Distributed Link Adaptation for Multicast Traffic in MIMO-OFDM Systems

Sungho Yun[a], Constantine Caramanis[a], Robert W. Heath Jr.[a]

[a] *Wireless Networking and Communications Group*
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas 78712-0240

Abstract

Multicast traffic exploits the broadcast nature of the wireless medium to deliver the same data to multiple users improving the bandwidth efficiency. Link adaptation can be used in multicast transmission to further increase data rates exploiting feedback from the users. However, it is not easy to have the quality of service (QoS) of every intended receiver met while pushing the data rate to the link capacity is not easy. Due to this difficulty, the conventional approach is to transmit isotropically with a fixed basic rate giving up the opportunity of increased throughput. For point-to-point unicast traffic, machine learning algorithms have recently found successful application in link adaptation due to their flexibility and ability to capture more environmental effects implicitly than classical adaptation algorithms. In this paper we propose a machine learning based distributed algorithm for link adaptation for multicast traffic in multiple-input multiple-output (MIMO) orthogonal frequency-division multiplexing (OFDM) systems. Our computations show that the data driven approach for link adaptation provides good prediction of the optimal modulation and coding scheme (MCS) outperforming the fixed MCS policies collectively. The distributed algorithm using dual decomposition reduces the required feedback amount significantly while maintaining the near-optimal throughput performance.

Keywords: link adaptation, multicast, machine learning, dual

Email addresses: `shyun@mail.utexas.edu` (Sungho Yun), `caramanis@mail.utexas.edu` (Constantine Caramanis), `rheath@ece.utexas.edu` (Robert W. Heath Jr.)

decomposition

1. Introduction

Multicast transmission exploits the broadcast nature of the wireless medium by providing the same copy of the data to multiple users, instead of sending the same data separately to different users. Therefore, multicast improves the bandwidth efficiency significantly when data need to be transmitted to more than one receiver at the same time, e.g., television like broadcasting services and newsflashes. Because the group of users viewing multicast traffic in a given cell may be small, the bandwidth efficiency of multicast transmission can be improved further through link adaptation. Unfortunately, link adaptation is challenging in multicast systems since there are multiple links and the quality of service (QoS) of each receiver has to be met. This becomes even more difficult in multiple-input multiple-output (MIMO) orthogonal frequency-division multiplexing (OFDM) systems, due to the increased number of control and environmental parameters that may impact the performance of each link.

Recently a new approach has been taken to link adaptation in MIMO-OFDM systems using machine learning algorithms, e.g. [1, 2, 3, 4]. Machine learning algorithms have advantages over conventional link adaptation algorithms in the sense that they are inherently data-driven, and rather than rely on model assumptions and approximations. They capture environmental effects implicitly. The authors in [1] propose an effective low dimensional feature set as an input to the k nearest neighbor (k-NN) algorithm for modulation and coding scheme (MCS) selection and extend the result and develop an online k-NN algorithm for real time link adaptation. Multi-class support vector machine (SVM) algorithm is used for link adaptation for MIMO-OFDM systems in [3] and online support vector regression (SVR) algorithm reduces the storage requirement and computation overhead, yet maintains the performance of offline learning algorithms in [4].

Although there has been some progress on data driven approaches for link adaptation in MIMO-OFDM systems [1, 2, 3, 4], as well as other strategies for adaptation [5, 6, 7], these approaches are tailored to unicast traffic and cannot be directly extended to multicast traffic. This is because the set of transmit parameters that is uniformly optimal over all the links cannot be inferred from feedback information of the optimal transmit parameter sets of individual links. Therefore, the conventional approach to wireless multicasting is to transmit isotropically with a fixed basic rate. This is a reasonable approach if there are a large number of users. If there are a small number of users, however, the spectral efficiency can be further improved through link adaptation [8, 9]. Because of the power of data driven link adaptation techniques in unicast settings, it makes sense to develop data driven adaptation algorithms that exploit features of multicast transmission.

In this paper, we propose a novel link adaptation scheme for multicast traffic in wireless MIMO-OFDM systems. We consider two important aspects in adaptive transmission: MCS selection and precoding (or beamforming depending on the number of streams). Correctly choosing the MCS involves achieving high throughput while maintaining QoS requirements. Proper precoding is required to provide the highest performance in MIMO systems by adapting the number of streams based in part on the number of significant eigenmodes in the channel. We propose a feedback-based link adaptation scheme in which the receivers compute the best MCS and precoder and feed back the information to the transmitter for the following transmission. It is demonstrated in [10] that a lightweight feedback-based scheme can offer substantial improvement in performance over feedback-free schemes for multicast. However due to the larger number of receivers in multicast compared to unicast, feedback limitation requirements are more demanding; hence we design the algorithm to work with as little feedback as possible.

Limited feedback in MIMO systems is extensively studied in many literatures (see an overview paper [11] and references therein). One popular approach is codebook based precoding [12], in which the best set of quantized precoding matrices are computed and the index of the optimal precoder is sent from the receiver to the transmitter. They have shown that good codebooks correspond to packings on the Grassmann manifold. We use codebook based precoding with a different codebook for each number of data streams implementing multi-mode limited feedback precoding [13]. Although given the limited budget of feedback this provides the best precoder in single-carrier and flat channel setting, it does not scale very well to OFDM systems. Ideally, one needs to compute as many precoding matrices as the number of subcarriers; hence the feedback required grows linearly in the number of subcarriers. An alternative is to compute precoders for subsets of carriers and to use clustering and/or interpolation, see e.g. [14] and [15].

A near optimal transmit beamforming vector for multicast is computed in [16]. The authors show that computing the optimal beamforming vector in the sense that it maximizes the smallest receiver SNR is an NP-hard problem. They show that near-optimal beamforming vector can be obtained using an appropriate semi-definite relaxation (SDR) of the original optimization problem. Authors in [17] compute the closed-form solution of the optimal transmit beamforming vetor for the case of 2 users and propose a successive algorithm for more users. While these approaches alleviate issues of computational complexity, issues of feedback complexity remain. These approaches assume that the full channel state information required for all the intended receivers is available at the transmitter side. We believe this is unrealistic, especially when there are many such receivers.

The authors in [18, 19] address this issue by letting the receivers feed back the quantized channel state information. The transmitter, then, computes the beamforming vector, that maximizes the smallest average receiver SNR, where the average is taken over the distribution of the channel direction vectors that can be mapped to the same quantized channel direction vector. Employing the similar SDR-randomization as in [16], the optimization problem can be solved in polynomial time. Also in [20], the optimal precoding matrix is chosen from the precoding codebook. Each receiver feeds back the index of its best precoding matrix. Assuming the transmitter knows the channel statistics, it computes the average packet drop rate of the receivers and the precoding matrix that minimizes it is chosen. However these results are not directly applicable to OFDM systems due to the increased feedback overhead. Moreover, even with quantized channel state information for all the subcarriers at the transmitter side, MCS selection is difficult because of the effect of interleaving and forward error correction (FEC) when we consider systems with a uniform MCS for all the subcarriers as in IEEE 802.11 WLANs.

Rate selection in multicast is considered in [21], [8] and [9]. The authors in [21] consider the layered source coding and Network Layer adaptation for IP multicast. In [8], SNR is fed back from the receiver to the transmitter and to avoid the collision due to the simultaneous feedback transmissions, random back off favoring the receivers with low SNR is used. At the transmitter side, the rate is decided according to the minimum receiver SNR. In [9], a similar approach is proposed, except that instead of dedicated SNR feedback, RTS and CTS are used to convey the rate information. CTS packets are transmitted at the same time, but the longer duration of CTS is used for the receiver with lower SNR and the rate is chosen depending on the duration of CTS, which means the rate adaptation chooses the lowest rate over the receivers conservatively to meet the QoS of the receiver with the lowest SNR. In both lines of work, the authors consider neither MIMO nor precoding. This makes the problem much simpler since in single-input single-output (SISO) without precoding the rates are generally comparable, i.e., some rates result in higher packet error rate (PER) than the other rates uniformly across channel state. However, this is not the case with MIMO and precoding as shown in [1].

In this paper, we propose a data-driven approach to predict performances of MCSs and incorporate precoding to improve the throughput. We show that simply using effective channel state information (channel matrix multiplied by a precoding matrix) as an input to the mapping function from the channel to PER results in good estimation on PER and this estimation can be used to choose the right precoder and MCS. This solves the problem of quantized precoding algorithms not being scalable to OFDM systems. Using this approach, we develop a frame work of limited feedback link adaptation for multicast traffic. A centralized adaptation algorithm is proposed which requires full performance information for each set of transmitting parameters per receiver. We compare the algorithm with fixed MCS policies and show that it works at least as well as the best fixed policy at each SNR level confirming the validity of use of machine learning algorithms for multicast link adaptation. Then, we propose a distributed algorithm using dual decomposition. We show that the required feedback amount is much smaller; growing only logarithmically in the number of choices of MCSs and precoding matrices, yet the performance is close to that of the centralized algorithm.

The rest of this paper is organized as follows. In Section 2, we define our system model and learning model for PER prediction. A centralized algorithm is developed in Section 3. We present a distributed algorithm and its variations in Section 4 and compare the performance in various environment settings. The required feedback amount is analyzed in Section 5 and finally Section 6 concludes the paper.

2. System Model

2.1. MIMO-OFDM Systems

In MIMO-OFDM systems with N_t transmit antennas and N_r receive antennas, data are transmitted over $N_s \leq \min\{N_t, N_r\}$ spatial streams. In the frequency domain, a baseband signal $\mathbf{x}[m,n]$ for the nth subcarrier of the mth OFDM symbol, is multiplied by a pre-coding matrix $\mathbf{F}[n]$, then transmitted over a wireless channel, $\mathbf{H}[n]$. We assume that the channel adds complex Gaussian noise, $\mathbf{v}[m,n] \sim CN(0,\sigma^2\mathbf{I})$. At the receiver, we use a linear equalizer, $\mathbf{G}[n]$ to recover the transmitted signal. Then, with transmit power E_s, N_{sub} subcarriers and N_{OFDM} OFDM symbols, the received signal has the form

$$\mathbf{y}[m,n] = \sqrt{E_s}\mathbf{G}[n]\mathbf{H}[n]\mathbf{F}[n]\mathbf{x}[m,n] + \mathbf{G}[n]\mathbf{v}[m,n], \tag{1}$$

where $n \in \{0, \cdots, N_{sub}-1\}$ and $m \in \{0, \cdots, N_{OFDM}-1\}$.

The optimal codebook of quantized precoding matrices is computed in [12] to minimize the average distortion introduced by quantization. With N_{MCS} MCSs ($MCS_1 \ldots MCS_{N_{MCS}}$) and $N_{\mathbf{F}}$ quantized precoding matrices ($\mathbf{F}_1 \ldots \mathbf{F}_{N_{\mathbf{F}}}$), we have $N_{MCS} \times N_{\mathbf{F}}$ pairs to choose from for each transmission. We enumerate them as $k \in \{1 \ldots N_{MCS}\}$, $l \in \{1 \ldots N_{\mathbf{F}}\}$ and $j = (k \times l) \in \{1 \ldots N = N_{MCS} \times N_{\mathbf{F}}\}$ and call j a parameter set which is a pair of an MCS and a precoding matrix. Assuming M receivers, we define the performance metric u_{ij}, $i \in \{1 \ldots M\}$ and $j \in \{1 \ldots N\}$ as the utility of receiver i using parameter set j. The utility can be defined depending on the QoS requirement of interest. Throughout this paper we use the following utility

$$u_{ij} = \begin{cases} (1 - PER_{ik}) \times r_k & \text{if } PER_{ik} < \mathcal{T} \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where r_k is the rate of MCS_k, $\mathbf{F}_l$ is the precoder l, PER_{ij} is the packet error rate of user i using parameter set j and $\mathcal{T}$ is the packet error rate threshold. One can imagine other application-specific utility functions. While we believe our results are generally applicable, we do not pursue this here.

2.2. Learning Model for Link Adaptation

For a given MCS, we would like to use the past sequence of observations to obtain a prediction function of its future performance, i.e., its PER for new channel measurements. Let X denote the set of all values that the channel measurement can take, and let Y be the average PER. A predictive function obtained from past data (i.e., from a sequence of observations $\{(\mathrm{x}_1, y_1), \cdots, (\mathrm{x}_t, y_t)\}$ that have used the MCS) is called a regression function. We obtain a regression function, f, that minimizes the following objective:

$$\int_{X \times Y} (f(\mathbf{x}) - y)^2 d\rho, \tag{3}$$

where ρ is a probability measure on $X \times Y$. Note that this regression function approximates the mapping from the channel measurement to failure rate of transmission for the given MCS. We use SVR which finds the regression function mapping from the channel state information to the PER given an MCS: $f(\mathbf{H}, MCS_i) \to PER$ as in [4]. SVR constructs the regression function as a form of hyperplane in the space of $X \times Y$ (sample space) or in a higher dimensional space (feature space) if we use kernels to get an non-linear regression function. The sample complexity depends on the dimension of the space; hence it needs to be limited. We use the ordered post-processing SNR as an input to the machine learning algorithms as developed in [1]. It is shown in [1] that the ordered post-processing SNR represents the channel state well while limiting the dimension of sample space low, thus solving the curse of dimensionality problem of machine learning algorithms. Regression functions are generated offline using previously observed training samples. In training phase, precoding may drive the channel samples toward a certain direction. Therefore training is done with samples without precoding to avoid the possible distributional bias of channel realizations and achieve higher sample diversity for better prediction. To estimate a packet error rate with a precoding matrix $\mathbf{F}_l$ given a channel matrix $\mathbf{H}[n]$, we simply use the effective channel matrix, $\mathbf{H}[n]\mathbf{F}_l$ as an input to the regression function.

3. Centralized Optimization

Given all the utility information for all the receivers and all the parameter sets, we can develop a centralized optimization to compute the optimal parameter set. For each parameter set, we take the minimum utility over all the receivers. This gives us guaranteed conservatively performance for all the receivers for each parameter set. Among them we choose one with the highest minimum utility. Using the definitions from Section 2 we formulate the following optimization for parameter set selection.

Algorithm 1. *Centralized Optimization*

$$j^* = \arg\max_j (\min_i u_{ij}). \tag{4}$$

Although in most wireless systems, ACK packets are not used in multicast traffic, since we consider feedback based link adaptation, we assume the use of ACK packets throughout this paper and also assume that the feedback information is piggybacked on ACK packets. The algorithm is described in Table 1.

Receiver	Measure channel matrix **H**
	Estimate PER for each pair of MCS and precoding matrix **F**
	Compute utility according to PER estimation
	Feedback the utility information
Trasmitter	Compute the optimal parameter set using (4)

Table 1: Centralized Optimization

Number of transmit antennas	2
Number of receive antennas	2
Number of taps	4
Message length	128 Bytes
Packet error rate threshold	10 %
Center frequency	2.4 GHz
Number of receivers	3 (or 5)
Receivers' SNR	0 2̃7 dB
Receivers' velocity	0 (,5 or 50) m/s
Number of packets	100
Number of iterations	100

Table 2: IEEE 802.11n system model

3.1. Comparison with Fixed MCS policies

To show the applicability of using SVR for PER estimation, we compare Centralized Optimization with the policy of fixed MCSs and no precoding. While naturally we would expect our algorithm to individually outperform each of those fixed policies, our simulations show we outperform them collectively, doing at least as well as the best fixed policy at each SNR level. Our simulation model adopting IEEE 802.11n MIMO-OFDM Wireless LAN systems is described in Table 2. We also compare the Centralized Optimization algorithm with and without precoding. In this paper, throughput is always based on the requirement that all packets are successfully transmitted to all the receivers. Therefore, if one receiver misses a packet, the throughput for the packet is zero.

As shown in Figure 1, Centralized Optimization without precoding closely follows the best case of all fixed MCS policies. This demonstrates the usefulness and advantages of using SVR for link adaptation. Also, it is worth noting that Centralized Optimization with precoding almost always outper-

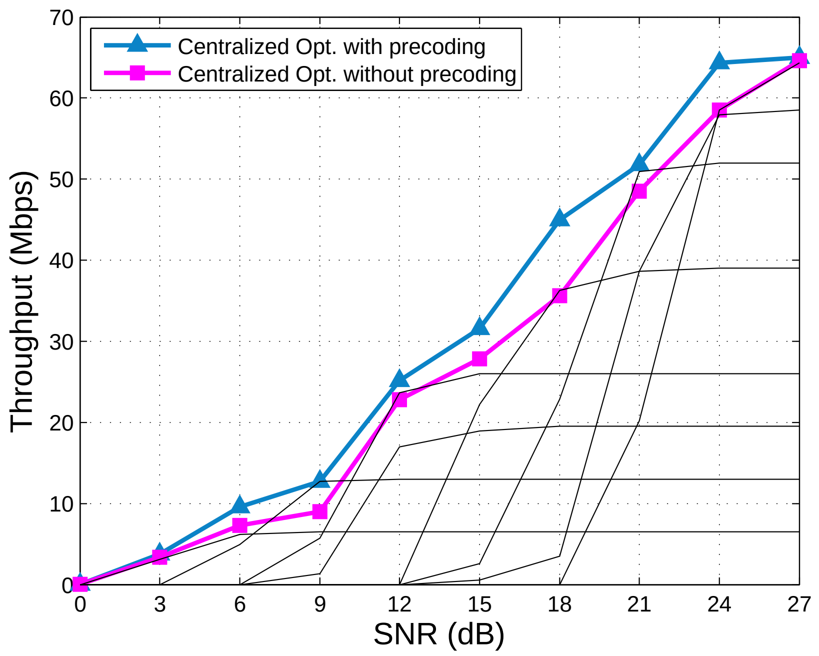

Figure 1: Throughput comparison between Centralized Optimization with and without precoding vs. fixed rate MCSs

forms Centralized Optimization without precoding by around 3 Mbps on average.

However, the biggest challenge to using Centralized Optimization is the amount of feedback overhead required. As we discuss in more detail later, the amount of feedback needed to convey all the utility information is too large and grows linearly in the number of quantized precoding matrices. Therefore, even though the precoding improves the performance, it is not plausible to use with Centralized Optimization.

3.2. Per Receiver Optimization

One way to address this challenge of Centralized Optimization regarding feedback overhead is a fully receiver-based optimization scheme, in which the only feedback required is the index of each receiver's optimal parameter set. In this way, the amount of feedback would grow only logarithmically. While in Centralized Optimization, we essentially take the parameter that maximizes the minimum utility, in this algorithm that we call Per Receiver Optimization, we take the parameter set with the minimum MCS rate among the receivers' optimal parameter sets. When more than one parameter set has the same MCS rate and differs only in precoding matrices, we break the tie randomly.

Algorithm 2. *Per Receiver Optimization*

$$\begin{array}{ll} i^* & = \arg\min_i(\max_j u_{ij}) \\ j^* & = \arg\max_j u_{i^*j} \end{array} . \tag{5}$$

Although Per Receiver Optimization significantly reduces the amount of feedback, it does not guarantee the globally optimal parameter set since the parameter sets are not comparable with MIMO and precoding. A parameter set with the lowest rate MCS may result in a higher PER for another user; hence conservative choice does not guarantee the successful transmission to all the receivers. As a result, the performance, while comparable, is significantly degraded compared to Centralized Optimization as shown in Figure 2. We see that the throughput difference is as large as 18 Mbps at 21dB SNR.

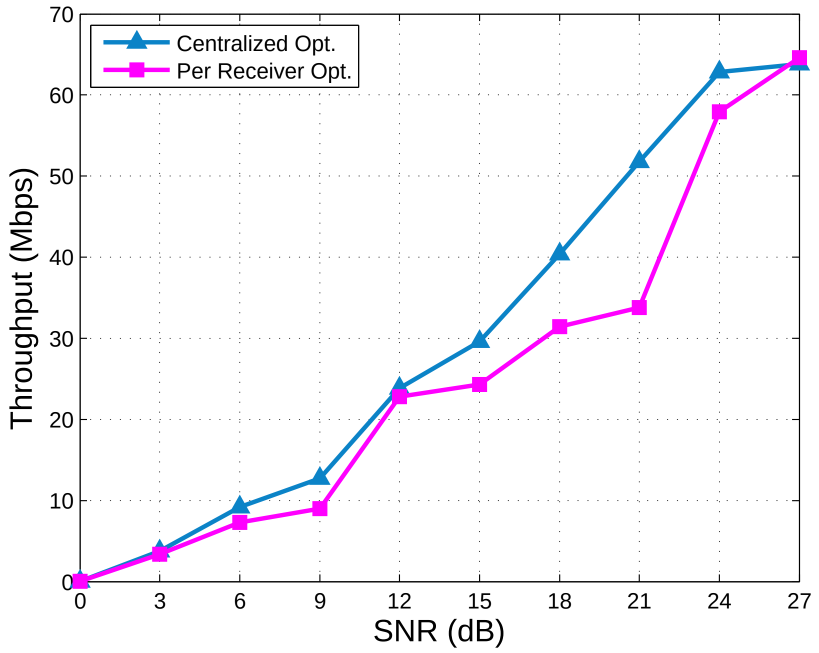

Figure 2: Throughput comparison between Centralized Optimization and Per Receiver Optimization

4. Distributed Optimization

In this section, we develop a distributed algorithm that aims to limit the amount of feedback, yet maintain the performance of the full-feedback Centralized Optimization. The basic idea is to have the main optimization happening at the receiver side, as in Per Receiver Optimization in order to minimize the feedback overhead, but the receiver side optimization being based on the price information per parameter set controlled by the transmitter. The transmitter updates these prices for parameter sets so that in a few iterations, the receivers agree on a parameter set which performs close to Centralized Optimization. Thus, we develop an approach based on tools from distributed optimization [22].

4.1. Dual Decomposition

Distributed optimization using dual decomposition has been studied extensively in the networking area, where it is used to control congestion in routing problems (e.g., [22], [23], [24] and [25]). We first formulate the dual decomposition of multicast link adaptation and describe the distributed algorithm we propose.

First we assume a time allocation profile $\vec{t}$ that divides time frames continuously to have time frame t_j dedicated to the parameter set j. Then the average utility for the receiver i using the profile $\vec{t}$ is $\vec{t} \cdot \vec{u}_i$. The optimal time profile is the one maximizes the minimum utility over all the intended receivers:

$$\arg \max_{\vec{t} \geq 0, \|\vec{t}\|_1 \leq 1} \{\min_i \vec{t} \cdot \vec{u}_i\} \tag{6}$$

To have a distributed algorithm, we assume a different time profiles, $\vec{t_i}$ for receivers. Introducing a new variable F to handle the minimum part, we get the following optimization. The first constraint means we are maximizing the minimum, while the second constraint means all the time profiles of receivers need to agree.

$$\begin{array}{lll} \min_{\vec{t_i},F} & -F & \\ \text{s.t.} & F \leq \vec{t_i} \cdot \vec{u}_i & \forall i \\ & \vec{t_i} = \vec{t}_{i+1} & \forall i \\ & \sum_j t_{ij} \leq 1 & \forall i \\ & \vec{t_i} \geq 0 & \forall i \end{array} \tag{7}$$

Lagrangian dual function of (7) is as follows.

$$g(\lambda,\xi) \triangleq \inf_{\sum_j t_{ij}\le 1, \vec{t}_i \ge 0} \big(-F + \sum_i \lambda_i (F - \vec{t}_i \cdot \vec{u}_i) + \sum_{i,j} \xi_{ij}(t_{ij} - t_{i+1,j})\big) \quad (8)$$

After grouping the terms according to the receivers and rearranging them we get the following dual problem.

$$\max_{\lambda\ge 0,\xi,F} -\left(\sup_{\sum_j t_{ij}\le 1, \vec{t}_i\ge 0, F} \big(\sum_i (\lambda_i \vec{t}_i \cdot \vec{u}_i - \sum_j \xi_{ij} t_{ij} + \sum_j \xi_{i-1,j} t_{ij}) + F - \sum_i \lambda_i F\big) \right) \quad (9)$$

Notice that the time profiles are decoupled and can be optimized at the receiver side with given λ and ξ.

$$\sup_{\sum_j t_{ij}\le 1, \vec{t}_i\ge 0} f(\vec{t}_i) \triangleq \sup_{\sum_j t_{ij}\le 1, \vec{t}_i\ge 0} \sum_j (\lambda_i u_{ij} - \xi_{ij} + \xi_{i-1,j}) t_{ij} \quad (10)$$

Then,

$$t^*_{ij} = \begin{cases} 1 & j = \arg\max_k \lambda_i u_{ik} - \xi_{ik} + \xi_{i-1,k} \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

Note that even though we assumed a continuously dividable time frame, the resulting time profile is a standard basis vector, which means only one optimal parameter set is used for a packet transmission. Also, by letting $\frac{\partial(1-\sum_i \lambda_i)F}{\partial F} = 0$, $\sum_i \lambda_i = 1$.

With the solution above, the Lagrangian Dual problem becomes

$$\max_{\sum_i \lambda_i = 1, \lambda\ge 0, \xi} \sum_i (-\lambda_i \vec{t}^*_i \cdot \vec{u}_i + \sum_j \xi_{ij} t^*_{ij} - \sum_j \xi_{i-1,j} t^*_{ij}) \quad (12)$$

We use the subgradient method to update the Lagrangian multipliers at the transmitter side. Letting $g(\lambda,\xi) \triangleq \sum_i (-\lambda_i \vec{t}^*_i \cdot \vec{u}_i + \sum_j \xi_{ij} t^*_{ij} - \sum_j \xi_{i-1,j} t^*_{ij})$ and take derivatives, we find the following.

$$\begin{aligned} \partial_{\lambda_i} g &= -\vec{t}^*_i \cdot \vec{u}_i &&= -u_{ij^*} \\ \partial_{\xi_{ij}} g &= t^*_{ij} - t^*_{i+1,j} \end{aligned} \quad (13)$$

Therefore, with step size α the transmitter updates the lagrangian multipliers as follows.

$$\begin{aligned} (\lambda_i)^{t+1} &= (\lambda_i)^t - \alpha u_{ij^*} \\ (\xi_{ij})^{t+1} &= (\xi_{ij})^t + \alpha(t^*_{ij} - t^*_{i+1,j}) \end{aligned} \quad (14)$$

Initialize:	
Receiver	Agree on the initial Lagrangian multipliers
For each transmission:	
Receiver	Update the Lagrangian multipliers according to (14)
	Compute the optimal parameter set according to (11)
	Feedback the index of the parameter set and the utility
Transmitter	Feed forward the collected feedback information

Table 3: Distributed Optimization

To meet the constraint of $\vec{\lambda}$, we take the positive value of $\vec{\lambda}$ and normalize it by picking the closest point to $\vec{\lambda}$ on the simplex: $\sum_i \lambda_i = 1, \lambda \geq 0$.

Important remarks of the above dual decomposition are listed below.

- Due to the structure of the solution, (11), the index of the optimal parameter set is the only information to be conveyed instead of $\vec{t}_i^*$.

- Instead of feeding forward all the Lagrangian multipliers from the transmitter, we first let the receivers agree on the initial multipliers and compute them individually. From (11) and (14) we can see that the only missing information while updating the multipliers at the receiver side is other receivers' optimal parameter sets and the corresponding utilities for those parameter sets. Therefore, each receiver feeds back the index of its optimal parameter set and the utility of that parameter set. The transmitter accumulates all the feedback information from the receivers and feeds forward it by piggy backing it on the next data transmission.

- The intuition behind (11) and (14) is that if a receiver's current utility is higher than the others, when choosing the next optimal parameter set, the importance of utility is discounted by decreased λ_i and the importance of parameter set agreement among receivers is more pronounced.

The process of the algorithm, we call Distributed Optimization, is summarized in Table 3

Before we proceed to the average throughput result of Distributed Optimization, we first analyze the result of Per Receiver Optimization in more detail to see why it did not work well. Looking at Figure 2, we may notice that the throughput is especially bad at around SNR of 15 dB and 21 dB. Detailed analysis reveals that there is an important cases that Per Receiver Optimization suffers from. We describe the case and show how Distributed Optimization handles it.

Case 1. *Individual optimal parameter sets and the global optimal parameter set from the Centralized Optimization have the same MCS, but different precoding matrices*

We give an example of Case 1. The following example is one of the 100 channel realizations with 21 dB SNR. In this example, all the individual optimal parameter sets have MCS_8 and differ in the choices of precoding matrices. The expected throughput per each precoder is shown in Table 4. Centralized Optimization chooses $\mathbf{F}_4$ as a precoder since its minimum utility over receivers is largest. On the other hand, with Per Receiver Optimization, the receiver 1, 2 and 3 choose precoders $\mathbf{F}_3$ or $\mathbf{F}_6$, $\mathbf{F}_5$ and $\mathbf{F}_1$ respectively. Therefore no matter which precoder the transmitter chooses among those, the resulted minimum utility is 0 Mbps. After thorough observation, we have found that this pattern happens more at the transition points from an MCS to a higher MCS at which Per Receiver Optimization suffers more.

	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$	$\mathbf{F}_4$	$\mathbf{F}_5$	$\mathbf{F}_6$	$\mathbf{F}_7$	$\mathbf{F}_8$	Max
Receiver 1	0.0	0.0	65.0	64.7	0.0	65.0	0.0	0.0	65.0
Receiver 2	64.9	64.6	64.9	64.8	65.0	64.8	64.7	60.6	65.0
Receiver 3	65.0	0.0	0.0	64.9	0.0	0.0	0.0	64.6	65.0
Min	0.0	0.0	0.0	64.7	0.0	0.0	0.0	0.0	

Table 4: SVR estimated throughput for three users for different precoders

Table 5 shows which precoders different optimization algorithms have chosen. As we can see in the table, with Distributed Optimization, the precoder choice of each receiver converges to $\mathbf{F}_4$, which is the optimal precoder of Central Optimization.

However, convergence rate is rather slow, thus the average throughput does not improve much over Per Receiver Optimization as shown in Figure 3.

To further improve the throughput performance, in the following two sections we make modifications in the Distributed Optimization.

Packet Idx	Centralized Opt.	Per Receiver Opt.	Distributed Opt.		
1	4	3	3	5	1
2	4	3	6	1	4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
33	4	3	4	4	1
34	4	3	4	4	4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 5: Precoder choice of different optimization algorithms

4.2. Distributed Optimization with Weighted Step Size

First we revisit the update rule for Lagrangian multipliers. In the previous update rule (14), the step size is uniform for Lagrangian multiplier ξ_{ij}, which corresponds to the price of parameter set j for receiver i. When updating, the importance of the current choice is discounted and more importance is put on the next receiver's choice. Suppose a receiver knows the utility of the next receiver with its previous optimal parameter set. Then, it makes more sense to discount more if the next one's throughput performance is too bad with that parameter set. On the other hand, if the next receiver's throughput is not too bad, it is not necessary to discount the previous choice. Following this reasoning, we put asymmetric weights on the step size for each update rule for ξ_{ij} as follows.

$$(\xi_{ij})^{t+1} = (\xi_{ij})^t + \alpha|u_{ij^*} - u_{i+1,j^*}|(t^*_{ij} - t^*_{i+1,j}) \tag{15}$$

This update rule requires the knowledge of the next receiver's utility. To obtain this information, each receiver feeds back the utility for its previous receiver's optimal parameter set and the transmitter collects and broadcasts this information.

4.3. Distributed Optimization with Local Message Passing

In addition to the modification to the update rule, we also change the receiver side optimization (11) as follows.

$$t^*_{ij} = \begin{cases} 1 & j = \arg\max_k \lambda_i u_{ik} - |u_{ij^*} - u_{i+1,j^*}|\xi_{ik} + |u_{i-1,j^*} - u_{ij^*}|\xi_{i-1,k} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

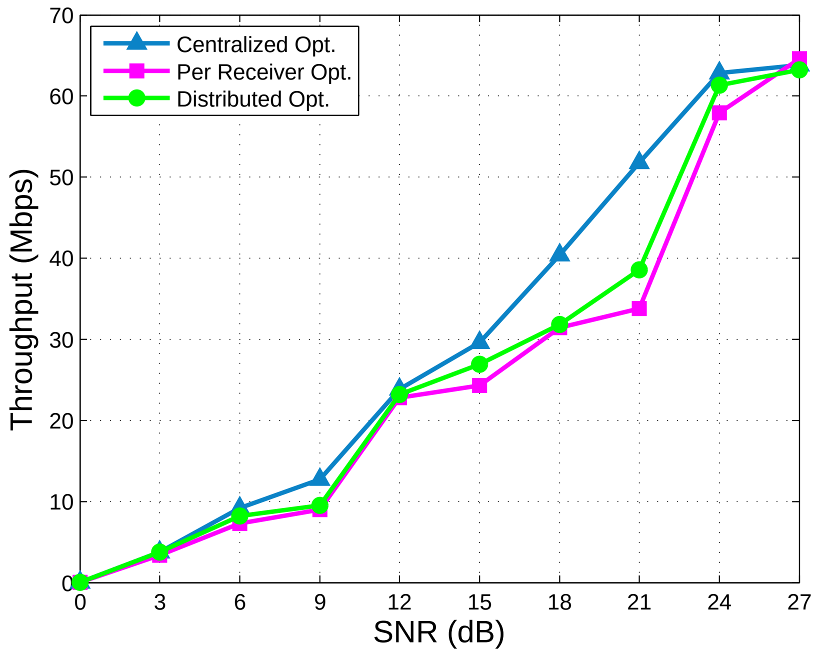

Figure 3: Throughput comparison between Centralized Optimization, Per Receiver Optimization and Distributed Optimization

The reasoning is similar to the update rule modification. If the next receiver's utility is bad for a certain parameter set, it puts higher price for the set and avoids choosing it. However, this modified algorithm requires the knowledge of utility for all the parameter sets of the previous and the next receivers and the overall amount of information exchange is the same as that of Centralized Optimization. One advantage of this algorithm over Centralized Optimization is that unlike Centralized Optimization, the receivers need their neighbors' utility information only. Therefore, if the receivers are far apart from each other, they can broadcast their information to the neighbors using usual Distributed Coordination Function (DCF) to exchange the information as much simultaneously as possible while avoiding collisions at the same time. This reduces the airtime compared to Centralized Optimization in which all the receivers have to take turns to feed back their utility information to the transmitter to avoid collisions at the transmitter side.

The average throughput comparison is shown in Figure 4. As shown in the figure, the performance of Distributed Optimization with local message passing follows the performance of Centralized Optimization very closely and Distributed Optimization with asymmetric weights improves the throughput over the original Distributed Optimization by at most 8 Mbps.

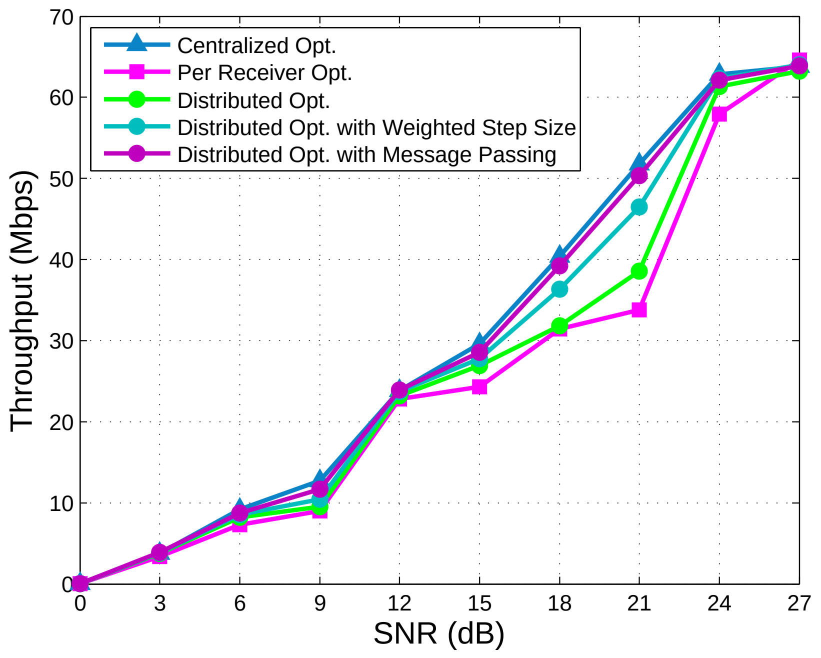

Figure 4: Throughput comparison between Centralized Optimization, Per Receiver Optimization and Modified Distributed Optimizations

In the example of Case 1, Distributed Optimization with asymmetric weights converges to the optimal precoder after 7 packets and Distributed Optimization with local message passing converges after 6 packets, while the original Distributed Optimization converges after 33 packets.

4.4. Number of Receivers

As we have seen so far, the most important factor that affects the performance of Distributed Optimization is the rate of convergence. In this section, we will analyze the effect of the number of receivers, one of two factors that can impact the rate of convergence. The other one is the rate of channel variation which will be discussed in the following section. Since in (7), the time profile agreement constraint is forced only between neighboring receivers in the sense of indexing, more receivers will slow down the convergence. With a given SNR 21 dB, relative performance of each algorithm to Centralized Optimization is shown in Figure 5 with varying number of receivers. The specific SNR level 21dB is chosen because in the previous result, it is the SNR level at which all the algorithms but Centralized Optimization performs most poorly. With the increased number of receivers up to 7, the relative throughput of Distributed Optimization, Distributed Optimization with asymmetric weights and Distributed Optimization with local message passing drop down to 46%, 58% and 67% respectively compared to Centralized Optimization. We can increase the convergence rate by having the time profile agreement constraints between all the pairs of receivers, but it will also increase the number of Lagrangian multipliers, thus the feedback amount and the computational complexity.

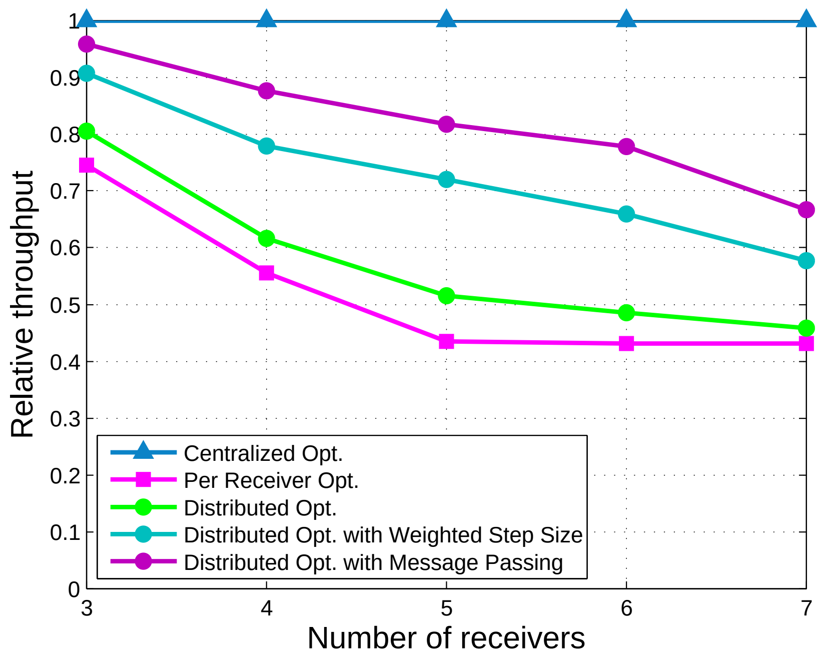

Figure 5: Relative throughput of Per Receiver Optimization and Modified Distributed Optimizations to Centralized Optimization with varying number of receivers

4.5. Channel Variation

Another factor that impacts the convergence rate is the variation of channel over time. If the convergence rate is slower than the channel variation, the information acquired from the previous packet transmissions becomes less useful and the improvement over Per Receiver Optimization will be decreased. To see the impact of channel variation, the simulation result with varying receivers' moving speed is shown in Figure 6. Same as in Section 4.4, SNR is 21 dB and the relative throughput of each algorithm is plotted. At the highest receivers' velocity 50 m/s ($\sim$ 110 mph), only Distributed Optimization with local message passing sustains its relative performance at around 90% and all the other algorithms show around 70% relative throughput performance. However, Distributed Optimization with asymmetric weights performs well up to 5 m/s maintaining its relative throughput performance above 80%, which demonstrates its usefulness in indoor environment. We expect the delayed feedback has the similar impact on the performance to channel variation since it also degrades the timeliness of information acquired from the feedback.

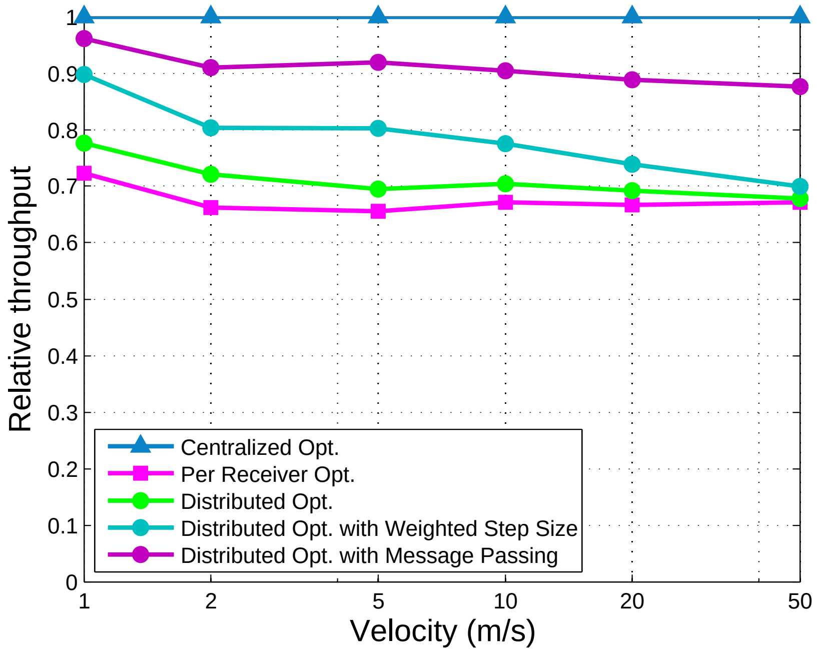

Figure 6: Relative throughput of Per Receiver Optimization and Modified Distributed Optimizations to Centralized Optimization with varying receiver velocity

Centralized Opt.	$N_{MCS} \cdot N_{\mathbf{F}} \cdot N_{bits}$
Per Receiver Opt.	$\log_2(N_{MCS} \cdot N_{\mathbf{F}})$
Distributed Opt.	$2(\log_2(N_{MCS} \cdot N_{\mathbf{F}}) + N_{bits})$
with asymmetric weights	$2(\log_2(N_{MCS} \cdot N_{\mathbf{F}}) + 2N_{bits})$
with loca message passing	$N_{MCS} \cdot N_{\mathbf{F}} \cdot N_{bits} + 2(\log_2(N_{MCS} \cdot N_{\mathbf{F}}) + N_{bits})$

Table 6: Feedback amount

5. Feedback Overhead

In this section, we analyze the feedback overhead of each link adaptation algorithm proposed in this paper. The number of feedback bits required for each receiver is computed in Table 6, where N_{bits} is the required number of bits to represent a floating point number. Assuming mandatory 16 MCSs for IEEE 802.11n and 64 bits for a floating point number, the feedback overhead amount per receiver is shown in Figure 7 with varying number of quantized precoding matrices. With 16 precoding matrices, the feedback amounts of Distributed Optimization and Distributed Optimization with asymmetric weights are 18 bytes and 34 bytes respectively, while that of Centralized Optimization is 2048 bytes. Considering the maximum size of data packets ($\sim$ 1500 bytes) and the size of control packets ($\sim$ 20 bytes), additional feedback overhead of Distributed Optimization is reasonable. Since the feedback overhead grows only logarithmically we may increase the number of precoding matrices without much additional overhead. However the increased number of precoding matrices may incur additional computational overhead for packet error estimation for each parameter set and slow down the convergence.

6. Conclusion

In this paper, we have proposed three link adaptation algorithms based on SVR that enable adaptive selection of MCS and precoding matrix for multicast traffic. Most of the times Centralized Optimization has outperformed the best fixed MCS policy, but the feedback overhead is so high that we cannot use the algorithm in reality. Per Receiver Optimization reduces the feedback amount significantly, but the throughput performance is poor compared to Centralized Optimization. Lastly we have proposed Distributed Optimization. The feedback overheads of Distributed Optimization and its

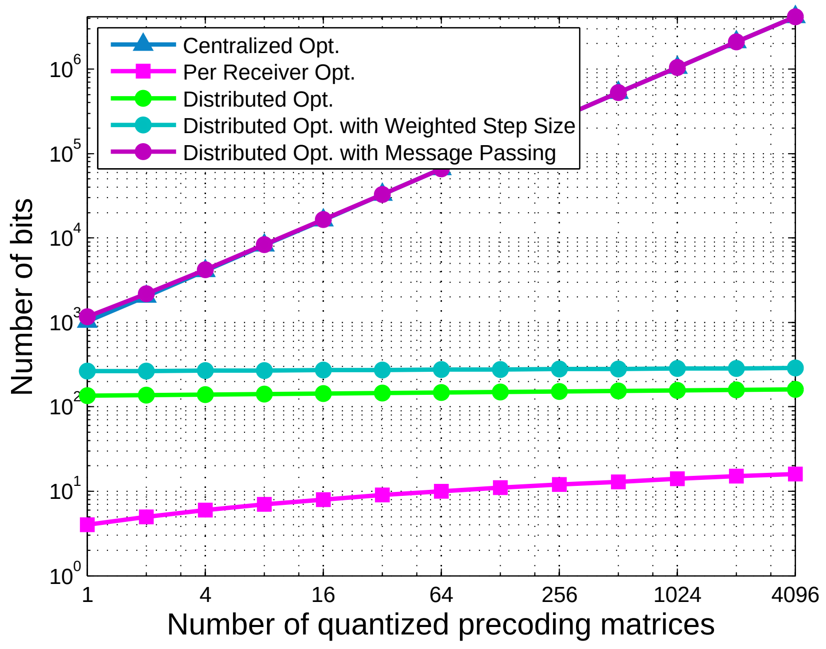

Figure 7: Feedback amount of Centralized Optimization, Per Receiver Optimization and Modified Distributed Optimizations

variation with asymmetric weights are much lower than Centralized Optimization and grow logarithmically as with Per Receiver Optimization. Although the throughput performance of Distributed Optimization is not a big improvement over Per Receiver Optimization, that of Distributed Optimization with asymmetric weights is much higher and close to that of Centralized Optimization. Another variation with local message passing improves the performance even further and very closely matches Centralized Optimization. Although the amount of required information exchange is similar to Centralized Optimization, the actual airtime may be shorter since local message passing can happen simultaneously depending on the locations of receivers.

References

[1] R. C. Daniels, C. Caramanis, R. W. Heath, Jr, Adaptation in convolutionally coded MIMO-OFDM wireless systems through supervised

learning and SNR ordering, IEEE Transactions on Vehicular Technology, Vol. 59, 2010, pp. 114–126.

[2] R. C. Daniels, R. W. Heath, Jr, An online learning framework for link adaptation in wireless networks, Information Theory and Applications Workshop, 2009, pp. 138–140.

[3] S. Yun, C. M. Caramanis, Multiclass support vector machines for adaptation in MIMO-OFDM wireless systems, Allerton Conference on Communication, Control, and Computing, 2009, pp. 1145–1152.

[4] S. Yun, C. M. Caramanis, Reinforcement learning for link adaptation in MIMO-OFDM wireless systems, IEEE Global Telecommunications Conference, 2010, pp. 1–5.

[5] J. Pavon, S. Choi, Link adaptation strategy for IEEE 802.11 WLAN via received signal strength measurement, IEEE International Conference on Communications, Vol. 2, 2003, pp. 1108–1113.

[6] D. Qiao, S. Choi, Fast-responsive link adaptation for IEEE 802.11 WLANs, IEEE International Conference on Communications, Vol. 5, 2005, pp. 3583–3588.

[7] J. Kim, S. Kim, S. Choi, D. Qiao, CARA: Collision-aware rate adaptation for IEEE 802.11 WLANs, IEEE International Conference on Computer Communications., 2006, pp. 1–11.

[8] Y. Park, Y. Seok, N. Choi, Y. Choi, J. M. Bonnin, Rate-adaptive multimedia multicasting over ieee 802.11 wireless lans, IEEE Consumer Communications and Networking Conference, Vol. 1, 2006, pp. 178–182.

[9] A. Basalamah, H. Sugimoto, T. Sato, Rate adaptive reliable multicast MAC protocol for WLANs, IEEE Vehicular Technology Conference, Vol. 3, 2006, pp. 1216–1220.

[10] S. Gorinsky, H. Vin, The utility of feedback in layered multicast congestion control, Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video, 2001, pp. 93–102.

[11] D. J. Love, R. W. Heath, Jr, V. K. N. Lau, D. Gesbert, B. D. Rao, M. Andrews, An overview of limited feedback in wireless communication systems, Vol. 26, 2008, pp. 1341–1365.

[12] D. J. Love, R. W. Heath, Jr, Limited feedback unitary precoding for spatial multiplexing systems, IEEE Transactions on Information Theory, Vol. 51, 2005, pp. 2967–2976.

[13] D. J. Love, R. W. Heath, Jr, Multimode precoding for MIMO wireless systems, IEEE Transactions on Signal Processing, Vol. 53, 2005, pp. 3674–3687.

[14] J. Choi, B. Mondal, R. W. Heath, Jr, Interpolation based unitary precoding for spatial multiplexing MIMO-OFDM with limited feedback, IEEE Transactions on Signal Processing, Vol. 54, 2006, pp. 4730–4740.

[15] N. Khaled, B. Mondal, G. Leus, R. W. Heath, Jr, F. Petre, Interpolation-based multi-mode precoding for MIMO-OFDM systems with limited feedback, IEEE Transactions onWireless Communications, Vol. 6, 2007, pp. 1003–1013.

[16] N. D. Sidiropoulos, T. N. Davidson, Z. Luo, Transmit beamforming for physical-layer multicasting, IEEE Transactions on Signal Processing, Vol. 54, 2006, pp. 2239–2251.

[17] I. H. Kim, D. J. Love, S. Y. Park, Optimal and successive approaches to signal design for multiple antenna physical layer multicasting, IEEE Transactions on Communications, Vol. 59, 2011, pp. 2316–2327.

[18] E. Chiu, V. K. N. Lau, Precoding design for multi-antenna multicast broadcast services with limited feedback, IEEE Systems Journal, Vol. 4, 2010, pp. 550–560.

[19] S. Li, X. Wang, Y. Zhao, Transmit beamforming scheme for multi-antenna multicasting system with limited-rate feedback, IEEE International Conference on Communication Technology, 2010, pp. 909–912.

[20] Y. Wu, H. Zheng, R. Calderbank, S. Kulkarni, H. V. Poor, On optimal precoding in wireless multicast systems, IEEE International Conference on Acoustics, Speech and Signal Processing, 2011, pp. 3068–3071.

[21] S. McCanne, V. Jacobson, M. Vetterli, Receiver-driven layered multicast, Conference proceedings on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '96, ACM, New York, NY, USA, 1996, pp. 117–130.

[22] L. Xiao, M. Johansson, S. P. Boyd, Simultaneous routing and resource allocation via dual decomposition, IEEE Transactions on Communications, Vol. 52, 2004, pp. 1136–1144.

[23] D. P. Palomar, M. Chiang, A tutorial on decomposition methods for network utility maximization, IEEE booktitle on Selected Areas in Communications, Vol. 24, 2006, pp. 1439–1451.

[24] R. Madan, S. Lall, Distributed algorithms for maximum lifetime routing in wireless sensor networks, IEEE Transactions on Wireless Communications, Vol. 5, 2006, pp. 2185–2193.

[25] D. Palomar, M. Chiang, Alternative distributed algorithms for network utility maximization: Framework and applications, IEEE Transactions on Automatic Control, Vol. 52, 2007, pp. 2254–2269.