Subcarrier Allocation In OFDMA With Time Varying Channel And Packet Arrivals

Engin Toktas
ASELSAN Inc.
Ankara, TURKEY
E-mail : toktas@aselsan.com.tr

Elif Uysal-Biyikoglu
Middle East Technical University
Ankara, TURKEY
E-Mail : elif@eee.metu.edu.tr

Ali Ozgur Yilmaz
Middle East Technical University
Ankara, TURKEY
E-Mail : aoyilmaz@eee.metu.edu.tr

Abstract—This study considers the design of efficient low-complexity algorithms for dynamic allocation of subcarriers to users in a multiuser transmitter using Orthogonal Frequency Division. The effects upon throughput and delay performances of several canonical algorithms of varying the number of users, the number of subcarriers, and the statistical characteristics of incoming packets are investigated. Consequently, a subcarrier allocation algorithm with low computational complexity and satisfactory performance in almost all cases of interest is developed. The significance of channel v.s. queue state information with respect to various statistical properties of packet arrival processes is explored through extensive simulations. The results carry implications for the amount of channel estimation and feedback necessary. Finally, a new experimental method is proposed and applied for obtaining the stability regions of the algorithms in consideration.

I. INTRODUCTION

An ever-increasing number of mobile communication customers in recent years have made wireless Metropolitan Area Network (MAN) technologies a topic of interest. Orthogonal Frequency Division Multiple Access (OFDMA) is one of the popular MAN technologies as it is part of the IEEE 802.16a and IEEE 802.16e standards.

The motivation of our study is the need for efficiently multiplexing a number of independent data streams (typically belonging to different users) in OFDMA. This is a typical dynamic resource allocation problem with the twist that at any time, the mapping between users and subcarriers can one-to-many, depending on the channel state seen by the users at their frequency bands, and their queue backlog.

More specifically, we consider an OFDMA system with one base station, N users and K subcarriers. Data packets arrive stochastically to each user and are stored in queues prior to transmission. Because of the limited number of subcarriers, the system must make a choice of assignment of subcarriers to users. This gives rise to a scheduling problem involving the allocation of the orthogonal channels to different data streams.

To the best of our knowledge, joint treatment of throughput, delay and complexity with respect to subcarrier allocation algorithms is rare in previous studies. Achieving ergodic capacity under a power constraint by multi-user waterfilling at each decision epoch has been addressed in [1], [2], [3], [4]. Of course, infinite buffers notwithstanding, this cannot go beyond maximizing the “instantaneous” throughput of the system [10]. When considering stochastic packet arrivals, waterfilling with respect to channel only is not throughput optimal [8]. Throughput-maximizing algorithms (based on Maximum Weight Matching) [5], [8], [6] have been presented. Another important performance measure, packet delay, has been addressed under less general conditions, leading to results that cannot be easily extended to other scenarios. For example, [7] proposes a subcarrier allocation method when the base station has only one subcarrier. Moreover, [6] studies a system which has an ON/OFF channel model, that is, the connectivity value of an ON/OFF channel can be either 1 or 0.

Analysis of long-term average throughput, and especially long-term average delay, is quite intractable in the system model under consideration. Yet, it is important to gain insights toward the design of low complexity algorithms with strong performance, under various conditions. The main contributions of this paper will be such insights that result from detailed and extensive simulations that aim to isolate a set of conditions under each of which a set of algorithmic approaches are suitable, and to offer explanations as to why. For example, under unbalanced load conditions, a simple algorithm (which we call LE) that selects longer queues without searching among all combined queue and channel states, does very well, more precisely, touches the boundary of the capacity region in the limit as demand gets more unbalanced. On the other hand, in the balanced case, when user rate demands are similar, another simple algorithm (which we call MIT) that simply selects the best states (without doing a joint search with queue state) is near the capacity region boundary. Especially when the number of users is large, being able to restrict the search to a smaller set as in the above examples, without sacrificing throughput or delay, is very valuable in practice [9]. Next, we will describe the system model, following which the subcarrier allocation algorithms and related results will be presented in detail.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

We make quite standard assumptions about the system in conformity with related literature: a single-hop OFDMA downlink composed of one base station and N users (queues) is considered. Buffer capacity is assumed infinite (i.e., there will be no packet drops.) Fixed-sized packets destined to different
The basic block diagram of the system is given in Figure 1.

The packet arrival processes \( a_i(t) \) form homogenous Poisson process with rate \( \lambda_i (i = 1, 2, ..., N) \).

The following notations will be used throughout the paper: subscript \( i \) denotes specific subcarrier/server, subscript \( j \) denotes specific user/queue. We use lowercase boldfaced letters without any subscription for vectors and capitalized boldfaced letters for matrices.

- \( \mathbf{a}(n) = (a_1(n), a_2(n), ..., a_N(n)) \), where \( a_i(n) \) is the number of packet arrivals to queue \( j \) during the \( n \)th time slot. (Arrivals during slot \( n \) can be served only in slot \( n + 1 \) or thereafter.)
- \( \mathbf{b}(n) = (b_1(n), b_2(n), ..., b_N(n)) \) is the vector of queue occupancies at the beginning of timeslot \( n \). In other words, \( b_j(n) \) is the number of packets in User-\( j \)'s buffer at the beginning of slot \( n \).
- \( \mathbf{H}(n) = \{ h_{ij}(n) \} \) is the K-by-N channel gain matrix at time \( n \) where \( h_{ij}(n) \) represents the channel gain for subcarrier \( i \) and user \( j \), which is Rayleigh distributed when \( \sigma^2 \) is equal to 1.
- \( \mathbf{C}(n) = \{ c_{ij}(n) \} \) is the K-by-N connectivity matrix at time \( n \) where \( c_{ij}(n) \) denotes the maximum number of packets that subcarrier \( i \) can serve from queue \( j \) in timeslot \( n \). The mapping from \( H(n) \) to \( C(n) \) assumes the ability to use adaptive modulation and coding, and is illustrated in Figure 2. In our model, \( c_{ij}(n) \in \{0, 1, 2, 3, 4\} \).
- \( \mathbf{W}(n) = \{ w_{ij}(n) \} \) is the K-by-N allocation matrix at time \( n \) where \( w_{ij}(n) \) is equal to 1 if subcarrier \( i \) is assigned to queue \( j \), and 0 otherwise.
- \( \mathbf{W}(\mathbf{b}(n), \mathbf{C}(n)) \) is the set of all non-idling allocation matrices at time \( n \).

III. Subcarrier Allocation Algorithms

Within our scope, the three axes of the tradeoff space are: time average throughput, time average packet delay, and computational complexity.

It is well known [12], [6] that maximizing instantaneous throughput -via, for example, waterfilling with respect to channel state in subcarriers- is sub-optimal with respect to long-term throughput. That is, by selecting at every scheduling interval a set of users that achieve maximum total rate, one cannot stabilize the system for all stabilizable input rate vectors. According to the notation of this paper, we define throughput-optimality as the following.

**Definition 3.1:** Define \( \mathbf{W}(\mathbf{b}, \mathbf{C}) \) is the set of all non-idling feasible allocation matrices. A throughput-maximizing policy \( \pi \) is a policy which chooses \( W^* \) among \( \forall \mathbf{W} \in \mathbf{W}(\mathbf{b}, \mathbf{C}) \) to maximize \( \lim_{T \to \infty} \frac{1}{T} \mathbb{E}[L_T] \), where

\[
L_T = \sum_{n=0}^{T} \left[ \sum_{i=1}^{K} \sum_{j=1}^{N} c_{ij}(n)w_{ij}(n) \right]
\]

From Little’s Theorem [13], one can get a handle on average delay using average backlog (queue size). Hence, we have the following.

**Definition 3.2:** Define \( \mathbf{W}(\mathbf{b}, \mathbf{C}) \) as the set of all non-idling feasible allocation matrices. System average delay minimization policy \( \pi \) is the subcarrier allocation policy which chooses \( W^* \) among \( \forall \mathbf{W} \in \mathbf{W}(\mathbf{b}, \mathbf{C}) \) such that it minimizes the cost function \( C_T^\pi \) at the finite horizon \( T \):

\[
C_T^\pi = \sum_{n=0}^{T} \sum_{j=1}^{N} b_j(n)
\]

It has been shown [8] that an adaptation of Tassiulas and Ephremides’s [5] Maximum Weight Matching (in the rest, MaxWeight) to this problem is throughput-optimal. However, implementing MaxWeight can be impractical in a wireless OFDMA system because of complexity as well as the difficulty of making physical-layer modulation and coding decisions informed by buffer state. It is of interest to design simpler schedulers that can perform satisfactorily close to optimal in terms of delay and throughput. In order to understand how to obtain such simple schedulers, we now study some algorithmic paradigms.
A. Maximum Instantaneous Throughput Algorithm (MIT)

Algorithm MIT essentially allocates the subcarrier $i$ to the user $j$ with the largest $c_{ij}$ without considering the backlog in the queues. The scheduling decision at each time slot is as follows:

- for $i=1$ to $K$;
  - choose $j' = \arg\max_{j \in J} c_{ij}$, ($j' \in J$ if $b_{j'} \geq c_{ij}$),
  - if there exists more than one $j'$ which has the same value of $c_{ij}$, then choose one of them randomly,
  - $w_{ij'} = 1$ elsewhere $w_{ij} = 0$,
  - $b_{j'} = b_{j'} - c_{ij'}$
- end.

Checking all $c_{ij}$s should be checked to find all $j'$s requires $KxN$ operations, resulting in a complexity of $O(KN)$. 

B. Load Equalizing Algorithm (LE)

The principle of LE is to distribute the workload among queues as evenly as possible in order to minimize future server idling without considering channel and queue state jointly. Under some conditions [7], [6] LE approaches throughput-optimality. However, in general LE sacrifices the current throughput (by giving priority to longer queues) for the future throughput.

In order to obtain a less complex algorithm, we modify LE as follows:

- for $i=1$ to $K$;
  - choose $j' = \arg\max_{j \in J} (b_{j})$, ($j' \in J$ if $b_{j'} \geq c_{ij}$ and $c_{ij} \neq 0$),
  - if there exists more than one $j'$ which has the same value of $(b_{j})$, then choose one of them randomly,
  - $w_{ij'} = 1$ elsewhere $w_{ij} = 0$,
  - $b_{j'} = b_{j'} - c_{ij'}$,
- end.

The complexity of LE is calculated similarly as that of MIT and is also equal to $O(KN)$, however it is in fact simpler since it needs to look only at queue state.

C. Maximum Weight Matching (MWM)

MWM [5] is throughput-optimal in our setting [8], [14]. However, no guarantees are offered on delay performance. The complexity is $O(KN)$. The scheduling decision according to MWM is [5]:

- $TX = \{TX_1, TX_2, ..., TX_N\} = 0$;
  - for $i=1$ to $K$;
    - choose $j' = \arg\max_{j} (b_{j}c_{ij})$,
    - if there exists more than one $j'$ which has the same value of $(b_{j}c_{ij})$, then choose one of them randomly,
    - $TX_{j'} = TX_{j'} + c_{ij'}$,
    - $w_{ij'} = 1$ elsewhere $w_{ij} = 0$,
  - end
  - for $i=1$ to $K$;
    - if $TX_{j'(i)} \geq b_{j'(i)}$, then $w_{ij} = 0$,
  - end.

The complexity of LE is calculated similarly as that of MIT and is also equal to $O(KN)$, however it is in fact simpler since it needs to look only at queue state.

D. Recursive Max. Weight Matching Algorithm (r-MWM)

In MWM, the same backlog information $(b(n))$ is used during the whole subcarrier assignment decisions at time $n$, without updating after any subcarrier assignment. This may cause imbalanced queues at the end of the assignment. [10] proposes an algorithm which uses the decision mechanism given below in order to avoid imbalanced queues. Algorithm r-MWM has a worst case complexity of $O(NK^2)$.

- $X = \{1, 2, ..., K\}$;
- Loop (until STOP)
  - if $X = \emptyset$ then STOP;
  - $(i', j') = \arg\max_{i \in I, j \in J} (b_{i}c_{ij})$;
  - if $b_{i'}c_{i'j'} > 0$, then $w_{i'j'} = 1$ else STOP;
  - $b_{j'} = b_{j'} - c_{i'j'}$ and $X = X - \{i'\}$;
- Assign $W^* = \{w_{ij}^*\}$.

E. Updated Max. Weight Matching Algorithm (u-MWM)

Considering system average throughput, delay and complexity, we propose an algorithm which has a maximum weight matching characteristic, with $O(NK)$ complexity, and further takes into account avoiding imbalanced queues as r-MWM in order to improve the delay performance. Our proposed algorithm is called the “Updated” Maximum Weight Matching, because after each subcarrier assignment, it updates the queue lengths without increasing the complexity. The complexity is again $O(KN)$. Here is the scheduling decision:

- for $i=1$ to $K$;
  - choose $j' = \arg\max_{j} (b_{j}c_{ij})$, ($j' \in J$ if $b_{j'} \geq c_{ij}$),
  - if there exists more than one $j'$ which has the same value of $(b_{j})$, then choose one of them randomly,
  - $w_{ij'} = 1$ elsewhere $w_{ij} = 0$,
  - $b_{j'} = b_{j'} - c_{ij'}$,
- end.

IV. SIMULATIONS AND RESULTS

Parameters of interest will include the number of users, number of subcarriers, arrival rates, etc. In order to make a fair comparison between algorithms, we must be careful about the interaction between these parameters. For example, increasing the number of users without changing the number of subcarriers leads to increased network capacity because of “multi-user diversity”. In this case, for instance, light/heavy traffic definition should be modified each time according to the number of users.

In addition to expressing all our throughput and delay results in terms of “packets/slot/subcarrier”, we also use a “load” term as the ratio between packet input traffic density and network capacity.

**Definition 4.1:** The load, $L$, is the ratio between the system average input packet rate and the network capacity

$$L = \frac{\sum_{j=1}^{N} E[a_{j}]}{KYN}.$$  (3)
where \( E[a_j] \) represents average incoming packet rate (in packets/slot) of user-\( j \) and \( \gamma_N \) represents average output capacity of one subcarrier when there exists \( N \) users.

We now present our network capacity definition, which captures the maximum expected service rate when all queues are non-empty:

**Definition 4.2:**

\[
\gamma_N = E\{\max\{c_1, c_2, ..., c_N\}\},
\]

where \( c_j \) is the connectivity matrix element for a specific subcarrier of user \( j \) and has an occurrence probability given at the caption of Figure 2.

In the literature, most of the studies such as [11], [6], [7], assume that the users are homogenous, i.e. they have statistically identical arrival processes. This means that all arrival rates, \( a_j \in \{1, 2, ..., N\} \), are equal to each other. In real life applications, it is not hard to imagine that user rate demands can be very unbalanced. For example, while several users are interested in HTML pages and others may want to watch a video broadcast at the same time. We therefore find it important to study unbalanced situations.

**Definition 4.3:** A “balance ratio" vector \( \mathbf{r} = [r_1, r_2, ..., r_N] \) is a 1x\( N \) vector such that each element \( r_j \) represents the ratio between the incoming packet rate of user \( j \) and the total incoming packet rate of the system.

\[
r_j = \frac{E[a_j]}{\sum_{i=1}^{N} E[a_i]}, \forall j \in \{1, 2, ..., N\}.
\]

Note that the sum of all balance ratios is equal to 1:

\[
\sum_{j=1}^{N} r_j = 1.
\]

In our simulations, the number of users (\( N \)), the number of subcarriers (\( K \)), load (\( L \)) and balance ratio (\( r \)) are all parametric. All simulations span 10,000 timeslots.

Simulation results clearly illustrate the relative strengths of MIT and LE algorithms. MIT aims to maximize the instantaneous throughput, and works quite well in the balanced arrival regime. However, LE is a better choice as the system’s unbalanced characteristic increases. Of course, MWM-type algorithms (MWM, r-MWM, u-MWM), which exploit both, always perform well in terms of throughput. They use both queue state information and channel state information.

In all different balance ratio cases, the delay performance of u-MWM is almost the same or exactly the same as the delay
performance of r-MWM which has the best delay performance in all cases. This shows that, there is no need to use a high order complex algorithm to obtain a better average delay. u-MWM algorithm can also do this with low order complexity in addition to keep having the highest throughhput performance capability. We have also simulated changing the number of users and the number of subcarriers cases differently and we have observed the same results, that is, the delay performance of u-MWM is almost the same or exactly the same as the delay performance of r-MWM which has the best delay performance in all cases. When simulating changing the number of users and the number of subcarriers cases while keeping the balance ratio constant \((r_i = r_j, \forall i, j = 1, ..., N)\), we have observed that the simulation results are very similar as the results given in Figure 3.

A. Checking Stability

Each subcarrier allocation algorithms has its own stability region ("network capacity region") and these regions give us valuable information about the algorithm's throughput performance. [5] shows that the stability region of a throughput optimal policy contains the stability region of any policy. In the literature, there are some studies such as [15], [16], [17] which are interested in identifying the network system whether it is stable or not via some simulations. With the help of these studies, we develop our own method which is simpler and more applicable to our OFDMA system model:

Step 1: Define the simulation length, \(S\) in slot time unit. The longer the simulation length, the more accurate the result is. In our simulations, \(S\) is equal to 10,000 slots.

Step 2: Define the batch number, \(B\). This number defines the number of subgroups of the simulation slots which are used for obtaining some statistical results. In our simulations, the batch number \(B\) is set to 20.

Step 3: Run the simulation with the investigated algorithm and investigated simulation parameters \((N,K,r)\) up to simulation slot time \(S\). At each slot time, log the each user's queue lengths. \((b(n) = (b_1(n), b_2(n), ..., b_x(n)), n \in \{1, 2, ..., S\}\)

Step 4: After the end of the simulation, calculate the average number of queue lengths for each user per each batch in pkt/subcarrier unit.

\[
\frac{1}{\sum_{n=x(r-1)+1}^{x(r)} b_j(n)} \sum_{n=x(r-1)+1}^{x(r)} b_j(n), \ j \in \{1, 2, ..., N\}, x \in \{1, 2, ..., B\}
\]

(7)

Step 5: Calculate the overall average number of queue lengths per subcarrier for each user. The first batch values are not included to calculations because reaching the steady state from transient state occurs in Batch 1 and the transient state information can fake the result.

\[
b_{j,\text{avg}} = \frac{1}{B} \sum_{x=1}^{B} b_{j,x}, \ j \in \{1, 2, ..., N\}
\]

(8)

Step 6: Per each user, find how many \(b_{j,x} (x \in \{2, 3, ..., B\})\) values are outside the \((b_{j,\text{avg}}-c_{\text{cap}}/2, b_{j,\text{avg}}+c_{\text{cap}}/2)\) region where \(c_{\text{cap}}\) represents the max. connectivity value. In our model, the maximum connectivity value is 4 (\(c_{\text{cap}} = 4\)).

Step 7: For any user, if the number of \(b_{j,x} (x \in \{2, 3, ..., B\})\) outside the region given in Step-6 is equal to or greater than \(B/2\), then the system is called “unstable”, otherwise the system is called “stable”.

Figure 7 which is obtained by using our proposed stability checking method per each different balance ratio values shows the stability regions of our interested algorithms when \(N=2\) and \(K=32\). This result can be compared with the Simulation results given in Figure 3, 4, 5 and can be seen that the stability boundary points in Figure 7, completely matches with the queue blow up points showing in Figure 3, 4, 5.

As seen in Figure 7, all three MWM algorithms have the largest stability region. This is an expected result because in [5],[8],[14] it is proved that the MWM type algorithms are throughput optimal and have the largest stability regions. The interesting results in the graph are the stability characteristics of MIT and LE algorithms. When there exists a balance i.e. \(r=[0.5,0.5]\), MIT which uses channel state information performs very well. On the other hand, LE which uses queue state information has a poor performance in terms of throughput in balanced case. However, when the system’s unbalanced characteristic increases, LE starts to become better than MIT algorithm.

V. Conclusion

This study aimed at gaining insight into low-complexity user-subcarrier mapping in OFDMA. We have seen that when high and low-rate users co-exist in the system, there is a
simple scheduler that essentially disregards channel state and schedules according to queue state, and balances queues. This algorithm, LE, performs well with respect to long term throughput and average delay under the unbalanced case. Similarly, in the balanced case, ignoring queue state does not seem to cost much in terms of throughput or delay. When changing the balance ratio in order to increase the “unbalanced” characteristic of the system, we have seen that the throughput performance of MIT decreases more rapidly than MWM-type algorithms, while the throughput performance of LE almost remains constant. Moreover, beyond some balance ratio, LE starts to perform better than MIT in terms of average throughput. In summary, with balanced arrivals, the channel state information is more critical; on the other hand, in unbalanced case, the queue state information and balancing the queues become more critical.

When considering the delay issues, simulations show us that r-MWM has the minimum system average delay under all simulated scenarios. However the interesting point is that, our proposed algorithm u-MWM also reaches almost or exactly the same delay performance of r-MWM with a lower computational complexity. So, u-MWM algorithm becomes a unique algorithm which meets the three criteria (maximum throughput, minimum delay, low complexity) at the same time.

We also presented an algorithm for obtaining the approximate stability region. By using this method, we have also obtained the stability regions of subcarrier allocation algorithms for N=2 users. Note that, this method can also be directly applied for N>2.

REFERENCES