This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

Employing the One-Sender-Multiple-Receiver Technique in Wireless LANs

Zhenghao Zhang, Steven Bronson, Jin Xie and Hu Wei
Computer Science Department
Florida State University Tallahassee, FL 32306, USA

***Abstract*—In this paper, we study the One-Sender-Multiple-Receiver (OSMR) transmission technique, which allows one sender to send to multiple receivers simultaneously by utilizing multiple antennas at the sender. We implemented a prototype OSMR transmitter/receiver with GNU Software Defined Radio, and conducted experiments in a university building to study the physical layer characteristics of OSMR. Our results are positive and show that wireless channels allow OSMR for a significant percentage of the time. Motivated by our physical layer study, we propose extensions to the 802.11 MAC protocol to support OSMR transmission, which is backward compatible with existing 802.11 devices. We also note that the AP needs a packet scheduling algorithm to efficiently exploit OSMR. We show that the scheduling problem without considering the packet transmission overhead can be formalized as a Linear Programming problem, but the scheduling problem considering the overhead is NP-hard. We then propose a greedy algorithm to schedule OSMR transmissions. We tested the proposed protocol and algorithm with simulations driven by traffic traces collected from wireless LANs and channel state traces collected from our experiments, and the results show that OSMR significantly improves the downlink performance.**

***Index Terms*—One-Sender-Multiple-Receiver, Packet Scheduling, Wireless LAN.**

I. Introduction

Wireless Local Area Networks (LAN) offer convenient access to the Internet. As new applications such as Internet TV are demanding more and more bandwidth, wireless LAN has attracted much attention in both the academia and the industry. In this paper, we study the One-Sender-Multiple-Receiver (OSMR) technique and its applications to wireless LANs. OSMR allows one sender to send distinct information to multiple receivers simultaneously on the same frequency by utilizing multiple antennas at the sender [2]. In wireless LANs, the Access Point (AP) may use OSMR to support multiple nodes more efficiently.

OSMR is basically *Multi-User Multiple-Input-Multiple-Output (MU-MIMO)* on the downlink [2], [3], [4]. As MU-MIMO also includes the uplink case, in this paper, we use the term OSMR for clarity. MU-MIMO is an active research field in the wireless communication community. However, most of the existing research focus on theoretical signal processing and often assume simplified network models, e.g., the availability of a feedback channel for channel state update, homogeneous and constant traffic load among nodes [3], [4]. In wireless LANs, the major challenge is that the traffic load of nodes are random and heterogeneous, which makes the scheduling problem very challenging. OSMR is different from the single-user MIMO transmission to be adopted in 802.11n [14], because single-user MIMO still supports only one-to-one transmissions, while OSMR supports one-to-many transmissions. Allowing one-to-many transmissions will help achieving an overall higher efficiency. For example, suppose node A and B both have very strong channels and are already operating at the highest data rate supported by the hardware. With OSMR, the AP may be able to transmit to them simultaneously, still at the highest data rate, hence doubling the downlink throughput. Code Division Multiplexing Access (CDMA) also allows multiple nodes to communicate simultaneously on the same frequency. Basically, OSMR takes advantage of multiple antennas and is more efficient in utilizing the bandwidth than CDMA. A CDMA transmitter has to spread the signal bandwidth to a much larger bandwidth [2], which is not required in OSMR.

In this paper, we focus on the case when OSMR involves two receiving nodes, which represents a significant percentage of practical cases because the AP usually has a limited number of antennas due to cost considerations. To study the physical layer characteristics and the practicability of OSMR for wireless LANs, we implemented a prototype OSMR transmitter/receiver with GNU Software Defined Radio (SDR) [12], [13] that allows one sender to send to two receivers simultaneously. OSMR transmission depends on the channel states of the receivers, because it requires the sender to process the signals according to the channel states. The most critical questions related to the practicability of OSMR include (1) how likely are two receivers *compatible*, where compatible means that the channel states of two receivers allow them to receive from the sender simultaneously at non-trivial data rates, and (2) whether the channel fluctuation speed is slow enough, such that the measured channel states remain valid until the sender finishes sending. Fortunately, our experiments reveal that two receivers are usually compatible for a significant percentage of the time. Also, in the indoor environment, the channel is typically stable for the OSMR transmission.

Motived by our physical layer study, we propose OSMR as an enhancement technique to wireless LANs. We propose a simple extension to the 802.11 MAC protocol to support OSMR transmissions. The extension is backward compatible, i.e., it allows the OSMR-capable nodes to coexist with legacy 802.11 nodes without interfering with each other. We also note that to fully exploit OSMR, the AP needs an algorithm to decide which packet(s) to send in order to optimize the performance, e.g., maximizing the throughput under the fairness and quality of service constraints mandated by the upper layer. We show that the scheduling problem without considering the packet transmission overhead can be formalized as a

978-1-4244-5837-0/10/$26.00 ©2010 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

Linear Programming problem, but the scheduling problem considering the overhead is NP-hard. We then propose a greedy algorithm to schedule OSMR transmissions. We tested the proposed protocol and algorithm with simulations driven by traffic traces collected from wireless LANs and channel state traces collected from our experiments, and the results show that OSMR significantly improves the downlink performance.

The rest of the paper is organized as follows. Section II describes our implementation of OSMR and the experiments. Section III discusses the extension to the 802.11 MAC protocol. Section IV describes the packet scheduling algorithm. Section V evaluates the proposed protocol and the packet scheduling algorithm with simulations. Section VI discusses the related works. Section VII concludes the paper.

II. OSMR Implementation and Experiments

In this section, we describe our implementation of OSMR and the experiments.

A. OSMR Background and Implementation

OSMR can be realized by a *zero-forcing* strategy [2], as explained in the following. We assume the channel is flat-fading. If the sender is sending a data symbol d, the receiver will receive $y = hd + n$, where h is the complex channel coefficient and n is the noise. If there are two receivers and the sender has two antennas, the sender can send two different symbols denoted as x_1 and x_2 on antenna 1 and antenna 2, respectively. Suppose the channel coefficient from antenna j to receiver i is h_{ij} for $i, j \in \{1, 2\}$. Let the received signal at receiver i be y_i, and let the noise at receiver i be n_i. The received signal is a linear combination of the signals sent from each antenna multiplied by the channel coefficients, plus the noise:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} n_1 \\ n_2 \end{pmatrix}$$

We will use $\mathbf{H}$ to denote the channel matrix. Suppose a *processing matrix*

$$\mathbf{U} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}$$

can be found such that $\mathbf{h}_1\mathbf{u}_2 = 0$ and $\mathbf{h}_2\mathbf{u}_1 = 0$, where $\mathbf{h}_i$ denotes a row vector of $\mathbf{H}$ and $\mathbf{u}_j$ denotes a column vector of $\mathbf{U}$. If such matrix can be found, let d_1 and d_2 denote the data that should be sent to receiver 1 and receiver 2, respectively. We let

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$$

Thus, receiver 1 will receive $\mathbf{h}_1(d_1\mathbf{u}_1 + d_2\mathbf{u}_2) + n_1 = d_1\mathbf{h}_1\mathbf{u}_1 + n_1$. Similarly, receiver 2 will receive $d_2\mathbf{h}_2\mathbf{u}_2 + n_2$. Therefore, distinct data is sent to each receiver. In this paper, $\mathbf{h}_i\mathbf{u}_i$ is referred to as the *effective channel* of receiver i. Two receivers are *compatible* if a processing matrix can be found such that the strength of their effective channels are above certain thresholds.

We implemented a prototype OSMR transmitter/receiver based on the above zero-forcing strategy in about 2,000 lines of C++ and Python code using GNU Software Defined Radio (SDR) [12], [13]. In our implementation, $\mathbf{U}$ is chosen such that the strength of effective channel is comparable to the original channel. Also, as the sender needs to know the channel matrix to process the data, we designed a channel estimation method. Basically, the sender sends a *channel estimation frame* to the receiver, with which the receiver calculates its channel state and informs the sender with a *channel estimation report*. The details of the selection of $\mathbf{U}$ and the channel estimation method used in our SDR implementation can be found in Section 3.3 of [22].

B. OSMR Experiments

We used our prototype OSMR transmitter/receiver to find the feasibility of OSMR. The first key question we seek to answer is: How likely are two receivers compatible? Because the wireless channel is constantly fluctuating, two receivers may be compatible at some times while not compatible at other times. For OSMR to be applicable to wireless LANs, the percentage of the time when the receivers are compatible must be non-trivial. In our experiments, the OSMR transmission is centered at 2.42GHz, which lies within the ISM band used by the 802.11g networks. Differential Binary Phase Shift Keying (DBPSK) modulation is used and the symbol rate is 500,000 symbols per second, which results in a bit rate of 0.5Mbps. We refer to the OSMR sender as S and the two OSMR receivers as $R1$ and $R2$, respectively. In our experiments, $R1$ and $R2$ are turned on first. The OSMR transmission is then carried out in three steps:

1) S transmits the channel estimation frames for 0.5 second, then switches to the listening mode to wait for the channel estimation reports from $R1$ and $R2$.
2) Both $R1$ and $R2$ wait until S stops sending. Then, $R1$ sends the channel estimation report to S for 0.01 second, then switches to the listening mode to wait for the data frames. After S stops sending, $R2$ waits for 0.01 second, then sends the channel estimation report to S for 0.01 second, then switches to the listening mode to wait for the data frames.
3) After getting both channel estimation reports, S waits for 0.01 second, then switches to the transmitting mode and sends the data frames for 1 second. One data frame is 1524 bytes with 1500 bytes of randomly generated data and 24 bytes as the preamble and the frame header.

Our experiments were conducted in a university building. The devices used in our experiments are shown in Fig.1. We picked 10 sender locations, and for each sender location, we conducted a set of 4 OSMR experiments at randomly selected receiver locations, where the distances between the sender and the receivers were between 6 to 30 feet. The sender location and the receiver locations in one set of the experiments, for example, are shown in Fig. 2. In each experiment, OSMR transmissions were attempted with random intervals between 2 to 5 seconds. Therefore, we basically randomly sample the channels and find the percentage of time the receivers are compatible. An OSMR transmission is considered successful if the both receivers get the first 3 data frames with no bit error. Only the first 3 frames were considered because the sender processes the signal based on the channel estimation

Fig. 1. Devices used in the experiments.

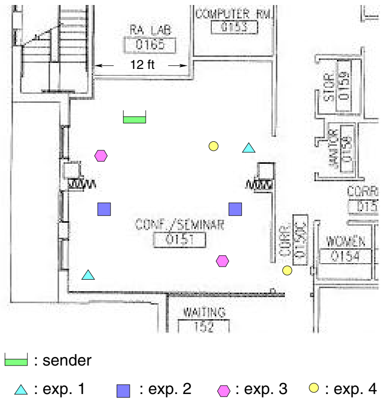


Fig. 2. The sender location and the receiver locations in one set of the experiments.

reports received before the transmission, but the channel states may have drifted after several frames, which leads to receiving errors that should not be interpreted as incompatibility. As a quantitative measure, the *compatibility ratio* is defined as the number of successful OSMR transmissions over the number of all OSMR transmissions carried out, where an OSMR transmission is carried out if the sender gets both channel estimation reports and sends the data frames. Not all attempted OSMR transmissions were carried out, because with the current GNU SDR, the switching between the transmitting and the receiving mode could take a non-trivial amount of time depending on the instantaneous state of the operating system. It could happen that two receivers send report at the same time, which results in a collision. If the sender did not get the channel estimation reports from both receivers, the sender will abort the transmission. We report the results of 35 experiments in which at least 25 OSMR transmissions were carried out and show the cumulative distribution function (c.d.f.) of the compatible ratio in Fig. 3. We can see that roughly, the compatible ratio is uniformly distributed in $[0, 0.9]$.

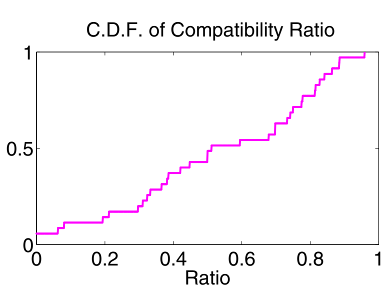
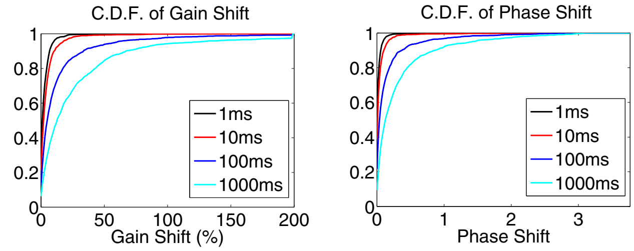


Fig. 3. The c.d.f. of the compatibility ratio found in the experiments.



(a) (b)

Fig. 4. The c.d.f. of channel ratio shift. (a). Magnitude. (b). Phase (in radians).

C. Wireless Channel Characteristics Relevant to OSMR

As mentioned earlier, the second key question is the stability of the channel. As the wireless channel may fluctuate randomly, before starting the OSMR transmission, the sender should have the up-to-date the channel states from the receivers to calculate the processing matrix. Because the sender does not have further feedbacks from the receiver, in order for the OSMR transmission to be successful, the shift of the channel during the frame transmission time must be limited. Therefore, we conducted experiments to find the channel characteristics in the indoor environments. In our experiments, there are one sender and one receiver, where the sender has two antennas and the receiver has one antenna. We picked 10 sender locations, and for each sender location, 4 receiver locations were picked randomly with both line-of-sight and non-line-of-sight paths. The sender transmits the OSMR channel estimation sequence every 1ms for a total of 50 seconds, and the receiver simply records the received samples. In OSMR transmissions, the *channel ratio* of a receiver, which is defined as ratio of the channel coefficients from sender antenna 1 over antenna 2, is the crucial parameter in determining whether the interferences can be suppressed successfully. Due to the limit of space, we only show the fluctuation of the channel ratio. If the ratio is $ae^{j\phi}$ at time t_0 and is $a'e^{j\phi'}$ at time t_1, the shift of the magnitude is defined as $\frac{|a'-a|}{a} \times 100\%$, and the shift of phase is defined as $\mid \phi' - \phi \mid$. The c.d.f.s of the channel ratio shift after 1ms, 10ms, 100ms, and 1000ms are shown in Fig. 4. We can see that for more than 90% of the times, after 10ms, the magnitude shifts less than 10%, and phase shifts less than $\frac{\pi}{18}$. As an example, Fig.5 shows a typical trace of the channel ratio magnitude. The fast fluctuations at the beginning of the trace were caused by fast movements of human beings. The rest of trace are relatively stable.

D. Remarks

Our experiments proved that in the indoor environments, for a significant percentage of time, OSMR transmissions are possible and the wireless channels are stable. Yet, the experimental results are implementation dependent. The results reported in this section on OSMR transmissions are based on our prototype implementation with software defined radio. Software defined radio has limitations at current stage for the implementation of OSMR. First, it does not allow fast switching between the transmitting mode and receiving mode, and we have to let the sender wait for around 20ms before starting the transmission, because the receivers have

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

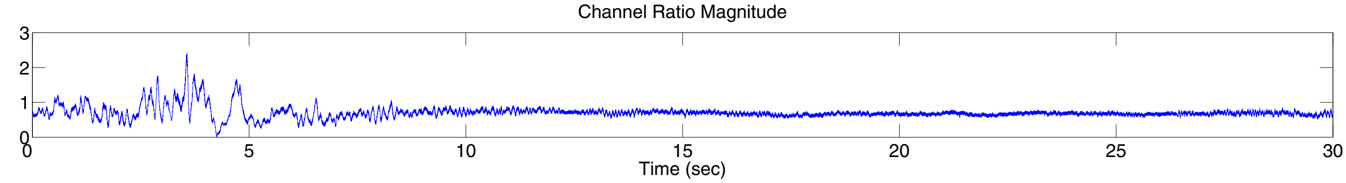


Fig. 5. A typical trace of the channel ratio magnitude.

to switch from the receiving mode to the transmitting mode. Therefore, the successful OSMR transmissions reported in our experiments belong to those cases when the channels allowed OSMR transmissions *and* did not shift significantly after the channel estimation, which is a subset of the cases when the channels allowed OSMR transmissions. In this regard, with a faster hardware implementation, the compatibility ratios may be higher than that in Fig.3. However, secondly, software defined radio relies on software to process the signals, and the data rate is not as high as hardware radios. When the data rate is higher, the Signal to Noise Ratio (SNR) requirements are higher, which may reduce the number of compatible pairs. As commercial hardware OSMR transmitters and receivers are not available yet, we will use the channel state traces collected from our experiments and simulations to study the high data rate regime in Section V. Note that the channel state traces were collected by simply recording the received samples and are not subject to the same limitations in the OSMR transmission experiments.

III. Extensions to 802.11 MAC

The 802.11 MAC protocol needs to be extended to support OSMR transmissions. We focus on the case when only the AP acts as the OSMR sender. This is motivated by the observation that the traffic in a wireless LAN is typically either from the AP to the nodes, or from the nodes to the AP [17]. Therefore, a node usually do not have to send to multiple other nodes. Nevertheless, the proposed protocol can also be readily extended to the case where the nodes can act as OSMR senders.

A. The OSMR Transmission Procedure

The AP competes for medium access according to the current 802.11 MAC, i.e., monitoring the medium and backoff for a random time if needed. Thus, the fairness in the 802.11 MAC is preserved. Once gained access to the medium, the AP may start an OSMR transmission. To get the channel states, we propose a procedure similar to our experiments. That is, if the AP gains access to the medium, it first broadcasts a channel estimation frame, then the involved nodes send back the channel estimation reports.

To be more specific, the AP first broadcasts a short control packet, referred to as the Channel Estimation Request (CRQ) packet. The format of the CRQ packet is shown in Fig.6. The AP determines a set of nodes whose channel states are needed, and lists their MAC addresses as well as a time offset for each node in the CRQ packet. The time offset is measured in μs and is used to determine the time for the node to send its response. At the end of the CRQ packet, a sequence of symbols is appended for channel estimation. If a node finds its MAC address listed in the CRQ packet, it will perform a

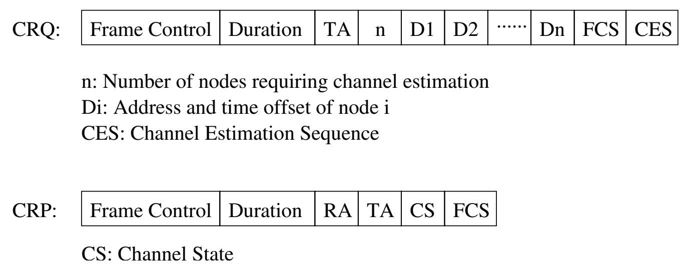


Fig. 6. CRQ and CRP packet format. TA: Transmitter Address. RA: Receiver Address. FCS: Frame Check Sequence.

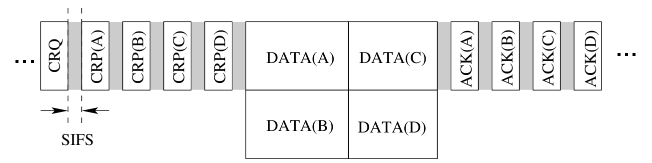


Fig. 7. Packet transmission with OSMR. CRQ: channel estimation request. CRP: channel estimation report.

channel estimation procedure based on the channel estimation sequence. A node should send a short control packet, referred to as the Channel Estimation Report (CRP) packet, back to the AP to report the measured channel state. The format of the CRP packet is also shown in Fig.6. The measured channel state is small and can be encoded in a few bytes. Note that no collision will occur because nodes send the CRP packets at times announced in the CRQ packet. SIFS after the AP gets the CRP packets, the AP starts the OSMR transmission. If the AP did not get the CRP packets from some nodes, the AP may adjust the OSMR transmission. For example, the AP may use OSMR to send only to nodes whose CRP packets are received. SIFS after the AP finishes its transmission, nodes that received packets should send acknowledgment (ACK) packets. Similar to sending the CRP packets, nodes should send the ACK packets according to the time offsets announced in the CRQ packet. The AP removes the data that has been acknowledged; other data will be scheduled for retransmission. The CRQ packet should be sent at a data rate such that all involved nodes can decode the packet with high probability. The CRP packet can be sent at the same data rate as the ACK packet.

For example, an OSMR transmission procedure is illustrated in Fig. 7, in which the AP is using OSMR to transmit first to nodes A and B and then to nodes C and D.

B. TXOP and Fragmentation

To improve the efficiency, we propose to allow the fragmentation of the packets. This is because the nodes may be at different data rates and the packet sizes may be different, such that it is unlikely that the AP can find packets to two nodes as a pair that occupy exactly the same amount of time. We note that packet fragmentation is an available option in the current 802.11. With packet fragmentation, the transmission time of the AP is no longer restricted by the length of the packets. From the AP's point of view, it would like to occupy the medium for as long as possible. However, this may cause unfairnesses to other nodes in the network; in addition, the channel estimation may become outdated. Therefore, the AP should transmit for no longer than a threshold. We assume that once the AP gains access to the medium, it transmits for no more than γ seconds, where γ is a system parameter. This is the concept of Transmission Opportunity (TXOP) defined in 802.11e, where a node can keep transmitting for a TXOP length [19]. Both the AP and the non-AP nodes can exploit the TXOP to send multiple packets, although only the AP sends the packets with OSMR. The length of the TXOP defined in 802.11e is several ms, e.g., 1.5ms or 3ms, and similar values can be used for the TXOP with OSMR.

Note that within one TXOP, the AP may send to multiple nodes with multiple OSMR transmissions, as well as sending to some nodes without OSMR. The transmission schedule in a TXOP can be represented as a list of four-tuples. A four-tuple can be $[(i,j),(x_{i,j},x_{j,i})]$, which means that the AP should send to node i and j simultaneously using OSMR for $x_{i,j}$ and $x_{j,i}$ bytes, respectively. The schedule could also have four-tuples such as $[(i,-),(x_i,-)]$, which means that the AP should send to node i without OSMR for x_i bytes. Each four-tuple in the list is called a *sub-transmission*. The number of bytes sent to two nodes belonging to the same sub-transmission should be proportional to their data rates and may be different. The sub-transmissions do not have to be separated by SIFS, because only the AP is in the transmitting mode during the TXOP. The OSMR data transmission is in the same format as stand-alone packets, i.e., must be preceded by the preamble, the PLCP header, and the MAC header, and must be trailed by the FCS.

C. Piggybacking the Channel States

The AP needs to keep track of the compatibility relations of nodes in the network. To achieve this, the AP may piggyback the channel estimation sequence to every data and ACK packet it sent. A node can inform the AP about its channel state by piggybacking it with the ACK or the data packets. This will not introduce much overhead, because the channel estimation sequence is only a few symbols, and the channel estimation report can be packed into a few bytes. Note that if the traffic load is high, it can be expected that the AP may receive the channel estimation reports from the heavily loaded nodes in a timely manner. For the lightly loaded nodes, the need to optimize transmissions to them is not as critical.

D. Backward Compatibility

We note that the OSMR transmission process is completely backward compatible. This is because the AP will announce the duration of the OSMR transmission in its CRQ packet, and all nodes overheard the CRQ packet should backoff until the OSMR transmission finishes. In addition, all packet transmissions are separated by SIFS. Even if an 802.11 node did not hear the CRQ packet, it will not attempt to transmit because it has to wait the medium to be free for DIFS which is longer than SIFS. The AP may use OSMR transmissions only to OSMR-capable nodes. When sending to other nodes, the AP may simply use the one-to-one transmission. Also, the uplink is unchanged because only the AP uses OSMR. Therefore, the OSMR-capable nodes and the OSMR-incapable nodes can coexist in the same LAN without interfering with each other. Yet, the OSMR-capable nodes will receive better services from the AP.

TABLE 1
LIST OF NOTATIONS

γ	Length of TXOP
N	Number of nodes
B_i	Number of bytes buffered for node i
m_i	Number of bytes must be sent to node i
μ_i	Base date rate of node i
$\mu_{i,j}$	OSMR date rate of node i with node j

IV. Downlink Packet Scheduling

In this section, we focus on packet scheduling when OSMR is adopted. Packet scheduling is needed because the AP must make smart decisions to "pair up" packets to improve the overall downlink performance, such as the throughput. We assume the scheduler is given the number of bytes that must be sent to every node by the upper layer. The upper layer decides this based on the considerations of many issues, such as fairness and Quality of Service (QoS) requirements. For instance, when a node is running Voice Over IP (VoIP) applications, certain number of bytes must be sent to this node in a timely manner. The scheduler takes this input, and finds a schedule that meets the requirements of the upper layer while sends as many bytes as possible.

A. Definitions and Notations

We use μ_i to denote the data rate of node i without using OSMR, and call it the *base rate*. We use $\mu_{i,j}$ to denote the data rate of node i if the AP sends to nodes i and j simultaneously using OSMR, and call it the *OSMR rate* of node i with node j. Note that $0 \leq \mu_{i,j} \leq \mu_i$, because when not using OSMR, the AP is focusing all power to transmit to one node. We assume that the data rate is known to the AP and is stable for the scheduled transmission, because the AP can derive the data rate based on the channel state and the channel state does not change very fast. Indeed, the data rates of mobiles in the 3G networks are selected based on channel state feedbacks and vary every several ms [18]. We use B_i to denote the number of bytes in the buffer for node i, and use m_i to denote the number of bytes that must be sent to node i in this TXOP given by the upper layer. For convenience, we also refer to the m_i bytes that must be sent to node i the "urgent bytes", and other bytes the "non-urgent bytes." The number of nodes is denoted as N. Table 1 lists the notations.

B. The Ideal Scheduler

We begin by considering an ideal case in which only the data bytes are sent and the overhead such as MAC header can be neglected. We use x_i to denote the number of bytes sent to node i without OSMR, and $x_{i,j}$ to denote the number of bytes sent to node i using OSMR with node j. We show that

Theorem 1: The optimal schedule is the solution to the following Linear Programming problem:

$$\max \sum_{i=1}^{N} x_i + \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} x_{i,j} \quad (1)$$

subject to

$$x_i + \sum_{j=1, j\neq i}^{N} x_{i,j} \leq B_i, \text{for all } i \quad (2)$$

$$x_i + \sum_{j=1, j\neq i}^{N} x_{i,j} \geq m_i, \text{for all } i \quad (3)$$

$$\frac{x_{i,j}}{\mu_{i,j}} - \frac{x_{j,i}}{\mu_{j,i}} = 0, \text{for all } i \neq j,\ \mu_{i,j}, \mu_{j,i} > 0 \quad (4)$$

$$\sum_{i=1}^{N} \frac{x_i}{\mu_i} + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \frac{x_{i,j}}{\mu_{i,j}} \leq \gamma \quad (5)$$

Proof: Basically, term 1 is the total number of bytes that are sent and should be maximized. Constraint 2 states that the number of bytes sent to node i cannot be more than the total number of bytes stored in the buffer for node i. Constraint 3 states that the number of bytes sent to node i cannot be less than the total number of bytes that must be sent to node i. Constraint 4 states that if the AP sends to nodes i and j simultaneously, the time spent in sending to i must be the same as the time spent in sending to j. Constraint 5 states total amount of time must be no more than a TXOP length. ■

Note that if the LP is not feasible, the set of urgent bytes is not feasible, and the scheduler may send a feedback to the upper layer to recalculate the urgent bytes. In practice, to reduce the scheduling time, the upper layer may always issue urgent bytes that are guaranteed to be feasible. For example, it may make sure that the total time to send the urgent bytes at their base rates is no more than γ seconds.

C. Scheduling Considering the Overhead

The LP formulation provides theoretical insights and serves as an upper bound of the performance. However, it considers the ideal case, in which there is no overhead for a sub-transmission. In practice, the overhead of a sub-transmission includes the preamble, the PLCP header, the MAC header, etc., and can be more than 20μs in 802.11 a/g, and more than 192μs in 802.11b. The solution of the LP may consist of many short sub-transmissions and thus incur too much overhead. When considering the overhead, we define the optimal schedule as the schedule in which either (1) all buffered bytes are sent in a minimum time which is less than γ seconds, or (2) all urgent bytes are sent and as many non-urgent bytes are sent in γ seconds, when the overhead of a sub-transmission is at least β seconds. The OSMR Transmission With Overhead (OTWO) problem is defined as the problem to find an optimal schedule. We prove that

Theorem 2: The OTWO problem is NP-hard.

Proof: We reduce the Maximum Independent Set (MIS) problem to the OTWO problem. In a graph, a set of vertices is *independent* if no two vertices are adjacent to each other.

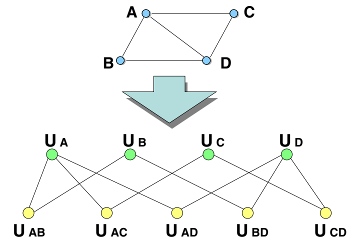


Fig. 8. The construction of the OTWO instance.

Given a graph G, we construct an instance of OTWO problem as follows. First, for any vertex A in G, create a "first level" node U_A. Second, for any edge in G, say, AB, create a "second level" node U_{AB}. U_{AB} is referred to as the "child node" of U_A. Note that U_{AB} is also a child node of U_B. In this sense, U_{AB} and U_{BA} refer to the same node in this construction. If there are N vertices and and E edges in G, a total of $N+E$ nodes are created. Fig.8 shows the construction of a simple instance. Let the base data rate of all nodes be r. For any first level node U_A and its child node U_{AB}, we let $\mu_{A,AB} = \frac{r}{d_A}$ and $\mu_{AB,A} = r$, where d_A is the degree of A in G. For any two nodes, if one is not the child node of the other, the OSMR rates of them with each other are both 0. Let $B_i = m_i = C$ for all $1 \leq i \leq N+E$, where C is a constant number. Note that since $B_i = m_i$, the optimal schedule is the schedule that uses minimum time to send the urgent bytes. C and r are chosen such that $\frac{C(N+E)}{r} < \beta$, that is, in this case, the overhead of a transmission is longer than the total transmission time of the data. We claim that the constructed OTWO instance has a schedule less than $(E+N-I+1)\beta$ seconds if and only if there exists an independent set of size at least I in G.

To see this, first note that if there is a schedule less than $(E+N-I+1)\beta$ seconds, the schedule has no more than $E+N-I$ sub-transmissions. Given such a schedule, we say a first level node U_A is "paired up" with its child node U_{AB}, if the schedule has a sub-transmission $[(U_A, U_{AB}), (x_{U_A,U_{AB}}, x_{U_{AB},U_A})]$. We say U_A is "fully paired up" with U_{AB} if $x_{U_A,U_{AB}} = \frac{C}{d_A}$ and $x_{U_{AB},U_A} = C$. If a first level node U_A is not fully paired with all its children, there must be a single sub-transmission in the schedule, denoted as $S_{U_A} = [(U_A, -), (x_{U_A}, -)]$, where $x_{U_A} < C$. In this case, we rearrange the schedule, and break all the pairs between U_A and its children in the original schedule. That is, we scan all sub-transmissions involving U_A. For sub-transmission $[(U_A, U_{AB}), (x_{U_A,U_{AB}}, x_{U_{AB},U_A})]$, we move the $x_{U_A,U_{AB}}$ bytes to S_{U_A}, and create a single sub-transmission for U_{AB}, denoted as $S_{U_{AB}} = [(U_{AB}, -), (x_{U_{AB}}, -)]$. This will not increase the total number of sub-transmissions. After that, for any newly created single sub-transmission for a second level node, say, $S_{U_{AB}}$ for U_{AB}, if $x_{U_{AB}} < C$, we again rearrange the schedule, and move all the bytes that should be sent to U_{AB} that are currently scheduled in other sub-transmissions to $S_{U_{AB}}$, breaking some pairs and create some new single sub-transmissions if necessary. Again, after this change, the total number of sub-transmissions is still no more than $E+N-I$. Finally, if multiple single sub-transmissions are created for a node, all such single sub-transmissions are merged into one. After this modification, the total number of sub-transmissions is still no more than $E+N-I$. However, a first level node is either fully paired with all its children, or is not paired with any children. Similarly, a second level node is either fully paired with one of its two parents, or not paired up with any parent.

Note that if two first level nodes correspond to two adjacent vertices in G, they cannot be both fully paired because they share a same child. Therefore, the set of fully paired nodes is an independent set in G. If a first level node with degree d is fully paired, only d transmissions are needed to send to $d+1$ nodes, including the first level node and its children. As each fully paired first level node can save one sub-transmission, if there are no more than $E+N-I$ sub-transmissions, there must be no less than I fully paired first level nodes. As a result, there must be an independent set of size at least I in G. Similarly, given an independent set of size I in G, we can find a schedule with $E+N-I$ sub-transmissions which needs less than $(E+N-I+1)\beta$ seconds. ∎

D. A Practical Scheduler

We propose a practical scheduler for OSMR transmissions. Our simulations show that it achieves close performance to the ideal LP scheduler, even when the sub-transmissions in the LP schedule have no overhead in the simulations.

1) A Greedy Algorithm: We propose a greedy algorithm to find the schedule, which runs in two phases. In short, in Phase 1, it considers only the urgent bytes, and exploits OSMR to find an efficient schedule with minimum time. After Phase 1, the partial schedule considering only the urgent bytes likely requires less than γ seconds, and the non-occupied time in this TXOP is referred to as the *available time*. In Phase 2, the algorithm considers all buffered bytes, and makes use of the available time and sends as many bytes as possible.

To elaborate, in every step in Phase 1, the algorithm searches for an OSMR transmission that reduces the maximum amount of time compared to sending the same bytes at the base rates. For two nodes i and j, let b be the maximum number of bytes that can be sent to node i using OSMR with node j. Clearly, $b=m_i$ if $\frac{m_i}{\mu_{i,j}}<\frac{m_j}{\mu_{j,i}}$, and $b=\frac{m_j\mu_{i,j}}{\mu_{j,i}}$ otherwise. The time to send to the nodes without OSMR is $\frac{b}{\mu_i}+\frac{b\mu_{j,i}}{\mu_j\mu_{i,j}}$, therefore, the saved time is

$$\frac{b}{\mu_i}+\frac{b\mu_{j,i}}{\mu_j\mu_{i,j}}-\frac{b}{\mu_{i,j}}.$$

After scheduling a sub-transmission $[(i,j),(b,\frac{b\mu_{j,i}}{\mu_{i,j}})]$, m_i and m_j should be updated accordingly. This is repeated until no OSMR transmissions can be found to save the transmission time.

In Phase 2, in every step, the algorithm searches for a transmission that is most efficient in utilizing the available time. To be more specific, the *efficiency* of a transmission is defined as the number of non-urgent bytes that can be sent in a unit time, and the algorithm always chooses the transmission with the highest efficiency. The transmission can be (1) a non-OSMR transmission to a node i for some non-urgent bytes, (2) an OSMR transmission to nodes i and j both for some non-urgent bytes, (3) an OSMR transmission to node i for some urgent bytes and to node j for some non-urgent bytes. Clearly, the efficiencies for the first two cases are μ_i and $\mu_{i,j}+\mu_{j,i}$, respectively. The efficiency of the third case is

$$\frac{\mu_i\mu_{j,i}}{\mu_i-\mu_{i,j}}.$$

To see this, note that this transmission consumes some available time *plus* the time that was scheduled to send to node i without OSMR. Suppose the number of bytes to node i is b when the consumed available time is one unit. Therefore, $\frac{b}{\mu_{i,j}}=1+\frac{b}{\mu_i}$. Hence, $b=\frac{\mu_i\mu_{i,j}}{\mu_i-\mu_{i,j}}$, and the total time of this OSMR transmission is $\frac{\mu_i}{\mu_i-\mu_{i,j}}$. By definition, $\frac{\mu_i\mu_{j,i}}{\mu_i-\mu_{i,j}}$ is the efficiency. After scheduling a transmission, m_i, B_i, and B_j should be updated if necessary. This is repeated until no available time is left, or until all buffered bytes are scheduled. Note that some scheduled transmissions in Phase 2 may be merged with some transmissions scheduled in Phase 1, if two transmission are to the same set of nodes.

Note that in Phase 1, the greedy algorithm will schedule one sub-transmission and reduce the number of urgent bytes of at least one node to 0 in every step; hence, it will not schedule more than N sub-transmissions in Phase 1. Similarly, in Phase 2, it will not schedule more than N sub-transmissions. Therefore, no more than $2N$ sub-transmissions will be scheduled. In each step in either Phase 1 or Phase 2, the scheduler needs $O(N^2)$ time. Therefore, the complexity is $O(N^3)$. Note that N is typically not very large in a wireless LAN.

2) Coping with Channel Fluctuation: As the example shown in Fig.5, the channel state of a node fluctuates with time, where the fluctuation may be faster at certain times than at other times. When the fluctuation is too fast, the processing matrix may become outdated during the transmission and the transmission will fail, because not all interferences can be canceled. To cope with this, the scheduler keeps track of the channel fluctuation speed of every node, and excludes a node from OSMR transmissions if its current fluctuation speed is above a threshold. The fluctuation speed can be estimated based on the channel state feedbacks from the node and the time when the feedbacks are received. Also, if the channel state has not been updated for longer than a threshold, the channel state may be outdated, and the node should be excluded from OSMR transmissions. Before an OSMR transmission, the AP runs the scheduler based on its current channel state records. After getting the CRP packets, the AP runs the scheduler again, because channel states of some nodes may have changed and the OSMR rates may need to be updated. However, because the AP schedules OSMR transmissions only to nodes with slow-varying channels, such update happens with low probability. We implemented these mechanisms in our simulations and the results show that they can effectively reduce packet loss.

V. Evaluations

To evaluate the proposed protocol and algorithm, we developed an event driven simulator. We relied on the simulator for performance evaluation, because the current GNU SDR does not support very accurate timing needed in the MAC protocol, and is operating at a lower data rate than hardware radios. Our simulation is driven by traffic traces collected from wireless

LANs [17] and the channel state traces collected from our experiments.

The simulator is set to be functioning as an 802.11g network and supports data rates 6,9,12,18,24,36,48,54 Mbps. The AP is at the center and the nodes are randomly located within a certain maximum distance to the AP. Based on the path loss model in [11] and the specifications of Cisco Aironet 802.11a/b/g wireless cardbus adapter [16], the average received signal strength is assumed to be $P_r = -31 - 30 \log d$ measured in dBm, where d is the distance between the sender and the receiver in meters. The channel state traces collected from our experiments with GNU SDR are amplified to have unit average power gain. The amplified traces preserve the channel fluctuation characteristics. For each node, a channel state trace is randomly selected and multiplied with the average signal strength between the node and the AP. The base data rate is determined based on the average receiving power strength and the specifications in [16]. The OSMR data rates are determined by the average receiving power of the effective channels; however, to account for channel fluctuation and channel estimation noise, an additional 7dBm margin is applied when determining the data rates. When not using OSMR, the SNR of a transmission is determined by the currently stronger antenna of the AP to achieve antenna diversity.

We refer to the greedy scheduler as OSMR-g. We implemented the Linear Programming formulation with the LP solver available at [15] and refer to it as OSMR-lp. In the simulation, for OSMR-lp, the sub-transmission overhead is set to be 0 to serve as an upper bound on the performance, because the LP formulation does not consider the overhead. For comparison, we also ran simulation disabling OSMR transmissions, and refer to it as No-OSMR. In No-OSMR, when the AP gains access to the medium, it transmits without OSMR for a TXOP length or until all buffered packets are sent. It is similar to the Frame Aggregation in 802.11n [14].

In our simulation, γ=3ms. The OSMR-g and OSMR-lp scheduler consider a node not eligible for OSMR transmission if the phase shift of the channel ratio is more than $\pi/100$ per ms or the channel state is more than 10ms old. The upper layer calculates the set of urgent bytes assuming that they are sent at their base rates to ensure feasibility. To ensure fairness, when calculating the number of urgent bytes, for the set of nodes who cannot send all their buffered bytes in this TXOP, everyone is given an equal amount of time to transmit. Other nodes are given time to send all their buffered bytes.

We used four traces in [17], Trace 2 to Trace 5. The data was collected by TCPDump seen at the wired port at the AP in a LAN with 75 nodes for about 10 minutes. The traces include traffic from realistic applications such as WWW and VoIP. To match the description of the trace collection in [17], we first set the maximum distance to the AP to be 20m in our simulation. We ran our simulation for 500 seconds. The results show that OSMR-g and No-OSMR have almost exactly the same throughput for all traces. For instance, the result for Trace 3 is shown in Fig. 9 where the two lines overlap. This is because the traffic load in the trace is not high. Note that the upper layer protocols, e.g., TCP, typically probe the capacity

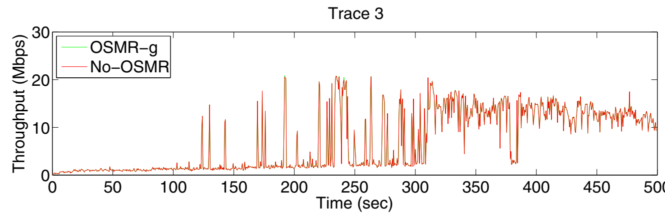


Fig. 9. Network downlink throughput in 500 seconds.

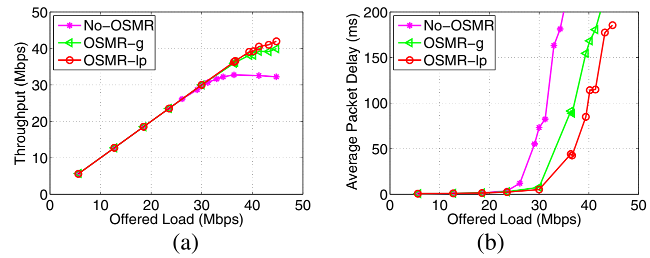


Fig. 10. Synthesized traffic combining 4 traces. The x axis is the offered load on the downlink. (a). Average downlink throughput. (b). Average packet delay.

of the network to avoid overloading the network, hence the traffic load in the trace is unlikely to exceed the capacity of an AP, and therefore not high enough to reveal the benefit of OSMR.

To evaluate the performance of the network at higher traffic load, we processed the trace files and combined Trace 2 to Trace 5 into one. As each trace contains 75 nodes, we created 10 merged nodes, and randomly combined the traffic of up to 30 actual nodes into one merged node. We used 30 seconds of the traffic trace from 400 seconds to 430 seconds. The maximum distance is set to be 60m. Because the combined traffic can be very heavy, we allow the AP to drop a packet if the total number of buffered packets exceeds 1000. Fig. 10 shows the network performance as a function of the offered load on the downlink averaged over 20 random seeds, where Fig. 10(a) is the average downlink throughput and Fig. 10(b) is the average downlink packet delay. We can see that OSMR-g achieves a very close throughput as OSMR-lp. It also performs significantly better than No-OSMR as the load increases.

To quantitatively measure the different schemes, we define the *sustainable throughput* as the maximum throughput when the average packet delay is less than 100ms. The sustainable throughputs of different schemes are shown in Table 2 for networks of various sizes, where the results for networks with 5, 15, and 20 nodes are obtained in a similar way as the network with 10 nodes. The improvements of OSMR-g over No-OSMR are also shown. We can see that the sustainable throughput is the highest when $N = 5$, because least number of nodes are competing for the air time. In fact, the sustainable throughput is determined by two related factors: (1) the number

TABLE 2
SUSTAINABLE THROUGHPUT (MBPS) AND OSMR-G IMPROVEMENT OVER NO-OSMR

	No-OSMR	OSMR-g	OSMR-lp	Improvement
N = 5	38.9	48.2	51.9	24%
N = 10	30.7	36.5	38.8	19%
N = 15	29.4	37.4	42.2	27%
N = 20	27.2	35.8	38.1	32%

of nodes (2) the number of compatible pairs. For example, the sustainable throughput of OSMR schemes for $N = 15$ is higher than for $N = 10$, because the gain by having more compatible pairs outweighs the loss of air time due to having more nodes. It is also interesting to notice that the improvement percentage is the lowest when $N = 10$. This is because for network with more nodes, there are more compatible pairs; for network with less nodes, the traffic to each individual node is heavier, such that compatible pairs are more likely to both have buffered packets. Note that the improvement is not as dramatic as one may have expected after being able to send multiple packets simultaneously. The major reason is that *compatible nodes may not both have buffered packets*. In a wireless LAN, it may happen that a node receives a large volume of traffic in a short period of time, while nodes compatible with this node receive little traffic, which causes the underutilization of the compatibility. This is true especially in larger networks even under a very high total network load. Nevertheless, overall, we can see that OSMR is capable of improving the performance by around 20-30% for networks of various sizes.

VI. Related Works

Recently, applying advanced signal processing techniques to wireless networks has drawn much interest in the networking community [6], [7], [8], [9], [10]. We note that these works consider single-antenna systems, and cannot exploit the capacity of multiple antennas. In [5], the IAC system was proposed, in which multiple senders coordinate transmissions to multiple receivers by interference alignment and interference cancellation with multiple antennas. We note that IAC is intended for a wireless LAN with multiple APs connected by high speed wired connections, while OSMR can be applied to networks with multiple APs as well as networks with only one AP. In addition, [5] considers a simple scenario in which all nodes have infinite load, while in this paper, we consider the more realistic scenario in which nodes may have arbitrarily random load, such that buffer state of the AP may be arbitrary, under which the scheduling problem is much more challenging. Our evaluation is also based on realistic traffic collected from wireless LANs, which shows that the randomness of the traffic has significant impact on the system performance, because the AP cannot exploit the compatibility of two nodes when one of them has no buffered packets.

In [20], [21], we proposed algorithms for OSMR focusing on sending the buffered packets in minimum time. The scheduler in this paper is different because it considers the QoS requirements and achieves higher efficiency by exploiting packet fragmentation. In addition, a detailed physical layer study is provided in this paper.

VII. Conclusions

In this paper, we gave a systematic study on employing the One-Sender-Multiple-Receiver (OSMR) transmission technique in wireless LANs. In the physical layer, we implemented prototype OSMR transmitter/receiver with GNU Software Defined Radio that allow one sender to send to two receivers simultaneously. We conducted experiments which show that wireless channels allow OSMR for a significant percentage of the time. We also studied the characteristics of wireless channels, and showed that wireless channel is stable for most of the time which is desirable for OSMR. Based on our physical layer study, in the MAC layer, we proposed extensions to the 802.11 MAC, as well as studied the packet scheduling problem and proposed a practical scheduler capable of exploiting OSMR efficiently and handling channel fluctuations. We evaluated the proposed protocol and scheduling algorithm based on simulations driven by wireless LAN traffic traces and wireless channel traces collected from our experiments. The results show that OSMR is capable of improving wireless LAN performance significantly.

References

[1] IEEE Computer Society LAN MAN Standards Committee, *IEEE Standard 802.11, Wireless LAN Medum Access Control (MAC) and Physical Layer (PHY) Specifications, 1999*.

[2] D. Tse and P. Viswanath, "Fundamentals of Wireless Communication," *Cambridge University Press*, May 2005.

[3] D. Gesbert, M. Kountouris, R. W. Heath, Jr., C. B. Chae, and T. Salzer, "From single user to multiuser communications: Shifting the MIMO paradigm," *IEEE Signal Processing Magazine*, vol. 24, no. 5, pp. 36-46, Oct., 2007.

[4] T. Yoo, N. Jindal, and A. Goldsmith, "Multi-antenna downlink channels with limited feedback and user selection," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 7, pp. 14781491, 2007.

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

[6] L. E. Li, R. Alimi, R. Ramjee, H. Viswanathan, and Y. R. Yang, "muNet: Harnessing multiuser capacity in wireless networks," *IEEE INFOCOM Minisymposium*, April 2009.

[7] D. Halperin, T. Anderson, and D. Wetherall, "Taking the sting out of carrier sense: Interference cancellation for wireless LANs," *ACM MOBICOM 2008*.

[8] S. Gollakota and D. Katabi, "ZigZag decoding: Combating hidden terminals in wireless networks," *ACM SIGCOMM 2008*.

[9] K. Jamieson and H. Balakrishnan, "PPR: Partial packet recovery for wireless networks," *ACM SIGCOMM 2007*.

[10] S. Katti, S. Gollakota and D. Katabi, "Embracing wireless interference: Analog network coding," *ACM SIGCOMM 2007*.

[11] A. Bose and C. H. Foh, "A practical path loss model for indoor WiFi positioning enhancement," *ICICS 2007*, Singapore, December 2007.

[12] "Gnu radio - gnu fsf project," *http://www.gnu.org/software/gnuradio*.

[13] Ettus. Inc, "Universal Software Radio Peripheral," *http://ettus.com*.

[14] "802.11n: Next-Generation Wireless LAN Technology," *http://80211n.com/white_paper/802_11n-WP100-R.pdf*.

[15] http://lpsolve.sourceforge.net/5.5/

[16] Cisco Aironet 802.11a/b/g wireless cardbus adapter, http://www.cisco.com/.

[17] http://www.winlab.rutgers.edu/ẽrgin/mobicom2007/

[18] http://www.umtsworld.com/technology/hsdpa.htm

[19] http://standards.ieee.org/getieee802/download/802.11-2007.pdf

[20] Z. Zhang, M. Zhao and Y. Yang, "Enhancing downlink performance in wireless networks by simultaneous multiple packet transmission," *IEEE Transactions on Computers*, vol. 58, no. 5, pp. 706-718, May 2009.

[21] Z. Zhang and S. Bronson, "A packet scheduling algorithm for optimizing downlink throughput in wireless LANs with the one-sender-multiple-receiver technique," *IEEE Globecom*, 2009.

[22] Z. Zhang, S. Bronson, J. Xie and H. Wei, "The One-Sender-Multiple-Receiver technique and downlink packet scheduling in wireless LANs," *Technical Report TR-090217*, Florida State University, 2009.