sGSA: An SDFMA-OFDMA Scheduling Solution

Anatolij Zubow Johannes Marotzke Daniel Camps-Mur † Xavier Perez Costa †

Humboldt University Berlin
{zubow, marotzke}@informatik.hu-berlin.de

† NEC Laboratories Europe, Network Research Division
{Daniel.CampsMur, Xavier.Perez-Costa}@nw.neclab.eu

***Abstract*—Future wireless networks need to address the predicted growth in mobile traffic volume, expected to have an explosive growth in the next five years mainly driven by video and web applications. Transmission schemes based on Orthogonal Frequency Division Multiple Access (OFDMA) combined with Space Division Multiple Access (SDMA) techniques are key promising technologies to increase current spectral efficiencies. A Joint SDMA-OFDMA system has to allocate resources in time, frequency and space dimensions to different mobile stations, resulting in a highly complex resource allocation problem. In this paper we take a comprehensive view at the complete SDMA-OFDMA scheduling challenge and propose a joint SDMA-OFDMA Greedy Scheduling Algorithm (sGSA) for WiMAX systems. The proposed solution considers feasibility constraints in order to allocate resources for multiple mobile stations on a per packet basis by using i) a cluster-based SDMA grouping algorithm and ii) a computationally efficient frame layout scheme which allocates multiple SDMA groups per frame according to their packet QoS utility. A performance evaluation of the proposed sGSA solution as compared to state of the art solutions is provided, based on a comprehensive WiMAX simulation tool.**

***Index Terms*—OFDMA, SDMA, WiMAX, MU-MIMO, 802.16, QoS, Scheduler, DL-MAP**

I. Introduction

Mobile wireless networks are faced with an ever increasing demand for higher data rates, expected to grow exponentially in the next years due to the smartphone market success and its corresponding set of bandwidth hungry applications, specially video and web. Transmission schemes based on Orthogonal Frequency Division Multiple Access (OFDMA) and Multi-User Multiple Input Multiple Output (MU-MIMO) techniques are promising technologies to increase current spectral efficiencies. A well-known MU-MIMO mode is Space-Division Multiple Access (SDMA) which can be used in the downlink direction (DL) to allow a group of spatially separable MSs to share the same time/frequency resources. SDMA-OFDMA systems have to allocate resources in time, frequency and space dimensions to different MSs resulting in a highly complex resource allocation problem. Therefore, in order to reduce the complexity of such a system the overall problem can be divided into several subproblems, each conquering the different dimensions separately [1]. However, these subproblems still hold a high level of complexity and thus, suboptimal strategies are often proposed in the literature [2].

The rest of the paper is structured as follows. Section II summarizes the most relevant work in the area. Next, Section III formalizes the problem description and formulates the SDMA-OFDMA scheduling as an optimization problem. Section IV describes our proposed sGSA solution along with the designed algorithms required. In Section V a performance evaluation is conducted through simulations. Finally, Section VI summarizes our main findings and concludes the paper.

II. Related Work

Nascimento et al. [3] proposed a joint utility packet scheduler and SDMA-based resource allocation architecture for 802.16e. Similar to our work the authors proposed the Exponential Effective SIR Mapping (EESM[1]) compression method to capture frequency selective channels and build SDMA groups consisting of spatially uncorrelated MSs. The proposed scheduling solution therein assigns MSs to beams, using a First Fit Algorithm [2]-like grouping algorithm and spatial correlation as a grouping metric. In addition, QoS is achieved through a prioritized assignment based on a utility framework. Performance is evaluated under the assumptions of a full queue traffic model. Similar to the previous work, Yao et al. [4] evaluated the MAC performance of an OFDMA system based on the superseded 802.16a-2003 standard, which suggests a frame layout separated in Adaptive Antenna System (AAS) and non-AAS capable zones. The scheduling solution presented therein prioritizes MSs based on their channel conditions and packet delays. Specifically, MSs are grouped according to their channel conditions, whereas, within a group, MSs are prioritized according to packet deadlines. Depending on the intra-beam interference MSs are assigned to a beam or moved to the regular non-AAS zone.

An important difference between these approaches and our work is the assumption of an idealized LOS channel allowing beamsteering based on the estimation of the angle of direction of an MS. Instead we explicitly model the wireless channel coefficients, taking advantage of the LOS component as well as possible scattering and multi-path effects.

III. Problem Statement

This section describes the physical and QoS models used throughout this paper and formulates our proposed SDMA-OFDMA scheduling solution as an optimization problem.

[1]EESM Reference, www.ieee802.org/16/tge/contrib/C80216e-05_141r3.pdf

A. PHY Model

We consider the DL of a single BS with a maximum transmit power that is being equally distributed among the served MSs. The BS is equipped with an Uniform Linear Array (ULA) and there are K single-antenna MSs associated with the BS. We consider OFDMA where the channel bandwidth is divided into S orthogonal subcarriers. In addition, physical subcarriers are mapped to logical subchannels using Partial Usage of SubCarriers (PUSC). Regarding CSI knowledge in the BS, we assume that for each MS u the BS knows the channel transfer matrix ($H_{u,f}$) of every Dth subcarrier, hence resulting in $\frac{S}{D}$ $H_{u,f}$ per MS, where S is the number of subcarriers. Furthermore, the BS has perfect CSI on each $H_{u,f}$, i.e. there are no estimation errors. The required channel coefficients are generated with the WIM simulator [5].

For an MS u the channel coefficient $h_{i,u}$ denotes the sampled frequency response of the channel between the BS antenna element i and the receiver antenna of the u-th MS. Thus, the channel coefficients belonging to a given MS u can be grouped into a column vector $M \times 1$, i.e. $H_u = [h_{1,u}, h_{2,u}, \ldots, h_{M,u}]^{\mathrm{T}}$, where H_u holds all spatial correlations and multi-path effects of the MS u channel. The BS uses precoding weights to adjust the transmitted signal in order to mitigate the propagation effects of the channel and to control the interference among MSs in a SDMA group. Therefore, every channel coefficient $h_{i,j}$ is associated with a beamforming weight $w_{i,j}$, and $W_u = [w_{1,u}, w_{2,u}, \ldots, w_{M,u}]^{\mathrm{T}}$ is the normalized weight vector for a given MS u.

The effective channel gain at the BS can then be computed as $||W_u^{\mathrm{H}} \cdot H_u||_2^2$, where $(\cdot)^{\mathrm{H}}$ denotes the conjugate transpose of a vector/matrix. Notice, that multiple MSs being served during the same time-frequency resource cause interference upon each other (intra-cell interference). Therefore, given a certain frequency resource $b = 1, \ldots, B$, and the corresponding channel coefficients, the SINR of MS u can be computed as [1]:

$$\gamma_{u,b} = \frac{P_{u,b} \cdot ||W_{u,b}^{\mathrm{H}} \cdot H_{u,b}||_2^2}{\sigma^2 + \sum_{v=1, u \neq v}^{M} P_{u,b} \cdot ||W_{v,b}^{\mathrm{H}} \cdot H_{u,b}||_2^2} \quad (1)$$

where $P_{u,b}$ represents the average received power at MS u on frequency resource b. We assume that the BS uses the *Minimum Mean Square Error* technique (MinMSE [6]) in order to compute beamforming weights.

B. Base Station Architecture and QoS Model

We consider an *offline* BS architecture as formally defined in [7]. In an offline architecture there are two main building blocks that cooperate in order to maximize the utility of the scarce radio resources; these blocks are the *QoS Scheduler* and the *SDMA-OFDMA* scheduler. The task of the QoS Scheduler is to select from the higher layer packets available in each flow's buffer a *candidate* list of packets to be transmitted in the next DL subframe. In addition, the QoS scheduler tags each individual packet P_i with a *utility* value u_i. Typical QoS schedulers that can be accommodated in this architecture are Proportional Fair or Deficit Round Robin [8]. Thus, after receiving a set of candidate packets from the QoS Scheduler, the task of the SDMA-OFDMA scheduler is to design the SDMA-OFDMA frame layout, i.e. assign the time, frequency and space resources, in order to maximize the amount of utility carried in the SDMA-OFDMA frame.

C. Problem Formulation

The main objective of a DL SDMA-OFDMA scheduler is to maximize the total utility carried in the DL subframe. However, optimally assigning time, frequency and space resources is a very complex task to be efficiently implemented in practice. Therefore, in order to reduce complexity we divide the overall optimization problem into two independent tasks:

1. *SDMA group formation*: we consider the problem of creating spatially compatible SDMA groups, where each group consists of a set of MSs which can be separated in the space domain.
2. *OFDMA frame construction*: given a set of SDMA groups, we consider the problem of scheduling the subset of these groups within the OFDMA DL subframe that maximizes the overall utility carried in the frame.

SDMA group formation: The objective of this problem can be formalized in the following way:

Instance: A set U of MSs having buffered packets as delivered by the QoS scheduler.

Objective: Find the optimal partition of U into a set $\mathcal{G} = \{G_1, G_2, \ldots, G_m\}$ of non-overlapping and non-empty sets of MSs that cover all of U, i.e. $\bigcup_{i=1}^{m} G_i = U$ and $G_i \cap G_j = \emptyset, \forall i \neq j$, such that the average group capacity is maximized. Here $G_i = \{u_{i,1}, u_{i,2}, \ldots\}$ represents a single SDMA group where $u_{i,j}$ represents the j-th MS in group i. Moreover, γ_{u,G_i} is the SINR of MS u in group G_i.

This optimization problem can be formulated as follows:

$$\mathcal{G} = \arg\max_{\mathcal{G}} \left\{ \frac{1}{|\mathcal{G}|} \sum_{i=1}^{|\mathcal{G}|} \sum_{u \in G_i} \log(1 + \gamma_{u,G_i}) \right\} \quad (2)$$

subject to:
(I) A maximum group size limited to M, i.e. the number of available antennas at the BS:

$$|G_i| \leq M, \forall G_i \in \mathcal{G} \quad (3)$$

(II) A minimum SINR constraint, that makes sure that each MS in an SDMA group can be served at least on the lowest Modulation and Coding Scheme (MCS):

$$\gamma_{u,G_i} \geq \gamma_{\mathrm{MIN}}, \forall G_i \in \mathcal{G}, \forall u \in G_i \quad (4)$$

Notice in eq. 2 that the target metric is divided by the number of SDMA groups, $|\mathcal{G}|$, in order to favor solutions with a fewer number of groups. The reason is that solutions with fewer groups tend to be more efficient because different groups can only be scheduled in different time-frequency resources.

Complexity: The optimal solution for this group formation problem was shown to be NP-complete in [2].

OFDMA frame construction: Before formalizing this problem, we introduce an additional design decision that consists of forcing an allocated SDMA group to span always an integer number of columns in the OFDMA frame. The main reason for this decision is that it enables the previous SDMA group formation problem to be agnostic to the actual frequency region where the data will be transmitted, i.e. the required SINR values in eq. 2 can be derived assuming that an allocation spans the whole frequency range. In addition, this design decision turns the packing of SDMA groups within an OFDMA frame into a one-dimensional packing problem. Thus, our OFDMA frame construction problem is formalized in the following way:

Instance: A partition of the set U of MSs into a set $\mathcal{G}$ of sets of MSs, i.e. a set of SDMA groups.

Objective: The packing area for each SDMA group $G_i \in \mathcal{G}$ in the OFDMA frame is controlled by an associated *container*. Find the optimal width $s_i \in [0 \dots DL_{slots}]$ for each container such that the total utility, $u_{(\cdot)}$, carried by the OFDMA frame is maximized, i.e. $\overline{S}^* \in [(s_0, s_1, \dots, s_{|\mathcal{G}|})]$, where DL_{slots} is the total number of time slots in the DL frame. Here, the first container of width s_0 represents the space reserved for the DL-MAP to signal the data bursts within the frame.

This optimization problem can be formalized as follows:

$$\{\overline{S}^*\} = \arg\max_{\{\overline{S}\}} \left\{ \sum_{i=1}^{|\mathcal{G}|} \sum_{u \in G_i} \sum_{\substack{p_u \in P_u^{'} \\ P_u^{'} \subseteq P_u}} u_{(\cdot)}(p_u) \right\} \quad (5)$$

subject to:
(I) A container layer space constraint that makes sure that the total size of packed data bursts $P_u^{'}$ in each container layer does not exceed the available space:

$$\sum_{p_u \in P_u^{'}} \mathrm{p2burst}(p_u, \gamma_{u,G_i}) \leq \overline{S_i} \times B \quad (6)$$

where γ_{u,G_i} represents the SINR of MS u in group G_i, $\overline{S_i}$ represents the width of the i-th container, and B represents the number of frequency resources in one time slot (equal to the number of subchannels). The function $\mathrm{p2burst}(\cdot)$ calculates the burst size in slots for a given packet size and SINR value.

(II) A DL-MAP space constraint that makes sure that all packed data bursts $P_u^{'}$ in each layer of each container can be properly signaled. For this purpose, the container at position zero is reserved for the DL-MAP.

$$\left\lceil \frac{F_0 + \sum_{u \in U}(F_1 + \sum_{p=1}^{|P_u^{'}|} F_2)}{\mathrm{SC}} \right\rceil \cdot R \leq \overline{S_0} \times B \quad (7)$$

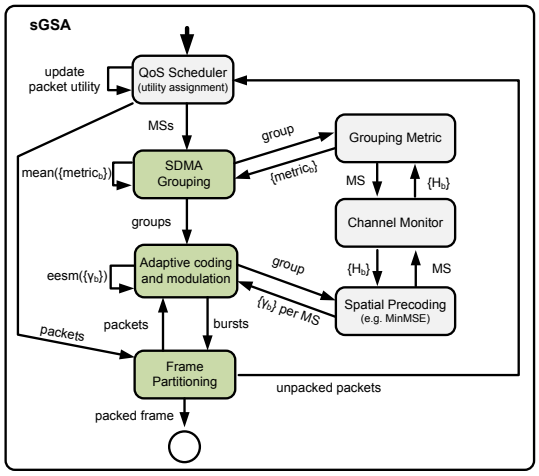


Fig. 1. Overview of the sGSA scheduling solution.

where F_0, F_1, F_2, SC and R represent various MAP overheads (Section V, Table I).

Complexity: The optimization problem is a variant of the Multiple-Choice Nested Knapsack Problem [9] and therefore NP-complete.

IV. Proposed Solution

In this section we present the design of our DL SDMA-OFDMA Greedy Scheduling Solution (*sGSA*). In order to implement the assignment of time, frequency and space resources in an efficient way, sGSA addresses sequentially the following two sub-problems: i) first, the problem of creating spatially compatible SDMA groups, and ii) second the problem of scheduling a given set of SDMA groups within an OFDMA frame. sGSA contains three different modules (Fig. 1):

1. The *Cluster-based Grouping Algorithm (CBA)*, that is a heuristic method to solve the SDMA group formation problem described in eq. 2.
2. The *Adaptive Modulation and Coding (AMC) module*, that selects the appropriate MCS to be assigned to each MS within an SDMA group.
3. The *Frame construction module*, that solves the OFDMA frame construction problem described in eq. 5 making use of an *SDMA-OFDMA packing algorithm*.

A. Cluster-based Grouping Algorithm

Our solution to the SDMA group formation problem is the *Cluster-Based grouping Algorithm (CBA)* described in Algorithm 1. The core idea behind CBA is the following. Initially, CBA creates $\lceil K/M \rceil$ clusters, assuming that with M antennas we can spatially separate M MSs, and randomly distributes MSs among these clusters (line 3 in Alg. 1). Afterwards, the initial random groups are iteratively improved by means of two elementary functions (*clusterSwap* and *clusterJump*) (lines 6-7 in Alg. 1), used to exchange the membership of MSs between groups. *ClusterSwap* exchanges two MSs from different clusters with each other, whereas *clusterJump* allows

Algorithm 1 Cluster-based SDMA Grouping Algorithm.

Require:
1: K - Number of MSs; M - Number of BS antennas
2: **procedure** CBA
3: $\mathcal{G} \leftarrow$ createRndClusters() ▷ Create $\lceil K/M \rceil$ many clusters with MSs randomly assigned
4: **while** true **do**
5: **for** $x = 1$ to K **do**
6: $\mathcal{G}' \leftarrow$ clusterSwap($\mathcal{G}$) ▷ Process cluster swap op.
7: $\mathcal{G}'' \leftarrow$ clusterJump($\mathcal{G}'$) ▷ Process cluster jump op.
8: **if** $\mathcal{G}'' == \mathcal{G}$ **then** ▷ Clustering does not change
9: break
10: $\mathcal{G} \leftarrow \mathcal{G}''$
11: **if** getOutageMSs($\mathcal{G}$) $== \emptyset$ **then** ▷ Estimate MSs in outage
12: break
13: $\mathcal{G} \leftarrow$ createNewClusters($\mathcal{G}$) ▷ Create additional clusters for MSs in outage
14: **return** $\mathcal{G}$ ▷ Return MS clustering/grouping

a MS to leave one cluster and join another one. In order to bound execution time, the number of swap and jump operations is bounded by K, i.e. the number MSs (line 5 in Alg. 1). In addition, if at the end of one iteration, some MSs are found to be in outage (meaning that the number of clusters is too small to handle all MSs) new clusters are added and the in-outage MSs are distributed among them (lines 11-13 in Alg. 1). Notice that in each iteration the number of clusters is increased by at least one. Finally, CBA terminates when no more MSs are in outage.

The *clusterSwap* operation is described in Algorithm 2. Its main task is to find the swap between two MSs from different clusters, where the improvement in terms of a given group metric $c(\cdot)$ is largest. Notice, that when computing the group metric, only MSs having a SINR above the minimum threshold are considered. In addition, the swapping operation preserves valid groups; i.e. if before the swap none of the MSs within a cluster is in outage then this property is preserved after performing the swap (lines 10-12 in Alg. 2).

The *clusterJump* operation is similar to *clusterSwap* except that a MS leaves a cluster and joins another cluster so that the improvement in terms of group metric is largest. *ClusterJump* is described in Algorithm 3. In addition, in *clusterJump*, the computed group metric, C', is divided by the number of clusters in order to account for the fact that having MSs grouped in less clusters is more efficient, because they can share the same time-frequency resources (line 11 in Alg. 3). Finally, a jump is only allowed if after the jump all MS in the new cluster are not in outage (lines 12-14 in Alg. 3).

a) Complexity: Given that *clusterSwap* and *clusterJump* have a complexity of $O(K^2)$, a maximum of K swap and jump operations are allowed (line 5), and in the worst case there is always at least one MS in outage after each iteration, the worst case computational complexity of CBA regarding its execution time is $O(K^4)$, where K is the number of MSs.

B. Adaptive Modulation and Coding (AMC) module

The Adaptive Modulation and Coding (AMC) module receives a set of proposed SDMA groups from CBA for each subframe and is in charge of deciding the Modulation and Coding Scheme (MCS) to be assigned to each MS in each

Algorithm 2 Cluster Swap Operation.

Require:
1: $\mathcal{G}$ - Set of SDMA groups
2: **procedure** CLUSTERSWAP($\mathcal{G}$)
3: $\Delta \leftarrow 0$ ▷ Capacity gain of best solution
4: **for** $i = 1$ to K **do**
5: **for** $j = 1$ to K **do**
6: **if** i == j OR cluster(i) == cluster(j) **then**
7: continue ▷ MS i and j are in the same cluster
8: $C \leftarrow c(\text{cluster}(i)) + c(\text{cluster}(j))$
9: $C' \leftarrow c(\text{cluster}(i) \backslash \{i\} \cup \{j\}) + c(\text{cluster}(j) \backslash \{j\} \cup \{i\})$
10: **if** $outageMS(\text{cluster}(i)) == \emptyset \wedge outageMS(\text{cluster}(i) \backslash \{i\} \cup \{j\}) \neq \emptyset$ **then**
11: continue ▷ Preserve valid clusters
12: **if** $outageMS(\text{cluster}(j)) == \emptyset \wedge outageMS(\text{cluster}(j) \backslash \{j\} \cup \{i\}) \neq \emptyset$ **then**
13: continue ▷ Preserve valid clusters
14: **if** $C' - C > \Delta$ **then**
15: $\mathcal{G}^* \leftarrow \text{updateCluster}(\mathcal{G}, i, j)$ ▷ Save solution with i and j being swapped
16: $\Delta \leftarrow C' - C$
17: **if** $\Delta > 0$ **then**
18: $\mathcal{G} \leftarrow \mathcal{G}*$ ▷ Process swap + return new cluster config
19: **return** $\mathcal{G}$ ▷ Return MS clustering/grouping

Algorithm 3 Cluster Jump Operation.

Require:
1: $\mathcal{G}$ - Set of SDMA groups
2: **procedure** CLUSTERJUMP($\mathcal{G}$)
3: $\Delta \leftarrow 0$ ▷ Capacity gain of best solution
4: **for** $i = 1$ to K **do**
5: **for** $j = 1$ to K **do**
6: **if** i == j OR cluster(i) == cluster(j) **then**
7: continue ▷ MS i and j are in the same cluster
8: $C \leftarrow (c(\text{cluster}(i)) + c(\text{cluster}(j)))/2$ ▷ Cap. before jump
9: $C' \leftarrow c(\text{cluster}(i) \backslash \{i\}) + c(\text{cluster}(j) \cup \{i\})$ ▷ After jump
10: **if** $\text{cluster}(i) \backslash \{i\} \neq \emptyset$ **then**
11: $C' \leftarrow C'/2$ ▷ cluster(i) is not empty after jump
12: **if** $C' - C > \Delta \wedge outageMS(\text{cluster}(j) \cup \{i\}) \neq \emptyset$ **then**
13: $\mathcal{G}^* \leftarrow \text{updateCluster}(\mathcal{G}, i, j)$ ▷ Save solution
14: $\Delta \leftarrow C' - C$
15: **if** $\Delta > 0$ **then**
16: $\mathcal{G} \leftarrow \mathcal{G}*$ ▷ Process jump + return new cluster config
17: **return** $\mathcal{G}$ ▷ Return MS clustering/grouping

spatial group. Notice though that the received SDMA groups were created based on some grouping metric which is not necessarily based on calculating the actual SINR (i.e. metrics considering the spatial correlation only). In order to derive the MCS to be assigned to each MS, the AMC module computes the actual pre-coding matrixes per frequency block that will be used. Once the pre-coding matrixes are computed, the AMC module can obtain a reliable SINR estimate for each MS and frequency block by means of eq. 1. Then, the AMC module collapses the set of per-frequency block SINR values, $\{\gamma_{u,b}\}$, into a single equivalent SINR value for each MS making use of the EESM mapping technique. Finally, a single scalar SINR value is available for each MS and given a target block error rate (1% in our case), the AMC module selects the proper MCS by doing a simple look up operation on a pre-computed *Block Error Rate-SNR* table.

C. SDMA-OFDMA Packing Algorithm

In this section we present our heuristic solution to the SDMA-OFDMA frame construction problem described in eq. 5. Given a set of SDMA groups we need to select a subset

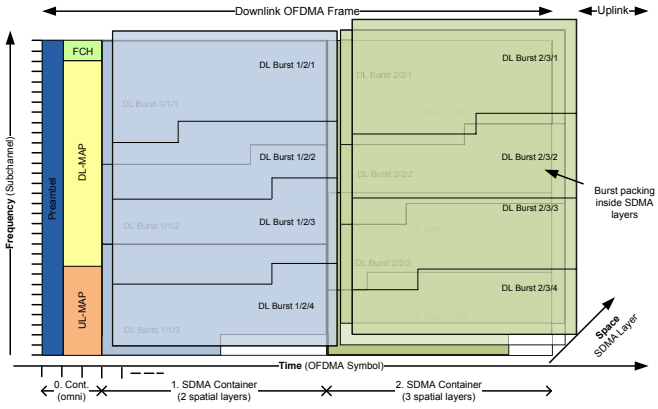


Fig. 2. WiMAX SDMA-OFDMA frame layout.

of groups to be scheduled in the OFDMA frame and allocate for each group a certain space within the frame, which we hereafter refer to as a *container*. In addition, we also need to allocate a certain amount of space for signaling (MAP). Fig. 2 illustrates an example frame layout to be generated by sGSA, where we can recognize three containers. Each container is associated with an SDMA group, whereas the container at position zero is reserved for the transmission of signaling data. Each container has a variable width (time slots) but always spans all subchannels of a given time slot. Moreover, each container consists of multiple spatial layers carrying data bursts belonging to the MSs of the associated SDMA group. In the example in Fig. 2 the number of spatial layers is two and three for the two data containers respectively.

Thus, our SDMA-OFDMA packing algorithm fulfills the tasks of selecting and allocating SDMA groups within the OFDMA frame, in order to maximize the overall carried utility. Algorithm 4 presents our SDMA-OFDMA packing algorithm that consists of two phases: i) the *preparation* phase (lines 3-4), and ii) the *extension* phase (lines 5-16).

Preparation phase - A container can be either in status *scheduled* or *unscheduled* where at the beginning all containers are set to *unscheduled*. The initial width of a container is the maximum between one time slot, and the minimum number of time slots required to carry at least one packet.

Extension phase - In this phase, upon each iteration, a *container metric* representing the value of each candidate container is used to evaluate different containers (lines 6-12). The way to compute this metric is described in Algorithm 5, where the basic idea is to greedily select the container that brings the highest increase in terms of utility per slot. In addition, a container is considered a suitable candidate only if there is enough space in the frame to include its required signaling (MAP, line 9). Thus, in each iteration of Algorithm 4 (lines 13-16) either an *unscheduled* container is added to the frame (set to status *scheduled*) or an already *scheduled* container is increased in size by one time slot. This phase is repeated until all time slots in the frame have been assigned.

Finally, the container metric procedure depicted in Algorithm 5 needs a way to obtain the list of the packets that can be packed in the spatial layers of a given container while respecting a maximum MAP size (lines 4-5, Algorithm 5). For this purpose, Algorithm 6 greedily iterates over the list of packets addressed to the MSs of the corresponding SDMA group and adds them to their respective spatial layers assuming burst concatenation[2] until the available capacity is reached. The considered packet list is pre-sorted according to utility per slot in descending order to ensure that the most valuable packets are packed first. In case the required MAP size is larger than the maximum allowed MAP size the excess number of packets is calculated, and the packets with the lowest utility are removed (lines 13-14 in Algorithm 6).

Algorithm 5 The Container metric is the ratio between the additional utility resulting from the enlargement of the container and the required additional frame space for the MAP and the container itself, i.e. the additional utility per slot gain.

Require:
1: C_i - container to be evaluated
2: P - a list of packets destined to the MSs in C_i
3: **procedure** contMetric(C_i, P)
4: $P_{\text{old}} \leftarrow \text{contPackEST}(C_i.\text{gr}, P, C_i.\text{schedSz}, C_i.\text{schedMSz})$
5: $P_{\text{new}} \leftarrow \text{contPackEST}(C_i.\text{gr}, P, C_i.\text{sz}, C_i.\text{maxM})$
6: $m \leftarrow \frac{u_{(\cdot)}(P_{\text{new}}) - u_{(\cdot)}(P_{\text{old}})}{\max((\text{map\_sz}(P_{\text{new}}) - \text{map\_sz}(P_{\text{old}})) + (C_i.\text{sz} - C_i.\text{schedSz}), 1_{\text{slot}})}$
▷ Container size of P_{old} and P_{new} is C_i.schedSz and C_i.sz respectively
7: **return** (m)

b) Complexity: In the worst case all MS are fully correlated and we have K containers, i.e. one MS per SDMA group. According to Algorithm 4 in each iteration the width of an already scheduled container is increased by one or an unscheduled container with its initial width is added to the frame (lines 13-16). Therefore, Algorithm 4 terminates after DL_{slots} iterations, where DL_{slots} is the number of time slots in the downlink OFDMA frame. In the worst case in each iteration the container metric (Algorithm 5) has to be calculated for each group due to a changing constraint for the maximum allowed signaling ($\text{MAP}_{\max}$). Thus the total number of times the container metric is executed is K times for the initial K containers and $DL_{slots} \times K$ times until the algorithm terminates. The container metric makes use of the packing estimator (Algorithm 6) which itself has computational complexity of $O(P)$, where P is the number of packets addressed to a given SDMA group (container). Therefore the overall complexity of our SDMA-OFDMA frame construction algorithm is $O(K + DL_{slots} \times K) \times O(P)$. Notice though that DL_{slots} is bounded by the frame size, and if required the number of packets P can be limited by the QoS scheduler.

V. Performance Evaluation

The performance of the proposed solution is analyzed by means of simulations according to the methodology described in [10]. The most important parameters are summarized in

[2]Concatenated packets are packets addressed to the same MS that are coded and modulated together. In this way a single entry in the MAP is required that reduces overhead.

Algorithm 4 The SDMA-OFDMA packing algorithm partitions a frame into containers representing the packing area for SDMA groups.

Require:
$\mathcal{G}$ - set of SDMA groups; P - list of packets destined to MSs in $\mathcal{G}$ sorted according to utility per slot; SCH - no. of subchannels; DL_{slots} - number of DL slots; $C_i.sz/C_i.schedSz$ - size demanded/scheduled by container C_i; $C_i.maxM/schedMSz$ - MAP size required/scheduled by container C_i
procedure framePart($\mathcal{G}$, P, SCH, DL_{slots})
  $C_{\text{all}} \leftarrow \text{initContainers}(\mathcal{G}, SCH)$ ▷ create container for each group
  $Q_{\text{up}} \leftarrow C_{\text{all}}$; $F \leftarrow SCH \times DL_{slots} - \text{FCH}$; $FS \leftarrow F$ ▷ initially all containers must be updated; calc total and available free space frame space
  while $|Q_{\text{up}}| > 0$ **do**
    for $C_i \in Q_{\text{up}}$ **do** ▷ for each competing container
      $C_i.\text{maxM} \leftarrow FS - C_i.\text{sz} + C_i.\text{schedSz} + C_i.\text{schedMSz}$ ▷ calculate maximum MAP size available for container C_i
      if $|P(C_i)| > |C_i.\text{schedP}|$ **and**
        $C_i.\text{maxM} > C_i.\text{schedMSz}$ **then** ▷ Sufficient packets and the maximum possible MAP size is greater than the currently required MAP size
        $C_i.\text{metric} \leftarrow \text{contMetric}(C_i, P(C_i))$ ▷ update container metric
      else if $FS/SCH < 1$ **then** ▷ if we cannot increase the container
        $C_i.\text{finished} \leftarrow \text{true}$ ▷ make container size fixed
    $maxId \leftarrow \text{getMaxId}([C_{\text{all}}.\text{metric}])$ ▷ select best container
    $C_{\text{all}}(maxId).\text{schedule}()$ ▷ meet demand of best container
    $FS \leftarrow F - \text{getUsedSpace}(C_{\text{all}})$ ▷ update free space
    $Q_{\text{up}} \leftarrow \text{incrDemand}(C_{\text{all}}, FS)$ ▷ update container demand (i.e. increased container width) and add them to list
  $\overline{S} \leftarrow [C_{\text{all}}.\text{schedSz}]$; $\overline{M} \leftarrow [C_{\text{all}}.\text{schedMSz}]$ ▷ Size of each container and DL-MAP
return $(\overline{S}, \overline{M})$ ▷ container and map size vector

Algorithm 6 Algorithm estimates the set of packets which can be packed in a given SDMA container where the maximum map size is restricted.

Require:
G - set of MSs representing a SDMA group
P - packet list for MSs in G sorted by utility per slot in descending order
C_{sz} - the capacity of the container in slots
$\text{MAP}_{\max}$ - maximum map size which can be used (in slots)
procedure contPackEST(G, P, C_{sz}, $\text{MAP}_{\max}$)
  $P^* \leftarrow \emptyset$
  $C \leftarrow new\ C(G, C_{\text{sz}})$ ▷ create container
  for $p \in P$ **do**
    if $C.\text{canAddToLayer}(p)$ **then** ▷ if sufficient space in spatial layer
      $P^* \leftarrow [P^*\ p]$ ▷ append to packet list
      $C.\text{addToLayer}(p)$ ▷ add packet to corresponding layer
  $map_{sz} \leftarrow \text{map\_sz}(P^*)$ ▷ required MAP size to address packets
  if $map_{sz} > \text{MAP}_{\max}$ **then** ▷ if max MAP size exceeded
    $P^* \leftarrow \text{removeLowestUtility}(\text{P}^*, map_{sz} - \text{MAP}_{\max})$ ▷ remove packets with smallest utility
return P^*

Parameter	Value
System bandwidth	10 MHz
FFT size	1024
Center frequency	2.5 GHz
Frequency reuse pattern	3x1x1
Transmit power	46 dBm
MS noise density (dBm/Hz)	-167 dBm/Hz
Cell radius	$\sim 288\ m$
WIM scenario	C2 (urban macro-cell, LOS)
Number of antennas at BS (M)	1-5 omni elements separated by half wavelength
MSs placement	uniform
MSs' speed	2 km/h
OFDMA frame duration	5 ms
DL/UL ratio	35/12
MAP	1 repetition, QPSK 1/2
F_0	fixed_dl_map_overhead (72 bits)
F_1	dl_map_ie_fixed_overhead (44 bits)
F_2	dl_map_ie_cid_size (16 bits)
SC	num_subcarriers_slot (48)
Tx Precoding algorithm	MinMSE
Fading outage margin	3 dB
WiMAX permutation scheme	PUSC
Packet buffer size	$K \times 12.96$ KiB
No. of placement seeds	256

TABLE I
SIMULATION PARAMETERS.

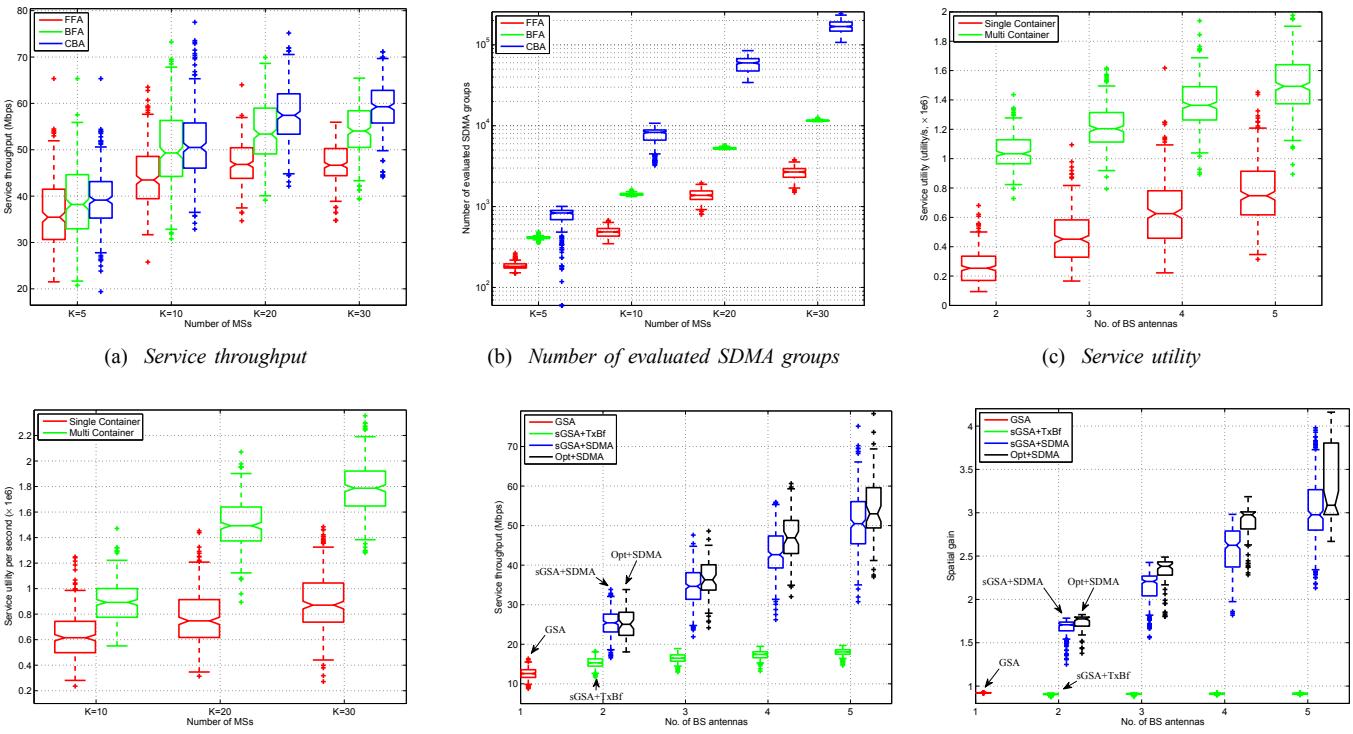
Table I. The following traffic model is used. The BS maintains per-MS packet buffers, and packets are generated based on a distribution derived from a data collection campaign by SPRINT[3]. The obtained packet size distribution is dominated by TCP flows. We do not considered the offered load to be equally distributed among all active MSs, but instead consider a more unbalanced situation where 50% of the MSs generate 80% of the traffic at the MAC layer. Finally, the following two different utility functions will be considered that capture possible QoS policies considered by a service provider: (i) *Proportional Fair Utility (PF)* [8] and (ii) *Random Utility*, whereas the later models an arbitrary policy set by a service provider by assigning a random utility value.

[3]https://research.sprintlabs.com/packstat/

A. Simulation Results

In this section we evaluate the performance of sGSA. First, we evaluate the performance of our proposed CBA grouping algorithm as compared to other grouping algorithms in the state of the art. Second, we validate the assumptions behind our SDMA-OFDMA packing algorithm. Finally, we evaluate the overall performance of sGSA against that of other algorithms in the state of the art.

1) Cluster-based Grouping Algorithm (CBA): In this section we investigate into the performance trade-offs of our *CBA* grouping algorithm, as compared to the traditional *Best Fit Algorithm* (BFA) and *First Fit Algorithm* (FFA) [2] algorithms. In particular, we are interested in studying the effect that the number of MSs in the network cell, K, has on the performance/complexity trade-off of these SDMA grouping



(a) *Service throughput* (b) *Number of evaluated SDMA groups* (c) *Service utility*

(d) *Service utility* (e) *Service throughput* (f) *Spatial multiplexing gain*

Fig. 3. Simulations results.

algorithms. Notice, that *CBA* has a worst-case complexity of $O(K^4)$, while it is $O(K^2)$ for BFA. In addition, all the considered SDMA grouping algorithms use *group capacity* as grouping metric.

Fig. 3(a) illustrates the service throughput achieved by the algorithms under study, when considering $M = 5$ antennas in the BS and $K = 5, 10, 20, 30$ MSs in the target cell. As we can see in the figure, *CBA* always achieves the highest service throughput, followed respectively by *BFA* and *FFA*. In particular, the gain of *CBA* is higher as the number of MSs in the cell, K, increases, and results in a 10% gain for $K = 30$ MSs. Notice that as K grows, the task of finding *optimal* groups becomes harder.

In order to evaluate the computational complexity of the different grouping algorithms, Fig. 3(b) illustrates for each algorithm the number of candidate SDMA groups evaluated. We can clearly see that *CBA*, by means of its *clusterJump* and *clusterSwap* operations, is the algorithm evaluating the highest number of candidate SDMA groups, which hence translates into a higher performance but also a higher complexity.

Therefore, we conclude that *CBA* and *BFA* provide a complementary performance/complexity trade-off, both being better than *FFA*, where a particular vendor could select *CBA* if it is more interested in performance, and *BFA* if it is more interested in reducing complexity. In the rest of the paper we assume that *CBA* is the SDMA grouping algorithm being used.

2) SDMA-OFDMA Packing algorithm: In this section we evaluate one of the main assumptions behind our SDMA-OFDMA packing algorithm, described in section IV-C. This assumption is our claim that packing several SDMA groups within a single OFDMA frame, instead of packing only the best group, delivers significant performance gains. For this purpose we compare the performance of a scheduling algorithm that only packs the best SDMA group in each OFDMA frame, *Single Container*, and our proposed packing algorithm, *Multi Container*, that selects from all the available SDMA groups the ones that have to be packed in each OFDMA frame. For the purpose of this experiment a *random* utility policy is used, and we vary the number of antennas in the BS, M, and the number of MSs in the cell, K. In particular, the selected random utility policy considers that 20% of the MS are premium MSs, and thus multiplies their packet utility by a factor of ten in order to account for their importance. Notice that this utility policy may represent a real-life scenario where a service provider prioritizes some MSs over the rest.

Fig. 3(c) depicts the results in terms of service utility per second for 20 MSs. The *Multi Container* approach results in significant gains, up to a four-fold increase, compared to the *Single Container* approach. Similarly, Fig. 3(d) depicts the same results for an experiment where the number of antennas in the BS is fixed to $M = 5$, and the number of MSs in the cell increases from 10 to 30. As observed in the figure, the *Multi Container* approach outperforms the *Single Container* approach, especially as K grows. Notice that the main reason behind the gains achieved by our *Multi Container* approach is its higher flexibility in being able to

conveniently schedule high priority packets from different MSs, which may not be spatially compatible, in different SDMA groups. Regarding computational complexity, the *Single Container* algorithm has of course lower complexity than the *Multi Container* algorithm. However, these differences are not significant when considering the overall complexity of an SDMA-OFDMA scheduler, because the biggest part of the complexity corresponds to the SDMA grouping algorithm.

Thus, given the observed results, we conclude that our proposed approach of packing multiple SDMA groups within a single OFDMA frame may translate in significant performance gains in practical scenarios. In the rest of the paper we consider the SDMA-OFDMA packing algorithm described in section IV-C as the packing solution employed by sGSA.

3) Overall performance of sGSA: In this section we evaluate the overall performance of the following algorithms:

- *Opt+SDMA*: This scheduler generates SDMA groups by doing an exhaustive search among all possible groups. It also performs an exhaustive search in order to select the SDMA groups that are packed in the OFDMA frame.
- *sGSA+SDMA*: This is our *sGSA* algorithm.
- *sGSA+TxBf*: Instead of transmitting to multiple MSs at the same time, *sGSA+TxBf* schedules a single MS in each time-frequency region but boosts the SINR perceived by this MS using *Transmit Beamforming*.
- *GSA*: GSA [7] is an OFDMA scheduling solution that uses two-dimensional packing. GSA though can only be used with $M = 1$ antennas in the BS.

In the following we analyze how the performance scales with the number of antennas available in the BS. For all the experiments we consider that the QoS Scheduler uses a PF utility. Fig. 3(e) depicts the service throughput achieved by our algorithms under study in a scenario where we have $K = 10$ active MSs in the target cell. The most remarkable result in this figure is that *sGSA+SDMA* clearly outperforms *sGSA+TxBf*, up to a three-fold gain for $M = 5$ antennas in the BS, and performs very close to the *Opt+SDMA* algorithm. We analyze the previous two results separately. First, the performance of *sGSA+TxBf* is low in this scenario, because with $K = 10$ MSs in the cell and the employed PF utility, the *Multi-User Diversity (MUD)* gain is already enough for the BS to transmit to MSs that experience good channel conditions. Hence, transmit beamforming results in small gains in a scenario where MSs already experience relatively good channel conditions. Instead, a dramatic gain is obtained when the BS uses SDMA and is able to simultaneously transmit to more than one MS. Second, although the performance of *sGSA+SDMA* is very close to the performance of the *Opt+SDMA* algorithm in Fig. 3(e), we want to be cautious in generalizing this result. The reason is that as the number of MSs in the cell, K, increases, we expect the gain of *Opt+SDMA* over *sGSA+SDMA* to increase, because as K increases it becomes more difficult for *sGSA* to generate SDMA groups that are close to the optimal ones. Given the computational requirements of the *Opt+SDMA* algorithm though, we leave as future work the study of its performance for large K's.

Finally, Fig. 3(f) depict for each algorithm the spatial multiplexing gain, i.e. with SDMA we can serve multiple MSs on the same time/frequency resource. As expected, *sGSA+SDMA* delivers a linear spatial gain, which is slightly below M due to MAP overhead which need to be transmitted omnidirectionally. For example, the gain is around 3 for $M = 5$ antennas, meaning that with five antennas *sGSA+SDMA* comes up on average with SDMA groups of size three. Given the observed gain, *sGSA+SDMA* seems to be a logical extension to *GSA* when multiple antennas are available at the BS.

VI. Conclusions

In this paper we took a comprehensive view at SDMA-OFDMA systems which are expected to be a key technology building block to increase current spectral efficiencies. SDMA-OFDMA systems have to allocate resources in time, frequency and space dimensions to different MSs, resulting in a highly complex resource allocation problem. In our work, we designed and analyzed a joint SDMA-OFDMA Greedy Scheduling Algorithm (sGSA) for WiMAX networks. The proposed solution considers feasibility constraints in order to allocate resources for multiple mobile stations on a per packet basis by using i) a cluster-based SDMA grouping algorithm and ii) a computationally efficient frame layout scheme which allocates multiple SDMA groups per frame according to their packet QoS utility. The overall performance of the proposed sGSA algorithm was evaluated by system-level simulation.

References

[1] T. F. Maciel, "Suboptimal Resource Allocation for Multi-User MIMO-OFDMA Systems," Ph.D. dissertation, Technical University of Darmstadt, 2008.

[2] F. Shad, T. D. Todd, V. Kezys, and J. Litva, "Dynamic Slot Allocation (DSA) in Indoor SDMA/TDMA Using a Smart Antenna Basestation," *IEEE/ACM Transactions on Networking*, vol. 9, 2001.

[3] A. Nascimento and J. Rodriguez, "A joint utility scheduler and sdma resource allocation for mobile wimax networks," in *Proceedings of the 5th IEEE international conference on Wireless pervasive computing*, ser. ISWPC'10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 589–594.

[4] C. Y. Huang, C.-Y. Chang, P.-H. Chen, and M.-H. Wu, "Performance Evaluation of SDMA Based Mobile WiMAX Systems," *International Wireless Communications and Mobile Computing Conference*, pp. 1042–1046, Aug. 2008.

[5] L. Hentilä, P. Kyösti, M. Käske, M. Narandzic, and M. Alatossava, "MATLAB implementation of the WINNER Phase II Channel Model ver1.1," December, 2007. [Online]. Available: https://www.ist-winner.org/phase_2_model.html

[6] F. Gross, *Smart Antennas for Wireless Communications: With MATLAB (Professional Engineering)*, 1st ed. McGraw-Hill Professional, Sep. 2005.

[7] A. Zubow, D.Camps-Mur, X.Pérez-Costa, and P. Favaro, "Greedy Scheduling Algorithm (GSA) - Design and Evaluation of an Efficient and Flexible WiMAX OFDMA Scheduling Solution," Elsevier Computer Networks Journal, Volume 54, Issue 10, July 2010.

[8] J. Lakkakorpi, A. Sayenko, and J. Moilanen, "Comparison of Different Scheduling Algorithms for WiMAX Base Station: Deficit Round-Robin vs. Proportional Fair vs. Weighted Deficit Round-Robin," *IEEE Wireless Communications and Networking Conference*, pp. 1991–1996, Mar. 2008.

[9] R. D. Armstrong, P. Sinha, and A. A. Zoltners, "The multiple-choice nested knapsack model," *Management Science*, vol. 28, no. 1, pp. pp. 34–43, 1982.

[10] A. Koc, "Wimax system evaluation methodology," *WiMAX Forum*, p. 209, 2008.