MIT-CSAIL-TR-2007-012 February 23, 2007

Embracing Wireless Interference: Analog Network Coding

Sachin Katti, Shyamnath Gollakota, and Dina Katabi

Embracing Wireless Interference: Analog Network Coding

Sachin Katti Shyamnath Gollakota Dina Katabi

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 codes signals not bits.

So, what if wireless routers forward signals instead of packets? Theoretically, we prove that such an approach doubles the capacity of the canonical 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.

1. INTRODUCTION

Wireless interference is considered harmful. Interference creates collisions, prevents reception, and wastes scarce bandwidth. 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 [6, 27]. With bandwidth being scarce in a wireless network, enabling concurrent receptions despite interference is essential.

This paper introduces Analog Network Coding (ANC). Instead of avoiding interference, we exploit the interference of strategically picked senders to increase network capacity. When multiple senders transmit simultaneously, the packets collide. But looking deeper at the signal level, collision of two packets means that the channel adds their physical signals after applying attenuations 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 the known packet after correcting for channel effects. The receiver is left with the signal of the packet it wants, which it decodes using standard methods. In a wireless network, packets traverse multiple hops. When 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 traditional digital network coding, senders transmit sequentially, the routers mix the content of the packets and broadcast the mixed version [17]. In analog network coding, senders transmit simultaneously. The wireless channel naturally mixes these signals. Instead of forwarding mixed packets, routers forward mixed signals.

Since it allows multiple transmissions to occur simultaneously yet still be received correctly, analog network coding increases network capacity. We show via analysis and implementation on software radios that our approach achieves higher throughput than both traditional wireless design and digital network coding.

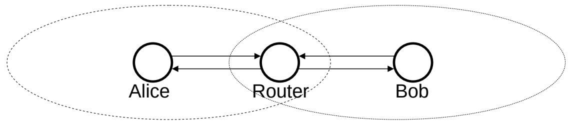
Our approach differs significantly from prior work on concurrent transmissions. Specifically, while spatial reuse allows multiple senders to transmit simultaneously if they do not interfere, our approach allows multiple transmissions despite interference. Schemes such as CDMA [24], FDMA [28], etc allow multiple senders to transmit at the same time. These schemes, however, simply divide the network's capacity among the users. In contrast, our approach increases network capacity by exploiting higher-layer information.

This is the first work to show that interfering signals can be exploited to increase network capacity and that such an approach is implementable. Our approach is independent of the underlying communication technology; thus, it can be used to increase the capacity of orthogonal techniques such as CDMA, FDMA, etc. Our contributions can be summarized as follows:

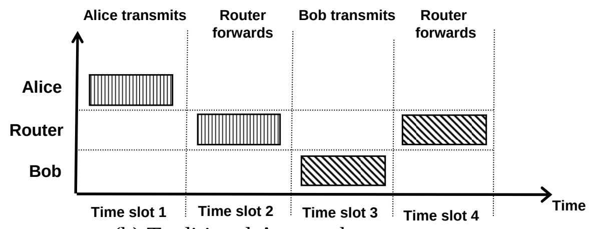
- We show how to perform analog network coding within a flow and across different flows that intersect at a router. These canonical scenarios create building blocks, common to any wireless multihop network.
- We analyze the capacity of the canonical 2-way relay network, both with and without our approach, and show that our approach doubles the capacity.
- We design and implement our approach in software radios, proving its practicality.
- We evaluate analog network coding 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%. As for the throughput, it increases by 70% in comparison with no-coding, and by 30% in comparison with traditional network coding.

2. CANONICAL EXAMPLES

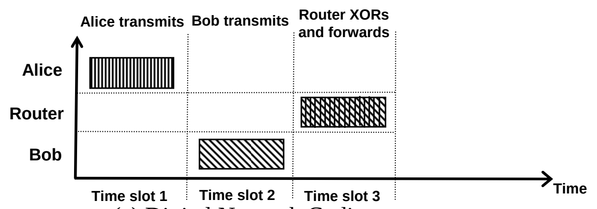
We explain our ideas using two canonical topologies, common in any mesh network. The two examples we give below constitute building blocks for larger networks.



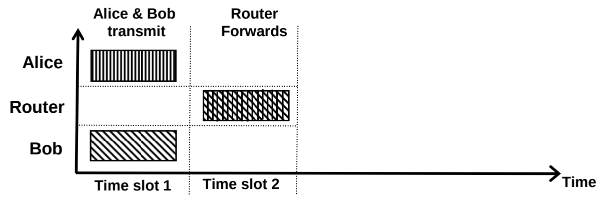
(a) Alice-Bob topology



(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.

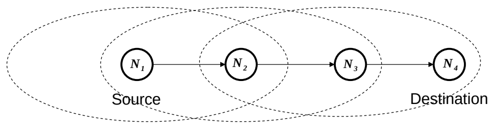
(a) Flows Intersecting at a Router: Consider the example introduced by Katti et al. in [17]. Alice and Bob want to send a message to each other. The radio range does not allow them to communicate without a router, as shown in Fig. 1(a). In the traditional approach, Alice sends her packet to the router, which forwards it to Bob, and Bob sends his packet to the router, which forwards it to Alice. Thus, to exchange two packets, the current approach needs 4 time slots. Network coding achieves the same goal, but with fewer transmissions. 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. This paper asks a simple question: 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. It is analogous to digital network coding but is done over physical signals in the wireless channel itself. As a result, we reduce the required time slots from 4 to 2, doubling the wireless throughput.

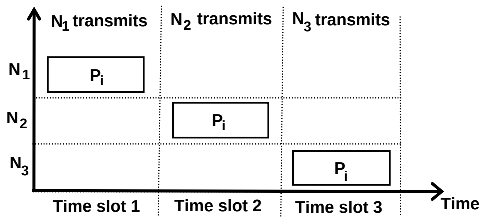
(b) Flows in a Single Direction: Analog network coding, not only increases the throughput beyond digital network coding, it also applies to new scenarios to which traditional digital network coding did not apply. Consider the chain topology in Fig. 2(a), where a single flow traverses 3 hops. The traditional 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}, whereas 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 $3/2 = 1.5$.

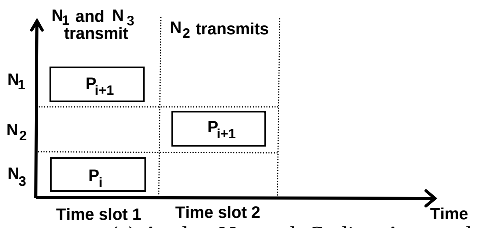
In practice, the throughput gain of the chain topology may be even higher. Without analog network coding, the nodes need an added mechanism to handle the hidden terminal problem in Fig. 2(a). They use either RTS-CTS or a statistical method like the exponential backoff built into the 802.11 MAC. Both 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. Analog network coding therefore solves the hidden terminal problem for chain topologies with both uni-directional as well as bi-directional traffic. The hidden terminal problem persists in networks with multiple interacting chains and general ad-hoc



(a) Chain topology



(b) Traditional Approach



(c) Analog Network Coding Approach

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 $3/2 = 1.5$

networks. Addressing the hidden terminal problem in this latter case 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 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. Thus, there will be a time shift between the two signals. Alice and Bob need to align the known signal with the interference in the received signal before they can cancel the interference.

Thus, to implement a proof of concept of analog network coding, we have to dive into the physical layer and adapt channel acquisition, modulation, clock recovery, and other signal processing techniques to this new environment, namely, we need to design a new communication system from the ground up.

3. RELATED WORK

Prior work falls into three main categories: schemes for addressing interference, traditional network coding, and theoretical work on network capacity.

(a) Addressing Interference: Prior work on wireless networks tries to avoid interference by probing the medium for idleness [30], scheduling senders in different time slots [28], or using small control packets called RTS-CTS [3]. Our work allows correct reception despite interference.

Multiple access techniques like CDMA [24], FDMA [28], and spatial reuse [28] allow multiple transmissions at the same time. Additionally, co-operative diversity [20], MIMO antennas [28], and cognitive radio [14] also allow multiple concurrent transmissions. These approaches, however, are simply means to avoid interference in time, code, space, or frequencies. They just divide channel capacity among multiple users. In contrast, our work expands the capacity of the network.

The work closest to ours is in the areas of blind signal separation and interference cancellation. These schemes decode two signals that have interfered without knowing any of the signals in advance [29, 4, 12]. Practical work in this domain, however, is limited to signals that differ significantly in their characteristics. They usually assume that the wanted signal has much higher power than the signal they are trying to cancel out. Our technique makes no such assumptions; it works even when the wanted signal has lower power than the interfering signal. Further, the prior work does not increase network capacity. Our work increases the capacity of the network due to its ability to exploit network layer information.

(b) Traditional Network Coding: Work on network coding started with a paper by Ahlswede et al. which 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 [21, 18, 16], randomized coding [13], wireless network coding [7, 25, 17, 31, 23], 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. We further show that our approach achieves higher throughput than traditional network coding and applies to scenarios that cannot benefit from traditional network coding.

(c) Theoretical Work on Network Capacity: The capacity of a general wireless network is an open problem in information theory. In fact, the capacity of a 3-node relay network, that is, a source-destination pair with a router in the middle, is itself an open problem. There is, however, considerable work that bounds the capacity or improves the throughput of the 3-node-relay [20, 6, 19] and the Alice-Bob topology [27, 26, 22]. The above work is hard to apply in practice because it assumes that wireless radios can send and receive at the same time, or that the transmissions are fully synchronized. Furthermore, none of the above work has been implemented. In contrast, our work makes no such assumptions and is proven practical via a prototype implementation in software radios.

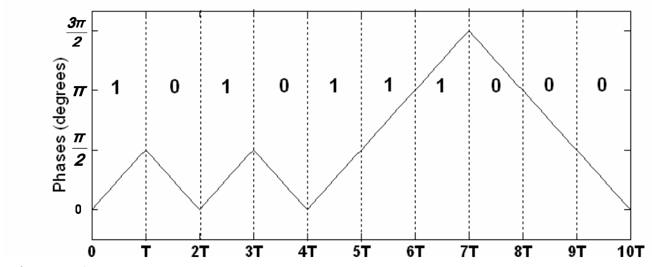


Figure 3—Example MSK modulation. MSK represents a bit of 1 as a phase difference of $\pi/2$ over for an interval T. It represents a a bit of 0 as a phase difference of $-\pi/2$ over T.

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 and underwater acoustic wireless networks [8] are also possible examples. In particular, 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, which is exactly the functionality they need to implement analog network coding [9].

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 want a modulation scheme that is widely used in many wireless technologies because it is infeasible to try the myriad of possible modulation schemes.

To this end, we choose phase shift keying (PSK), which is widely used in modern communication systems. For example, 802.11 uses Binary and Quadrature Phase Shift Keying (BPSK/QPSK) and GSM, a widely used cell-phone standard, uses a variant of Minimum Shift Keying (MSK), which is another form of phase shift keying. Our implementation uses MSK. MSK has very good bit-error properties, has a simple demodulation algorithm and excellent spectral efficiency. But the ideas we develop in this paper, especially §6.1, are applicable to any phase shift keying modulation.

5. SINGLE SIGNAL CASE

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 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 Wireless Communication Systems

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

$$s(t) = A_s(t)e^{i\theta_s(t)},$$

where $A_s(t)$ is the amplitude of the waveform and $\theta_s(t)$ is its phase. To transmit a stream of bits, we needs to map "0" and "1" to two different complex functions. Then, we divide time into consecutive slots of duration T. During each slot, we transmit the complex function corresponding to "1" or "0", depending on the bit value we want to transmit.

Although the transmitted signal is a continuous function, modern communication systems are digital. They produce discrete samples of the continuous signal. The wireless transmitter interpolates the samples to generate a continuous signal, which it transmits over the wireless channel. Thus, for the rest of this paper, we will talk about complex samples, of the form $A_s[n]e^{i\theta_s[n]}$.

5.2 The Sender Side

Say that we have a packet to transmit over the wireless channel. As said above, 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, then the phase of the signal would 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_se^{i0}. Since the first bit is a "1", the signal sample at time $t = T$ should be $A_se^{i(\pi/2)}$. The second bit is a "0", hence the signal sample at time $t = 2T$ should be $A_se^{i(\pi/2-\pi/2)} = A_se^{i0}$. 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.

5.3 The Receiver Side

How 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 T. But these samples differ 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. The receiver needs to map the received complex samples back into a bit stream.

Demodulation is the process of mapping the received signal to the transmitted bits. 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 $h\,A_s\,e^{i(\theta_s[n+1]+\gamma)}$ and $h\,A_s\,e^{i(\theta_s[n]+\gamma)}$. First, we calculate the ratio of these complex numbers,

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

To demodulate, we simply compute the angle of the complex number r, which gives us the phase difference, i.e., $\arg(r) = \theta_s[n+1] - \theta_s[n]$, where $\arg(x)$ is the angle of the complex number x. We map these phase differences to "0" and "1" bits 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. Phase modulation schemes like MSK are therefore very attractive and are widely used in cellular communications and other networks.

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, $y_A(t) + y_B(t)$. However, $y_A(t)$ and $y_B(t)$ 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 [28]. 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; it is not the index of the bit transmitted by Alice or Bob.[1].

At first, it seems that to decode the interfered signals, Alice should 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.

In practice, however, this subtraction method does not work. It is fragile and depends on the errors in Alice's estimate of the channel parameters. Though we tend to think of those parameters as constant, they do vary with time. Further, the channel model is approximate. There are other sources of noise that add up to the estimation errors.

We need a more robust method. Indeed, the main reason for the robustness of MSK is that demodulation does not require estimating the channel. Specifically, Eq. 1 computes the phase difference without worrying about the exact values of γ and h. This gives us a hint of how to design a more robust demodulation scheme for interfered signals. In particular, one should focus on discovering the phase differences for the two signals, namely $\Delta\theta$ and $\Delta\phi$. 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 will 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 $\phi[n] = \phi_s[n] + \gamma''$.

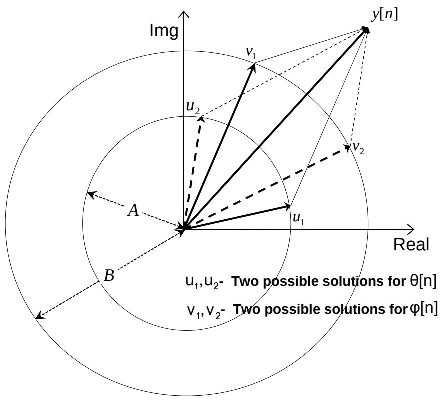
How do you calculate phase differences when two signals interfere and you know the phase differences of one of the signals? We will use a two-step process. First, Alice uses her received signal to calculate pairs $(\Delta\theta, \Delta\phi)$ that could have produced the observed signal. Next, Alice uses 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 the appendix, we prove the following lemma.

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 val-*

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



Img
y[n]
v_1
u_2
v_2
A
u_1
Real
B
u_1,u_2- Two possible solutions for θ[n]
v_1,v_2- 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 $(\theta[n], \phi[n])$.

ues.

$$\theta[n] = \arg(y[n](A + BD \pm iB\sqrt{1 - D^2})) \quad (3)$$

$$\phi[n] = \arg(y[n](B + AD \mp iA\sqrt{1 - D^2})) \quad (4)$$

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 $\phi[n]$. Thus, 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 detailed proof of Lemma 6.1 is in the appendix, but 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 again with the same sequence at the receiver to get the original bits. Hence,

$$E[|y[n]|^2] = A^2 + B^2.$$

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

$$\mu = \frac{1}{N}\sum_{n=1}^{N} |y[n]|^2 = A^2 + B^2. \quad (5)$$

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. We show in the appendix that σ can be reduced to,

$$\sigma = A^2 + B^2 + 4AB/\pi. \quad (6)$$

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

6.3 Estimating Phase Differences for Bob's Signal

Alice's next step is to estimate the phase differences of Bob's signal, i.e., $\phi[n + 1] - \phi[n]$. She 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:

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

Next, 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.

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

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 phases that Alice estimates do not match exactly the phases 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 transmission? This is a standard problem in communication systems. To detect a transmission, Alice looks at the energy in the received signal. During transmission 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 [2]. 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.

We calculate energy and energy variance over moving windows of received samples. Our detection algorithm declares occurrence of a packet if the energy is greater than $20dB$, which is a typical threshold [11]. It detects interference if the variance in the energy is greater than $20dB$. This threshold is picked because when two MSK signals interfere, the energy of the interfered signal varies from $(A+B)^2$ to $(A-B)^2$, depending on whether they interfere constructively or destructively. Thus, the variance is on the order of $((A+B)^2 - (A-B)^2)^2 = 16A^2B^2$, which is greater than the energy of either of the signals constituting the interfered signal (i.e., greater than A^2 and B^2).

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

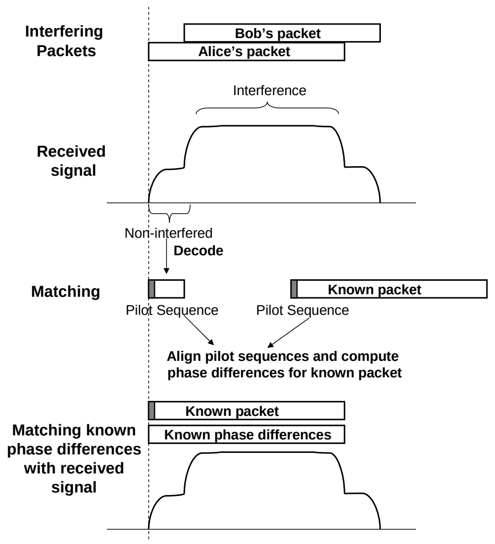


Figure 5—Aligning known phase differences with received signal. Alice finds where her packet starts using the pilot bits at the beginning of the packet, which are interference free. Bob, whose packet starts second, uses the pilot bits at the end of the packet and runs the alignment process backward.

7.2 How Does Alice Deal 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, 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. But without synchronization, 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 parts 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. 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 the slot is dependent on the transmission rate, packet size, modulation scheme used, 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 a 64-bit pseudo-random sequence. It is used to detect when ex-

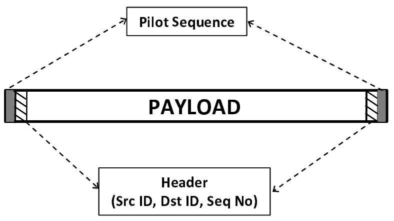


Figure 6—Frame Layout for Analog Network Coding

actly the known signal starts showing in the received signal.

We describe our solution assuming Alice's packet starts first. Bob's decoding algorithm is described in Sec. §7.4. 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.

At the end of the pilot sequence, Alice starts applying the algorithm in §6 which 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 How does Alice know which packet to 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.

7.4 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 backward. 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

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
        Discover whether my known signal starts first or second using the headers
        Lookup known packet from the headers
        Match phase differences of known signal with received signal using algorithm from §7.2
        Decode packet using algorithm from §6
        Collect the decoded bits and frame it into a packet and pass it to the upper layers
    else
        Decode signal using normal MSK demodulation
        Collect the decoded bits and frame it into a packet and pass it to the upper layers
    end if
end if
```

with the last sample and going backward in time. Our packets have the header and the pilot sequence both at the beginning and 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.

7.5 What does the router do?

In the Alice-Bob experiment, the router has to amplify the interfered signal it receives from Alice and Bob, and broadcast it. But in the chain topology, 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 decodes the interfered signal. If none of the headers correspond 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 the interference decoding algorithm.

7.6 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 reception of the trigger.[3] 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 re-

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

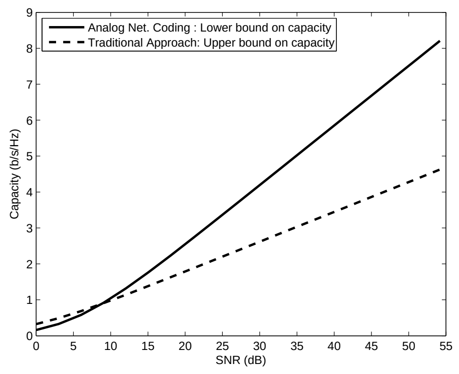


Figure 7—Capacity bounds as functions of SNR, for half-duplex nodes. At high SNRs, analog network coding doubles the throughput compared to traditional routing.

spond by transmitting as soon as the transmission from the router ends. In the chain topology in Fig. 2, 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 known 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

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 analyze the capacity of the Alice-Bob network, shown in Fig. 1. The information theory literature refers to this network as a 2-way relay channel. Many researchers, including Shannon [27], have studied this network [22, 26]. They typically assume a full-duplex relay, that is, a radio that sends and receives at the same time. In contrast, we focus on the practical case, where radios are half-duplex.

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 the traditional routing approach, and a lower bound on the capacity with our approach. Channel capacity depends on the received signal strength in comparison to the noise power at the receiver–i.e., a function of Signal-to-Noise Ratio (SNR). 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 the following theorem in the appendix.

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

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

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

$$C_{analog\ netcode} = 4\alpha log(1+\frac{SNR^2}{3SNR+1}),$$

where α is a constant. Thus, 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 deleterious effect at Alice and Bob, since the transmission power is quite low.

Note, however, that practical wireless systems typically operate around 20-40dB [11]. The low SNR region is not used because it is hard to design practical receivers that decode at such low power. For example, WLANs operate at SNR around 25-40dB. When SNR is about 5-10dB, 802.11 devices cannot associate with the local access point [11]. 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

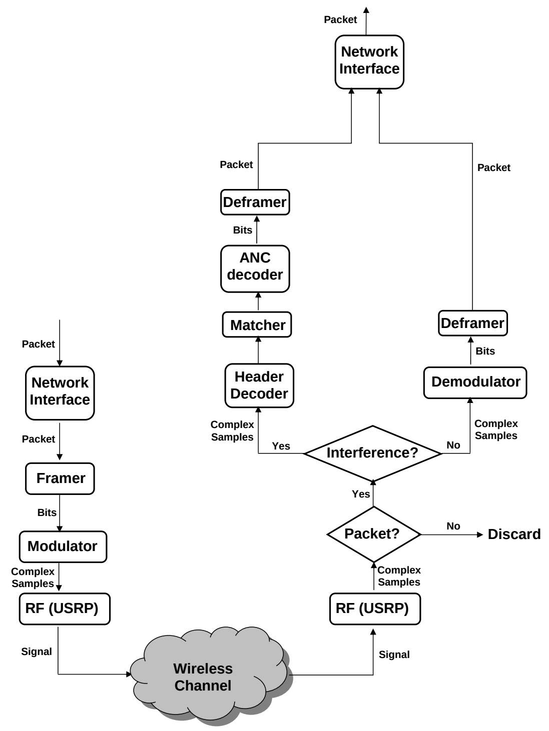


Figure 8—**Flow chart of our implementation.**

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 the DAC to the carrier frequency and transmits it over the wireless channel. At the receiver side, the process is inverted. First, the downconverter 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.

We use the Universal Software Radio Peripheral (USRP) [15] as our RF frontend. The software for the signal processing blocks is from the open source GNURadio project [10]. USRP is a generic RF frontend developed specifically for the GNURadio SDR. 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 transmission.

10. SYSTEM ARCHITECTURE

We export a network interface to the user, which can be treated like any other network device (e.g., eth0). Fig. 8 abstracts the system below the network interface.

On the sending side, the network interface pushes the packets to the *Framer*, which adds the pilot sequence and the header to the packet as described in §7.2 and §7.4, and creates a frame. The framer also stores a copy of the frame, which could be used later for decoding interfered packets. Next, the modulator encodes the bit sequence to create complex samples of the signal. It pushes the samples to the USRP RF frontend, which transmits them on the channel.

On the receiving side, we continuously get complex samples from the USRP. The *Packet Detector* checks whether the received samples constitute a packet or just noise. If a packet is being received, the *Interference Detector* checks whether the packet has been subjected to interference using the interference detection algorithm described in §7.1. If no interference is detected, standard MSK demodulation is performed to decode the bits. The bits are then passed through the *Deframer*, which converts them into a packet and forwards the packet to the network interface.

If interference is detected, the received complex samples are passed to the *Header Decoder* which detects the pilot sequences and the headers at the start and end of the set of complex samples constituting the interfered packet. From the headers it discovers which two packets constitute the interfered packet and checks if it can be decoded or should be re-amplified and broadcast. If it can be decoded, the complex samples are passed on to the next module, the *Phase Difference Matcher* which looks up the known packet, and matches the phase differences of the known packet with the received interfered signal. Once the matching is done, the complex samples are passed through the *ANC Decoder*, which decodes the unknown bits out of the interfered signal. The bits are then passed through the *Deframer*, which converts them into a packet and pushes the recovered packet out on the network interface.

11. 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. 11, 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.

11.1 Compared Approaches

We compare ANC against two other approaches.

(a) No Coding (Traditional Approach): We implement traditional routing but with an optimal MAC, i.e., the MAC employs an optimal scheduler and benefits from knowing the traffic pattern and the topology. Thus, the 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.

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

11.2 Metrics

We use the following 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 the network's throughput in ANC to the network's throughput in the traditional approach for two consecutive runs in the same topology and for the same traffic pattern.
- *Gain Over COPE:* This is the ratio of the network's throughput in ANC to the network's throughput in COPE for two consecutive runs in the same topology and for the same traffic pattern.
- *Bit Error Rate (BER):* the percentage of erroneous bits in an ANC packet, i.e., a packet decoded using our approach.

11.3 Summary of Results

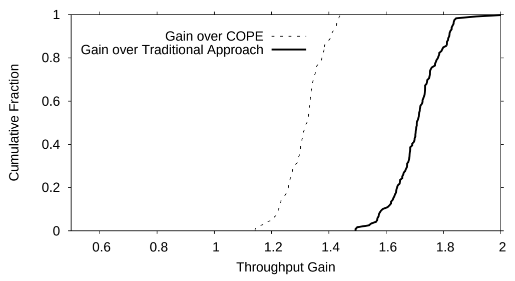
Our experiments reveal the following findings:

- ANC provides significant throughput gains. For the Alice-Bob topology, ANC increases the network's throughput by 70% compared to the traditional approach. Compared to network coding the throughput increases by 30%.
- ANC improves the throughput for the "X" topology by 65% when compared to the traditional approach, and 28% when compared to COPE.
- For unidirectional flows in the chain topology, ANC improves 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 URSP software radios, our decoding algorithm works with signal to interference ratio as low as $-3dB$. In contrast, typical interference cancellation schemes require a signal to interference ratio of $6dB$ [12]. (Note that these schemes do not use ANC and cannot achieve our capacity gains, as explained in §3.)

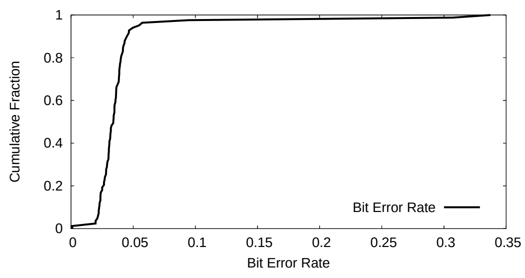
11.4 Alice-Bob topology

We compare ANC to both the traditional approach and COPE over 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. 9.

Fig. 9(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.



(a) CDF of throughputs for Alice-Bob topology



(b) CDF of BERs for Alice-Bob topology

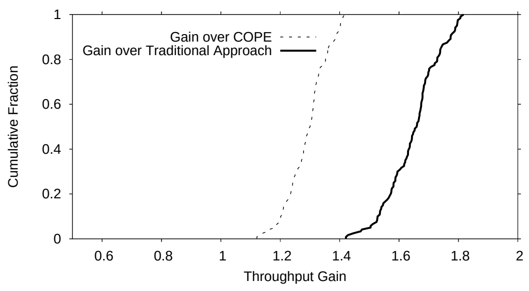
Figure 9—Results for the Alice-Bob topology: ANC has 70% average throughput gain over the traditional approach and 30% over COPE. The average BER is around 4%, which can be easily corrected by a small amount of error correcting codes.

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 those from Bob's is 80%. The imperfect overlap is due to the *random delay* our protocol introduces so 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.

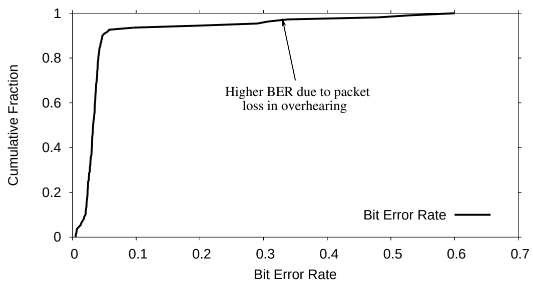
The second factor affecting ANC's practical gains is the non-zero bit error rate. Fig. 9(b) plots the CDF of bit error rates for Alice and Bob, when using our approach. The bit error rate is computed by decoding the packet from the interfered signal and then comparing it against the payload that was 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.

11.5 "X" Topology

Next, we evaluate ANC over the "X" topology in Fig. 11. This topology is analogous to the Alice-Bob, but in contrast to Alice which knows the interfering signal because she has



(a) CDF of throughputs for "X" topology



(b) CDF of BERs for "X" topology

Figure 10—**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 packet losses in overhearing. The BERs for the experiments where there were packet losses in overhearing is correspondingly higher.

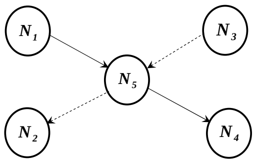
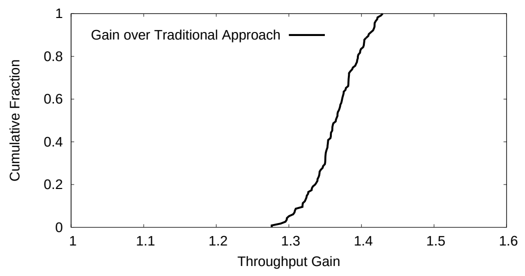
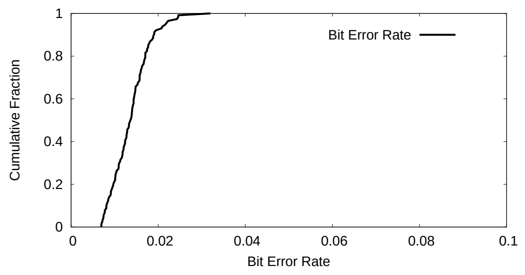


Figure 11—**"X" topology with two flows intersecting at node N_5.**



(a) CDF of throughputs for chain topology



(b) CDF of BERs for chain topology

Figure 12—**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.

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, N_1 and N_3 are sending to N_4 and N_2, respectively. Node N_2 can overhear N_1's transmission, and similarly N_4 can overhear N_3's transmission. Thus, we make N_1 and N_3 transmit simultaneously. The router N_5 amplifies and retransmit the interfered signal to the destinations N_2 and N_4. The destinations use the overheard packets to cancel the interference and decode the packets they want.

Fig. 10(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 increase the throughput by 50% when compared to COPE. The reasons for the difference between practical and theoretical gains are fairly similar to 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, namely, imperfect decoding of overheard packets. Theoretical gains assume that when N_1 transmits, N_2 overhears the packet and correctly decodes it. This is not always true and N_2 sometimes fails in decoding the overheard packets, particularly because node N_3 is transmitting too; hence nodes N_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 N_4 overhearing N_3's transmission. Hence, the throughput gain is slightly lower.

11.6 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. 12(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. But interestingly, the bit error rate is lower for the chain than for the other topologies. Fig. 12(b) plots the BER CDF at node N_2. The average bit error rate is 1%, which is significantly lower than the 4% bit error observed in the Alice-Bob topology. This is because in the chain, 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

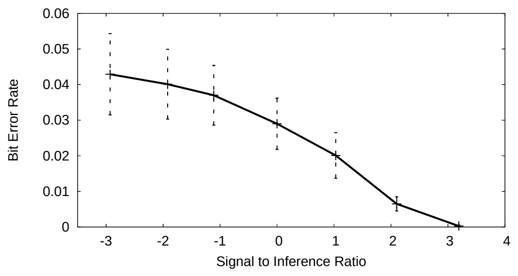


Figure 13—BER vs. Signal-to-Interference Ratio (SIR) for decoding at Alice. Even for low SIR, i.e., when the signal Alice wants to decode has relatively low signal strength, the BER is less than 5%.

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.

11.7 Impact of relative signal strengths

Is ANC's decoding algorithm sensitive to the relative signal strengths of the interfering signals? For example, does Alice's ability to decode Bob's packet depend on the relative received strengths of Alice's and Bob's signals? We evaluate this by calculating the bit error rate for the decoded packet as the relative signal strengths vary.

In order to quantify the relative signal strengths of Alice and Bob's signals we define Signal to Interference Ratio (SIR),

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

where P_{Bob} and P_{Alice} are the received powers for Bob's and Alice's signals respectively at Alice. The intuition behind this definition is simple, since Alice wishes to decode Bob's packet, her own signal which is mixed up with Bob's signal is treated as interference.

We vary Bob' transmission power, while Alice's power is kept constant. Fig. 13 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%. 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.

Hence, our approach works very well even when the relative signal strengths are vastly different or the same. Prior work on blind signal separation usually works only when the signal being decoded has a SIR of 6dB [12], i.e., the signal being decoded is four times stronger than the signal interfering with it. On the other hand our ability to leverage already known network level information allows us to decode signals which have very low signal strength compared to the signal which is interfering with it.

12. CONCLUSION

The success of wireless networks is due to contributions from both electrical engineers and computer scientists. So far, however, 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.

13. REFERENCES

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

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

[3] V. Bharghavan, A. J. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LAN's. In *ACM SIGCOMM 1994*.

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

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

[6] T. Cover and A. E. Gamal. Capacity theorems for the relay channel. *IEEE Trans. Inf. Theory*, 25(5):572–584, Sept. 1979.

[7] 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 *IWWAN*, 2005.

[8] J. G. P. E. M. Sozer, M. Stojanovic. Underwater acoustic networks. *IEEE Journal of Oceanic Engineering*, 25(1), Jan. 2000.

[9] Chapter 5. https://www.ist-winner.org/DeliverableDocuments/D3.4.pdf.

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

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

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

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

[14] J. M. III and J. G. Q. Maguire. Cognitive radio: Making software radios more personal. *IEEE Personal Communications*, 1999.

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

[16] 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*, 2003.

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

[18] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking, 2003.*

[19] G. Kramer, M. Gastpar, and P. Gupta. Cooperative strategies and capacity theorems for relay networks. *IEEE Trans. Inf. Theory*, 51(9):3037–3063, Sept. 2005.

[20] J. N. Laneman, D. N. C. Tse, and G. W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Trans. Inform. Theory, Dec. 2004*.

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

[22] E. C. V. D. Meulen. Three-terminal communication channels. *Adv. Appl. Prob.*, 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] L. B. M. R. L. Pickholtz and D. L. Schilling. Spread spectrum for mobile communications. *IEEE Trans Veh. Technology*, 1991.

[25] 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.

[26] S. Zhang, S. Liew, and P. Lam. Physical layer network coding. In *ACM MOBICOM*, 2006.

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

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

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

[30] I. . WG. Wireless lan medium access control (mac) and physical layer (phy) specifications. *Standard Specification,IEEE, 1999.*

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

APPENDIX

A. Proof of Lemma 6.1

Since,

$$y[n] = Ae^{i\theta[n]} + Be^{i\phi[n]} \tag{10}$$

the square of the magnitude of $y[n]$ is given by,

$$|y[n]|^2 = A^2 + B^2 + 2AB\cos(\theta[n] - \phi[n]) \tag{11}$$

Let us denote $\cos(\theta[n] - \phi[n])$ by D. Hence, D can be computed as,

$$D = \cos(\theta[n] - \phi[n]) = \frac{|Y[n]|^2 - A^2 - B^2}{2AB} \tag{12}$$

We use D, to separate out the two phases $\theta[n]$ and $\phi[n]$ into separate independent expressions. For brevity, we only show the computation for $\theta[n]$, the analysis for $\phi[n]$ is similar.

From Eq. 10, we can rewrite $e^{i\theta[n]}$ as

$$e^{i\theta[n]} = \frac{y[n](A + Be^{-i(\phi[n]-\theta[n])})}{A^2 + B^2 + 2AB\cos(\phi[n] - \theta[n])}$$

The complex number's phase is $\theta[n]$, which is the quantity of interest. Hence all we have to do is compute the phase of this complex number,

$$\begin{aligned} \theta[n] &= \arg(y[n](A + Be^{-i(\phi[n]-\theta[n])})) \\ \theta[n] &= \arg(y[n](A + B\cos(\phi[n] - \theta[n]) \\ &\quad -iBsin(\phi[n] - \theta[n]))) \end{aligned} \tag{13}$$

where $\arg(x)$ represents the phase of the complex number x.

Notice that we already know $\cos(\phi[n] - \theta[n]) = \cos(\theta[n] - \phi[n]) = D$. We can use this to compute $\sin(\phi[n] - \theta[n])$.

$$\sin(\phi[n] - \theta[n]) = \mp\sqrt{1 - D^2} \tag{14}$$

Substituting in Eqn. 13 we get two solutions for $\theta[n]$,

$$\theta[n] = \arg(y[n](A + BD \pm iB\sqrt{1 - D^2})) \tag{15}$$

Similarly we can compute $\phi[n]$ as,

$$\phi[n] = \arg(y[n](B + A\cos(\phi[n] - \theta[n]) + iAsin(\phi[n] - \theta[n]))) \tag{16}$$

Thus, we get two corresponding solutions for $\phi[n]$ as well,

$$\phi[n] = \arg(y[n](B + AD \mp iA\sqrt{1 - D^2})) \tag{17}$$

B. Proof of Eq. 6

From our definition,

$$\sigma = \frac{2}{N} \sum_{|y[n]|^2 > \mu} |y[n]|^2. \tag{18}$$

where μ is $A^2 + B^2$. Essentially we are calculating the expectation of those $|y[n]|^2$s which are greater than $A^2 + B^2$. From Eq. 11 we have,

$$|y[n]|^2 = A^2 + B^2 + 2ABcos(\theta[n] - \phi[n]) \tag{19}$$

Thus, in the computation of σ we are using only those $|y[n]|^2$ which have $cos(\theta[n] - \phi[n])$ greater than zero. Hence, we can rewrite σ as

$$\begin{aligned} &E(|y[n]|^2 \,|cos(\theta[n] - \phi[n]) > 0) \\ &= E(A^2 + B^2 + 2ABcos(\theta[n] - \phi[n]) \,|cos(\theta[n] - \phi[n]) > 0) \\ &= A^2 + B^2 + 2AB\, E(cos(\theta[n] - \phi[n]) \,|cos(\theta[n] - \phi[n]) > 0) \end{aligned}$$

Assuming that we are sending random bit patterns, we can derive $E(cos(\theta[n] - \phi[n])|cos(\theta[n] - \phi[n]) > 0)$ as $2/\pi$ by taking the average of a cosine over its positive lobes. Thus, finally we get

$$\sigma = A^2 + B^2 + 4AB/\pi \tag{20}$$

C. Proof of Theorem. 8.1

Let $X_A(1), \ldots, X_A(i), \ldots, X_A(m)$ and $X_B(1), \ldots, X_B(i), \ldots, X_B(m)$ be Alice and Bob's complex symbols. We assume a line of sight channel which attenuates the signal and the presence of white noise at the receiver. We also assume that all nodes transmit with the same power P.

(1.) Routing: The Outer Bound

We first compute the outer bound for traditional routing. We assume that the network is time-shared fairly between Alice and Bob's flows. We analyze the capacity of the relay network using the cutset bound [6]. We assume that transmissions are asynchronous but the channel gains are known. Then, the capacity from Alice to Bob is upper-bounded by $min\{C_1, C_2\}$, where C_1 and C_2 are given by:

$$\begin{aligned} C_1 &= \arg \max_{\rho, 0 \le \rho < 1} \tfrac{1}{4}\log(1 + (h_{AB}^2 + h_{RA}^2)P) + \tfrac{1}{4}\log(1 + (1 - \rho^2)h_{AB}^2 P) \\ C_2 &= \arg \max_{\rho, 0 \le \rho < 1} \tfrac{1}{4}\log(1 + (h_{AB}^2 + h_{RB}^2)P) + 2\rho P\sqrt{h_{AB}^2 h_{RB}^2}) + \tfrac{1}{4}\log(1 + h_{AB}^2 P) \end{aligned} \tag{21}$$

Similar equations exist for Bob, with h_{RA} and h_{RB} interchanged.

(2.) Analog Network Coding: The Inner Bound

We compare the outer bound for traditional routing with an achievable inner bound for ANC. In this case the signal received at the relay can be written as,

$$Y_R[n] = \sqrt{2P}h_{AR}X_A[n] + \sqrt{2P}h_{BR}X_B[n] + Z_R[n] \tag{22}$$

where h_{AR} and h_{BR} are the attenuations on the links Alice-Relay and Bob-Relay respectively. P is the transmission power and Z_R is the noise at the relay.

The relay amplifies the signal and broadcasts it to Alice and Bob. Let the amplification factor be A. The signal received at Alice is:

$$\begin{aligned} Y_A[n] &= Ah_{RA}Y_R[n] + Z_A[n] \\ &= A(\sqrt{2P}h_{RA}(h_{AR}X_A[n] + h_{BR}X_B[n]) + h_{RA}Z_R[n]) + Z_A[n] \end{aligned} \tag{23}$$

where Z_A is the noise at Alice, and h_{RA} is the attenuation on the link from the relay to Alice. The amplification factor A is set such that the power is still equal to P. Therefore $A = \sqrt{P/(Ph_{AR}^2 + Ph_{BR}^2 + 1)}$

Assuming Alice perfectly knows the attenuations h_{RA} and h_{AR}, she can cancel her signal out and get:

$$Y_A^{'}[n] = A\sqrt{2P}h_{RA}h_{BR}X_B[n] + Ah_{RA}Z_R[n] + Z_A[n] \tag{24}$$

For simplicity, we assume that all noise powers are the same and equal to 1. Thus, the SNRs of the received signal at Alice and Bob can be computed as:

$$SNR_{Alice} = \frac{A^2Ph_{RA}^2h_{BR}^2}{(A^2h_{RA}^2+1)}, \quad SNR_{Bob} = \frac{A^2Ph_{RB}^2h_{AR}^2}{(A^2h_{RB}^2+1)} \tag{25}$$

Thus the total throughput of the system is given by,

$$C_{anc} = \frac{1}{2}(\log(1 + SNR_{Alice}) + \log(1 + SNR_{Bob})) \tag{26}$$

The ratio C_{anc}/C_r, where C_r is the routing throughput therefore tends to 2 as $P \longrightarrow \infty$ since the ratio $\frac{\log(1+x)}{\log(1+kx)} \longrightarrow 1$ as $x \longrightarrow \infty$.