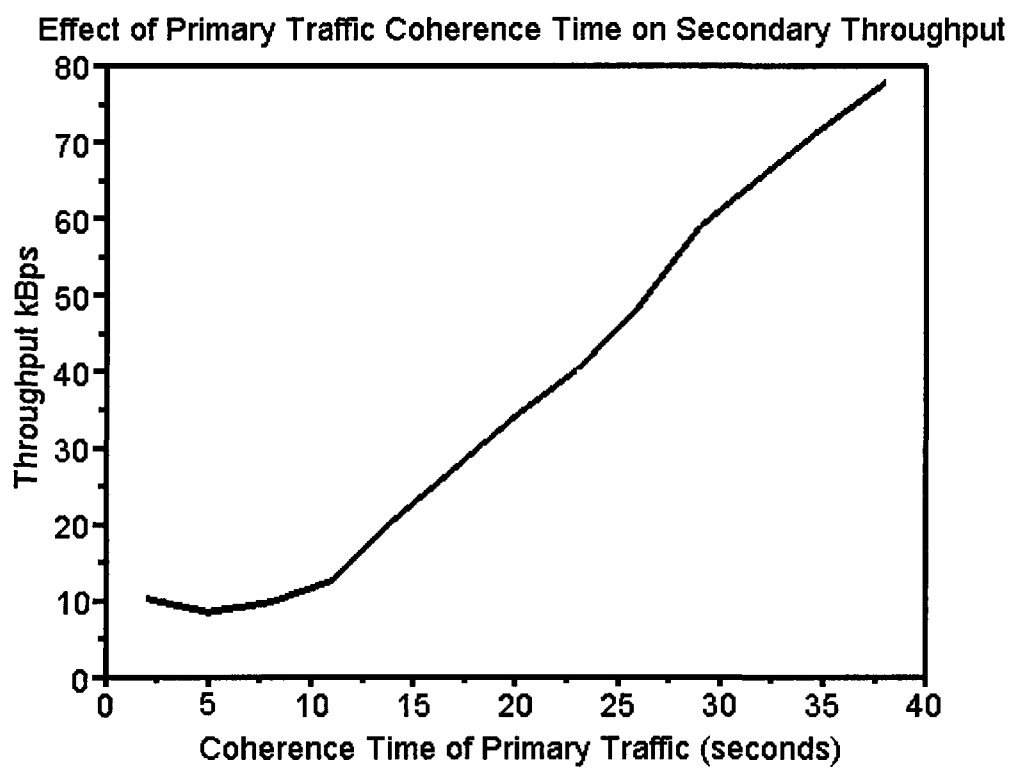


Figure 3.18: Throughput versus $T_{coherence}$

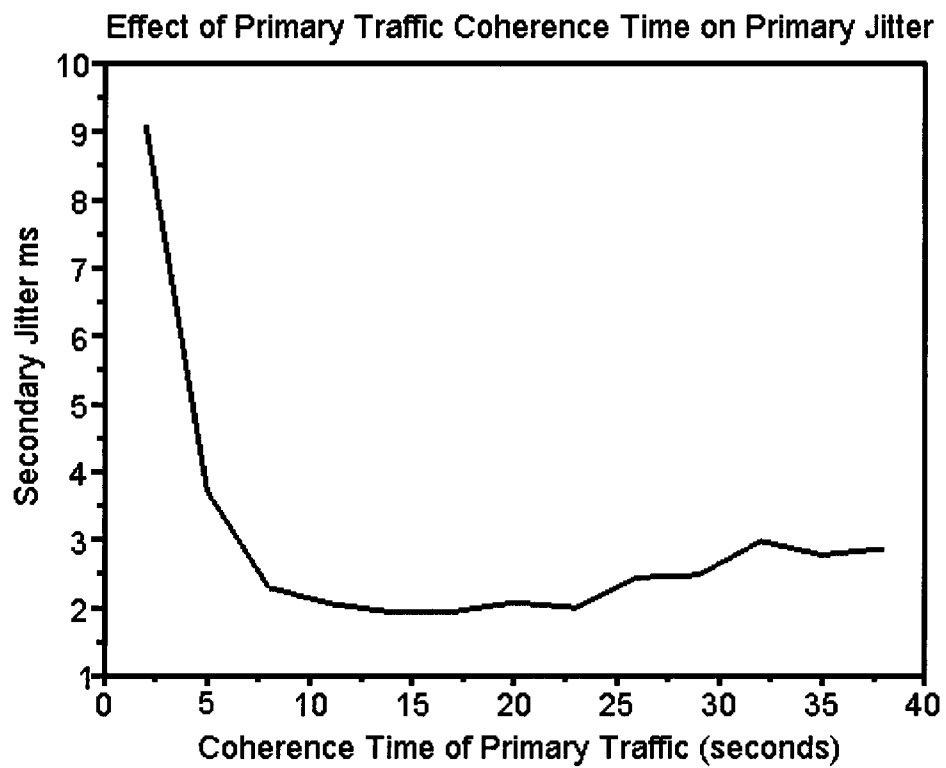


Figure 3.19: Jitter versus $T_{coherence}$

Future Work and Conclusion

4.1 Future Work

4.1.1 The Rate Step Algorithm in Bit Burglar Dillinger

In Chapter 3, when the Secondary transmitter increases its rate (for converging to residual capacity) or decreases it (due to affecting the primary flow), the secondary transmitter accomplishes this in strictly linear manner. This does not have to be the case: the secondary transmitter could, for example, use a linear growth and exponential backoff pattern (or vice-versa) for either affecting the primary flow less or converging faster. More work should be done to analyze this, and perhaps obtain an optimal rate-growth and back-off algorithm given constraints.

4.1.2 Bit Burglars That Learn

As mentioned in the results of analyzing Bit Burglar Dillinger, the implementation only performs as well as the volatility of the primary flow will allow. One possible (and simple) way to overcome this is to give secondary nodes a historical recollection of achieved CBR rates for a sensed primary rate. In effect, the secondaries can learn the rate-region with time to approach a solution comparable to the genie-aided case.

At the very least, this implies that a secondary transmitter need not completely reset its CBR rate to zero upon detecting an intrinsic change in primary traffic. This could translate into potentially large gains in throughput, especially when one refers back to the results of Figure 3.15 and Table 3.1.

4.1.3 Sharing Knowledge Between Secondary Nodes

Improvements in estimates can be gained if the secondary receiver is allowed to spy on the primary flow and share this information with the secondary transmitter. The primary benefit of this would be solving the hidden node problem. Another rudimentary example of how this could benefit the secondary flow is as follows: it can be assumed that whichever node hears more energy on the channel has a more accurate estimate of the channel use, as false-alarms in channel use (hearing energy when it's not present) do not occur. Thus, if the secondary receiver has collected more energy readings, this is knowledge it could share with the secondary transmitter to help improve its decision making. Within the context of BBD, sharing this information would aid in a more accurate estimate of how much the secondary has affected the primary, which translates into the primary being affected less.

4.1.4 More Than One Secondary Flow or Primary Flow

In the case of multiple flows, the bottleneck becomes the topology and who can hear whom. The simplest situation would be multiple secondary flows all within listening range of a singular primary flow. This problem would degrade into a resource sharing case between the secondaries, where BBD should work. An example addition would be that the secondaries could use some form of TDMA to decide who gets to use the available bandwidth at a given point in time.

A more complex situation would be dealing with multiple primaries and a single secondary flow, at which point the distribution of the aggregate primary flow would

change significantly, as well as the secondary's achievable rate region. At this point, work would need to be done that looks at devising a framework for getting a secondary traffic profile distribution to play "nice" given some arbitrary aggregate primary flow distribution. Alternatively, the secondary may keep track of F_p for each primary link and increase R_s up until the point where the first primary link is affected.

In the most complex situation, with multiple primary and secondary flows, none of which are guaranteed to hear each other, each secondary node would have to keep a table of all flows it could hear and then share this knowledge with other secondary nodes in a distributed fashion. Upon each secondary having knowledge of all flows, a secondary would then need to set R_s constrained to both the first primary flow it affects and sharing the residual bandwidth with other secondaries that could affect the flow.

4.2 Conclusion

While the status of the protocol framework is young, the preliminary BBP implementation indicates that this can provide a powerful solution to the spectrum sharing problem. It has been demonstrated that the protocol is able to successfully push secondary data through the system while only affecting the primary flow within some user-defined known error in sensing. There are numerous improvements to BBD that can be made, however, that would make for a significantly more robust protocol implementation. The framework must also be approached and modeled from a more theoretical perspective, as there are many potential new insights to be gained by applying optimization problems to the defined parameters.

WARP Implementation

As proof of concept, BBD has also been implemented on the Wireless open-Access Research Platform (WARP), although work has yet to be performed yet in analyzing the protocol's performance. An example of the protocol in action can be seen in Figure A.1. In the snapshot of the real-time graph, rate is measured by packet-release period, such that smaller values represent a higher transmission rate. The other two Figures, A.2 and A.3, show the real-time throughput tracking of the primary stream, which periodically cycles between high and low transmission rates, which are affected under both aggressive and conservative protocol conditions.

The WARP implementation is almost exactly the same compared to its NS2 counterpart except for two key differences. First, rate control is performed on the physical (PHY) layer, where a timer-interrupt is used to release packets of the PHY queue, so that the secondary rate is set by this timer's period (the smaller the period, the higher the rate). We opted to control the CBR traffic source itself from NS2, as it proved to be much easier to implement. Second, the protocol was tested on a UDP-based primary stream. In our tests, TCP traffic performed quite poorly over-the-air: as soon as the primary source became crushed, it appeared to take a very long time to recover, even if the secondary shut off its stream entirely.

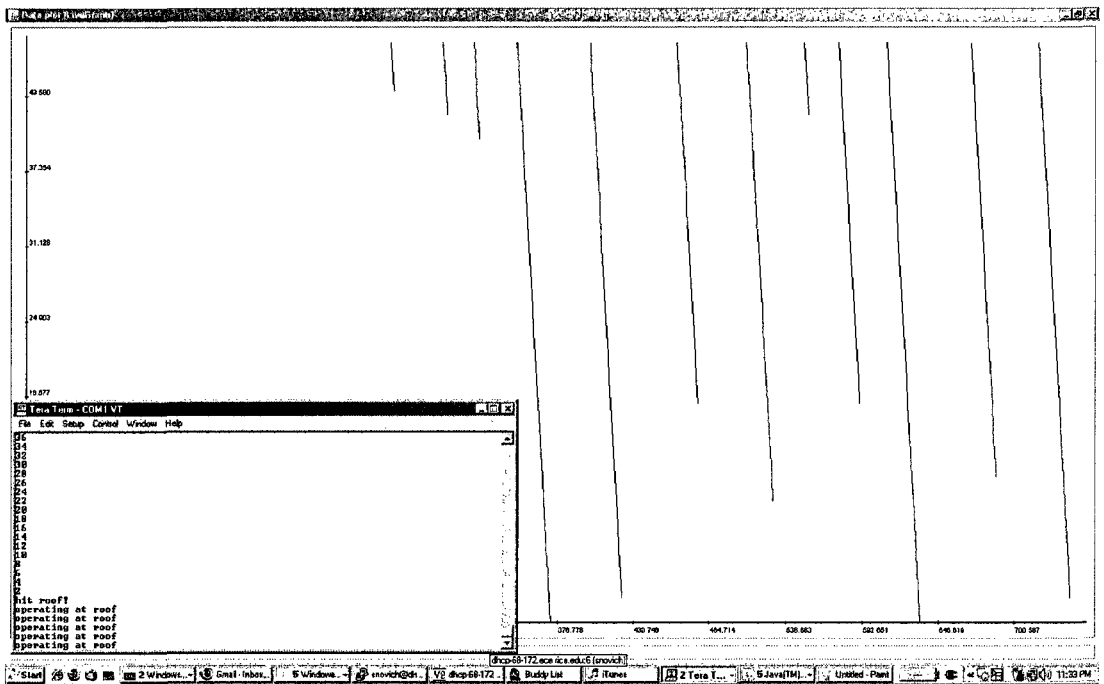


Figure A.1: Secondary Rate Adapting in BBP on WARP

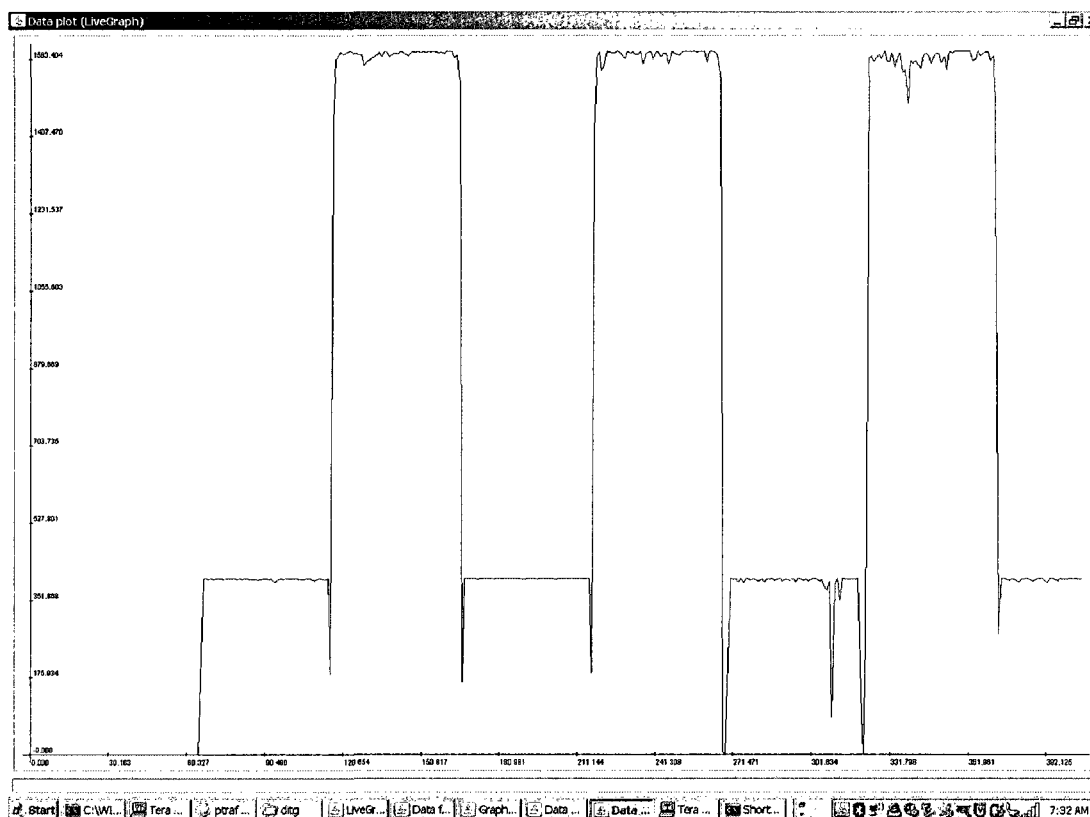


Figure A.2: Primary Throughput with Conservative BBP on WARP

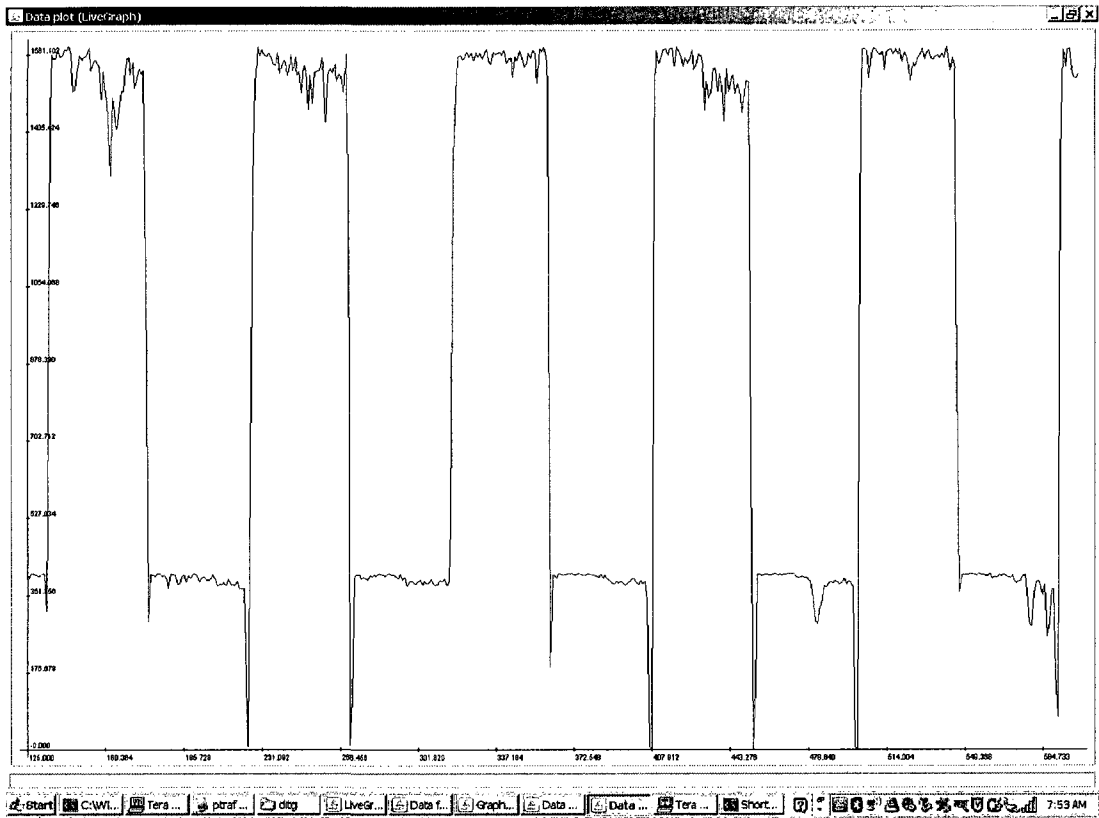


Figure A.3: Primary Throughput with Aggressive BBP on WARP

References

- [1] W.-F. Alliance. (2009, Jan.) Wi-fi® chipset sales grew 26 percent to 387 million in 2008. Press Release. [Online]. Available: http://www.wi-fi.org/pressroom_overview.php?newsid=770 1
- [2] R. H. Coase, “The federal communications commission,” *Journal of Law and Economics*, vol. 2, pp. 1–40, 1959. [Online]. Available: <http://www.jstor.org/stable/724927> 1
- [3] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, “Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey,” *Computer Networks*, vol. 50, no. 13, pp. 2127 – 2159, 2006. [Online]. Available: <http://www.sciencedirect.com/science/article/B6VRG-4K00C5M-4/2/3ad0ce34d1af95ffc509bf0ccaa455ab> 1
- [4] R. W. Brodersen, A. Wolisz, D. Cabric, and S. M. M. (uc Berkeley, “Corvus: A cognitive radio approach for usage of virtual unlicensed spectrum,” 2004. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.4475> 1
- [5] N. Devroye, P. Mitran, and V. Tarokh, “Achievable rates in cognitive radio channels,” *Information Theory, IEEE Transactions on*, vol. 52, no. 5, pp. 1813–1827, May 2006. 1, 2.1
- [6] J. Mo, H.-S. So, and J. Walrand, “Comparison of multichannel mac protocols,” *Mobile Computing, IEEE Transactions on*, vol. 7, no. 1, pp. 50–65, Jan. 2008. 1, 2.3
- [7] A. Sabharwal, A. Khoshnevis, and E. Knightly, “Opportunistic spectral usage: Bounds and a multi-band csma/ca protocol,” *Networking, IEEE/ACM Transactions on*, vol. 15, no. 3, pp. 533–545, June 2007.
- [8] A. Chia-Chun Hsu, D. Weit, and C.-C. Kuo, “A cognitive mac protocol using statistical channel allocation for wireless ad-hoc networks,” *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE*, pp. 105–110, 11-15 March 2007.

-
- [9] B. Hamdaoui and K. Shin, "Os-mac: An efficient mac protocol for spectrum-agile wireless networks," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 8, pp. 915–930, aug. 2008.
- [10] H. Nan, T.-I. Hyon, and S.-J. Yoo, "Distributed coordinated spectrum sharing mac protocol for cognitive radio," in *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, april 2007, pp. 240–249.
- [11] I. Joe and S. Son, "Dynamic spectrum allocation mac protocol based on cognitive radio for qos support," in *FCST '08: Proceedings of the 2008 Japan-China Joint Workshop on Frontier of Computer Science and Technology*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 24–29. 1, 2.3
- [12] T. Han and K. Kobayashi, "A new achievable rate region for the interference channel," *Information Theory, IEEE Transactions on*, vol. 27, no. 1, pp. 49–60, jan 1981. 2.1
- [13] R. Etkin, D. Tse, and H. Wang, "Gaussian interference channel capacity to within one bit," *CoRR*, vol. abs/cs/0702045, 2007. 2.1
- [14] A. Kumar, D. Manjunath, and J. Kuri, *Communication Networking: An Analytical Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004. 2.1
- [15] H.-P. Fan, C.-C. Kung, and C.-F. Chou, "Adaptive transmission protocol for protection of primary users in cognitive radio," in *Communications, 2008. ICC '08. IEEE International Conference on*, may 2008, pp. 3203–3207. 2.3
- [16] P. Popovski, H. Yomo, K. Nishimori, and R. D. Taranto, "Rate adaptation for cognitive radio under interference from primary spectrum user," *CoRR*, vol. abs/0705.1227, 2007. 2.3
- [17] J. Huang and V. Krishnamurthy, "Transmission control in cognitive radio systems with latency constraints as a switching control dynamic game," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, dec. 2008, pp. 3823–3828. 2.3