A Unified Graph Labeling Algorithm for Consecutive-Block Channel Allocation in SC- FDMA

Lei Lei, Di Yuan, Chin Keong Ho and Sumei Sun

Linköping University Post Print

N.B.: When citing this work, cite the original article.

©2009 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Lei Lei, Di Yuan, Chin Keong Ho and Sumei Sun, A Unified Graph Labeling Algorithm for Consecutive-Block Channel Allocation in SC-FDMA, 2013, IEEE Transactions on Wireless Communications, (12), 11, 5767-5779.

<http://dx.doi.org/10.1109/TWC.2013.092313.130092>

Postprint available at: Linköping University Electronic Press

<http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-102855>

A Unified Graph Labeling Algorithm for Consecutive-Block Channel Allocation in SC-FDMA

Lei Lei¹, Di Yuan¹, Chin Keong Ho², and Sumei Sun²

¹Department of Science and Technology, Linköping University, Sweden

²Institute for Infocomm Research (I²R), A*STAR, Singapore
Emails: {lei.lei; di.yuan@liu.se}, {hock; sunsm}@i2r.a-star.edu.sg

Abstract—Optimal channel allocation is a key performance engineering aspect in single-carrier frequency-division multiple access (SC-FDMA). In SC-FDMA with localized channel assignment, the channels of each user must form a consecutive block. Subject to this constraint, various performance objectives, such as maximum utility, minimum power, and minimum number of channels, have been studied. We present a unified graph labeling algorithm for these problems, based on the structural insight that SC-FDMA channel allocation can be modeled as finding an optimal path in an acyclic graph. By this insight, our algorithm applies the concept of labeling and label domination that represent non-trivial extensions of finding a shortest or longest path. The key parameter in trading performance versus computation is the number of labels kept per node. Increasing the number ultimately enables global optimality. The algorithm’s approach is further justified by its global optimality guarantee with strong polynomial-time complexity for two specific scenarios, where the input is user-invariant and channel-invariant, respectively. For the general case, we provide numerical results demonstrating the algorithm’s ability of attaining near-optimal solutions.

Keywords: algorithm, channel allocation, optimization, single carrier frequency division multiple access

I. INTRODUCTION

Single-carrier frequency-division multiple access (SC-FDMA) has attracted much attention in recent years [1]. In comparison to orthogonal frequency-division multiple access (OFDMA), SC-FDMA offers lower peak-to-average-power ratio (PAPR) [1]. For uplink communications with SC-FDMA, the power amplifier at the mobile user can then transmit at a smaller backoff from the peak power, ultimately leading to less energy consumption. As such, SC-FDMA has been adopted as the uplink multiple access scheme in the Third Generation Partnership Project Long Term Evolution (3GPP-LTE) standard [2].

For OFDMA systems, optimizing system performance deals with resource allocation among users in the frequency domain, and to some extent power assignment over the channels. These topics have been extensively addressed. Many algorithms have been proposed for OFDMA channel allocation with the objective of maximizing the sum utility or minimizing the power (e.g., [3]–[5]). The algorithms are mainly heuristics, justified by that OFDMA resource allocation is in general NP-hard even if the rate function is well-behaved [6], [7].

In SC-FDMA systems, there are two schemes of assigning channels to users: localized and interleaved [1]. In the former,

each user is assigned a block of channels that are consecutive in the spectrum. In the latter, aka the distributed scheme, the channels of a user is spread out, with equal spacing between them. Performance comparison and system implementation of the schemes are addressed in [1], [8], [9].

There has been an increasing amount of attention on various resource optimization aspects of SC-FDMA with the localized channel allocation scheme [10]–[16]. In the current paper, we investigate optimal channel allocation problems with localized channel allocation. We remark that our analysis and solution algorithm are applicable to the problem class of optimal channel allocation subject to the consecutive-channel constraint, for which SC-FDMA for LTE uplink serves as a good example of practical systems. In [13], it has been identified that the constraint, by itself, makes uplink scheduling a hard problem. Thus optimization approaches exploiting the constraint’s structure are of significance. To this end, we focus on understanding the complexity of and developing algorithmic notions for channel allocation when the constraint is in place, with applicability to multiple optimization objectives.

Heuristic algorithms for SC-FDMA scheduling and channel allocation and their performance evaluation are presented in [11]–[14], [16]. Besides maximizing the sum utility, other performance measures have been considered [15], [17]. As mobile users typically employ battery-powered handsets, it is of significance to consider minimum sum power, subject to meeting specified uplink demand target. Another problem amounts to minimizing the number of allocated channels required to meet given demand target [15]. This problem setting is relevant to making as much resource available as possible for elastic traffic, while guaranteeing the rates for real-time applications. In comparison to maximum utility, the problems of minimizing sum power and the number of consumed channels with consecutive-channel allocation are less studied.

In this paper, we present a unified study of consecutive-channel resource allocation for the aforementioned three objectives. We do not restrict to any particular rate function in order to stress the generality of the proposed approach. The key contributions are as follows.

- We prove that the three resource allocation problems are NP-hard. For utility maximization, our conclusion extends the known complexity result. For the two mini-

mization problems, our NP-hardness proof is original.

- We provide the structural insight that allocating blocks of consecutive channels optimally can be mapped to finding an optimal path in an acyclic graph. With this insight, we develop a unified graph labeling algorithm (GLA). The algorithm carries the following features. First, multiple labels are used for each graph node, which allows multiple partial solutions to be tentatively stored, enabling better solution quality than allocating channels in a pure greedy manner. Second, the labels are organized into buckets, where each bucket corresponds to partial solutions with the same number of users that have been allocated channel blocks so far. This bucket classification eases competition among partial solutions to those in the same bucket, and hence avoids an overly-greedy approach. Third, we introduce the concept of label domination to identify partial solutions that can be dropped without any loss of optimality.
- By treating bucket size as an algorithm parameter, the GLA allows a flexible trade-off of complexity and system performance. Analytically, the algorithm ultimately approaches global optimum when the bucket size is sufficiently large.
- We present and formally prove the global optimality guarantee of GLA with bucket size one for two specific scenarios, namely, where the input is either user-invariant or channel-invariant. These analytical results further support the rationale of the algorithm design.
- Numerically, GLA is highly competitive in attaining close-to-optimal solutions. Even with bucket size one, the performance is superior in comparison to known algorithms in the literature for all three objectives. Moreover, GLA exhibits very promising performance in reaching feasible allocations for the two minimization problems, where finding a feasible solution is challenging for high user demand.

II. SYSTEM MODEL AND PROBLEM DEFINITIONS

Let $\mathcal{M} \triangleq \{1, \dots, M\}$ and $\mathcal{N} \triangleq \{1, \dots, N\}$ denote the sets of users and channels, respectively. For uplink, the users in $\mathcal{M}$ send data concurrently to a base station. Each user has a total power limit, denoted by P^u . Moreover, for a user, the power has to be equal on all allocated channels, subject to a given channel peak power limit P^s . Therefore, a user being allocated n channels will use power $\min\{\frac{P^u}{n}, P^s\}$ on each channel. Throughout the paper, we use the term channel to refer to the resource unit in the frequency domain, solely for the sake of convenience. The term is interchangeable with subchannel and resource block (RB) of the LTE uplink specification.

For localized channel allocation, i.e., subject to the consecutive-block constraint, the total number of blocks of possible consecutive channels equals $1 + 2 + \dots + N = \frac{N(N+1)}{2}$, for a total of N channels. We use $\mathcal{B}$ to denote the set of these $\frac{N(N+1)}{2}$ channel blocks. Each element $b \in \mathcal{B}$ is a set containing the consecutive channels of the corresponding block. Thus a channel allocation corresponds to M mutually disjoint channel blocks $b_1, b_2, \dots, b_M \in \mathcal{B}$.

We use u_{ib} to denote the utility value of assigning block b to user i . The utility may include the rate achieved by user i on b with power $\min\{\frac{P^u}{|b|}, P^s\}$ on each channel, as well as weighing and scaling factors determined by a scheduler (e.g., for proportional fairness). We use $f(i, j, p)$ to denote the rate of user i on channel j with power level p . Here, we take the function f to be given and do not restrict to any particular rate function (e.g., a logarithmic function), or scaling parameter setting in u_{ib} , in order to stress the generality of the proposed solution approach. The first problem of maximizing sum utility is formalized below. Note that $\emptyset$ denotes the option of not assigning any channel block to a user.

[Max-Utility] Find a channel-user allocation $b_1, b_2, \dots, b_M \in \mathcal{B} \cup \{\emptyset\}$ maximizing $\sum_{i \in \mathcal{M}} u_{ib_i}$ subject to $b_i \cap b_h = \emptyset, \forall i, h \in \mathcal{M}, i \neq h$. $\square$

The second problem we consider is to minimize the total uplink power required to support given user demand target, denoted by d_i for user i . Again, for the sake of not losing generality, we do not assume any specific power function (i.e., the inverse of $f(i, j, p)$). Instead, we use p_{ib} to denote the total power required to satisfy user i 's demand on channel block b . As power has to be equal on all channels in the block, $p_{ib} = \{p : \min |b| \sum_{j \in b} f(i, j, p) \geq d_i\}$, subject to $|b|p \leq P^u$ and $p \leq P^s$. Given f , this minimization is straightforward (e.g., bi-section search assuming $f(i, j, p)$ is monotonic in p). If the power limits are not exceeded, the allocation is *feasible*, otherwise the allocation is *infeasible*.

[Min-Power] Find a channel-user allocation $b_1, b_2, \dots, b_M \in \mathcal{B}$ minimizing $\sum_{i \in \mathcal{M}} p_{ib_i}$ where b_i is a feasible allocation for user $i, \forall i \in \mathcal{M}$, subject to $b_i \cap b_h = \emptyset, \forall i, h \in \mathcal{M}, i \neq h$, or determine it is not feasible to meet the demands of all users within the power limits. $\square$

The third problem considered in the literature [14] is to minimize the number of channels used subject to demand targets. This problem, as formally defined below, is similar to **Min-Power** except for the cost function.

[Min-Channel] Find a channel-user allocation $b_1, b_2, \dots, b_M \in \mathcal{B}$ minimizing $\sum_{i \in \mathcal{M}} |b_i|$ where b_i is a feasible allocation for user $i, \forall i \in \mathcal{M}$, subject to $b_i \cap b_h = \emptyset, \forall i, h \in \mathcal{M}, i \neq h$, or determine it is not feasible to meet the demands of all users within the power limits. $\square$

Max-Utility, Min-Power, and Min-Channel can be formulated by means of integer programming models [10]–[12], [14]. Although integer linear programming (ILP) is not considered a very practical approach, it is useful for gauging the performance of polynomial-time but sub-optimal algorithms.

III. PROBLEM COMPLEXITY

For SC-FDMA resource allocation, it is widely accepted that global optimality is not computationally tractable. Yet few investigations [18] have dealt with this aspect with formal proofs. Note that ILP formulations do not qualify as evidence of hardness, because most tractable combinatorial optimization problems (e.g., matching) can be modeled by ILP. In this section, we conclude the NP-hardness of **Max-Utility** by extending slightly the result of a previous work. Next, as an original contribution, we present and formally prove the NP-hardness of **Min-Power** and **Min-Channel**.

Theorem 1. Max-Utility is NP-hard.

Proof: In [18], Lee et al. provided an NP-hardness proof based on a polynomial-time reduction from Hamiltonian path to the problem of maximizing SC-FDMA utility, assuming that the rate of every user on each channel is given as an input parameter. Consider the special case of **Max-Utility** where $P^s \leq \frac{P^u}{N}$. To maximize the utility function value, the power on a channel is always P^s for all users, and thus the corresponding rate for each channel is independent of channel block size. In this case, **Max-Utility** reduces to the problem considered in [18], and the result follows. ■

Theorem 2. Min-Power and Min-Channel are NP-hard.

Proof: The proof uses a polynomial-time reduction from the well-known 3-satisfiability (3-SAT) problem that is NP-complete [19]. Consider a 3-SAT instance with m Boolean variables $x_1, x_2, \dots, x_m$, and n clauses. A variable or its negation is referred to as a literal. Denote by $\hat{x}_i$ the negation of $x_i, i = 1, \dots, m$. Each clause is composed by a disjunction of exactly three distinct literals, e.g., $(x_1 \vee \hat{x}_2 \vee x_3)$. The 3-SAT problem amounts to determining whether or not there exists an assignment of true/false values to the variables, such that all clauses are satisfied (i.e., at least one literal has value true in every clause). In the sequel, we use t_i and $\hat{t}_i$ to denote the numbers of clauses in which variable x_i and its negation $\hat{x}_i$ appear, respectively. It is assumed that no clause contains both a variable and its negation; such clauses become always satisfied, thus they can be eliminated by pre-processing. Moreover, a literal appears in at least one clause as otherwise the corresponding value assignment is trivial. Consequently, a literal is present in at most $n - 1$ clauses.

We perform a duplication process, by replacing the t_i occurrences of variable x_i in the clause by separate Boolean variables $x_{i1}, \dots, x_{it_i}$. The duplication is carried out for the negation $\hat{x}_i$ as well. It is obvious that, as long as all the literals corresponding to a common variable in the original 3-SAT instance take the same Boolean value, and the literals representing its negation all take the opposite Boolean value, the satisfiability problem after duplication remains equivalent to the original one. Next, we construct a reduction from the 3-SAT instance after duplication as follows. The number of users is $M = m + n$, referred to as literal and clause users, respectively. The number of channels is $N = \sum_{i=1}^m (t_i + \hat{t}_i) + m + (m + 1)n^2$. Among the channels, $\sum_{i=1}^m (t_i + \hat{t}_i)$ channels are referred to as the literal channels. For literal x_i , there are t_i literal channels representing the occurrences of this literal in the clauses. For convenience, the literal notation is reused for these channels. Thus $x_{i1}, \dots, x_{it_i}$ denote the t_i literal channels created for x_i . In addition to the literal channels, the channel set contains m channels that we refer to as dummy channels, denoted by symbol y . The remaining $(m + 1)n^2$ channels are called surplus channels, denoted by symbol s .

The channel sequence has two parts. The first part contains m segments for the m variables. For variable x_i , the segment consists in the t_i literal channels, a dummy channel, followed by the $\hat{t}_i$ literal channels for the negation $\hat{x}_i$. The second part of channels consists of the $(m + 1)n^2$ surplus channels. In symbol

representation, the channels are in the order $x_{11} \cdots x_{1t_1} y \hat{x}_{1\hat{t}_1} \cdots \hat{x}_{1\hat{t}_1} \cdots x_{i1} \cdots x_{it_i} y \hat{x}_{i\hat{t}_i} \cdots \hat{x}_{i\hat{t}_i} \cdots x_{m1} \cdots x_{mt_m} y \hat{x}_{m\hat{t}_m} \cdots \hat{x}_{m\hat{t}_m} s \cdots s$. Without loss of any generality, we assume the rate function f can achieve value 1.0, and set the demand to be 1.0 for all users. Denote by p a positive integer. We set $P^u = (m + 1)np$, and $P^s = p$, and construct our **Min-Power** and **Min-Channel** instances such that possible channel assignments for the literal and clause users are as specified below. It is easily verified that finding a specification of p , channel gain values, and the rate function f itself (e.g., the logarithmic function) to meet the conditions is straightforward.

For each literal user i , $p_{i\{x_{i1}, \dots, x_{it_i}\}} = p_{i\{\hat{x}_{i\hat{t}_1}, \dots, \hat{x}_{i\hat{t}_{\hat{t}_i}\}} = p$. That is, the demand of this user can be met using total power p on either of the two literal channel blocks $x_{i1}, \dots, x_{it_i}$ and $\hat{x}_{i1}, \dots, \hat{x}_{i\hat{t}_i}$. The remaining channels are inferior such that no allocation other than the two literal channel blocks is power-feasible for i . Thus literal user i has to be allocated $x_{i1}, \dots, x_{it_i}$ or $\hat{x}_{i1}, \dots, \hat{x}_{i\hat{t}_i}$ (without overlap, because of the dummy channel in-between). This corresponds to the Boolean value assignment in the original 3-SAT instance. Thus, for the literal users, the total power equals mp for **Min-Power** for any feasible channel allocation. In **Min-Channel**, a feasible allocation means that the total number of allocated channels for the literal users is strictly less than mn because $t_i < n$ and $\hat{t}_i < n$ for any variable i .

For each clause user, the demand is met with power p on any of the three literal channels in the corresponding clause in the 3-SAT instance with literal duplication. As an example, consider clause $(x_1 \vee \hat{x}_2 \vee x_3)$ and assume, after literal duplication, the clause is $(x_{14} \vee \hat{x}_{21} \vee x_{35})$. The corresponding clause user h has $p_{h\{x_{14}\}} = p_{h\{\hat{x}_{21}\}} = p_{h\{x_{35}\}} = p$. As literal channels in any clause are non-consecutive, a clause user will not split its rate on more than one literal channel in the clause. The other literal channels are all power-infeasible for the clause user. The demand of a clause user can be alternatively met by allocating exactly $(m + 1)n$ (but not less) consecutive surplus channels, with power p on each, with total power $(m + 1)np = P^u$. Thus, each clause user is allocated either one of the three clause channels, or a block of $(m + 1)n$ surplus channels if none of the former is available. This corresponds to whether or not a clause is satisfied in the 3-SAT instance.

By the construction above, which is clearly polynomial, the optimum value of **Min-Power** is no less than $mp + np$. If the 3-SAT instance is satisfiable, this value is indeed attained and hence optimal. Otherwise, at least one clause user is allocated $(m + 1)n$ surplus channels, and the total power is at least $mp + (n - 1)p + (m + 1)np > mp + np$. For **Min-Channel**, the optimal value is no more than $mn + n$ if the 3-SAT instance is satisfiable. If the opposite holds, then the number of channels required, in the best case, is $m + (n - 1) + (m + 1)n$, because the m literal users will use at least m channels, and at least one of the n clause users will consume $(m + 1)n$ surplus channels. The sum is strictly greater than $mn + n$. Thus whether or not there exists a channel allocation with no more than $mp + np$ in power for **Min-Power** or $mn + n$ in the number of channels for **Min-Channel** gives the correct answer to 3-SAT. Therefore the recognition versions of **Min-Power** and **Min-Channel** are NP-complete, and their optimization versions are NP-hard. ■

IV. A GRAPH LABELING ALGORITHM

A. Motivation

The complexity of **Max-Utility**, **Min-Power**, and **Min-Channel** justifies the use of sub-optimal algorithms, such as the ones proposed in [11], [13], [14]. To motivate our algorithmic notion and provide intuition, we take **Max-Utility** as an example and outline two algorithms for the problem: the riding-peaks (RP) algorithm in [13] and the maximum-utility-increase (MUI) algorithm in [14]. The latter is an improvement of a well-known greedy algorithm proposed in [11].

Both MUI and RP allocate one channel in each step. Merely for simplifying the numerical example, suppose P^u is large, such that the utility is only dependent on the user-channel pair and channel peak power P^s . Thus for each user i and channel j , the utility u_{ij} can be pre-calculated. In the RP algorithm, the utility values $\{u_{ij}, i \in \mathcal{M}, j \in \mathcal{N}\}$ are sorted in descending order. The algorithm goes through the sorted list. For each element, the corresponding user-channel assignment is carried out, if the channel is adjacent to those already allocated to this user, or if the user is not yet assigned any channel; otherwise the algorithm moves to the next element in the list. Each time a user-channel assignment is performed, the remaining elements in the list involving the assigned channel are removed. The process terminates until the list is exhausted. The MUI algorithm also makes one user-channel allocation at a time. However, in the MUI algorithm, the metric of a candidate allocation is the incremental utility in relation to the current total utility achieved by the user. Moreover, to account for consecutive channel allocation, the MUI algorithm defines the incremental utility of assigning channel j to user i as the average utility increase of assigning $j-1$, j , and $j+1$ (or two of the three, if $j=0$ or $j=N$) to i . Thus, although each allocation assigns one channel only, the evaluation metric takes into account the average effect of assigning multiple adjacent channels, improving the metric in [13]. The MUI algorithm terminates until all channels are assigned or no further assignment leads to any incremental utility.

For both algorithms, performance calculation in each step is carried out separately for the channels, and for each channel, confined locally to the channel. Moreover, once an assignment is made, modifying the choice in a latter stage is not possible. It is instructive to illustrate why these aspects form the key performance-limiting factors by numerical examples.

Example 1. This example extends a scenario in [18]. Consider two users and N channels. The utility vector of user one is $(u, 0, \frac{u-\epsilon}{2}, \frac{u-\epsilon}{2}, \dots, \frac{u-\epsilon}{2})$, with $0 < u$ and $0 < \epsilon \ll u$. For user two, the vector is $(0, u, 0, 0, \dots, 0)$. Both algorithms will allocate channel one to user one, and channel two to user two, resulting in a total utility of $2u$. At optimum, the total utility is $u + \frac{N-2}{2}(u - \epsilon)$ for $N > 4$ and $\epsilon \rightarrow 0$. For large N , the algorithms' solution is highly sub-optimal.

Example 2. Consider the scenario with $N = 4n$, where n is a positive integer satisfying $n > M$. All users have the utility vector $(\epsilon, u, \epsilon, 0, \epsilon, u, \epsilon, 0, \dots, \epsilon, u, \epsilon, 0)$. The two algorithms will allocate M channel blocks of three channels each, with a total utility of $M(u + 2\epsilon)$. The optimum is to

allocate all channels in one shot, achieving a total utility of $n(u + 2\epsilon)$. The performance gap grows by n . Here, the algorithms fail because the single-channel greedy selection makes the allocations scattered and approaching optimum is then prohibited by the adjacent-channel requirement.

The above illustration and discussion motivate further investigations of algorithms. In particular, it is highly desirable to design an algorithm that not only runs in polynomial time in its base form, but also has the following features.

- 1) The algorithm provides a unified solution approach for **Max-Utility**, **Min-Power**, and **Min-Channel**.
- 2) The algorithm has a mechanism for a more global view than a pure greedy heuristic to overcome the inherent weakness of the latter.
- 3) There is a simple means in the algorithm for the trade-off of computation and solution performance.
- 4) The algorithm, by design, approaches global optimality as computational effort increases.
- 5) Last but not least, the algorithm exhibits the rationale of guaranteeing global optimality for tractable scenarios of **Max-Utility**, **Min-Power**, and **Min-Channel**.

We derive an algorithm exhibiting the above features via graph labeling, based on the structural insight that solving **Max-Utility**, **Min-Power**, and **Min-Channel** can be modeled as finding an optimal path in a graph. The algorithm design overcomes the weakness of the local performance view, as illustrated by the examples, by considering the allocation of channel blocks without restricting the block size, and keeping multiple partial solutions. For Example 1, our algorithm will examine, by its construction (Section IV-C), the allocation of channel three to N as one single block to user one, leading to the global optimum. In Example 2, the users are identical in problem input. For this problem class, the optimality guarantee of our algorithm is formally proven in Section V.

B. Graph Representation

We define a directed and acyclic graph $G = (V, A)$. The node set $V = \{0, \dots, N\}$. The first node (numbered zero) is an auxiliary source node. The remaining nodes represent the N channels. A node pair (j, k) is in the arc set A , if and only if $j < k$. An illustration is provided in Figure 1.

Figure 1. A graph representation of SC-FDMA channel allocation.

In graph G , an arc corresponds to grouping channels into a channel block. Specifically, arc (j, k) represents channel block $j+1, \dots, k$. An arc is associated with M values to specify the performance metric of assigning the block to users in $\mathcal{M}$, i.e., utility, power, or number of channels. Traversing arc (j, k) corresponds to considering allocating the $k-j$ consecutive channels to candidate users that were not yet allocated any channel resource, as well as the option of not

assigning the block to any user. We denote the performance value of allocating channel block $j + 1 \dots k$ to user i as v_{jk}^i .

From Figure 1, a solution to the three channel allocation problems has a unique mapping to a path from node zero to node N in graph G . Each of the arcs in the path is associated with at most one user, and a user may appear at most once along the path. This is in fact a type of optimal path problem, i.e., to find either a longest (**Max-Utility**) or shortest (**Min-Power** and **Min-Channel**) path in the acyclic graph G .

The above structural insight leads to the notion of graph labeling. Labeling is a concept used by various algorithms, for example the Dijkstras algorithm, for solving the shortest path problem (see, e.g., [20] for a comprehensive treatment). The general idea is to create labels at nodes to represent partial paths, and update a label if another partial path with better value is found. In the sequel, we present major extensions of the classical labeling procedure to derive a unified labeling algorithm for **Max-Utility**, **Min-Power**, and **Min-Channel**.

C. Key Components of the Graph Labeling Algorithm (GLA)

Because each user is allocated *at most* one channel block in **Max-Utility**, and *exactly* one channel block in **Min-Power** and **Min-Channel**, a labeling algorithm has to keep track on the set of users for which channel blocks have been allocated. To this end, a label ℓ in GLA is a tuple of format $(v_\ell, M_\ell, \mathcal{M}_\ell)$, where v_ℓ is the accumulated performance value (utility, power, and the number of channels for **Max-Utility**, **Min-Power**, and **Min-Channel**, respectively), $\mathcal{M}_\ell \subseteq \mathcal{M}$ is the associated subset of users for which channel blocks have been allocated, and $M_\ell = |\mathcal{M}_\ell|$ is introduced for the sake of convenience.

In GLA, label processing for an arc has to be carried out for multiple users. Consider two arbitrary nodes j and k with $j < k$, the corresponding arc (j, k) , and a label $\ell = (v_\ell, M_\ell, \mathcal{M}_\ell)$ at node j . The set of candidate users is $\mathcal{M} \setminus \mathcal{M}_\ell$. The option of not assigning the block at all is considered as well. Therefore there are $M - M_\ell + 1$ choices. For the choice of not using the block, label $(v_\ell, M_\ell, \mathcal{M}_\ell)$ appears at node k without any change. For candidate user $i \in \mathcal{M} \setminus \mathcal{M}_\ell$, GLA augments label ℓ , resulting in $(v_\ell + v_{jk}^i, M_\ell + 1, \mathcal{M}_\ell \cup \{i\})$.

A key difference between classical shortest path and optimal path for SC-FDMA channel allocation is that, in the latter, a partial path that is globally optimal may not necessary be locally optimal. Moreover, note that, locally, allocating a channel block to any user is always better than not using it at all in **Max-Utility**, even if the latter choice may be globally optimal. For **Min-Power** and **Min-Channel**, not assigning any user always performs best in the accumulated power or number of channels, but this clearly does not necessarily lead to better performance globally. To account for this aspect, algorithm GLA keeps multiple labels per node, and organizes them into $M + 1$ separate buckets, numbered as $0, 1, \dots, M$ to correspond to the number of users allocated so far. Each bucket stores up to K labels, where $K \geq 1$ is an algorithm parameter. A label ℓ with M_ℓ users is stored as one of the K labels in bucket M_ℓ . The first bucket admits the possibility of zero

user assignment¹. Each label at node j is a partial candidate solution (with a given number of users) for channels one to j . Using a larger K stores more partial candidate solutions and hence potentially improves the overall solution.

Instead of arbitrarily accumulating labels, GLA adopts domination rules to eliminate labels that are evidently non-optimal. Consider two labels $(v_\ell, M_\ell, \mathcal{M}_\ell)$ and $(v_h, M_h, \mathcal{M}_h)$ located at the same node, with $v_\ell \geq v_h$ and $\mathcal{M}_\ell \subseteq \mathcal{M}_h$. Intuitively, for **Max-Utility**, the latter will not achieve a better overall performance, because for the same set of resource, the former yields better or same utility with a subset of the users of the latter. In this case, $(v_\ell, M_\ell, \mathcal{M}_\ell)$ is said to *dominate* $(v_h, M_h, \mathcal{M}_h)$. For **Min-Power** and **Min-Channel**, the reverse relation holds, namely, $(v_\ell, M_\ell, \mathcal{M}_\ell)$ dominates $(v_h, M_h, \mathcal{M}_h)$, if both are present at the same node, and $v_\ell \leq v_h$ and $\mathcal{M}_\ell \supseteq \mathcal{M}_h$. Dominated labels can be dropped without loss of optimality, as formalized in the lemma below.

Lemma 3. *Consider two solutions of allocating the first j channels, denoted by labels $\ell = (v_\ell, M_\ell, \mathcal{M}_\ell)$ and $h = (v_h, M_h, \mathcal{M}_h)$ at node j , respectively, with $1 \leq j \leq N$. For **Max-Utility**, if $v_\ell \geq v_h$ and $\mathcal{M}_\ell \subseteq \mathcal{M}_h$, then ℓ dominates h , that is, for any solution derived from the latter, there exists a solution enabled by the former with better or equal performance. For **Min-Power** and **Min-Channel**, ℓ dominates h if $v_\ell \leq v_h$ and $\mathcal{M}_\ell \supseteq \mathcal{M}_h$.*

Proof: Assume $v_\ell \geq v_h$ and $\mathcal{M}_\ell \subseteq \mathcal{M}_h$. An extension of $(v_h, M_h, \mathcal{M}_h)$ with allocation of channels $j + 1, \dots, N$ is a *complete solution* for **Max-Utility** if and only if none of the channel blocks of $j + 1, \dots, N$ is assigned to any user in $\mathcal{M}_h$. Because $\mathcal{M}_\ell \subseteq \mathcal{M}_h$, any complete solution extending $(v_h, M_h, \mathcal{M}_h)$ is also a complete solution of $(v_\ell, M_\ell, \mathcal{M}_\ell)$. Moreover, the extension with the latter has better and equal performance because $v_\ell \geq v_h$. The corresponding result for **Min-Power** and **Min-Channel** follows analogously. ■

D. Algorithm Summary

Denote by $\mathcal{L}(j, m)$ the set of labels in bucket m of node j . The initialization step of GLA is to set $\mathcal{L}(j, 0) = \{(0, 0, \emptyset)\}$ for $j = 0, \dots, N$. The bulk of the algorithm processes nodes $1, \dots, N$ one by one. For each node, the algorithm creates, evaluates, and stores labels for buckets $1, \dots, M$, each being able to hold up to K labels. Note that, for j with $1 \leq j \leq M - 1$, j buckets are sufficient, as no more than j users can be allocated channels. In general, the number of buckets to be processed is $\min\{j, M\}$.

Consider **Max-Utility**. For each node $j = 1, \dots, N$, GLA sets labels in ascending order of the bucket number, i.e., $m = 1, \dots, M$. For bucket m , its content $\mathcal{L}(j, m)$ is derived by processing the labels present at the previous nodes $i = 1, \dots, j - 1$. Two types of processing are performed as follows, corresponding respectively to the cases that one user and no user is allocated the channel block. First, the labels in bucket $m - 1$ of node i are augmented by assigning channel block $i + 1, \dots, j$ to one additional user. Second, the labels in

¹For row one, $K = 1$ is always sufficient, though we do not explicitly make this distinction in algorithm description for the sake of clarity.

bucket m of node i are reused directly for bucket m of node j . Each new label for $\mathcal{L}(j, m)$ undergoes a domination check. That is, the label is discarded, if it is dominated by any label in $\mathcal{L}(j, n)$ with $n = 1, \dots, m-1$. A non-dominated label is put into bucket m if it is not yet full (i.e., $|\mathcal{L}(j, m)| < K$). If $|\mathcal{L}(j, m)| = K$, the new label replaces the most inferior one in $\mathcal{L}(j, m)$, in case the former has better performance value, otherwise the new label is discarded.

The algorithm terminates when the treatment of node N is complete. At this stage, the label yielding the best performance value in $\mathcal{L}(N, m), m = 1, \dots, M$, represents the outcome. Note that **Min-Power** and **Min-Channel** require that all users are allocated channel blocks, hence only $\mathcal{L}(N, M)$ need to be considered for the final outcome. The flow of computations is given in Algorithm 1. In the description, clarity and compactness are preferred over implementation details. For the same reason, information storage for retrieving the channel allocation solution is omitted. The bulk of computation starts at Line 5. For each node j and its bucket m , the labeling process is performed in Lines 7–20. Notation $\mathcal{L}'$ denotes the set of all tentative labels, derived from labels at a previous node i . Set $\mathcal{L}'$ is obtained by processing the labels in $\mathcal{L}(i, m)$ and $\mathcal{L}(i, m-1)$, respectively. Thus, the labels in $\mathcal{L}(i, m)$ are reused in $\mathcal{L}'$, representing the choice of skipping channel block $i+1, \dots, j$. Each label $\ell \in \mathcal{L}(i, m-1)$ contributes to $M - M_\ell$ labels in $\mathcal{L}'$, obtained via augmenting ℓ by allocating channel block $i+1, \dots, j$ to one of the remaining $M - M_\ell$ users, see Lines 10–11. Each candidate label $\ell' \in \mathcal{L}'$ is subject to domination check in Line 13. A label passing the check enters bucket $\mathcal{L}(j, m)$ either because there is empty space available, or because ℓ' outperforms the currently most inferior label $\bar{\ell}$ in $\mathcal{L}(j, m)$, see Lines 14–15 and Lines 17–20, respectively.

Applying Algorithm 1 to **Min-Power** and **Min-Channel** requires the following minor adaptations. First, the loop starting at Line 6 comes in the reverse order, i.e., m goes from $\min\{j, M\}$ down to one. This is because for **Min-Power** and **Min-Channel**, domination check of labels of bucket m is performed versus labels present in buckets $m+1, \dots, \min\{j, M\}$. Second, the left-hand and right-hand sides in the two conditions of domination check (Line 13) switch their positions, and, in same line, s takes values $m, \dots, \min\{j, M\}$. Third, maximization replaces minimization in Line 17, and the reversed condition applies to Line 18. Finally, the final output is obtained by using $\text{argmin}_{\ell \in \mathcal{L}(N, M)} v_\ell$ in Line 21.

For **Min-Power** and **Min-Channel**, assigning a channel block to a user is infeasible, if the corresponding power exceeds any of the two limits. In GLA, this can be taken care of by skipping the corresponding label, accomplished by adding one additional check before Line 11.

E. Algorithm Complexity

To see the computational complexity of Algorithm GLA, we begin by observing that K is an algorithm parameter that is independent of input size M or N . Therefore, to ease the presentation without any loss of its applicability to complexity, in the following we analyze the amount of computation with $K = 1$, that is, each bucket holds up to one label.

Algorithm 1 The GLA algorithm for Max-Utility.

Input: $\mathcal{M}, \mathcal{N}, \mathcal{B}, u_{i\bar{b}}, i \in \mathcal{M}, \bar{b} \in \mathcal{B}$

Output: ℓ^* (the best label value)

```

1: for  $j = 0 : N$  do
2:    $\mathcal{L}(j, 0) \leftarrow \{(0, 0, \emptyset)\}$ 
3:   for  $m = 1 : M$  do
4:      $\mathcal{L}(j, m) \leftarrow \emptyset$ 
5:   for  $j = 1 : N$  do
6:     for  $m = 1 : \min\{j, M\}$  do
7:       for  $i = 0 : j - 1$  do
8:          $\mathcal{L}' \leftarrow \mathcal{L}(i, m)$ 
9:         for all  $\ell \in \mathcal{L}(i, m - 1)$  do
10:          for all  $w \notin \mathcal{M}_\ell$  do
11:             $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{(v_\ell + v_{i+1,j}^w, M_\ell + 1, \mathcal{M}_\ell \cup \{w\})\}$ 
12:          for all  $\ell' \in \mathcal{L}'$  do
13:            if  $\nexists h \in \mathcal{L}_{js}, s = 1, \dots, m: v_h \geq v_{\ell'} \wedge \mathcal{M}_h \subseteq \mathcal{M}_{\ell'}$  then
14:              if  $|\mathcal{L}(j, m)| < K$  then
15:                 $\mathcal{L}(j, m) \leftarrow \mathcal{L}(j, m) \cup \{\ell'\}$ 
16:              else
17:                 $\bar{\ell} \leftarrow \text{argmin}_{h \in \mathcal{L}(j, m)} v_h$ 
18:                if  $v_{\ell'} > v_{\bar{\ell}}$  then
19:                   $\mathcal{L}(j, m) \leftarrow \mathcal{L}(j, m) \setminus \{\bar{\ell}\}$ 
20:                   $\mathcal{L}(j, m) \leftarrow \mathcal{L}(j, m) \cup \{\ell'\}$ 
21:  $\ell^* \leftarrow \text{argmax}_{\ell \in \mathcal{L}(N, m), m=1, \dots, M} v_\ell$ 
22: return  $\ell^*$ 

```

The bulk of computation is formed by Lines 8–20, consisting in setting labels in bucket m of node j , via the labels in buckets $\mathcal{L}(i, m)$ and $\mathcal{L}(i, m-1)$ of node i . Processing the label in $\mathcal{L}(i, m)$ does not form the computational bottleneck, because it gives one candidate label for bucket $\mathcal{L}(j, m)$, whereas for any label ℓ in $\mathcal{L}(i, m-1)$, $M - M_\ell$ candidate labels must be considered. In the following, we consider the complexity of processing one label in $\mathcal{L}(i, m-1)$. From now on, we assume that the user set $\mathcal{M}_\ell$ of each label ℓ is implemented as a list. Thus reading or writing a label, and a comparison of users of two labels all require $O(M)$ time.

Consider a naive way of processing $\mathcal{L}(i, m-1)$. Its label may lead to $O(M)$ new labels in Lines 10–11. The new labels are subject to domination check versus j 's labels in buckets $\mathcal{L}_{j,1}$ to $\mathcal{L}_{j,m}$. Because there are $O(M)$ labels in these buckets and each check requires $O(M)$ time, the complexity is of $O(M^3)$. However, this complexity can be reduced by one magnitude by a less trivial implementation. The idea is to integrate domination check into the process of creating new labels, instead of creating candidate labels first and then carrying out domination check. For label $\ell \in \mathcal{L}(i, m-1)$, consider examining, one by one, the $O(M)$ labels used in the domination check, located in buckets $\mathcal{L}(j, 1)$ to $\mathcal{L}(j, m)$ of node j . Let h be the label under consideration. Clearly, if label ℓ , after being augmented by one additional user, would be dominated by label h , then either $\mathcal{M}_h \subseteq \mathcal{M}_\ell$, or deleting exactly one element from $\mathcal{M}_h$ will make it a subset of $\mathcal{M}_\ell$. If none of the two conditions holds, h will not dominate any new label derived from ℓ . Examining these conditions obviously

runs in $O(M)$ time. Next, if $\mathcal{M}_h \subseteq \mathcal{M}_\ell$, domination tests for the users in $\mathcal{M} \setminus \mathcal{M}_\ell$ requires $O(M)$ time, because checking if a user is in $\mathcal{M}_\ell$ as well as a performance value comparison run on $O(1)$ time. If potential domination originates from the second condition, domination has to be checked for one user only and requires less time. Following the approach of combining label creation with domination, determining the domination relation between $\ell \in \mathcal{L}(i, m-1)$ and a label h of node j , for all users that potentially augment ℓ , or concluding no domination applies, runs in $O(M)$ time. As there are $O(M)$ labels to check against, the complexity of domination test for label $\ell \in \mathcal{L}(i, m-1)$ becomes of $O(M^2)$.

The above operations result in a binary indicator list for users in $\mathcal{M} \setminus \mathcal{M}_\ell$, representing whether or not augmenting ℓ with each of these users is dominated by some label at node j . The list is of size $O(M)$. Selecting the best-valued element and determining if it shall replace the current label of $\mathcal{L}(m, j)$ require clearly $O(M)$ time. Writing the resulting label of the selection takes $O(M)$ time. Thus, once the domination check for label $\ell \in \mathcal{L}(i, m-1)$ is complete, updating the label in $\mathcal{L}(m, j)$ (Lines 17–20) is of complexity $O(M)$.

From the analysis above, the bulk of computation in Lines 8–20 requires $O(M^2)$ time. Next, note that the loops starting at Lines 5–7 require no more than N , M , and N iterations, respectively, giving $O(M^3N^2)$ as the overall complexity. The result applies provided that the utility values are computed before running GLA. The complexity impact of utility computation, along with complexity comparison with the algorithms for performance evaluation, are detailed in Section VI-E.

V. OPTIMALITY RESULTS

In this section we provide insights into the performance of GLA in terms of optimality. The first result states the ability of the algorithm in ultimately approaching global optimum.

Theorem 4. *For any instance of **Max-Utility**, and any feasible instance of **Min-Power** and **Min-Channel**, there is a finite value of K , for which GLA returns the global optimum.*

Proof: Without loss of any generality, assume that at the global optimum, the first assignment along the channel sequence $1, \dots, N$ is the allocation of channel block $i, \dots, j$ to user m . Label $(0, 0, \emptyset)$ is present at node i in the initialization step. Thus, while generating $\mathcal{L}(j, m)$, label ℓ with $\mathcal{M}_\ell = \{m\}$ will be considered. As there are a finite number of combinations for labels at j , there is a finite value of K for which this label is kept. Repeating the argument, the observation applies to the remaining channel assignments of the global optimum. By Lemma 3, eliminating labels by domination check does not compromise optimality, and the result follows. ■

Next, we present and prove the global optimality guarantee of GLA for two classes of **Max-Utility**, **Min-Power**, and **Min-Channel** that are more structured than the general case. Namely, the input is either user-invariant or channel-invariant. The user-invariant class means that the users are not distinguishable in the problem input. In this case, the channel gains differ by channel but not by user, or, equivalently, all users have the same gain distribution over the channels, which

become of relevance in the scenario where the users are located very close to each other, such that the channel variation between users is negligible. User-invariant input implies also that the users are not differentiated by priority and thus achieve the same utility value for any channel block in **Max-Utility**, and that the users have uniform demand in **Min-Power** and **Min-Channel**. For the channel-invariant class, the channel gains are uniform for every user, but remain nonuniform over the users. This class provides a good approximation if users maintain line-of-sight to the base station, leading to flat-fading channels. Note that identifying the two classes is straightforward with a running time of $O(MN)$. We prove that simplification or slight adaptation of the GLA algorithm with only one label per bucket (i.e., $K = 1$) enables the global optimum for these two problems classes.

For the user-invariant problem class, the performance value of allocating a channel block b does not depend on who the actual user is. Hence, for any channel block, the decision reduces to determining whether or not to allocate the block to any user, whereas to which specific user the block is allocated has no significance. In other words, a solution is fully characterized by at most N non-overlapping channel blocks for **Max-Utility**, and exactly N non-overlapping channel blocks for **Min-Power** and **Min-Channel**. The GLA algorithm is thus simplified – it is sufficient that a label ℓ contains the performance value v_ℓ and the number of users M_ℓ . Moreover, while processing a label and a channel block, the allocation is not user-specific. The theorem below states the optimality of GLA with $K = 1$.

Theorem 5. *Algorithm GLA guarantees global optimality for the user-invariant class of **Max-Utility**, **Min-Power**, and **Min-Channel**, with $K = 1$.*

Proof: Consider **Max-Utility**, and denote by u_b the utility of allocating channel block b to one user. By the observation made above, user index is not needed. Suppose that, at optimum, the first allocated channel block, in the sequence $1, \dots, N$, is $i_1, \dots, j_1$. By algorithm construction, label $(0, 0)$ exists at node i_1 , and the label will be processed with this channel block, generating candidate label ℓ_1 , where $v_{\ell_1} = u_{\{i_1, \dots, j_1\}}$ and $M_{\ell_1} = 1$, for bucket one of node j_1 . No other label at j_1 will outperform label ℓ_1 , because otherwise there is a better solution of allocating channels $1, \dots, j_1$ to a single user. This new single-user allocation, combined with the remaining allocations in the solution in question, contradict the assumption of optimality. Hence ℓ_1 is the final label kept at node j for $K = 1$. Denote by $i_2, \dots, j_2$ the next allocated block at optimum, with $j_1 \leq i_2$. The very same label ℓ_1 is the final result of bucket one at node i_2 , as any label in this bucket having better performance would mean allocating $i_1, \dots, j_1$ is not the optimal choice for channels up to i_2 . While processing ℓ_1 at i_2 , $\ell_2 = (v_{\ell_1} + u_{\{i_2, \dots, j_2\}}, 2)$ is a candidate label for bucket two of node j_2 . Assuming that a label of the same bucket has a better performance value would again contradict the optimality assumption of the solution in question. Repeating the argument, algorithm GLA yields global optimum of **Max-Utility**. The proof with slight modification applies to **Min-Power** and **Min-Channel**, and the theorem follows. ■

For the channel-invariant problem class, searching for optimality can be restricted to solutions without any gap between the allocated channel blocks. The observation follows from the channel-invariant property, which allows the allocated channel blocks to be shifted to eliminate the gaps, if any. Moreover, it is easily realized that, along the sequence of channels $1, \dots, N$, allocation can take place for any fixed order of users, e.g., $1, \dots, M$, without loss of optimality. Based on the properties, we make slight adaptations of GLA as follows. First, in processing any label, the option of not allocating a channel block is excluded. Second, if m is the highest user index in a label, then extending the label applies to users $m+1, \dots, M$. The meaning of M_ℓ in label ℓ is updated accordingly for **Max-Utility**; the notation has the meaning that channel allocation has been performed for users among which the highest index is M_ℓ . Note that for **Min-Power** and **Min-Channel**, the new definition coincides with the original one, because all users must be allocated channel blocks. Below we prove that the adapted algorithm is globally optimal.

Theorem 6. *The adapted version of Algorithm GLA guarantees global optimality for the channel-invariant class of **Max-Utility**, **Min-Power**, and **Min-Channel**, with $K = 1$.*

Proof: Based on the discussion above, there exists an optimal solution such that the channel blocks allocated are contiguous and the allocations follow user indices $1, \dots, M$. Consider such an optimal solution with inter-user break points $i_1, i_2, \dots$, along the sequence $1, \dots, N$, having $m_1, m_2, \dots$ as the corresponding users for channel allocation, i.e., user m_1 is allocated block $1, \dots, i_1$, user m_2 is allocated block $i_1 + 1, \dots, i_2$, and so on. For **Min-Power** and **Min-Channel** $m_k = k, k = 1, \dots, M$, whereas this condition does not hold for **Max-Utility** because not all users are necessarily allocated channel blocks. Starting from label $(0, 0, \emptyset)$ at node zero, label ℓ_1 with $v_{\ell_1} = v_{1i_1}^{m_1}$, $M_{\ell_1} = m_1$ and $\mathcal{M}_{\ell_1} = \{m_1\}$ is a candidate label of bucket m_1 of node i_1 . Assume that there is another label ℓ' outperforming ℓ_1 for the same bucket. The assumption can be immediately discarded for **Min-Power** and **Min-Channel**, because $m_1 = 1$ and this is the only eligible user for the bucket. For **Max-Utility**, the existence of ℓ' means that ℓ' contains user m_1 and additional users with indices smaller than m_1 , giving a contradiction because ℓ' together with the remaining allocations of the solution in question would lead to a better overall solution. Thus the assumption is invalid, and ℓ_1 is the final result of bucket one of node i_1 . For the same reason, while setting labels at node i_2 , extending ℓ_1 with user m_2 leads to the best possible label ℓ_2 for **Max-Utility**, with $v_{\ell_2} = v_{\ell_1} + v_{i_1+1, i_2}^{m_2}$, $M_{\ell_2} = m_2$, and $\mathcal{M}_{\ell_2} = \{m_1, m_2\}$, as otherwise a contradiction arises. For **Min-Power** and **Min-Channel**, user $m_2 = 2$. If there is a better label than ℓ_2 at i_2 for bucket two, then it means there is a better allocation of channels $1, \dots, i_2$ to the first two users. This, together with the remaining allocations, contradict the optimality assumption. Repeating the argument for the remaining allocations completes the proof. ■

Theorems 5–6 not only support the rationale of graph labeling but also provide understanding of the two structured problem classes' complexity, as stated in the following corol-

lary. The result follows immediately from the polynomial-time complexity of the GLA algorithm.

Corollary 7. *The user-invariant and channel-invariant classes of **Max-Utility**, **Min-Power**, and **Min-Channel** have polynomial-time tractability.*

VI. PERFORMANCE EVALUATION

A. Experimental Setup

For performance evaluation, we consider SC-FDMA uplink of a cell with randomly and uniformly distributed users. Table I summarizes the key parameters. The channel gain consists of path loss, shadowing, as well as Rayleigh fading. The path loss follows the widely used COST 231 model that extends the Okumura-Hata model for urban scenarios. By the COST 231 model, path loss is frequency dependent. Log-normal shadowing model with 8 dB standard deviation is used [14]. A channel corresponds to a resource block in LTE with twelve subcarriers. To gain a comprehensive performance picture, four sets of data with $(M, N) = (10, 64), (20, 64), (10, 128)$, and $(20, 128)$, respectively, have been used. Cell size as well as the number of users and channels are comparable to those in previous simulations [11], [13], [14]. For each data set, we generate 100 instances and consider the average performance. Note that the theoretical results in Sections III and V hold for any given deterministic channel realization, regardless of the channel model used.

Table I
SIMULATION PARAMETERS.

Parameter	Value
Cell radius	1000 m
Carrier frequency	2 GHz
Number of users M	10, 20
Number of channels N	64, 128
Channel bandwidth	180 KHz
Path loss	COST-231-HATA
Shadowing	Log-normal, 8 dB standard deviation
Multipath fading	Rayleigh fading
User power limit P^u	200 mW
Channel peak power limit P^s	10 mW
Noise power spectral density	-174 dBm/Hz

We examine three performance aspects. First, performance evaluation of GLA has been carried out with respect to global optimal solution and several other known heuristic algorithms for SC-FDMA resource allocation. Next, we examine the impact of parameter K on algorithm performance. Finally, for **Min-Power** and **Min-Channel**, the feasibility aspect is considered by progressively increasing the demand and conducting a comparative study of GLA's performance in finding a feasible channel allocation.

We remark that the system model (Section II) and the GLA algorithm are not restricted to any particular definition of the utility function or power function. For performance comparison, the utility values in **Max-Utility** are set to be the data rates derived from the logarithmic function. The setting is coherent with the literature [11], [14]. Thus, for user i and channel block b , the utility is $u_{ib} = \sum_{j \in b} B \log_2(1 + \frac{\min\{P^u/|b_i|, P^s\}g_{ij}}{\sigma^2})$, where B is the channel bandwidth, g_{ij}

is the channel gain for i on channel j , and σ^2 is the noise power spectral density times the channel bandwidth. For **Min-Power** and **Min-Channel**, the assignment is feasible if the achieved rate over the channel block meets the demand, i.e., if $\sum_{j \in \mathcal{B}} B \log_2(1 + \frac{\min\{P^u/|\mathcal{B}_i|, P^s\} g_{ij}}{\sigma^2}) \geq d_i$, otherwise the assignment is infeasible and cannot be performed. Uniform demand is used for **Min-Power** and **Min-Channel**, with $d_i = 1$ Mbps, $\forall i \in \mathcal{M}$. If assigning block $\mathcal{B}$ to user i is feasible, the cost of the assignment is clearly $|\mathcal{B}_i| \cdot \min\{P^u/|\mathcal{B}_i|, P^s\}$ in **Min-Power**, and for **Min-Channel** the cost parameter is simply the number of channels, i.e., $|\mathcal{B}_i|$.

For benchmarking, integer linear programming (ILP) proposed for **Max-Utility**, **Min-Power**, and **Min-Channel** (see [10]–[12], [14]) has been used to compute the global optimum. Throughout this section, the term optimality gap refers to the relative difference in comparison to the benchmarking results by ILP, obtained via [21].

Four previously proposed algorithms have been implemented for comparison. For **Max-Utility**, the maximum-utility-increase (MUI) algorithm [14] and the riding-peaks (RP) algorithm [13], which have been presented in Section IV-A, are considered. For **Min-Power** and **Min-Channel**, the minimum-power-decrease (MPD) algorithm and the block-allocation-for-minimum-number-of-subchannels (BMNS) in [14] are used respectively for comparison. The MPD algorithm adapts MUI for **Min-Power**. In each step, the algorithm computes the power reduction of allocating one channel to a user, subject to that the channel is adjacent to those previously allocated to the user. The allocation giving the highest power reduction is selected. The process is repeated until the channels are exhausted. Note that the power limits P^s and P^u are relaxed in the process. If the final allocation violates any of the limits, the algorithm is not successful in solving **Min-Power**. The BMNS algorithm allocates one channel block to one user in every step. Block selection is based on its size (i.e., the cost function of **Min-Channel**). For each user, the difference between the sizes of the smallest and second smallest channel block that are available and satisfy the user demand is computed. The user having the largest difference value is selected, to whom the feasible channel block of smallest size is allocated. The pool of available channels is then updated accordingly. The algorithm stops when all users have been allocated channel blocks or none of the blocks left is feasible for any of the remaining users. The latter case represents an unsuccessful solution for **Min-Channel**.

B. Performance in Optimality

Table II summarizes the performance values of the algorithms. Each entry in the table is the average of 100 instances. For GLA, parameter K is set to ten. In Figure 2, the corresponding optimality gaps of the algorithms' performance values in Table II, in terms of the respective relative deviation from column "optimum", are further illustrated for comparison. We make the following observations based on the results.

- For **Max-Utility**, GLA delivers a total throughput being very close to what is optimally achievable, see Table

Figure 2. Optimality gap for **Max-Utility**, **Min-Power**, and **Min-Channel**.

II. The optimality gap is 2% or less in Figure 2(a). Algorithms MUI and RP give significantly lower utility, resulting in an optimality gap ranging between 12% and 18%. For MUI and RP, doubling N leads to noticeably larger optimality gap, whereas doubling M has the opposite effect; the latter can be attributed to that, for

Table II
PERFORMANCE COMPARISON OF THE ALGORITHMS.

	Max-Utility (Mbps)			
	Optimum	GLA	MUI	RP
(10, 64)	62.21	61.88	54.58	53.53
(20, 64)	70.99	70.59	62.65	62.34
(10, 128)	125.59	123.87	104.00	104.64
(20, 128)	142.27	141.01	120.36	121.62
	Min-Power (mW)			
	Optimum	GLA	MPD	
(10, 64)	116.72	128.98	145.26	
(20, 64)	381.61	426.98	491.81	
(10, 128)	82.63	89.47	101.98	
(20, 128)	290.81	319.16	370.37	
	Min-Channel (channels)			
	Optimum	GLA	BMNS	
(10, 64)	23.80	24.02	25.37	
(20, 64)	54.04	55.58	58.71	
(10, 128)	18.76	18.99	20.18	
(20, 128)	50.54	51.89	54.67	

many users, greedy channel allocation is likely a good choice for maximizing utility. For GLA, the results have little fluctuation. The main limitation of algorithms MUI and RP, as was highlighted in Section IV-A, is that performance evaluation of each channel-user assignment is carried out locally for the channel. That MUI may achieve better utility value than RP is most likely attributed to the fact of accounting for channel variation by considering the two adjacent channels for each candidate assignment in the former algorithm. By construction, GLA takes a more global view than both MUI and RP, by considering allocation of blocks of any size. This is the underlying reason that GLA outperforms both MUI and RP.

- For problem **Min-Power**, the total power consumption by GLA in Table II is much closer to optimum in comparison to the MPD algorithm. The optimality gap for **Min-Power** in Figure 2 is greater than that for **Max-Utility**, indicating that the former problem is harder. This is explained by a higher competition among the users because all demand targets have to be fulfilled. The explanation is indeed confirmed by Figure 2(b), as the performance of both algorithms consistently improves when there is more resource (i.e., number of channels) available, even if the problem size becomes larger. Increasing the number of users results in larger gap, although the growth is moderate.
- Examining the results for **Min-Channel** in Table II, one observes that, in terms of the number of channels consumed, both algorithms GLA and BMNS deliver values being fairly close to the minimum. GLA remains consistently better, however, as can be seen from the values as well as from Figure 2(c). Similar to **Min-Power**, increasing the number of users leads to harder instances for **Min-Channel**, because of higher competition.
- By the numerical results in Table II and Figure 2, the empirical hardness of the problems grows in the order **Max-Utility**, **Min-Channel**, and **Min-Power**, due to the following reasons. First, solving **Min-Channel** and **Min-Power** has to deal with the fulfillment of user demands that is not present in **Max-Utility**. Second, for **Min-Channel**, there is structural preference of always allocating a smaller channel block to a user, as long as the demand can be met. **Min-Power**, on the other hand, does not submit to the same preference.

C. The Impact of Number of Labels

As our next part of numerical study, the impact of parameter K on algorithm performance is evaluated. The results are displayed in Table III. For each combination of M and N , the table provides the optimal value, the performance value of GAL, as well as the optimality gap in percentage.

As expected, the performance of GLA improves progressively in respect of K . Note that, even with $K = 1$, GLA still outperforms significantly the other algorithms for **Max-Utility** and **Min-Power** (cf. Table II and Figure 2). For **Min-Channel** and $K = 1$, the performance of GLA becomes close to that of the BMNS algorithm, yet GLA remains noticeably better.

Using a large K , the largest gap goes below 1% for **Max-Utility**, 10% for **Min-Power**, and 2% for **Min-Channel**. The improvement is steep up to $K = 25$. The results also confirm the previous observation that **Min-Power** is empirically harder than the other two problems.

The results in Table III further shed light on the relation between problem structure and algorithm performance. For the two minimization problems, the optimality gap grows in M . Recall that the number of potential labels to be considered is directly related to M . When M increases and K is kept constant, the likelihood of accommodating a locally inferior but globally better partial solution decreases. In **Max-Utility**, the allocation is utility based – increasing M does not necessarily lead to worse performance because of better user diversity, see Table III. However, sub-optimality due to problem size does appear for N for **Max-Utility**. In contrast, for **Min-Power** larger N consistently brings the solutions closer to the optimal values, because resource competition among the users is more dominating than problem size. For **Min-Channel**, the performance difference due to N is very small.

D. Success Rates for Min-Power and Min-Channel

For **Min-Power** and **Min-Channel**, the success rate of finding a feasible channel allocation is of significance. Determining the feasibility of **Min-Power** and **Min-Channel** is NP-complete (see Section III). To examine the aspect computationally, we consider $(M, N) = (10, 64)$ and increase successively the user demand from 1 Mbps to 2 Mbps. While the demand grows, more and more user-block pairs become infeasible for resource allocation, and thus finding an overall feasible resource allocation becomes increasingly challenging.

Performance comparison is carried out for GLA with $K = 1$ and $K = 10$, and then algorithms MPD and BMNS in [14] for **Min-Power** and **Min-Channel**, respectively. The results are provided in Figure 3. For the demand target of 1 Mbps, all algorithms are able to achieve feasibility for all instances. For MPD and BMNS, the drop in the success rate over demand is apparent. The decrease becomes sharp when the demand goes beyond 1.4 Mbps. For GLA with $K = 1$, the degradation in success rate is significantly smaller and more gradual. Setting

Table III
PERFORMANCE OF GLA WITH RESPECT TO K .

	Max-Utility (Mbps, gap in %)				
	Optimum	$K=1$	$K=10$	$K=25$	$K=50$
(10, 64)	62.21	61.39, 1.32	61.88, 0.53	61.95, 0.42	62.04, 0.27
(20, 64)	70.99	70.08, 1.28	70.59, 0.56	70.68, 0.44	70.72, 0.38
(10, 128)	125.59	123.21, 1.90	123.87, 1.37	124.15, 1.15	124.43, 0.92
(20, 128)	142.27	140.40, 1.31	141.01, 0.89	141.21, 0.75	141.31, 0.67
	Min-Power (mW, gap in %)				
	Optimum	$K=1$	$K=10$	$K=25$	$K=50$
(10, 64)	116.72	130.90, 12.15	128.98, 10.50	127.66, 9.37	126.87, 8.70
(20, 64)	381.61	431.25, 13.01	426.98, 11.89	421.14, 10.36	418.13, 9.57
(10, 128)	82.63	90.77, 9.85	89.50, 8.31	88.70, 7.35	88.55, 7.16
(20, 128)	290.81	320.96, 10.37	319.16, 9.75	317.43, 9.15	316.49, 8.83
	Min-Channel , (channels, gap in %)				
	Optimum	$K=1$	$K=10$	$K=25$	$K=50$
(10, 64)	23.80	25.13, 5.59	24.02, 0.92	23.92, 0.50	23.84, 0.17
(20, 64)	54.04	57.60, 6.59	55.58, 2.85	55.23, 2.20	54.83, 1.46
(10, 128)	18.76	19.89, 6.02	18.99, 1.23	18.89, 0.69	18.81, 0.27
(20, 128)	50.54	53.85, 6.55	51.89, 2.67	51.54, 1.98	51.24, 1.39

Figure 3. Success rate of feasible allocation.

$K = 10$ clearly improves the success rate. In fact, with this setting GLA is capable of delivering feasible allocations with a 100% success rate for the entire demand range.

E. Time Complexity Comparison

There are M users and N channels with $M \leq N$. Clearly, the number of blocks of consecutive channels is of magnitude $O(N^2)$. Because the per-channel power depends on block size, calculating the utility in **Max-Utility** or determining feasibility in **Min-Power** and **Max-Utility** of one user-block assignment requires $O(N)$ time. Overall, to simply examine the performance values of all candidate user-block assignments (with or without making any allocation) has a time complexity of $O(MN^3)$. Therefore $O(MN^3)$ sets a reference of computational complexity, because any algorithm considering the $O(MN^2)$ possible user-block assignments will have a time complexity of $O(MN^3)$ and in fact also $\Omega(MN^3)$.

From the above, it is clear that the only way of getting a lower complexity than the reference one of $O(MN^3)$ is to carry out performance calculation separately for the channels, without bundling channels together to form candidates of channel block assignments. Algorithms RP, MUI, and MPD do follow this approach. As there are $O(MN)$ user-channel assignments to select among per iteration, and at most N iterations are required, the three algorithms run in $O(MN^2)$ time. However, the local view induced by single-channel allocation disregards the fact that the performance of a user is a result of its assigned channels all together, not the individual channels. This weakness is illustrated by Examples 1 and 2 in

Section IV-A, and further confirmed by the numerical results in Sections VI-B and VI-D.

Algorithm BMNS does compute the candidate user-block assignments in its construction. Once the values are available, it follows from the description in [14] that the bulk of computational steps requires $O(M^2N^2)$ time. Thus the overall complexity is $\max\{O(M^2N^2), O(MN^3)\} = O(MN^3)$ because $M \leq N$, matching exactly the reference value.

By the analysis in Section IV-E, algorithm GLA runs in $O(M^3N^2)$ time, provided that the user-block performance values are available, leading to an overall complexity of $\max\{O(M^3N^2), O(MN^3)\}$. Which of the two terms dominates depends on the relation between M^2 and N . Thus the complexity is bounded by one magnitude (in the number of users) higher than the reference, though the view is pessimistic because typically M is significantly smaller than N . This moderate increase in complexity in comparison to the reference value is justified by three facts. First, GLA provides a unified solution approach for **Max-Utility**, **Min-Power**, and **Min-Channel**, in contrast to the other algorithms that are specific for one of the problems. Second, GLA delivers superior performance in terms of optimality and success rate, as shown in Sections VI-B and VI-D. Third, GLA has global optimality guarantee for the two tractable problem classes (Section V), whereas the other algorithms do not.

VII. CONCLUDING REMARKS

We have presented an algorithmic approach for three SC-FDMA resource allocation problems, namely **Max-Utility**, **Min-Power**, and **Min-Channel**. The approach exploits the problems' structure, to enable the view of these problems as finding an optimal path in a graph. The resulting GLA algorithm exhibits three features. First, solution procedures for tackling the three resource allocation problems are unified within a common algorithm framework of graph labeling. Second, the algorithm parameter allows for a trade-off between computation and optimality. Third, the design guarantees global optimality for the user-invariant and channel-invariant classes of all three problems.

Performance evaluation shows that the proposed algorithmic framework is highly competitive in attaining close-to-optimal solutions. The results also demonstrate the effect of the amount of label storage on performance. In addition, the algorithmic design yields high success rate of delivering feasible solutions for **Min-Power** and **Min-Channel**.

Although the algorithm and the theoretical insights have been derived for a specific setup of uplink resource allocation, the results apply and extend along the following lines. The observations not only strengthen the applicability of the current work, but also reveal interesting topics for further research.

- Following the basic algorithm design (Section IV), the GLA algorithm is not restricted to the assumption of equal-power distribution over the channels. Indeed, the performance of assigning channel blocks to users are part of the input. Thus the algorithm and the theories in Section V are applicable to other types of power settings, such as power allocation with water-filling to maximum the sum rate over the assigned channel block.

- The algorithmic concept is also applicable to scenarios where the performance values (e.g., utility) of channel assignment address a fairness metric, such as proportional fairness (PF) at uplink [13]. Thus the GLA algorithm can act as the core module in a scheduling context with fairness embedded in defining the performance value.
- The current work has been motivated by SC-FDMA for LTE uplink. However, as long as the consecutive-channel constraint is present, our results extend to other technologies (not necessarily cellular networks) as well as both directions of communications. For example, a WiMAX system scenario that includes allocation of adjacent channels in downlink is described in [22]. For downlink, typically only a peak power P^s applies, which however does not affect the validity of the algorithm.
- In a multi-cell setup, the uplink transmission of a user generates interference to cells other than the serving cell. Provided that cells coordinate their resource allocation, the generalization to a multi-cell scenario amounts to augmenting a label to represent a joint decision of allocating a channel block to users in respective cells, with the effect of accounting for interference. Thus, using the notion of graph labeling and extending it with interaction between cells, the current work forms a basis for uplink resource allocation in the multi-cell case.

From the above discussion, an extension for further investigation consists of performance studies of uneven power assignment and resource allocation with a fairness metric that is adapted over time. Another interesting line of research is to generalize the solution approach to the multi-cell setup with presence of interference.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable comments and suggestions. The work of the first author has been supported by the Chinese Scholarship Council (CSC) and the overseas phd research internship scheme from Institute for Infocomm Research (I²R), A*STAR, Singapore. The work of the second author has been supported by the Linköping-Lund Excellence Center in Information Technology (ELLIIT), Sweden.

REFERENCES

- [1] H. G. Myung, J. Lim, and D. J. Goodman, "Single carrier FDMA for uplink wireless transmission," *IEEE Vehicular Technology Magazine*, vol. 1, no. 3, pp. 30–38, Sept. 2006.
- [2] 3rd Generation Partnership Project, Technical Specification Group Radio Access Network, "Physical layer aspects for evolved universal terrestrial radio access (utra)," 3GPP TR 25.814 v.7.1.0, Oct. 2006.
- [3] D. Kivanc, G. Li, and H. Liu, "Computationally efficient bandwidth allocation and power control for OFDMA," *IEEE Transactions of Wireless Communications*, vol. 2, no. 6, pp. 1150–1158, Nov. 2003.
- [4] A. Feiten, R. Mathar, and M. Reyer, "Rate and power allocation for multiuser OFDM: An effective heuristic verified by branch-and-bound," *IEEE Transactions of Wireless Communications*, vol. 7, no. 1, pp. 60–64, Jan. 2008.
- [5] J. Joung, D. Yuan, C. K. Ho, and S. Sun, "Energy efficient network-flow-based algorithm for multiuser multicarrier systems," *IET Networks*, vol. 1, no. 2, pp. 66–73, Jun. 2012.
- [6] D. Yuan, J. Joung, C. K. Ho, and S. Sun, "On tractability aspects of optimal resource allocation in OFDMA systems," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 2, pp. 863–873, Feb. 2012.

- [7] Y. Liu and Y.-H. Dai, "On the complexity of joint subcarrier and power allocation for multi-user OFDMA systems," Dec. 2012. [Online]. Available: <http://arxiv.org/abs/1212.5024>
- [8] H. Nam, "Interpolation-based SC-FDMA transmitter with localized resource allocation," *IEEE Communications Letters*, vol. 14, no. 10, pp. 948–950, Oct. 2010.
- [9] S. H. Song, G. L. Chen, and K. B. Letaief, "Localized or interleaved? a tradeoff between diversity and CFO interference in multipath channels," *IEEE Transactions on Wireless Communications*, vol. 10, no. 9, pp. 2829–2834, Sept. 2011.
- [10] A. Ahmen and M. Assaad, "Polynomial-complexity optimal resource allocation framework for uplink SC-FDMA systems," in *Proc. of IEEE GLOBECOM*, Dec. 2011, pp. 1–5.
- [11] I. C. Wong, O. Oteri, and W. McCoy, "Optimal resource allocation in uplink SC-FDMA systems," *IEEE Transactions on Wireless Communications*, vol. 8, no. 5, pp. 2161–2165, May 2009.
- [12] D. Kim, J. Kim, and H. Kim, "An efficient scheduler for uplink single carrier FDMA system," in *Proc. of IEEE PIMRC*, Sept. 2010, pp. 1348–1353.
- [13] S.-B. Lee, I. Pefkianakis, A. Meyerson, S. Xu, and S. Lu, "Proportional fair frequency-domain packet scheduling for 3GPP LTE uplink," in *Proc. of IEEE INFOCOM*, Apr. 2009, pp. 2611–2615.
- [14] F. I. Sokmen and T. Girici, "Uplink resource allocation algorithms for single-carrier FDMA systems," in *Proc. of IEEE European Wireless Conference*, Apr. 2010, pp. 339–345.
- [15] H. Safa and K. Tohme, "LTE uplink scheduling algorithms: performance and challenges," in *Proc. of IEEE International Conference on Telecommunications (ICT)*, Apr. 2012, pp. 1–6.
- [16] L. R. de Temifio, G. Berardinelli, S. Frattasi, and P. Mogensen, "Channel-aware scheduling algorithms for SC-FDMA in LTE uplink," in *Proc. of IEEE PIMRC*, Sept. 2008, pp. 1–6.
- [17] W. Vereecken, W. V. Heddeghem, M. Deruyck, B. Puype, B. Lannoo, W. Joseph, D. Colle, L. Martens, and P. Demeester, "Power consumption in telecommunication networks: overview and reduction strategies," *IEEE Communications Magazine*, vol. 49, no. 6, pp. 62–69, Jun. 2011.
- [18] S.-B. Lee, I. Pefkianakis, A. Meyerson, S. Xu, and S. Lu, "Proportional fair frequency-domain packet scheduling for 3GPP LTE uplink," University of California, Los Angeles, USA, Tech. Rep. UCLA TR-0900001, 2009.
- [19] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [20] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, 1993.
- [21] "IBM ILOG CPLEX Optimizer," <http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/>, 2012.
- [22] A. Nussairat and X.-Y. Li, "WiMAX/OFDMA burst scheduling algorithm to maximize scheduled data," *IEEE Transactions on Mobile Computing*, vol. 11, no. 11, pp. 1692–1705, Nov. 2012.

and optimization, and energy-efficient communications.

Lei Lei received his B.Eng. degree in electronic information engineering and M.Eng. degree in weapon systems and utilization engineering (First-Class Honors) at Northwestern Polytechnical University, Xi'an, China, in 2008 and 2011, respectively. He is currently working toward the Ph.D. degree at the Department of Science and Technology, Linköping University, Sweden. Since June 2013, he has been a research assistant at Institute for Infocomm Research (I²R), A*STAR, Singapore. His current research interests include wireless network resource allocation

Di Yuan received his MSc degree in computer science and engineering, and PhD degree in operations research at Linköping Institute of Technology in 1996 and 2001, respectively. At present he is full professor in telecommunications at the Department of Science and Technology, Linköping University, and head of a research group in mobile telecommunications. His current research mainly addresses network optimization of 4G systems.

Dr Yuan has been guest professor at Technical University of Milan (Politecnico di Milano), Italy, in 2008, and senior visiting scientist at Ranplan Wireless Network Design Ltd, United Kingdom, in 2009 and 2012. In 2011 and 2013 he has been part time with Ericsson Research, Sweden. He is an area editor of the Computer Networks journal. He has been in the management committee of four European Cooperation in field of Scientific and Technical Research (COST) actions, invited lecturer of European Network of Excellence EuroNF, and Principle Investigator of five European FP7 Marie Curie projects. He is co-recipient of IEEE ICC'12 Best Paper Award.

Chin Keong Ho (S'05-M'07) received the B.Eng. (First-Class Hons., Minor in Business Admin.), and M. Eng degrees from the Department of Electrical Engineering, National University of Singapore in 1999 and 2001, respectively. He obtained his Ph.D. degree at the Eindhoven University of Technology, The Netherlands, where he concurrently conducted research work in Philips Research. Since August 2000, he has been with Institute for Infocomm Research (I²R), A*STAR, Singapore. He is currently Lab Head of Energy-Aware Communications Lab,

Department of Modulation and Coding, in I²R. His research interest includes green wireless communications with focus on energy-efficient solutions and with energy harvesting constraints; cooperative and adaptive wireless communications; and implementation aspects of multi-carrier and multi-antenna communications.

Sumei Sun (SM'12) received the B.Sc. (with honors) degree from Peking University, Beijing, China; the M.Eng. degree from Nanyang Technological University, Singapore; and the Ph.D. degree from National University of Singapore, Singapore. She has been with Institute for Infocomm Research (I²R), Agency for Science, Technology, and Research (A*STAR), Singapore, since 1995, where she is currently Head of the Modulation and Coding Department, developing energy- and spectrum-efficient technologies for the next-generation communication

systems. Her recent research interests include 5G transmission technologies, renewable energy management and cooperation in wireless systems and networks, and wireless transceiver design.

Dr. Sun served as the Technical Program Committee Chair of the 12th IEEE International Conference on Communications Systems (ICCS) in 2010, General Co-chair of the Seventh and Eighth IEEE Vehicular Technology Society Asia Pacific Wireless Communications Symposium, and Track Co-chair of Transmission Technologies of the 75th IEEE Vehicular Technology Conference in 2012. She is also an Associate Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY and an Editor for IEEE WIRELESS COMMUNICATION LETTERS. She was a co-recipient of the 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications Best Paper Award.