Superposition Coding for Wireless Mesh Networks

Paper ID: 1569031665

ABSTRACT
A major barrier for the adoption of wireless mesh networks is severe limits on throughput. In this paper, we design a new wireless mesh network architecture based on superposition coding to substantially improve network capacity in large, dense wireless mesh networks. Superposition coding is a physical layer technique that allows a transmitter to simultaneously send independent packets to multiple receivers. While superposition coding has been studied extensively by the physical layer community, we present the first design of practical and effective MAC and routing protocols to take advantage of superposition coding in wireless mesh networks. Extensive evaluations show that superposition coding can be a practical method to increase the throughput of large, dense wireless mesh networks. Specifically, in a mesh network with 2 to 64 active receivers and one gateway, we show that our system can increase throughput up to 283%, with average gain ranging from 15% to 51%. When there are multiple gateways, our system gains up to 190%, with average gain ranging from 31% to 71%. These results clearly demonstrate the potential benefits of our system. We also present a prototype implementation using GNU Radio.

1. INTRODUCTION
Wireless mesh networks are becoming a major paradigm for constructing user access networks that provide community or city-wide Internet connectivity [4]. However, recent theoretical analysis (e.g., [19]) and experimental measurements (e.g., [23, 17]) have shown that the current wireless mesh networks are severely limited in throughput and do not scale as they become large and dense.

In this paper, we conduct the first study and design of wireless mesh networks that use superposition coding to substantially improve network capacity. Unlike previous studies, which focus mainly on physical layer issues of applying superposition coding, our study proposes simple, practical, and effective MAC and routing protocols to take advantage of superposition coding.

Specifically, superposition coding is a physical layer technique by which a transmitter can simultaneously send independent messages to multiple receivers with asymmetric channel conditions. In this paper, for simplicity, we restrict ourselves to the two-receiver case, in which the transmitter superimposes an additional message destined to a secondary receiver on a basic message destined for a primary receiver. The additional message appears as noise to the primary receiver; the decoder at the secondary receiver extracts the additional message by applying successive interference cancellation (SIC) to first remove the interference caused by the basic message. In contrast, receivers using traditional orthogonal schemes (e.g., time, frequency, or code division) do not use SIC and thus cannot effectively extract the additional message [28]. Extensive theoretical analysis has shown that superposition coding and SIC decoding are more efficient than the orthogonal division physical layer techniques used in current wireless mesh networks. For example, in [8], Bergmans has shown that superposition coding achieves optimal capacity for an additive white Gaussian noise (AWGN) physical channel, while the traditional schemes can be far away from reaching capacity.

To appreciate the potential benefits of superposition coding, consider typical 802.11 wireless mesh networks. Recent measurement studies (e.g., [2, 5, 6, 22]) have shown that nodes in a wireless network may be distributed unevenly and the channel quality of the links in such mesh networks varies widely. Such rate diversity can drastically reduce network throughput using standard 802.11. However, since superposition coding benefits from channel diversity, this presents a setting where it can be particularly effective. Consider a simple scenario of one 802.11g-like transmitter and two receivers. The channel quality to receiver 1 is low and can support only 6 Mbps; the channel quality to receiver 2 is high and can support 54 Mbps. Without superposition coding, the transmitter using FIFO scheduling may alternate transmissions to the two receivers. Thus, the transmitter spends $\frac{54}{54 + 0.6} = 90\%$ of the time transmitting to receiver 1 and 10% to receiver 2. The total throughput is only $10.8(=6 \times 0.9 + 54 \times 0.1)$ Mbps. One way to improve total throughput and still maintain fairness is to use a different scheduler such as proportional fairness. However, this reduces the throughput of the receiver with poor quality by 44%. With superposition coding, the transmitter can superimpose the messages to receiver 2 as additional messages on the basic messages to receiver 1. Thus, the transmitter maintains a constant throughput of 60 $(=54+6)$ Mbps. This throughput is $5.55 \times$ that of the scenario without superposition coding. In addition to these scheduling gains, the presence of multiple flows in the network allows routing to create opportunities for superposition coding.

The objective of this paper is to explore how much superposition coding can gain in a typical wireless mesh
network, with simple, practical MAC and routing protocols. We show that superposition coding can be a practical method to increase the throughput of large, dense wireless mesh networks. In a mesh network with 2 to 64 active receivers and one gateway, we show that our system can increase throughput up to 283%, with average gain ranging from 15% to 51%. When there are multiple gateways, our system gains up to 190%, with average gain ranging from 31% to 71%. These results clearly demonstrate the potential benefits of our system.

To demonstrate the feasibility of implementing superposition coding, we also present a prototype implementation using GNU Radio. Our overall implementation consists of just 1000 lines of Python code and 3400 lines of C++ code.

The rest of the paper is organized as follows. In Section 2, we review related work and introduce background on superposition coding. Theoretical potential gains of superposition coding are then analyzed in Section 3. In Section 4, we present a practical design including both the MAC layer and routing layer. The details of a prototype implementation based on GNU Radio are presented in Section 5. In Section 6, using extensive evaluations, we demonstrate the throughput gains of our architecture. Our conclusions and future work are in Section 7.

2. BACKGROUND AND RELATED WORK

In this section, we provide a brief introduction to the superposition coding technique in the physical layer. While the focus of this paper is on the MAC and routing layers, we need a thorough understanding of the physical layer to design effective MAC and routing layers. For more details on the physical layer, please refer to [28].

The use of superposition coding was first introduced in [12, 9, 13] by Cover et al. in their information theoretic study of additive white Gaussian noise (AWGN) broadcast channels. In particular, Cover showed that simultaneous transmissions to multiple receivers is more efficient than using orthogonal division of the channel. He proposed superposition coding as a technique to achieve simultaneous transmissions. In [8], Bergmans showed that this technique achieves the optimal capacity for AWGN channels. Motivated by these promising results, many researchers have since considered coding and modulation strategies (e.g., [3, 25, 24, 27, 29, 30, 10]), as well as applications of superposition coding especially in the context of digital audio and video broadcasting [26]. More recently, superposition coding has been considered for standardization in cellular systems for broadcasting [1].

Superposition coding is effective in improving network throughput when the channel gains of the two receivers are asymmetric. Figure 1 shows the transmitter $S$ and two receivers. Let the channel from the transmitter to receiver 2 be better than that to receiver 1; i.e., the channel gain $h_2$ is higher than $h_1$. We refer to the two packets used in a single superposition-based transmission as two layers. The packet intended for the weaker receiver (i.e., receiver 1) is referred to as the basic layer. We also use the terms first, primary or lower layer. We refer to the other layer as the additional, second, or upper layer. It is natural to extend this technique to more than two layers.

A transmitter using superposition coding splits the available transmission power between the two layers, selects the transmission rate for each of the layers, then encodes and modulates each of the packets separately at the selected rate. The modulated symbols are scaled appropriately to match the chosen power split and summed to obtain the transmitted signal. For example, if quadrature phase shift keying (QPSK) is used for modulation of both layers, then the superposed transmitted signal will have a 16-point constellation. Figure 2 shows the example modulation.

The two receivers decode their received signal using different schemes. The weaker of the two receivers decodes its packet treating the superimposed additional layer as interference. The stronger of the two receivers first decodes the basic layer, re-encodes it, and then subtracts it from the original signal. It then decodes the remaining signal. This process is referred to as successive interference cancellation (SIC).

A major advantage of superposition coding and SIC over the traditional schemes is the expanded capacity region. Let $P$ be the total transmission power, and $p$ the power allocated to the basic layer (i.e., to receiver 1). Then according to Shannon capacity formula for an AWGN channel, the achievable rate to receiver 1 is

$$ \log_2 \left( 1 + \frac{p|h_1|^2}{(P-p)|h_1|^2 + N_0} \right) \text{bits/s/Hz}, $$
The achievable rate to receiver 2 is due to successive interference cancellation at receiver 2, where the SNR of receiver 2 is 20 dB, while that of receiver 1 is 0 dB.

Figure 3: Capacity regions with and without superposition coding. The SNR of receiver 2 is 20 dB, while that of receiver 1 is 0 dB.

where $N_0$ is the background noise. On the other hand, due to successive interference cancellation at receiver 2, the achievable rate to receiver 2 is

$$\log_2 \left(1 + \frac{(P - p)|h_2|^2}{N_0}\right) \text{ bits/s/Hz.}$$

Figure 3 shows the capacity regions with and without superposition coding [28]. The dashed line shows the boundary of the capacity region using orthogonal division with optimal power and freedom partition, while the solid line shows that of using superposition coding. The advantage of SC is clear from the figure. For example, if we use orthogonal channel division, the optimal rate that receiver 2 can achieve is 1 when the rate of receiver 1 is 0.9. Using superposition coding, without decreasing the rate of receiver 1, we can increase the rate of receiver 2 to 3, a 200% increase.

Despite the advantages of superposition coding and SIC, studies thus far have focused mainly on the physical layer and single-hop communications. Integration of this technique into the upper layers, in particular, the link and routing layers, has not been well-studied. One previous theoretical study [20] investigated the impact of superposition using a slotted ALOHA-based MAC on ad-hoc networks, it leaves many issues unaddressed and the results cannot be extrapolated to current 802.11-based wireless mesh networks.

In contrast, there are extensive recent studies on the MAC and routing layers in mesh networks using a traditional wireless channel [4]. However, such MAC and routing protocols do not take advantage of superposition coding and thus cannot fully utilize available capacity.

3. THROUGHPUT GAIN ANALYSIS

We first demonstrate the potential throughput gains of using superposition coding.

3.1 Quantization Gains

One way to apply superposition coding is to take advantage of channel rate quantization. For example, the 802.11a/g standards allow only 8 discrete rates (not including the 802.11b fallback rates). The transmission rate to a given receiver can be adapted to the particular channel condition to maximize effective throughput. A higher rate requires a higher received signal power. Depending on the receiver’s channel, there may be more than enough power to support the current discrete rate, but not enough to support the next highest discrete rate. With superposition coding, this excess power may be allocated towards sending data simultaneously to a receiver with a better channel.

To quantify the gains, we use the network in Figure 1. For simplicity, assume that the channel gains $h_1$, $h_2$ to the two receivers are given by a simple path loss model $h = \frac{d}{d^k}$, where $d$ is the distance between the sender and receiver and $k$ is the path loss exponent. We assume that receiver 2, which has the stronger channel, is at a fixed distance from the transmitter with channel gain $h_2$. The location of receiver 1 is uniformly distributed in a bounded ring centered at the transmitter with the inner radius being the distance of receiver 2 (i.e., $h_2 > h_1$).

Now we derive the probability that the excess power due to rate quantization allows superposition coding to give receiver 2 at least rate $r_2$ conditioned on receiver 1 achieving some discrete rate $r_1^*$ when there is no superposition coding and all power $P$ is used to transmit to receiver 1.

$$\Pr\{R_2 \geq r_2 | R_1 = r_1^*\} = \frac{\left(\frac{Z(r_2^*)}{Z(r_1^*)}\right)^{\frac{1}{k}} \left(1 - \frac{N_0 Z(r_2^*) (1+Z(r_1^*))}{P h_2^{W-k+1}}\right)^{\frac{1}{k}} - 1}{\left(\frac{Z(r_2^*)}{Z(r_1^*)}\right)^{\frac{1}{k}} - 1},$$

where $Z(r) = \frac{1}{I_m} \left(2^r - 1 \right)$, $W$ is the bandwidth, $I_m$ is the implementation margin, $r_1^*$ is the next highest discrete rate, and $N_0$ is the noise. See Appendix B for the derivation.

Figure 4 plots the results when receiver 2 is 40 m away from the transmitter, path loss exponent is $k = 4$, noise $N_0 = -90$ dBm, 802.11g symbol rate $W = 12 \times 10^6$, and the implementation margin $I_m$ set such that the maximum distance at which 6 Mbps can be achieved is similar to the value of Cisco Aironet 802.11g in a typical outdoor environment [11]. For clarity, this figure plots the discrete probabilities instead of the CDF indicated by Equation (3.1).

It is clear from this figure that there are many opportunities for superposition coding. For example, when receiver 1 achieves 6 Mbps, the probability that receiver 2 achieves 54 Mbps is over 50%. When receiver 1 achieves 9 or 12 Mbps, there is over a 70% probability that receiver 2 achieves 36 Mbps or higher. The reason for the high probabilities is that receiver 2 is close to the transmitter, and thus requires only a small fraction of the total power to achieve a high rate.
It can be observed in this plot that the probabilities for each \( r_1 \) are not always monotonically decreasing. For example, consider \( \Pr\{R_2 = 48|R_1 = 6\} < \Pr\{R_2 = 36|R_1 = 6\} \). One possible explanation for this is that the difference between 36 Mbps and the next highest data rate is 12 Mbps, while the difference between 48 Mbps and the next highest data rate is only 6 Mbps.

At this point, we would like to point out that quantization gains rely on using excess power that will not benefit receiver 1 in improving its rate, to superimpose messages to receiver 2 and thereby improve capacity. Thus, it is primarily helpful in improving capacity only when receiver 2 is fairly close to the transmitter. While this is not an issue in dense, mesh networks, in sparsely populated mesh networks, it will only be useful in the last hop between the mesh node and the subscribers (with gains up to 6x possible in that hop). There may be deployment requirements under which quantization gains are the only possibility to explore superposition coding. In Section 4.2, our scheduler also explores opportunities beyond quantization gains.

3.2 Rate Diversity MAC Gain

The preceding analysis considers the gain of a single transmission. In a typical 802.11 network with rate diversity due to diverse channel conditions, some receivers may achieve high rates, while others may achieve low rates. Since 802.11 MAC provides equal opportunity medium access, receivers with lower rates occupy the medium for longer durations. Let \( m \) be the ratio of the highest rate to the lowest rate. Assume that there are two receivers with one receiver at the highest rate and the other at the lowest rate. With enough power for superimposed transmission for two receivers at the two rates, using simple algebra, we can show that superposition coding can gain more than \( ml/2 \). Assume 54 Mbps and 6 Mbps (example in the introduction). Then \( m = 54/6 \), and this gives a gain of \( 4.5 \times \). For a network using the full rate set of 802.11g, \( m = 54/1 \), and this gives a gain of more than \( 20 \times \).

4. WIRELESS MESH USING SUPERPOSITION CODING

In this section, we present our MAC and routing protocols utilizing superposition coding.

4.1 Medium Access Control

To maximize reuse of previous design and increase backwards compatibility, we base the message flow of our MAC on 802.11 RTS/CTS/DATA/ACK. We make small extensions to address two issues: (1) enable superposition coded messages to be sent to two receivers; and (2) enable feedback of estimated SINRs to the transmitter.

Specifically, we extend RTS by adding an extra address. The first address denotes the receiver of the basic packet, while the second address denotes the receiver of additional packet. Since an RTS is addressed to two receivers, it triggers the transmission of one CTS packet from each. These two CTSs are separated by SIFS to avoid collision. Each CTS message contains the estimated SINR calculated using the pilot symbols in the preceding RTS message. The reported SINRs will be stored in the SINR table. Each node maintains a such table which maps each link to the corresponding estimated SINR. Superposition coding will be done based on the stored SINR values. Each superposition coded DATA packet will require two ACK packets, one from each receiver. This is handled similarly to the two CTS packets.

It is possible for either the RTS/CTS mechanism to be disabled or for there to be insufficient data traffic on which to piggyback SINR estimations. To keep the SINR table up-to-date in these cases, we add periodic keep-alive packets to provide SINR estimation and feedback. Since nodes in wireless mesh networks are fixed, large amounts of feedback are typically not required. In deployment scenarios with significant environment mobility, more sophisticated channel estimation techniques may be adopted, or the rate of feedback could be increased.

4.2 MAC Scheduling: A Greedy Scheduler

Besides control flow, the MAC layer will need a scheduling algorithm to select the data packets to transmit. We design a scheduling algorithm that takes advantage of superposition coding.

Consider a node \( u \) and assume that the links to its \( n_u \) neighbors are numbered \( 1, \ldots, n_u \). Let \( v_1, \ldots, v_{n_u} \) be the corresponding neighbors. Let the estimated channel gain to link \( i \) be \( h_i \). The routing algorithm determines the next hop and thus the link to be used for each data packet. Let \( Q_i \) be the queue of packets waiting for transmission on link \( i \). We use \( |Q_i| \) to denote the length of \( Q_i \).

For simplicity, we assume that the total power of each
transmission is fixed to be \( P \), all packets are of the same size \( s \), and the background noise is fixed at \( N_0 \).

For each transmission, we assume that the transmitter picks one rate for each layer. Let the set of discrete transmission rates be \( R \). We use the rate functions \( R^{(1)}(i, p) \) and \( R^{(2)}(i, p) \) to determine the transmission rate for link \( i \) at transmission power level \( p \), when the packet to link \( i \) is encoded in the first layer and second layer, respectively. For \( R^{(1)}(i, p) \), since the packet in the second layer with transmission power \( P - p \) will become interference to the first layer packet, the SINR used is \( \frac{\rho h_i^2}{(P - p)h_0^2 + N_0} \). On the other hand, due to successive interference cancellation when a receiver recovers a second layer packet, the SINR to determine \( R^{(2)}(i, p) \) is just \( \frac{\rho h_i^2}{N_0} \). With SINR, we search a table generated from [15] showing the relationship between packet error rate (PER) and SINR at different 802.11 rates. We pick a rate so that the PER is small, say 10%.

We impose two objectives on our scheduling algorithm. First, since a network may already have a MAC scheduler (e.g., FIFO, round robin, or proportional fairness) to determine which data packet to transmit next, our scheduling algorithm shall be generic and easily integrated with the existing scheduler to take advantage of superposition coding. We refer to the existing scheduler as the basic scheduler and our scheduler the SC scheduler. Second, superposition shall only increase throughput.

We design a greedy scheduler, referred to as \( Gopp \), to achieve the preceding objectives. Figure 5 shows the complete scheduler.

Let \( pkt_1 \) be the packet to be transmitted next as determined by the basic scheduler. Without loss of generality, we assume the next hop of \( pkt_1 \) is \( v_1 \). For ease of presentation, let \( pkt_1 \) be transmitted in the first layer. It is possible that \( pkt_1 \) is transmitted in the second layer. To handle this, we run the algorithm the second time assuming \( pkt_1 \) is in the second layer and take the better of the two solutions.

We first consider each link \( i \neq 1 \) to determine the maximum total number of packets that can be transmitted if we use the packets for link \( i \) as the second layer. Specifically, the total number of packets \( N_i \) that can be transmitted using superposition coding during the time to transmit a packet at the first layer at a power level \( p \) is:

\[
N_i = 1 + \min \left( \frac{R^{(2)}(i, p - P)}{R^{(1)}(1, p)}, |Q_i| \right),
\]

where \( \frac{R^{(2)}(i, p - P)}{R^{(1)}(1, p)} \) is the number of second layer packets we can transmit and \( |Q_i| \) is the number of backlogged packets at link \( i \). The number of packets we transmit is upper bounded by \( |Q_i| \).

Dividing \( N_i \) by the transmission time \( s/R^{(1)}(1, p) \), and ignoring \( s \) since it is the same for all packets, we define normalized effective throughput:

\[
T_i = N_i R^{(1)}(1, p).
\]

**Figure 5: A superposition scheduling algorithm \( Gopp \) at node \( u \).**

We vary \( p \) to maximize \( T_i \). Let \( p_i^* \) be the optimal power. We choose the best performing link \( i^* \), and denote the power \( p_i^* \) by \( P^* \).

However, we apply superposition coding on links 1 and \( i \) only when it achieves better throughput than scheduling the two links separately during the same time interval. That is, \( \frac{s}{R^{(1)}(1, p)} \) is used to transmit \( pkt_1 \) and the remaining time \( s - \frac{s}{R^{(1)}(1, p)} \) to transmit packets of link \( i \). Thus, we have the following constraint:

\[
1 + \min \left( \frac{R^{(2)}(i, p - P)}{R^{(1)}(1, p)}, |Q_i| \right) > 1 + \min \left( \frac{R^{(2)}(i, p)}{R^{(1)}(1, p)}, |Q_i| \right).
\]

If the link \( i \) is backlogged, the preceding equation is simplified to:

\[
\frac{R^{(1)}(1, p)}{R^{(1)}(1, p)} + \frac{R^{(2)}(i, p - P)}{R^{(2)}(i, p)} > 1.
\]

Finally, we need to update certain scheduler specific parameters. For example, if the scheduler is proportional fair, one needs to update the average rate of a given next hop.

We can further extend the scheduler to directly handle PER. Let \( q_1 \) be the PER for the first layer of link \( (u, v_1) \). Let \( q_2, q_3 \) be PER of the first and second layer of link \( (u, v_i) \) respectively. Then the expression for \( N_i \), which reflects expected number of delivered packets under inde-
dependent losses, is revised to
\( (1-q_1)+\min\left(\frac{R^{[2]}(i,p-p)}{R^{[2]}(1,p)},|Q_i|\right) \times (1-q_2)(1-q_3). \) We can use the revised expression to compute \( N_i \) and \( N_r \) in our scheduler and add a loop for PER to search for the best combination.

We now illustrate with a simple example. Suppose we have two links \( l_1 \) and \( l_2 \). The maximum individual rates achievable by the two receivers are 9 and 36 respectively. In other words, \( R^{[1]}(1,P) = 9 \) and \( R^{[1]}(2,P) = 36 \). Assume that when they are transmitted using superposition coding, they can achieve 6 and 24. Thus, \( R^{[1]}(1,p^*) = 6 \) and \( R^{[2]}(2,P-p^*) = 24 \). Assuming sufficient queued packets, then \( N^* = N_2 = 1 + 24/6 = 5 \); \( N_s = 1 + (1/6 - 1/9)36 = 3 \). Since \( N^* > N_s \), it is beneficial to use superposition coding.

### 4.3 Jointly Optimized Scheduler

The Gopp algorithm treats the basic scheduler as a blackbox. Now we show how we can design a scheduling algorithm that jointly schedules two packets. We use the proportional fairness (PF) scheduler considering multiuser diversity as an example [28]. The decision of the PF scheduler is to schedule the link \( \arg \max_{l_s} \frac{R^{[1]}(l_s,P) - R^{[2]}(l_s,P-p)}{A_l} \), where \( r_i \) is the rate achievable by link \( (u,v_i) \) and \( A_l \) is the average rate of link \( (u,v_i) \). The average rate is computed over a time window as a moving average:

\[
A_l(t+1) = (1-\alpha) A_l(t) + \alpha r_l \quad \text{if scheduled;}
\]
\[
A_l(t+1) = (1-\alpha) A_l(t) \quad \text{if not scheduled.}
\]

To apply superposition coding, we schedule the link pair \((i,k)\) that maximizes the following objective function:

\[
\max_{(i,k),p\in[0,P]} \frac{R^{[1]}(i,p)}{A_i} + \frac{R^{[2]}(k,P-p)}{A_k}.
\]

This function naturally balances throughput and fairness, and makes use of superposition coding only when needed. In this paper, we focus on the evaluation of Gopp. We leave the further study of this jointly optimized scheduler to future work.

### 4.4 Routing

Similar to our MAC scheduler, our routing algorithm achieves two objectives. First, since a network may already have a routing algorithm (e.g., WCETT [16] or ETX [14]), our routing algorithm shall be generic and easily integrated with the existing routing algorithm to take advantage of superposition coding. Second, superposition shall only increase throughput.

We design an algorithm, referred to as local 2-hop rerouting, to opportunistically take advantage of superposition coding.

Figure 6 illustrates the idea. The number associated with each link is its max achievable rate. Flow 1 going through link \((w,t)\) has a data rate of 9; flow 2 going through link \((u,v)\) has a date rate of 24. Assume that the two flows interfere with each other, and each shares the medium equally. Then the throughput for the two flows are 4.5 and 12 respectively.

Suppose that, with superposition coding of two flows at \( u \), link \((u,t)\) achieves a rate of 9 and link \((u,v)\) gets 18. By using local rerouting with superposition coding, we can improve the throughput of flow 1 without decreasing the throughput of flow 2. Specifically, we reroute flow 1 from \((w,t)\) to \((w,u)\) and \((u,t)\). By flow conservation of flow 1, we require that link \((w,u)\) access the media 1/3 of the time (thus, superposed links \((u,t)\) and \((u,v)\) get 2/3). Therefore, flow 1 achieves a throughput of \( 18 \times 1/3 = 9 \times 2/3 = 6 \), a 33% gain over the uncoded case while flow 2 achieves \( 18 \times 2/3 = 12 \), which is the same as before.

We now formalize the intuition. We assume that flow \( i \) gets \( \tau_i \) time in one unit of time. For equal time share, \( \tau_1 = \tau_2 \). The total throughput for no re-routing is

\[
T_1 = R^{[1]}(l_{wu},P)\tau_1 + R^{[1]}(l_{wu},P)\tau_2. \quad (6)
\]

We need to achieve a better throughput when using superposition coding. Let the time allocated to \( l_{wu} \) be \( \tau'_1 \). For flow conservation, we have

\[
R^{[1]}(l_{wu},P)\tau'_1 = R^{[2]}(l_{wu},P-p)\tau'_2. \quad (7)
\]

Let

\[
T_2 = (R^{[1]}(l_{wu},P) + R^{[2]}(l_{wu},P-p))\tau'_2. \quad (8)
\]

We also require that each individual flow achieve a throughput no less than the no superposition coding case. Thus, with these constraints, we want to find \( p \) that maximizes \( T_2 \) as follows,

\[
\max \quad T_2 = (R^{[1]}(l_{wu},P) + R^{[2]}(l_{wu},P-p))\tau'_2
\]
\[\text{s.t.} \quad \begin{align*}
R^{[1]}(l_{wu},P)\tau'_1 &= R^{[2]}(l_{wu},P-p)\tau'_2; \\
R^{[1]}(l_{wu},P)\tau'_2 &\geq R^{[1]}(l_{wu},P)\tau'_1; \\
R^{[2]}(l_{wu},P-p)\tau'_2 &\geq R^{[1]}(l_{wu},P)\tau'_2; \\
\tau'_1 + \tau'_2 &= \tau'_1 + \tau'_2.
\end{align*} \quad (9)
\]

Solving it, we have \( \tau'_1 = \frac{R^{[2]}(l_{wu},P-p)}{R^{[2]}(l_{wu},P-p) + R^{[1]}(l_{wu},P)}(\tau_1 + \tau_2) \).

Our routing protocol can be implemented as follows. A sender \( w \) can piggyback a solicitation message in its data packet to \( t \) if it is transmitting without using superposition coding, and its data rate is deemed to be low. Its
Figure 7: The 2-hop rerouting algorithm at node $u$.

The data rate $R(1)(l_{rt}, P)$ can be stamped in the solicitation message. Each node opportunistically estimates channel conditions of neighbors through promiscuous listening. When node $u$ gets the solicitation from $w$, it can start the calculation shown in Figure 7 using estimated channel parameters. If the calculation shows that superposition coding is beneficial, then $u$ will notify $w$ either through a separate message or piggyback in its packet to $v$ that is opportunistically listened by $w$.

If there is no more traffic from $u$ to $v$, and the 2-hop route is worse than the direct path, then $u$ can notify $w$ to switch back. Because the channel condition may change, $w$ can periodically send a packet directly to $t$ and estimate the current channel condition. This estimation will be piggybacked in packet to $u$ later. Therefore, $u$ can reassess the situation periodically.

Note that, $\tau_1$ and $\tau_2$ depend on how they share the medium. For 802.11 without any modification, each node gets equal probability of channel access, and therefore the occupation time is inversely proportional to the data rate.

5. PROTOTYPE IMPLEMENTATION

Although the focus of this paper is on the MAC and routing layers, a key concern of our system is the feasibility of implementing superposition coding. Furthermore, it is crucial to build a flexible experimental testbed to further develop our system. For these purposes, we have developed a prototype implementing superposition coding using GNU Radio [18]. GNU Radio is an open-source software radio environment for processing digital signals.

Similar to our MAC message-flow design, our physical layer design is also based on 802.11 to maximize reuse of previous design and increase backwards compatibility. We extend 802.11 packet format and processing to handle superposition coding. Below we describe both parts.

5.1 Packet Format

Standard 802.11 packet format must be extended slightly so that the receiver has sufficient information to demodulate and decode each layer. In particular, the receiver must estimate channel status and know the modulation and power for each layer.

We use a packet format header as shown in Figure 8. It is modeled after the 802.11b PLCP header, and incorporating the information in the 802.11a/g PLCP header is a straightforward extension. The existing 802.11b Start Frame Delimiter (SFD) field is used as a pilot signal for channel gain and SINR estimation. We expand the SIGNAL field to be of variable length, and add the two required parameters, the rate and power, for each layer in this field. Note that the header cannot be superposition coded, and is always transmitted with the base data rate using full transmit power.

For better performance, superposition coding requires payload data to use error-correction codes. This allows the basic signal to be accurately reconstructed despite some bit errors, thus improving the quality of the additional layer’s signal that remains after the basic signal is subtracted. Error-correction codes are not used in 802.11b, but are present in 802.11a and 802.11g. We use Reed-Solomon encoding to provide this functionality in our packet format. For simplicity, we omit packet header and payload scrambling dictated by the 802.11b standard.

5.2 Software Superposition Processing

We process packets with superposition coding using a GNU Radio implementation consisting of individual processing blocks that are implemented in software. Each processing block contains simple code that processes one or more streams of input samples and produces one or more streams of output samples.

An existing implementation [7] of 802.11b exists for GNU Radio, but due to the requirements for superposition coding detailed above, we develop our own solution. Our implementation of the transmitter is simple. It modulates and scales the individual layers of a packet in parallel, adds the resulting signals, and then prepends a header. The receiver processes the header, parses the variable-length SIGNAL field, and then demodulates and decodes the first layer of the payload. If there is another
layer to process, it reconstructs the signal and subtracts it from the original. The demodulation and decoding is done recursively until there are no more layers to process. The extracted packets are then ready to be sent to the MAC layer. If multiple packets are packed into a single layer, they will be split by the MAC layer. For a more detailed description of the receiver, see Appendix C.2.

![Block diagram of successive interference cancellation.](image)

Figure 9: Block diagram of successive interference cancellation.

```python
# Scale the remodulated signal
self.connect(self.block_mult_scale_sgnl, gr.multiply_cc())
fg.connect(self.block_cast_sgnl, self.block_mult_scale_sgnl, 0)
fg.connect(self.block_power_to_complex, self.block_mult_scale_sgnl, 1)

# Subtract the scaled signal from original signal
self.connect(self.block_sub_sgnl, gr.multiply_cc())
fg.connect(self.block_extract_sgnl, self.block_sub_sgnl, 0)
fg.connect(self.block_mult_scale_sgnl, self.block_sub_sgnl, 1)
```

Figure 10: Source code for configuration of SIC blocks.

To give the reader a sense of the implementation, Figure 9 shows the blocks that implement successive interference cancellation. Figure 10 shows a code segment configuring these necessary blocks. The multiplication and subtraction operations are simple blocks provided with GNU Radio. Other blocks we use require customized implementations. Blocks themselves are written in C++ because they are computationally intensive, while code for creating and connecting the blocks is written in Python as in the example above. Our overall implementation consists of 1000 lines of Python code and 3400 lines of C++ code.

6. EVALUATIONS

As we discussed in the preceding section, we have built a prototype implementing superposition coding. However, since we have a limited number of prototype nodes as of this writing, we conduct extensive simulations using ns-2 to evaluate our protocols.

6.1 Methodology

We have implemented superposition coding of packets at the physical layer as well as our MAC protocol, scheduling algorithm, and 2-hop rerouting in ns-2 (Ver. 2.28). We assume the underlying scheduler in FIFO.

For our simulation setup, we use the 2-ray ground model with 0 dBi antenna gains and antenna heights of 1 meter. We use wireless nodes with maximum transmit power $P = 200$ mW. To implement a realistic packet decoding model, we use a lookup table for packet error rates (PER) [15] given an observed SNR. There is a separate PER curve for each 802.11g data rate. The SNR depends on the received signal power $P_r$, noise power $N_0 = -86$ dBm, and implementation margin $I_m = 5$ dB specified in the 802.11g standard. These parameters produce maximum transmission ranges similar to those of the Cisco Aironet 802.11g in a typical outdoor environment [11].

When a packet is received in the physical layer, we decode it as follows. If the packet is not a superposition coded packet, we compute $SNR = \frac{P_r}{I_m}$ and look up the PER corresponding to the data rate at which the packet was transmitted. The packet is dropped with a probability equal to the PER.

If the packet is a superposition coded packet, we compute the SNR for the basic layer as $SNR = \frac{P_r p_1 I_m}{P_r (1-p_1) + N_0}$ where $p_1 \in [0,1]$ is the fraction of power allocated to the basic layer packet. The SNR lookup and decoding for the basic layer are done in the same way as a non-superposition coded packet. If the basic layer is decoded successfully, we compute the SNR for the additional layer as $SNR = \frac{P_r (1-p_1) I_m}{N_0}$. SNR lookup and decoding for the additional layer are also done in the same way as the non-superposition coded packet.

Note that there may be multiple packets packed into the additional layer of a superposition coded packet. These are unpacked in the MAC layer before being delivered further up the protocol stack.

To handle channel estimation, the physical layer calculates the channel gain $h = \frac{p_r}{P}$ for each received transmission. This value along with the transmission source address is passed up to the MAC layer, which manages a table of channel gains for each link. These channel gains are used to compute SNR values in the scheduling and routing layers.

We assume that all transmitted packets are 1500 bytes. We also disable RTS/CTS.

6.2 Results

6.2.1 Single gateway

We start with the case of two receivers and a single gateway transmitter. The location of receiver 2 is fixed at 10 m away from the transmitter. We vary the position of receiver 1 at 100 random locations. We run two UDP flows for a total of 60 seconds. Each UDP flow generates CBR traffic.
Figure 11: Throughput gain using the SC scheduler. Receiver 2 has a fixed position close to the transmitter.

Figure 12: CDF of throughput gain ratio. Receiver 2 has a fixed position close to the transmitter.

Figure 13: Throughput gain ratio of multiple receivers. There is a single transmitter.

Figure 14: Throughput gain ratio of multiple flows and multiple gateways.

Figure 11 is a scatter plot comparing the total throughput with and without using our SC scheduler. The x-axis is the throughput with SC scheduling turned off, and the y-axis repeats each experiment with SC scheduling turned on. We add two lines in the figure: \( y = x \) and \( y = 6x \). We observe that with standard 802.11 scheduler, the transmitter achieves total throughput in the range from 6 to 14 Mbps; with our SC scheduler, the total throughput improves to the range from 10 to 42 Mbps.

Figure 12 plots the cumulative distribution function (CDF) of throughput gain ratio, where the throughput gain ratio is defined as the ratio of throughput with SC scheduling to that without it. The median gain ratio is 2.5, corresponding to a throughput increase of 150%. We observe that the typical ratio is around 2 to 3, as indicated by the steep increase of the curve. However, there are gain ratios as high as 6. This is even higher than our theoretical analysis, due to inefficiency of a real 802.11 implementation.

Next we study the effect of the number of receivers. In this class of experiments, we choose all receivers randomly. Thus, this class of experiments is different from the preceding one which always has a receiver close to the transmitter. Typically gateways are placed close to a set of receivers, so the preceding experiment might be closer to reality and the results from this set of experiments provide a lower bound. We vary the number of receivers from 2 to 64. Figure 13 plots the min (the bottom of each bar), max (the top), average (the heavy middle horizontal line), and individual results for each given number of receivers. We observe that when we initially increase the number of receivers from 2 to 4, it increases diversity and creates opportunities for superposition coding. For example, the average gain increases from 15% when the number of receivers is 2 to 51% when the number of receivers is 4. The maximum gain we observed is 283% when the number of receivers is 4. Further increasing the number of receivers can reduce the average gain. However, even when there are 64 simultaneous active receivers, we still achieve an average gain of 17% and a maximum observed gain of 51%.

6.2.2 Multiple gateways

Finally, we evaluate the throughput gain when there are multiple gateways. This evaluates the scalability of our system with respect to the number of transmitters. We randomly place 50 nodes in a square area of 600 meters by 600 meters. We designate a number of nodes with highest degree as gateway nodes. The number of gateways ranges from 1 to 10, thus making the network dense relative to typical deployments. We fix the number of flows to 25. All flows are from gateway nodes to non-gateway nodes. A non-gateway node communicates with the closest gateway nodes through a multi-hop path. We
use the inverse of the link data rate as the routing metric. Only links with PER < 10% are used. This routing metric tries to minimize the total transmission time for a given flow. For our approach, we further apply the two-hop rerouting algorithm on top of the routing protocol using this link metric.

Figure 14 plots the results. We see that, our system consistently achieves average gains ranging from 31% to 71%. When the number of gateways is 7, our system gains up to 190%, with average at around 71%.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have designed a new wireless mesh network architecture based on superposition coding. We demonstrated that our practical and simple MAC and routing protocols are highly effective in taking advantage of superposition coding in improving network throughput with gains ranging up to 283%. We have also presented a prototype implementation of superposition coding using GNU Radio.

There are many avenues for future studies. First, we would like to build a large-scale network consisting of prototype nodes to evaluate the performance of our system. Such evaluations may reveal further opportunities to improve network throughput. Second, in this paper, we adopt a fixed overall power level. There can be gains with adaptive total power level. Third, network coding of multiple unicast sessions is emerging as an effective technique for improving network throughput (e.g., [21]). Superposition coding and network coding are two independent techniques to improve network throughput. Co-design may further improve network throughput.

8. REFERENCES


**APPENDIX**

**A. NOTATIONS**

- $c_j$: discrete rates
- $d$: distance, indexed by
- $h$: link (channel) gain
- $i$: link index
- $I_m$: implementation margin
- $j$: index of rates
- $k$: path loss exponent
- $J$: number of discrete rates
- $l$: links
- $m$: rate ratio
- $n_u$: number of neighbors at node $u$.
- $N_0$: noise
- $N_j$: number of packets that can be sent by SC
- $p$: power level
- $P$: maximum power level
- $q$: packet error rate
- $r$: rate
- $R$: function used in MAC?
- $R$: $R_l$ as random variable of rate
- $\mathcal{S}$: set of available rates
- $s$: packet size
- $S$: sender
- $t$: transmission time
- $u$: transmitting node
- $v$: receiving nodes, $v_1$, $v_2$, $v_i$
- $W$: bandwidth
- $Z$: function defined in Section 3.1

**B. QUANTIZATION GAINS**

Before deriving Equation (3.1), we first show a basic claim on superposition coding: since $h_2 > h_1$, if the encoding rate $r_1$ for the basic layer is within the Shannon capacity of receiver 1, it is also within that of receiver 2. Here the Shannon capacity of the basic layer is $W \log_2 \left(1 + \frac{p_1 h_1}{p_2 h_1 + N_0}\right)$, where $p_1$ is the power allocated to the basic layer, $p_2$ to the additional layer, and $h$ the channel gain to the receiver. Rearranging the formula, we obtain $W \log_2 \left(1 + \frac{p_1 h_1}{p_2 + N_0} \right)$. Thus, the capacity is an increasing function of the channel gain $h$, and the claim is true.

Assume the distance of receiver 2 to the transmitter is fixed and the corresponding gain is $h_2$. An implication of the preceding claim is that receiver 2 can always (w.h.p.) decode the basic message; thus it can subtract the signal of the basic layer with power $p_1 h_2$ from the original signal before recovering the additional message to itself. Thus receiver 2 decodes its message iff $r_2$ is no larger than $W \log_2 \left(1 + \frac{p_1 h_1}{N_0 h_2}\right)$. The corresponding minimum power required to achieve $r_2$ to receiver 2 is $p_2^{\text{min}}(r_2) = \frac{N_0 Z(r_2)}{h_2}$.

Now we first determine the area for receiver 1 to achieve a given discrete rate. Let the set of available discrete rates be $\mathcal{R} = \{c_0 = 0 < c_1, \ldots < c_J\}$, where $J$ is the number of discrete rates. Note that we add 0 into the set. Without superposition coding, when the position of receiver 1 corresponding to a channel gain of $h_1$, the maximum achievable rate to receiver 1 according to the Shannon capacity formula is $r_1 = W \log_2 \left(1 + \frac{p h_1}{N_0}\right)$. Then
the achieved discrete rate of receiver 1 is \( r_1^* = \max\{c \in \mathcal{R} : c \leq r_1^1\} \). We are only interested in \( r_1^+ = \min\{c \in \mathcal{R} : c > r_1^1\} \) be the next highest discrete rate. Thus, the range of channel gain \( h_1 \) for receiver 1 to achieve discrete rate \( r_1^1 \) is: \( h_1^{\text{min}} = \frac{N_0 Z(r_1^1)}{P} \leq h_1 < h_1^{\text{max}} = \frac{N_0 Z(r_1^*)}{P} \).

Converting the channel gains into distances from the transmitter, we get \( d_1^{\text{min}} = 1/h_1^{\text{max}} \), and \( d_1^{\text{max}} = 1/h_1^{\text{min}} \). The switch in the subscripts occurs because channel gains are inversely proportional to distance. The total area of the region in which receiver 1 can achieve rate \( r_1^1 \) is given by \( \pi \left( (d_1^{\text{max}})^2 - (d_1^{\text{min}})^2 \right) \).

When the distance of receiver 1 to the transmitter is \( d_1^{\text{max}} \) (and thus with a channel gain of \( h_1^{\text{min}} \)), it will use up all of the power \( P \) to achieve \( r_1^1 \). However, as it moves closer to the transmitter and the distance is reducing towards \( d_1^{\text{min}} \) (with a channel gain of \( h_1^{\text{max}} \)), the transmitter can allocate part of the total power \( P \) to \( p_2 \) for transmitting the additional packet. Specifically, when the position of receiver 1 achieves a channel gain of \( h_1 \), the maximum value of \( p_2 \) can be obtained by solving

\[
    r_1^1 = W \log_2 \left( 1 + \frac{p_1 h_1 I_m}{p_2 h_1 + N_0} \right). \tag{10}
\]

This gives \( p_2 = \frac{p - Z(r_1^1) N_0 / h_1}{\pi Z(r_1^1) + 1} \). Solving the inequality \( p_2 \geq p_2^{\text{min}}(r_2) \), we obtain the minimum channel gain \( h_1 \) (and thus the maximum distance \( d_1^{\text{max}} \)) for the inequality to hold. Since the distance cannot drop below \( d_1^{\text{min}} \), let \( d_1^{\text{min}} = \max(d_1^{\text{max}}, d_1^{\text{min}}) \).

We compute the final probability as

\[
    \Pr \{ R_2 \geq r_2 | R_1 = r_1^1 \} = \frac{(d_1^{\text{max}})^2 - (d_1^{\text{min}})^2}{(d_1^{\text{max}})^2 - (d_1^{\text{min}})^2}. \tag{11}
\]

Simplifying gives Equation (3.1).

C. GNU RADIO IMPLEMENTATION DETAILS

C.1 Available Data Rates

There are some adjustments required to support the 802.11b 1Mbps and 2 Mbps data rates. Since superposition coding requires a coherent modulation scheme, the 1Mbps and 2Mbps rates 802.11b cannot be supported as specified by the standard because they employ differential modulation (DBPSK and DQPSK, respectively). In contrast, the data rates in 802.11a/g can be supported. In 802.11a/g, each OFDM subcarrier uses a coherent modulation (BPSK, QPSK, 16-QAM, or 64-QAM). Superposition coding can be applied independently to each subcarrier after which the decoded packets may be reconstructed and forwarded up to the MAC layer. For these reasons, we use BPSK and QPSK in the context of 802.11b in our implementation with the understanding that it can be extended to 802.11a and 802.11g.

The block diagram for the receiver is shown in Figure 15. Phase recovery is required because differential modulations are no longer used. Phase recovery uses a Costas Loop. Since the Costas Loop may lock to either one of two phases (0 or \( \pi \)), the receiver demodulates according to each in parallel. (The chain for the \( \pi \) rotation is omitted from the diagram.) The SFD field of the PLCP header will only match for one of the phase rotations, so processing will only continue beyond the header processing block in one of the two chains. Once the header processing block detects a valid SFD, the phase of the Costas Loop is locked and not be allowed to sync to the incoming signal for the duration of the frame. This is required for superposition coding because the phases of the symbols in a superposition-coded payload do not necessarily align with the phases of uncoded BPSK or QPSK symbols (e.g., QPSK superposition coded with QPSK becomes 16-QAM).

The header processing block forwards the received signal corresponding to the payload to the appropriate demodulator. The basic layer is then demodulated and decoded. If decoding is successful, the data is forwarded to the MAC layer. The data is also re-encoded and re-modulated. If decoding is unsuccessful, processing stops since it is assumed that the channel quality was poor enough that there is small probability of extracting any further layers.

Once the signal for the basic layer is reconstructed, it is scaled and subtracted from the originally-received signal. The result is then sent to an appropriate demodulator for the next layer. The receiver structure takes the form of a tree with the depth being the maximum number of layers of superposition coding we support.