# A Near-Optimal Randomized Algorithm for Uplink Resource Allocation in OFDMA Systems

Yang Yang*, Changwon Nam†, Ness B. Shroff*

*Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH
†Department of EECS, Seoul National University, Korea

*Abstract*—**OFDMA has been selected as the multiple access scheme for emerging broadband wireless communication systems. However, designing efficient resource allocation algorithms for OFDMA systems is a challenging task, especially in the uplink, due to the combinatorial nature of subcarrier assignment and the distributed power budget for different users. Inspired by Glauber dynamics, in this paper, we propose a randomized iteration-based uplink OFDMA resource allocation algorithm. We show that our algorithm is near-optimal in the sense that by increasing the number of iterations (which scales up the complexity), with arbitrarily large probability, the algorithm can converge to the subcarrier/power allocation pattern with the maximum sum-utility. We also show that this algorithm can be generalized to solve a joint uplink-downlink allocation problem in full-duplex OFDMA systems. Simulations are conducted to compare the performance of our algorithm with existing ones.**

## I. Introduction

Orthogonal frequency division multiplexing has been the key technology for most wide-band communication systems due to its low-complexity implementation, efficient and flexible spectrum management, and robustness against frequency selective fading. By dividing the available spectrum into many parallel subcarriers and allowing adaptive modulation techniques on each subcarrier, OFDMA scheme inherently permits both multi-user diversity and frequency diversity at a fine granularity. To fully exploit such promising gains, it is essential to have a radio resource allocation algorithm that efficiently manage both subcarrier and power assignment among all users.

However, designing resource allocation algorithms is a challenging task, especially in the uplink, due to the following reasons: 1) The exclusive nature of the subcarrier assignment leads to an integer optimization problem, which is generally hard to solve. 2) Different users may have different power constraints, average channel qualities, or even rate-utility functions, which makes the problem even harder.

Due to these inherent difficulties, most existing works have focused on using the insight gained from either the necessary condition of optimality, or the optimal solution for a relaxed version of the integer optimization problem to develop suboptimal algorithms. For example, in [1], the Lagrangian method is used to obtain a necessary condition for the resource allocation pattern with the maximum sum-rate, then two greedy allocation algorithms are developed based on the intuition obtained from the necessary condition. In [2], the authors generalize the necessary condition to the sum-utility maximization problem and develop a new algorithm. They further show that the algorithm is Pareto-optimal within a large neighborhood. In [3], the authors relax the exclusive subcarrier assignment constraint, allow different users to access the same subcarrier using orthogonalization, and derive an optimal solution to this relaxed problem. This optimal solution, in turn, is used to guide the design of a suboptimal algorithm that maximize the sum weighted rate of all users.

Different from existing algorithms which are mostly derived from the Lagrangian methods, our work is based on the so-called Glauber dynamics from statistical physics [4], [5]. Glauber dynamics, originally a model of how physical systems reach equilibrium, has been well-studied for its use as a Markov Chain Monte Carlo (MCMC) algorithm, which samples from probability distributions by constructing a Markov chain that has the desired distribution as its unique stationary distribution. It is a versatile tool with many applications such as graph coloring, approximate counting, and sampling of independent set. Glauber dynamics also leads to the recent development of throughput-optimal queue-length-based dynamic CSMA algorithms [6], [7], which has become a very active research area (a good survey is presented in [8]).

In our work, the idea of Glauber dynamics is used to develop a randomized resource allocation algorithm for single-cell OFDMA systems. Starting from any resource allocation pattern, our algorithm tries to alter the allocation pattern in a randomized and iterative fashion, where the alteration in each iteration is limited to only two or three users, and the probability of each alteration is carefully chosen such that the optimal allocation pattern is found with high probability when the Markov chain representing the iterations reaches steady state.

The main contribution of this paper is the following:

- We propose an iterative randomized joint-subcarrier-power allocation algorithm for uplink OFDMA systems, in which the allocation pattern evolves as a

Markov chain, and the steady state probability of the allocation pattern with the maximum sum-utility can be made arbitrarily close to one, which we term "near-optimal". (Section III)

- By integrating some changes into this randomized algorithm, we develop an enhanced version of the randomized algorithm that has better performance under finite number of iterations. (Section IV)
- Unlike existing uplink resource allocation algorithms that have fixed complexity and performance, our proposed algorithms can trade off complexity against performance, in the sense that by increasing the number of iterations, the optimal allocation can be found with higher probability. Also, given that our algorithm is near-optimal as the number of iterations approaches infinity, it can also be used to benchmark the performance of other algorithms for different network configurations. (Section VI)
- We show that the randomized algorithm can be easily generalized to solve a joint uplink-downlink resource allocation problem in a OFDMA system where the BS and all users have wireless full-duplex capabilities. (Section V)

## II. System Model

We consider the problem of joint subcarrier assignment and power allocation in a single-cell uplink OFDMA system with $K$ mobile users and $S$ subcarriers. Since each subcarrier can only be assigned to a single user, the total number of all possible subcarrier assignment patterns is $K^S$. To represent an allocation pattern, we use an $S-$dimension vector such that the $s^{\text{th}}$ element is $k$ if the $s^{\text{th}}$ subcarrier is allocated to the $k^{\text{th}}$ user. We use $V$ to denote such a vector, and use superscripts (e.g., $V^{(x)}$ and $V^{(y)}$) to differentiate between different allocation patterns.

The set of all possible subcarrier allocation patterns is denoted as $\mathcal{V} = \{1, 2, \ldots, K\}^S = \left\{V^{(x)} \middle| x \in \mathbb{N}, 1 \leq x \leq K^S\right\}$, where $V^{(x)} = \left[V^{(x)}(1), V^{(x)}(2), \ldots, V^{(x)}(S)\right]$ for any $x$. Without loss of generality, when $K \leq S$, we assume that the first $S!/(S-K)!$ allocation patterns are the ones that assign at least one subcarrier for each user, and denote the set of all such allocation patterns as $\mathcal{W} = \left\{V^{(x)} \in \mathcal{V} \middle| x \in \mathbb{N}, 1 \leq x \leq S!/(S-K)!\right\}$

We assume an AWGN channel for each user in every subcarrier, and denote the normalized signal to noise ratio for user $k$ in subcarrier $s$ as $g_{s,k}$. We also assume that the scheduler has the complete knowledge of all the channel SNRs $\{g_{s,k} | 1 \leq s \leq S, 1 \leq k \leq K\}$. The power budget for user $k$ is denoted as $P_k$ and the utility function $U_k$ of user $k$ is assumed to be a non-decreasing function of the rate of user $k$.

For a fixed subcarrier allocation pattern $V^{(x)}$, it is well-known that the optimal power allocation for user $k$ is the water-filling solution over its assigned subcarriers. More precisely, the optimal power allocation of user $k$ on subcarrier $s$ is

$$p_{s,k}\left(V^{(x)}\right) = \mathbf{1}_{\left\{V^{(x)}(s)=k\right\}} \left[\nu_k - 1/g_{s,k}\right]^+,$$

where $[x]^+ = \max\{0, x\}$, and $\nu_k$ is a constant which is commonly called the water level of user $k$ and satisfies

$$\sum_{s=1}^{S} p_{s,k}\left(V^{(x)}\right) = P_k, \text{ for any } 1 \leq k \leq K.$$

Based on the water-filling solution, we can then calculate the maximum utility for user $k$ under a fixed allocation pattern $V^{(x)}$, which is

$$U\left(k, V^{(x)}\right) = U_k\left(\sum_{s=1}^{S} \log\left(1 + g_{s,k} p_{s,k}\left(V^{(x)}\right)\right)\right).$$

Finally, the joint subcarrier and power allocation problem for the uplink OFDMA system can be formulated as the utility maximization problem below.

$$\max_{1 \leq x \leq K^S} \sum_{k=1}^{K} U\left(k, V^{(x)}\right). \tag{1}$$

Note that the above formulation is quite general in the sense that (a) We do not assume *either concavity or differentiability* on the utility function, and thus our results can be applied to a large class of utility functions. (b) Even though we are dealing with instantaneous sum-utility maximization, this formulation can be used to solve the sum-utility maximization problem for the long-term average user rate through a gradient-based scheduling framework [3], given that the utility function of the long-term average rate is concave and differentiable.

## III. Randomized Algorithm

Next, we propose a randomized iteration algorithm in which the allocation pattern keeps evolving, where the pseudo-code is provided in Algorithm 1. We use superscripts in square bracket (e.g., $V^{[t]}$) to indicate the allocation pattern in different iterations.

### A. Algorithm Description

At the beginning of the algorithm, we pick an initial allocation pattern $V^{[0]}$ by letting all subcarriers sequentially and randomly choose a user among the ones that have the least number of associated subcarriers to associate with. Then, the system enters an iteration loop, where the allocation pattern at the beginning of the $t^{\text{th}}$ iteration is denoted as $V^{[t]}$. In each iteration, the system tries to generate a new allocation pattern $\overline{V}^{[t]}$, also called the candidate allocation for the next iteration, by following some simple steps. Initially, $\overline{V}^{[t]}$ is set to be a replica of $V^{[t]}$.

**[Step 1]** Based on $V^{[t]}$, the base-station picks a user-pair $\{k_A, k_B\}$ according to the following procedure: pick a subcarrier uniformly at random, and denote the user it

**Algorithm 1:** **R**andomized **A**lgorithm [$T$ iterations]

**Data**: $S$ subcarriers, denoted as $\mathcal{S} = \{1, 2, \ldots, S\}$; $K$ users, denoted as $\mathcal{K} = \{1, 2, \ldots, K\}$; $\mathbb{P}_{\text{flip}} \in (0, 1)$; $\alpha > 0$.

**Initialization**
- **for** $s = 1 \to S$ **do**
  - randomly pick a user $k$ with the least number of subcarriers and set $V^{[0]}(s) = k$;

**for** *every allocation iteration* $t = 0 \to T - 1$ **do**
- set $\overline{V}^{[t]} = V^{[t]}$;
- **Step-1 [pick a user-pair** $\{k_A, k_B\}$**]**
  - randomly pick $s$ from $\mathcal{S}$ and set $k_A \triangleq V^{[t]}(s)$;
  - pick $k_B$ from $\mathcal{K} \backslash \{k_A\}$ uniformly at random;
- set $p_1 = \begin{cases} 1 & \text{with probability } \mathbb{P}_{\text{flip}} \\ 2 & \text{otherwise} \end{cases}$;
- **if** $p_1 = 1$ **then**
  - **Step-2a [flip operation]**
    - pick $s_{AB}$ from $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k_A \text{ or } k_B\right\}$ uniformly at random;
    - set $\overline{V}^{[t]}(s_{AB}) = k_B$ if $V^{[t]}(s_{AB}) = k_A$;
    - set $\overline{V}^{[t]}(s_{AB}) = k_A$ if $V^{[t]}(s_{AB}) = k_B$;
- **else**
  - **Step-2b [swap operation]**
    - **if** $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k_A\right\}$ *and* $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k_B\right\}$ *are both nonempty* **then**
      - pick $s_A$ from $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k_A\right\}$ uniformly at random;
      - pick $s_B$ from $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k_B\right\}$ uniformly at random;
      - set $\overline{V}^{[t]}(s_A) = k_B$ and $\overline{V}^{[t]}(s_B) = k_A$;
- **Step-3 [self-transition or not]**
  - $p_2 = \begin{cases} 1 & \text{w.p. } \mathbb{P}_3\left(\overline{V}^{[t]}; k_A, k_B, V^{[t]}, \alpha\right) \\ 0 & \text{otherwise} \end{cases}$;
  - set $V^{[t+1]} = \overline{V}^{[t]}$ if $p_2 = 1$;
  - set $V^{[t+1]} = V^{[t]}$ if $p_2 = 0$;

**Result**: $V = V^{[T]}$.

is associated with in $V^{[t]}$ as $k_A$. Then, from the rest of the users, randomly pick one and denote it as $k_B$.

Next, the algorithm follows either **Step 2a** or **Step 2b**. It follows **Step 2a** with probability $\mathbb{P}_{\text{flip}}$, and **Step 2b** with probability $1 - \mathbb{P}_{\text{flip}}$. Both of these two steps try to modify $\overline{V}^{[t]}$.

**[Step 2a]** Among all the subcarriers that are allocated to user $k_A$ or $k_B$ in $\overline{V}^{[t]}$, randomly pick one and toggle its associated user between $\{k_A, k_B\}$.

**[Step 2b]** If both user $k_A$ and user $k_B$ have subcarriers that are associated with them in $\overline{V}^{[t]}$, then both users randomly pick one of its associated subcarriers, and the two chosen subcarriers switch their associated users. Otherwise, no operation is performed.

After the above steps, **Step 3** is carried out to determine if $\overline{V}^{[t]}$ is accepted as the next allocation or not.

**[Step 3]** The system computes the maximum utility that can be achieved by user $k_A$ and $k_B$ under $\overline{V}^{[t]}$ by finding the water-filling solution according to their power budgets as well as the channel qualities of their associated subcarriers in $\overline{V}^{[t]}$. In other words, the system finds $U(k_A, \overline{V}^{[t]})$ and $U(k_B, \overline{V}^{[t]})$. Then, the candidate allocation pattern $\overline{V}^{[t]}$ is accepted to be $V^{[t+1]}$ with probability $\mathbb{P}_3$. Here $\mathbb{P}_3$ is calculated according to the equation below.

$$\mathbb{P}_3\left(\overline{V}^{[t]}; k_A, k_B, V^{[t]}, \alpha\right) = \frac{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k, \overline{V}^{[t]}\right)\right)}}{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k, \overline{V}^{[t]}\right)\right)} + \prod_{k=k_A,k_B} e^{\alpha\left(U\left(k, V^{[t]}\right)\right)}}, \tag{2}$$

where $\alpha > 0$ is a parameter of the algorithm which we will talk about later. In the case when $\overline{V}^{[t]}$ is rejected, $V^{[t+1]}$ is set to be the same as $V^{[t]}$.

### B. Algorithm Analysis

In this part, we establish the near-optimality property of the randomized algorithm. First, we show in the lemma below that the iterations in Algorithm 1 follow a Markov chain.

**Lemma 1.** *$\{V^{[t]}\}_t$ in Algorithm 1 evolves as an irreducible and aperiodic discrete-time Markov chain, with the state space being 1) $\mathcal{V}$, if $U_k(0) \neq -\infty$ for all $k$; 2) $\mathcal{W}$, if $K \leq S$ and $U_k(0) = -\infty$ for all $k$.*

*Proof.* Because $V^{[t+1]}$ is constructed based on the previous allocation pattern $V^{[t]}$ as well as some random variables determined by $V^{[t]}$, $\{V^{[t]}\}_t$ forms a Markov chain. In the following proof, we first focus on the case when $U_k(0) \neq -\infty$.

To show that the chain is irreducible, assume w.l.o.g. that $V^{(a)}$ is an $S-$dimension vector with all the entries equal to 1. This means that all the subcarriers are allocated to the 1st user under allocation $V^{(a)}$. Since the utility function will never be negative infinity, we know, according to Equation (2), that any candidate allocation pattern $\overline{V}^{[t]}$ can be accepted with a non-zero probability. Then, starting from any arbitrary allocation pattern, by following **Step 2a** in Algorithm 1, we can flip the user association of all the subcarriers to the 1st user, which means $V^{(a)}$ is accessible from any allocation pattern in $\mathcal{V}$. Similarly, any allocation pattern in $\mathcal{V}$ is accessible from $V^{(a)}$. Therefore, all states in $\mathcal{V}$ communicate with each other and the chain is irreducible.

In case when $U_k(0) = -\infty$ for all $k$, Equation (2) indicates that if the allocation pattern in any iteration belongs to $\mathcal{W}$, the transition probability to any allocation pattern in $\mathcal{V} \backslash \mathcal{W}$ will be zero. Then by a similar argument as the previous paragraph, we can show that the states in $\mathcal{W}$ communicate with each other by following a combination of flip and swap operations, and have positive self transition probabilities, which implies that $\mathcal{W}$ forms a closed communicating class and the chain is irreducible over state space $\mathcal{W}$. □

By Lemma 1 and Theorem 4.3.3 in [9], we know that the Markov chain has a unique stationary distribution. In the next proposition, we find the stationary distribution of this Markov chain.

**Proposition 1.** *The stationary distribution of the discrete time Markov chain $\{V^{[t]}\}$ is*

$$\pi\left(V^{(x)}\right) \triangleq \mathbb{P}\left(V^{[\infty]} = V^{(x)}\right) = \frac{1}{Z}\prod_{k=1}^{K} e^{\left(\alpha U\left(k, V^{(x)}\right)\right)} = \frac{1}{Z} e^{\left(\alpha \sum_{k=1}^{K} U\left(k, V^{(x)}\right)\right)},$$

*where*

$$Z = \sum_{y=1}^{K^S}\prod_{k=1}^{K} e^{\left(\alpha U\left(k, V^{(y)}\right)\right)}.$$

*Proof.* We demonstrate the correctness of this product-form stationary distribution by verifying the local balance equations below.

$$\pi\left(V^{(x)}\right)\mathbb{P}\left(V^{(x)} \to V^{(y)}\right) = \pi\left(V^{(y)}\right)\mathbb{P}\left(V^{(y)} \to V^{(x)}\right), \forall 1 \le x, y \le K^S,$$

where $\mathbb{P}\left(V^{(x)} \to V^{(y)}\right)$ denotes the one-step transition probability from state $V^{(x)}$ to $V^{(y)}$, and it is the product of two probabilities,

$$\mathbb{P}\left(V^{(x)} \to V^{(y)}\right) = \mathbb{P}\left(\overline{V}^{[t]} = V^{(y)} \middle| V^{[t]} = V^{(x)}\right) \times \mathbb{P}_3\left(V^{(y)}; k_A^{(x,y)}, k_B^{(x,y)}, V^{(x)}, \alpha\right),$$

where $k_A^{(x,y)}$ and $k_B^{(x,y)}$ denote the two users that have different subcarrier allocations in $V^{(x)}$ and $V^{(y)}$. The first two steps in Algorithm 1 are designed such that

$$\mathbb{P}\left(\overline{V}^{[t]} = V^{(y)} \middle| V^{[t]} = V^{(x)}\right) = \mathbb{P}\left(\overline{V}^{[t]} = V^{(x)} \middle| V^{[t]} = V^{(y)}\right), \forall 1 \le x, y \le K^S.$$

Note that $\mathbb{P}\left(V^{(x)} \to V^{(y)}\right) = 0$ when there are more than two users that have different subcarrier allocations in $V^{(x)}$ and $V^{(y)}$.

Then, it is easy to check that the local balance equation holds by plugging in the expression of $\mathbb{P}_3$ in Equation (2). A detailed version of the proof can be found in [10]. □

In the following proposition, we show that, by tuning the parameter $\alpha$, the stationary distribution for the allocation pattern with the maximum sum-utility can be made arbitrarily close to one, which justifies our claim that the randomized algorithm is near-optimal.

**Proposition 2.** *Define*

$$x^* \triangleq \arg\max_{1 \le x \le K^S} \sum_{k=1}^{K} U\left(k, V^{(x)}\right).$$

*$x^*$ is the index of the allocation pattern for which the system achieves the maximum sum-utility. Assume $x^*$ is unique, then for any $\epsilon > 0$, we can find an $\alpha$ large enough such that $\pi\left(V^{(x^*)}\right) > 1 - \epsilon$.*

*Proof.* Let

$$\beta \triangleq \sum_{k=1}^{K} U(k, V^{(x^*)}) - \max_{x \ne x^*}\sum_{k=1}^{K} U(k, V^{(x)}).$$

Since $x^*$ is unique, $\beta$ is strictly positive. For any positive $\epsilon > 0$, let $\alpha = \frac{1}{\beta}\log\frac{(1-\epsilon)K^S}{\epsilon}$. We have

$$\begin{aligned}\pi\left(V^{(x^*)}\right) &= \frac{\prod_{k=1}^{K}\exp\left(\alpha U\left(k, V^{(x^*)}\right)\right)}{\sum_{x=1}^{K^S}\prod_{k=1}^{K}\exp\left(\alpha U\left(k, V^{(x)}\right)\right)} \\ &\ge \frac{1}{1 + (K^S - 1)\exp(-\alpha\beta)} \\ &\ge \frac{1}{1 + K^S\frac{\epsilon}{(1-\epsilon)K^S}} = 1 - \epsilon.\end{aligned}$$

□

### C. Algorithm Complexity

We adopt a coarse measurement of the complexity by simply counting the number of times the function for obtaining the water-filling solution is invoked. As in Algorithm 1, it is clear that, for each iteration, the function for water-filling solution is invoked at most twice to compute $\mathbb{P}_3$ when the candidate allocation pattern $\overline{V}$ is different from the previous allocation pattern in user $k_A$ and $k_B$.

Although Proposition 2 suggests that as the number of iterations approaches infinity, the algorithm will converge to the optimal allocation with high probability, in practical implementation, the number of iterations is limited by the computational power of the scheduler. To account for such limitation, we set the number of water-filling operations as an argument of Algorithm 1, and denote the algorithm as RA($w$). In other words, the iterations in RA($w$) will terminate after the algorithm has invoked the water-filling operation for $w$ times.

In the following section, we propose an enhanced version of this randomized algorithm by adding some new features. We denote the enhanced algorithm as ERA($w$), and show through simulations in Section VI that under the same complexity constraint $w$, ERA($w$) has a better performance than RA($w$).

## IV. Enhanced Algorithm

In this part, we obtain an enhanced version of the randomized algorithm, namely ERA, by adding some features, where the pseudo-code is provided in Algorithm 2. We will use Figure 1 to illustrate the benefits of these features at a conceptual level.

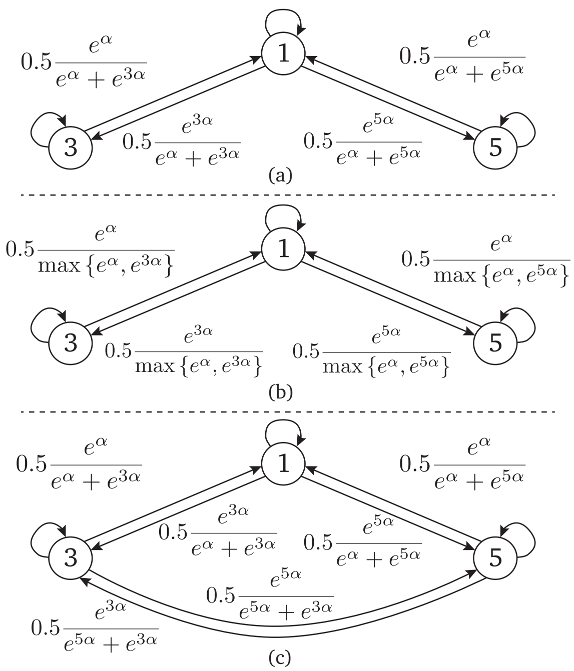


Fig. 1. Three discrete-time Markov chains with the same stationary distribution. The number on each state represents its utility. The stationary distribution for state 1, 3, and 5 is $\vec{\pi} = [0.090, 0.244, 0.665]$ for $\alpha = 0.5$ and $\vec{\pi} = [0.0000, 0.0003, 0.9997]$ for $\alpha = 4$.

In the DTMC shown in Figure 1(a), there are three states, with the number on each state representing its utility. Since each state has at most two neighboring states, we set the probability of selecting any neighboring state as the candidate state to be $0.5$. According to Equation (2) and Proposition 1, by choosing the transition probability as shown in the figure, the stationary distribution for each state is proportional to the exponential of its utility. More precisely, by choosing $\alpha = 0.5$, the stationary distribution for state 1, 3, and 5 is $\vec{\pi} = [0.090, 0.244, 0.665]$, while for $\alpha = 4$, $\vec{\pi} = [0.0000, 0.0003, 0.9997]$, which shows that the steady state distribution of the state with the maximum utility can be made arbitrarily close to $1$ by tuning $\alpha$.

Central to the performance of our randomized algorithm under finite number of iterations is the mixing time of the underlying Markov chain, i.e., the time it takes for the chain to converge to the stationary distribution we want. Smaller mixing time indicates less iterations and less complexity. While the analysis of the mixing time is out of the scope of this work, here we use several techniques to improve the performance of our randomized algorithm under finite iterations.

### A. Initial Condition

Instead of randomly choosing the subcarrier assignment for each user as the initialization step in Algorithm 1, we can allow each subcarrier to greedily choose the best user to associate with. This technique, to some extent, would reduce the distance of the initial allocation to the optimal allocation in terms of number of jumps, compared with the random initial allocation, and thus has the potential to decrease the number of iterations needed to reach the optimal allocation.

### B. Transition Probability

From the proof of Proposition 1, we know that the key design trick that makes the underlying Markov chain of the randomized iteration reversible and has desirable stationary distribution is to make sure the following equation holds.

$$\frac{\mathbb{P}_3\left(\overline{V}^{[t]}; k_A, k_B, V^{[t]}, \alpha\right)}{\mathbb{P}_3\left(V^{[t]}; k_A, k_B, \overline{V}^{[t]}, \alpha\right)} = \frac{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k,\overline{V}^{[t]}\right)\right)}}{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k,V^{[t]}\right)\right)}},$$

where $\mathbb{P}_3$ is the probability of the transition from one state to another, when a certain transition direction is chosen. While it is easy to check that the expression of $\mathbb{P}_3$ in Equation (2) satisfies the above equality, we can find a new transition probability larger than $\mathbb{P}_3$ that gives the exact same stationary distribution. We denote it as $\mathbb{P}_{3E}$ and it is given as

$$\mathbb{P}_{3E}\left(\overline{V}^{[t]}; k_A, k_B, V^{[t]}, \alpha\right) = \frac{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k,\overline{V}^{[t]}\right)\right)}}{\max\left\{\prod_{k=k_A,k_B} e^{\alpha\left(U\left(k,\overline{V}^{[t]}\right)\right)}, \prod_{k=k_A,k_B} e^{\alpha\left(U\left(k,V^{[t]}\right)\right)}\right\}}.$$

Since $\mathbb{P}_{3E} \geq \mathbb{P}_3$, we know that the transition probability for each pair of neighboring states increases after changing $\mathbb{P}_3$ to $\mathbb{P}_{3E}$. As a result, the conductance of the Markov chain, which is a measure of how well connected the state space graph is, is larger with $\mathbb{P}_{3E}$ than that with $\mathbb{P}_3$. [11] shows that the mixing time of a reversible Markov chain can be upper bounded by a value that is inversely proportional to the square of the conductance of the Markov chain. Therefore, by changing $\mathbb{P}_3$ to $\mathbb{P}_{3E}$, the mixing time of the Markov chain could be shortened, which would improve the performance of the algorithm under finite number of iterations.

Taking the DTMC in Figure 1(b) as an example, it is easy to check that Figure 1(a) and 1(b) have exactly the same stationary distribution. However, by choosing $\alpha = 0.5$ and fixing the initial state of both chains to be state 3, the distributions of the two chains after 20 iterations are $\vec{\pi}(a, 20) = [0.098, 0.329, 0.571]$ and $\vec{\pi}(b, 20) = [0.096, 0.293, 0.610]$, respectively. It is obvious that $\vec{\pi}(b, 20)$ is closer to $\vec{\pi} = [0.090, 0.244, 0.665]$ than $\vec{\pi}(a, 20)$.

### C. Rotation Operation

Again, let us focus on the Markov chain in Figure 1(a). It is clear that state 3 is a local optimal state since all its neighbors (state 1 in this case) have a lesser utility. Then according to the way the transition probability is chosen, when the initial condition is picked as state 3, it may take a long time before state 5, the optimal state, can be reached. One way to alleviate this problem is to expand the number of states that a single state can jump to in one step. For example, we can introduce an edge between state 3 and state 5, as shown in Figure 1, and pick the transition probability according to Equation (2). While the two chains in Figure 1(a) and 1(c) have exactly the same stationary distribution, the distribution of chain (a) with $\alpha = 0.5$ after 5 iterations is $\vec{\pi}(a, 5) = [0.129, 0.649, 0.221]$, while the one for chain (c) is $\vec{\pi}(c, 5) = [0.090, 0.252, 0.655]$, which is closer to the stationary distribution.

Similarly, in the randomized algorithm, we expand the number of neighboring allocation patterns by introducing another operation called *rotation*. Different from flip or swap operation in which at most two users' allocations could be modified after a single iteration, in the rotation operation, we randomly pick three users $k_A$, $k_B$, and $k_C$ by following Step-1E in Algorithm 2, then, the algorithm decides which operation to carry out in a randomized fashion. If the rotation operation is chosen, each of these three users randomly pick an associated subcarrier and then randomly permute the association of these three subcarriers among the three users. This way, each iteration could at most modify the allocation pattern for three users instead of two, which expand the number of allocation patterns that a state can jump to in a single iteration.

While the rotation operation could be easily extended to involve more than three users as a way to further improve the connectivity of the Markov chain, it comes at a cost of increased computational complexity and longer execution time at each iteration. If we keep involving more and more users in a single iteration, it will reach a point where further improvement in the connectivity no longer justifies the corresponding increase in the complexity at each iteration. Therefore, we restrict our rotation operation to only three users in our algorithm.

### D. Tracking

Finally, we modify Algorithm 1 such that the final allocation decision is the allocation with the largest sum utility for all iterations, instead of simply picking the allocation in the last iteration to be the allocation decision.

**Algorithm 2:** Enhanced Randomized Algorithm

**Data**: $\mathbb{P}_{\text{flip}}, \mathbb{P}_{\text{swap}} \in (0, 1)$ with $\mathbb{P}_{\text{flip}} + \mathbb{P}_{\text{swap}} < 1$.

**Initialization**
- $V^{[0]}(s) = \arg\max_k \{g_{s,k}, 1 \le k \le K\}$ for any $s$;
- $U^o = U_{\max} = \sum_{k=1}^{K} U(k, V^{[0]})$ and $V_{\max} = V^{[0]}$;

**for** *every allocation iteration* $t = 0 \to T - 1$ **do**
- set $\overline{V}^{[t]} = V^{[t]}$;
- **Step-1E [pick a user-triplet $\{k_A, k_B, k_C\}$]**
  - randomly pick $s$ from $\mathcal{S}$ and set $k_A \triangleq V^{[t]}(s)$;
  - randomly pick $k_B$ and $k_C$ from $\mathcal{K} \backslash \{k_A\}$;
- set $p_1 = \begin{cases} 1 & \text{with probability } \mathbb{P}_{\text{flip}} \\ 2 & \text{with probability } \mathbb{P}_{\text{swap}} \\ 3 & \text{otherwise} \end{cases}$;
- **Step-2a [flip operation]** if $p_1 = 1$;
- **Step-2b [swap operation]** if $p_1 = 2$;
- **Step-2E [rotation operation]** if $p_1 = 3$
  - set $(k'_A, k'_B, k'_C)$ to be a random permutation of $\{k_A, k_B, k_C\}$;
  - **if** *all three users have at least one subcarrier* **then**
    - randomly pick $s'_A$, $s'_B$, and $s'_C$ from $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k'_A\right\}$, $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k'_B\right\}$, and $\left\{s \in \mathcal{S} \middle| V^{[t]}(s) = k'_C\right\}$, respectively;
    - set $\overline{V}^{[t]}(s'_A) = k'_B$, $\overline{V}^{[t]}(s'_B) = k'_C$, and $\overline{V}^{[t]}(s'_C) = k'_A$;
- **Step-3 [self-transition or not]**
  - $p_2 = \begin{cases} 1 & \text{w.p. } \mathbb{P}_{3E}\left(\overline{V}^{[t]}; k_A, k_B, k_C, V^{[t]}, \alpha\right) \\ 0 & \text{otherwise} \end{cases}$;
  - set $V^{[t+1]} = \overline{V}^{[t]}$ if $p_2 = 1$;
  - set $V^{[t+1]} = V^{[t]}$ if $p_2 = 0$;
- **if** $p_2 = 1$ **then**
  - $U^o = U^o + \sum_{k=k_A,k_B,k_C} \left(U(k, \overline{V}^{[t]}) - U(k, V^{[t]})\right)$;
  - set $V_{\max} = \overline{V}^{[t]}$ & $U_{\max} = U^o$ if $U^o > U_{\max}$;

**Result**: $V = V_{\max}$.

## V. Joint Uplink-Downlink allocation in full-duplex OFDMA systems

Recent breakthroughs in full-duplex wireless communication [12]–[14] show that it is possible for a wireless device to simultaneously transmit and receive in the same frequency band by using advanced signal processing techniques. If we assume that this full-duplex capability is available on both the base station and all the users in the single-cell OFDMA system we consider, then bidirectional transmissions are feasible for each user on its allocated subcarriers. In other words, we can use a single resource allocation pattern for both uplink and downlink transmission. In this section, we formulate a joint uplink-downlink resource allocation problem for full-duplex OFDMA systems, and point out that our randomized algorithm can be easily modified to find the optimal resource allocation pattern.

Assuming the power budget for the base station is $P_{BS}$ and the rate-utility function for the base station is $U_{BS}(.)$, then the maximal downlink utility under a certain allocation pattern $V^{(x)}$, denoted as $U_D(V^{(x)})$, can be achieved using water-filling solution on all the user-subcarrier pairs in $V^{(x)}$. More precisely,

$$U_D\left(V^{(x)}\right) = U_{BS}\left(\sum_{s=1}^{S}\log\left(1+g_{s,k}p_s\left(V^{(x)}\right)\right)\right),$$

where

$$p_s\left(V^{(x)}\right) = \sum_{k=1}^{K}\mathbf{1}_{\left\{V^{(x)}(s)=k\right\}}\left[\nu - 1/g_{s,k}\right]^+,$$

$$\text{and } \sum_{s=1}^{S} p_s\left(V^{(x)}\right) = P_{BS}.$$

Then, the joint uplink-downlink subcarrier and power allocation problem can be formulated as

$$\max_{1\le x\le K^S}\left(\sum_{k=1}^{K}U\left(k,V^{(x)}\right)+U_D\left(V^{(x)}\right)\right).$$

By changing the transition probability in Equation (2) of the randomized algorithm to be

$$\mathbb{P}_{3F} = \mathbb{P}_{3E}\times\frac{e^{\alpha U_D\left(\overline{V}^{[t]}\right)}}{\max\left\{e^{\alpha U_D\left(\overline{V}^{[t]}\right)}, e^{\alpha U_D\left(V^{[t]}\right)}\right\}},$$

it is easy to show that our randomized algorithm can approach the allocation pattern that achieves the maximal sum-utility of the base station and all its users by following the same prove technique we have used in Proposition 1 and Proposition 2.

## VI. Simulation Result

In this section, we conduct simulations to evaluate the performance of our randomized algorithm and compare the results with that achieved by existing algorithms. In particular, we choose four existing low-complexity sub-optimal uplink allocation algorithms as shown in Table I to compare with. Among the four existing algorithms, MaxCh is the simplest one as it simply allows each subcarrier to choose the user with the largest channel gain to associate with, while all the other three follow a similar greedy structure where one (user, subcarrier) association is chosen at a time until all the subcarriers are allocated. These existing algorithms are developed base on the insight gained from either the necessary condition of optimality of the original resource allocation problem, or the optimal solution for a relaxed version of the integer optimization problem.

We consider a model of the *i.i.d.* Rayleigh block fading channel, where the channel gain for each subcarrier follows an independent Rayleigh distribution across both different users and different time-slots. In other words, at each time-slot, the normalized SNR $\{g_{s,k}\}$ follows an exponential distribution with mean 1 for all $s$ and $k$. In order to account for the difference in path loss across different users, for each simulation setting (each $(S,K)$ pair), we assume that the normalized power budget $P_k$ for different user $k$ is uniformly distributed in $[3S/K, 6S/K]$.

TABLE I
Summary of some existing sub-optimal algorithms

| | Ref. | Computational Complexity | Number of water-fillings | Objective: Maximization of |
|---|---|---|---|---|
| MaxCh | [1] | $O(S)$ | $K$ | Sum-rate |
| MaxRt | [1] | $O(KS^2)$ | $KS$ | Sum-rate |
| NS[1] | [2] | $O(KS\log S)$ | $KS$ | Sum-utility |
| SOA[2] | [3] | $O(KS\log S)$ | $K$ | Sum-weighted-rate |

Since some existing algorithms can only deal with sum-rate maximization, we only focus on the case when $U_k(u) = u$ for all $k$. For each $(S,K)$ pair, we run each algorithm over 100 time-slots and present the average achieved sum-rate in Table II. In Table II, we show the average sum-rate achieved by three versions of the randomized algorithm, RA($\omega$), ERA($\omega$), and SOA+ERA($\omega$), where SOA+ERA($\omega$) denotes the algorithm when the initial allocation pattern of ERA($\omega$) is chosen to be the outcome of the SOA algorithm. For RA algorithm, we set $\mathbb{P}_{\text{flip}} = 1/2$, and for ERA and SOA+ERA algorithm, we set $\mathbb{P}_{\text{flip}} = \mathbb{P}_{\text{swap}} = 1/3$.

Since our algorithm is near-optimal when $\omega$ is large enough, it can be used to benchmark the performance of sub-optimal algorithms. From Table II, we have the following observations:

- With the water-filling budget $w$ set to be $SK$, the enhanced randomized algorithm ERA($w$) always performs better than the original RA($w$) algorithm.
- A larger sum-rate can be achieved by increasing $w$, the number of times the water-filling solution can be invoked, for the randomized algorithms.
- Both ERA($S^2K$) and SOA+ERA($S^2K$) have a consistently better performance than all the other existing sub-optimal algorithms.
- While SOA is outperformed by ERA($S^2K$) in all settings, the performance gap is small, especially under the case when $K < S$, which shows that SOA is a desirable low-complexity algorithm when the number of subcarriers is larger than the number of users. Note, however, that the SOA algorithm is not proven to be optimal and can only deal with the sum-weighted-rate maximization problem, whereas our randomized algorithm can deal with *any utility function of the instantaneous rate*.
- We can combine the sub-optimal algorithm with the randomized algorithm to achieve a better performance. For example, by setting the initial allocation

[1] We use the authors' last-name initials to denote the algorithm in [2].
[2] There are several variations of the SOA algorithm in [3]. Here we pick SOA1-4B5A as it consistently achieves better performance than the other variations.

TABLE II
SUM-RATE MAXIMIZATION, WHERE THE SUM-RATE IS AVERAGED OVER 100 TIME-SLOTS (100 SAMPLES OF THE CHANNEL MATRIX) WITH $\alpha = 10$.

| Average sum-rate | | Existing Algorithms | | | | RA($w$) | ERA($w$) | | | SOA+ERA($w$) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MaxCh | MaxRt | NS | SOA | $w = SK$ | $w = SK$ | $5SK$ | $S^2K$ | $w = SK$ | $5SK$ | $S^2K$ |
| (K = S) | | | | | | | | | | | | |
| 20 | 20 | 58.8 | 66.4 | 65.6 | 67.0 | 63.3 | 66.1 | 68.3 | 69.1 | 67.9 | 68.7 | 69.3 |
| 40 | 40 | 122.6 | 141.6 | 136.7 | 142.6 | 137.0 | 140.4 | 144.5 | 146.9 | 143.6 | 145.4 | 147.2 |
| 60 | 60 | 175.9 | 208.9 | 197.1 | 210.3 | 210.8 | 208.9 | 212.5 | 215.7 | 211.3 | 213.4 | 215.8 |
| 80 | 80 | 237.1 | 285.0 | 264.6 | 287.0 | 276.9 | 284.0 | 289.0 | 292.7 | 288.4 | 290.1 | 292.7 |
| 100 | 100 | 296.7 | 358.6 | 329.3 | 360.7 | 349.9 | 356.5 | 361.9 | 365.7 | 362.0 | 363.3 | 365.7 |
| (K < S) | | | | | | | | | | | | |
| 20 | 40 | 130.1 | 136.3 | 137.3 | 137.7 | 130.7 | 135.4 | 138.6 | 140.2 | 138.3 | 139.1 | 140.0 |
| 20 | 60 | 201.4 | 209.4 | 211.0 | 211.1 | 200.7 | 208.1 | 211.6 | 213.5 | 211.7 | 212.4 | 213.5 |
| 20 | 80 | 269.0 | 276.3 | 278.0 | 277.9 | 265.1 | 273.9 | 277.9 | 280.2 | 278.4 | 279.2 | 280.2 |
| 20 | 100 | 350.6 | 356.8 | 357.8 | 358.0 | 341.0 | 353.7 | 357.7 | 360.1 | 358.5 | 359.0 | 360.2 |

pattern of the enhanced randomized algorithm to be the allocation obtained by SOA, the performance of SOA+ERA($w$) is always better than ERA($w$).

## VII. CONCLUSION

We propose a new iteration-based randomized uplink joint power-subcarrier allocation algorithm for single-cell OFDMA systems that solves the instantaneous sum-utility maximization problem. The randomized algorithm is near-optimal and can trade-off complexity against performance in the sense that a better performance can be obtained by increasing the number of iterations, and the optimal allocation can be found with high probability as the number of iterations approaches infinity. An enhanced version is also presented that can achieve a better performance under finite number of iterations. Further, we show that this randomized algorithm can be easily modified to solve a joint uplink-downlink resource allocation problem in a full-duplex OFDMA system where the bidirectional transmission is feasible for every user-subcarrier pair. Finally, we conduct extensive simulations to compare the performance of our proposed algorithm with some existing low-complexity sub-optimal ones, and show that our algorithm can achieve a consistently better performance than existing ones by increasing the number of iterations. For future work, we plan to generalize this algorithm to solve the utility maximization problem under a multi-cell OFDMA scenario.

## REFERENCES

[1] K. Kim, Y. Han, and S.-L. Kim, "Joint Subcarrier and Power Allocation in Uplink OFDMA Systems," *Communications Letters, IEEE*, vol. 9, no. 6, pp. 526–528, 2005.
[2] C. Y. Ng and C.-W. Sung, "Low Complexity Subcarrier and Power Allocation for Utility Maximization in Uplink OFDMA Systems," *Wireless Communications, IEEE Transactions on*, vol. 7, no. 5, pp. 1667–1675, 2008.
[3] J. Huang, V. Subramanian, R. Agrawal, and R. Berry, "Joint Scheduling and Resource Allocation in Uplink OFDM Systems for Broadband Wireless Access Networks," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 2, pp. 226–234, 2009.
[4] F. Martinelli, *Lectures on Glauber Dynamics for Discrete Spin Models*, ser. Lecture Notes in Mathematics. Berlin, Germany: Springer Berlin Heidelberg, 1999.
[5] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. American Mathematical Soc., 2009.
[6] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," *Networking, IEEE/ACM Transactions on*, vol. 18, no. 3, pp. 960–972, 2010.
[7] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-Length-Based CSMA/CA Algorithms for Achieving Maximum Throughput and Low Delay in Wireless Networks," *Networking, IEEE/ACM Transactions on*, vol. 20, no. 3, pp. 825–836, 2012.
[8] S.-Y. Yun, Y. Yi, J. Shin, and D. Y. Eun, "Optimal CSMA: A survey," in *Communication Systems (ICCS), 2012 IEEE International Conference on*, 2012, pp. 199–204.
[9] S. M. Ross, *Stochastic Processes*, 2nd ed. New York: John Wiley & Son, 1996.
[10] Y. Yang, C. Nam, and N. B. Shroff, "A Near-Optimal Randomized Algorithm for Uplink Resource Allocation in OFDMA Systems," *Technical Report*. [Online]. Available: http://www2.ece.ohio-state.edu/∼yangy/
[11] A. Sinclair, *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Basel, Switzerland, Switzerland: Birkhauser Verlag, 1993.
[12] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha, "Practical, real-time, full duplex wireless," in *ACM MobiCom 2011*, New York, NY, USA, 2011, pp. 301–312.
[13] M. Duarte and A. Sabharwal, "Full-duplex wireless communications using off-the-shelf radios: Feasibility and first results," in *ASILOMAR 2010*, Nov., pp. 1558–1562.
[14] J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving single channel, full duplex wireless communication," in *ACM MobiCom 2010*, New York, NY, USA, 2010, pp. 1–12.