Design and Implementation of an Integrated Beamformer and Uplink Scheduler for OFDMA Femtocells

Mustafa Y. Arslan *
University of California, Riverside
marslan@cs.ucr.edu

Karthikeyan Sundaresan
NEC Labs America, Inc.
Princeton, NJ, USA
karthiks@nec-labs.com

Srikanth V. Krishnamurthy
University of California, Riverside
krish@cs.ucr.edu

Sampath Rangarajan
NEC Labs America, Inc.
Princeton, NJ, USA
sampath@nec-labs.com

ABSTRACT

Beamforming is a signal processing technique with numerous benefits. Unlike with omni-directional communications, it focuses the energy of the transmitted and/or the received signal in a particular direction. Although beamforming has been extensively studied on conventional systems such as WiFi, little is known about its *practical* impact on OFDMA femtocell deployments. Since OFDMA schedules multiple clients (users) in the same frame (in contrast to WiFi), designing intelligent scheduling mechanisms and at the same time leveraging beamforming, is a challenging task.

Unlike downlink, we show that the integration of beamforming with uplink scheduling projects an interesting trade-off between beamforming gain on the one hand, and the power pooling gain resulting from joint multi-user scheduling on the other hand. This, in turn, makes the uplink scheduling problem even hard to approximate. To address this, we propose algorithms that are simple to implement, yet provably efficient with a worst case guarantee of half. We implement our solutions on a real WiMAX femtocell platform integrated with an eight-element phased array beamforming antenna. Evaluations from both prototype implementation and trace-driven simulations show that our solution delivers throughput gains of over 40% compared to an omni-directional scheme.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

Keywords

OFDMA, Femtocells, WiMAX, Beamforming

*Mustafa Arslan interned in the Mobile Communications and Networking department at NEC Laboratories America Inc., Princeton during this work.

MobiHoc'12, June 11–14, 2012, Hilton Head Island, SC, USA.

1. INTRODUCTION

To meet the demands for increased capacity driven by the exponential growth in mobile data traffic [1], broadband network deployments are moving towards smaller cells – called femtocells – that use Orthogonal Frequency Domain Multiple Access (OFDMA). Femtocells inherit OFDMA and their synchronous access nature from macrocells, which allows for easier deployment and seamless operations with mobile clients.

By focusing the energy in specific directions using antenna arrays, beamforming can serve as a valuable tool in enhancing the performance of OFDMA femtocells. While beamforming is extensively studied for WiFi systems (e.g., [2, 3]), the fundamental differences between WiFi and OFDMA – scheduling of multiple users in the same frame and the synchronous access in OFDMA – render such studies inapplicable to femtocells. The future releases of OFDMA systems (e.g., WiMAX 802.16m [4]) support beamforming as part of the standard. They allow for multiple beams at the base-band level (one beam per user) to be used on the data payload within the same frame (*user-level beamforming*). However, potential beamforming gains are limited as the standards typically support a limited number of antennas at femtocells (e.g., four in WiMAX and LTE-Rel.8). In addition, the specific beamforming implementations vary from one standard to another, making a globally compatible beamforming solution impractical.

A more flexible alternative to user-level beamforming is to employ *frame-level beamforming*, which applies a single beam (common to all users) to the entire frame (not only the data payload but also the control part) at the RF level. Since user-specific beams are not leveraged, such an approach may incur sub-optimal performance for the data payload. This is due to the fact that a femtocell has to find a beam pattern that "fits" all of its clients. However, we identify that frame-level beamforming still has desirable properties: **(a)** it is realizable as a plug-and-play external RF beamformer and hence, is agnostic of the access technology (WiMAX, LTE, etc.); and **(b)** beamforming gain is not limited by the number of antennas supported by the standards. Even more interestingly, we show that frame-level beamforming also offers a unique, complementary benefit for interference mitigation, in a multicell context (discussed in detail later in §3). However in realizing the true potential of frame-level beamforming, an essential first step is to integrate it with the scheduler at each femtocell.

The integration of frame-level beamforming with downlink scheduling can be accomplished by extending existing solutions [5, 6] - by

running the scheduler for each beam and picking the one yielding the best objective. However, the corresponding uplink problem encounters two aspects unique to the uplink that make the problem challenging:

Power pooling vs. beamforming: When multiple users are jointly scheduled in a frame (i.e., the frame resources are shared among clients), each user pools his transmit power on a smaller set of sub-channels (instead of using all sub-channels). This results in a higher user rate per sub-channel and hence higher frame throughput called *power pooling gain*. While beamforming improves a user's rate, using a single beam (common to all users) for a frame may not yield the optimal performance for every user scheduled in the frame. Hence, the higher the number of users scheduled in a frame, the higher is the power pooling gain but lower is the beamforming gain, and vice versa.

Batch scheduling: For a femto base station (BS) to compute the uplink allocation, the scheduling requests have to come from the clients (as is the case in macrocells). Clients contend for making such requests, which they send in response to changes in their buffer occupancies. This incurs both contention delay and overhead, where multiple requests are made for a transmission buffer worth of data. A better option is to allow the BS to poll clients for their buffer occupancies at periodic intervals (e.g., called the polling service in WiMAX [7]). Polling avoids contention and reduces overhead. However, scheduling is now done for a group of frames jointly (batch scheduling), based on client buffer occupancies. More generically, even without polling, batch scheduling allows a user's data to be spread across multiple frames, thereby accentuating power pooling gains and hence, aggregate throughput (details in §4.1).

In this study, we consider the batch scheduling problem on the uplink for non-real-time traffic. While the problem is optimally solvable for omni-directional communications, we show that it is hard to even approximate when integrated with frame-level beamforming, where it also requires the determination of a beam pattern for each frame. In addressing this problem, we make the following contributions:

- We propose simple but efficient algorithms for both continuous and discrete rate functions with a worst case guarantee of $\frac{1}{2}$.
- We implement our solutions on a real WiMAX femtocell platform that is integrated with an eight-element phased array antenna used for beamforming.
- We conduct comprehensive over-the-air evaluations with both a prototype implementation and trace-driven simulations; our results indicate an average gain of 40% over an omni-directional scheme, in practical settings.

The rest of the paper is organized as follows. §2 presents a brief WiMAX overview and related work. In §3 and 4, we describe the motivation and the design of our system. §5 describes our algorithms. §6 contains implementation details and §7 shows evaluation results. We conclude in §8.

2. BACKGROUND

WiMAX: While our solutions apply to OFDMA femtocells in general, our implementation is on WiMAX. Hence, we provide brief background on relevant WiMAX components (details in [7]). OFDMA divides the spectrum into multiple frequencies (sub-carriers) and several sub-carriers are grouped to form a sub-channel. The sub-carriers can be grouped in a contiguous, partially contiguous or fully distributed manner. Partially contiguous grouping, being the mandatory model, is alone implemented on most WiMAX devices and is hence, considered in our work. Here, sub-carriers forming a sub-channel are permuted to allow for a common transmission rate for a user on all sub-channels. This requires only a single channel feedback per user. Permuting also helps average out inter-cell interference.

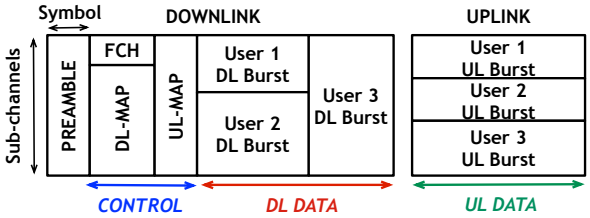


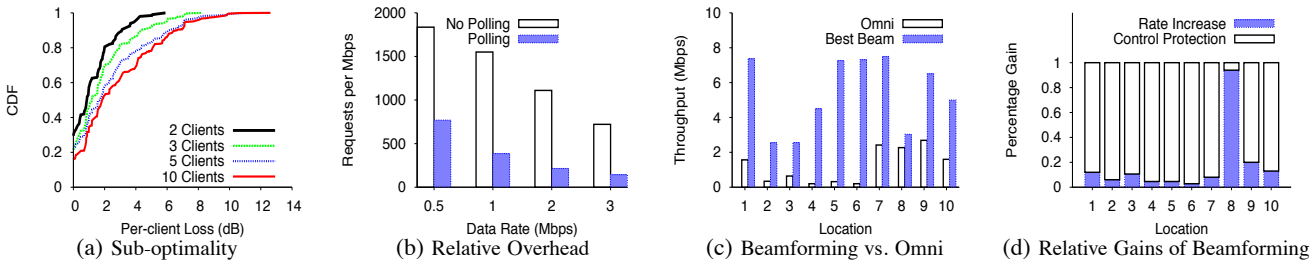
Figure 1: WiMAX Frame Structure.

A WiMAX frame is a two-dimensional template that is scheduled at the MAC with data to/from multiple mobile stations (MSs), across both time (symbols) and frequency (sub-channels). Data to/from users are allocated as rectangular bursts of resources in a frame (see Fig. 1). For e.g., the uplink allocation in Fig. 1 can be visualized to be at the sub-channel granularity. The frame consists of the preamble, control and data payload. While the preamble is used by the MS to lock on to the BS, the control consists of FCH (frame control header) and MAP. The BS, using MAP, indicates the schedule of transmissions both on the downlink and the uplink. The DL-MAP specifies the location of each burst, which MS it is intended for, and what modulation decodes it. Similarly the UL-MAP tells the MS where to place its data in the uplink and how to modulate it.

Beamforming: The ability to transmit (receive) signal energy in (from) specific directions is called beamforming. This is achieved by weighing the signals transmitted (received) from an antenna array in both magnitude and phase. Each applied weight vector generates a specific beam pattern. Such weight vectors can be determined and stored a priori (switched beamforming), or can be computed on-the-fly based on instantaneous channel feedback from the clients (adaptive beamforming).

To achieve low complexity, standards use code-book based beamforming, whereby weight vectors (pre-determined by the code-book) are employed and the one yielding the best SNR at the client is chosen. Since beams are enabled only for the data part, a beam vector suited for each client scheduled in a frame can be chosen, to encode data in the digital signal space (user-level beamforming). In contrast, applying a single beam physically at the RF level (frame-level beamforming), will provide beamforming for the entire frame, including the control part (advantageous in a multicell context as explained in §3). While its beamforming gain for the data part may be sub-optimal compared to user-level beamforming, frame-level beamforming is agnostic of the standards and thus, is not limited by them. Indeed, this has allowed us to integrate an eight-element phased array antenna with a WiMAX BS, although the current femtocells do not support beamforming and the next generation ones (e.g., 802.16m) support only four antennas. Given our experience, we believe that frame-level beamforming has potential in enabling universal beamforming solutions compatible across different standards. We consider frame-level beamforming for uplink, where the beam patterns emphasize reception in desired directions and could either correspond to physical directions covering the 360° azimuth (e.g., [3]), or be based on Gaussian code-books (e.g., [8]).

Related Work: Several works have studied the design of throughput [5, 6, 9, 10] and QoS [11, 12] schedulers for OFDMA. How-



(a) Sub-optimality (b) Relative Overhead (c) Beamforming vs. Omni (d) Relative Gains of Beamforming

Figure 2: Sub-optimality of frame-level beamforming (a) and relative overhead with batch scheduling (b).

ever, their focus has been on per-frame scheduling. While batch scheduling is not very different from per-frame scheduling on the downlink (in terms of optimization), the ability to leverage power pooling on the uplink makes the problem challenging and has not been addressed. While some studies (e.g., [13]) have looked at the design of polling intervals specifically for the uplink of WiMAX, they have not addressed the batch scheduling problem.

User-level beamforming has been jointly optimized with per-frame scheduling for OFDMA systems [14, 15]. However, frame-level beamforming and its various benefits have not been explored; its integration with uplink batch scheduling and the challenges it encounters have also not been considered. We contribute by designing and implementing an efficient uplink scheduler that integrates frame-level beamforming with batch scheduling for OFDMA.

3. MOTIVATION

We motivate the importance of frame-level beamforming and batch scheduling for the uplink using measurements from an experimental testbed. We defer however, a detailed description of the testbed to §6.

Small Loss Compared to User-level Beamforming: One drawback of frame-level beamforming is the constraint of using a single beam for all the users scheduled in a frame, thereby potentially limiting the per-user beamforming gain. In strong line-of-sight environments, where very few beams work well for each client, the sub-optimality of such a common beam could be significant. To understand this sub-optimality indoors, we placed a client at thirty locations in the building and recorded the uplink SNR (as measured at the BS) with various beams for each of these locations. Then, different subsets of client locations were grouped together to emulate different sets of clients being scheduled in the same frame. For each of these subsets, we chose a single beam (i.e., frame-level beamforming) with the following criteria: the chosen beam minimizes the aggregate loss (in terms of SNR) compared to the case where each client operates with its best beam (i.e., user-level beamforming). Here, the best beam of a client refers to the beam pattern yielding the highest received SNR from that particular client.

We show the CDF of the loss per client (in dB) in Fig. 2(a). It is seen that even when five clients (reasonably high for femtocells) are multiplexed in the same frame, the loss is less than 2 dB for over 60% of the scenarios, indicating that the sub-optimality of using a common beam for a frame is not significant. This can be attributed to the multipath nature of the indoor environment, which allows for multiple beams to perform reasonably well for a given location, thereby increasing the possibility of finding a mutually *good* beam for multiple clients. However, as more and more clients are multiplexed together, the sub-optimality will tend to increase.

Reduced Overhead with Batch Scheduling: Client requests for resources on the uplink typically experience contention, delay and overhead, with redundant requests being sent out prior to transmission. To overcome these issues, typical OFDMA systems allow the BS to poll clients for data periodically (the period adjusted based on traffic variations). We profile the relative overhead between client-initiated and BS-polled requests, by measuring the number of requests generated for a fixed amount of data for a single client, in Fig. 2(b). We see that BS polling reduces the overhead by as much as five folds. Further, the overhead does not grow with input rate unlike with client-initiated requests. With polling, buffer occupancy information of clients is available only once every polling interval. During this interval, frames can be scheduled one at a time or jointly as a batch, with the latter yielding better performance owing to joint optimization.

Improved Decoding of Control Part: One not-so-obvious benefit of frame-level beamforming is its ability to improve the decoding of the control part, through improved beamforming gain. As described earlier in §2, the control part carries vital information that helps clients decode and transmit their data payload. If the control part in a frame cannot be decoded by the clients (e.g., due to interference from other cells), the frame fails to deliver even a single byte of data. To understand the impact of interference on the control part, we experiment with two BSs that have one client associated with each. To make sure that the interference impact is only on the control part, we configure the two BSs to use orthogonal sets of sub-channels for their data payloads (i.e., data payload is protected from interference). While one BS is omni-directional, the other is equipped with a beamforming antenna. The client associated with the beamforming BS is placed at various locations to generate varying interference scenarios. In each of the locations, we measure the downlink throughput observed by the client using each beam pattern.

The throughputs delivered by the best beam and the omni-directional beam are presented in Fig. 2(c). We see that there is a lot of room for improvement with beamforming ($\approx 7X$ on average) even when the data parts are immune to interference. There are two factors behind the observed throughput: (i) the rate (modulation) increase due to beamforming gain on the data part, and (ii) improved resilience of the control part and hence, reduced frame losses. Fig. 2(d) presents the relative contributions of these two components. We see that the improved decoding of the control part owing to frame-level beamforming is the dominant component and provides the bulk of the gain in most scenarios. This clearly indicates that the performance of existing resource isolation solutions such as [16] can be enhanced even further, if the control part decoding is improved with the help of frame-level beamforming[1]. To realize the benefits of frame-level beamforming[2] and batch scheduling, we next present our design of an integrated beamformer and uplink scheduler – *iBUS*.

[1]While a scheme that combines an external common beam on the control part with user-level beamforming on the data part may be ideal, the varying size of the control part prevents its realization due to lack of standards support.

[2]Hereafter referred to as simply beamforming.

4. DESIGN OF IBUS

4.1 Overview

In batch scheduling, given a polling interval (F frames) and the buffer occupancies of the clients, the problem is to effectively pack (schedule) the out-standing client data jointly over F frames, with potentially different schedules across frames. When batch scheduling is integrated with beamforming, in addition to packing, one also needs to determine a beam for each frame in the batch. Picking a beam for a frame and a subset of clients (for data packing) are inter-twined since the beam choice for a frame will affect the rate supported by clients in that frame and hence, the amount of data that can be packed. Similarly picking a subset of clients will influence the beam choice since the sub-optimality of beamforming will vary depending on the subset of clients.

Two aspects make the integrated scheduling problem challenging: **(a) Power Pooling:** In OFDMA, when multiple clients are multiplexed in the same uplink frame, they pool their powers on a smaller set of sub-channels, thereby potentially supporting higher rates per sub-channel. Batch scheduling across multiple frames enhances power pooling further - spreading a client's data across multiple frames will require fewer sub-channels per frame, resulting in higher power pooling gain per frame. **(b) Beamforming:** Spreading data across multiple frames for each client will maximize the power pooling gain. However, this results in more clients being scheduled in the same frame, making it harder to find a *good* common beam for each frame; this increases the sub-optimality of beamforming. The scheduler's objective is to strike the right tradeoff between these two components.

Remarks: We note that unlike uplink, in the downlink, there is no notion of power pooling (due to fixed BS total transmit power) and no signaling overhead that may necessitate batch scheduling. Hence, downlink is limited to per-frame scheduling, solutions for which exist even when a user's rate varies across sub-channels [5], which is typically considered a hard problem. Further, the downlink can be integrated with beamforming by simply running existing schedulers for various beams and picking that schedule and beam, which maximizes an objective function. However for the uplink, the combination of batch scheduling and power pooling with beamforming, makes the problem challenging even when a user's rate does not vary across sub-channels.

4.2 Formulation and Hardness

The integrated uplink scheduling problem can be formally stated as the following non-linear integer program.

$$\text{ISP:} \quad \text{Maximize} \sum_i U_i(R_i)$$

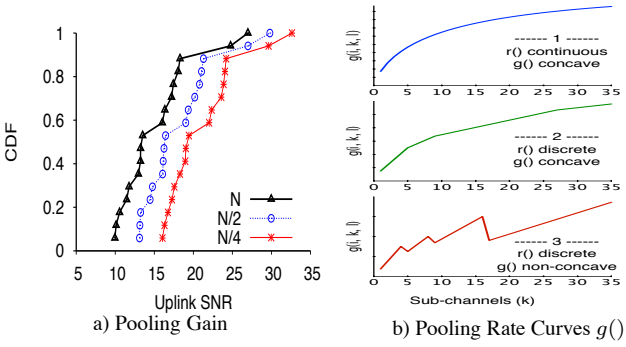
$$R_i = \sum_{j,k} x_{i,j,k} \left\{ \sum_{\ell} y_{j,\ell} \cdot k \cdot r_{i,k,\ell} \right\} \leq B_i, \ \forall i \in \mathcal{K}$$

$$\sum_{\ell} y_{j,\ell} \leq 1, \ \forall j \in \mathcal{F}$$

$$\sum_{i,k} x_{i,j,k} \cdot k \leq N, \ \forall j$$

$$\sum_{k} x_{i,j,k} \leq 1, \ \forall i, j$$

where $x_{i,j,k}$ and $y_{j,\ell}$ are binary indicator (output) variables indicating the assignment of k contiguous sub-channels in frame j to user i and the assignment of beam ℓ to frame j respectively. $\mathcal{K}, \mathcal{F}, \mathcal{L}$ indicate the set of users, frames in the batch and the available beams, respectively. The objective is to maximize the aggregate client utility, where the utility function can be any concave function (e.g., logarithmic function captures proportional fairness [9]) of the data scheduled and can vary from one client to another. The first constraint limits the net allocation to a user to be limited to its buffer occupancy B_i (in bits). Further, it indicates the inter-dependence of power pooling (# sub-channels assigned, k) and beam chosen (ℓ) on the user's rate per sub-channel ($r_{i,k,\ell}$ in bits). The remaining constraints are all conservation constraints indicating the (i) assignment of a single beam per frame, (ii) net allocation of sub-channels being limited to N in a frame, and (iii) assignment of one contiguous set of sub-channels to a user in a frame, respectively.



a) Pooling Gain b) Pooling Rate Curves $g()$

Figure 3: Power Pooling.

We now show that it is hard to even obtain a PTAS (polynomial time approximation scheme) for the ISP problem. Specifically, we have the following result.

THEOREM 1. *For some $\epsilon > 0$, there is no $(1-\epsilon)-$ approximation algorithm for ISP unless P=NP.*

PROOF. Scheduling of multiple sub-channels in a frame across users with finite buffers and varying rates across sub-channels (SCF) has been studied in [5] and shown to not admit a PTAS, using a reduction from 3-bounded 3-matching problem. We next show that SCF is a special case of ISP.

Consider a simpler version of ISP, where the beam choices for the frames do not have to be determined but are given a priori (ℓ_j for frame j). Further, consider only one sub-channel per frame ($N = 1$) available for allocation in each of the F frames. Since a user's (i) rate will vary from one beam to another and across frames ($r_{i,1,\ell_j}, \forall j$), the resulting ISP problem is now equivalent to an SCF problem with F sub-channels (mapping from frames in ISP) in a frame and users with finite buffers and varying rates across the sub-channels, where the rate of user i on sub-channel c is $r_{i,c,\ell_c}, c \in [1, F]$. Hence, the desired result. □

4.3 Components of iBUS

To address the above problem, we design and implement iBUS, which consists of the following key components.

Measurement of Beam SNRs: If there are $|\mathcal{L}|$ beam patterns, $|\mathcal{L}|$ frames are used for measurement (one for each pattern). On the uplink part of each frame, the BS schedules all the active users such that their data spans all the sub-channels (number of time symbols allocated varies across users). Such a schedule can be visualized as user bursts being arranged vertically rather than horizontally as in Fig. 1 (see uplink). The resulting SNR (ρ) for each user with each beam is measured directly at the BS and corresponds to that when power is split on all sub-channels ($\rho(i, N, \ell), \forall i, \ell$).

Rate Estimation using Power Pooling: Rate tables are determined separately to identify the best modulation and coding rate

(MCS) to employ for a given SNR that yields a specific loss rate (e.g., < 0.1). However, since the rate would vary with the number of sub-channels allocated (due to power pooling), a direct SNR measurement for each possible set of sub-channels for each beam ($\rho(i,k,\ell)$) would constitute significant overhead. Hence, based on the SNR measurement from all sub-channels ($\rho(i,N,\ell)$), we extrapolate the SNR for other subsets of sub-channels, taking power pooling into account. For e.g., if half of the sub-channels are allocated to a user, then its resulting SNR $\rho(i, \frac{N}{2}, \ell) = \rho(i,N,\ell) + 3$ (in dB). Since the power gets distributed over $\frac{1}{2}$ of sub-channels, the per sub-channel power is doubled, corresponding to $+3$ in dB scale. In general, $\rho(i, \frac{N}{\alpha}, \ell) = \rho(i,N,\ell) + 10\log_{10}(\alpha)$, $\alpha \in [1, \frac{N}{N-1}, \ldots, \frac{N}{1}]$. Fig. 3(a) validates the modeling of power pooling by measuring the SNR in practice for a client, when data is transmitted on different sets of sub-channels. As expected, as we reduce the number of sub-channels by a factor of half, the SNR at each client location increases by ≈ 3 dB.

From $\rho(i,k,\ell)$, one can obtain $r_{i,k,\ell}$ using the rate table. Essentially, every user (i) has a power-pooling rate curve ($g(i,k,\ell)$) as a function of number of sub-channels (k) and the beam chosen (ℓ). Depending on the nature of SNR-rate mapping, these curves may or may not be concave. If continuous Shannon rates are employed, then we have $g(i,k,\ell) = k \cdot \log_2(1+\frac{N}{k}\rho(i,N,\ell))$, which is a concave function (see Fig. 3(b)). If discrete rate tables are used then $g(i,k,\ell) = k \cdot r_{i,k,\ell}$ results in a piece-wise linear (between SNR thresholds) function, where the mapping $r_{i,k,\ell} \leftarrow \frac{N}{k}\rho(i,N,\ell)$ is determined by the rate table. Depending on the rate-SNR thresholds, $g(i,k,\ell)$ may still be concave or not (Fig.3(b) shows $g()$ for two such examples). As we shall see in §5, depending on the nature of $g()$ (concave or arbitrary), different algorithms will be required.

Buffer Estimation using Polling: The BS polls all its clients towards the end of the current polling interval to estimate their buffer occupancies for scheduling during the next polling interval. Note that if the total buffer occupancy of all clients is less than the total frame resources in the polling interval, then there will be underutilization. Further, the schedules computed for the polling interval will be relevant only if the SNRs are relatively static during that interval. Since we focus on indoor femtocell deployments with stationary clients, the coherence time of SNRs are reasonably high (several frames) as we have experimented on our prototype (details in §6).

Schedule Determination: Once the BS has all the information needed to determine the batch schedule, it executes its algorithms. The batch schedule consists of two parts. The first part is a schedule for $F - |\mathcal{L}|$ frames ($|\mathcal{L}| << F$), where the ISP problem is solved over $F - |\mathcal{L}|$ frames. The latter part is a schedule for the remaining $|\mathcal{L}|$ frames, and serves as the measurement interval to obtain the SNR information for the next polling interval. Hence, in the latter part, the beam choices for the $|\mathcal{L}|$ frames are fixed (one beam each) during measurement, and allocation is done only across time symbols by employing the same algorithm used for solving ISP, albeit with some fixed variables. While the main purpose of the second part is measurement, one can also optimize the packing of client data that remains after the first $F - |\mathcal{L}|$ frames, given the beam choices. We next describe our solution to the ISP problem, where we determine the data packing along with the beam choices over a given set of frames.

5. ALGORITHMS IN IBUS

Given the hardness of the scheduling problem ISP, we focus on algorithms that have a provable worst case guarantee and are simple to implement. Our algorithms can be classified based on the nature of the power pooling rate curve.

Algorithm 1 Integrated Scheduler: iBUS1

1: INPUT: Buffer occupancy B_i, $\forall i \in \mathcal{K}$; rates $g(i,k,\ell)$, $\forall i,k \in [1,N], \ell \in \mathcal{L}$
2: OUTPUT: Beam choices $\ell_j, \forall j \in \mathcal{F}$; per-frame user allocation A_{ij}, $\forall i,j$
3: **for** $f \in [1 : |\mathcal{F}|]$ **do**
4: $O_{max} = 0$
5: **for** $\ell_f \in [1 : |\mathcal{L}|]$ **do**
6: $D_i \leftarrow B_i, \forall i$; $a_{ij} = A_{ij}, \forall i,j \in [1, f-1]$ and $a_{ij} = 0$ $\forall i, j = f$
7: **for** $k = 1 : N$ **do**
8: $i^* = \arg\max_{i \in \mathcal{K}} \Big\{ U_i \left(\sum_{j=1}^{f} g(i,a_{ij},\ell_j) + \min\left(g(i,a_{if}+1,\ell_f) - g(i,a_{if},\ell_f), D_i\right)\right) - U_i(\sum_{j=1}^{f} g(i,a_{ij},\ell_j)) \Big\}$
9: **if** $i^* \neq \emptyset$ **then**
10: $D_{i^*} \leftarrow D_{i^*} - \min\left(g(i^*, a_{i^*f}+1, \ell_f) - g(i^*, a_{i^*f}, \ell_f), D_{i^*}\right)$
11: $a_{i^*f} \leftarrow a_{i^*f} + 1$
12: **else** break **endif**
13: **end for**
14: $O_f = \sum_{i \in \mathcal{K}} U_i \left(\sum_{j=1}^{f} g(i,a_{ij},\ell_j) \right) - U_i \left(\sum_{j=1}^{f-1} g(i,a_{ij},\ell_j) \right)$
15: **if** $O_f > O_{max}$ **then**
16: $O_{max} = O_f$; $\ell_{max} = \ell_f$; $A_{if} = a_{if}, \forall i$
17: **endif**
18: **end for**
19: $\ell_f = \ell_{max}$; ; $B_i \leftarrow B_i - \min(g(i, A_{if}, \ell_f), B_i), \forall i$
20: **end for**

5.1 Concave Rate Curves

We present two greedy algorithms called iBUS1 and iBUS2, outlined in Algs. 1 and 2, respectively. In iBUS1, resource allocation and beam selection are performed one frame at a time in the batch, sequentially. For a given frame (iteration), based on the remaining data available for the users, resource allocation is performed for every choice of the beam, and the beam yielding the best aggregate marginal utility for the frame is chosen (steps 14-16). For resource allocation within each frame given a beam (steps 6-13), iBUS1 assigns each sub-channel to the user who yields the highest marginal utility taking the users' buffer status into account (step 8). For concave utility functions, such an allocation is indeed optimal. Further, allocation based on the marginal utility also helps multiplex multiple users in the same frame thereby maximizing the benefits of power pooling. Once the best beam and its corresponding resource allocation are determined (steps 14-17) for the current frame, the user buffers are updated (step 19) and the procedure is repeated for the remaining frames in the batch, sequentially. iBUS1 has a time complexity of $O(|\mathcal{F}||\mathcal{L}||\mathcal{K}|N)$.

The key difference between iBUS1 and iBUS2 is that while iBUS1 performs resource allocation within each frame in isolation, iBUS2 performs joint resource allocation across all frames considered in an iteration. Hence, in addition to enabling power pooling within each frame, it allows a user's data to be spread across multiple frames thereby pooling the user's power across frames as well. This, in turn, results in a more efficient data packing and hence, higher aggregate throughput. Specifically, when considering a beam for a given frame (f) during a considered iteration, resource allocation is performed for all frames till the current frame jointly (steps 6-13) to determine the resulting utility for the beam. The beam yielding the highest utility is then chosen for that frame (steps 15-19). Hence, each iteration of resource allocation now involves de-

Algorithm 2 Integrated Scheduler: iBUS2

1: INPUT: Buffer occupancy B_i, $\forall i \in \mathcal{K}$; rates $g(i,k,\ell)$, $\forall i,k \in [1,N], \ell \in \mathcal{L}$
2: OUTPUT: Beam choices $\ell_j, \forall j \in \mathcal{F}$; per-frame user allocation A_{ij}, $\forall i,j$
3: **for** $f \in [1:|\mathcal{F}|]$ **do**
4: $O_{max} = 0$
5: **for** $\ell_f \in [1:|\mathcal{L}|]$ **do**
6: $D_i \leftarrow B_i, \forall i; \quad A_{ij} = 0\ \forall i, j = [1,f]; k_j = N, \forall j \in [1,f]$
7: **while** $k_j \neq 0, \exists j \in [1,f]$ **do**
8: $(i^*,j^*) = \arg\max_{i\in\mathcal{K}, j\in[1,f]} \Big\{ U_i\left(\sum_{m=1}^{f} g(i,A_{im},\ell_m) + \min\{g(i,A_{ij}+1,\ell_j) - g(i,A_{ij},\ell_j), D_i\}\right) - U_i\left(\sum_{m=1}^{f} g(i,A_{im},\ell_m)\right)\Big\}$
9: **if** $(i^*,j^*) \neq \emptyset$ **then**
10: $D_{i^*} \leftarrow D_{i^*} - \min\left(g(i^*, A_{i^*j^*}+1, \ell_{j^*}) - g(i^*, A_{i^*j^*}, \ell_{j^*}), D_{i^*}\right)$
11: $A_{i^*j^*} \leftarrow A_{i^*j^*} + 1; \quad k_{j^*} \leftarrow k_{j^*} - 1$
12: **else** break **endif**
13: **end while**
14: **end for**
15: $O_f = \sum_{i\in\mathcal{K}} U_i\left(\sum_{j=1}^{f} g(i,A_{ij},\ell_j)\right)$
16: **if** $O_f > O_{max}$ **then**
17: $O_{max} = O_f; \ell_{max} = \ell_f$
18: **endif**
19: $\ell_f = \ell_{max}$
20: **end for**

termining not only the user to whom a sub-channel must be allocated but also in which frame ($\in [1,f]$, step 8). Note that once the beam is chosen for a given frame, user buffers do not have to be updated for the next frame since resource allocation will then be re-performed. Thus, for every newly considered frame, user buffers are re-initialized (step 6). Essentially, resource allocation for all frames till the one under consideration is done mainly for the purpose of determining the beam yielding the highest utility for the considered frame. The actual resource allocation is the one that is computed during the final frame of iteration, where the allocation for all the frames in the batch is jointly determined along with the beam for that frame. While iBUS2 enables power pooling both within and across frames, the joint resource allocation across frames results in an additional time complexity of $O(|\mathcal{F}|)$ and hence a net time complexity of $O(|\mathcal{F}|^2|\mathcal{L}||\mathcal{K}|N)$.

While iBUS2 results in a better performance than iBUS1, we now show that *even* iBUS1's worst case performance can be bounded. We provide some definitions on matroid and sub-modularity that are relevant for the proof.

Partition Matroid: Consider a ground set Ψ and let S be a set of subsets of Ψ. S is a matroid if, (i) $\emptyset \in S$, (ii) If $P \in S$ and $Q \subseteq P$, then $Q \in S$, and (iii) If $P, Q \in S$ and $|P| > |Q|$, there exists an element $x \in P \backslash Q$, such that $Q \cup \{x\} \in S$. A partition matroid is a special case of a matroid, wherein there exists a partition of Ψ into components, $\phi_1, \phi_2, \ldots$ such that $P \in S$ if and only if $|P \cap \phi_i| \leq 1,\ \forall i$.

Sub-modular function: A function $f(\cdot)$ on S is said to be sub-modular and non-decreasing if $\forall x, P, Q$ such that $P \cup \{x\} \in S$ and $Q \subseteq P$ then,

$$\begin{aligned} f(P\cup\{x\}) - f(P) &\leq f(Q\cup\{x\}) - f(Q) \\ f(P\cup\{x\}) - f(P) &\geq 0, \quad \text{and } f(\emptyset) = 0 \end{aligned}$$

THEOREM 2. *iBUS1's worst case performance is within $\frac{1}{2}$ of the optimum.*

PROOF. The sub-optimality of maximizing a sub-modular function over a partition matroid using a greedy algorithm of the form $x = \arg\max_{x\in\phi_i} f(P\cup\{x\}) - f(P)$ in every iteration was shown to be bounded by $\frac{1}{2}$ in [17]. We will now show that iBUS1 is such an algorithm, with our scheduling objective corresponding to a sub-modular function to obtain the desired result.

Let the ground set be composed of the following triplets.

$$\Psi = \{(j,\ell_j,\mathbf{a}_j): \quad j \in [1:|\mathcal{F}|], \ell_j \in [1:|\mathcal{L}|], \mathbf{a}_j = \cup_{i\in\mathcal{K}} a_{ij} \text{ s.t. } \textstyle\sum_i a_{ij} \leq N\}$$

Now Ψ can be partitioned into $\phi_j = \{(j,\ell_j,\mathbf{a}_j) : \ell_j \in [1:|\mathcal{L}|], \mathbf{a}_j = \cup_{i\in\mathcal{K}} a_{ij} \text{ s.t. } \sum_i a_{ij} \leq N\}, \forall j$. Let S be defined on Ψ as a set of subsets of Ψ such that for all subsets $P \in S$, we have (i) if $Q \subseteq P$, then $Q \in S$; (ii) if element $x \in P\backslash Q$, then $Q \cup \{x\} \in S$; and (iii) $|P \cap \phi_j| \leq 1,\ \forall j$. This means that S is a partition matroid. Further, any $P \in S$ will provide a feasible schedule with at most one feasible allocation and beam choice for each frame. Note that, every feasible schedule is contained in S by definition since it allows for the possibility of all feasible channel allocations and beam choices to be selected for each frame in the schedule. This, in turn, allows the partition matroid to capture our scheduling problem. Our scheduling objective is given as,

$$\begin{aligned} f(P) &= \sum_{i\in\mathcal{K}} \mu_i(P) \\ \text{where, } \mu_i(P) &= U(\min\{\sum_{j:(j,\ell_j,\mathbf{a}_j)\in P} g(i,a_{ij},\ell_j), B_i\}) \end{aligned}$$

It can be seen that if $Q \subseteq P$, then $\mu_i(Q) \leq \mu_i(P)$. Hence, for an element $(j,\ell_j,\mathbf{a}_j)$ such that $P \cup \{(j,\ell_j,\mathbf{a}_j)\}$ forms a valid schedule, it follows that $f(P\cup\{(j,\ell_j,\mathbf{a}_j)\}) - f(P) \leq f(Q\cup\{(j,\ell_j,\mathbf{a}_j)\}) - f(Q)$. This results both from the potential buffer limitation in subsequent frames as well as the concave nature of the utility function. This establishes that the function $f(P)$ is indeed sub-modular. Further, our scheduling problem aims to maximize this non-decreasing sub-modular function over a partition matroid. Hence, if the optimal allocation corresponding to every beam were given by some oracle for each frame, then picking the beam yielding the highest marginal utility for a frame in iBUS1 (steps 14-17) would correspond to determining

$$(j,\ell_j^*,\mathbf{a}_j^*) = \arg\max_{(j,\ell_j,\mathbf{a}_j)\in\phi_j} \{f(P\cup\{(j,\ell_j,\mathbf{a}_j)\}) - f(P)\}$$

Thus, the sub-optimality of $\frac{1}{2}$ would then follow from the result in [18].

Note that, there are exponential allocations possible that satisfy $\sum_i a_{ij} \leq N$ for every beam choice. Hence, bypassing the oracle assumption would require us to find the optimal allocation for a given beam, which is a problem in itself. However, since the utility functions are concave, this would correspond to a concave optimization problem, which iBUS1 solves optimally by employing a steepest gradient-like approach based on maximum marginal utility (steps 7-13). Hence, iBUS1 is able to bound its sub-optimality by $\frac{1}{2}$. □

5.2 Arbitrary Rate Curves

For arbitrary power pooling rate curves, we present the following algorithm iBUS3, which is a modified version of iBUS1. Specifically, iBUS3 is similar to iBUS1 in picking the beam yielding the highest marginal utility for each frame. However, its frame allocation for a given beam choice is significantly different. This is because if the power pooling rate curves are not concave, then allocations based on marginal utilities will not work. Hence, allocations to users in each frame must be made as a subset of sub-channels instead of individual sub-channels. Thus, we model the allocation problem for a given frame and beam choice (f,ℓ_f) with arbi-

Algorithm 3 Integrated Scheduler: iBUS3

1: INPUT: Buffer occupancy B_i, $\forall i \in \mathcal{K}$; rates $g(i,k,\ell)$, $\forall i,k \in [1,N], \ell \in \mathcal{L}$
2: OUTPUT: Beam choices $\ell_j, \forall j \in \mathcal{F}$; per-frame user allocation A_{ij}, $\forall i,j$
3: **for** $f \in [1:|\mathcal{F}|]$ **do**
4: $O_{max} = 0$
5: **for** $\ell_f \in [1:|\mathcal{L}|]$ **do**
6: $D_i \leftarrow B_i, \forall i$; $a_{ij} = A_{ij}, \forall i,j \in [1,f-1]$ and $a_{ij} = 0$ $\forall i, j = f$
7: Run an FPTAS for multiple-choice knapsack problem $FA(f,\ell_f)$; Obtain $a_{if}, \forall i$
8: $O_f = \sum_{i\in\mathcal{K}} U_i\left(\sum_{j=1}^{f} g(i,a_{ij},\ell_j)\right)$ $-U_i\left(\sum_{j=1}^{f-1} g(i,a_{ij},\ell_j)\right)$
9: **if** $O_f > O_{max}$ **then**
10: $O_{max} = O_f; \ell_{max} = \ell_f; A_{if} = a_{if}, \forall i$
11: **endif**
12: **end for**
13: $\ell_f = \ell_{max};;\ B_i \leftarrow B_i - \min(g(i,A_{if},\ell_f), B_i), \forall i$
14: **end for**

trary rate curves as the following multiple choice knapsack problem (MCKP), which is a NP-hard problem in itself.

$$FA(f,\ell_f): \quad \text{Maximize} \sum_i \sum_{k\in\mathcal{N}_i} u_{ik}(f,\ell_f)x_{ik}(f)$$
$$\text{s.t.} \sum_i \sum_{k\in\mathcal{N}_i} k \cdot x_{ik}(f) \le N$$
$$\sum_{k\in\mathcal{N}_i} x_{ik}(f) = 1;\ \forall i$$

where $x_{ik}(f) \in \{0,1\}$ and $\mathcal{N}_i \in \{0,1,\ldots,N\}$.

Essentially, there are $N+1$ allocations ($\mathcal{N}_i$) possible for each user and the MCKP problem is to pick exactly one allocation for each user such that the total frame allocation does not exceed N. The user's profit or utility for each allocation is computed based on the current and previous frames' allocations as follows:

$$u_{ik}(f,\ell_f) = \sum_{i\in\mathcal{K}} U_i\left(\sum_{j=1}^{f-1} g(i,a_{ij},\ell_j) + \min\{g(i,x_{ik}(f),\ell_f), B_i\}\right)$$

where $a_{ij} = k:$ s.t. $x_{ikj} = 1,\ \forall i,j \in [1,f-1]$. Note that the user's buffer occupancy is automatically accommodated in its profit. Now a FTPAS based on dynamic programming from [19] can be employed to efficiently solve $FA(f,\ell_f)$ to within $(1-\epsilon)$ of the optimal at a time complexity of $O(\frac{|\mathcal{K}|N}{\epsilon})$. This results in a net time complexity of $O(\frac{|\mathcal{F}||\mathcal{L}||\mathcal{K}|N}{\epsilon})$ for iBUS3.

In establishing a bound on the worst case performance of iBUS3, we employ the following lemma proved in [20].

LEMMA 1. *If the sub-modular function is only $\alpha-$ approximable, then the approximation guarantee of greedy maximization changes to $\frac{\alpha}{p+\alpha}$, where the maximization is subject to a $p-$ independence system.*

THEOREM 3. *iBUS3 provides an approximation guarantee of $\frac{1}{2}-\epsilon$.*

PROOF. Note that matroids are 1-independence systems (see [21] for an exposition). Given that the FPTAS yields a $(1-\epsilon)$ approximation in computing the frame allocation and hence the sub-modular function, we have $\alpha = 1-\epsilon$ and hence the resulting performance guarantee of iBUS3 reduces to $\frac{1}{2}-\epsilon$. □

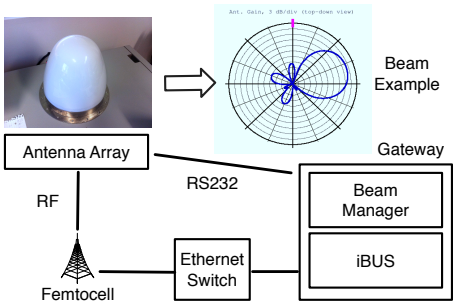


Figure 4: iBUS system architecture.

Remarks: While we have shown that our algorithms have an approximation guarantee of half for both concave and arbitrary power pooling rate curves, one might wonder how much further can this guarantee be improved without increasing the complexity significantly. Unfortunately, the answer is not optimistic. We had shown that ISP does not admit a PTAS and hence a $(1-\epsilon)$ approximation. Further, recall from §4.2 that single frame scheduling with varying sub-channel rates and finite user buffers (SCF) is a special case of our ISP problem. In [5], it was shown that one can improve the guarantee for SCF to 0.63 albeit through a complex LP relaxation and rounding procedure, where an exponential number of subsets is involved in the LP formulation. In light of this, we believe that our greedy algorithms strike a good balance between both performance and complexity. Further, §7 reveals their close-to-optimal performance in practice, on average, compared to their worst case guarantees of half.

6. IMPLEMENTATION

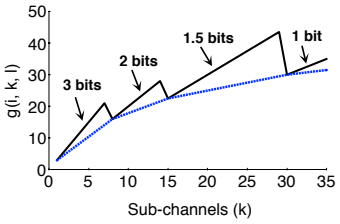
Fig. 4 depicts our system design. The testbed consists of a WiMAX femtocell platform and commercial clients (USB dongles attached to laptops). Our experiments are conducted over-the-air (10 MHz bandwidth) with an experimental license from FCC. We have an eight-element phased array antenna designed by Fidelity Comtech, attached via a RF cable to the BS. The array generates 16 beam patterns of 45° each, spaced 22.5° apart to cover the entire azimuth of 360°. We also have a Linux PC for the gateway functionality required for WiMAX. In WiMAX, the gateway manages the service flows needed to transmit/receive data to/from the clients.

The BS and the gateway communicate using sockets via an Ethernet switch. When the BS decodes user data on the uplink, it passes it to the application at the gateway (*iperf*) via the switch. In addition, the switch is used to pass the client buffer information and the measured beam SNRs to the gateway. The gateway both implements our algorithms in Java, and at the same time controls the array by a serial port (RS232) application that we have developed in C.

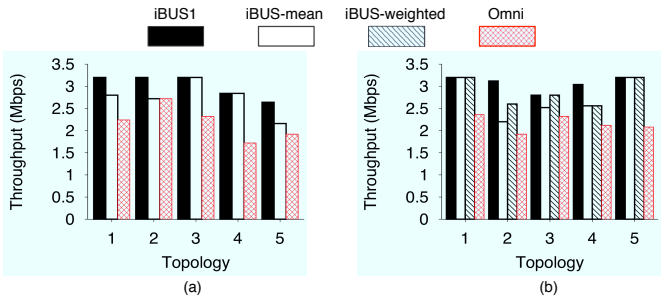
When the gateway receives the buffer and SNR information, it executes the iBUS algorithms and sends the computed batch schedule to the BS, one frame at a time. A beam corresponding to a frame in the batch is applied to the array just before the corresponding schedule is sent to the BS. However, note that there is inevitably a delay before a particular beam is applied by the antenna following the gateway command. We measured this delay to be $\approx$ 6ms on average (8ms max.). Since each WiMAX frame is 5ms, applying one beam for a single frame is difficult. To circumvent this, each schedule corresponding to a frame in the batch is *extended* over 20 frames at the BS, by taking into account the resources from the 20 frames

Modulation	Bits per Carrier	SNR
QPSK$^{1/2}$	1	10 dB
QPSK$^{3/4}$	1.5	12 dB
16QAM$^{1/2}$	2	15 dB
16QAM$^{3/4}$	3	18 dB

(a) MCS Thresholds



(b) $g()Transformation$



(c) Equal rates (left), different rates (right)

Figure 5: Parameters and performance results with iBUS in real experiments.

jointly. In addition, each beam l_i is applied 10ms (two frames) prior to schedule i. With this a priori beam application, when the BS starts executing schedule i, the hardware is ready to transmit with beam l_i. One can alternatively implement the algorithms and the beam management component at the BS, for tighter beam and schedule synchronization. However, this would add complexity and overhead to the BS, making the deployment less practical. Our design realizes a light-weight, standards-compatible OFDMA beamforming system that does not result in significant processing overhead at the BS. In addition, since the gateway is agnostic of the underlying technology, we believe that it can flexibly be extended to other OFDMA systems such as LTE, which use a similar notion of sub-channels.

In our implementation, the BS polls the clients for buffer occupancies every 100 frames, thereby forming the batch scheduling interval. Since there are 16 beam patterns, 16 frames are used to measure the beam SNRs of each client. Selecting a large polling interval will not sufficiently capture the channel diversity (e.g., the SNRs can change in the interim), making the computed schedule sub-optimal. Conversely, a short polling interval will not sufficiently reduce the contention overhead for buffer requests. We have verified in our experiments that the SNRs with each beam are relatively stable during a 100 frame (500ms) interval, and that the client buffers are large enough to hold pending data.

Practicality of our Implementation: We wish to point out here that our implementation is compatible with off-the-shelf clients since it does not require any modification on the standard client functions. Polling is supported by the standard and each client, by default, responds to the BS with the amount of bits in their buffer. Further, the clients are oblivious to the beam selection by the BS. Note that the clients are omni-directional and it is the BS that applies different beam patterns to the signals it receives from these clients and measures the SNRs. We also keep the modifications required at the BS to a minimum. Indeed, the only change we made to the BS was to disable the stock scheduler and make the BS accept and apply schedules provided externally by the gateway (14 lines of code where the entire BS code is around 10K lines). We believe that this small modification can easily be realized as a firmware update. Naturally, the "brains" of our solution resides on the gateway, which can be configured with minimal deployment effort (e.g., the scheduler and the beam manager can be downloaded from a repository).

7. PERFORMANCE EVALUATION

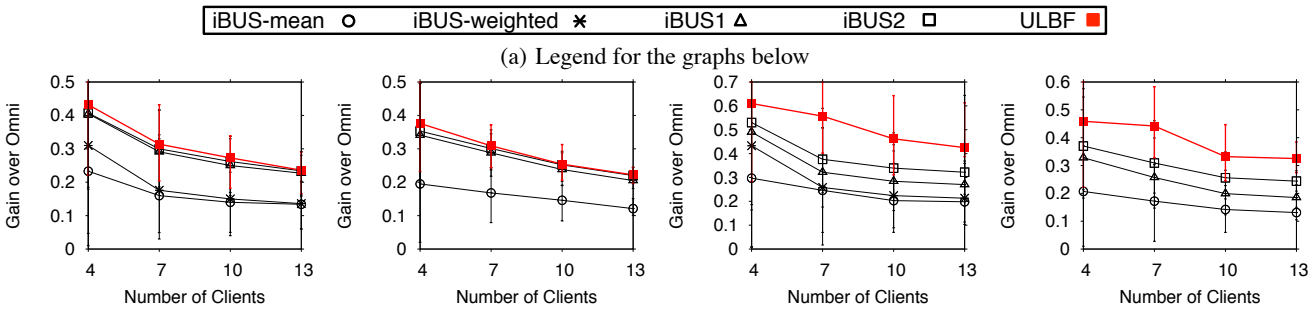
In this section, we evaluate iBUS using both our prototype implementation and trace-driven simulations. To isolate the gains achieved with power pooling (i.e., packing) and beam selection, we propose two simpler versions of iBUS1 that do not require beam adaptation across frames: **(a) iBUS-mean** schedules multiple clients in the same frame like iBUS1 but uses a fixed beam for every frame. The fixed beam is chosen to minimize the mean loss from the best beams of each client. Each client i has SNR $\rho^i{}_{max}$ with some beam yielding the max. SNR from its perspective. If client i has SNR $\rho^i{}_l$ for beam l, then the loss is $\rho^i{}_{max} - \rho^i{}_l$. iBUS-mean picks beam $l \in \mathcal{L}$ that minimizes the mean loss over all clients, **(b) iBUS-weighted** uses a fixed beam as in **iBUS-mean** but the loss is weighted based on the buffer occupancy of each client, to account for asymmetric application rates. In this case, the loss for a client is calculated as $(\rho^i{}_{max} - \rho^i{}_l) * B_i$. Again, the beam $l \in \mathcal{L}$ that minimizes this weighted mean loss over all clients is fixed for all frames in the batch. We compare our algorithms to an omni-directional scheme (labelled **omni**) that leverages power pooling by employing the same packing procedure as iBUS1 (but each frame is transmitted omni-directionally).

7.1 Prototype Evaluation

In WiMAX, the modulation to be used by the clients for uplink communications is determined by the BS. Since the BS directly measures the received SNR from clients, it uses a threshold-based table to determine the modulation for each client. Fig. 5(a) enlists the thresholds used in our platform.

With such discrete rate tables, all SNRs in between two consecutive thresholds map to the MCS with the lower threshold (similar to WiFi). For example in Fig. 5(a), the BS instructs the clients to use $QPSK^{\frac{3}{4}}$ for all SNRs between 12dB and 15dB. Depending on the thresholds (determination of which is not specified by the standard and is left to the vendor), this may result in non-concave client power-pooling curves ($g()$). Indeed with our femtocells, we observed that such non-concavity occurs in practice. While iBUS3 can address non-concavity, we choose to implement iBUS1, iBUS-mean and iBUS-weighted on our prototype because of their low complexity and ease of implementation.

To make iBUS1 compatible with non-concave $g()$, we transform a given $g()$ into a corresponding concave function. Fig. 5(b) depicts an example transformation where $g()$ is computed as the product of number of sub-channels and the number of bits that a given MCS encodes. To do the transformation, we first determine the number of sub-channels where each MCS transition occurs. This can easily be done by computing the SNR for a given number of sub-channels using the power pooling formula. Then, the transition points are connected by lines with an appropriate slope. Note that the resulting concave function assumes that the error rate at a given MCS level increases in conjunction with the SNR drop as the power is distributed over more sub-channels. Even if the table maps



(a) Legend for the graphs below

(b) Discrete $r()$, different rates (c) Discrete $r()$, equal rates (d) Continuous $r()$, different rates (e) Continuous $r()$, equal rates

Figure 6: Performance of iBUS obtained with trace-driven simulations.

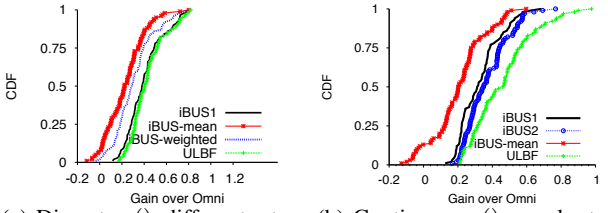
two SNRs to the same MCS, the lower SNR will likely result in a higher bit (or symbol) error rate, reducing the throughput in practice. This observation is also leveraged by prior studies (e.g., [22]). In our experiments, we observed that the throughput predicted by the concave transformation matched the observed client throughput in practice reasonably well. On average, the predicted throughput was 94% of the actual throughput.

We run each scheme on several topologies, created by placing three clients in different locations. The utility function of a user is equal to the total data scheduled for that user in the batch (i.e., $U_i(R_i) = R_i$). The clients initiate UDP flows to the BS, generated by iperf. Fig. 5(c)(a) shows the aggregate throughput for five sample topologies of clients with equal application rates (iBUS-weighted yields the same fixed beam as iBUS-mean and hence, is not considered). We observe that iBUS1 outperforms omni by 42%, on average. In addition, the beam adaptation component in iBUS1 helps outperform iBUS-mean by 15% - 25%. However, cases exist where iBUS-mean performs as well as iBUS1, indicating that the proper choice of a *fixed* beam may provide significant gains, depending on the topology. Fig. 5(c)(b) shows the aggregate throughput for clients with different application rates. We observe that iBUS1 outperforms omni by 45% on average. For iBUS-weighted, we see that it can improve performance over iBUS-mean in some cases (e.g., topology 2 and 3). In addition, there are again cases where iBUS-mean and iBUS-weighted perform as well as iBUS1. To summarize, while iBUS1 consistently outperforms the omni-directional scheme by ≈45%, an appropriate choice of a fixed beam can also yield significant gains in practice.

7.2 Trace-driven Simulations

To evaluate with a large client population, we resort to SNR traces measured at 30 different client locations in our testbed. We consider 100 topologies, generated by picking random subsets of the client locations from our traces. To evaluate iBUS1 and iBUS2 with MCS tables in practice (i.e., discrete $r()$), we employ the same $g()$ transformation as before. We also consider a continuous $r()$ function (Shannon). We introduce an alternative scheme labelled **ULBF**, that mimics user-level beamforming and serves as a loose upper bound. In ULBF, packing is executed by spreading a client's data over multiple frames as in iBUS2. However, rather than using a common beam for a frame, each client can operate using its best beam (i.e., beam with the max. SNR.) even within a frame.

Fig. 6 shows the the mean gains achieved over omni for a varying number of clients. With discrete rate tables (Fig. 6(b) and 6(c)), we observe that iBUS1 delivers around 40% gain over the omni scheme for four clients (inline with our prototype evaluation). However, iBUS2 and ULBF do not significantly improve the performance further. Even though there is increased power pooling and hence higher SNR, the effective rate is not substantially higher than iBUS1 due to the nature of discrete MCS tables. In addition for all the schemes, the gains tend to drop with increasing numbers of clients. With a larger client population, finding a common beam comes at the cost of significant SNR loss from their best beam for some clients. On the other hand, the omni benefits from client diversity (increased chance of a client having high SNR with the omni beam) and is able to deliver increased throughput. However, since the femtocell client populations are typically small (<8), appreciable gains can still be attained. Thus, we conclude that iBUS1 is a low-complexity alternative to user-level beamforming in practical deployments. Since the increased power pooling and beamforming gains of user-level beamforming cannot fully be harnessed with discrete MCS tables, iBUS1 serves as a flexible alternative (i.e., number of antennas is not limited) with similar performance. Fig. 6(d) and 6(e) show the mean gains over the omni scheme for the continuous rate (Shannon) function. This time, we observe that iBUS2 outperforms iBUS1 by 10% - 20% due to its better power pooling capability across frames. Further, the gap with respect to ULBF is around 30% - 40%. This behavior is expected since the continuous rate function will benefit more from better power pooling and proper beam selection, compared to a discrete rate table.



(a) Discrete $r()$, different rates (b) Continuous $r()$, equal rates

Figure 7: CDF of the gain distribution with four clients. iBUS2 is not shown in (a) for clarity (it achieves similar performance as iBUS1).

In the above experiments, we have demonstrated the mean gains over the omni scheme. To better visualize the distribution, we plot the CDF of the gain for four clients. Fig. 7(a) shows that the maximum gain with iBUS1 can be as high as 60% - 80% for 20% of the scenarios. Further, an interesting observation that is revealed is that in 10% - 15% of the scenarios, iBUS-mean performs worse than omni. Recall that iBUS-mean picks a common beam that minimizes the mean loss from the best beam of each client. In such scenarios, some clients may have to operate with a beam, yielding a SNR that is several dB less than their optimal beam and even worse

than the omni-directional beam. However, adapting the beam in iBUS1 addresses this issue to deliver improved performance. Similar observations hold for both types of rate functions (see Fig. 7(b)).

8. CONCLUSIONS

In this paper, we investigate the joint problem of uplink batch scheduling and frame-level beamforming (shown to not admit a PTAS) in OFDMA femtocells. We propose a set of algorithms with a worst case approximation guarantee of $\frac{1}{2}$. Our algorithms address the unique tradeoff between scheduling multiple users in a frame (benefiting from power pooling) and harnessing the gains from directional beams (by applying a common beam for multiple users). We implement the algorithms on a real WiMAX femtocell platform. Our prototype evaluation and trace-driven simulations demonstrate that our algorithms provide benefits of over 40% over an omni-directional scheme. To our best knowledge, this is the first study that integrates and evaluates frame-level beamforming on an actual OFDMA femtocell platform.

For future work, we would like to investigate the uplink scheduling problem, relaxing the assumption that the best user beams remain stable during a batch interval (i.e., the users are not necessarily static and can be mobile). In this paper, we have demonstrated the motivation for using beamforming for interference mitigation in a multi-cell context considering downlink traffic. We would also like to further investigate this direction, while considering beam selections for downlink scheduling in each femtocell.

Acknowledgements

We thank our shepherd, Dr. I-Hong Hou, for his guidance through the camera-ready submission process. The authors would also like to thank Meilong Jiang and Rajesh Mahindra from NEC Laboratories America Inc. for their invaluable support and suggestions during the preparation of this work. This work was partially supported by the Multi University Research Initiative (MURI) grant W911NF-07-1-0318.

9. REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2010-2015," Feb 2011.

[2] V. Navda, A. P. Subramanian, K. Dhansekaran, A. Timm-Giel, and S. Das, "Mobisteer: Using Steerable Beam Directional Antenna for Vehicular Network Access," in *ACM MobiSys*, Jun 2007.

[3] M. Blanco et. al., "On the effectiveness of switched beam antennas in indoor environments," in *Passive and Active Measurements (PAM)*, Apr 2008.

[4] IEEE 802.16m 2011 Part 16, "Air Interface for Broadband Wireless Access Systems - Advanced Air Interface," *IEEE 802.16m standard*.

[5] M. Andrews and L. Zhang, "Scheduling Algorithms for Multi-carrier Wireless Data Systems," in *ACM MobiCom*, Sept 2007.

[6] S. Lee, I. Pefkianakis, A. Meyerson, S. Xu, and S. Lu, "Proportional Fair Frequency-Domain Packet Scheduling for 3GPP LTE Uplink," in *IEEE INFOCOM Mini-conference*, Apr 2009.

[7] IEEE 802.16e 2005 Part 16, "Air Interface for Fixed and Mobile Broadband Wireless Access Systems ," *IEEE 802.16e standard*.

[8] D.J. love, R.W. Heath, and T. Strohmer, "Grassmannian Beamforming for Multiple-Input Multiple-Output Wireless Systems," *IEEE Trans. on Information Theory*, vol. 49, no. 10, pp. 2735–2747, Oct 2003.

[9] G. Song and Y. Li, "Cross-Layer Optimization for OFDM Wireless Networks - Part I: Theoretical Framework," *IEEE Transactions on Wireless Communications*, vol. 4, no. 2, Mar 2005.

[10] A. Pokhariyal, G. Monghal, K. I. Pedersen, P. E. Mogensen, I. Z. Kovacs, C. Rosa, and T. E. Kolding, "Frequency Domain Packet Scheduling Under Fractional Load for the UTRAN LTE Downlink," in *IEEE VTC*, 2007.

[11] S. Deb, S. Jaiswal, and K. Nagaraj, "Real-Time Video Multicast in WiMAX Networks," in *IEEE INFOCOM*, 2008.

[12] B. Bai, W. Chen, Z. Cao, and K. B. Letaief, "Uplink Cross-Layer Scheduling with Differential QoS Requirements in OFDMA Systems," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, no. 168357, 2010.

[13] C. Nie, M. Venkatachalam, and X. Yang, "Adaptive polling service for next-generation ieee 802.16 wimax networks," in *IEEE GLOBECOM*, 2007.

[14] P. Svedman, S. K. Wilson, Jr. L. J. Cimini, and B. Ottersten, "Opportunistic Beamforming and Scheduling for OFDMA Systems," *IEEE Trans. on Communications*, vol. 55, no. 5, May 2007.

[15] N. Wei, A. Pokhariyal, T. B. Sorensen, T. E. Kolding, and P. E. Mogensen, "Performance of MIMO with Frequency Domain Packet Scheduling in UTRAN LTE Downlink," in *IEEE VTC*, 2007.

[16] M. Y. Arslan, J. Yoon, K. Sundaresan, S. V. Krishnamurthy, and S. Banerjee, "FERMI: A Femtocell Resource Management System for Interference Mitigation in OFDMA Networks," in *ACM MobiCom*, Sept 2011.

[17] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, "Tight Approximation Algorithms for Maximum General Assignment Problems," in *ACM SODA*, 2006, pp. 611–620.

[18] M. Fisher, G. Nemhauser, and G. Wolsey, "An analysis of approximations for maximizing submodular set functions-II," in *Mathematical Programming Study*, 1978.

[19] M. Bansal and V. Venkaiah, "Improved Fully Polynomial time Approximation Scheme for the 0-1 Multiple-choice Knapsack Problem," in *International Institute of Information Technology Tech Report*, 2004, `http://people.csail.mit.edu/mukul/01MCKP.pdf`.

[20] G. Calinescu, C. Chekuri, M. PǴl, and J. VondrǴk, "Maximizing a Submodular Set Function subject to a Matroid Constraint (Extended Abstract)," in *Proc. of 12th Conf. on Integer Programming and Combinatorial Optimization*, 2007.

[21] P.R. Goundan and A.S. Schulz, "Revisiting the Greedy Approach to Submodular Set Function Maximization," in *Optimization Online Pre-print*, 2007, `http://www.optimization-online.org/DB_FILE/2007/08/1740.pdf`.

[22] M. Vutukuru, H. Balakrishnan, and K. Jamieson, "Cross-Layer Wireless Bit Rate Adaptation," in *ACM SIGCOMM*, 2009.