MegaMIMO: Scaling Wireless Capacity with User Demands

Hariharan Rahul Swarun Kumar Dina Katabi
Massachusetts Institute of Technology

ABSTRACT

We present MegaMIMO, a *joint* multi-user beamforming system that enables independent access points (APs) to beamform their signals, and communicate with their clients on the same channel as if they were one large MIMO transmitter. The key enabling technology behind MegaMIMO is a new low-overhead technique for synchronizing the phase of multiple transmitters in a distributed manner. The design allows a wireless LAN to scale its throughput by continually adding more APs on the same channel. MegaMIMO is implemented and tested with both software radio clients and off-the-shelf 802.11n cards, and evaluated in a dense congested deployment resembling a conference room. Results from a 10-AP software-radio testbed show a linear increase in network throughput with a median gain of 8.1 to 9.4×. Our results also demonstrate that MegaMIMO's joint multi-user beamforming can provide throughput gains with unmodified 802.11n cards.

Categories and Subject Descriptors

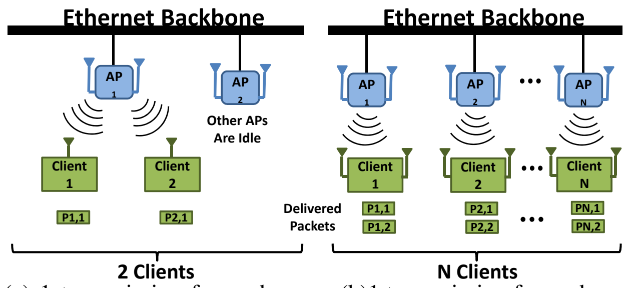
C.2.2 [**Computer-Communication Networks**]: Network Protocols

Keywords

Wireless Networks, Multi-user MIMO, Distributed MIMO

1. INTRODUCTION

Wireless spectrum is limited; wireless demands can, however, grow unlimited. Busy Wi-Fi networks, for instance, in conference rooms, hotels, and enterprises are unable to keep up with user demands [40, 16], even causing high profile failures like the wireless network collapse during the Steve Jobs iPhone 4 keynote. Cellular networks are in a similar predicament, with their demands forecast to exceed available capacity within the next few years [32]. This is not for lack of improvement in the performance of wireless devices. Indeed, individual wireless devices have improved dramatically in recent years through innovations like the introduction of multi-antenna systems, better hardware, and lower receiver noise. The problem however is that there is a mismatch between the way user demands scale and network throughput scales; user demands scale with the number of devices in the network but network throughput does not. Unless network throughput also scales with the number of devices, wireless networks will always find it hard to keep up with their demands, and the projected demands will keep exceeding the projected capacity.



(a) 1 transmission for each antenna on a *single* AP

(b)1 transmission for each antenna on *all APs*

Figure 1: **Traditional vs. Joint Multi-User Beamforming.** In a traditional multi-user beamforming system with multiple 2 antenna APs, only 1 AP can transmit on a given channel at any given time. This leads to a maximum of 2 simultaneous packet transmissions regardless of the total number of APs. In contrast, MegaMIMO enables all APs to transmit on the same channel, allowing up to $2N$ simultaneous packet transmissions if there are N 2-antenna APs.

In this paper, we present a system that enables a network to scale its throughput with the number of transmitting devices. We focus on the scenario of typical busy wireless environments such as multiple users in a conference room, enterprise, hotel *etc.* We enable a wireless LAN to keep increasing its total throughput by continuously adding more access points (APs) on the same channel.

The key technical idea behind our system is joint multi-user beamforming (MegaMIMO). Multi-user beamforming is a known technique that enables a MIMO transmitter to deliver multiple independent streams (i.e., packets) to receivers that have fewer antennas, as shown in Fig. 1(a), where a 2-antenna access point delivers two packets concurrently to two single antenna receivers. In contrast, as shown in Fig. 1(b), MegaMIMO enables multiple access points on the same channel to deliver their packets concurrently to multiple receivers, without interfering with each other. This system scales network throughput with the number of devices, and delivers as many concurrent streams/packets as the total number of antennas on all APs. Furthermore, it can leverage the continuing performance and reliability improvements of individual devices (e.g., more antennas per device).

The main challenge in implementing MegaMIMO stems from the need to synchronize the phases of distributed transmitters. Specifically, the goal of beamforming is to ensure that each client can decode its intended signal without interference. Thus, at each client, the signals intended for the other clients have to cancel each other out. This requires the transmitters to control the relative phases of their transmitted signals so that the desired cancellation can be achieved. Such a requirement is naturally satisfied in the case of a single device performing multi-user beamforming. However, in the case of MegaMIMO, the transmitters have independent oscillators, which are bound to have differences in their carrier frequencies. If one simply tries to jointly beamform these independent signals from different transmitters, the drift between their oscillators will make the signals rotate at different speeds relative to each other, causing the phases to diverge and hence preventing beamforming.

At first blush, it might seem that it would be sufficient to estimate the frequency offset (i.e., the drift) $\Delta\omega$ between the transmitters, and compensate for the beamforming phase errors as $\Delta\phi = \Delta\omega t$, where t is the elapsed time. However, such an approach is not practical. It is well known [1, 36, 15] that frequency offset estimates have errors due to noise, and using such estimates to compute phases causes rapidly accumulating errors over time. Even a small error of, say, 10 Hz (4×10^{-3} ppm, which is several orders of magnitude smaller than the mandated 802.11 tolerance of 20 ppm, or cellular tolerance of 1-2 ppm), can lead to a large error of 20 degrees (0.35 radians) within a short time interval of 5.5 ms. Such a large error in the phase of the beamformed signals will cause significant interference at the receivers, preventing them from decoding.

MegaMIMO presents a simple and practical approach for synchronizing the phases of multiple distributed transmitters. The key idea underlying MegaMIMO is to elect one of the APs as a lead and use its phase as a reference for the whole system. Other APs (i.e., the slaves) directly measure the phase of the lead AP and change the phase of their signals to maintain a desired alignment with respect to the lead. In particular, MegaMIMO precedes every data packet with a couple of symbols transmitted by the lead AP. The slave APs use these symbols to directly measure the required phase correction for proper beamforming. Since this is a direct phase measurement as opposed to a prediction based on frequency offsets, it has no accumulated errors. After correcting for this phase error, the slave APs use the estimate for their frequency offset to predict any phase changes throughout the packet and correct for it. This bounds the maximum phase error accumulation to the duration of a packet. One can use a simple long term average for the frequency offset to ensure that the phase error accumulated for the duration of a packet is within the desired performance bounds.

In the rest of the paper, we expand on this basic idea and demonstrate that it can deliver accurate joint beamforming across distributed transmitters. Further, we also extend this idea to work with off-the-shelf 802.11 cards. This would allow organizations to directly leverage MegaMIMO by simply upgrading their AP infrastructure, without requiring any modification to the clients.

We implemented MegaMIMO in two environments:

- The first environment consists of USRP2 APs and receivers, where both APs and clients can be modified. We use this environment to verify the scaling properties of MegaMIMO, and also to perform finer grained analysis of the individual components of MegaMIMO.
- The second environment consists of USRP2 APs and receivers with Intel Wi-Fi Link 5300 adapters. Each AP in this second testbed consists of two USRP2s connected via an external clock and configured to act as a 2-antenna MIMO AP. Correspondingly, each receiver Wi-Fi card has 2 antennas enabled. We use this testbed to verify that MegaMIMO can provide throughput gains with off-the-shelf 802.11n cards, and further, that MegaMIMO can provide these gains with multi-antenna devices.

We evaluated MegaMIMO in an indoor testbed using APs and receivers deployed densely in a room to simulate a conference room scenario. Our results reveal the following findings:

- **USRP testbed:** MegaMIMO's throughput increases linearly with the number of APs. In particular, in our testbed, which has 10 APs, MegaMIMO can achieve a median throughput gain of $8.1 - 9.4\times$ over traditional 802.11 unicast, across the range of 802.11 SNRs.
- **802.11 testbed:** MegaMIMO's ability to linearly scale the network throughput with the number of transmitters applies to off-the-shelf 802.11 clients. Specifically, MegaMIMO can transmit simultaneously from two 2-antenna APs to two 2-antenna 802.11n clients to deliver a median throughput gain of $1.8\times$ compared to traditional 802.11n.
- **Phase Synchronization:** MegaMIMO's distributed phase synchronization algorithm is accurate. The 95th percentile misalignment between APs observed at the receiver is less than 0.05 radians. Further, for the whole range of operational SNRs of 802.11 (5-25 dB), the reduction in SNR at each client due to misalignment, (*i.e.*, the total power of interference from all signals not intended for this client to the noise floor) increases on average by 0.13 dB for every additional AP-client pair.

Contributions: This work presents the first system that scales wireless throughput by enabling joint beamforming from distributed independent transmitters. To achieve this, we design a simple and practical approach for performing phase synchronization across multiple distributed transmitters. Finally, we also show that our system can deliver throughput gains from joint beamforming with off-the-shelf 802.11n cards.

2. RELATED WORK

(a) Empirical systems: Recent years have seen a few systems that tried to capture the gains promised by distributed multi-user beamforming [8, 31, 7, 27]. These systems, however, do not address phase synchronization, which is a basic problem in achieving such a system. In particular, they either require the base stations to be tightly synchronized with a Global Positioning System (GPS) clock[1], or assume that all the transmit antennas are driven by a single oscillator [7], or even assume that the receivers can jointly decode the data by exchanging all the received signals [27]. The closest to our work is [17], which addresses phase synchronization, but does not perform distributed joint transmission and achieves large errors (around 20 degrees) that cannot support distributed MIMO. In contrast, MegaMIMO provides the first system that achieves phase synchronization using independent oscillators at the devices in the network. As a result, MegaMIMO can enable devices to operate independently without having to share a common clock or use external clocks such as GPS. Finally, since MegaMIMO does not require any modifications to existing hardware, it can work with off-the-shelf 802.11n cards.

MegaMIMO is related to work on enabling concurrent transmissions across different nodes in the network like MU-MIMO in LTE and WiMAX [24, 21], SAM [37], IAC [10], multi-user beamforming [3] , and n+ [19]. However, these systems do not scale with the number of devices in the network. In particular, the throughput of these systems is limited either by the number of antennas on a single AP [24, 21, 37, 3], or the maximum number of antennas on any device in the network [19], or twice the number of antennas on any device in the network [10]. In contrast, MegaMIMO is the first system that enables the number of concurrent transmissions to scale with the number of APs, independent of the number of antennas on a single device. This allows MegaMIMO to support multiple independent APs communicating simultaneously with multiple independent clients.

MegaMIMO is also related to work on harnessing channel diversity gains such as distributed antenna systems [22, 5], and SourceSync [30], as well as work on phased arrays [11], which provide directional gain by sending the same signal on different antennas with different, carefully calibrated delays. However, these systems can not provide multiplexing benefits and hence, unlike MegaMIMO, cannot scale network throughput with the number of

[1]While promising, GPS typically does not work indoors, rendering such a GPS-based system hard to use in practice.

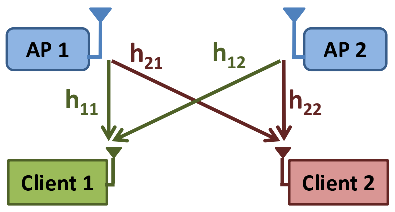


Figure 2: **Channel matrix with 2 APs transmitting to 2 clients.**

APs. Finally, recent work has shown how to synchronize concurrent transmissions in time and frequency [36, 30]. MegaMIMO builds on these results to deliver a distributed MIMO system. However, time and frequency synchronization alone are not sufficient, since joint multi-user beamforming intrinsically depends on the ability of the distributed APs to achieve phase synchronization, without which it is impossible to allow independent clients to decode simultaneously.
(b) Theoretical results: There is some theoretical work [38, 4] that addresses distributed phase synchronization, but assumes frequency synchronous oscillators and only provides one-time phase offset calibration. Further, the promise of distributed MIMO to improve the scalability of wireless networks has been explored in the theoretical community [2, 35, 41]. Work by Ozgur *et al.* [28] theoretically proved that such a setup can scale wireless capacity with the number of nodes. While MegaMIMO builds on this foundational work, MegaMIMO is the first empirical system that shows that linear scaling of throughput with the number of transmitters is possible in practical systems with unsynchronized oscillators and resulting time-varying phase differences.

3. MegaMIMO OVERVIEW

MegaMIMO is designed for the wireless downlink channel. It is applicable to wireless LANs, especially in dense deployments like enterprises, hotels, and conference rooms. MegaMIMO APs can operate with off-the-shelf WiFi client hardware. The techniques in MegaMIMO are also applicable to cellular networks, but the potential of integrating them with off-the-shelf cellular clients and evaluating them in the cellular context are beyond the scope of this paper.

MegaMIMO APs are connected by a high throughput backend, say, GigE, like APs are today. Packets intended for receivers are distributed to all APs over the shared backend. MegaMIMO enables the APs to transmit concurrently to multiple clients as if they were one large MIMO node, potentially delivering as many streams (*i.e.*, packets) as the total number of antennas on all APs.

In the next few sections, we describe how MegaMIMO works. We start with the basic idea that enables distributed phase synchronization. We then describe our protocol implementing this basic idea for emulating a large MIMO node. We then extend our system to integrate our design with off-the-shelf WiFi cards.

4. DISTRIBUTED PHASE SYNCHRONIZATION

The chief goal of distributed phase synchronization is to enable different transmitters powered by different oscillators to emulate a single multi-antenna transmitter where all antennas are driven by the same oscillator. Intuitively our solution is simple: We declare one transmitter the lead, and make all other transmitters synchronize to the oscillator of the lead transmitter, *i.e.*, each transmitter measures the offset between its oscillator and the lead oscillator and compensates for the offset by appropriately correcting the phase of its transmitted signal. This behavior makes all transmitters act as if they were antennas on the same chip controlled by the same oscillator.

We now demonstrate how this intuitive design can deliver the proper MIMO behavior and hence enable each receiver to correctly decode its intended signal without interference. For simplicity, we consider a scenario of 2 single-antenna APs transmitting to 2 single-antenna clients, as shown in Fig. 2. Let h_{ij}, where, $i,j \in \{1,2\}$ be the channel to client i from AP j, $x_j(t)$ the symbol that needs to be delivered to client j at time t, and $y_j(t)$ the symbol that is received by client j at time t. Correspondingly, let $\mathbf{H} = [h_{ij}], i,j \in \{1,2\}$ be the 2x2 channel matrix, $\vec{x(t)} = [x_1(t)\ x_2(t)]^T$ be the desired symbol vector, and $\vec{y(t)} = [y_1(t)\ y_2(t)]^T$ be the received symbol vector.

No Oscillator Offset: Assume first that there are no oscillator offsets between any of the APs and clients. If each AP i simply transmits the signal $x_i(t)$, each client will receive a linear combination of the transmitted signals. Since each client has only one antenna, client 1 receives $y_1(t) = h_{11}x_1(t) + h_{12}x_2(t)$ and client 2 receives $y_2(t) = h_{21}x_1(t) + h_{22}x_2(t)$. Each of these equations has two unknowns, and hence, neither client can decode its intended data.

In order to deliver two concurrent packets to the two clients, the APs need to ensure that each client receives only the signal intended for it (*i.e.*, it experiences no interference from the signal intended for the other client). Specifically, we need the effective channel experienced by the transmitted signal to be diagonal, *i.e.*,, it should satisfy:

$$\begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \begin{pmatrix} g_{11} & 0 \\ 0 & g_{22} \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}, \qquad (1)$$

where g_{11} and g_{22} are any non-zero complex numbers. In this case, the received signal will simply appear at each receiver as if it has experienced the channel g_{ii}, which each receiver can estimate using standard techniques.

The APs can achieve this result by using *beamforming*. In beamforming, the APs measure all the channel coefficients from the transmitters to the receivers at time 0. Then, instead of transmitting $x_1(t)$ and $x_2(t)$ directly, the APs transmit:[2]

$$\begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \mathbf{H}^{-1} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} \qquad (2)$$

In this case, the two clients receive:

$$\begin{aligned} \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} &= \mathbf{H} \begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} \\ &= \mathbf{H}\mathbf{H}^{-1} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} \end{aligned}$$

Since $\mathbf{H}\mathbf{H}^{-1} = \mathbf{I}$, the effective channel experienced by the clients in this case is a diagonal matrix, *i.e.*, Eq. 1 is satisfied. Hence, each client can now decode its intended data without interference from the signal intended for the other client.

With Oscillator Offset: What happens when the oscillators of the APs and clients have different frequencies? Let ω_{Ti} be the oscillator frequency of AP i, and ω_{Rj} the oscillator frequency of client j, $i,j \in \{1,2\}$. In this case, the channel at time t, $\mathbf{H(t)}$, can be written as:

$$\mathbf{H(t)} = \begin{pmatrix} h_{11}e^{j(\omega_{T1}-\omega_{R1})t} & h_{12}e^{j(\omega_{T2}-\omega_{R1})t} \\ h_{21}e^{j(\omega_{T1}-\omega_{R2})t} & h_{22}e^{j(\omega_{T2}-\omega_{R2})t} \end{pmatrix},$$

where $j = sqrt(-1)$. Because the oscillators rotate with respect to each other, the channel no longer has a fixed phase.

Now, if the APs try to perform beamforming as before, using the channel value they computed at time $t = 0$ and transmitting $\mathbf{H}^{-1}\vec{x}$,

[2]The APs also need to normalize $\mathbf{H}^{-1}$ to respect power constraints, but we omit that detail for simplicity.

the clients receive:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \mathbf{H(t)H^{-1}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

The product $\mathbf{H(t)H^{-1}}$ is no longer diagonal, and hence the receivers cannot decode their intended signal. Thus, standard MIMO beamforming does not work in this case.

So how can one do beamforming with such a time varying channel? A naive approach would try to make each transmitter compute $\mathbf{H(t)}$ at every t and then multiply its time signal by $\mathbf{H(t)^{-1}}$. Say that the network has N APs and N clients. Then such an approach would require each transmitter to maintain accurate estimates of N^2 frequency offsets of the form $\Delta\omega_{ij} = \omega_{Tj} - \omega_{Ri}$. (Further since nodes can only measure frequency offsets relative to other nodes, but not the absolute frequencies of their oscillators, the number of estimates cannot be reduced to N.) Measurement errors from all of these estimates will accumulate, prevent accuracy of beamforming, and create interference at the receivers. However, according to our intuition at the beginning of this section, we can make multiple transmitters act as if they were one big MIMO node, and hence do accurate beamforming, by having each transmitter estimate only its frequency offset to the lead transmitter. Said differently, our intuition tells us that it should be possible to reduce the number of frequency offset estimates that each transmitter maintains from N^2 to one. Let us see how we can achieve this goal.

Observe that we can decompose the channel matrix at time t as $\mathbf{H(t)} = \mathbf{R(t)HT(t)}$, where $\mathbf{H}$ is time invariant, and $\mathbf{R(t)}$ and $\mathbf{T(t)}$ are diagonal matrices defined as:

$$\mathbf{R(t)} = \begin{pmatrix} e^{-j\omega_{R1}t} & 0 \\ 0 & e^{-j\omega_{R2}t} \end{pmatrix}$$

and

$$\mathbf{T(t)} = \begin{pmatrix} e^{j\omega_{T1}t} & 0 \\ 0 & e^{j\omega_{T2}t} \end{pmatrix}$$

Since $\mathbf{R(t)}$ is diagonal, it can function analogous to the $\mathbf{G}$ matrix in Eq. 1. Thus, if the transmitters transmit the modified signal $\mathbf{T(t)^{-1}H^{-1}}\vec{x}$ at time t, then the received signal can be written as:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \mathbf{R(t)HT(t)T(t)^{-1}H^{-1}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3}$$

which reduces to the desired form of Eq. 1

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \mathbf{R(t)} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{4}$$

Note that $\mathbf{T(t)}$ is also diagonal, and as a result the transmitter phase correction matrix

$$\mathbf{T(t)^{-1}} = \begin{pmatrix} e^{-j\omega_{T1}t} & 0 \\ 0 & e^{-j\omega_{T2}t} \end{pmatrix} \tag{5}$$

is also diagonal. Further, the phase correction entry for each AP depends only on the oscillator phase of that AP. This means that if each AP, i, knows its phase, $e^{j\omega_{Ti}t}$, at time t, it can simply compensate for that phase and the AP will not need any additional frequency or phase measurements. Unfortunately, this is not practical. An AP has no way to measure the exact phase change of its oscillator locally.

We address this difficulty by observing that the channel equation is unchanged when we multiply by $1 = e^{j\omega_{T1}t}e^{-j\omega_{T1}t}$, *i.e*,

$$\begin{aligned} \mathbf{H(t)} &= e^{j\omega_{T1}t}\mathbf{R(t)HT(t)}e^{-j\omega_{T1}t} \\ &= \begin{pmatrix} e^{j(\omega_{T1}-\omega_{R1})t} & 0 \\ 0 & e^{j(\omega_{T1}-\omega_{R2})t} \end{pmatrix} \mathbf{H} \begin{pmatrix} 1 & 0 \\ 0 & e^{(j(\omega_{T2}-\omega_{T1})t} \end{pmatrix} \end{aligned}$$

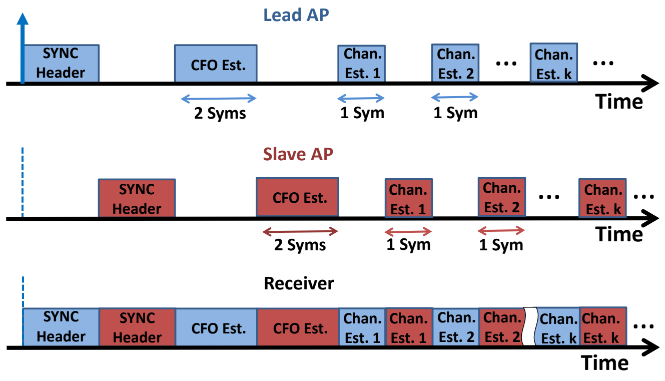


Figure 3: **Packet Structure from the perspective of APs and the receiver.** Symbols in blue are transmitted by the lead AP, symbols in red by the slave AP, and symbols in white reflect silence periods.

Since the new observed channel matrix is still diagonal, the clients can still continue to decode the received signal as before.

The resulting system implements our intuition at the beginning of this section.

5. MegaMIMO PROTOCOL

We start by describing the protocol at a high level, and follow by the detailed explanation. MegaMIMO's distributed transmission protocol works in two phases:

- MegaMIMO starts with a **channel measurement phase**, in which the APs measure two types of channels: 1) the channels from themselves to the receivers (i.e., the matrix $\mathbf{H}$), which is the beamforming channel whose inverse the APs use to transmit data concurrently to their clients; and 2) the channels from the lead AP to the slave APs (the h_i^{lead}'s), which enables each slave AP to determine its relative oscillator offset from the lead AP.
- The channel measurement phase is followed by the **data transmission phase**. In this phase, the APs transmit jointly to deliver concurrent packets to multiple receivers. Data transmission uses beamforming after having each slave AP corrects for its frequency offset with respect to the lead AP.

Note that a single channel measurement phase can be followed by multiple data transmissions. Channels only need to be recomputed on the order of the coherence time of the channel, which is several hundreds of milliseconds in typical indoor scenarios [9]. §7 describes how MegaMIMO reduces channel measurement overhead in greater detail.

We now describe the channel measurement and data transmission phases in greater detail. (The description below assumes symbol level time synchronization, for which we use the scheme in [30], which provides tight synchronization up to a few nanoseconds. Our experimental results also incorporate an implementation of that scheme).

5.1 Channel Measurement

The goal of channel measurement is to obtain a snapshot of the channels from all APs to all clients, i.e., $\mathbf{H}$ and the reference channels from the lead AP to the slave APs, i.e., the $h_i^{lead}, \forall i$.

The key point is that *all these channels have to be measured at the same time*, which is the reference time $t = 0$. Otherwise the channels would rotate with respect to each other due to frequency offsets and hence be inconsistent. Below, we divide channel measurement into a few sub-procedures.

(a) Collecting Measurements. The lead AP starts the channel measurement phase with a synchronization header, followed by channel measurement symbols, *i.e.*, known OFDM symbols that the clients can use to estimate the channel. The channel measurement symbols are separated by a constant gap, whose value is chosen to permit the slave APs to send their channel measurement symbols interleaved with the symbols from the lead AP. When the slave APs hear the synchronization header, they know to transmit their channel measurement symbols in the gap, one after another, as shown in Fig. 3.

Thus, channel measurement symbols are repeated and interleaved. They are repeated to enable the clients to obtain accurate channel measurements by averaging multiple estimates to reduce the impact of noise. They are interleaved because we want the channels to be measured as if they were measured at the same time. Since exactly simultaneous transmissions will lead the APs to interfere with each other, MegaMIMO performs a close approximation to simultaneous transmission by interleaving symbols from different APs.

(b) Estimating H at the clients. Upon reception of the packet in Fig. 3, each client performs three tasks: it computes its carrier frequency offset (CFO) to each AP; it then uses its knowledge of the transmitted symbols and the CFO to compute the channel from each AP to itself; and finally it uses its knowledge of the CFOs to rotate the phase of the channels so that they look as if they were measured exactly at the same time. We detail these tasks below.

Different transmitters (*i.e.*, APs) have different oscillator offsets to receivers, and each receiver needs to measure the frequency offset from each transmitter to correct the corresponding symbols from that transmitter appropriately. To enable this, the channel measurement transmission uses CFO symbols from each AP followed by channel estimation symbols similar to traditional OFDM [15]. The only departure is that the receiver computes and uses different CFO and channel estimates for symbols corresponding to different APs.

Note that these channel estimates are still not completely simultaneous, in particular, the channel estimation symbols of slave AP i is separated from the symbol of the lead AP by $i-1$ symbol widths, as shown in Fig. 3. The receiver compensates for this by rotating the estimated channel for AP i by $e^{-j\Delta\omega_i(i-1)kT+D}$ (in each OFDM subcarrier), where T is the duration of one OFDM symbol, k is the index of the interleaved symbol, and D is the duration of the lead AP synchronization header. This ensures that all channels are measured at one reference time, which is the start of the synchronization header. The receiver averages the channel estimates (in each OFDM subcarrier) from each AP to cancel out the noise and obtain an accurate estimate. The receivers then communicate these estimated channels back to the transmitters over the wireless channel.

(c) Estimating the h_i^{lead}'s at the Slave APs. Each slave AP uses the synchronization header to compute the value of the channel from the lead AP to itself at the reference time $h_i^{lead}(0)$.

Note that at the end of the channel measurement phase, each slave AP i has the entire channel matrix to be used for beamforming, as well as a reference channel, $h_i^{lead}(0)$ from the lead AP which it will use during data transmissions, with all channels measured with respect to one reference time.

5.2 Data Transmission

Now that the channels are measured, the APs can use beamforming to transmit data concurrently without interference.

(a) AP Coordination: The APs need to agree on which packets are sent concurrently in one beamforming frame. To do this we leverage the bandwidth of the backend Gigabit Ethernet to send all client packets to all APs. The lead AP makes all control decisions and communicate them to the slave APs over the Ethernet. In particular, it determines which packets will be combined in a data transmission and communicates it to the slave APs over the wired backend.

(b) Beamforming: Client packets are transmitted by joint beamforming from the MegaMIMO APs participating in the system. Note that slave APs need to correct the phase of their signal prior to transmission. One way to do this would be for each slave to estimate the frequency offset $\omega_{lead} - \omega_{slave}$ from the lead to itself (using the synchronization header from the previous phase) and then compute the net elapsed phase by calculating $(\omega_{lead} - \omega_{slave})t$, where t is the time elapsed since the channel measurement was taken. However, this would lead to large accumulated errors over time because of any inaccuracies in the measurement of the initial frequency offset. For example, even a small error of 100 Hz in the measurement of the initial frequency offset can lead to a large phase error of π radians in as short a timespan as 20 ms, and hence significantly affect the phase alignment required for correct beamforming. Unless addressed, this error would prevent MegaMIMO from amortizing the cost of a single channel measurement over the coherence time of the channel, e.g., 250 ms, and would force the system to repeat the process of measuring **H** every few milliseconds, which means incurring the overhead of communicating the channels from all clients to the APs almost every packet.

MegaMIMO avoids this issue of accumulating error over large timescales by directly measuring the phase difference between the lead AP and the slave AP. Said differently instead of multiplying the frequency offset $\Delta\omega(=\omega_{lead} - \omega_{slave})$ by the elapsed time (which leads to errors that accumulate over time), MegaMIMO directly measures the phase difference $\Delta\phi(t)(=(\omega_{lead} - \omega_{slave})t)$.

In MegaMIMO the lead AP initiates data transmission using a synchronization header, as in channel estimation. Each slave AP use this synchronization header to measure the current channel, $h_i^{lead}(t)$ from the lead AP to itself. Note that the current channel will be rotated relative to the reference channel because of the oscillator offset between the lead AP and slave AP. In particular, $h_i^{lead}(t) = h_i^{lead}(0)e^{j(\omega_{T1}-\omega_{T2})t}$. Each slave can therefore compute $e^{j(\omega_{T1}-\omega_{T2})t}$ directly, from its two measurements of the lead AP channel. Such an estimate does not have errors that accumulate over time because it is purely a division of two direct measurements. The slave then multiplies its transmitted signal by this quantity, as described in §4.

Now that all AP oscillators are synchronized at the beginning of the data transmission, the slave AP also needs to keep its oscillator synchronized with the lead transmitter through the actual data packet itself. It does this by multiplying its transmitted signal by $e^{j(\omega_{T1}-\omega_{T2})t}$ where t is the time since the initial phase synchronization at the beginning of the joint transmission. Note that this offset estimate only needs to be accurate within the packet, *i.e.*, for a few hundred microseconds or about 2 ms at most. MegaMIMO APs maintain a continuously averaged estimate of their offset with the lead transmitter across multiple transmissions to obtain a robust estimate that can maintain accurate phase synchronization within a packet.

Two additional points about MegaMIMO's synchronization are worth noting.

First, for ease of exposition, we have discussed the entire system so far in the context of correcting carrier frequency offsets. However, any practical wireless system has to also account for the sampling frequency offsets. Note that any offset in the sampling frequency just adds to the phase error in each OFDM subcarrier. Since our phase offset estimation using the synchronization header, described in §5, estimates the overall phase, it automatically accounts for the initial phase error accumulated from sampling frequency offset. Within each packet, the MegaMIMO slave APs correct for the effect of sampling frequency offset during the packet by using a long-term averaged estimate, similar to the carrier frequency offset.

Second, as mentioned earlier, in §5, MegaMIMO APs are synchronized in time using [30]. As described in [30], due to differences in propagation delays between different transmitters and different receivers, one cannot synchronize all transmitted signals to arrive exactly at the same time at all receivers. It is important to note that MegaMIMO works correctly even in the presence of different propagation delays between different transmitters and receivers. This is because the signals from different MegaMIMO APs will arrive within a cyclic prefix of each other at all receivers.[3] The delay differences between the signals from different APs at a receiver translate to a relative phase difference between the channels from these APs to that receiver. MegaMIMO's channel measurement phase captures these relative phase differences in the channel matrix, and MegaMIMO's beamforming then applies the effect of these phase differences while computing the inverse of the channel matrix.

5.3 Overarching Principles

In summary, the core challenge met by MegaMIMO's design is to accurately estimate and track the phase differences between each of the *N* clients and *N* APs. This challenge is particularly arduous for two reasons: (1) each receiver must simultaneously track the phase of *N* independent transmitters, and (2) errors in the estimates in the CFO result in phase offsets that accumulate over time, quickly leading to very large errors. Our general approach to tackling these challenges is to have all transmitters and receivers synchronize their phase to that of a single lead transmitter. Our implementation of this approach has been guided by following three overarching principles:

- **Between APs and within a packet we can use estimated frequency and sampling offsets to track phase:** We can measure the frequency and sampling offsets *between APs* accurately enough that the accumulated phase differences within a single packet (10s to a few 100s of *microseconds*) are not significant enough to harm performance. Specifically, since APs are a part of the infrastructure, and CFOs do not change significantly over time, we can get very accurate estimates of the CFO between APs by averaging over samples taken across many packets.
- **Between APs and *across* packets we *cannot* use estimated frequency and sampling offsets to track phase:** The across packet time scales (10s to 100s of *milliseconds*) are large enough that even with extremely accurate estimates of the frequency and sampling offsets, the accumulated phase differences from residual errors will lead to significant performance degradation. To handle this, MegaMIMO uses a single header symbol to directly estimate the total *phase offset* and re-sync the phases of all nodes at the beginning of each packet.
- **Between a client and an AP, we cannot use estimated frequency and sampling offsets to track phase even through a packet:** Since clients are a transient part of the network, we cannot get accurate enough estimates of frequency and sampling offsets to use for phase tracking even within a single packet. Thus each client uses standard OFDM techniques to track the phase of the lead AP symbol by symbol, Additionally when performing channel estimation, the APs interleave their packets so that the correction of the channels to a common reference time has minimal error.

6. COMPATIBILITY WITH 802.11

In order for MegaMIMO to work with clients using off-the-shelf 802.11 cards, MegaMIMO needs to address two challenges:

[3]In fact, since the common design scenario for MegaMIMO is confined locations like conference rooms and auditoriums, the propagation delay differences between different APs to a receiver are in the tens of nanoseconds, which is smaller than the 802.11 cyclic prefix of 400 or 800 ns, which is designed for worst case multipaths.

1. **Sync header:** The sync header transmitted by the lead AP to allow the slave APs to compute their oscillator offset, and trigger their transmission, is not supported by 802.11.

2. **Channel measurement:** Recall that MegaMIMO requires a snapshot of the channel from all transmitters to all receivers measured at the same time. In §4, we described how to do this with a custom channel measurement packet format with interleaved symbols that allows a receiver to measure channels from all transmitters. However, such a packet format is not supported by 802.11, and hence 802.11 cards cannot simultaneously measure channels from all APs at the same time.

MegaMIMO solves these issues in the context of 802.11n by leveraging 802.11n channel state information (CSI) feedback for beamforming. We now describe MegaMIMO's solutions to each of the challenges listed above.

6.1 Sync Header

The lead AP in MegaMIMO needs to prefix each transmission with a sync header that allows the slave transmitters to measure their relative oscillator offset from the lead, and also triggers their joint transmission. A mixed mode 802.11n packet essentially consists of an 802.11n packet prefixed with 5 legacy symbols. These legacy symbols are only intended to trigger carrier sense in 802.11a/g nodes, and are not used by 802.11n receivers. Thus, the lead MegaMIMO can use these legacy symbols as a sync header. MegaMIMO slave APs use the legacy symbols to measure their oscillator phase offset from the lead, correct their transmission signal, and join the lead AP's transmission after the legacy symbols when the actual 802.11n symbols are transmitted.

6.2 Channel Measurement

802.11n does not support the interleaved packet format that allows MegaMIMO to measure a snapshot of the channels from all the transmitters to a receiver simultaneously. Even more fundamentally, an 802.11n receiver with *K* (at most 4) antennas can measure at most *K* channels at a time. In a MegaMIMO system, the total number of transmit antennas across all APs is larger than the number of antennas on any single receiver. Thus, a receiver with off-the-shelf 802.11n cards will be unable to simultaneously measure channels from all transmit antennas to itself.

Naively, one could measure the channels from all transmit antennas by transmitting a separate packet from each AP, and then correcting these channel measurements using the estimated frequency offsets to the receiver like in §5.1. Unlike the scenario in §5.1 where the transmissions from different APs are separated from each other by only a few symbols (using interleaving), the transmissions from different APs here are separated by at least one packet width. As discussed in §5.3, it is not practical to compute receiver frequency offsets accurately enough to ensure that the accumulated phase error across packets will be tolerable.

MegaMIMO instead performs robust channel measurement by "tricking" the receiver into measuring channels from different AP antennas simultaneously. This trick allows MegaMIMO to measure the channel from each AP antenna to the receiver in conjunction with a common reference channel to the receiver. Using such a common reference across all measurements allows MegaMIMO to avoid measuring the *receiver frequency offset*, and instead directly estimate the oscillator *phase offset* between different channel measurements, and therefore compensate for it, as we describe below.

For simplicity, we focus on the scenario in Fig. 4 with 2 APs (a lead and a slave) and 1 client, where all nodes have 2 antennas each. In the rest of this discussion, we will focus only on the channel mea-

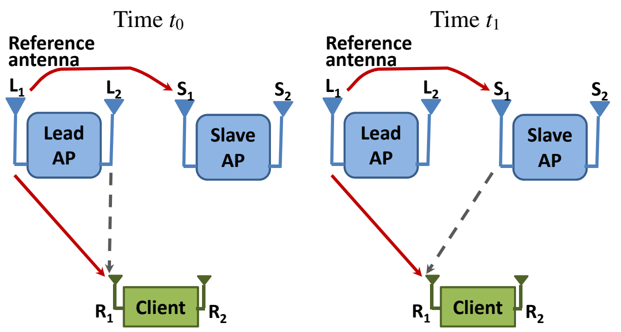


Figure 4: **802.11 Channel Measurement:** MegaMIMO measures channels with 802.11 clients by sending a series of two-stream transmissions. Every transmission includes the reference antenna, L_1, as well as one other antenna (either L_2 or S_1 in our example). Note that for clarity, the figure does not show the transmissions to/from R_2 and S_2, but MegaMIMO naturally measures the channels to R_2 simultaneously.

surements to R_1 since the channels to R_2 are naturally measured simultaneously with R_1 in exactly the same manner.

At time t_0, L_1 and L_2 transmit a 2-stream packet jointly to R_1. This measurement gives us the channels $L_1 \rightarrow R_1$ and $L_2 \rightarrow R_1$ at time t_0. In addition, S_1 measures the channel $L_1 \rightarrow S_1$ using the synchronization header.

At time t_1, L_1 and S_1 trick the receiver by jointly transmitting a 2-stream packet from 2 different APs. This measurement gives us the channels $L_1 \rightarrow R_1$ and $S_1 \rightarrow R_1$ at time t_1. Again, S_1 measures the channel $L_1 \rightarrow S_1$ using the synchronization header.

The challenge is that we would like to obtain the channel $S_1 \rightarrow R_1$ at time t_0 but we have only the channel $S_1 \rightarrow R_1$ measured at t_1.

We therefore need to correct our measured channel by the accumulated phase offset between S_1 and R_1 in the time interval t_0 to t_1. To do this, we take advantage of the fact that we can compute the accumulated phase offset between both L_1 and R_1, and between L_1 and S_1 in the time interval t_0 to t_1.

- L_1 **and** R_1**:** We can compute this accumulated phase offset using the measurements of the channel $L_1 \rightarrow R_1$ at time t_0 and time t_1.
- L_1 **and** S_1**:** We can compute this accumulated phase offset using the measurements of the channel $L_1 \rightarrow S_1$ at time t_0 and time t_1.

The difference between these two accumulated phase offsets gives us the desired accumulated phase offset between S_1 and R_1 in the time interval t_0 to t_1.

We can similarly measure the channel $S_2 \rightarrow R_1$ in the next time slot, say t_2, and rotate it back to time t_0. We can repeat this process for all AP antennas.

7. DECOUPLING MEASUREMENTS TO DIFFERENT RECEIVERS

The scheme described in §4 assumed that the channels from all APs to all receivers are all measured at the same time. In §6.2, we showed how MegaMIMO could measure channels from different APs to a single receiver at different times and compensate for differences in oscillator offset by using a shared reference measurement across all APs for that receiver. But what about the channels to a different receiver? If this receiver joins the wireless network after the channels to the first receiver are measured, there is no opportunity for a shared reference measurement between the two receivers. It might therefore seem that MegaMIMO's requirement for all channels to be measured at the same time would necessitate measurement of channels to all receivers whenever a receiver joins the network, or when a single receiver's channels change.

In fact, we can show that such full measurement is not necessary, and that MegaMIMO can decouple channel measurements to different receivers. The key idea is that MegaMIMO can use the channels from the lead AP to slave APs as a shared reference in this case, instead of the channel from the lead AP to a receiver as was the case in §6.2. We prove in the appendix that using such a shared reference allows MegaMIMO to measure channels to different receivers at different times, and still correctly perform multi-user beamforming using distributed phase synchronization.

8. DIVERSITY

MIMO systems can provide both multiplexing and diversity gains. So far, we have described the use of MegaMIMO for multiplexing. The same discussion applies to diversity except that in this case, we have all the APs transmitting jointly to a single client, say client 1. Each AP then computes its beamformed signal as $\frac{h^*_{1i}}{\|h_{1i}\|}x_1$ and slaves continue to perform distributed phase synchronization as before.

9. MegaMIMO'S LINK LAYER

So far, we have described MegaMIMO's physical layer, which enables multiple APs to transmit simultaneously to multiple receivers. We now describe how MegaMIMO's link layer is designed to use this capability.

MAC and Carrier Sense: In MegaMIMO, all downlink packets are sent on the Ethernet to all MegaMIMO APs. Thus, all APs in the network have the same downlink queue. Each packet in the queue has a *designated AP*, which is the AP with the strongest SNR to the client to which that packet is destined. MegaMIMO always uses the packet at the head of the queue for transmission, and nominates the designated AP of this packet as the lead AP for this transmission. The lead AP then chooses additional packets for joint transmission with this packet in order to maximize the network throughput. There are a variety of heuristics [43, 33, 42] that can be adopted for selecting the packets for joint transmission, and we leave the exact algorithm for making this choice for future work.

The lead AP contends on behalf of all slave APs, with its contention window weighted by the number of packets in the joint transmission as described in [29]. Clients contend for the medium as they do today using 802.11 CSMA. When the lead AP wins a contention window, it starts transmitting its synchronization header, which causes the slave APs to join the transmission. We note that contention for joint transmission by the lead AP is robust to hidden terminals for two reasons. First, MegaMIMO is intended for dense deployments like conference rooms where access points can hear each other, and the overall wireless capacity is limited by interference between access points. Further, even in the unlikely event of hidden terminals, situations causing persistent packet loss due to repeated collisions can be detected using mechanisms like in [34], and the lead AP can ensure that JMB access points that trigger hidden terminal packet loss above a threshold are not part of the joint transmission. This ensures that both MegaMIMO joint transmissions, as well as other transmissions, do not encounter persistent hidden terminals.

Rate Selection using Effective SNRs: In systems like MegaMIMO where different sets of APs transmit concurrently for different packets, the rate to a client can change from packet to packet as the effective channel at each client changes as a result of beamforming. Such systems therefore need to use a rate-selection algorithm [19]. MegaMIMO uses the **effective SNR** algorithm, which is designed for rate selection for 802.11-like frequency selective wideband channels [13]. Since APs in MegaMIMO know the full channel matrix, **H**, prior to transmission. APs multiply the signals by $k\mathbf{H}^{-1}$ (k accounts for the maximum power constraint at APs). Thus, the effective channel is $k\mathbf{H}^{-1}\mathbf{H} = k\mathbf{I}$, giving a signal strength of k^2 at each client. Client cards report noise N as in [11]. Clients send the noise N to APs along with the measured channels. Since APs know the signal strength, k^2, in each subcarrier, and the associated noise N, they can compute the SNR in each subcarrier as $\frac{k^2}{N}$. They can then map this set of SNRs to rate by performing a table lookup [13]. Thus, each client in a MegaMIMO joint transmission gets the same rate, which is similar to traditional 802.11 fairness.

Acknowledgments: MegaMIMO disables synchronous ACKs at clients and uses higher layer asynchronous acknowledgments like in prior work such as MRD and ZipTx [25, 20]. Further, similar to systems like Maranello [14], MegaMIMO can modify the firmware on clients to implement an optimized joint synchronous acknowledgment protocol consisting of a single SIFS, followed by back-to-back acknowledgments from all the clients.

Packet losses and retransmissions: It is important to note that, even though APs transmit packets jointly to different receivers, packet losses at different clients are decoupled. Specifically, if APs have stale channel information to a client, only the packet to that client is affected, and packets at other clients will still be received correctly. As in regular 802.11, APs in MegaMIMO keep packets in the queue until they are ACKed. If a packet is not ACKed, they can be combined with other packets in the queue for future concurrent transmissions.

10. TESTBED AND IMPLEMENTATION

We implement MegaMIMO's AP design in software radios and evaluate it with both off-the-shelf 802.11 clients and software-radio clients.

(a) Implementation for the software radio testbed: In this testbed, each node is equipped with a USRP2 board [6], and an RFX2400 daughterboard, and communicates on a 10 MHz channel in the 2.4 GHz range. We implement OFDM in GNURadio, using various 802.11 modulations (BPSK, 4QAM, 16QAM, and 64QAM), coding rates, and choose between them using the effective-SNR bitrate selection algorithm [13].

Our MegaMIMO implementation includes the following modules: distributed phase alignment, beamforming for multiplexing and diversity, and bitrate selection. We do not implement ACKs, CSMA, or retransmissions. To perform correct phase alignment, concurrent transmitters must be synchronized tightly at the sample level. We do this by using USRP2 timestamps to synchronize transmitters despite delays introduced by software. Before every data packet, the lead AP sends a trigger signal on the medium at $t_{trigger}$. All other APs log the timestamp of this signal, add a fixed delay $t_{\triangle}$ to it, and then transmit concurrently at this new time. We select $t_{\triangle}$ as 150μs based on the maximum delay of our software implementation. Finally, to optimize the software turnaround, we did not use GNURadio, but wrote our own C code that directly interacts with the USRP hardware.

(b) Implementation for the 802.11 testbed: There are two main differences between this testbed and the one above. First, each client in this testbed uses an off-the-shelf 802.11 card. Second, each node in this testbed has two antennas and can act as a MIMO node. Our objective is to show that MegaMIMO extends beyond single antenna systems; For example, it can combine two 2x2 MIMO systems to create a 4x4 MIMO system.

Each AP is built by connecting two USRP2 nodes via an external clock and making them act as a 2-antenna node. Each client is a PC equipped with a Intel Wi-Fi Link 5300 a/b/g/n wireless network adapter on which 2 antennas are enabled. The Intel Wi-Fi Link 5300 adapters are updated with a custom firmware and associated `iwlwifi` driver in order to obtain the channel state information in user space [12].

The AP software implementation is similar to the other testbed except that we make the channel width 20 MHz to communicate with actual 802.11 cards. The packet format is also changed to match 802.11 packets. The client software collects the channel measurements from the firmware and logs correctly decoded packets.

(c) Testbed Topology: We evaluate MegaMIMO in an indoor wireless testbed that simulates a conference room or classroom, with APs deployed on ledges near the ceiling, and clients scattered through the room. Fig. 5 shows node locations in the experimental environment. In every run, the APs and clients are assigned randomly to these locations. Note that the testbed exhibits significantly diverse SNRs as well as both line-of-sight and non line-of-sight paths due to obstacles such as pillars, furniture, ledges etc. The APs transmit 1500 byte packets to the clients in all experiments.

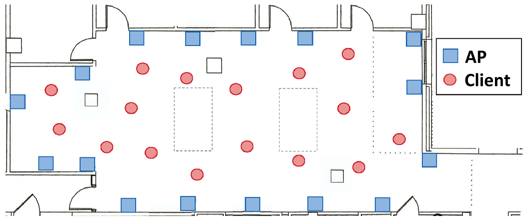


Figure 5: **Testbed Topology.** Client locations are marked by red circles, and AP locations by blue squares. *Note that the figure shows the set of possible locations for clients and APs, and different subsets of locations are picked for different experiments.*

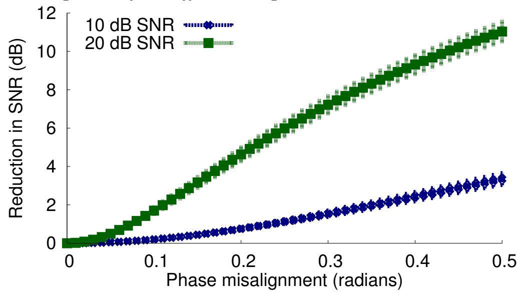


Figure 6: **Degradation of SNR due to phase misalignment.** Even with only 2 transmitters, a misalignment of just 0.35 radians can reduce the SNR by almost 8 dB at the receivers due to interference.

11. RESULTS

We evaluate MegaMIMO both through microbenchmarks of its individual components, as well as an integrated system on both USRP and 802.11n testbed.

11.1 Microbenchmarks

(a) Necessity of Phase Alignment: A key challenge for a distributed MIMO system is that it must compensate for oscillator offsets between the transmitters. In this section, we demonstrate the impact of misalignment between transmitters on the received SNR.

Method. We simulate a simple 2-transmitter, 2-receiver system where different data is intended for each receiver. The transmitters measure the initial channel matrix to the receivers, and use this matrix to compute their beamforming vectors. We then introduce a phase misalignment at the slave transmitter, and compute the reduction in SNR at each receiver as a result of this misalignment. We

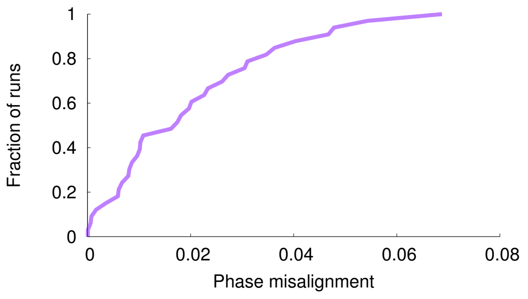


Figure 7: **CDF of observed phase misalignment.** The median misalignment is 0.017 radians, and the 95th percentile misalignment is 0.05 radians.

repeat this process for 100 different random channel matrices, phase misalignments from 0 to 0.5 radians, and for two systems - one in which the average SNR is 10 dB, and other in which the average SNR is 20 dB.

Results. Fig. 6 shows the average reduction in SNR in dB, as a function of phase misalignment. As one would expect, an increase in phase misalignment increases the interference at each receiver. As the graph shows, even a phase misalignment as small as 0.35 radians[4] , can cause an SNR reduction of almost 8 dB at an SNR of 20 dB. This SNR reduction will be greater as we add more and more transmitters to the system. Further, phase misalignment causes a greater reduction in SNR when the system is at higher SNR. This is because the impact of additional noise added by interference is higher when the original noise itself is low, *i.e.*, at high SNR.

(b) Accuracy of MegaMIMO's Phase Alignment: We now examine the accuracy of MegaMIMO's lightweight distributed phase alignment algorithm.

Method. In this experiment, we place two MegaMIMO nodes at random AP locations and a third MegaMIMO node at a receiver location. We randomly pick one of the two APs to be the lead and the other to be the slave. The slave transmitter implements MegaMIMO's distributed phase synchronization algorithm, and performs phase correction on its transmission before joining the lead transmitter's data transmission. In order to measure the accuracy of MegaMIMO's phase synchronization algorithm, we make the lead and the slave APs alternate between transmitting OFDM symbols. In particular, each transmitter's transmission consists of pairs of an OFDM symbol followed by an OFDM symbol length of silence. The transmissions of the lead and slave transmitter are offset by 1 symbol so that the receiver sees alternating symbols from lead and the slave transmitter. The receiver estimates the lead and slave transmitter's channels separately, and computes the relative phase between them. We then perform several rounds of this measurement. The receiver uses the first measurement of relative phase as a reference, and computes the deviation of relative phase from this reference in subsequent transmissions.

Result. Fig. 7 plots the CDF of the absolute value of the deviation in relative phase across all the experiments. If the lead and slave transmitter are always perfectly aligned, the deviation should be zero. However, estimation errors due to noise and oscillator drift due to the delay from when the slave measures the lead's channel and turns around to jointly transmit data will induce misalignment. As can be seen, however, the median misalignment is less than 0.017 radians, and the 95th percentile misalignment is less than 0.05 radians. Based on Fig. 6, with two transmitters, MegaMIMO's phase

[4]0.35 radians is much smaller than the misalignment expected with the mandated 802.11 tolerance of 20 ppm.

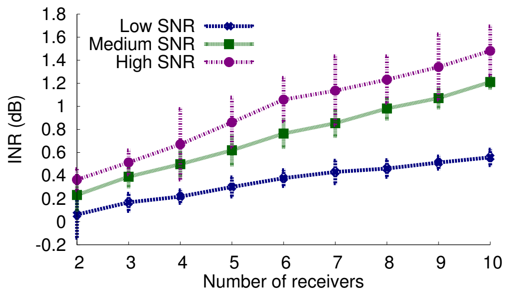


Figure 8: **Accuracy of Phase Alignment.** Average INR at receivers for various numbers of receivers across different SNR ranges. INR stays below 1.5dB across SNRs even with 10 receivers.

alignment algorithm can ensure that the SNR of joint transmission is not reduced by 0.4 dB at the 95th percentile.

(c) How does SNR reduction scale? The previous experiments examined in depth the impact of misalignment and MegaMIMO's precise alignment performance in the case of a 2x2 distributed MIMO system. In this experiment, we observe how the SNR reduction grows as we increase the number of transmitters in the system.

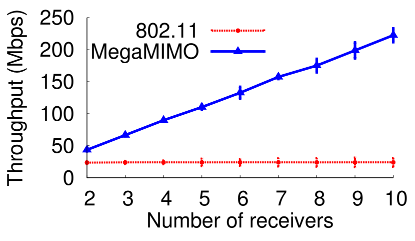
Method. We evaluate the SNR reduction in MegaMIMO in three effective SNR [13] ranges: low (6-12 dB), medium (12-18 dB) and high ($>$ 18 dB). For each range, we place several MegaMIMO nodes in random AP locations in the testbed. We then place the same number of MegaMIMO nodes in random client locations, such that all clients obtain an effective SNR in the desired SNR range. For each placement, we then choose a client at which all APs null their interference, *i.e.* the expected signal at that client is zero and measure the received signal power at that client. If phase alignment is perfect, the received signal power should be comparable to noise, *i.e.* the ratio of the received signal power to noise should be 0 dB. Any inaccuracy in phase misalignment will lead to interference, manifest as a higher ratio of received power to noise, and produce a corresponding reduction in SNR if data were actually transmitted to that client. For each topology, we null at each client, and compute the average interference to noise ratio (INR) across clients. We repeat this experiment for different topologies, and different numbers of MegaMIMO APs at low, medium and high SNRs.

Result. Fig. 8 shows the INR as a function of the number of MegaMIMO APs at low, medium and high SNRs. Note that, as before, the reduction in SNR increases with SNR, but is below 1.5 dB even at high SNR. The INR also increases with the number of APs, as the number of interferers increases, but increases gradually: only $\sim$ 0.13 dB for every additional AP-client pair even at high SNR.

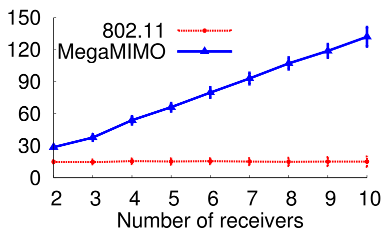
11.2 Increase of Network Throughput with the Number of APs

MegaMIMO's key goal is to increase the network throughput as the number of APs increases. This experiment verifies if MegaMIMO delivers on that promise.

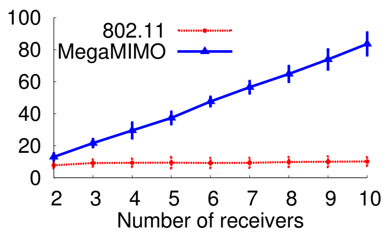
Method. We evaluate MegaMIMO's performance in three effective SNR ranges: low (6-12 dB), medium (12-18 dB) and high ($>$ 18 dB). For each range, we place a certain number of MegaMIMO nodes in random AP locations in the testbed. We then place the same number of nodes in random client locations such that all clients obtain an effective SNR in the desired range. For each such topology, we measure the throughput obtained both with regular 802.11 and MegaMIMO. Since USRP2 cannot perform carrier sense due to software latency, we measure the throughput of 802.11 by scheduling each client so that it gets an equal share of the medium. We repeat the experiment for 20 different topologies and



(a) High SNR (> 18 dB)

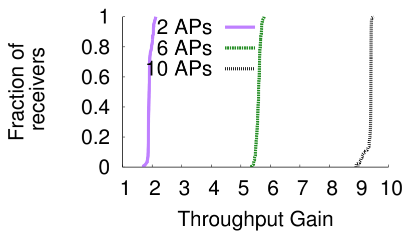


(b) Medium SNR (12-18 dB)

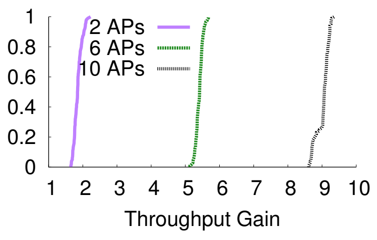


(c) Low SNR (6-12 dB)

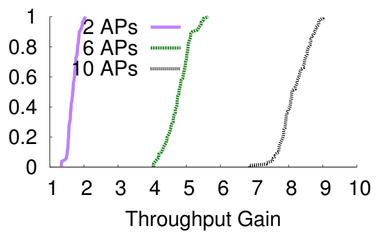
Figure 9: **Scaling of throughput with the number of APs.** In this experiment, the number of APs equals the number of receivers. At all SNRs, MegaMIMO's network throughput increases linearly with the number of APs while total 802.11 throughput remains constant.



(a) High SNR (> 18 dB)



(b) Medium SNR (12-18 dB)



(c) Low SNR (6-12 dB)

Figure 10: **Fairness.** CDFs of per-client throughput gain. Across all SNRs, MegaMIMO provides all clients with very similar gains.

also vary the number of MegaMIMO APs for each SNR range.

Results. Figs. 9(a), (b), and (c) show the total throughput obtained by 802.11 and by MegaMIMO for different numbers of APs, and different SNR ranges. Note that, as one would expect, the obtained throughput increases with SNR (802.11 throughput at low SNR is 7.75 Mbps, at medium SNR is around 14.9 Mbps, and at high SNR is 23.6 Mbps). There are two main points worth noting:

- 802.11 cannot benefit from additional APs operating in the same channel, and allows only one AP to be active at any given time. As a result, its throughput stays constant even as the number of APs increases. This throughput might vary with the number of APs in a real 802.11 network due to increased contention; however, since USRPs don't have carrier sense, we compute 802.11 throughput by providing each client with an equal share of the medium. In contrast, with MegaMIMO, as we add more APs, MegaMIMO can use these APs to transmit concurrent packets to more receivers. As a result, we see that the throughput of MegaMIMO increases linearly with the number of APs.
- The absolute gains provided by MegaMIMO are higher at high ($\sim$9.4$\times$ for 10 APs) and medium ($\sim$9.1$\times$) SNRs, than at low SNRs ($\sim$8.1$\times$). This is a consequence of the theoretically predicted throughput of beamforming. In particular, the beamforming throughput with N APs scales as $N\log(\frac{\text{SNR}}{K}) = N\log(\text{SNR}) - N\log(K)$, where K depends on the channel matrix $\mathbf{H}$ and is related to how well conditioned it is [39]. Natural channel matrices can be considered random and well conditioned, and hence K can essentially be treated as constant for our purposes. The 802.11 throughput scales roughly as $\log(\text{SNR})$ [39]. The expected gain of MegaMIMO over 802.11 can therefore be written as $N(1 - \frac{\log(K)}{\log(\text{SNR})})$ and hence becomes closer to N as SNR increases. This is why, MegaMIMO's gains at lower SNR grow at a lower rate than the gains at high SNR.

11.3 Fairness

In this experiment, we verify if MegaMIMO is fair, *i.e.*, it delivers the above throughput gains to all nodes.

Method. We perform the same experiment as in §11.2. We then compute the throughput gain of each node as the ratio of its throughput with MegaMIMO to its throughput with 802.11. As before, we perform this experiment varying the number of APs from 2 to 10, and across the full range of SNRs.

Results. Figs. 10(a), (b), and (c) plot the CDF of the throughput gain for 2, 6 and 10 APs at high, medium, and low SNRs. The results show that MegaMIMO is fair *i.e.* all nodes see roughly the same throughput gains, and these match the gains in total throughput shown in §11.2. Note that the CDF is wider at lower SNR. This is a consequence of greater measurement noise at low SNR causing larger throughput differences between clients.

11.4 Diversity

As described in §8, in addition to providing multiplexing gains, MegaMIMO can also provide throughput gains through diversity. In this section, we investigate MegaMIMO's diversity gains.

Method. We place several APs in random AP locations in the testbed, and one node at a client location, ensuring it has roughly similar SNRs to all APs. We then compute the throughput with regular 802.11 and MegaMIMO. We repeat the experiment with the number of APs varying from 2 to 10, and plot the results for the range of operational SNRs of 802.11.

Results. Fig. 11 shows throughput of 802.11 and MegaMIMO as a function of SNR for 2, 4, 6, 8 and 10 APs. Note that MegaMIMO provides significant gains over 802.11, especially at low SNRs. For instance, a client that has 0 dB channels to all APs (*i.e.* its received power from each AP is about the same as the noise) cannot get any throughput with 802.11. However the figure shows that, with 10 APs, such a client can achieve a throughput of 21 Mbps with MegaMIMO. Thus, using MegaMIMO for diversity can significantly expand the coverage range of an 802.11 deployment, and alleviate dead spots. This is expected because with MegaMIMO's coherent diversity, using APs to coherent combine the signal can provide a multiplicative increase in the SNR of N^2 [39]. This results in significant throughput improvements in the low SNR regime.

11.5 Compatibility with 802.11

Finally, as described in §6, MegaMIMO is compatible with existing 802.11n cards. In this section, we investigate whether MegaMIMO can deliver significant throughput gains when used with commodity 802.11n cards. Further, since each AP and each 802.11n

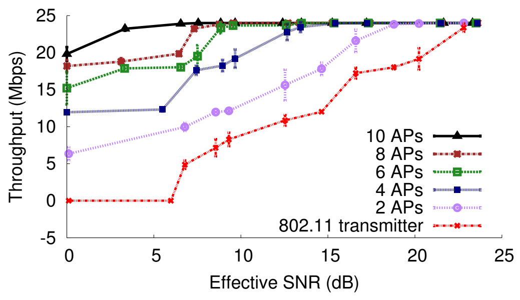


Figure 11: **Diversity Throughput.** Throughput of a MegaMIMO client when using diversity with 2, 4, 6, 8 and 10 APs. MegaMIMO can achieve close to maximum rate even to a client unable to receive any packets with 802.11.

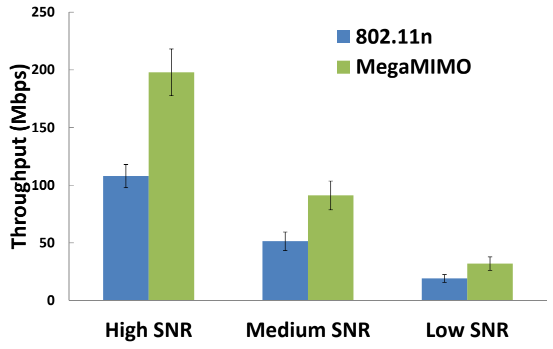


Figure 12: **Throughput achieved using MegaMIMO on off-the-shelf 802.11n cards.** MegaMIMO significantly improves the performance of off-the-shelf 802.11n cards at high (>18 dB), medium (12-18 dB) and low (6-12 dB) SNRs.

card in this system has 2 antennas, this experiment also verifies that MegaMIMO can provide its expected gains with multi-antenna transmitters and receivers.

Method. We place 2 MegaMIMO nodes at random AP locations in the testbed, and 2 802.11n receivers at random client locations in the testbed. For each topology, we compute the total throughput with 802.11n and with MegaMIMO. As before, we compute the 802.11n throughput by giving each transmitter an equal share of the medium. We repeat the experiment across multiple topologies and the entire range of SNRs.

Results. Fig. 12 shows the total throughput with and without MegaMIMOat high, medium and low SNRs. Since we have two receivers in this experiment, the theoretically throughput gain compared to 802.11n is 2×. The chart shows that MegaMIMO delivers

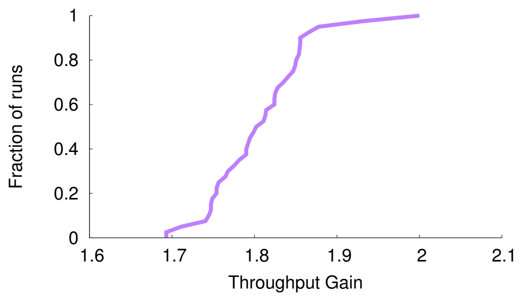


Figure 13: **Fairness Results.** For all nodes in our testbed, MegaMIMO delivers a throughput gain between 1.65-2×, with a median gain of 1.8× across SNRs. This shows that MegaMIMO provides similar throughput gains for every node in the network.

an average gain of 1.67-1.83× across all SNR ranges. Similar to the case with USRP receivers, the gains in the high SNR regime are larger than gains in the low SNR regime.

We now investigate MegaMIMO's fairness, *i.e.* whether MegaMIMO can deliver its throughput gains for every receiver in the network across all locations and SNRs. Fig. 13 shows the CDF of the throughput gain achieved by MegaMIMO as compared to 802.11n across all the runs. The results show that MegaMIMO delivers throughput gains between 1.65-2× for all the receivers and hence is fair to the receivers in the network.

12. CONCLUSION

This paper enables joint beamforming from distributed independent transmitters. The key challenge in delivering this system is to perform accurate phase synchronization across multiple distributed transmitters. The lessons learnt from building the system and testing it with real hardware are : 1) Estimates of frequency offset can be made accurate enough to predict (and hence correct) phase misalignment within an 802.11 packet; however, these estimates cannot be used across multiple packets due to large build-ups in phase errors over time; and 2) Joint multi-user beamforming can be achieved by synchronizing the phases of all senders to one lead sender, and does not impose any phase synchronization constraints on the receivers.

We believe that the design of MegaMIMO has wider implications than explored in this paper. In particular, several areas of information theory like lattice coding, noisy network coding, and transmitter cooperation for cognitive networks [26, 18, 23] assume tight phase synchronization across transmitters. We are optimistic that the algorithms presented in this paper can bring these ideas closer to practice.

Acknowledgments: We thank Dan Halperin for answering various questions about CSI feedback from the Intel cards, and Shyam Gollakota and Nate Kushman for their helpful comments on the paper. This work is funded by NSF.

13. REFERENCES

[1] IEEE standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: Enhancements for higher throughput. *IEEE Std 802.11n-2009 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, and IEEE Std 802.11w-2009)*, pages c1 –502, 29 2009.

[2] S. Aeron and V. Saligrama. Wireless ad hoc networks: Strategies and scaling laws for the fixed SNR regime. *IEEE Transactions on Inf. Theor.*, 53(6), 2007.

[3] E. Aryafar, N. Anand, T. Salonidis, and E. Knightly. Design and experimental evaluation of multi-user beamforming in wireless LANs. In *Mobicom 2010*.

[4] S. Berger and A. Wittneben. Carrier phase synchronization of multiple distributed nodes in a wireless network. In *8th IEEE Workshop on Signal Processing Advances for Wireless Communications (SPAWC), Helsinki, Finland*, June 2007.

[5] Distributed Antenna Systems. http://medicalconnectivity.com/2008/02/05/distributed-antenna-systems-no-replacement-for-wireless-strategy.

[6] USRP. `http://www.ettus.com`. Ettus Inc.

[7] A. Forenza, R. W. H. Jr., and S. G. Perlman. *System and Method For Distributed Input-Distributed Output Wireless Communications*. U.S. Patent Application number 20090067402.

[8] System for increasing capacity in future mobile communications networks. `http://www.hhi.fraunhofer.de/fileadmin/hhi/downloads/BM/PR_Demonstration_Network_MIMO.pdf`. Fraunhofer Heinrich Hertz Institute.

[9] A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[10] S. Gollakota, S. Perli, and D. Katabi. Interference alignment and cancellation. *ACM SIGCOMM*, 2009.

[11] Greentouch consortium. `https://www.youtube.com/watch?v=U3euDDr0uvo`. GreenTouch Demonstrates Large-Scale Antenna.

[12] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Tool release: gathering 802.11n traces with channel state information. *SIGCOMM Comput. Commun. Rev.*, 41:53–53.

[13] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. In *ACM SIGCOMM*, 2010.

[14] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller. Maranello: practical partial packet recovery for 802.11. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 14–14, Berkeley, CA, USA, 2010. USENIX Association.

[15] J. Heiskala and J. Terry. *OFDM Wireless LANs: A Theoretical & Practical Guide*. Sams Publishing, 2001.

[16] The iPad and its impact on hotel owners and operators. `http://www.ibahn.com/en-us/public/docs/The_Impact_of_iPad.pdf`. iBAHN.

[17] D. C. Jenn, J. H. Ryu, T. Yen-Chang, and R. Broadston. Adaptive phase synchronization in distributed digital arrays. In *Adaptive Hardware and Systems (AHS), 2010 NASA/ESA Conference on*, pages 199 –204, june 2010.

[18] S. Lim, Y. Kim, A. El Gamal, and S. Chung. Noisy network coding. In *IEEE Info. Theor. Workshop*, 2010.

[19] K. C.-J. Lin, S. Gollakota, and D. Katabi. Random access heterogeneous mimo networks. In *Proceedings of the ACM SIGCOMM 2011 conference*, SIGCOMM '11, pages 146–157, New York, NY, USA, 2011. ACM.

[20] K. C.-J. Lin, N. Kushman, and D. Katabi. ZipTx: Harnessing Partial Packets in 802.11 Networks.

[21] LTE: MIMO techniques in 3GPP-LTE. `http://lteportal.com/Files/MarketSpace/Download/130_LTEMIMOTechniquesFreescaleNov52008.pdf`.

[22] Z. Ma, M. Zierdt, J. Pastalan, A. Siegel, T. Sizer, A. J. de Lind van Wijngaarden, P. R. Kasireddy, and D. M. Samardzija. Radiostar: Providing wireless coverage over gigabit ethernet. *Bell Lab. Tech. J.*, 14:7–24, May 2009.

[23] I. Maric, N. Liu, and A. Goldsmith. Encoding against an interferer's codebook. In *Allerton*, 2008.

[24] MIMO schemes in 16m (WiMAX 2.0). `http://www.wimax360.com/profiles/blogs/mimo-schemes-in-16m-wimax-20`. WiMax360.

[25] A. Miu, H. Balakrishnan, and C. E. Koksal. Improving loss resilience with multi-radio diversity in wireless networks. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, 2005.

[26] B. Nazer and M. Gastpar. The case for structured random codes in network capacity theorems. *European Transactions on Telecommunications*, 19(4), 2008.

[27] Network MIMO. `http://www.alcatel-lucent.com/wps/DocumentStreamerServlet?LMSG_CABINET=Docs_and_Resource_Ctr&LMSG_CONTENT_FILE=Data_Sheets/Network_MIMO.pdf`. Alcatel-Lucent.

[28] A. Ozgur, O. Leveque, and D. Tse. Hierarchical cooperation achieves optimal capacity scaling in ad hoc networks. *IEEE Trans. on Info. Theor.*, 2007.

[29] H. Rahul, F. Edalat, D. Katabi, and C. Sodini. Frequency-Aware Rate Adaptation and MAC Protocols. In *ACM MOBICOM 2009*, Beijing, China, September 2009.

[30] H. Rahul, H. Hassanieh, and D. Katabi. Sourcesync: a distributed wireless architecture for exploiting sender diversity. *SIGCOMM*, 2010.

[31] Distributed-input distributed-output wireless technology. `http://www.rearden.com/DIDO/DIDO_White_Paper_110727.pdf`. Rearden Companies.

[32] Mobile broadband capacity constraints and the need for optimization. `http://rysavy.com/Articles/2010_02_Rysavy_Mobile_Broadband_Capacity_Constraints.pdf`. Rysavy Research.

[33] L. Shen-fa and W. Wei-ling. Fast antenna selection algorithms for distributed mimo systems. In *Journal of Beijing University of Posts and Telecommunication*, volume 30, pages 50–53, Jun. 2007.

[34] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra. Centaur: realizing the full potential of centralized wlans through a hybrid data path. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, pages 297–308, New York, NY, USA, 2009. ACM.

[35] O. Simeone, O. Somekh, H. Poor, and S. Shamai. Distributed MIMO in multi-cell wireless systems via finite-capacity links. In *ISCCSP*, 2008.

[36] K. Tan, J. Fang, Y. Zhang, S. Chen, L. Shi, J. Zhang, and Y. Zhang. Fine-grained channel access in wireless lan. In *ACM SIGCOMM 2010*.

[37] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. M. Voelker. SAM: enabling practical spatial multiple access in wireless LAN. In *MobiCom 2009*.

[38] I. Thibault, G. Corazza, and L. Deambrogio. Phase synchronization algorithms for distributed beamforming with time varying channels in wireless sensor networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, pages 77 –82, july 2011.

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

[40] Turn off your wi-fi network! `http://www.youtube.com/watch?v=fFiJ5rnIPVw`. Steve Jobs iPhone4 Keynote.

[41] S. Venkatesan et al. A WiMAX-based implementation of network MIMO for indoor wireless. *EURASIP*, '09.

[42] Z. Xin-sheng, Y. Xiao-hu, and Z. Ding-qian. An investigation on dynamic rau selection method for distributed radio mobile communications system. In *Journal of Electronics ïijĘ Information Technology*, volume 28, pages 2334–2338, Dec. 2006.

[43] L. Ying-Dong and Z. Guang-Xi. Transmit-receive antenna selection for distributed multi-user mimo downlink channels. In *Information Science and Engineering (ICISE), 2009 1st International Conference on*, pages 2767 –2770, dec. 2009.

APPENDIX

A. DECOUPLING MEASUREMENTS TO DIFFERENT RECEIVERS.

For simplicity, we focus on the example of two APs and two clients in Fig. 2. Let us consider a system where the channels, h_{11} and h_{12}, to receiver 1 are measured at time t_1 and the channels, h_{21} and h_{22}, to receiver 2 are measured at time t_2. For a subsequent transmission at time t, the channels experienced to receiver 1 experience a rotation corresponding to the time $t - t_1$, while the channels experienced to receiver 2 experience a rotation corresponding to time $t - t_2$. In particular, the channel matrix experienced at time t can be written as:

$$\mathbf{H}(\mathbf{t}) = \begin{pmatrix} h_{11}e^{j(\omega_{R1}-\omega_{T1})(t-t_1)} & h_{12}e^{j(\omega_{R1}-\omega_{T2})(t-t_1)} \\ h_{21}e^{j(\omega_{R2}-\omega_{T1})(t-t_2)} & h_{22}e^{j(\omega_{R2}-\omega_{T2})(t-t_2)} \end{pmatrix} \tag{6}$$

Recall that for MegaMIMO to perform distributed phase synchronization, we need to decompose $\mathbf{H}(\mathbf{t})$ into the form $\mathbf{R}(\mathbf{t})\mathbf{H}\mathbf{T}(\mathbf{t})$ where $\mathbf{H}$ is time-invariant, and the time-dependent matrices $\mathbf{R}(\mathbf{t})$ and $\mathbf{T}(\mathbf{t})$ are diagonal, and the i^{th} diagonal entry of $\mathbf{T}(\mathbf{t})$ (similarly) depends only on parameters that the i^{th} AP (similarly i^{th} receiver) can estimate locally. The APs can then all use the time invariant matrix $\mathbf{H}$ to calculate their beamforming signal, and perform correction using the relevant entry of $\mathbf{T}(\mathbf{t})$.

We observe that $\mathbf{H}(\mathbf{t})$ can indeed be written in this desired form. Specifically, we can write $\mathbf{H}(\mathbf{t})$ as $\mathbf{R}(\mathbf{t})\mathbf{H}\mathbf{T}(\mathbf{t})$, where

$$\mathbf{R}(\mathbf{t}) = \begin{pmatrix} e^{j(\omega_{R1}-\omega_{T1})(t-t_1)} & 0 \\ 0 & e^{j(\omega_{R2}-\omega_{T1})(t-t_2)} \end{pmatrix} \tag{7}$$

$$\mathbf{H} = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22}e^{-j(\omega_{T1}-\omega_{T2})(t_2-t_1)} \end{pmatrix} \tag{8}$$

$$\mathbf{T}(\mathbf{t}) = \begin{pmatrix} 1 & 0 \\ 0 & e^{j(\omega_{T1}-\omega_{T2})(t-t_1)} \end{pmatrix} \tag{9}$$

Note that $\mathbf{H}$ is now time invariant as desired. The entries only depend on the oscillator offset between times t_1 and t_2. Slave AP i can easily compute this offset by using the reference channel, h_i^{lead} from the lead measured at time t_1 and t_2.

Further, note that the diagonal entries of the matrix $\mathbf{T}(\mathbf{t})$ only depend on the frequency offset of the corresponding slave AP from the lead AP, and hence each slave AP can, as before, observe the channel of the sync header, compute the oscillator offset using the channel measured at time t_1 as reference, and correct its transmission appropriately.

Similarly the diagonal entries of the matrix $\mathbf{R}(\mathbf{t})$ only depend on the frequency offset of the corresponding receiver from the lead AP, and hence each receiver can independently decode its packet as if it were sent from a single transmitter.

Intuitively, this scheme can be understood as the slave AP rotating its measured channel to receiver 2 back to the time t_1 by multiplying h_{22} by $e^{-j(\omega_{T1}-\omega_{T2})(t_2-t_1)}$, and then performing all future channel corrections relative to the time t_1. This is why it corrects by the time dependent quantity $e^{-j(\omega_{T1}-\omega_{T2})(t-t_1)}$ shown in $\mathbf{T}(\mathbf{t})$.