DESIGN AND IMPLEMENTATION OF PHYSICAL LAYER NETWORK CODING PROTOCOLS

A Thesis

by

DUMEZIE MADUIKE

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

MASTER OF SCIENCE

August 2009

Major Subject: Electrical Engineering

DESIGN AND IMPLEMENTATION OF PHYSICAL LAYER NETWORK CODING PROTOCOLS

A Thesis

by

DUMEZIE MADUIKE

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

MASTER OF SCIENCE

August 2009

Major Subject: Electrical Engineering

ABSTRACT

Design and Implementation of Physical Layer

Network Coding Protocols. (August 2009)

Dumezie Maduike, B.S., Rutgers University, New Brunswick

Co–Chairs of Advisory Committee: Dr. Alex Sprintson
Dr. Henry Pfister

There has recently been growing interest in using physical layer network coding techniques to facilitate information transfer in wireless relay networks. The physical layer network coding technique takes advantage of the additive nature of wireless signals by allowing two terminals to transmit simultaneously to the relay node. This technique has several performance benefits, such as improving utilization and throughput of wireless channels and reducing delay.

In this thesis, we present an algorithm for joint decoding of two unsynchronized transmitters to a modulo-2 sum of their transmitted messages. We address the problems that arise when the boundaries of the signals do not align with each other and when their phases are not identical. Our approach uses a state-based Viterbi decoding scheme that takes into account the timing offsets between the interfering signals. As a future research plan, we plan to utilize software-defined radios (SDRs) as a testbed to show the practicality of our approach and to verify its performance. Our simulation studies show that the decoder performs well with the only degrading factor being the noise level in the channel.

To my professors, family and friends

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest thanks and appreciation towards my committee chairs, Dr. Henry Pfister and Dr. Alex Sprintson. Their ideas were the driving force for our research. Without their constant support and guidance along the way, I could not have accomplished what I set out to do in this thesis. I would also like to extend my gratitude towards my other committee members, Dr. Karen Butler-Purry and Dr. Tiffani Williams, for their advice and support. Special thanks go to my loving family, especially to my parents and siblings. Their unending words of support were integral in pushing me through the toughest stages of my thesis. To my closest friends who stood by me during this whole process, I am truly grateful for you all.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

A. Network Overview

In the world today, many communication systems are wireless or in the process of making a transition to wireless. As an example, the concept of wireless internet was quite uncommon a few decades ago. Most networks were wired and communication was limited through point-to-point or unicast links [1]. Wireless networking at home for the average person was also not a common thing. As time goes on, many improvements will be made to networking and the results should be noticeable and most importantly, satisfactory. The internet, whose beginnings can be traced back to the early 1960s when the telephone was the world's dominant communication network [1], has made tremendous improvements and become one of the most common forms of communication. There are two different types of networks, wired and wireless, where each one is unique in its method of operation.

To clarify this further, consider a user[1] who desires to communicate with multiple users at once in a wired network. He/she will need to perform this action through the closest hub or switch in the network which will route the data along the network to its final destination(s). This, however, does not allow the user to communicate with all its desired recipients at the same time. In the most basic model of data transmission, a user will initiate n transmissions for relaying data to n users in a network. To alleviate this restriction in the wired network, different routing schemes

The journal model is *IEEE Transactions on Automatic Control.*

[1]In this thesis, a user is referred to as a terminal or node in a network. Examples include computers, cell phones and other communication devices.

were implemented to increase network efficiency and throughput.

The multicast routing scheme is a feature of networks where instead of initiating multiple transmissions by the user to all of its recipients, copies of the message or data can be made. The copies are then routed along the way to the recipient thus increasing the efficiency of the network as a whole. This method is slightly more secure than routing schemes such as flooding in that not all the users in the network can listen in on or access what is being sent. The internet utilizes a class D multicast address which is a single identifier for the recipients of the multicast packet [1, 2]. Multicast is depicted in Fig. 1. The user U_1 initiates transmission to the selected users U_2, U_3 and U_4. The switches, which are for linking multiple users together, will form copies of the original packet P_1 as it is routed to the 3 end-users.

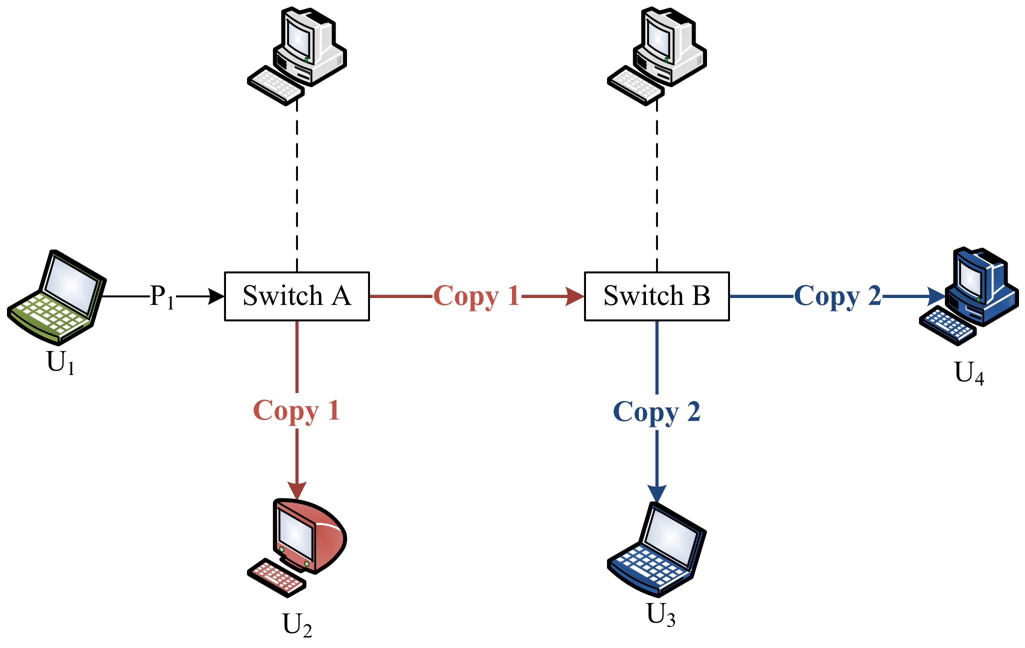

Fig. 1. Multicast routing scheme

Multicast allows for multiple users to receive a packet with only one transmission from the sender. However, there is a disadvantage in that the packet traverses many hops or nodes before reaching its final destination. Consider 3 nodes in a network (A, B and C) depicted in Fig. 2. The node A might be able to access C faster than being

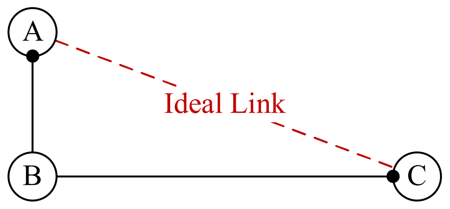

Fig. 2. 3 nodes in a network with ideal and non-ideal paths

routed through B based on its physical location coordinates. This is a restriction that multi-hop transmission presents.

The broadcast routing scheme, which is present in wireless networks, eliminates this issue and allows for A to communicate directly with C. In more general terms, broadcast allows a sender to deliver a packet to all its end users at one time if they are within range. Broadcast is similar to the anycast routing scheme which sends the packet to all its neighbors. Broadcast, however, does not restrict the user to send the packet to one neighbor at a time but to multiple neighbors at a time hence increasing network throughput. The broadcast scheme is depicted in Fig. 3. To make a parallel to an everyday situation, the home network is used once more. The transmitter at the top of Fig 3 acts as the router in the home while N_1 through N_4 are the desktop computers or laptops in the home. The nodes U_1 and U_2 are neighbors or people who are outside the home who the home network is not intended for.

An important note is that during broadcast, many users can listen in or eavesdrop on the transmission of the packet from the sender even if it is not intended for them. This poses both security and privacy issues [3, 4]. These security issues are alleviated

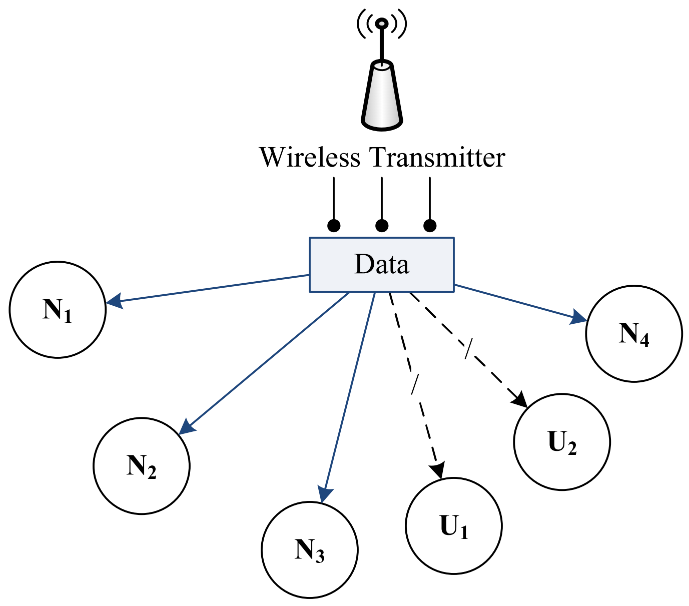

Fig. 3. Broadcast routing scheme

in these networks with different levels of data encryption to secure the network and thus allow only the intended users to decipher what was originally broadcasted. An everyday example of this is in home wireless networks. The router in your home constantly broadcast signals to its neighboring area thus allowing outsiders to enter the network and possibly hack into your system. This is an example of an unsecured network. Using a security protocol such as Wi-Fi Protected Access (WPA), you can restrict the access to the network to only the computers and devices in your home as long as those devices know your security key. This is why it is important to set up your home network as a secured network.

B. Relay Networks

A wireless relay network is a type of network in which there are central elements (relays), such as routers or switches, that act as a middle step in signal transmission.

The relays act as a ”handoff-medium” between users in the network and is integral in guaranteeing data accuracy during transmission and generally relays signals through broadcasting as it receives it or after it processes it. Consider the 3-node cooperative relay network [5, 6, 7, 8] depicted in Fig. 4. The user N_1 wishes to transmit data to

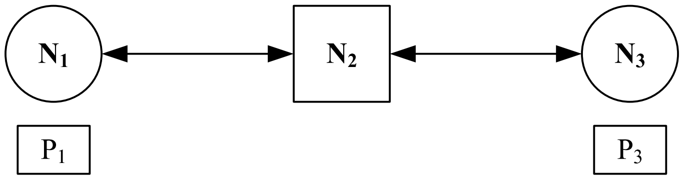

Fig. 4. Simple 3-node relay network

the user N_3 and vice versa. The two users may choose to do this through the relay N_2 due to a number of restrictive factors such as their distances from each other. There are ways that this can be done successfully with limited errors. However, there are trade-offs between error rates and efficiency. For example, N_1 can send its packet P_1 to N_2 which then forwards it to N_3. After this is done, N_3 repeats the same process with its packet P_3. This takes a total of 4 transmissions. Is there a way that this number can be reduced to 3 or maybe even 2? Network Coding [9, 10, 11, 12] aims to do this by utilizing the bits in the packets P_1 and P_3 or by mixing the signals sent by the users at the physical layer (analog waves) and obtaining the bit information from the carrier signal[2]. In wireless networks, signals are sent as electromagnetic (EM) waves. These waves interfere with each other constructively when their frequencies and signal strengths are of similar values. Even with the presence of interference and with the common notion that interference is harmful, the broadcast nature of the

[2]The carrier signal is what the bits in the packet get encoded with. Based on these bits, the carrier's phase, frequency or amplitude will be varied

network can still be utilized along with Network Coding to ensure data transmission reliability.

This thesis will discuss different ways that data transmission can be performed effectively when the two users in the network have signals that do not align in time or in phase. This is known as lack of symbol synchronization. It will also present solutions that reduce the negative effects that exist in this type of network such as noise and phase distortion. Channel noise affects the transmitted signal's phase and amplitude as it traverses the channel. Consider a signal to be transmitted in a wireless channel whose representation at time t is $y_{source}(t)$. Now consider a channel whose distortion at time t can be defined by the complex number $\delta(t)$. After traversing the channel, the resulting signal will be $y_{channel}(t) = \delta(t)y_{source}(t)$.

The 3-node relay network will be the main focus of discussion in this thesis. The design and implementation of these networks will be done using MATLAB and will aim to utilize software-defined radios (SDRs) in conjunction with GNU Radio [13] which is a free software toolkit for implementing these types of radios. This will give a more realistic view of the relay networks and highlight the effects of the wireless channels at the same time.

C. Description of Sections

The rest of this thesis will be organized as follows. Chapter II will provide background information and related work pertinent to Network Coding and relay networks and will also provide a description of SDRs and GNU Radio. Chapter III will discuss our implementations and results from the research. It will discuss the methods and tools utilized in arriving at our solutions as well as the simulation and analysis of previous work. Chapter IV will be for the summary and conclusions.

CHAPTER II

BACKGROUND AND PREVIOUS WORK

This chapter of the thesis will provide the pertinent background information needed in analyzing the methods developed and utilized in Chapter III. Signal properties in the wireless channel and all the assumptions made on its properties will be highlighted as well. The chapter will also describe the previous work done in this field of Network Coding (specifically in the 3-node relay network) and thus provide motivation and insight into the work developed in this thesis. Software-defined radios and GNU Radio will be introduced in this chapter with its applications and capabilities highlighted.

A. Relay Networks

A wireless relay network is a system of communication elements that possess transmit and receiving capabilities. Unique to this type of network is a central or a series of central elements (relays) whose main purpose is for gathering data from other elements and either broadcasting the received signal or performing signal processing operations on it before forwarding. As the word suggests, the relay acts as a *handoff medium* for signals. There are many uses for the relay in this type of network. It is useful when two transmitters need to communicate with each other and due to limiting factors, such as their relative distances from each other, they may not be able to transmit a strong enough signal to maintain reliable two-way communication with each other. Also, a relay can alleviate the effects of a fading channel due to its presence between two network elements [14]. A relay or even a series of relays can prove useful in a situation like this by receiving one's signal, amplifying or decoding it and then transmitting a signal to either another relay or to the final destination. These processes are referred to as the Amplify-and-Forward or the Decode-and-Forward method of transmission respectively. A simple relay network (3-node relay network) is depicted in Fig. 5. The nodes N_1 and N_3 are the senders who desire to communicate with each other by

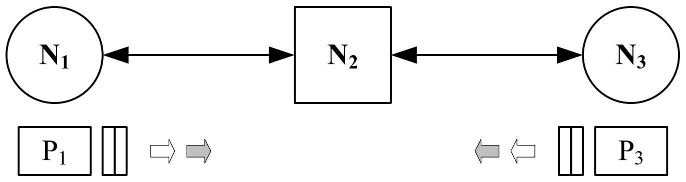

Fig. 5. Detailed 3-node relay network

sending their packets P_1 and P_3 to each other respectively while the node N_2 acts as a relay between them. It is necessary to consider how this can be done efficiently.

Let us now consider what N_2 really does with the signal it receives. Amplify-and-Forward allows all the decoding[1] and signal processing to be performed at the senders instead of at the relay hence lowering the load on the relay medium. As the name suggests, N_2 will amplify the signal it receives from the senders and broadcast it along the network. This is depicted in Fig. 6. Decode-and-forward is a technique

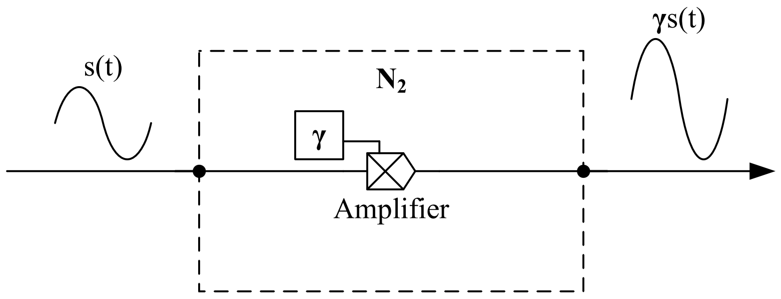

Fig. 6. Amplify-and-Forward system diagram

[1]Decoding in this thesis comprises of the demodulation process as well as the decision process that leads to the recovery of bits. Modulation can be done using schemes such as DPSK, PSK, etc.

in which the relay performs decoding on the received signal before broadcasting its own signal forward. A method such as this would be useful if a noisy channel with a low Signal-to-Noise Ratio (SNR) will affect the data in the signal if it were to be relayed further using Amplify-and-Forward. Zhang's paper [10] states that in cooperative transmission, the relay can decide on which transmission strategy is best suited for the network based on different SNR situations. The paper focuses on the Decode-and-Forward strategy.

Consider the situation in which N_1 transmits P_1 to N_2 with its destination being N_3 as Fig. 5 shows. Also, consider the channel distortion effect for the channel $N_1 \rightarrow N_2$ as the function $\delta_{12}(\cdot)$ and the channel effect for $N_2 \rightarrow N_3$ as $\delta_{23}(\cdot)$. If the Amplify-and-Forward scheme is utilized, the signal that N_3 will receive from the relay will be defined as

$$R_3 = \delta_{23}(\gamma\delta_{12}(P_1)) \tag{2.1}$$

where γ is the amplification factor at the relay. Eq. 2.1 shows that the signal that is received at N_3 (R_3) is P_1 with effects from two channels. If the noise power from the channels is high compared to the signal strength (low SNR), this might lead to improper decoding at the transmitters. Decode-and-Forward aims to limit the additive effects from the two channels. In this process, N_2 will decode $\delta_{12}(P_1)$ to form its own packet P_2 which it then sends to N_3 who then receives

$$R_3 = \delta_{23}(P_2) \tag{2.2}$$

P_2 will be chosen in such a way that N_3 can decode P_1 from it. In total, only one channel effect ($N_2 \rightarrow N_3$) will play a role in distorting the transmitted data. The methods discussed in this thesis utilize the Decode-and-Forward method of relaying.

We will now briefly highlight the effect of SNR on the wireless channel. An upper bound on the channel capacity of the 3-node relay network in a Gaussian channel with SNR snr, is given by $\frac{1}{2}\log(1+snr)$ [5, 15]. Channel capacity is a metric on how much information can be transmitted reliably over a communication channel. In [5], a method was developed using lattice codes and *joint physical layer* network coding to obtain a capacity rate of $\frac{1}{2}\log(\frac{1}{2}+snr)$ bits per transmitter. The channel capacity presented approaches the upper bound for higher SNR. This thesis will not highlight lattice codes as it is beyond the scope of the research. Fig. 7 depicts the channel capacity as a function of SNR.

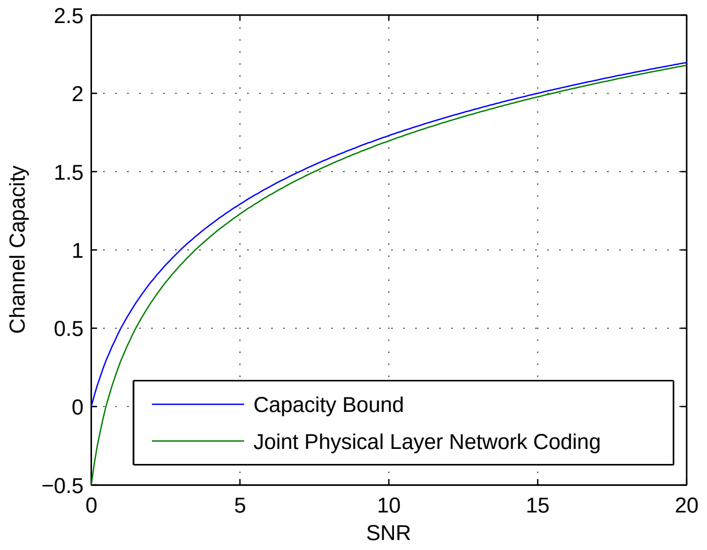

Fig. 7. Channel capacity vs. SNR

1. Methods of Transmission

In the standard 3-node relay network depicted in Fig. 5 in which N_1 wishes to send a packet to N_3 and vice versa, there are various ways with varying time allocations that this can be done. There are different methods in place that restrict interference between P_1 and P_3 and thus increase data accuracy. In addition to these methods, there are those that mix the signal at N_2 either at the bit level[2] or at the analog level (physical layer). Unique and critical to these methods is the number of transmission time slots required to successfully complete the transmissions. An important feature of the wireless channel is that the signals are received at the relay not in bits but as actual sinusoids. These sinusoids are subject to the effects of noise in the channel before reception. It is only after decoding that the bits and thus data can be pulled from the signals. Analog Network Coding (ANC) which is discussed in [11] takes advantage of the mixing properties of the wireless signals and uses it in increasing the throughput gain in the channel. There are three main methods, which are highlighted in [10, 11], that are used in successful data transmission in this type of network. These methods all fall under a category of communications known as Network Coding [9, 16].

a. Standard Transmission

Standard transmission in the 3-node relay network is quite simple and done in the following manner. N_1 sends P_1 to N_2 which follows by sending P_1 to N_3. After this, N_3 sends P_3 to N_2 which relays P_3 to N_1. The exchange of packets in this type of transmission is a 4-step process. This guarantees no interference between P_1 and P_3 and allows the relay to simply amplify (when necessary) the received signal to the other senders without performing any type of decoding. The decoding of the data from the analog signal is done on N_1's and N_3's end. This method of transmission is depicted in Fig. 8. Since throughput gains are desired in wireless channels, a method of transmission in this network that reduces the amount of time slots to complete the data exchange from 4 to something lower would be beneficial.

[2]Bit level is used to define the data bits that are extracted after decoding or demodulating the analog signal received at the relay.

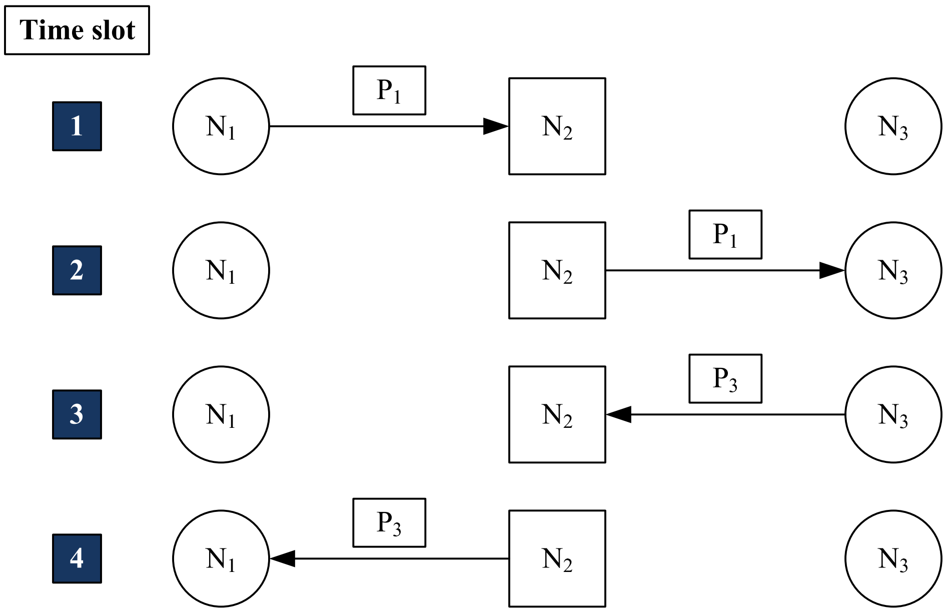

Fig. 8. Standard relay transmission

b. Digital Network Coding

Digital Network Coding (DNC) is a method of transmission that performs decoding at the relay. After decoding, the relay broadcasts a signal back to N_1 and N_3. In DNC, N_1 sends P_1 to N_2 and this is held at the relay. N_3 proceeds by sending P_3 to N_2. N_2 then demodulates both signals that it received and performs an XOR on them to produce its own packet P_2. After this, N_2 broadcasts P_2 back to the senders. This method is done in 3 time slots as opposed to the 4 it takes in standard transmission. This method utilizes the broadcast property of the wireless channel which allows for signals to be passed to multiple receivers at once. Since N_1 and N_3 know the packets that they sent, they can decode P_3 and P_1 respectively from the signal P_2. This method of transmission is depicted in Fig. 9. Table I shows how the decoding is done at the senders after receiving the XOR signal P_2 from N_2. The bits are presented in the table as data to be modulated or data that was demodulated.

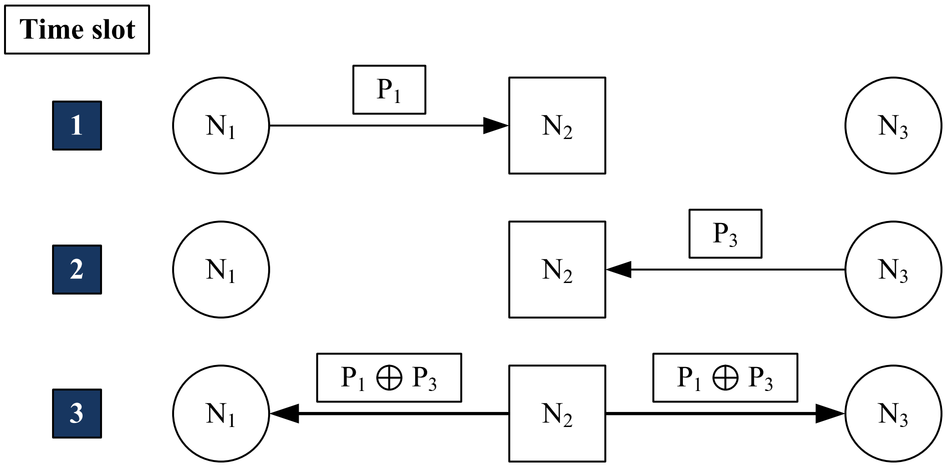

Fig. 9. Digital Network Coding relay transmission

Table I. Digital Network Coding Truth Table

N_1 sends	N_3 sends	N_2 sends	N_1 decodes	N_3 decodes
P_1	P_3	$P_2 = P_1 \oplus P_3$	$P_3' = P_1 \oplus P_2$	$P_1' = P_3 \oplus P_2$
0	0	0	0	0
0	1	1	1	0
1	0	1	0	1
1	1	0	1	1

Table I shows that after the senders XOR their original packet with P_2, they get the other sender's respective packet. The proof for this can be shown from N_3's end as follows:

$$
\begin{aligned}
P_3 \oplus P_2 &= P_3 \oplus (P_1 \oplus P_3) && (2.3)\\
&= P_3 \oplus (P_3 \oplus P_1) && (2.4)\\
&= (P_3 \oplus P_3) \oplus P_1 && (2.5)\\
&= 0 \oplus P_1 && (2.6)\\
&= P_1 && (2.7)
\end{aligned}
$$

This method of transmission works logically and can be applied in the wireless channel as well.

c. Analog Network Coding

DNC utilizes the broadcast feature of the wireless channel and performs XOR operations at the bit level. What if this same idea could be applied on the physical layer (analog level)? This is achieved through Analog Network Coding (ANC) [11] which utilizes wireless interference. Wireless interference in the 3-node network is something that is typically avoided, especially in the 802.11 channel, but is exploited by ANC when the signals from the packets of the two senders collide [11]. In the 802.11 network, different Time Division Multiple Access (TDMA) schemes are utilized to ensure that only one transmission is occurring during any time instant. In ANC, N_1 and N_3 transmit simultaneously to N_2 who receives the combined signal $P_1 + P_3$. N_2 then forwards, through broadcasting, the mixed signal back to N_1 and N_3. Fig. 10 shows how the transmission is done in this method without noise being factored in[3]. It should be noted that the complete transference of the packets P_1 and P_3 is done

[3]In the wireless channel, the mixed signal is subject to noise along with symbol level synchronization and phase synchronization issues.

in 2 time slots which delivers a throughput gain of 3/2 = 1.5 over DNC [11]. This

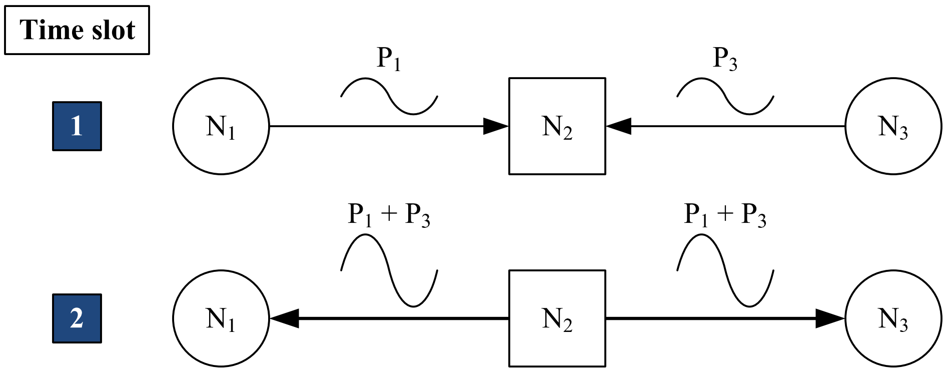

Fig. 10. Analog Network Coding relay transmission

method is useful in that the relay does not have to perform any decoding while it is necessary when DNC is utilized. The relay simply amplifies and forwards the mixed signal.

B. Signals in the Wireless Channel

Before delving into the main topic of this thesis, a brief background on signal properties in the wireless channel will be discussed. This is necessary in that it will highlight some of the assumptions that are made for these signals thus helping clarify the main methods utilized in this thesis. Data is transmitted in the wireless channel through electromagnetic (EM) waves [10] or sinusoids. When two of these waves interfere, their respective amplitudes and phases combine with each other. Fig. 11 depicts this property. For this reason mainly, 802.11 networks utilize carrier-sensing techniques that limit the amount of transmissions to one node at a time [17]. A sinusoid, which is a form in which these waves traverse the channel, is usually represented as a complex

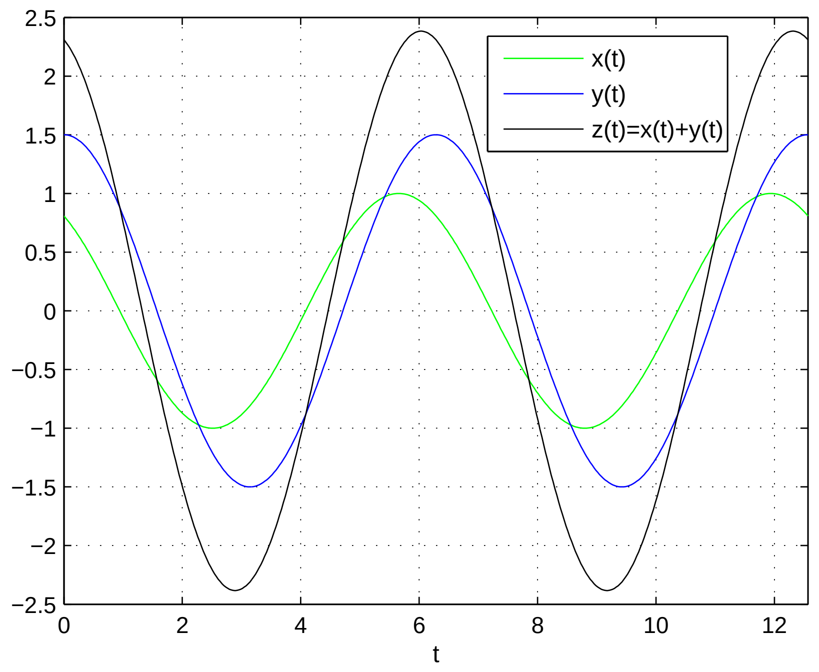

Fig. 11. Signal interference with $x(t) = \cos(t + .2\pi)$ and $y(t) = 1.5\cos(t)$

and discrete function of time [11] as

$$s[n] = A_s[n]e^{i\theta_s[n]} \tag{2.8}$$

This shows that each sample of the wireless signal at time n can be represented by a complex number. The value $A_s[n]$ is the amplitude of the n^{th} sample and $\theta_s[n]$ is the phase of the same sample. A note worth stating is that (2.8) stems from the definition of complex numbers which are represented as $z = re^{i\theta}$ and illustrated in Fig. 12. Since z can also be represented by the more common notation as $z = x + yi$,

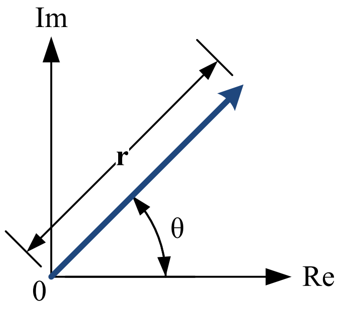

Fig. 12. Vector $z = re^{i\theta}$ in the complex plane

r and θ can be defined as

$$r = |z| = \sqrt{x^2 + y^2} \tag{2.9}$$

$$\theta = arg(z) = \angle z \tag{2.10}$$

1. Encoding Data in the Wireless Channel

How does a sender's packet or data, which contains strings of binary 1's and 0's, get encoded and sent over the wireless channel? The answer is through modulation. The way that the data (bits) is encoded depends on which modulation scheme is utilized. There are numerous schemes available with different defining characteristics such as error rates or data throughput rates. The modulation process involves bits being converted into carrier waves (can be sinusoids such as a cosine wave). The carrier waves are varied and modulated depending on what bit is present. Some encode the bits in the phase difference of consecutive samples $(\theta_s[n], \theta_s[n+1])$. This is known as differential encoding. After the bits are encoded and the carrier waves modulated, they are sent into the wireless channel. Fig. 13 depicts the typical modulation process from the binary bits through to the wireless channel.

We consider Binary Phase Shift Keying (BPSK) which is a modulation scheme

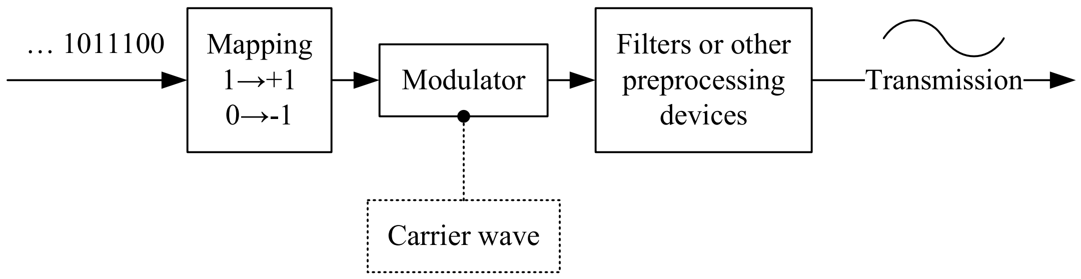

Fig. 13. Modulation block diagram

that varies the carrier wave by $\pi/2$ or 180° when a binary 0 is present in the data stream and held steady otherwise. The bits are first mapped to +1 or -1 and the carrier wave is multiplied by the mapped value to produce the signal that will be converted to the passband frequency before being sent into the wireless channel. Suppose that the bits 1011 are to be sent over the wireless channel with a sine wave as the carrier wave. Fig. 14 shows how the sine wave is modulated. The complex representation of each sample of the modulated signal is:

$$y[0] = 1e^{j0} = cos(0) + jsin(0) = 1 + 0 = 1 \tag{2.11}$$

$$y[1] = -1e^{j\pi} = -1 \tag{2.12}$$

$$y[2] = 1e^{j0} = 1 \tag{2.13}$$

$$y[3] = 1e^{j0} = 1 \tag{2.14}$$

Fig. 15 shows the complex plane constellation of these complex samples.

To illustrate differential encoding, we consider Differential Binary Phase Shift Keying (DBPSK) modulation. The DBPSK method of modulation encodes the bits from the data stream in the phase of the sample. The phase of each sample depends on the phase difference between sets of consecutive complex samples (s_n, s_{n+1}). A general rule of thumb for setting the phase difference of these samples in DBPSK is

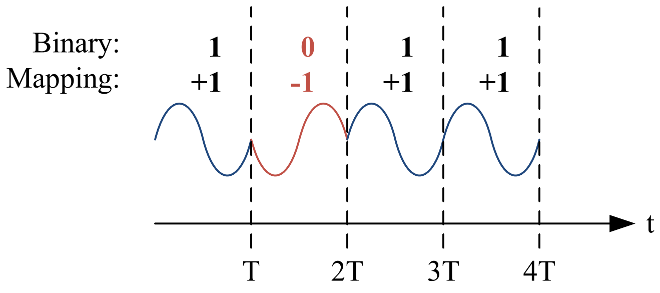

Fig. 14. BPSK encoding for 1011

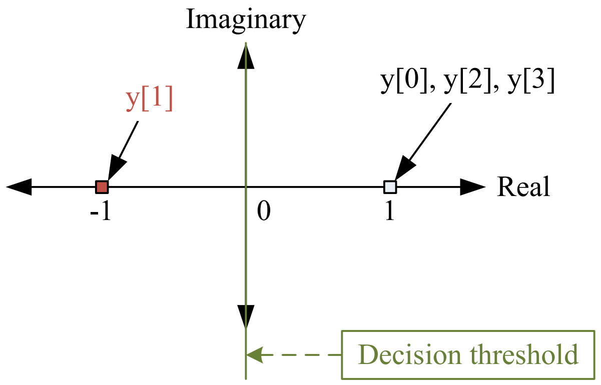

Fig. 15. BPSK constellation for 1011

shown in Table II.

There are also different methods of encoding that encode data in the phase such as Differential Quadrature Phase Shift Keying (DQPSK) and Minimum Shift Keying (MSK). DQPSK encodes 2 bits at a time and based on these pairs, the phase differences of consecutive samples will be based on Table III. MSK modulation adds a phase difference of $\pi/2$ when a binary 1 is encountered and a $-\pi/2$ otherwise. To illustrate this, the encoded phase values of an 18-bit sample such as 100101111010100110 is shown in Fig. 16 with a few points marked.

Table II. DBPSK Encoding Method

Bit n	Phase Difference $(\theta_n - \theta_{n-1})$
0	$0°$
1	$180°$ or π

Table III. DQPSK Encoding Method

Bit Pair	Phase Difference
00	$0°$
01	$90°$ or $\pi/2$
10	$180°$ or π
11	$-90°$ or $-\pi/2$

a. Detailed Modulation Example

Suppose that the data to be transmitted across the wireless channel is 11001011010 using DBPSK modulation. It will be encoded in the following manner. At time $t = 0$, the sample will be set as $s[0] = A_s[0]e^{j\theta_{init}}$ where θ_{init} is the initial phase of the carrier. Next, the first bit (1) will be encoded as $s[1] = A_s[1]e^{j(\theta_{init}+\pi)}$. The change in phase (π) was appended to θ_{init} by using the rule from Table II. The second bit (1) will be encoded in such a way that it will also be $\pi/2$ out of phase but this time in reference to $s[1]$. The result of the encoded second bit will be

$$s[2] = A_s[2]e^{j(\theta_{init}+\pi+\pi)} \tag{2.15}$$

$$= A_s[2]e^{j(\theta_{init}+2\pi)} \tag{2.16}$$

$$= A_s[2]e^{j(\theta_{init}+0)} \tag{2.17}$$

$$= A_s[2]e^{j(\theta_{init})} \tag{2.18}$$

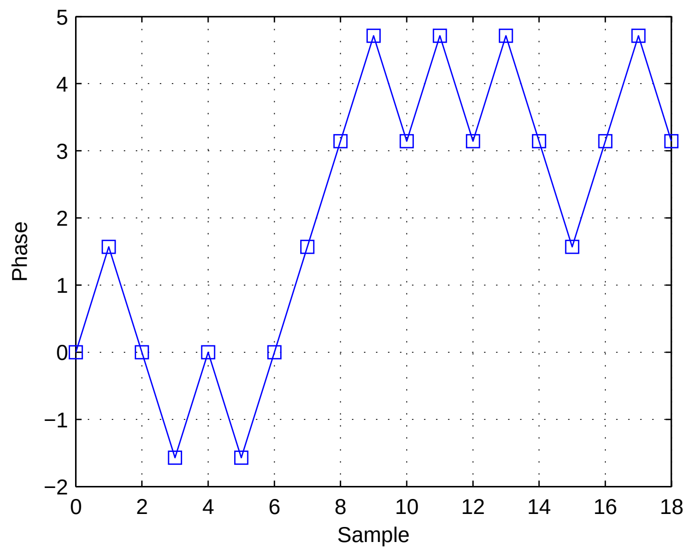

Fig. 16. MSK-encoded phase for 100101111010100110

After all the bits have been encoded, they will be transmitted through the wireless channel as sinusoids. Assuming that θ_{init} is set to be 0 and that the carrier wave is a cosine wave, Fig. 17 shows the first few samples of the transmitted signal through the channel. This shows that for each time a binary 1 is presented, the phase of the carrier wave is shifted by π or 180° in reference to the previous wave.

Decoding for the first bit will be done by sampling the first two sinusoids at the first two symbol periods. The difference in phase will be used to determine the first bit. This will be done for subsequent symbol periods. An important thing to note is that the initial phase θ_{init} does not need to be known in order to properly decode the bits in the data stream. Only the difference between the consecutive samples are important or critical in decoding.

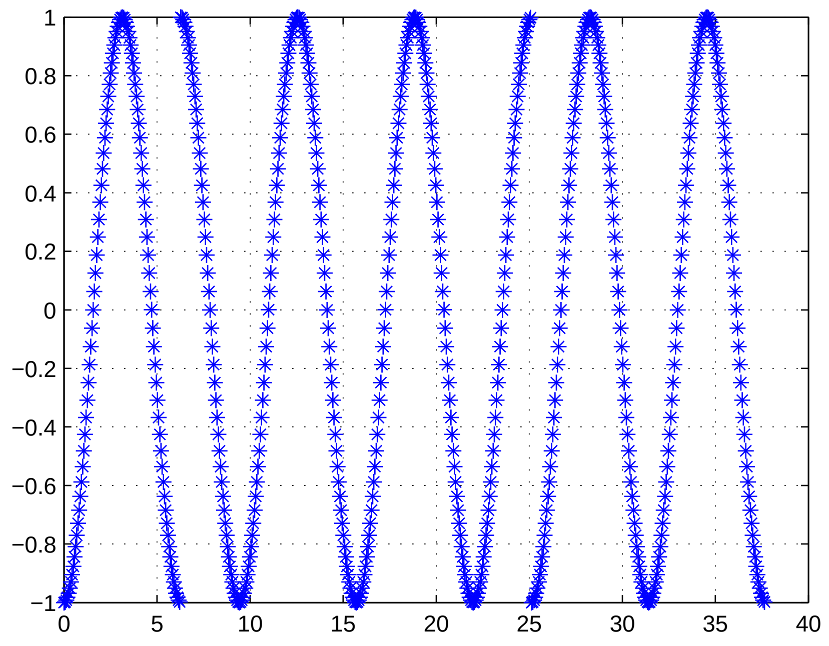

Fig. 17. DBPSK-modulated carrier wave for 11001011010

2. Reception of Two Signals

In the previous section where ANC was discussed, only the task that was performed at the relay N_2 was analyzed. Specifically, N_2 would simply amplify and then forward the mixed signal it received from the senders. It was not discussed what the ideal scenario was as to how P_1 and P_3 should mix in the wireless channel. In order for N_1 or N_3 to decode P_3 or P_1 respectively from N_2's packet P_2, each sample at sample time n of P_1 and P_3 need to be aligned with each other when combining to form P_2. If ANC is to be utilized properly in theory, N_1 and N_3 would need P_2 to be exactly $P_1 + P_3$. Namely, N_1 can decode P_3 by subtracting its known P_1 from P_2 to obtain P_3. Unfortunately, in the wireless channel, there are issues of channel distortions which lead to added phase shifts in each sample as they traverse the channel to N_2. Also, there are timing offsets that are present since the samples of P_1 and P_3 do not align perfectly.

a. Phase Synchronization

Consider the ideal situation where there are no timing offsets, channel phase shifts or noise in the 3-node relay network from Fig. 5. The mixed signal that N_1 receives from N_2 after it broadcasts it is as follows:

$$y[n] = y_1[n] + y_3[n] \tag{2.19}$$

$$= A_1[n]e^{i\theta_1[n]} + A_3[n]e^{i\theta_3[n]} \tag{2.20}$$

where A_i and θ_i are the respective amplitudes and phases at the transmitters. Since N_1 knows its n^{th} sample, it can obtain the same sample of N_3 by subtracting $y_1[n]$ from $y[n]$. This is a useful property of ANC since the signals were mixed at the physical layer. What will happen when this ideal situation is eliminated? The phase differences introduced by the channel from N_1 to N_2, N_3 to N_2 and from N_2 while broadcasting back to the senders will need to be considered.

The actual signal that is received with the channel phase distortion appended is

$$y[n] = h_1[n]y_1[n] + h_3[n]y_3[n] \tag{2.21}$$

$$= h_1[n]A_1[n]e^{i\theta_1[n]} + h_3[n]A_3[n]e^{i\theta_3[n]} \tag{2.22}$$

where $h_1[n]$ and $h_3[n]$ are complex numbers representing the phase shift and attenuation introduced by the channel on the n^{th} sample of P_1 and P_3 respectively. If we define $h_1[n] = h'[n]e^{i\gamma'[n]}$ and $h_3[n] = h''[n]e^{i\gamma''[n]}$, Eq. 2.21 can be expanded to

$$y[n] = h'[n]e^{i\gamma'[n]}A_1[n]e^{i\theta_1[n]} + h''[n]e^{i\gamma''[n]}A_3[n]e^{i\theta_3[n]} \tag{2.23}$$

$$= h'[n]A_1[n]e^{i(\theta_1[n]+\gamma'[n])} + h''[n]A_3[n]e^{i(\theta_3[n]+\gamma''[n])} \tag{2.24}$$

If we further assume that both the attenuation and channel phase is constant for all time, we can rewrite the received signal as

$$y[n] = h'A_1[n]e^{i(\theta_1[n]+\gamma')} + h''A_3[n]e^{i(\theta_3[n]+\gamma'')} \tag{2.25}$$

From (2.21), you cannot simply subtract $y_1[n]$ from $y[n]$ to obtain $y_3[n]$ as in the ideal case. N_1 will need to estimate the values of the corresponding phase shifts in order to properly decode the right sample from N_3 hence leading to consequent decoding of the rest of N_3's bits.

b. Symbol Synchronization

In the previous section, only the channel phase shift and attenuation effect on the samples of N_1 and N_3 were discussed. What happens when the symbol boundaries of P_1 and P_3 do not align when forming P_2. This is considered a lack of symbol synchronization and also a timing offset. Consider N_1 sending P_1 and N_3 sending P_3 simultaneously in the 3-node relay network. Ideally, N_2 wants to receive the following (without noise at the receiver):

$$p_1(t) + p_3(t) \tag{2.26}$$

However, due to P_1's sample at time t not aligning with P_3's sample at time t when received at N_2, the resulting signal will be as follows:

$$p_1(t+\epsilon_1) + p_2(t+\epsilon_2) \tag{2.27}$$

where the ϵ_i are the timing or sampling offsets between the senders' signals. In the ideal situation depicted in Fig. 18 where BPSK is used, the signals at the transmitter are 180° or $\pi/2$ out of phase with each other. The mixed signal at the relay will produce a desired constant value of 0. In the non-ideal situation depicted in Fig. 19,

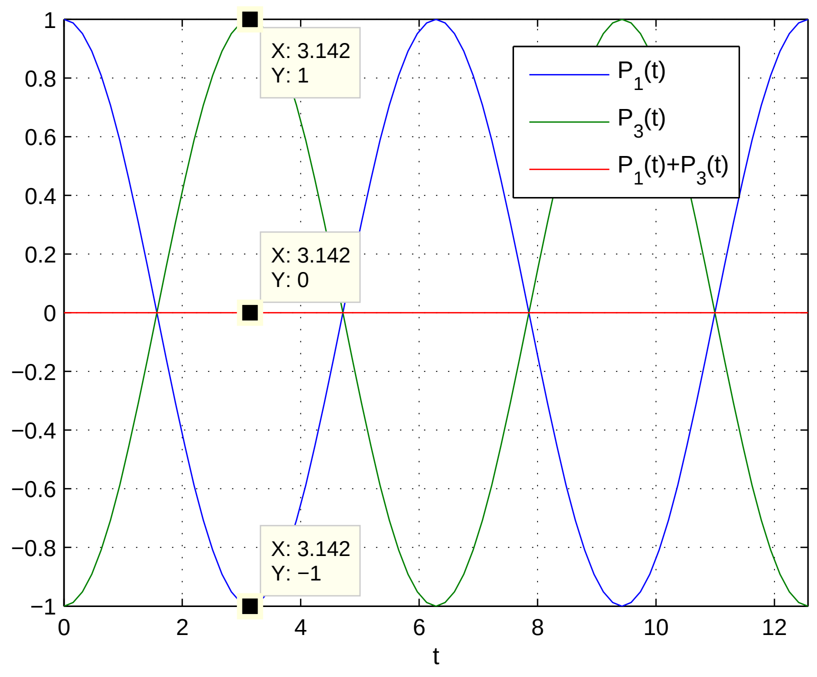

Fig. 18. Reception at the relay with timing synchronization

which accurately models the wireless channel, the effect of a high timing offset (.7π in this example) can lead to misleading information at the relay especially if it needs to utilize a scheme such as Decode-and-Forward. The relay needs a signal close to 0 in order to decode that the two signals were indeed 180° out of phase with each other. Instead, the relay assumes that the two sent signals were identical thus producing peak amplitudes at +2. Since Fig. 18 can be used to describe a situation where $P_1 = 1$ and $P_3 = 0$, a {1,0} row from a BPSK table similar to Table I can be used in

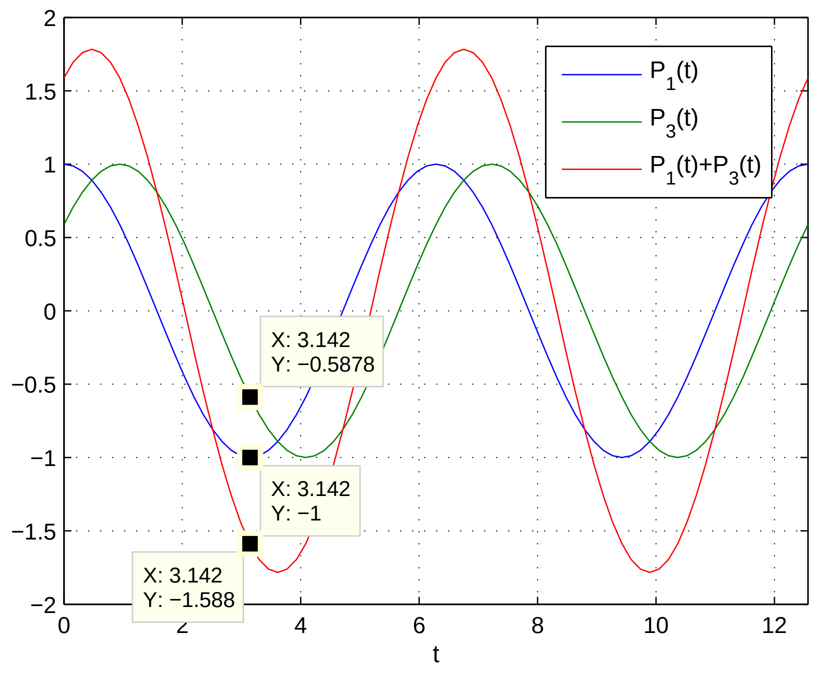

Fig. 19. Reception at the relay without timing synchronization

determining what the relay is to do next. Fig. 19 describes a situation in which the relay assumes that $P_1 = 1$ and $P_3 = 1$ with P_3 being predicted in error.

C. Related Work

There has been previous work done in the field of telecommunications with an emphasis specifically on wireless relay networks. The research often present methods to improve network capacity and throughput by utilizing DNC or ANC [10, 11] as mentioned in an earlier section of this thesis. In the 3-node relay network where there is a reception of two wireless signals simultaneously (ANC), the issue of synchronization of phase as well as symbol synchronization presents itself as a frequently discussed topic. Some papers on these networks already assume that there is synchronization at the symbol level meaning that the samples of P_1 and P_3 at sample time n are perfectly aligned.

The first paper discussed in this section is on ANC and its applications [11]. It utilizes the Amplify-and-Forward scheme of transmission from the relay. The second paper [10] works on Physical Network Coding (PNC) which is similar to ANC except that in this paper, the Decode-and-Forward scheme is used. Both papers work under the same 3-node relay network topology of Fig. 5.

1. Analog Network Coding with Amplify-and-Forward

Consider the reception of two Minimum Shift Key (MSK) modulated signals simultaneously at a relay (N_2) as depicted in Fig. 10[4]. Since this is done on the analog level, the EM waves simply sum with each other. Suppose that Alice (N_1) and Bob (N_3) transmit at the same time and are received at N_2 as $y = y_A + y_B$. If N_2 samples the received signal at time n, the resulting sample will be defined (stated earlier) as [11]

$$y[n] = y_A[n] + y_B[n] \tag{2.28}$$

$$= h' A_s e^{i(\theta_s[n]+\gamma')} + h'' B_s e^{i(\phi_s[n]+\gamma'')} \tag{2.29}$$

where, on Alice's side, A_s is the amplitude at her transmitter before traversing the channel, h' is the channel's attenuation, $\theta_s[n]$ is the phase of the n^{th} sample and γ' is the channel phase. N_2 takes this signal and simply forwards it to Alice and Bob. N_2 does no form of decoding or signal processing on $y[n]$. When one transmission is taking place with MSK modulation being utilized, the amplitude, A_s, of every sample at time n will stay constant and will not be relevant in decoding the data

[4]In this section N_1 will be referred to as Alice and N_3 will be referred to as Bob. This type of relay network is also referred to as the Alice-Bob topology

bits from consecutive samples. To prove this, consider the received complex signal characterized by

$$x[n] = hA_s e^{i(\theta_s[n]+\gamma)} \tag{2.30}$$

To calculate the phase difference between consecutive samples of x and hence the encoded bit, the parameter r, for any n, will be calculated as

$$\begin{aligned}
r &= \frac{x[n+1]}{x[n]} && (2.31)\\
&= \frac{hA_s e^{i(\theta_s[n+1]+\gamma)}}{hA_s e^{i(\theta_s[n]+\gamma)}} && (2.32)\\
&= \frac{e^{i(\theta_s[n+1]+\gamma)}}{e^{i(\theta_s[n]+\gamma)}} && (2.33)\\
&= e^{i(\theta_s[n+1]+\gamma-\theta_s[n]-\gamma)} && (2.34)\\
&= e^{i(\theta_s[n+1]-\theta_s[n])} && (2.35)
\end{aligned}$$

Since r is a complex number, obtaining its phase $\theta_s[n+1] - \theta_s[n]$, which is the phase difference, can be done by calculating its angle. A general rule of thumb which factors in the presence of channel noise and estimation errors, is that a positive phase difference gets mapped to a Binary 1 while a negative phase difference gets mapped to a Binary 0. From the derivation of r, it shows that the channel's phase shift and attenuation does not matter since MSK encodes the bits in consecutive phase differences (this assumes that the channel phase shift and attenuation is constant across all samples).

In Katti's paper [11], the main focus is on what Alice does when she receives y. If Alice can estimate the channel parameters h' and γ', she can recreate her signal $y_1[n]$ and simply subtract it from $y[n]$ and recreate Bob's $y_3[n]$. Calculating γ' requires the utilization of coherent phase tracking algorithms. This paper, however, presents a method that will take advantage of r and the non-coherent decoding property of

MSK. The presented algorithm starts by restating (2.29) as

$$y[n] = Ae^{i\theta[n]} + Be^{i\phi[n]} \tag{2.36}$$

where $A = h'A_s$, $B = h''B_s$, $\theta[n] = \theta_s[n] + \gamma'$ and $\phi[n] = \phi_s[n] + \gamma''$. Alice then needs to calculate the phase difference pairs ($\Delta\theta$,$\Delta\phi$) that could have produced $y[n]$. Alice obtains the best pair by matching her known phase difference $\Delta\theta_s$ with each $\Delta\theta$. Based on which one produces the smallest error, the corresponding $\Delta\phi$ will be Bob's phase difference. With this information, Alice can apply Eq. 2.31 and decode Bob's bits. This method of ANC yielded a BER of around 4%.

2. Physical Network Coding with Decode-and-Forward

Zhang's paper [10] performs ANC slightly differently than Katti's paper [11]. Instead of utilizing Amplify-and-Forward, it utilizes Decode-and-Forward at the relay. QPSK modulation is used at all nodes of the relay network. The signal received at N_2 is defined in this paper as

$$\begin{aligned} r_2(t) &= s_1(t) + s_3(t) \qquad (2.37)\\ &= [a_1\cos(\omega t) + b_1\sin(\omega t)] + [a_3\cos(\omega t) + b_3\sin(\omega t)] \qquad (2.38)\\ &= (a_1 + a_3)\cos(\omega t) + (b_1 + b_3)\sin(\omega t) \qquad (2.39)\end{aligned}$$

where $s_i(t)$, for $i \in \{1, 3\}$, is the signal transmitted by N_1 and N_3 respectively. The signal $r_2(t)$ is separated to its in-phase (I) and quadrature (Q) components which differ by 90° or $\pi/2$ as

$$I = a_1 + a_3 \tag{2.40}$$

$$Q = b_1 + b_3 \tag{2.41}$$

Based on what N_1 and N_3 send simultaneously to N_2, it will decode the combined I value and Q value, create and modulate its own bit s_2 so that N_1 and N_3 can receive and decode this bit. Tables IV, V and VI show, respectively, the encoding of the I components at the senders, N_1 and N_3, at the relay, N_2, and the decoding process done at the senders after receiving data from the relay. The parameters s_i and a_i,

Table IV. PNC In-Phase Mapping at N_1 and N_3

s_1	s_3	N_1 sends a_1	N_3 sends a_3
1	1	1	1
0	1	-1	1
1	0	1	-1
0	0	-1	-1

Table V. PNC In-Phase Mapping at N_2

N_2 receives $(a_1 + a_3)$	s_2	N_2 sends a_2
2	0	-1
0	1	1
0	1	1
-2	0	-1

for $i \in 1, 2, 3$, represent the bits to be encoded and the actual values that are used to modulate the carrier cosine and sine waves respectively. In modulating these carrier waves, a_i and b_i are defined as $2s_i - 1$. The Q components which are represented by b_i, for $i \in \{1, 2, 3\}$, can utilize these same tables as well with an $a_i \to b_i$ mapping.

The bit error rate (BER) performance of this method was calculated as well. In [10], the BER of the broadcast frames in Fig. 10 is that of standard BPSK modulation and defined as $Q(\sqrt{2/N_0})$ [18] where N_0 is the noise power spectral density. The

Table VI. PNC In-Phase Decoding at N_1 and N_3

N_1 and N_3 receive $(a_2 \rightarrow s_2')$	N_1: $s_3' = s_2' \oplus s_1$	N_3: $s_1' = s_2' \oplus s_3$
$-1 \rightarrow 0$	1	1
$1 \rightarrow 1$	1	0
$1 \rightarrow 1$	0	1
$-1 \rightarrow 0$	0	0

function $Q(.)$, also known as the *Q-function*, is the complementary cumulative distribution function for the zero-mean ($\mu = 0$), unit-variance ($\sigma = 1$) Gaussian random variable. It is defined for a normal[5] Gaussian random variable, X, as

$$\begin{aligned} Q(x) &= 1 - \Phi(x) && (2.42)\\ &= 1 - \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(u-\mu)^2}{2\sigma^2}} du && (2.43)\\ &= 1 - \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{u^2}{2}} du && (2.44)\\ &= \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-\frac{u^2}{2}} du && (2.45) \end{aligned}$$

where $\Phi(x)$ is the cumulative distribution function (CDF) of X. The paper concludes that the BER of PNC is slightly less than traditional QPSK transmission but does have better performance than DNC. This method of ANC does decoding at the bit level at the relay and the senders. However, in contrast to DNC, it requires 2 transmission time slots as opposed to 3. Due to this, there is an apparent trade-off between BER performance and network throughput.

[5]A normal Gaussian random variable is one with zero-mean and unit-variance.

3. Phase and Symbol-level Synchronization Assumptions

In [11], symbol-level synchronization is assumed in the main results of the paper. It does, however, assume a lack of phase synchronization which is accurate in modeling real-time communication systems. Equations 2.28 or 2.36 assume that samples align and there is no sampling offset such as ϵ_A or ϵ_B. There is also a constant channel phase, γ' and γ'', that is applied to every sample of y_A and y_B, respectively, as they go through the channel. This allows for the algorithm to work quite robustly. In the wireless channel, the applied phase distortion is not necessarily constant and is independent from sample to sample. Despite this notion, it is safe to assume that it is fairly constant especially for non-large sequences. In [10], the algorithm presented already assumes symbol-level and carrier-phase synchronization. This thesis tries to tackle both symbol-level asynchrony as well as carrier-phase asynchrony in order to decode transmitted bits properly.

D. GNU Radio and the USRP

This thesis and its ongoing research aims to design and implement network coding on real-life communication systems. Using the 3-node relay network as a system model, the relay and transmitters are to be implemented and tested in hardware. Software-defined radios (SDRs) are chosen as a testbed in running these tests. SDR is a system in which components that are generally implemented in hardware (modulators/demodulators, filters, converters, amplifiers, probes, etc.) are all done in software. The RF front-ends used in our research are the Universal Software Radio Peripheral (USRP) boards with one depicted in Fig. 20. The USRPs have 4 analog-to-digital converters (ADCs) and 4 DACs. They connect to PCs through USB 2.0 cables which have a maximum data rate of 480 Mbits/s. The ADC operates at

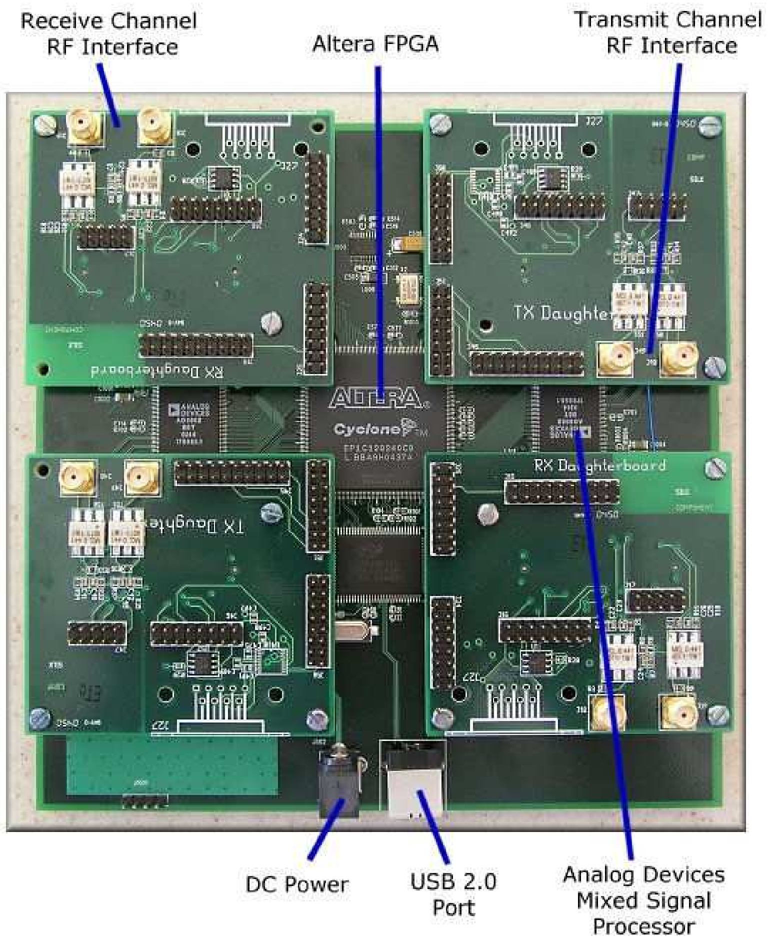
Receive Channel RF Interface
Altera FPGA
Transmit Channel RF Interface
DC Power
USB 2.0 Port
Analog Devices Mixed Signal Processor

Fig. 20. Universal Software Radio Peripheral board (image from *Cornell University - Research*)

64MSamples/s while the DAC operates at 128MSamples/s. To illustrate the flow of data and signal to and from the radio boards, consider the case where data is to be sent from the USRP to another communication device. When transmitting signals through the USRP, complex samples that have been preprocessed in software are sent from the computer to the USRPs Digital-to-Analog (DAC) converter which upconverts the data sent from the computer to the carrier frequency range[6]. What about the converse of this? The ADC downconverts the passband signal, which is at a high data rate, to a rate that USB 2.0 can process. The digital samples are passed to the computer as complex samples. Once this is done, the user can manipulate the samples and if needed, transmit them back through the USRP.

The next question becomes how the signal processing blocks are implemented in software and how does the DAC receive the complex samples. The answer involves GNU Radio [13] which is an open-source project developed by Eric Blossom. GNU Radio is a software tool under the GNU project used in building SDR systems. GNU Radio applications are programmed primarily in python with most of the critical signal processing blocks implemented in C++. Most blocks in GNU Radio have input and output ports with the input ports denoted as sinks and the output ports denoted as sources. Python *glues* the C++ elements together using flow graphs. GNU Radio uses flow graph classes denoted as *gr_top_block* which require that sources be connected to sinks. Fig. 21 depicts a high-level flow diagram of the software and its interaction with the USRP for signal transmission and reception. The python script takes in the C++ blocks and connects them together along with either the *usrp_sink_c* block or the *usrp_source_c* block. The block *usrp_sink_c* takes in complex samples that are to be sent out by the USRP through the DAC while *usrp_source_c* are the complex samples that the ADC sent to the PC for processing. To illustrate how this works in python with a high-level example, assume that a set of complex samples are to be transmitted by the USRP. First, the complex samples will be converted to a suitable GNU Radio complex source block that will be connected to the URSP sink. A snippet of this code is shown in Table VII with lines beginning with # representing

[6]We use 2.4GHz as the center frequency for all transmissions.

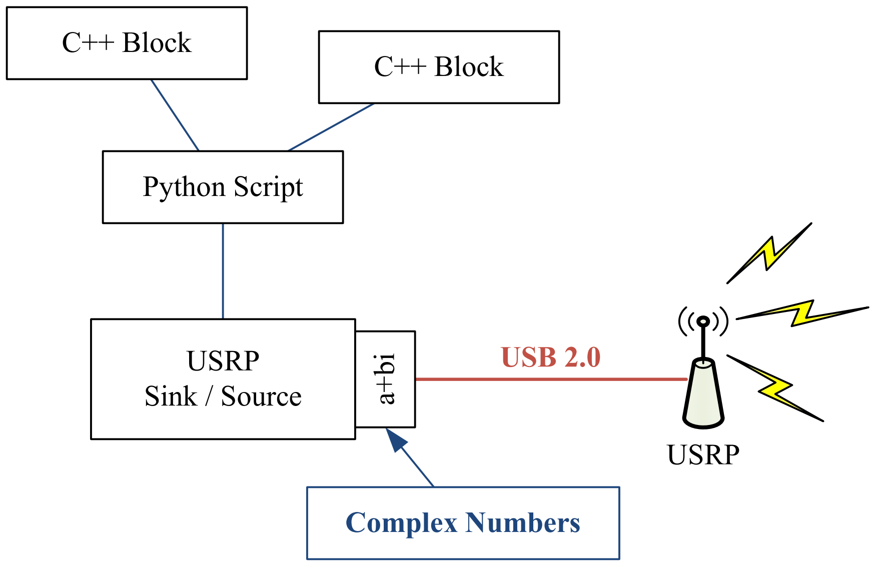

Fig. 21. GNU Radio flow diagram

comments.

E. Goals and Motivation

It has been shown that GNU Radio is a convenient tool in modeling real communication systems. It would be beneficial to be able to model real systems practically instead of modeling them in theory. For example, noise in a channel can be simulated by generating random real and complex data through scripts. However, using actual hardware such as the USRP would not only model the noise realistically but will also give insight into the *flavor* of noise present in different networks. The issue of phase and symbol-level synchronization can also be tackled with the radio boards. The 3-node relay network can be modeled with 3 USRPs and based on the channel in the simulation area, realistic phase distortions and timing offsets can be realized.

The aim of this research is to propose and present an algorithm or method that can be used in decoding mixed signals by detecting phase and timing offsets. After this is accomplished, it can be utilized in a testbed of software radios thus highlighting the practicality of the results. After presenting a system model on paper, it is quite interesting to see how it fares in a real-life system. GNU Radio presents many open-source tools that will aid in signal manipulation and processing which will be used in testing these algorithms.

Table VII. GNU Radio python script snippet

Line	**Python Code**
1	# Sets the transmit amplitude to 12000
2	tx_amplitude = 12000
3	# Creates the flow graph which connects all the blocks together
4	flow_graph = gr.top_block()
5	# Creates the USRP sink that sends complex samples to the DAC
6	sink = usrp.sink_c()
7	# Complex data to be transmitted
8	complex_data = (-1+.3j, 1+.5j, -.7+1.2j, 1+0j)
9	# Formats the complex data in a vector block format
10	complex_data_source = gr.vector_source_c(complex_data)
11	# Creates the amplifier that the complex data passes through
12	amplifier = gr.multiply_const_cc(1)
13	# Sets the amplify factor to the transmit amplitude (12000)
14	amplifier.set_k(tx_amplitude)
15	# Connects the blocks together in the flow graph in preparation for it to be started (transmission)
16	flow_graph.connect(complex_data_source, amplifier, sink)
17	# Starts the flow graph. Ends when all the samples have reached the USRP sink.
18	flow_graph.run()

CHAPTER III

IMPLEMENTATION AND RESULTS

This chapter of the thesis will focus on simulating the algorithm presented in the previous chapter on the 3-node relay network [11]. The chapter will also provide a unique approach to decoding interfered signals that utilizes Decode-and-Forward and more importantly, takes into account the timing offsets between the interfering signals. One of the goals of this thesis is to implement and test this new approach using the USRP in conjunction with GNU Radio in order to better understand its decoding capabilities and practicality. This will be done by collecting and sending complex signal data from the USRPs to data files (.dat files for example) which MATLAB will use in applying the methods developed in this thesis. This process hopefully will lead to low and stable BERs. Katti's main algorithm from [11] was simulated in MATLAB by following the channel and signal models that they provided. The purpose of simulating it was to provide a reference point in making comparisons with the new decoding approach and for analyzing both performances. Their method was relatively robust but somewhat limited by channel parameter estimation errors (magnitude and attenuation) being the limiting factors. From Eq. 2.36, these parameters are A and B.

Our method for decoding interfered signals utilizes a Viterbi decoder [18, 19] which performs decoding based on the trellis representation of the transmitted signal. In the trellis, the states are based on the possible values of the sent bits (a_k and b_k) from both senders (N_1 and N_3 from Fig. 5) at any symbol period k. Based on the values of the bits from either sender, there is a branch metric $B_{i,j}^{(k)}$ for each transition path (previous state to next state) that is calculated after comparing the received signal (for the same symbol period) with the expected value given the transition path. This approach can give a maximum likelihood (ML) decoder by finding the predefined signal that matches most closely to the actual received signal.

A. Analog Network Coding Simulation

In Katti's paper [11] and Ph.D thesis [20], the main approach to decoding transmitted and interfered signals is presented for the network depicted in Fig. 5. It assumes that the channel phase distortion is roughly constant on every sample of the signals transmitted. Recall that N_1 (Alice) and N_3 (Bob) transmit simultaneously to N_2 which follows by broadcasting the mixed signal back to the senders (Amplify-and-Forward). Analysis at Alice's end is done but similar techniques can be performed for Bob.

Using Eqs. 2.28 and 2.29 as Alice's received signal, with $y[n] = Ae^{i\theta[n]} + Be^{i\phi[n]}$ (2.36) being a simplified version, the decoding will be performed at the senders by utilizing the following two equations:

$$E[|y[n]|^2] = \mu = A^2 + B^2 \tag{3.1}$$

$$\sigma = A^2 + B^2 + 4AB/\pi \tag{3.2}$$

where the parameter σ is defined as

$$\sigma = \frac{2}{N} \sum_{|y[n]|^2 > \mu} |y[n]|^2 \tag{3.3}$$

We are left with two equations with two unknowns (A and B). Obtaining an expression for A is done as follows:

$$\sigma = \mu + 4AB/\pi \tag{3.4}$$

$$\sigma - \mu = 4AB/\pi \tag{3.5}$$

$$\pi(\sigma - \mu) = 4AB \tag{3.6}$$

$$\frac{\pi(\sigma - \mu)}{4} = AB \tag{3.7}$$

$$\frac{\pi(\sigma - \mu)}{4B} = A \tag{3.8}$$

Substituting (3.8) into (3.1) with $\pi(\sigma - \mu)/4 \rightarrow K$, B can be solved in the following manner:

$$\mu = \frac{K^2}{B^2} + B^2 \tag{3.9}$$

$$\mu B^2 = K^2 + B^4 \tag{3.10}$$

$$0 = B^4 - \mu B^2 + K^2 \tag{3.11}$$

We are left with a 4^{th} degree polynomial equation of the form:

$$z(x) = ax^4 + bx^3 + cx^2 + dx + e = 0 \tag{3.12}$$

which yields 4 possible solutions for B. Using parameters from (3.11), (3.12) can be rewritten by setting $a = 1$, $b = 0$, $c = -\mu$, $d = 0$ and $e = K^2$ with B as the independent variable. The four solutions for B are:

$$B_1 = \sqrt{\frac{\mu + \sqrt{\mu^2 - 4K^2}}{2}} \tag{3.13}$$

$$B_2 = -B_1 \tag{3.14}$$

$$B_3 = \sqrt{\frac{\mu - \sqrt{\mu^2 - 4K^2}}{2}} \tag{3.15}$$

$$B_4 = -B_3 \tag{3.16}$$

The choice for B from all B_i is up to the designer. One might want to restrict B to real numbers or limit it to some threshold. It should be noted that the unknown parameters in the samples $y[n]$ are A, B, $\theta[n]$ and $\phi[n]$. After solving for the amplitudes A and B, the possible phase difference pairs $(\theta[n], \phi[n])$ are calculated with equations defined in [11, 20].

This algorithm was simulated in MATLAB. At first, the values of A and B were hard-coded to their real values and not to the estimated values. As a result, the BER was essentially 0 as expected because the system has no noise besides parameter estimation noise. This demonstrates the robustness of Katti's decoder when these parameters are estimated correctly. Next, the values for A and B were estimated using Eqs. 3.1 and 3.2 with the choice for B being

$$B = \frac{1}{4}\sum_{i=1}^{4} |B_i| \tag{3.17}$$

The simulations were run 1000 times with 300 randomly generated bits in Alice and Bob's packets and the channel phases being uniformly selected from 0 to 2π. The mean BER for the simulation was around .0579 or 5.79% which was somewhat close to their reported BER of around 4%. In Fig. 22, the BER performance of the decoder for the first 30 trials is shown along with the estimation errors for A and B. There exist sharp spikes in the BERs when the estimation errors were quite high meaning a large deviation from the actual value of A and B. Apart from these situations, the BER was roughly around 0. Fig. 23 shows the BERs for the first 100 trials and Fig. 24 shows the CDF of the BERs for all trials.

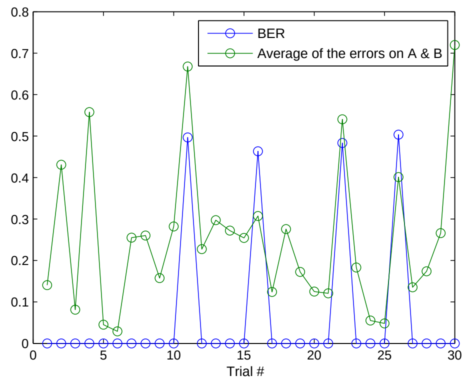

Fig. 22. BER and estimation errors for ANC simulations (1st 30 trials)

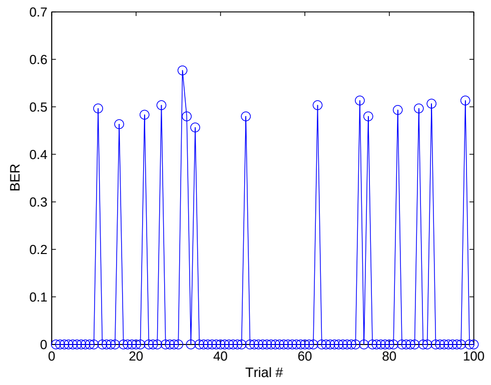

Fig. 23. BER of ANC simulations (1st 100 trials)

B. Proposed Decoding Algorithm

Katti's design from [11] does not include a timing offset between the senders' signals nor does it factor in noise from the receiver or from the wireless channel. We now

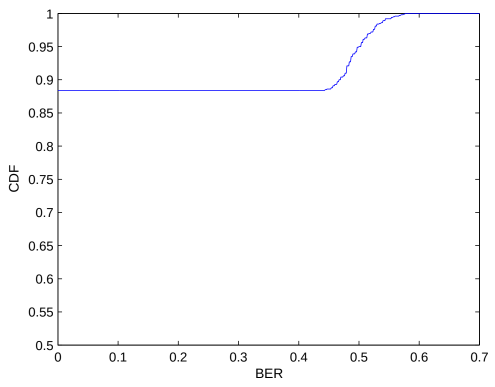

Fig. 24. BER CDF of ANC simulations

present an approach that does both and includes the channel phase distortions as well. This method for decoding mixed signals utilizes Viterbi decoding which, as mentioned earlier, performs decoding based on possible state transitions. Since the mixed signal provides no information as to which bit was transmitted by both senders (mainly due to the timing offsets), a state-based decoder that provides transition weights based on a reliable metric on what could have been transmitted (combination of bits) proves useful. The main goal is for the relay to decode the sum of the bits sent and relay this back to the senders. The Viterbi decoder assigns a branch metric to paths that connect different states in a state transition diagram together. Based on these metrics, the decoder will trace through all paths in an efficient manner using comparisons between the received mixed signal and an expected mixed signal with that path's parameters. Given the received signal, the branch metric from the current state i to the next state j for a given symbol period k is the probability that the received signal stemmed from the edge $E_{i,j}$ or state transition leading to that state.

1. Background

Before delving into the decoder, a description of how we model the transmitted signal will be explained. Suppose that two users A and B (N_1 and N_3 respectively from Fig. 5) wish to transmit bits using BPSK modulation to each other through N_2. Also, suppose that the BPSK signals have signal levels a_k (± 1) and b_k (± 1) which are derived from the following mapping:

$$\text{Binary } 0 \rightarrow \text{-1}$$

$$\text{Binary } 1 \rightarrow +1$$

The symbol period is set to T seconds and $p(t)$ from Fig. 25 is used as the baseband modulating pulse for transmitting the data. The pulse $p(t)$ is described analytically

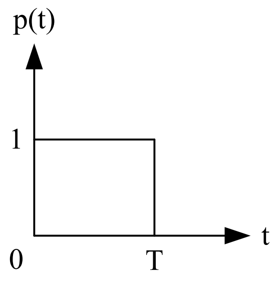

Fig. 25. Square pulse with signal duration T seconds

as

$$p(t) = \begin{cases} 1 & \text{if } 0 \leq t < T \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

The choice for the modulating pulse is based on its simplicity. In practice, a raised cosine filter might be utilized due to its bandlimited nature in the frequency spectrum. Typically, a square pulse will waste bandwidth when transmitted. The bits are transmitted in the channel as a series of pulses each with duration T seconds[1]. The transmitted signal by A is then defined as

$$s_a(t) = \sum_{k=-\infty}^{\infty} a_k p(t - kT) \tag{3.19}$$

$$= \sum_{k=0}^{N-1} a_k p(t - kT) \tag{3.20}$$

where N is the number of bits being transmitted. We assume that $a_k = 0$ for values of $k < 0$ and $k > N - 1$. Most assumptions made on A's signal can be applied to B's signal as well. The wireless channel is modeled as an Additive White Gaussian Noise (AWGN) channel.

After A and B's signals are generated, they are transmitted simultaneously to the relay. This mixed signal passes through a matched filter, which is an optimal linear filter for maximizing SNR. Matched filters correlate a known signal $p(t)$ with the received signal in order to detect the presence of the known signal in it. The output of the matched filter is sampled and a decision on the sum of the two signals will be made based on these samples. Fig. 26 shows the typical flow of the signal from the sender all the way through to the decoder. From the diagram, the received signal at the relay before filtering can be defined as

$$r(t) = y_a(t) + y_b(t) + n(t) \tag{3.21}$$

$$= \alpha s_a(t) + \beta s_b(t) + n(t) \tag{3.22}$$

$$= \alpha \sum_{k=0}^{N-1} a_k p(t - kT) + \beta \sum_{k=0}^{N-1} b_k p(t - kT) + n(t) \tag{3.23}$$

[1]In the model used in this research, T is chosen to be 1s in order to simplify computations.

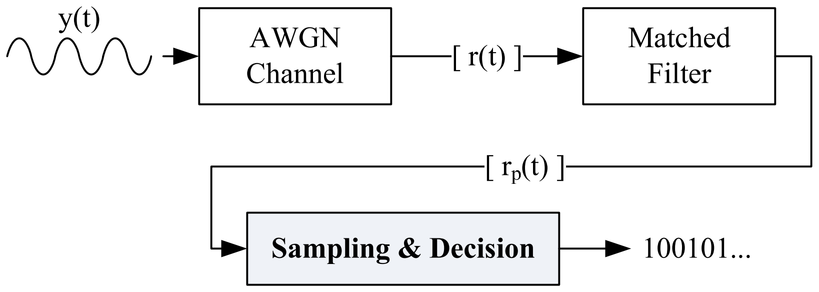

Fig. 26. Signal flow from source to decoder

where $n(t)$ is the noise introduced from the AWGN channel or the receiver and α and β are complex numbers that represent the channel phase and attenuation effect on A and B respectively. In our present model, we assume that the channel attenuation is normalized to unity. The output of the matched filter is then defined as

$$\begin{aligned}
r_p(t) &= p(t) * r(t) && (3.24)\\
&= \int_{-\infty}^{\infty} p(\sigma) r(t-\sigma) d\sigma && (3.25)\\
&= \int_{0}^{T} \underbrace{p(\sigma)}_{=1} r(t-\sigma) d\sigma && (3.26)\\
&= \int_{0}^{T} r(t-\sigma) d\sigma && (3.27)\\
&= \alpha \int_{0}^{T} s_a(t-\sigma) d\sigma + \beta \int_{0}^{T} s_b(t-\sigma) d\sigma + \int_{0}^{T} n(t-\sigma) d\sigma && (3.28)
\end{aligned}$$

Eq. 3.28 demonstrates the linearity of the matched filter in that its input can be broken down into its components (summands), filtered individually and then added to produce the same effect. The effect of the filter is to average the values of $r(t)$ and reduce the noise contributed by $n(t)$.

Suppose that A passes the following two symbols $a_0 = +1$ and $a_1 = +1$ through the matched filter with $p(t)$ as the correlating signal. This transmitted signal is

$$s_a(t) = \sum_{k=0}^{1} a_k p(t - kT) \tag{3.29}$$

$$= a_0 p(t) + a_1 p(t - T) \tag{3.30}$$

$$= p(t) + p(t - T) \tag{3.31}$$

In the absence of noise ($n(t) = 0$), the output of the matched filter will be as follows:

$$r_p(t) = p(t) * s_a(t) \tag{3.32}$$

$$= \int_0^T s_a(t - \sigma) d\sigma \tag{3.33}$$

$$= \int_0^T p(t - \sigma) + p(t - \sigma - T) d\sigma \tag{3.34}$$

The plot of $r_p(t)$ for this example is shown in Fig. 27. The output of the filter will

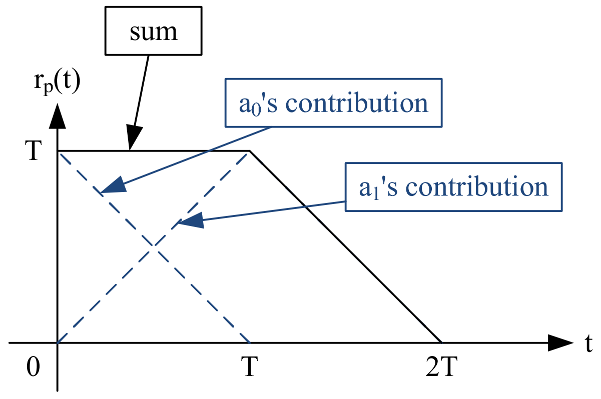

Fig. 27. Matched filter output of interference-free $s_a(t)$ for $a_0 = +1$ and $a_1 = +1$

be sampled at intervals kT for $k = 0, 1, ..., N - 1$ where N is the number of bits or a limit that can be set by the receiver.

To better understand the sampling of the filter output with the mixed signal being the input, we assume that a_k is a stream of +1's (Binary 1) and b_k is a stream of -1's (Binary 0) sent to the relay N_2. Assume that A's signal reaches the relay first with B's signal lagging by τ seconds and $0 \leq \tau < T$. This is the timing offset that was discussed in the previous chapter. The resulting output of the matched filter for this example is depicted in Fig. 28 with the sampling offset τ shown. For explanation purposes and clarity, Fig. 28 is the output with the absence of noise and with the channel phase distortion effect negligible (α=1 and β=1). We also assume that the

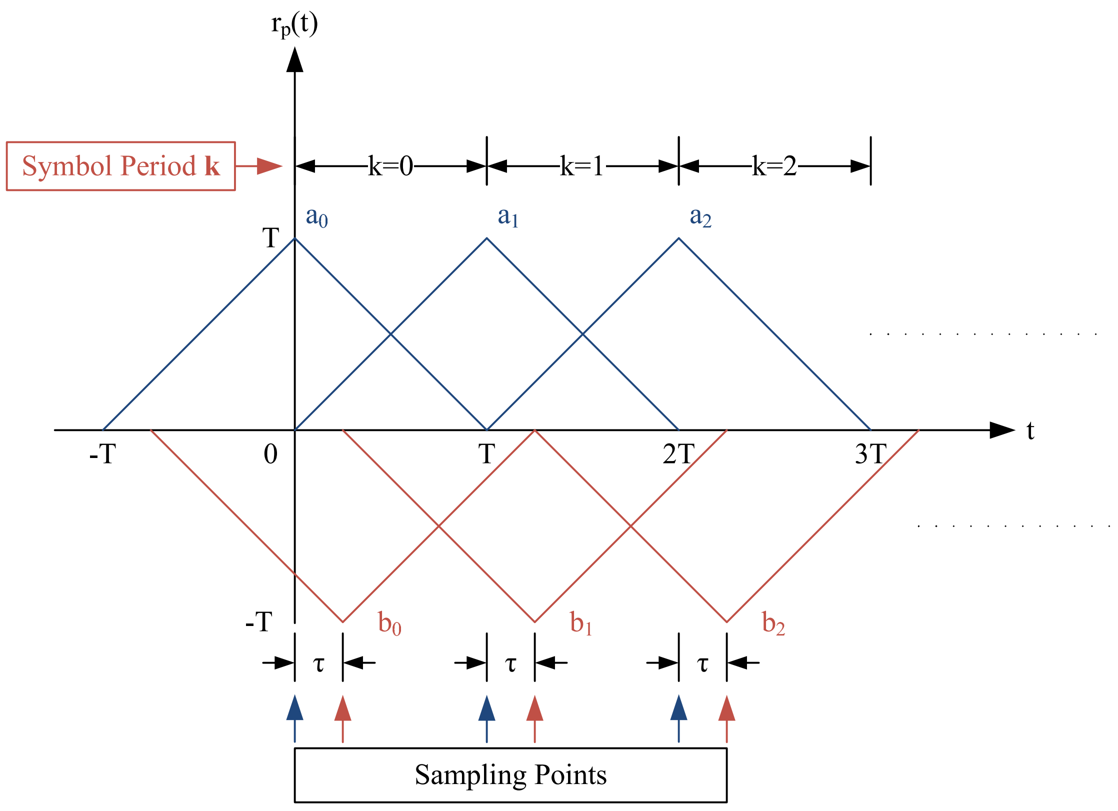

Fig. 28. Matched filter output for $a_k = +1$ and $b_k = -1$

receiver is able to lock onto A's signal first and can estimate accurately its channel phase distortion α using test transmissions coupled with phase locking loops. This is done before the bits a_k reach the receiver and will therefore allow the receiver to know the peak of A's first bit a_0. We would like to be able to form an expression for the matched filter output at both A and B's peaks which includes the timing offset τ in it. Using the symmetry and the triangular shape of the output to our advantage, the samples of A's peak which are taken at time $t = kT$ for symbol period k is defined as:

$$\begin{aligned} y_k^{(a)} &= \alpha T a_k + \beta T b_{k-1}\left(1 - \frac{T-\tau}{T}\right) + \beta T b_k \left(1 - \frac{\tau}{T}\right) + n_k && (3.35)\\ &= \alpha T a_k + \beta T b_{k-1}\left(\frac{\tau}{T}\right) + \beta T b_k \left(1 - \frac{\tau}{T}\right) + n_k && (3.36)\\ &= \alpha T a_k + \tau\beta b_{k-1} + \beta T b_k - \tau\beta b_k + n_k && (3.37)\\ &= \alpha T a_k + \tau\beta(b_{k-1} - b_k) + \beta T b_k + n_k && (3.38) \end{aligned}$$

The same thing can be done for B by sampling at time $t = kT + \tau$ for the same symbol period. The samples are defined as:

$$\begin{aligned} y_k^{(b)} &= \beta T b_k + \tau\alpha a_{k+1} + \alpha T a_k - \tau\alpha a_k + n_k && (3.39)\\ &= \beta T b_k + \tau\alpha(a_{k+1} - a_k) + \alpha T a_k + n_k && (3.40) \end{aligned}$$

Sampling $r_p(t)$ at these different peaks allows us to obtain more information on the signal and in turn, aides in the decoding process of the mixed signal. One important note is that (3.28) does not demonstrate or show the correlation between the noise component and the senders' signals. Without the step in Eq. 3.26, the equation would have taken the form of a sum of autocorrelation functions. Based on where the sampling points of $r_p(t)$ are, there may (non-integer multiples of T) or may not be (otherwise) a correlation between the noise and all the previous samples of A and B. Since, in our design, the samples are not integer multiples of T apart, noise is correlated. The derivation and proof of this notion is beyond the scope of this research and thesis.

2. Viterbi Decoder Design

The next step is to design our Viterbi decoder. The states of the decoder will depend on combinations (or pairs) of a_k and b_k. This is shown in Table VIII. At symbol

Table VIII. State Assignments for a_k and b_k

(a_k, b_k)	State #
1,1	1
1,-1	2
-1,1	3
-1,-1	4

period k, the state transitions will be based on the present values of a_k and b_k, the possible values of a_{k+1} and b_{k+1} and the mean of both $y_k^{(b)}$ ($\mu_k^{(b)}$) and $y_{k+1}^{(a)}$ ($\mu_{k+1}^{(a)}$). The means are similar to the original equations except that the term n_k is dropped. This is because the noise component n_k has an expected value $E[n(t)]$ equal to 0 in the AWGN channel. For any symbol period, the samples will be taken at B's peak within that period and at A's peak, NOT in the beginning of the period, but at the end of the symbol period. This allows for the decoder to be accurately modeled as a state machine. This is why the transition is dependent on $\mu_{k+1}^{(a)}$ and NOT on $\mu_k^{(a)}$. Table IX shows the truth table for the mean values of the sampling points for the different combinations of A and B's bits during symbol period k. Based on what is received at the sampling points, it will be compared to both means for each row of the truth table. The valid symbol periods range from 0 to $N-2$ where N is the number of bits in the packets transmitted. This symbol period restriction does not allow for the initial states to be calculated since $\mu_k^{(b)}$ and $\mu_{k+1}^{(a)}$ are used in determining a_{k+1} and b_{k+1}.

The initial state of the decoder (a_0, b_0) will be calculated first by sampling $r_p(t)$ at time $t = 0$ and utilizing only $\mu_{k+1}^{(a)}$ with $k = -1$ and $b_{-1} = 0$. Next, a state transition

Table IX. Decoding Algorithm Truth Table

Edge	a_k	b_k	a_{k+1}	b_{k+1}	$\mu_k^{(b)}$	$\mu_{k+1}^{(a)}$
1	1	1	1	1	$\beta T + \alpha T$	$\alpha T + \beta T$
2	1	1	1	-1	$\beta T + \alpha T$	$\alpha T + 2\tau\beta - \beta T$
3	1	1	-1	1	$\beta T - 2\tau\alpha + \alpha T$	$-\alpha T + \beta T$
4	1	1	-1	-1	$\beta T - 2\tau\alpha + \alpha T$	$-\alpha T + 2\tau\beta - \beta T$
5	-1	1	1	1	$\beta T + 2\tau\alpha - \alpha T$	$\alpha T + \beta T$
6	-1	1	1	-1	$\beta T + 2\tau\alpha - \alpha T$	$\alpha T + 2\tau\beta - \beta T$
7	-1	1	-1	1	$\beta T - \alpha T$	$-\alpha T + \beta T$
8	-1	1	-1	-1	$\beta T - \alpha T$	$-\alpha T + 2\tau\beta - \beta T$
9	1	-1	1	1	$-\beta T + \alpha T$	$\alpha T - 2\tau\beta + \beta T$
10	1	-1	1	-1	$-\beta T + \alpha T$	$\alpha T - \beta T$
11	1	-1	-1	1	$-\beta T - 2\tau\alpha + \alpha T$	$-\alpha T - 2\tau\beta + \beta T$
12	1	-1	-1	-1	$-\beta T - 2\tau\alpha + \alpha T$	$-\alpha T - \beta T$
13	-1	-1	1	1	$-\beta T + 2\tau\alpha - \alpha T$	$\alpha T - 2\tau\beta + \beta T$
14	-1	-1	1	-1	$-\beta T + 2\tau\alpha - \alpha T$	$\alpha T - \beta T$
15	-1	-1	-1	1	$-\beta T - \alpha T$	$-\alpha T - 2\tau\beta + \beta T$
16	-1	-1	-1	-1	$-\beta T - \alpha T$	$-\alpha T - \beta T$

diagram will be constructed for the states with the output for each transition being $\mu_k^{(b)}$ and $\mu_{k+1}^{(a)}$. It shows the states' dependency on the previous state and the present inputs in determining which path to take. The state diagram is depicted in Fig. 29. The branch metric of the decoder will be defined as the sum of the squared differences between the received signal at each sampling point within symbol period k and the mean values at that same sampling point. We define this branch metric as

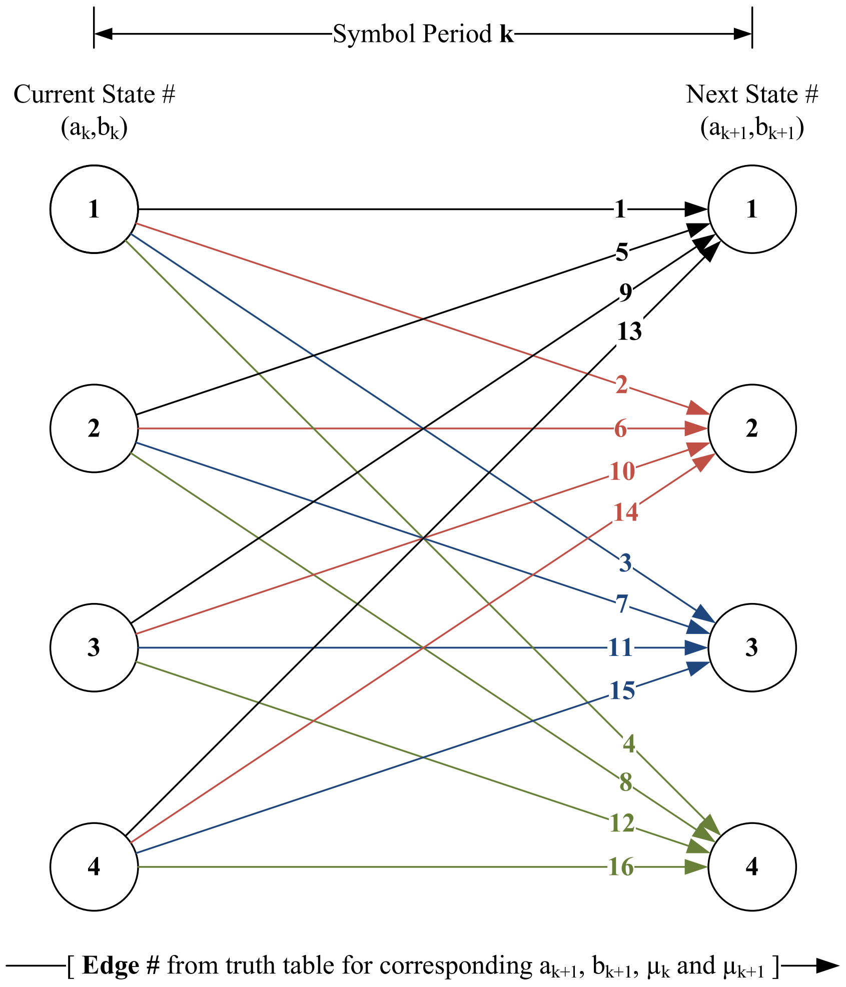

Fig. 29. Decoding algorithm's state transition diagram for (a_k, b_k)

$$B_{i,j}^{(k)} = \left| y_{k+1}^{(a)} - \mu_{k+1}^{(a)} \right|^2 + \left| y_k^{(b)} - \mu_k^{(b)} \right|^2 \tag{3.41}$$

for all $i, j \in 1, 4$. The branch metric can also be stated as

$$B_{i,j}^{(k)} \cong -\ln P(Y_k^{(b)} Y_{k+1}^{(a)} | E_{i,j}) + \kappa \tag{3.42}$$

where κ is a constant. This equation states that the branch metric is congruent to the joint probability of the received signal's sampling points' values given the edge or transition path $E_{i,j}$. During each stage and for each state of the decoder, a running sum of the branch metrics for all the possible paths that might have led to that state will be calculated. When all the symbol periods have been traversed, the total path with the smallest cumulative branch metric sum will be selected as the most likely path. The traceback routine of the Viterbi decoder allows it to trace through the selected paths and provide the sum of the decoded bits $(a_k + b_k)$. The accumulation or running sum of branch metrics at symbol period k can be defined as

$$W_{i_{k+1}}^{(k+1)} = \min_{i_1, i_2, \ldots, i_k \in \{1,4\}} \sum_{l=0}^{k} B_{i_l, i_{l+1}}^{(l)} \tag{3.43}$$

This states that the accumulated weight is based on the best previous branch metrics leading up to the particular state being tested for. This is the main advantage of using Viterbi decoding. It performs an extensive search for the most likely path and its effectiveness is affected by the reliability of the branch metric.

3. Simulations and Results

Now that we have formulated the decoding algorithm, the next question is how does it all work in practice. The value of τ is unknown and the value of B's channel phase distortion is also unknown. The equations for the mean from Table IX depend on these unknown values. Also, these values determine which transition paths are actually traversed in our model. If we are able to predict these unknown values closely, we can be sure that the only performance degrading factor will be the noise introduced from the channel which the matched filter does a great job in minimizing its effect. The next step will be to cycle through test values of the timing offset (τ_{test}) and also for the channel phase distortion on B (β_{test}). This will allow us to key in on the real τ and β. With these test parameters, we plug them into our equations and select which total accumulated sum ($W_j^{(N-1)}$) is the best and add it to the pool of best accumulated sums. The best accumulated sum within this pool will be the one whose corresponding a_k and b_k values will be used for decoding the sum.

To clarify this, our decoding algorithm was simulated in MATLAB. N_1 and N_3 sent 200 bits of data to each other with a 64-bit access code attached to the beginning of their packets ($N = 1064$). Having an access code in a packet allows for the decoded sum to be correlated or aligned with the access code's sum. This is required because the symmetry of the system allows multiple transmitted sequences to generate the same received values (with different τ and β). This proved extremely useful for the cases where β_{test} was π or 180° away from the true β. This allowed for flipped bits to appear correct because they yielded a low branch metric similar to the true value's branch metric. This caused the wrong row to be selected from the truth table in Table IX since both rows produced the same branch metric.

The SNR was selected from the range 0 to 10dB. For each value of the SNR, a variable number of trials was run and the average error rate was calculated for the corresponding SNR. During each simulation, we set $0 \leq \tau < T$ and chose $\angle\alpha$ and $\angle\beta$ to be in the range $(0, 2\pi)$. We then proceeded to cycle through τ_{test} from 0 to T (T was non-inclusive) in $T/10$ increments. For each τ_{test}, we cycled through β_{test} in the range $(0, 2\pi)$ in $\pi/16$ increments. We hard-coded α since it is a parameter that can be estimated ahead of time. We ran each test for the combinations of τ_{test} and β_{test} and whichever combination yielded the smallest accumulated weight sum was chosen as the estimated τ and β. The corresponding decoded sum was what the relay would then send back to the senders.

The next set of figures show the performance of the decoder for different scenarios which comprise of different combinations of SNR, the timing offset and channel phase

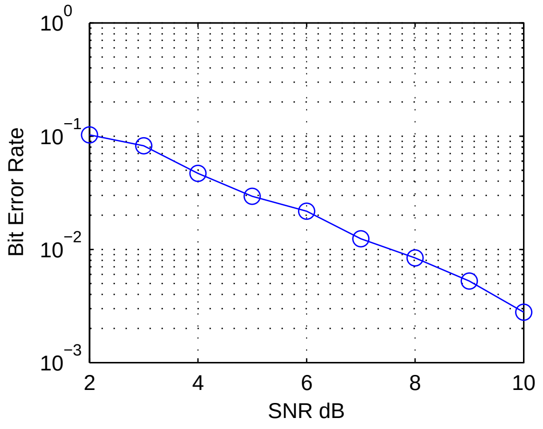

Fig. 30. BER vs. SNR for randomly selected τ, α and β (log-scale)

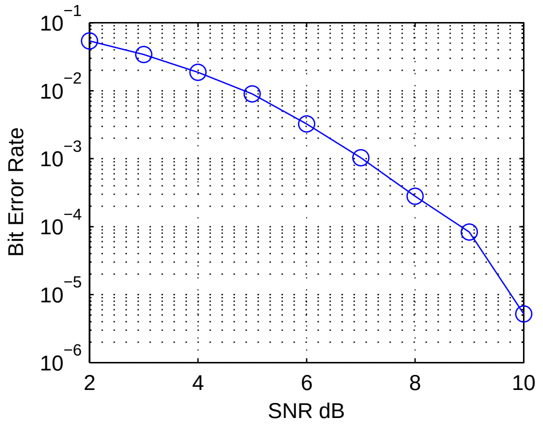

Fig. 31. BER vs. SNR for $\tau = 0$, $\alpha = 1$ and $\beta = 1$ (log-scale)

distortions. Fig. 30 shows that the error rates are higher than those in Fig. 31 where there is phase alignment and no timing offset. The error rates are strictly from the noise present in the channel. Figs. 32 - 36 show more performance results for our decoder. The plots show that our decoder performs well with the only performance degrading factor being noise as expected. For higher SNR values, it performs quite well by yielding low and stable error rates. For SNR regions above 5dB, the error rates were around 2% and below which is lower than previously reported results.

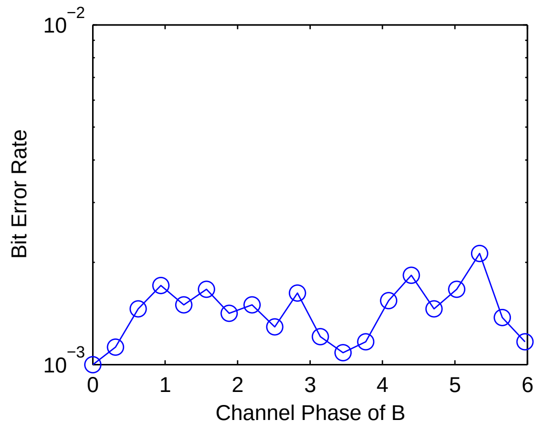

Fig. 32. BER vs. channel phase ($N_3 \rightarrow N_2$) for $\tau = 0$, $\alpha = 1$ and 7dB SNR

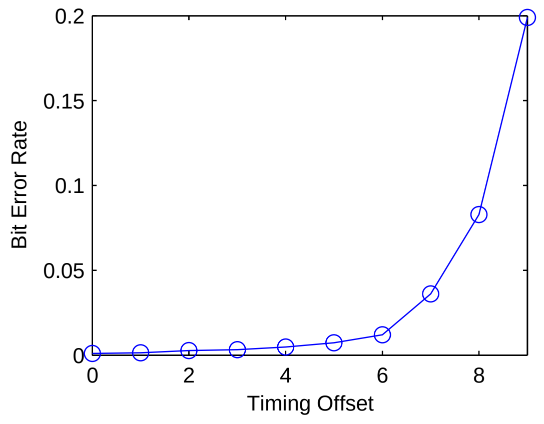

Fig. 33. BER vs. timing offset for $\alpha = 1$, $\beta = 1$ and 7dB SNR

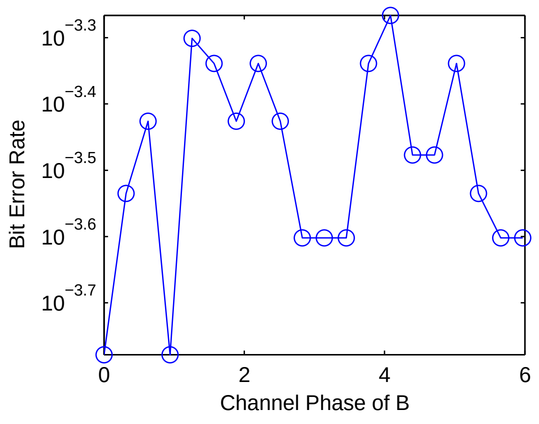

Fig. 34. BER vs. channel phase ($N_3 \rightarrow N_2$) for $\tau = 0$, $\alpha = 1$ and 8dB SNR

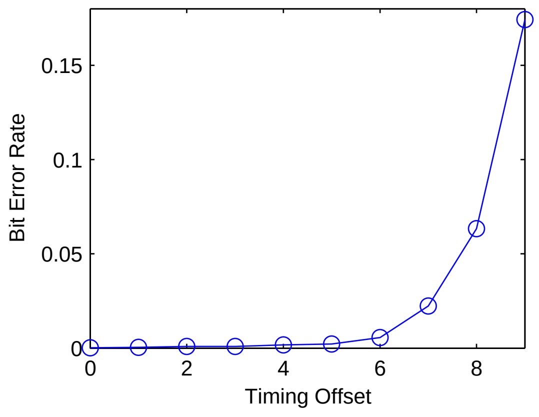

Fig. 35. BER vs. timing offset for $\alpha = 1$, $\beta = 1$ and 8dB SNR

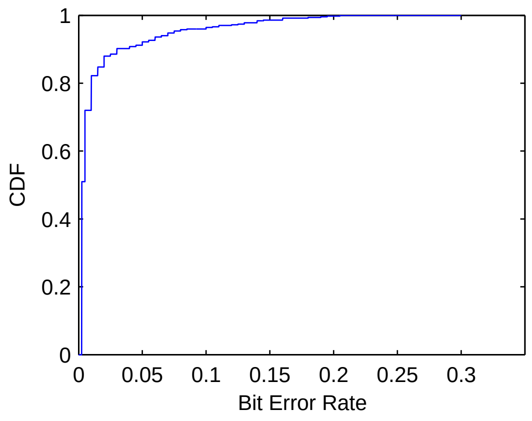

Fig. 36. BER CDF for randomly selected τ, α and β with 7dB SNR

CHAPTER IV

SUMMARY AND CONCLUSIONS

The decoder that we designed from the previous chapter offers a more realistic way to decoding interfered signals since the timing and sampling offsets are taken into consideration. This method cycles through and tests for a wide range of possibilities for these timing offsets as well as the channel phase distortion on the lagging user. This provides us with a more exhaustive approach to decoding the sum of the two transmitted signals. Even though Katti's algorithm is an effective way of decoding the bits via Amplify-and-Forward, it does not consider the timing offsets that exist as the signals mix with each other. The BER performance of our decoder is quite solid especially for low SNR regions. This can be attributed to our choice of using a matched filter to smooth the received signal or to the testing increments used during our exhaustive search. A disadvantage to our design is that it is computationally expensive and might increase the load and work required at the relay. There is an apparent tradeoff between complexity and performance especially since we model the real signals well with all critical channel properties considered. The computations can be reduced by not testing for all values of τ within one symbol period or not testing for all phases from 0 to 2π. These are decisions that are up to the designer or might be dependent on how noisy the wireless channel is.

A. Further Research

There is still more work to be done in this area of wireless communications. The BERs can be improved by possibly choosing a different metric that takes into account previous data collected or processed. We can expand τ to values greater than T to see how timing offsets greater than one symbol period can be decoded. As mentioned earlier, we did not consider the correlation between the noise, A's signal and B's signal when sampling at non-integer multiples of T. This is an important property of the matched filter output that if considered, might improve the performance of our decoder. Overall, our research can be expanded to a wide range of possibilities. We can extend our design to different modulation schemes such as QPSK or utilize different carrier waves. We have created a stepping stone into research that will hopefully continue to contribute and strengthen the field of wireless communications.

REFERENCES

[1] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*, 3rd edition, Boston: Addison Wesley, 2004.

[2] A. Leon-Garcia and I. Widjaja, *Communication Networks: Fundamental Concepts and Key Architectures*, Boston: McGraw-Hill School Education Group, 1999.

[3] E. Ekrem and S. Ulukus, "Secrecy capacity of a class of broadcast channels with an eavesdropper," *CoRR*, vol. abs/0812.0319, 2008.

[4] G. Bagherikaram, A. S. Motahari, and A. K. Khandani, "The secrecy rate region of the broadcast channel," *CoRR*, vol. abs/0806.4200, 2008.

[5] M. P. Wilson, K. Narayanan, H. Pfister, and A. Sprintson, "Joint physical layer coding and network coding for bi-directional relaying," *Allerton Conference on Communication, Control, and Computing*, 2008.

[6] Y. Liang and V.V. Veeravalli, "Cooperative relay broadcast channels," *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, 2005, vol. 2, pp. 1449–1454.

[7] G. Kramer, M. Gastpar, and P. Gupta, "Cooperative strategies and capacity theorems for relay networks," *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 3037–3063, 2005.

[8] A. Nosratinia, T.E. Hunter, and A. Hedayat, "Cooperative communication in wireless networks," *IEEE Communications Magazine*, vol. 42, no. 10, pp. 74–80, 2004.

[9] R. Ahlswede, N. Cai, S.-Y.R. Li, and R.W. Yeung, “Network information flow,” *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[10] S. Zhang, S. C. Liew, and P. P. Lam, “Physical-layer network coding,” in *MobiCom ’06: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, New York, 2006, pp. 358–365.

[11] S. Katti, S. Gollakota, and D. Katabi, “Embracing wireless interference: Analog network coding,” in *SIGCOMM ’07: Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, New York, 2007, pp. 397–408.

[12] J. M. Paredes, B. H. Khalaj, and A. B. Gershman, “Cooperative transmission for wireless relay networks using low-rate feedback,” *IEEE Transactions on Signal Processing*, 2009.

[13] T. J. O’Shea, T. C. Clancy, and H. J. Ebeid, “Practical signal detection and classification in gnu radio,” in *SDR Forum Technical Conference*, 2007.

[14] W. Pu, C. Luo, S. Li, and C. W. Chen, “Continuous network coding in wireless relay networks,” in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008, pp. 1526–1534.

[15] S. Zhang and S. Liew, “The capacity of two way relay channel,” in *Globecom ’08*, 2008.

[16] S. Katti, H. Rahul, Wenjun H., D. Katabi, M. Medard, and J. Crowcroft, “Xors in the air: Practical wireless network coding,” *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 497–510, 2008.

[17] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 159–170, 2008.

[18] J. G. Proakis and D. K. Manolakis, *Digital Signal Processing*, Upper Saddle River, NJ: Prentice Hall, 4th edition, 2006.

[19] B. Sklar, *Digital Communications*, Upper Saddle River, NJ: Prentice Hall, 2000.

[20] S. Katti, "Network coded wireless architecture," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA: 2008.

VITA

Dumezie Maduike was born in Glasgow, Scotland. He moved to Nigeria at the age of 1 and to the United States at the age of 9. As a Carr Scholar, he received his Bachelor of Science degree from Rutgers University, New Brunswick, with High Honors, in Electrical and Computer Engineering on May 2007. He entered his Master of Science degree program in the Telecommunications & Signal Processing group within the Electrical Engineering department of Texas A&M University on August 2007. He received his Master of Science degree in Electrical Engineering in August 2009. His research interests are in wireless communications and signal processing. Dumezie Maduike can be reached at 214 Zachry Engineering Center, College Station, Texas 77843-3128.