Embracing Wireless Interference: Analog Network Coding

Sachin Katti
MIT CSAIL
skatti@mit.edu

Shyamnath Gollakota
MIT CSAIL
gshyam@mit.edu

Dina Katabi
MIT CSAIL
dina@csail.mit.edu

ABSTRACT

Traditionally, interference is considered harmful. Wireless networks strive to avoid scheduling multiple transmissions at the same time in order to prevent interference. This paper adopts the opposite approach; it encourages strategically picked senders to interfere. Instead of forwarding packets, routers forward the interfering signals. The destination leverages network-level information to cancel the interference and recover the signal destined to it. The result is *analog network coding* because it mixes signals not bits.

So, what if wireless routers forward signals instead of packets? Theoretically, such an approach doubles the capacity of the canonical 2-way relay network. Surprisingly, it is also practical. We implement our design using software radios and show that it achieves significantly higher throughput than both traditional wireless routing and prior work on wireless network coding.

Categories and Subject Descriptors

C.2.2 [**Computer Systems Organization**]: Computer-Communications Networks

General Terms

Algorithms, Design, Performance, Theory

Keywords

Network Coding, Wireless Networks, Cooperative Transmission

1. INTRODUCTION

Wireless interference is typically considered harmful. Wireless networks strive to prevent senders from interfering. They may reserve the medium to a specific node using TDMA or probe for idleness as in 802.11. This fear of interference is inherited from single-channel design and may not be the best approach for a wireless network [19, 25, 30, 31]. With bandwidth being scarce in the frequencies allocated to wireless networks, it is desirable to enable concurrent receptions despite interference.

This paper presents Analog Network Coding (ANC). Instead of avoiding interference, we exploit the interference of strategically picked senders to increase network capacity. When two senders transmit simultaneously, the packets collide. The signal resulting from a collision, however, is nothing but the sum of the two colliding signals after incurring attenuation and phase and time shifts. Thus, if the receiver knows the content of the packet that interfered with the packet it wants, it can cancel the signal corresponding to that known packet after correcting for channel distortion. The receiver is left with the signal of the packet it wants, which it decodes using standard methods. In a wireless network, when two packets collide, nodes often know one of the colliding packets by virtue of having forwarded it earlier or having overheard it. Thus, our approach encourages two senders to transmit simultaneously if their receivers can leverage network-layer information to reconstruct the interfering signal, and disentangle it from the packet they want.

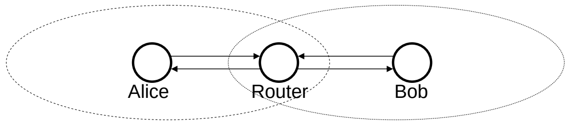
Note the analogy between analog network coding and its digital counterpart. In digital network coding, senders transmit sequentially, and routers mix the content of the packets and broadcast the mixed version [17]. In ANC, senders transmit simultaneously. The wireless channel naturally mixes these signals. Instead of forwarding mixed packets, routers amplify and forward the mixed signals they receive.

Prior work in information theory has noted the potential for analog network coding and shown that, in theory, it doubles the capacity of the canonical 2-way relay network [16, 22, 27, 28, 30]. Prior work, however, focuses on capacity bounds and does not provide an algorithm for delivering the resulting throughput benefits. This is with the exception of [29], which describes an algorithm for physical-layer network coding. But, the algorithm therein assumes symbol-level synchronization, carrier-frequency synchronization, and carrier-phase synchronization. In practice, however, it is unlikely that two signals arrive at the exact same time at the router and incur the same distortion over the wireless medium.

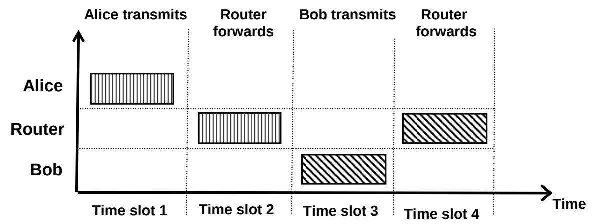
Our work builds on prior foundations, but provides an algorithm for analog network coding that makes no synchronization assumptions. Indeed, our approach exploits the lack of synchronization between interfering signals and enforces it by inserting random delays before a transmission. Lack of synchronization means that the two signals do not perfectly align; one signal starts first with a few bits that do not interfere with the other signal, while the second signal ends last with a few bits that do not interfere with the first signal. The receivers use these interference-free bits on both sides of the interfered signal to estimate the wireless channels from the two senders, compensate for their effects on the packets they know, and properly decode the packets they want.

This paper is the first to present a practical design that exploits analog network coding to increase network throughput. Our contributions can be summarized as follows:

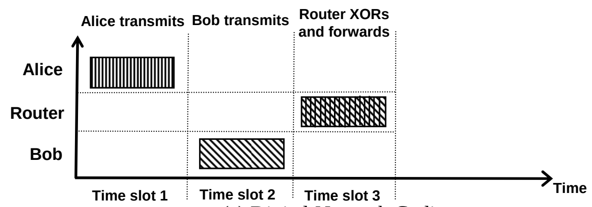
- We present a novel algorithm for analog network coding that does not make any synchronization assumptions. We further show how our algorithm codes packets within a single flow, and across different flows that intersect at a router.
- We implement our approach in software radios, demonstrating its practicality.
- We evaluate our implementation in a testbed of software radios. Empirical results show that our technique decodes interfered packets with an average bit error rate as low as 2-4%, which is masked by the use of error correcting codes. As for the throughput, it increases by 70% in comparison to traditional wireless routing, and by 30% in comparison to digital network coding.



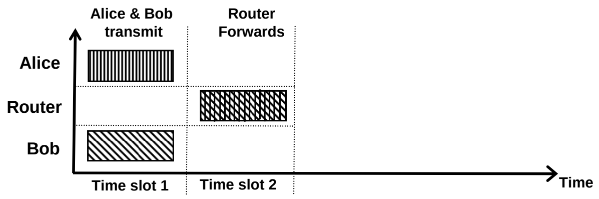
(a) Alice-Bob topology. Dotted lines show the radio range.



(b) Traditional Approach



(c) Digital Network Coding



(d) Analog Network Coding

Figure 1—Alice-Bob Topology: Flows Intersecting at a Router. With analog network coding, Alice and Bob transmit simultaneously to the router, the router relays the interfered signal to Alice and Bob, who decode each others packets. This reduces the number of time slots from 4 to 2, doubling the throughput.

2. ILLUSTRATIVE EXAMPLES

We illustrate the benefits of analog network coding using two canonical topologies, common in a mesh network. These two examples constitute building blocks for larger networks.

(a) Flows Intersecting at a Router: Consider the canonical example for wireless network coding shown in Fig. 1(a) [17]. Alice and Bob want to send a message to each other. The radio range does not allow them to communicate without a relay. In traditional 802.11, Alice sends her packet to the router, which forwards it to Bob, and Bob sends his packet to the router, which forwards it to Alice. Thus, to exchange two packets, the traditional approach needs 4 time slots. Network coding achieves the same goal, but with fewer time slots. In particular, Alice and Bob send their packets to the router, one after the other; the router then XORs the two packets and broadcasts the XOR-ed version. Alice recovers Bob's packet by XOR-ing again with her own, and Bob recovers Alice's packet in the same way. Thus, network coding reduces the number of time slots from 4 to 3. The freed slot can be used to send new data, improving wireless throughput.

But, can we reduce the time slots further? Can we deliver both packets in 2 time slots? The answer is "yes". Alice and Bob could transmit their packets simultaneously, allowing their transmissions to interfere at the router. This consumes a single time slot. Due to interference, the router receives the sum of Alice's and Bob's signals, $s_A(t) + s_B(t)$. This is a collision and the router cannot decode the bits. The router, however, can simply amplify and forward the received interfered signal at the physical layer itself without decoding it. This consumes a second time slot. Since Alice knows the packet she transmitted, she also knows the signal $s_A(t)$ corresponding to her packet. She can therefore subtract $s_A(t)$ from the received interfered signal to get $s_B(t)$, from which she can decode Bob's packet. Bob can similarly recover Alice's packet. We call such an approach analog network coding (ANC). It is analogous to digital network coding but is done over physical signals in the wireless channel itself. As a result, ANC reduces the required time slots from 4 to 2, doubling the wireless throughput.

(b) Flows in a Single Direction: Analog network coding, not only increases the throughput beyond digital network coding, it also applies to new scenarios to which traditional digital network coding would not apply. Consider the chain topology in Fig. 2(a), where a single flow traverses 3 hops. The traditional routing approach needs 3 time slots to deliver every packet from source to destination. Digital network coding cannot reduce the number of time slots in this scenario, but analog network coding can.

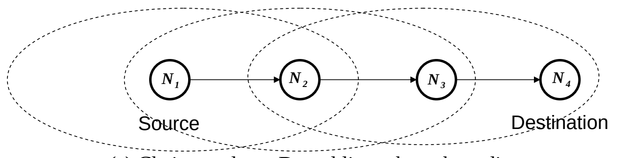
Analog network coding improves the throughput of the chain topology in Fig. 2(a) because it allows nodes N_1 and N_3 to transmit simultaneously and have their packets received correctly despite collisions. In particular, let node N_2 transmit packet p_i to N_3. Then, N_1 transmits the next packet p_{i+1}, and N_3 forwards p_i to N_4. These two transmissions happen concurrently. The destination, N_4, receives only p_i because it is outside the radio range of node N_1. But, the two packets collide at node N_2. With the traditional approach, N_2 loses the packet sent to it by N_1. In contrast, in our approach, N_2 exploits the fact that it knows the data in N_3's transmission because it forwarded that packet to N_3 earlier. Node N_2 can recreate the signal that N_3 sent and subtract that signal from the received signal. After subtraction, N_2 is left with the signal transmitted by N_1, which it can decode to obtain packet p_{i+1}. Thus, instead of requiring a time slot for transmission on each hop, we can transmit on the first and third hops simultaneously, reducing the time slots from 3 to 2. This creates a throughput gain of $3/2 = 1.5$.

In practice, the throughput gain of the chain topology may be even higher. Without analog network coding, the nodes need an added mechanism to handle the hidden terminal problem in Fig. 2(a). They may use RTS-CTS or a statistical method like the exponential backoff built into the 802.11 MAC. Both methods incur a cost and reduce the achievable throughput [1]. With our approach, hidden terminals are harmless, and there is no need for an additional synchronization mechanism beyond carrier sense. ANC therefore solves the hidden terminal problem for chain topologies with both uni-directional as well as bi-directional traffic. Addressing the hidden terminal problem in general ad-hoc networks, however, is beyond the scope of this paper.

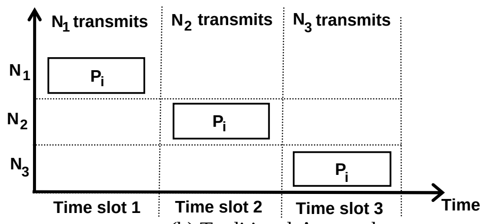
The description above has intentionally ignored important details. For analog network coding to become practical, we need to address the following important challenges.

- The wireless channel distorts the signals, and hence Alice and Bob cannot simply subtract the signal they sent from the one they received to obtain each other's packet. They need to compensate for channel effects before they can cancel the interfering signal.
- Also, it is impossible for Alice's and Bob's transmissions to be fully synchronized. There will be a time shift between the two signals. A practical design has to work despite lack of synchrony between the interfering signals.

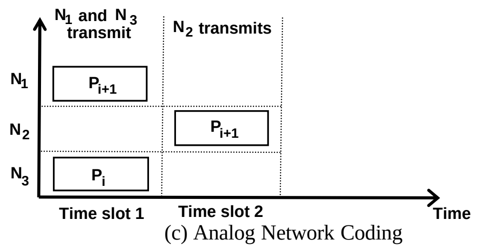
The rest of this paper shows how to address these two challenges. Our main observation is that we can exploit the lack of synchroniza-



(a) Chain topology. Dotted lines show the radio range.



(b) Traditional Approach



(c) Analog Network Coding

Figure 2—Chain Topology: Flows in one Direction. Nodes N_1 and N_3 can transmit at the same time. N_2 gets an interfered signal, but can recover N_1's packet because it already knows the content of the interfering signal sent by N_3. This reduces the time slots to deliver a packet from 3 to 2, producing a throughput gain of $3/2 = 1.5$.

tion instead of ignoring it. Since Alice's and Bob's signals do not perfectly align, there will be a few interference-free bits at the beginning and the end of the interfering signal. Alice can use these interference-free bits to estimate how the channel modified her signal. Alice then applies the same modifications to the signal she transmitted, cancels out the resulting signal from the interfering signal, and obtains Bob's signal.

3. RELATED WORK

Prior work falls into the following three categories.

(a) Theoretical Work on the Capacity of the 2-Way Relay Network: Our work builds on prior work on the capacity of the 2-way relay channel (i.e., the Alice-Bob topology) [16, 22, 27, 28, 30]. This work, however, focuses on theoretical capacity bounds. It does not provide an algorithm for delivering the resulting throughput benefits. Also, it assumes symbol-level synchronization and the channel functions being known.

Our work is closely related to the work on physical layer network coding [29], which provides an algorithm for disentangling two physical-layer signals using higher-layer information. The approach in [29] makes the router decode the interfered signal and re-encode a different packet to Alice and Bob, whereas in our approach the router simply amplifies and forwards the interfered signal. More importantly, the algorithm in [29] assumes that the interfering signals are synchronized at the symbol boundaries; it is unclear how to handle signals that do not arrive synchronized at the router. Second, it assumes that both signals have undergone the same attenuation when arriving at the router. Third, it assumes that different channels do not introduce different phase-shifts, which is unlikely given the propagation delays in the channels. In contrast, our work makes no assumptions about phase shifts, attenuation or synchronization. Further, we implement our design in a wireless network using software radios and demonstrate its throughput gains.

(b) Multiple Access and Space Time Coding: Multiple access techniques like CDMA [24], FDMA [31], and spatial reuse [31] allow multiple transmissions at the same time. These approaches, however, are simply means to avoid interference in code, frequencies, or space. They just divide channel capacity among multiple users. In contrast, ANC expands the capacity of the network.

Co-operative diversity [19], analog forwarding [25] and MIMO systems [31] allow multiple concurrent transmissions using space-time coding techniques [31]. Some of this work assumes antenna arrays and coherent combining at the receiver, which we do not assume. More importantly, these techniques differ from ANC because they do not exploit the receiver's knowledge of one of the interfering signals to expand the capacity of the network.

Also some related work falls in the area of interference cancellation and blind signal separation. These schemes decode two signals that have interfered without knowing any of the signals in advance [32, 3, 10]. Practical work in this domain is limited to signals that differ in their characteristics. For example, it is typical to assume that one signal has much higher power than the other. We can decode the higher power signal assuming that the lower power signal is noise, and then remove the high power signal to decode the weaker signal. Our algorithm makes no such assumptions and can operate on signals of the same strength.

(c) Traditional Network Coding: Work on network coding started with a paper by Ahlswede et al. that establishes the benefits of coding in routers and bounds the capacity of such networks [2]. This work has been extended by papers on linear network codes [20, 18, 14], randomized coding [11], wireless network coding [6, 21, 26, 17, 33, 23, 4], and network coding for content distribution [5]. All of the above mix bits in routers or hosts. In contrast, our work makes the senders transmit concurrently and has the wireless channel mix the analog signals representing the packets.

4. SCOPE

Analog network coding is a general technique, independent of the underlying wireless technology. It is applicable in a wide variety of scenarios, with 802.11 mesh networks being an obvious example. Cellular networks are another possible example. Particularly, cellular networks deploy inexpensive bi-directional relays to expand their coverage area. These nodes intervene between the mobile device and the base station. They simply amplify and retransmit the signal they receive [7], which is exactly the functionality they need to implement analog network coding.

Our goal is to design and implement a proof of concept of ANC. Since ANC works at the signal level, this implies designing an entire communication system from the ground up. Hence, we have to make a number of design choices at the physical layer. Most importantly, we have to choose a modulation/demodulation scheme. We picked MSK because the GNURadio software project [8] has a mature MSK implementation. This facilitates implementing our design in software radios and allows us to start with an already functional DSP code. MSK also has very good bit-error properties and a simple demodulation algorithm, and constitutes the basis for GMSK which is used in the GSM cell-phone standard.

5. BACKGROUND: A SINGLE SIGNAL

Before talking about disentangling interfering signals, we need to explain how a single signal is transmitted and received over the wireless channel. For the sake of simplicity, we will intentionally gloss

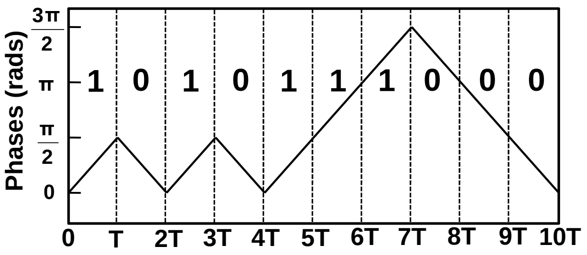


Figure 3—Example MSK modulation. MSK represents a "1" bit of as a phase difference of $\pi/2$ over for an interval T. It represents a a "0" bit as a phase difference of $-\pi/2$ over T.

over some details that are unnecessary for understanding the technical ideas proposed in this paper (e.g., pass-band vs. base-band, error correction, upconversion, and downconversion). We will describe how MSK transmits and receives a packet of bits.

5.1 The Sender Side

A wireless signal is represented as a complex and discrete function of time. Thus, the signal transmitted by the sender, which we annotate with the subscript s, can be represented as:

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

where $A_s[n]$ is the amplitude of the nth sample and $\theta_s[n]$ is its phase.

Say that we have a packet to transmit over the wireless channel. We need to map "0" and "1" into two different complex representations. This is called modulation. In particular, the MSK modulation represents bits by varying the phase difference between consecutive complex samples. A phase difference of $\pi/2$ represents "1", whereas a phase difference of $-\pi/2$ represents a "0".

To see how MSK works, let us go through an example. Assume the data being sent is 1010111000, the phase of the signal would then vary as seen in Fig. 3. The signal itself is the complex function whose phase changes as shown in the figure. Initially, at time $t = 0$, the signal is $A_s e^{i0}$. Since the first bit is a "1", the signal sample at time $t = T$ should be $A_s e^{i(\pi/2)}$. The second bit is a "0", hence the signal sample at time $t = 2T$ should be $A_s e^{i(\pi/2-\pi/2)} = A_s e^{i0}$. This is repeated for all the bits. Note that in MSK, the amplitude of the transmitted signal, A_s, is a constant. The phase embeds all information about the bits.

5.2 The Receiver Side

What does the signal look like at the receiver, after traversing the wireless channel? The received signal is also a stream of complex samples spaced by the sampling interval T. These samples differ, however, from the transmitted samples, both in amplitude and phase. In particular, if the transmitted sample is $A_s[n]e^{i\theta_s[n]}$ the received signal can be approximated as:

$$y[n] = h\,A_s[n]e^{i(\theta_s[n]+\gamma)},$$

where h is channel attenuation and γ is a phase shift that depends on the distance between the sender and the receiver.[1]

The receiver needs to map the received complex samples back into the transmitted bit stream, i.e., it needs to demodulate the received signal. For MSK, this amounts to discovering the phase differences between consecutive complex samples separated by T, and then mapping that phase difference back to a bit value.

Calculating phase differences of the complex samples is simple. Recall that in MSK, the amplitude of the samples is fixed and does not change from one signal sample to the next. Consider the following consecutive complex samples $h\,A_s\,e^{i(\theta_s[n]+\gamma)}$ and $h\,A_s\,e^{i(\theta_s[n+1]+\gamma)}$. First, we calculate the ratio of these complex numbers,

$$r = \frac{h\,A_s\,e^{i(\theta_s[n+1]+\gamma)}}{h\,A_s\,e^{i(\theta_s[n]+\gamma)}} = e^{i(\theta_s[n+1]-\theta_s[n])}. \qquad (1)$$

To demodulate, we simply compute the angle of the complex number r, which gives us the phase difference, $\theta_s[n+1] - \theta_s[n]$. Ideally, "1" maps to a phase difference of $\frac{\pi}{2}$ and "0" to a phase difference of $-\frac{\pi}{2}$.[2] But channel noise and estimation errors make the decoded phase differences differ from the ideal scenario. Still the bits can be decoded using a simple rule: a positive phase difference is a "1", whereas a negative phase difference is a "0".

The most important fact about the computation in Eq. 1 is its invariance to both the channel attenuation h and the channel phase shift γ. This makes MSK demodulation very robust because the receiver does not need to accurately estimate the channel.

6. DECODING INTERFERED MSK SIGNALS

So, how does Alice (or Bob) decode the interfered signals? The first step in answering this question is to understand what Alice receives. As described earlier, when Alice and Bob transmit simultaneously, the router receives the sum of their signals, amplifies this composite signal, and broadcasts it to Alice and Bob. Thus, Alice receives an interfered signal, $y_A[n] + y_B[n]$. However, $y_A[n]$ and $y_B[n]$ are not the two signals Alice and Bob have sent. Rather, they are the two transmitted signals after they traversed the channels from their corresponding senders to the router and the channel from the router to their corresponding receivers. The effect of the wireless channels can be approximated by an attenuation and phase shift [31]. Thus, the signal that Alice receives is:

$$\begin{aligned} y[n] &= y_A[n] + y_B[n] \\ y[n] &= h' A_s e^{i(\theta_s[n]+\gamma')} + h'' B_s e^{i(\phi_s[n]+\gamma'')}, \end{aligned}$$

where θ_s refers to the phase of the signal transmitted by Alice and ϕ_s refers to the phase of the signal transmitted by Bob, whereas A_s and B_s are the amplitudes at the transmitter.

Note that we use the subscript s to refer to the transmitted signal as opposed to the received signal, for which we use no subscripts. Note also that n refers to the index of the received sample as sampled by Alice's receiver. It does not mean that the n^{th} bit in Alice's signal is aligned with the n^{th} bit in Bob's signal.[3]

One approach to decoding the interfered signals would have Alice estimate the channel parameters h' and γ'. Once she knows these parameters, Alice recreates the version of her signal that interfered with Bob's signal, and subtracts it from the received signal. The result is $y_B[n]$, a sampled version of Bob's signal that Alice can decode using the standard method described in §5. Though estimating h' is relatively easy (we show how to estimate h' in §6.2), estimating γ' is more complex and requires accurate coherent phase tracking algorithms. Since MSK can be decoded non-coherently (without any phase tracking algorithms), we can simplify the decoder by taking advantage of this property. Specifically, Eq. 1 computes the phase difference without worrying about the exact values of γ. This gives us a hint of how to design a non-coherent demodulation scheme for interfered signals. One should focus on discovering the phase differences for the two signals, namely $\Delta\theta$ and $\Delta\phi$. Estimating phase

[1]This models flat-fading quasi-static channels.

[2]For clarity, we will ignore over-sampling and assume one sample per bit/symbol. Our implementation has an over-sampling factor of 2.

[3]Our design does not assume synchronization of Alice's and Bob's signals. We will talk about that issue in detail in §7.2.

differences does not require the value of γ and thus obviates the need for phase tracking. It is phase differences that carry all information about Alice's and Bob's bits, not the values of the phases themselves.

Thus, in the rest of this section, we develop an algorithm that allows Alice to decode the phase differences between the consecutive samples of Bob's signal. For simplicity of notation, we will represent the received signal at Alice as:

$$y[n] = Ae^{i\theta[n]} + Be^{i\phi[n]}, \tag{2}$$

where $A = h'A_s, B = h''B_s, \theta[n] = \theta_s[n]+\gamma'$, and $\phi[n] = \phi_s[n]+\gamma''$.

How do you calculate phase differences when two signals interfere and you know the phase differences of one of the signals? We will use a two-step process. First, Alice uses her received signal to calculate pairs $(\Delta\theta, \Delta\phi)$ that could have produced the observed signal. Next, Alice uses her knowledge of her phase difference $\Delta\theta_s$ to pick the most likely pair. This gives Alice an estimate of $\Delta\phi$, Bob's phase difference. Based on this estimate Alice decides whether Bob sent a "0" or a "1".

6.1 Possible Phases of Both Signals

Say that Alice receives the interfered signal in Eq. 2, can she tell the values of $\theta[n]$ and $\phi[n]$ just by analyzing the received signal? The answer is "No"; without extra information, Alice cannot tell the exact phases. She can, however, calculate possible values for those phases. In particular, the following lemma is proved in [10, 15].

LEMMA 6.1. *If $y[n]$ is a complex number satisfying Eq. 2, then the pair $(\theta[n], \phi[n])$ takes one of the following two values.*

$$\theta[n] = \arg(y[n](A + BD \pm iB\sqrt{1 - D^2})) \tag{3}$$
$$\phi[n] = \arg(y[n](B + AD \mp iA\sqrt{1 - D^2})) \tag{4}$$

where, $D = \frac{|y[n]|^2 - A^2 - B^2}{2AB}$, $|y[n]|$ is the norm, and arg *is the angle of the complex number.*

Note that for each solution to $\theta[n]$, there is a unique solution for $\phi[n]$. For example, when $\theta[n] = \arg(y[n](A+BD+iB\sqrt{1 - D^2}))$, the corresponding solution is $\phi[n] = \arg(y[n](B + AD - iA\sqrt{1 - D^2}))$. The solutions come in two pairs.

The intuition underlying the proof can be explained geometrically. As a complex number, $y[n]$ can be represented with a vector, as in Fig. 4. According to Eq. 2, $y[n]$ is the sum of two vectors, which have lengths A and B respectively. Thus, we want to find a pair of vectors, (u, v), that sum up to the received complex sample, $y[n]$. The constraint is that the first vector is of length A and the second of length B, i.e., the two vectors lie on two circles with radius A and B. From the figure, there are only two such pairs of vectors. Therefore, there are two solutions for the pair $(\theta[n], \phi[n])$.

6.2 Estimating the Amplitudes A and B

If Alice knows the amplitude of the two signals, i.e., A and B, she can substitute those values and the received complex sample $|y[n]|$ into the equations in Lemma 6.1 to calculate the phases. In fact, Alice can estimate A and B from the received signal. Since she has two unknowns (A and B), she needs two equations.

The first equation for computing A and B comes from the energy of the received signal. When two signals interfere, their energies add up. In particular, the energy is:

$$E[|y[n]|^2] = E[A^2 + B^2 + 2AB\cos(\theta[n] - \phi[n])],$$

where $E[.]$ is the expectation. The value of $E[\cos(\theta[n] - \phi[n])] \approx 0$ for a random bit sequence. To ensure the bits are random, we XOR them with a pseudo-random sequence at the sender, and XOR them

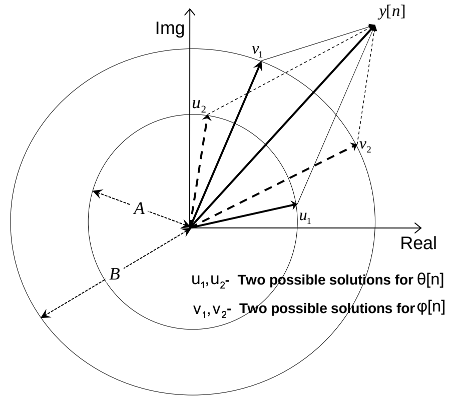

Img
y[n]
v1
u2
v2
A
u1
Real
B
u1,u2- Two possible solutions for θ[n]
v1,v2- Two possible solutions for φ[n]

Figure 4—Geometric representation of the phase computation. the received complex sample $y[n]$ is the sum of two complex numbers u and v. The length of the first complex number is A and the length of second is B. There are exactly two pairs of such complex numbers (u, v) that sum up to $y[n]$. Thus, two solutions exist for the pair $(\theta[n], \phi[n])$.

again with the same sequence at the receiver to get the original bits. Hence,

$$E[|y[n]|^2] = \mu = A^2 + B^2. \tag{5}$$

Alice estimates the expectation by averaging the energy of the complex samples over a window of size N.

Alice still needs a second equation to estimate A and B. She computes the following quantity.

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

Said differently, Alice computes the average energy in samples whose squared norm is greater than the mean energy μ. It can be shown that σ can be reduced to [10, 15],

$$\sigma = A^2 + B^2 + 4AB/\pi. \tag{6}$$

Given Eqs. 5 and 6, Alice has two equations with two unknowns and can solve for A and B.

Note that because of the lack of synchronization between Alice and Bob's signals, there are a few bits at the beginning and end of the interfered signal that are interference free. One can use these interference-free bits to estimate A and B directly. Such estimates however are a bit noisier than the ones described above.

6.3 Estimating Phase Differences for Bob's Signal

Next, we describe the main step in our algorithm, i.e., we describe how Alice estimates the phase differences of Bob's signal, $\phi[n+1] - \phi[n]$.

Alice uses the phases from Lemma 6.1 to calculate phase differences of both her signal, $\theta[n + 1] - \theta[n]$, as well as Bob's signal $\phi[n + 1] - \phi[n]$. There is, however, ambiguity in these calculations because this lemma gives two solutions for each phase, at any sample time n. Alice cannot tell which of the two solutions is the correct one. Alice therefore computes all possible phase differences based on Lemma 6.1. Let us denote the two solutions pairs as $(\theta_1[n], \phi_1[n])$ and $(\theta_2[n], \phi_2[n])$. Then, Alice has the following four possible phase difference pairs:

$$(\Delta\theta_{xy}[n], \Delta\phi_{xy}[n]) = (\theta_x[n + 1] - \theta_y[n], \phi_x[n + 1] - \phi_y[n]) \quad \forall x, y \in \{1, 2\} \tag{7}$$

Alice has to pick the right phase difference pair from the four choices in Eq. 7. This is where she leverages *network layer information*. Alice knows the signal she transmitted earlier, and which interfered with Bob's signal. Thus, she knows the phase difference of her transmission $\Delta\theta_s[n]$. Phase differences are fairly robust to channel distortion (if you take the phase difference the γ term cancels out). Thus, she can use the known $\Delta\theta_s[n]$ to pick the correct $\Delta\theta_{xy}$.

Alice calculates the error for each of the four choices she got from Eq. 7.

$$err_{xy} = |\Delta\theta_{xy}[n] - \Delta\theta_s[n]| \quad, \forall x, y \in \{1,2\} \tag{8}$$

Alice picks the $\Delta\theta_{xy}[n]$ that produces the smallest error err_{xy}. She finds the matching $\Delta\phi_{xy}[n]$ phase difference for Bob's signal. Alice repeats this for all values of n, to estimate the sequence of Bob's phase differences. She uses these estimated phase differences to decode Bob's bits.

6.4 Obtaining Bob's Bits

Recall that MSK modulation maps "1" to a phase difference of $\pi/2$ and "0" to a phase difference of $-\pi/2$. In the last step above, Alice has an estimate of the phase differences of Bob's signal, $\Delta\phi[n]$. She now maps them back to bits. Because of estimation errors and the distortion of the received signal, the phase differences that Alice estimates do not match exactly the phase differences sent by Bob. Thus, Alice follows a simple rule.

if $\Delta\phi[n] \geq 0$, the n^{th} bit is "1", else it is "0".

7. PRACTICAL ISSUES

Is the scheme described above feasible in practice? The short answer is "yes". Building an operational communication system, however, involves many practical challenges.

7.1 How Does Alice Detect Interference?

We begin with the most basic question: How does Alice detect a packet reception? This is a standard problem in communication systems. To detect a reception, Alice looks at the energy in the received signal. During reception the energy level is much higher than the noise energy.

Next, how can Alice tell whether a packet has been subjected to interference? If it is an interfered packet, Alice needs to run the interference decoding algorithm described in §6; otherwise, Alice runs standard MSK decoding.

To answer this question, Alice uses the variance in the energy of the received signal. Recall that, in MSK, the transmitted signal amplitude is constant; MSK encodes the bits in the phase, not the magnitude of the complex sample. Hence, the energy of a non-interfered MSK signal is nearly constant.[4] Packet interference destroys this property of nearly constant signal energy. When two packets collide, the signals interfere with each other in a random fashion. The constant energy property of MSK no longer holds. We use this insight to detect interference. We quantify this variation in energy by measuring the variance in the energy of the received samples. If the variance is greater than a threshold, Alice detects interference and applies the decoding algorithm from §6.

7.2 Dealing with Lack of Synchronization

In an ideal world, Alice's and Bob's signals arrive at the router at the same instant, and interfere exactly at the beginning of the two packets. In reality, there is a time shift between the two signals. This time shift complicates our algorithm described in §6. In particular,

[4]The energy of a complex sample $Ae^{i\theta}$ is A^2.

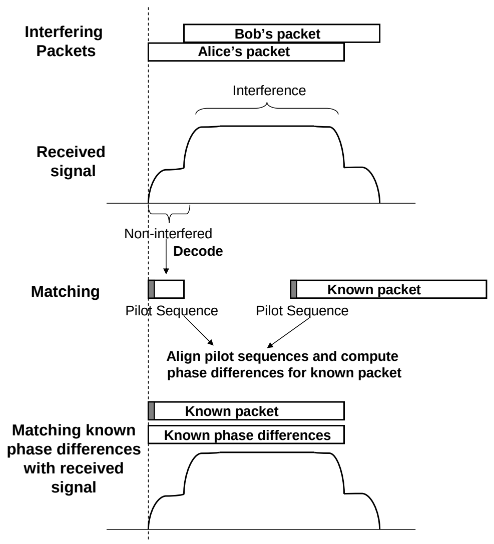


Figure 5—Aligning known phase differences with received signal. Alice finds where her packet starts using the pilot bits at the beginning of the packet, which are interference free. Bob, whose packet starts second, uses the pilot bits at the end of the packet and runs the alignment process backward.

the algorithm needs Alice to match the phase difference of the signal she sent against four possible solutions, in order to pick the right one. Without synchronization, however, Alice does not know the index of the first interfering sample.

Interestingly, our solution to the problem leverages the lack of complete synchronization. Since packets do not interfere perfectly, there are bits at the start and end of the received signal which do not have any interference. For example, assume Alice's signal arrived before Bob's. Then, the first few bits of Alice's packet are interference free. Assuming Alice and Bob have similar packet sizes, the last few bits of Bob's packet are also interference free.[5] Indeed, our approach enforces this incomplete overlap between the two packets to ensure that there are a few bits at the beginning and end of the interfered signal that are interference free, which can be used to synchronize. Specifically, we use a randomization scheme similar to 802.11 MAC. Nodes start their transmission after a *random delay*. They do this by picking a random number between 1 and 32, and starting their transmission in the corresponding time slot. The size of a slot should depend on the transmission rate, modulation scheme, etc.

Our solution attaches a *known pilot bit sequence* to the beginning of each packet. It also attaches a mirrored version of the pilot sequence to the end of the packet. The pilot is 64-bit length. It is used to synchronize with the beginning of the known packet.

We describe our solution assuming Alice's packet starts first. Bob's decoding algorithm is described in §7.5. Alice first detects the beginning of a packet using the *energy detector* from §7.1. She then looks for the known pilot sequence in the interference-free part of the signal at the start of the packet. She decodes this part using standard MSK demodulation. Fig. 5 displays the matching process that Alice performs over the received signal. After decoding the interference-free part, she tries to match the known pilot sequence with every sequence of 64 bits. Once a match is found, she aligns her known signal with the received signal starting at that point, i.e., starting at the end of the pilot. If Alice fails to find the pilot sequence, she drops the packet.

[5]Similarly to digital network coding, when two packets are not the same size the shorter packet is padded.

At the end of the pilot sequence, Alice starts applying the algorithm in §6 that detects the two interfering signals. By then Bob's signal might not have started yet. Despite this Alice can still apply our decoding algorithm from §6. The values for the initial estimated phase differences, $\Delta\phi[n]$ could be random and dependent on the noise since Bob's signal might not have started yet. Once Bob's signal starts, the estimated phases differences $\Delta\phi[n]$, will correspond to the pilot sequence at the start of Bob's packet. At that point, Alice detects the beginning of Bob's packet.

Thus, the pilot sequence helps Alice align her own sent signal with respect to the received signal. It also helps her detect the beginning of Bob's signal in the received signal.

7.3 Channel and Hardware Distortions

So far, we have assumed that phase differences remain unchanged through a wireless channel. In practice, phase differences also get distorted due to hardware limitations and channel effects. Fortunately, Alice can use the pilot sequence in the interference-free bits to estimate these distortions. Alice then applies the same distortions to her known phase differences before comparing them to the phase differences in the received signal; i.e., she applies the distortions to $\Delta\theta_s$ before computing the errors in Eq. 8 to ensure that the values she compares against match those in the received signal. Below, we describe each of these distortions and how we estimate them.

Frequency Offset: Transmitters and receivers use oscillators to transmit or receive signals at the desired carrier frequency (e.g., 2.45 GHz for 802.11b/g). These oscillators have fundamental limitations; they cannot be tuned exactly. There is always some error between the desired frequency and the one the oscillator locks to. Due to this error, there is an offset between the center frequencies at the transmitter and the receiver. This frequency offset causes a displacement of the phase differences by $2\pi\Delta f\Delta t$, where Δf is the frequency offset and Δt is the sampling period. Thus, the received phase differences in MSK differ from the transmitted ones by a value linearly proportional to the frequency offset.

To compensate for this effect, Alice has to shift her known phase differences by $2\pi\Delta f\Delta t$. She knows her sampling period Δt, but needs to estimate the frequency offset Δf. To do so, Alice uses the pilot sequence in the interference-free bits. The pilot sequence is chosen such that the mean of the transmitted phase differences is zero (i.e., the pilot sequence has an equal number of ones and zeros). Due to frequency offset, the mean of the phase differences in the received pilot sequence is non-zero; it is $2\pi\Delta f\Delta t$. Alice decodes the pilot sequence using standard methods which is possible because the pilot sequence is interference free. Alice computes the mean of the phase differences of the pilot sequence, and uses that value to estimate the frequency offset.

Channel Distortion: The wireless channel distorts the transmitted signal because it affects different frequency components differently. In MSK, bit transitions, such as the ”10” sequence, cause a sudden change in the phase difference, and hence a sudden change in the frequency of the sinusoid. As a result, the channel distorts the received phase differences at these transition points, reducing the sharpness of the transition.

We use the pilot sequence to estimate how the channel distorts the phase differences at transition points. We choose a pilot sequence that is rich in bit transitions of the forms “10”, “1100”, and “111000”. We experimentally observe that the distortion of any sequence can be estimated using the distortions for these three transition patterns. This is because the phase distortion for a sequence 1^n0^n is very close to that of 111000, for any $n \geq 3$. Alice observes how the channel attenuates the transition patterns in her pilot sequence and applies the same distortions to the transition patterns in her known signal.

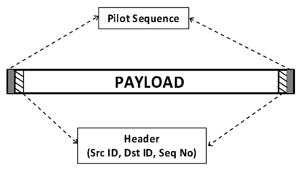


Figure 6—Frame Layout for Analog Network Coding

Sampling Offset: Due to lack of synchronization, receivers cannot sample the received signals exactly at the symbol boundaries; there is always a sampling offset. Similarly to the channel, in MSK, the sampling offset distorts the phase differences at bit transitions (e.g., “10” sequences) because the sinusoid being transmitted abruptly changes at the symbol boundary. The sampling offset causes the phase difference to be computed using values from two separate sinusoids, which results in an attenuation of the phase differences.

As before, we estimate the impact of the sampling offset by examining the distortion of the phase differences at the transition points in the pilot sequence. Sampling offset is also alleviated using oversampling (taking two samples per bit/symbol), which is a typical approach in communication systems.

Once Alice estimates all of the above distortions, she applies the same distortions to her known phase differences, and then uses the resulting known phase differences in the decoding algorithm in §6.3.[6]

7.4 Which Packet Does Alice Use to Decode?

Alice keeps copies of the sent packets in a *sent-packet buffer*. When she receives a signal that contains interference, she has to figure out which packet from the buffer she should use to decode the interfered signal. Hence, we add a header after the pilot sequence that tells Alice the source, destination and the sequence number of the packet. Using the decoded header information, Alice can pick the right packet from her buffer to decode the interfered signal and get Bob's packet.

7.5 How Does Bob Decode?

Bob's signal starts second in the interfered signal. Thus, he cannot blindly use the same decoding algorithm as Alice. Bob instead decodes the packet by running the decoding procedure backwards. More precisely, he stores the received complex samples until the end of the packet, i.e., until the energy drops to the noise level. Then he runs the algorithm starting with the last sample and going backward in time. Our packets have the header and the pilot sequence both at the beginning and at the end, as shown in Fig. 6. Bob starts from the end of the packet, decodes the header and the pilot sequence there, discovers which packet in his *sent-packet buffer* to use to cancel the interference, and decodes Alice's packet backwards, using the interference decoding algorithm.

7.6 What Does the Router Do?

In the Alice-Bob scenario, the router has to amplify the interfered signal it receives from Alice and Bob, and broadcast it. In the chain topology, however, the router, N_2, has to decode the packet itself. Thus, the router needs to make a decision about what to do with an interfered signal. The router uses the headers in the interfered signal to discover which case applies. If either of the headers corresponds to a packet it already has, it will decode the interfered signal. If none

[6] Also, decoding Bob's phase differences to the bits accounts for the distortions that affected his signal.

1 Pseudocode for the Interference Decoding Algorithm

```
Use energy detector from §7.1 to detect signal reception
if Signal detected then
  Use variance detector from §7.1 to detect interference
  if Interfered Signal then
    Decode start and end of received signal to get both headers and pilots
    Use headers to discover whether my known signal starts first or second
    Match the start of my known signal with the received signal using algorithm from §7.2
    Apply the corrections in §7.3 to my known phase differences and decode using algorithm from §6
    Collect the decoded bits and frame them into a packet and pass it to the upper layers
  else
    Decode signal using normal MSK demodulation
    Collect the decoded bits and frame them into a packet and pass it to the upper layers
  end if
end if
```

of the headers corresponds to packets it knows, it checks if the two packets comprising the interfered signal are headed in opposite directions to its neighbors. If so, it amplifies the signal and broadcasts the interfered signal. If none of the above conditions is met, it simply drops the received signal.

Finally, Alg. 1 summarizes our interference decoding algorithm.

7.7 How to get the right packets to interfere?

We want to encourage interfering transmissions from the right senders, i.e., those whose interfered signal can be correctly decoded at both destinations. To do so, we design a simple *trigger* protocol. To "trigger" simultaneous transmissions, a node adds a short trigger sequence at the end of a standard transmission. The trigger stimulates the right neighbors to try to transmit immediately after the current transmission.[7] For example, in the Alice-Bob topology, the router adds the trigger sequence to the end of its transmission, triggering both Alice and Bob to transmit. Alice and Bob respond by transmitting as soon as the transmission from the router ends. In the chain topology, node N_2 triggers nodes N_1 and N_3 to transmit simultaneously by adding the appropriate trigger sequence to the end of its transmission. Thus, the triggering mechanism encourages positive interference that we can exploit to increase network capacity.

Clearly, for a node to trigger its neighbors to interfere, it needs to know the traffic flow in its local neighborhood. We assume that this information is provided via control packets that the nodes exchange.

In our context, the "trigger" protocol provides a simplified MAC for ANC. Designing a general MAC protocol for ANC depends on the environment in which it is used. For example, cellular networks already have strict scheduling-based MAC protocols (TDMA, CDMA etc). The trigger protocol for ANC in these networks can be easily integrated into the scheduling mechanism. In contrast, 802.11 wireless mesh networks use random access. In this case, short control sequences may be used as triggers. However, customizing the MAC protocol for ANC in 802.11 or other networks is beyond the scope of this paper.

8. CAPACITY ANALYSIS

This section bounds the capacity gains of analog network coding for the Alice-Bob network. We have proved these bounds in [16]. Here, however, we summarize and discuss the bounds to illustrate the theoretical benefits of ANC, before delving into the experimental results in §10.

[7]The nodes still insert the short random delay mentioned in §7.2.

Note that the capacity of a general wireless network is an open problem in information theory. In fact, the exact capacity of a 3-node relay network, that is, a source-destination pair with a router in the middle, is itself an open problem. The standard approach is to compute upper and lower bounds on the capacity of these networks, which is what we do in this section.

We compare the capacity of the Alice-Bob network, under analog network coding and the traditional routing approach. To do so, we compute an upper bound on the capacity under traditional routing and a lower bound on the capacity under ANC. We compute our bounds for a wireless channel with additive white Gaussian noise. For simplicity, we assume the channel between Alice and the router is similar to the channel between Bob and the router, and all nodes transmit at the same power. When the router receives the interfered signal, it simply re-amplifies the signal and broadcasts it to Alice and Bob. We prove in [15] that:

THEOREM 8.1. *An upper bound on the capacity of the traditional routing approach is given by:*

$$C_{traditional} = \frac{1}{4}(log(1+2SNR) + log(1+SNR)),$$

and a lower bound on the capacity of analog network coding is given by:

$$C_{ANC} = log(1 + \frac{SNR^2}{3SNR+1}),$$

where SNR is the signal to nosie ratio at the receiver. Hence, the capacity gain of analog network coding over the traditional approach asymptotically approaches 2 *as the SNR increases.*

Fig. 7 illustrates the capacity bounds for analog network coding and the traditional approach. The figure shows two SNR regions with different characteristics.

(a) Moderate to High SNR: At medium-to-high SNR, analog network coding almost doubles the throughput when compared to the traditional routing approach. At these SNRs, the gain is primarily dominated by the reduction in the number of time slots needed to send the packets (from 4 to 2).

(b) Low SNR: In contrast, at low SNRs around 0-8dB, the throughput of analog network coding is lower than the upper bound for the traditional approach. This is because when the router amplifies and broadcasts the interfered signal to Alice and Bob, it also amplifies the noise that the channel adds to the interfered signal. At low SNR, this amplified noise has a negative effect at Alice and Bob, since the transmission power is quite low.

Note that practical wireless systems operate above 10dB and are typically around 20-40dB [9]. The low SNR region is not used because it is hard to design practical receivers that decode at such low power. For example, when SNR is about 5-10dB, 802.11 devices cannot associate with the local access point [9]. So, for most practical cases, analog network coding has a capacity gain of 2*x* for the Alice-Bob network.

9. IMPLEMENTATION

We have implemented ANC using Software Defined Radios (SDR). SDRs implement all the signal processing components (source coding, modulation, clock recovery, etc.) of a wireless communication system entirely in software. The hardware is a simple radio frequency (RF) frontend, which acts as an interface to the wireless channel. The RF frontend passes the complex samples generated by the SDR to the Digital to Analog Converter (DAC), which produces the analog signal. The upconverter converts the output of

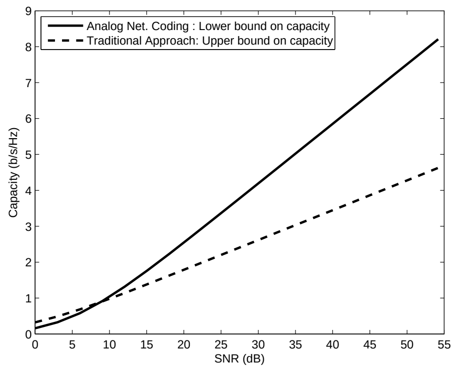


Figure 7—Capacity bounds as functions of SNR, for half-duplex nodes. At high SNRs, analog network coding doubles the throughput compared to traditional routing.

the DAC to the carrier frequency and transmits it over the wireless channel. At the receiver side, the process is inverted. First, the down-converter converts the received signal to its baseband frequency and passes it to an Analog to Digital Converter (ADC). The discrete samples produced by the ADC are converted into complex numbers and passed to the SDR software.

For SDR hardware, we use the Universal Software Radio Peripheral (USRP) [13] with RFX2400 daughterboards, which operate in the 802.11 frequency range (i.e., 2.4GHz) and have a transmit power of 50mW (17dBm)[12]. The USRP connects to the PC via USB 2.0. Thus, its throughput is limited to 32MB/s. This also limits the bandwidth of the signal to at most 4MHz, which is enough for most narrowband data transmissions.

The software for the signal processing blocks is from the open source GNURadio project [8]. We use the default GNURadio configuration,[8] which results in a bit rate of 500kb/s. We work in the SNR region of 25-35 dB, which is a typical range for 802.11 [9]. The packet consists of a 32-bit preamble, a 1500-byte payload, 32-bit CRC, 32-bit preamble, two 64-bit access codes and two headers of length 64 bits each. Except for repeating the header and the access code at the end of each packet, all are default GNURadio software configurations.

10. EXPERIMENTAL EVALUATION

This section uses results from a software radio testbed to study the performance of our approach. We run our experiments on three canonical topologies: the Alice-Bob topology in Fig. 1, the "X" topology in Fig. 10, and the chain topology in Fig. 2. These topologies form the basis for larger networks and provide examples of both 2-way and unidirectional traffic.

10.1 Compared Approaches

We compare ANC with two other approaches.

(a) No Coding (Traditional Approach): We implement traditional routing but with an optimal MAC. This means that the MAC employs an optimal scheduler and benefits from knowing the traffic pattern and the topology. Such MAC never encounters collisions or backoffs, and hence outperforms the conventional carrier sense based MAC.

(b) Digital Network Coding (COPE): We compare against packet-based network coding whenever applicable. We use the COPE protocol as an example network coding protocol [17]. Again we implement an optimal MAC that schedules transmissions knowing the traffic pattern and the topology.

[8]DAC Rate: 128e6 samples/s, Interpolation Rate: 128, 2 samples/symbol, 1 bit/symbol.

Since the MAC is optimal for all three designs, the differences between them are due to their intrinsic characteristics rather than how the MAC works.

10.2 Metrics

- *Network Throughput:* This is the sum of the end-to-end throughput of all flows in the network. Note that ANC has a higher bit-error rate than the other approaches and thus needs extra redundancy in its error-correction codes. We account for this overhead in our throughput computation.
- *Gain Over Traditional Approach:* This is the ratio of network throughput in ANC to network throughput in the traditional approach in back-to-back runs, while keeping the topology and traffic pattern constant.
- *Gain Over COPE:* This is the ratio of network throughput in ANC to network throughput in COPE in back-to-back runs, while keeping the topology and traffic pattern constant.
- *Bit-Error Rate (BER):* the percentage of erroneous bits in an ANC packet, i.e., a packet decoded using our approach.

10.3 Summary of Results

Our experiments reveal the following findings:

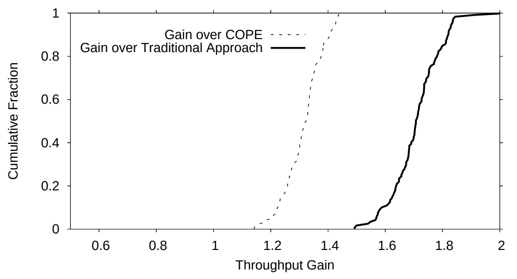
- ANC provides significant throughput gains. For the Alice-Bob topology, ANC increases network throughput by 70% compared to the traditional approach. Compared to COPE, the throughput increases by 30%.
- ANC improves the throughput of the "X" topology by 65% when compared to the traditional approach, and by 28% when compared to COPE.
- For unidirectional flows in the chain topology, ANC improves the throughput by 36% when compared to the traditional approach. (COPE does not apply to this scenario.)
- Differences between the theoretical gains of ANC and its practical gains are dominated by imperfect overlap between interfering packets, where only 80% of the two packets interfere on average.
- We evaluate ANC's sensitivity to the relative strength of the two interfering signals. On USRP software radios, our decoding algorithm works with signal to interference ratio in the range of -3dB to 0dB. Said differently, ANC works even when the two interfered signals have more or less the same power. In contrast, typical interference cancellation schemes cannot work in that range and require a signal to interference ratio of about 6dB [10].

10.4 Alice-Bob Topology

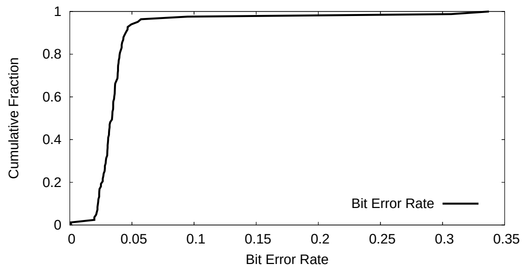
We compare ANC to both the traditional approach and COPE in the Alice-Bob topology in Fig. 1. Each run transfers 1000 packets in each direction, first using ANC, then using the traditional approach, and last using COPE. We repeat the experiment 40 times and plot the results in Fig. 8.

Fig. 8(a) plots the CDF of ANC's throughput gain over the traditional approach and COPE. The figure shows that ANC's average gain is 70% compared to the traditional approach and 30% compared to COPE.

Our practical throughput gains are significant, but less than the theoretical optimum. Theoretically, ANC doubles the throughput compared to the traditional approach and provides 50% gain over COPE. Practical gains are lower due to two reasons. First, the theoretical computation assumes that packets interfere perfectly, i.e., it assumes that Alice and Bob are perfectly synchronized. In practice, the average overlap between Alice's packets and Bob's is 80%. The

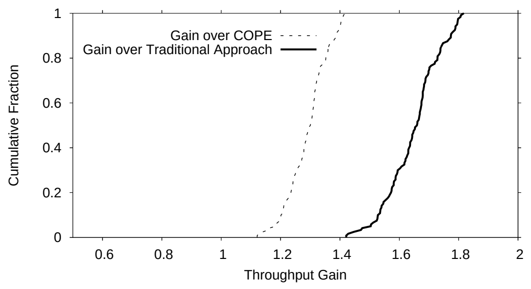


(a) CDF of throughputs for Alice-Bob topology

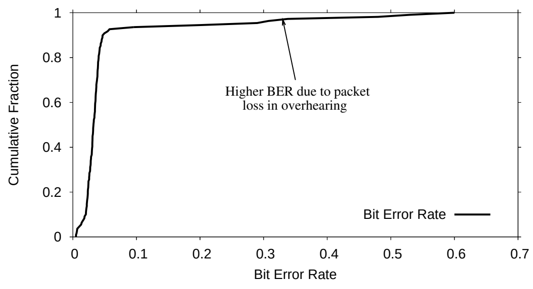


(b) CDF of BERs for Alice-Bob topology

Figure 8—Results for the Alice-Bob topology: ANC has 70% average throughput gain over the traditional approach and 30% over COPE. The average BER is around 4%, which can be easily corrected by a small amount of error correcting codes.



(a) CDF of throughputs for "X" topology



(b) CDF of BERs for "X" topology

Figure 9—Results for the "X" topology. Our approach provides an average of 65% gain over the traditional approach and 28% over traditional network coding. It is slightly less than the Alice-Bob topology due to unreliable overhearing. The BERs for the experiments is correspondingly higher.

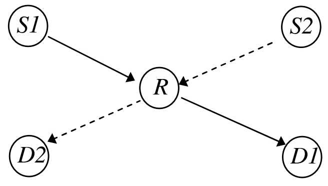


Figure 10—"X" topology: Two flows intersecting at node R.

imperfect overlap is due to the *random delay* introduced by our protocol to ensure that the pilot sequences are interference free. Further, because our implementation runs in user-space, there is significant jitter in how fast Alice and Bob transmit after receiving the "trigger" from the router. We believe that with a kernel-space implementation, one could get higher overlap in the packets and consequently higher gains.

The second factor affecting ANC's practical gains is the non-zero bit-error rate. Fig. 8(b) plots the CDF of bit-error rates for Alice and Bob, when using our approach. The bit-error rate is computed by decoding the packet from the interfered signal and then comparing it against the payload that was sent. The bit-error rate for most packets is less than 4%. To compensate for this bit-error rate we have to add 8% of extra redundancy (i.e., error correction codes) compared to the traditional approach. This overhead is another reason why the practical gains are a little lower than the theoretical gains.

10.5 "X" Topology

Next, we evaluate ANC in the "X" topology in Fig. 10. This topology is analogous to the Alice-Bob topology, but in contrast to Alice's node, which knows the interfering signal because she has generated it, the receivers in the "X" topology know the interfering signal because they happen to overhear it while snooping on the medium. In particular, S_1 and S_2 are sending to D_1 and D_2, respectively. Node D_2 can overhear S_1's transmission, and similarly D_1 can overhear S_2's transmission. We make S_1 and S_2 transmit simultaneously. The router R amplifies and transmits the interfered signal to the destinations D_1 and D_2. The destinations use the overheard packets to cancel the interference and decode the packets they want.

Fig. 9(a) plots the CDF of throughput gains for the "X" topology. The figure shows that ANC provides a 65% increase in throughput compared to the traditional approach, and a 28% increase in throughput compared to COPE.

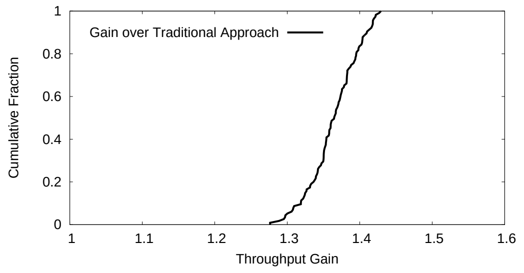
As expected, practical gains are lower than theoretical gains. Theoretically, ANC doubles the throughput when compared to the traditional approach, and increases the throughput by 50% when compared to COPE. The reasons for the difference between practical and theoretical gains are fairly similar to those in the Alice-Bob case. First, packets do not overlap perfectly. Second, the decoded packets have a non-zero BER, hence extra redundancy is required. There is, however, an additional error factor in the "X" topology, which is imperfect decoding of overheard packets. Theoretical gains assume that when S_1 transmits, D_2 overhears the packet and correctly decodes it. This is not always true and D_2 sometimes fails in decoding the overheard packets, particularly because node S_2 is transmitting too; hence node D_2's reception faces additional interference. When a packet is not overheard, the corresponding interfered signal cannot be decoded either. The same reason holds for node D_1 overhearing S_2's transmission. Hence, the throughput gain is slightly lower.

10.6 Unidirectional Traffic: Chain Topology

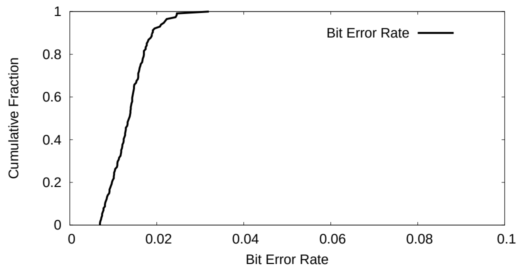
Unlike COPE, analog network coding is useful even when the flows are uni-directional. To demonstrate these gains, we evaluate our approach in the chain topology shown in Fig. 2, where traffic is flowing from node N_1 to node N_4.

Figure. 11(a) plots the CDF of the throughput gains with our approach, compared to the traditional approach. ANC increases the throughput by 37%, on average.

Note that for the chain topology, the throughput gain is close to the theoretical prediction. Theoretically, ANC has a gain of 50%, since it reduces the number of time slots required to deliver a packet on average from 3 to 2. The slight loss in gain is due to the same factors as before. Packets do not overlap perfectly and we have to provision for extra redundancy to correct for the slightly higher bit-error rate.



(a) CDF of throughputs for chain topology



(b) CDF of BERs for chain topology

Figure 11—Results for the chain topology. Our approach provides an average of 36% gain over the traditional approach. The average BER is 1.5%, which is lower than the Alice-Bob topology since here the router directly decodes the interfered signal and does not amplify and broadcast it.

Interestingly though, the bit-error rate is lower for the chain than for the other topologies. Fig. 11(b) plots the BER CDF at node N_2. The average bit-error rate is about 1.5%, which is significantly lower than the 4% bit-error observed in the Alice-Bob topology. This is because in the chain topology, decoding is done at the node that first receives the interfered signal. In the Alice-Bob case, the interference happens at the router, but the router has to then amplify and broadcast the signal to Alice and Bob. This also amplifies the noise in the interfered signal, resulting in higher bit-error rates at Alice and Bob.

10.7 Impact of Relative Signal Strengths

Typical interference cancellation schemes require that one of the two interfered signals is significantly stronger than the other. Then they can decode the strong signal, cancel it out, and decode the weaker signal. Thus, when the signal to interference ratio (SIR) is higher than 6 dB, one can apply known interference cancellation schemes [10]. It is interesting to ask whether ANC also requires such constraints to work? We evaluate this by calculating the bit-error rate for the decoded packet as the relative signal strengths vary. We focus on the region where typical interference cancelation algorithms do not work, i.e., when the signal strengths of the interfering signals are roughly the same and no single signal is significantly stronger. We consider the Alice-Bob topology, and compute the Signal to Interference Ratio (SIR),

$$SIR = 10\log_{10}(\frac{P_{Bob}}{P_{Alice}}) \tag{9}$$

where P_{Bob} and P_{Alice} are the received powers for Bob's and Alice's signals respectively at Alice.

We vary Bob's transmission power, while Alice's power is kept constant. Fig. 12 plots the BER of the decoded Bob's packet as a function of the received SIR at Alice. Even when Bob's signal strength is half that of Alice's signal, i.e., for a SIR of -3dB, the BER is less than 5%, which can be masked by error-correcting codes. When the signals are of equal strength (SIR = 0dB), the BER drops to 2%. At the other end of the spectrum, when Bob's signal is twice as strong as Alice's signal, the BER drops to 0.

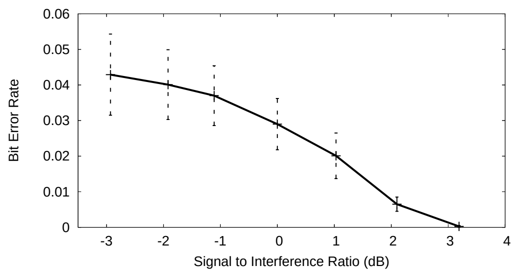


Figure 12—BER vs. Signal-to-Interference Ratio (SIR). ANC shows low BER when the relative signal strengths are more or less the same. As one signal becomes significantly weaker than the other, the BER increases at one receiver and naturally decreases at the other receiver.

Thus, ANC works well even when the relative signal strengths are more or less the same. This is because of its ability to leverage already known network level information to decode a desired signal even if it has a strength comparable or lower than the signal that is interfering with it.

Note that the SIR at Alice and the SIR at Bob have the same value but with opposite signs, as indicated in Eq. 9. As Bob's signal becomes weaker, Alice has harder time decoding it, but it becomes easier for Bob to decode Alice's signal from the interference. In Fig. 12, when the SIR at Alice decreases below -3dB, Alice experiences 5% BER decoding Bob's signal, but Bob has 0% BER decoding hers. Thus, as one of the two signals becomes increasingly weaker than the other, ANC's performance degenerates to that of the current approach, where the stronger signal captures the medium and the weaker one gets ignored.

11. DISCUSSION

This paper provides a proof of concept of analog network coding. However, more research is needed to realize the full potential of this inter-disciplinary approach. Here, we list a number of future research topics that can help further develop this technology.

(a) Modulation independent scheme: In principle, ANC is applicable independent of the modulation scheme. Though parts of this paper exploit the characteristics of MSK, we believe the approach can be extended to other modulation schemes. Conceptually, our ANC technique can be divided into two parts. First, ANC exploits the lack of complete synchronization to estimate the channel transfer function using a pilot sequence at the beginning and at the end of a packet. Then, it uses knowledge of one of the signals along with the estimated channel transfer function to decode the desired signal. The same two-step methodology can be applied to other modulation schemes. This will require estimating the channel phase and tracking it, which we did not need to do in the case of MSK.

(b) Reducing the bit-error rate: ANC achieves good BER performance using a simple decoding algorithm. But one can use a more advanced decoding algorithm to decrease the BER even further. Specifically, one could develop an iterative Viterbi decoding algorithm to find the most likely bit sequence. Such an algorithm has a higher computational complexity but can potentially reduce the BER further.

(c) More than two interfered signals: ANC has focused on decoding two signals that interfered with each other. The next question to ask is how to generalize ANC to more than two interfered signals. ANC can theoretically apply to more than two signals. We are currently working on a practical system for accomplishing this task.

(d) Towards a general MAC: In this paper, we designed a basic MAC for using ANC with toy topologies. Designing a general ANC

MAC protocol that works in a wide variety of topologies is an interesting question for future research. One potential solution is to investigate a scheduling based MAC (based on RTS/CTS) that exploits situations in which interference could be useful and avoids interference when it is harmful.

(e) Packet Size: We note that though our description assumed that packets are of the same size, this is not necessary. Similar to digital network coding [17], a smaller packet can be padded to the length of the larger packet. Also, one may decide to apply ANC only to relatively large packets because the throughput gains from coding small packets may be negligible.

12. CONCLUSION

The success of wireless networks is due to contributions from both electrical engineers and computer scientists. So far, these two groups have proceeded largely in isolation, having agreed a few decades ago that their contract would be a digital one: the electrical engineers would design components that present binary data to the computer scientists, and in return, could ignore network layer questions; while computer scientists would design the network layer and overlook physical layer details. In this paper, we question whether this divide is suitable in *every* context. In particular, we show that, for wireless networks, by poking a hole in this digital abstraction, i.e., by combining physical-layer and network-layer information we can substantially increase network capacity. We believe that, because of the substantial gains possible, this inter-disciplinary approach is worthy of further investigation.

Acknowledgments

We thank Micah Brodsky, Dawei Shen, Nate Kushman, Samuel Madden, Jayashree Subramaniam, Hariharan Rahul, Ashish Kisti, Mythili Vutukuru, our shepherd, Craig Partridge and the anonymous reviewers for their insightful comments; and Michael Walfish for his inputs on the conclusion. This work is supported by NSF award CNS-0627021, and DARPA ITMANET. The opinions and findings in this paper are those of the authors and do not necessarily reflect the views of NSF or DARPA.

13. REFERENCES

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proc. of ACM SIGCOMM 2004, Philadelphia, USA*.

[2] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network Information Flow. In *IEEE Trans. on Info. Theory, pages 1204-1216, July*, 2000.

[3] J. F. Cardoso. Blind signal separation: statistical principles. *Proceedings of the IEEE*, 86(10):2009–2025, 1998.

[4] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proc. of ACM SIGCOMM 2007, Kyoto, Japan*.

[5] Christos Gkantsidis and Pablo Rodriguez. Network Coding for Large Scale Content Distribution. In *Proc. of INFOCOM 2005, Miami, USA*.

[6] S. Deb, M. Effros, T. Ho, D. R. Karger, R. Koetter, D. S. Lun, M. Médard, and N. Ratnakar. Network coding for wireless applications: A brief tutorial. In *Proc. of IWWAN 2005, London, UK*.

[7] Definition and assessment of relay based cellular deployment concepts for future radio scenarios considering 1st protocol characteristics, Ch5. https://www.ist-winner.org/DeliverableDocuments/D3.4.pdf.

[8] G. FSF. Gnu radio - gnu fsf project. http://www.gnu.org/software/gnuradio.

[9] J. Geier. Snr cutoff recommendations, 2005. http://www.wi-fiplanet.com/tutorials/article.php/3468771.

[10] J. Hamkins. An analytic technique to separate cochannel fm signals. *IEEE Transactions on Communications*, 48(11):2980–2989, 2000.

[11] T. Ho, R. Koetter, M. Médard, D. Karger, and M. Effros. The Benefits of Coding over Routing in a Randomized Setting. In *Proc. of ISIT, 2003, Yokohoma, Japan*.

[12] E. Inc. Datasheet for the basicrx, basictx, lfrx, lftx, tvrx, and dbsrx daugtherboards. http://www.ettus.com/Download.html.

[13] E. Inc. Universal software radio peripheral. http://ettus.com.

[14] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory, Page(s):1973 - 1982, June, 2005*.

[15] S. Katti, S. Gollakota, and D. Katabi. Analog network coding. Working Draft. Available on the authors webpage.

[16] S. Katti, I. Maric, A. Goldsmith, D. Katabi, and M. Médard. Joint Relaying and Network Coding in Wireless Networks. In *Proc. of ISIT 2007, Nice, France*.

[17] S. Katti, H. Rahul, D. Katabi, W. H. M. Médard, and J. Crowcroft. XORs in the Air: Practical Wireless Network Coding. In *Proc. of ACM SIGCOMM 2006, Pisa, Italy*.

[18] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking, Volume 11, Issue 5, Oct. 2003, Page(s):782 - 795*.

[19] J. N. Laneman, D. N. C. Tse, and G. W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Trans. on Inform. Theory, Volume 50, Issue 12, Dec. 2004 Page(s):3062 - 3080*.

[20] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Transactions on Information Theory*, 2003.

[21] D. S. Lun, M. Médard, and R. Koetter. Efficient operation of wireless packet networks using network coding. In *Proc. of IWCT 2005, Oulu, Finland*.

[22] E. C. V. D. Meulen. Three-terminal communication channels. *Adv. Appl. Probab.*, 3:120–154, June 1971.

[23] J. S. Park, M. Gerla, D. S. Lun, Y. Yi, and M. Medard. Codecast: A network-coding based ad hoc multicast protocol. *IEEE Wireless Communications Magazine*, 2006.

[24] R. L. Pickholtz, L. B. Milstein, and D. L. Schilling. Spread spectrum for mobile communications. *IEEE Trans Veh. Technology*, 1991.

[25] R. Ramanathan. Challenges: A Radically New Architecture for Next Generation Mobile Ad Hoc Networks. In *ACM MOBICOM*, 2005.

[26] A. Ramamoorthy, J. Shi, and R. Wesel. On the capacity of network coding for wireless networks. In *41st Annual Allerton Conference on Communication Control and Computing*, Oct. 2003.

[27] B. Rankov and A. Wittneben. Achievable rate regions for the two-way relay channel. In *Proc. of ISIT 2006, Seattle, USA*.

[28] B. Rankov and A. Wittneben. Spectral efficient protocols for half-duplex fading relay channels. *IEEE Journal on Selected Areas in Communications*, 25, Feb. 2007.

[29] S. Zhang, S. Liew, and P. Lam. Physical layer network coding. In *Proc. of ACM MOBICOM 2006, Los Angeles, USA*.

[30] C. E. Shannon. Two-way communication channels. *4th Berkeley Symposium Math. Stat. Prob.*, 1:611–644.

[31] D. Tse and P. Vishwanath. *Fundamentals of Wireless Communications*. Cambridge University Press, 2005.

[32] S. Verdu. *Multiuser Detection*. Cambridge University Press, 1998.

[33] J. Widmer and J.-Y. L. Boudec. Network Coding for Efficient Communication in Extreme Networks. In *Proc. of SIGCOMM WDTN 2005, Philadelphia, USA*.