Scheduling Algorithms for Single-Carrier and Multi-Carrier Wireless Data Systems

(Invited Paper)

Matthew Andrews
Bell Labs, Murray Hill NJ
Email: andrews@research.bell-labs.com

Lisa Zhang
Bell Labs, Murray Hill NJ
Email: ylz@research.bell-labs.com

***Abstract*— In this article we consider a set of scheduling problems for wireless data systems, including the single-carrier model motivated by CDMA technologies and the multi-carrier model motivated by OFDM technologies. We survey a number of single-carrier scheduling algorithms together with the assumptions on traffic patterns and channel conditions under which the algorithms perform well. We also present recent work that adapts some of the single-carrier algorithms to the multi-carrier environment. The challenge in adapting these algorithms partially lies in the NP-hard nature of handling resource wastage, which happens when multiple carriers collectively overallocate resources to the same user. We present solutions based on the greedy approach for maximizing submodular functions over a matroid. We conclude with several open issues in multi-carrier scheduling.**

I. Introduction

In this article we survey a set of problems related to scheduling in wireless data systems. The article is divided into two parts. In the first part we consider a *single-carrier* wireless data system in which scheduling decisions can be made on a timeslot-by-timeslot basis. Such a model is appropriate for studying wireless data systems that are based on a CDMA physical layer. We survey a number of different models and show how the assumptions on traffic patterns and channel conditions affect what performance can be achieved and what algorithms are most suitable for achieving high performance.

We then consider some more recent work that is motivated by wireless data systems that are based on an OFDM physical layer. The major difference is that we now have to schedule multiple *carriers* in each timeslot. Another difference in these systems is that in order to minimize the amount of communication overhead to implement the protocol we sometimes have to schedule data in fixed *templates* consisting of multiple timeslots. Adapting single-carrier scheduling algorithms to future multi-carrier OFDM systems raises a number of issues that we discuss in this work.

Throughout this paper we consider a single-cell model in which a basestation needs to exchange data with multiple mobile users. (See Figure 1.) We focus on the downlink (basestation-to-mobile) direction although some of our results can also be applied to the uplink direction. The basic goal of a scheduler is to determine which mobile user(s) should receive data in each time step. We would like to do this in such a way that total throughput is high, as much of the arriving data as possible is transmitted, and fairness is maintained to the largest extent possible.

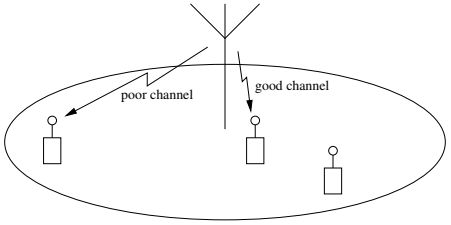


Fig. 1. A cellular wireless system.

A. Single-carrier wireless data scheduling

The main feature of a single-carrier data system is that the basestation transmits to only one user in each timeslot. This is an appropriate model for a CDMA-based data system such as EV-DO where the resources are time-shared among the multiple users. We assume that time is divided into slots. The channel condition between the basestation and a user is *user-dependent* and *time-varying*. For each user i and for each timeslot t, we let $r_i(t)$ represent the *channel conditions* on the link between the basestation and user i at time t. That is, $r_i(t)$ represents the amount of data that can be transmitted to user i if it is selected for service. (Note that although channel conditions are ultimately a physical quantity such as signal-to-noise ratio, it makes sense to think of them as transmission rates assuming that modulation and coding schemes are fixed. We make this assumption throughout this paper and therefore use the terms channel condition and channel rate interchangeably.)

At time t the job of the scheduler is to pick one user to be serviced. If user i is chosen then data of total size $r_i(t)$ may be transmitted from the basestation to user i. We set $x_i(t) = 1$ if timeslot t is assigned to user i and $x_i(t) = 0$ otherwise. Therefore, for each user the total amount of data that it receives in timeslot t is $r_i(t)x_i(t)$.

The scheduling algorithm that is most appropriate for a given situation is determined by the type of channel conditions that we expect to encounter and the type of traffic that we must serve. We consider two types of channel conditions, those governed by a stationary stochastic process and those

governed by an adversarial process. We say that the traffic conditions are of the former type if there is some stationary stochastic process $m(t)$ (e.g. an ergodic Markov chain) such that at time t all of the channel conditions are a function of $m(t)$.

A large amount of the theory of wireless scheduling assumes that channel conditions are of the above form. However, in some contexts such a model is inappropriate. For this reason it sometimes makes sense to consider channel conditions governed by some *adversarial* process that allows for more general inputs. For example, in many situations a mobile user could be moving away from or towards the basestation which would mean that the channel conditions would have an overall drift. In Section II-C we discuss how scheduling can be adapted for such an adversarial model.

We consider three types of traffic patterns. Two of them are defined by external arrival processes where each user has a finite backlog of data to serve, and the third type has infinitely backlogged queues for all users at all times. For external arrivals we let $a_i(t)$ be the amount of data that arrives in the queue for user i at time t. We also let $q_i(t)$ be the queue size of user i which is updated according to $q_i(t+1) = \max\{0, q_i(t) + a_i(t) - r_i(t)x_i(t)\}$. The value of $a_i(t)$ can either be governed by some stationary stochastic process or by a more general adversary. For external arrivals, a typical goal is *stability*, i.e., we wish to keep the size of the queues bounded whenever possible. For infinitely backlogged queues, a typical goal is to optimize some function of the long-term transmission rates to the users. We give an example of such a function in Section II-A.

B. Multi-carrier wireless scheduling

As already mentioned, the single-carrier model described above is appropriate for modeling third generation CDMA-based wireless data systems such as EV-DO. However, most of the proposed fourth generation systems such as the Long-Term Evolution (LTE) of UMTS, the Ultra Mobile Broadband system (UMB) and WIMAX are expected to utilize an OFDM physical layer in which data is transmitted on multiple OFDM carriers. The exact nature of an OFDM carrier will not concern us in this work but we can think of it as a "slice of bandwidth" that hops over some or all of the available spectrum. At each time step, each of these carriers may be assigned to a different user. We therefore have a *two-dimensional* scheduling problem. We need to schedule over both time and frequency.

Formally we model the multi-carrier scheduling problem as follows. Once again time is divided into slots. We use $r_{i,c}(t)$ to denote the amount of data that user i could receive on carrier c at time t, if carrier c is assigned to user i during that timeslot. We set $x_{i,c}(t) = 1$ if that occurs and $x_{i,c}(t) = 0$ otherwise. Hence the total amount of service that user i receives in timeslot t is given by $R_i(t) = \sum_c r_{i,c}(t)x_{i,c}(t)$. The goal of a scheduler is to decide which user should be served by carrier c in each time slot. Each carrier can be assigned to at most one user but each user can potentially receive service from multiple carriers.

As in the single-carrier model, different results apply depending on whether we assume that the conditions are governed by a stationary stochastic process or are governed by some more general worst-case process. An important question when designing a multi-carrier scheduler is the extent to which channel condition depends on the carrier. That is, can it be the case that the $r_{i,c}(t)$ values are very different for different values of c, or should we assume that these values are essentially the same? If the former case holds then "frequency-selective scheduling" makes sense, i.e. there is a benefit to paying attention to the different rates on each carrier when making a scheduling decision. If the latter case holds then we only need to worry about *how many* carriers are assigned to each user, not *which* carriers. Which case is more representative depends to a large extent on whether a carrier only hops over small part of the available spectrum or whether it hops over the entire spectrum. In the former case frequency-selective scheduling is beneficial. In the latter case it is not. Throughout this work we make the assumption that channel rates can be different on different on carriers and so frequency-selective scheduling can provide gains.

There are two main problems that need to be addressed in the multi-carrier setting. First, if we have some objective that we wish to meet in each time step, can we design a carrier assignment algorithm that meets that objective? In particular, we would like to maximize some weighted function of the service assigned to users, subject to the constraint that we do not overallocate more service to a user than it can actually use. Second, in some multi-carrier systems there is not enough bandwidth available to signal the entire schedule in each timeslot without significantly compromising efficiency. One solution in this case is to schedule using *templates* of multiple timeslots. Each template consists of a schedule for multiple timeslots in succession. The template can then be repeated for multiple times without requiring us to signal the schedule to the receivers in every slot.

The template creation problem can be divided into two parts. First, we decide *how many* timeslots on carrier c are assigned to user i during the template. This problem can be solved in essentially the same way as the regular one-slot multi-carrier scheduling problem since we can just treat the carriers on subsequent timeslots as new carriers. Second, we decide *which* timeslots on carrier c are assigned to user i. The objective is to serve each user as "smoothly" as possible so as to minimize the delay and jitter that the user experiences.

II. Single-carrier Scheduling

The single-carrier scheduling problem is defined on a time-slotted system with one carrier and n users. During each time slot t, the scheduler decides which user to service. The chosen user receives the entire resource during the slot.

As an example of the timescales and channel rates that we typically work with, the following describes the Qualcomm High Data Rate (HDR) [8], [13] system for high speed data

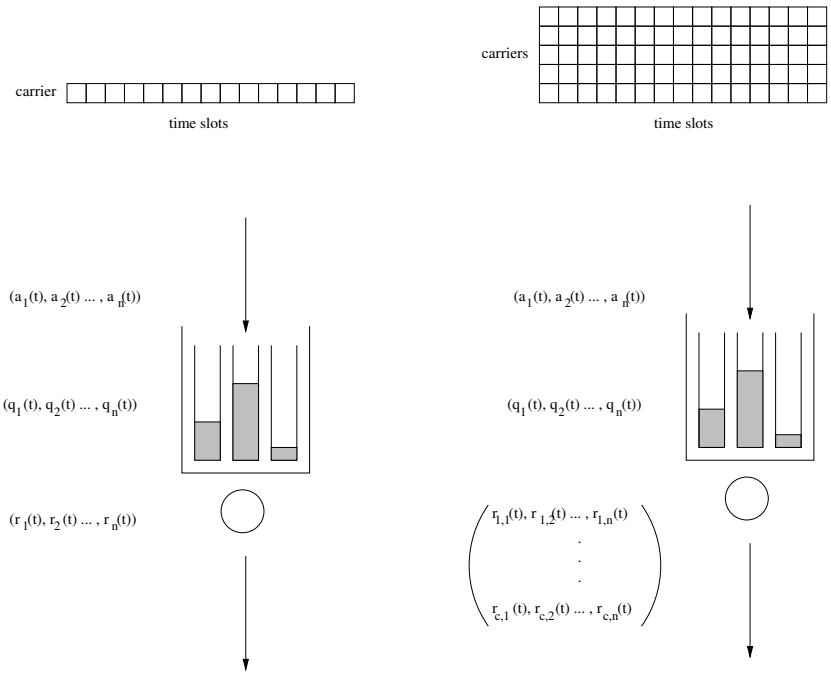


Fig. 2. Single-carrier vs multi-carrier scheduling.

transmission that formed the basis for EV-DO. In HDR one second is partitioned into 600 time slots, i.e. 1.667ms per slot. Each mobile monitors the quality of a pilot signal transmitted by the basestation and estimates the channel quality. From this estimation mobile i decides on the rate $r_i(t)$ at which it can receive data in the next time slot. The mobile then transmits this information to the basestation in a *Data Rate Control* message (DRC). From these messages the basestation has the complete rate vector. HDR uses the following finite set of rates, the left column is in kbits per second and the right in bits per slot.

kb/s	b/slot
0	0
38.4	64
76.8	128
153.6	256
307.2	512
614.4	1024
921.6	1536
1228.8	2048
1843.2	3072
2457.6	4096

In the following we survey three scheduling algorithms together with their properties and the models under which the properties hold.

A. Proportional Fair

The standard scheduling algorithm used by EV-DO is known as PROPORTIONAL FAIR. This algorithm always chooses the user i that maximizes $r_i(t)/\mu_i(t)$ where $\mu_i(t)$ is an exponentially smoothed average of the channel rate to user i up to time t. More precisely,

$$\mu_i(t+1) = (1 - 1/\tau)\mu_i(t) + r_i(t)x_i(t)/\tau,$$

where τ is a "time constant" for the averaging. In the HDR system the value of τ is typically 1024 and so the averaging is taken over roughly 1.7 seconds. The rationale for PROPORTIONAL FAIR is that if all users have large backlogs then the scheme maximizes $\sum_i \log \mu_i(t)$ [18], [1], [14], [15]. (This is sometimes known as the *proportional fair criteria*). This implies that increasing $\mu_i(t)$ by some factor has the same effect on the objective as increasing $\mu_j(t)$ by the same factor.

Theorem 1: ([18], [1], [14], [15]) When the queues are permanently backlogged and the channel rates are governed by a stationary stochastic process, PROPORTIONAL FAIR maximizes the utility function of $\sum_i \log \mu_i(t)$ at each time step t.

B. Max Weight

MAX WEIGHT is another well known algorithm. For each time slot t, MAX WEIGHT always chooses the user i that maximizes $r_i(t)q_i(t)$. The algorithm is shown to have the following desirable stability properties [16], [17]. The proof relies on showing that if the queue sizes are large then MAX WEIGHT creates a negative drift in the Lyapunov function $\sum_i (q_i^s(t))^2$.

Theorem 2: ([16], [17]) MAX WEIGHT is stable for arrival rates and channel rates that are governed by stationary stochastic processes.

C. Tracking

As motivated in the Introduction, channel conditions governed by stationary processes do not appropriately model some common scenarios. Here we consider a more inclusive *worst case model*, in which the channel rates and the arrival rates are generated by an adversary. By "worst case" we mean that the adversary is attempting to cause the chosen scheduling algorithm as many problems as possible.

In order for the problem to be interesting we require that the system is not overloaded. More formally, we say that a sequence of channel rates and arrivals is *admissible* if there is some schedule that can serve all the data. That is, there exist integers $x_i(t) \in \{0, 1\}$ (which we call the *adversary's schedule*) such that for some window w and some loading parameter ε,

$$\sum_{t=\tau}^{\tau+w-1} a_i(t) \leq \sum_{t=\tau}^{\tau+w-1} (1-\varepsilon) x_i(t) r_i(t) \qquad \forall i, \tau, \quad (1)$$

$$\sum_i x_i(t) = 1 \qquad \forall t \quad (2)$$

In other words, in any window of w time steps the adversary is able to schedule the users so that each user receives more service than the amount of data injected.

In [4] and [5] a family of *tracking algorithms* were introduced. The key aim of the tracking algorithms is to follow the adversary's schedule as closely as possible whilst still being realizable. The adversary's *past* schedule during each window of w steps can be formulated via an integer linear program (ILP). The solution to such an ILP specifies a set of $x_i(t)$ that satisfy the admissibility condition for each window in the past. Unfortunately, obtaining integral $x_i(t)$'s involves solving an NP-hard assignment problem. We circumvent the problem of NP-hardness by only calculating a fractional schedule for the adversary rather than an integral schedule.

Once we have calculated the fractional solution we use it to update a set of *counters* which measure how far away the tracking algorithm is from the adversary's schedule. Then, during the next window the tracking algorithm uses these counters to decide which user it is actually going to serve. There are a number of versions of the tracking algorithm that differ in how the counters are defined and used. For example, in the simplest LINEAR TRACKING algorithm, the counter is defined for each user and channel rate pair (i, r). The intuitive meaning of the counter is that it measures the amount of service that the adversary has given user i when $r_i(t) = r$ minus the amount of service that the LINEAR TRACKING has given user i when $r_i(t) = r$. At time t LINEAR TRACKING serves user i at rate $r_i(t)$ if the counter defined on i and $r_i(t)$ is the maximum. We unfortunately do not have any stability results for LINEAR TRACKING. However, in [4] we define a more complex QUADRATIC TRACKING algorithm where the counters are defined for every triple consisting of a user and a pair of channel rates. This algorithm has the following desirable stability and queue properties.

Theorem 3: If the channel rate set is finite, QUADRATIC TRACKING is stable for any (w, ε)-admissible system. This includes critical systems in which $\varepsilon = 0$.

D. Models vs Algorithm Performance

We have surveyed three scheduling algorithms for the single-carrier problem, each of which has desirable properties under certain assumptions on the traffic patterns and channel conditions. In the following we illustrate through two examples that the model assumptions critically affect the performance that can be achieved by a scheduling algorithm.

Recall that PROPORTIONAL FAIR maximizes a utility function assuming infinitely backlogged queues and stationary channel rates. In reality, queues may drain. It is possible to define multiple versions of PROPORTIONAL FAIR depending on how they handle users with small queues. For example, in some variant user i may not be eligible for service unless $q_i(t) \geq r_i(t)$. In other variants, all users are eligible. However, PROPORTIONAL FAIR may choose to serve user i using the value $\min\{q_i(t), r_i(t)\}/\mu_i(t)$ instead of $r_i(t)/\mu_i(t)$, and update $\mu_i(t+1)$ accordingly. In [2] the following instability result of PROPORTIONAL FAIR is shown.

Theorem 4: All versions of PROPORTIONAL FAIR can be unstable against a benign adversary which only injects data constant arrival rates and which only uses 2 rate vectors that appear in a periodic fashion.

Recall also that under stationary processes MAX WEIGHT maintains system stability. In addition, the stability proof implies that the algorithm maintains polynomial queue sizes in the sense that the expected value of $Q(t) = \sum_i (q_i(t))^2$ has a negative drift whenever $Q(t)$ is larger than a certain value that is polynomial in the number of users [16], [17]. On the contrary, MAX WEIGHT can give poor queue performance when the channel process is nonstationary [5].

Theorem 5: When arrival rates and channel rates are governed by an adversary, the queue size under MAX WEIGHT can be exponential in the number of users.

The proof of the above theorem in [5] relies on a somewhat artificial example. However, the same paper also illustrates that queues can become large even in more realistic scenarios. In one example a stream of evenly spaced cars move at a constant speed along the highway with evenly spaced basestations. As each car moves away from one basestation and moves closer to the next one, it cycles through a sequence of channel rates which goes from a high rate to a low rate and then back again. Under MAX WEIGHT the queues exhibit an exponential behavior. In contrast the tracking algorithms can easily maintain small queues.

III. MULTI-CARRIER SCHEDULING

In this section we give a unified view of multi-carrier scheduling that is motivated by our previous work [7] and [6]. We begin by examining the situation where we can make a scheduling decision in every timeslot. We then look at the problem of template scheduling where we create a schedule

for a fixed number of timeslots that can be repeated multiple times.

A. Slot-by-slot scheduling

Suppose that a scheduling decision is made in every timeslot. At each timeslot t we know the channel rate $r_{i,c}(t)$ that represents the amount of service that carrier c can provide to user i in timeslot t. The goal is to calculate an assignment of carriers to users. The main constraint is that each carrier can be assigned to at most one user. Note that each user can receive service from multiple carriers. We let $x_{i,c}(t) = 1$ if carrier c is assigned to user i at time t and $x_{i,c}(t) = 0$ otherwise. The total amount of service that user i receives in time slot t is given by $R_i(t) = \sum_c r_{i,c}(t)x_{i,c}(t)$. We let $q_i(t)$ be the queue for user i at time t and $a_i(t)$ be the amount of data that arrives for user i during that time period. The queue is updated according to $q_i(t+1) = \max\{0, q_i(t) + a_i(t) - R_i(t))\}$. Throughout this section we concentrate on the traffic model with external arrivals since this is where the majority of the interesting problems lie.

We have now formulated the basic framework of multi-carrier scheduling. But what objective should we be trying to optimize at each time step? In order to answer this question we look back at the single-carrier algorithms MAX WEIGHT and PROPORTIONAL FAIR. Note that both of these algorithms are *weight-based*, i.e. they always serve the user that maximizes $w_i(t)r_i(t)$ for some weight i. For MAX WEIGHT the weight $w_i(t) = q_i(t)$ and for PROPORTIONAL FAIR the weight $w_i(t) = 1/\mu_i(t)$.

In light of the above, a natural goal for a multi-carrier scheduler is to optimize an analogous weight-based metric. The most natural such objective is to maximize $\sum_i w_i(t)R_i(t)$, where $w_i(t)$ is one of the weights given above. Optimizing this objective is simple. At each time step t we simply assign carrier c to the user i for which $w_i(t)r_{i,c}(t)$ is maximum. However, this approach has one big drawback, namely that if $R_i(t) > q_i(t)$ then the user cannot utilize all of the service that it is assigned. This would not necessarily affect properties such as the stability of the MAX WEIGHT algorithm. However, it could lead to larger queue sizes and delays since some of the bandwidth can be wasted. Simulation results for an example where this occurs are shown in Figure 3. The left plot shows queue sizes for an algorithm that maximizes $\sum_i w_i(t)R_i(t)$. The right plot shows the results for Algorithm 1 (described below) which takes the queuesizes into account when making scheduling decisions.

We would like to design a scheduling algorithm that no longer receives "credit" for allocating service to a mobile user that it cannot use.[1] An improved objective is therefore to maximize $\sum_i w_i(t)\min\{R_i(t), q_i(t)\}$. However, it is not clear *a priori* whether or not we can implement this objective since we can no longer find the optimum value by simply considering each carrier in isolation.

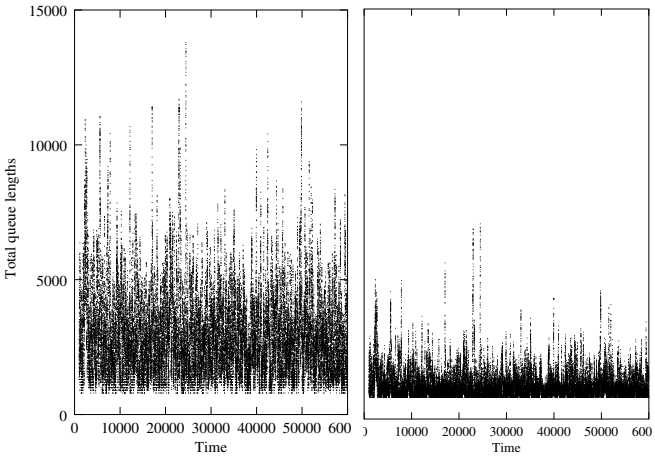


Fig. 3. Simulated trace: total queue size.
(Left) $\max \sum_i w_i(t)R_i(t)$. Mean: 3008, 95%: 5827, Median: 2744. (Right) Algorithm 1 for $\max \sum_i w_i(t)\min\{R_i(t), q_i(t)\}$. Mean: 1257, 95%: 2228, Median: 1104.

In [7] two main results were established. The first result states the hardness of the new objective.

Theorem 6: The objective $\max \sum_i w_i(t)\min\{R_i(t), q_i(t)\}$ is NP-hard to optimize. Moreover, for some $\varepsilon > 0$, there is no polynomial time $1-\varepsilon$ approximation algorithm for this problem unless P=NP.

We therefore turn our attention to what approximations can be achieved. Luckily we are able to show that optimizing $\sum_i w_i(t)\min\{R_i(t), q_i(t)\}$ takes the form of the well-known combinatorial problem that is maximizing a submodular function over a matroid. For this the Greedy algorithm provides a good approximation. In order to make sense of this let us define our terms.

A matroid is a set system over some ground set Ω. It consists of a set I of subsets of Ω that we call *independent* sets. In order for I to define a matroid its constituent sets must satisfy,

- $\emptyset \in I$.
- If $A \in I$ and $B \subseteq A$ then $B \in I$.

[1]The issue of whether or not to adjust the service rate based on queuesize when making a scheduling decision is also an issue for single-carrier scheduling as well. However, in that case the solution is easy. We simply replace the rule serve $\text{argmax}_i w_i(t)r_i(t)$ with the rule serve $\text{argmax}_i w_i(t)\min\{r_i(t), q_i(t)\}$. This change is relatively simple to make. However, our experience is that in general the modification does not affect performance a great deal. The reason that we get more of a benefit in the multi-carrier case is twofold. First, overall bandwidth is likely to be higher in next-generation OFDM-based systems and so it is more likely that assigning all of the bandwidth to a single user in a timeslot will be more service than it is able to utilize. Second, multi-carrier scheduling gives us the freedom to optimize scheduling assignments by allocating different fractions of the bandwidth to different users. If we do not take into account the current queuesize for a user then we cannot take full advantage of this freedom.

- If $A, B \in I$ and $|A| > |B|$ then $\exists a \in A$ such that $B \cup \{a\} \in I$.

A function $f(\cdot)$ on I is submodular if $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for all $A, B \in I$.

Although $\max_{A \in I} f(A)$ for a submodular function f over a matroid I is NP-hard in general the following greedy algorithm is known to always find a solution that is within a factor $1/2$ of optimal [10].

- Initially $A = \emptyset$.
- Let $a = \text{argmax}\{f(A \cup \{a\} - f(A) : A \cup \{a\} \in I\}$.
- Set $A = A \cup \{a\}$.
- Repeat until no such a can be found.

Using the above framework, the second result in [7] presents a number of approximation algorithms for multi-carrier scheduling. We include a brief description of two of them below. The two algorithms differ in how the matroid is defined.

Algorithm 1: This algorithm is a greedy algorithm that goes through the carriers one by one. It assigns each carrier to the user that maximizes $w_i(t) r_{i,c}(t)$. The only twist is that a user is no longer a candidate for a carrier if we have already assigned it service more than $q_i(t)$. That is, we keep a running count of $R_i(t)$ that we initialize to 0. We then go through the carriers in order. When considering carrier c we first disqualify all users for which $R_i(t) \geq q_i(t)$. From the remaining users we assign the carrier to the one that maximizes $w_i(t) r_{i,c}(t)$. If user i is chosen for the assignment we then update $R_i(t)$ by $R_i(t) = R_i(t) + r_{i,c}(t)$.

In order to bound the performance of this algorithm we consider a ground set $\Omega = \{(i, c)\}$ that consists of all possible user-carrier pairs. A subset $I \subseteq \Omega$ is declared to be an independent set if and only if $(i, c), (i', c') \in I \Rightarrow c \neq c'$, i.e. each carrier c appears at most once in I. It is proved in [7] that this defines a matroid over the ground set. Moreover, the objective function $\sum_i w_i(t) \min\{R_i(t), q_i(t)\}$ defines a submodular function over this matroid and the above multicarrier scheduling algorithm corresponds to the greedy algorithm for maximizing this submodular function over the matroid. We can therefore apply the result of [10] to claim that the scheduling algorithm always provides a solution that is within a factor $1/2$ of optimal.

Theorem 7: Algorithm 1 is a $1/2$-approximation for the objective $\max \sum_i w_i(t) \min\{R_i(t), q_i(t)\}$.

Algorithm 2: This algorithm is an alternative greedy algorithm that works by going through the users in order instead of going through the carriers in order. In particular, we now consider a ground set of the form $\Omega = \{(i, S_i)\}$, where the set S_i represents a set of carriers assigned to user i. A subset $I \subseteq \Omega$ is now declared to be an independent set if and only if $(i, c), (i', c') \in I \Rightarrow i \neq i'$, i.e. each user appears at most once. The objective $\sum_i w_i(t) \min\{R_i(t), q_i(t)\}$ once again corresponds to a submodular function over this matroid. (An important fact to note however is that when considering $(i, S_i), (j, S_j)$ such that some carrier c is contained in both S_i and S_j, we must only count the service due to carrier c once. In particular when calculating the objective function we would assign such a carrier to the user that gives the maximum benefit with respect to the objective function.)

Algorithm 2 is a little more complex than algorithm 1. It operates by considering the users in turn. At all times we keep track of the total contribution of carrier c to the objective. We use y_c to refer to this quantity. For each user i we assign it the set of carriers that maximizes the increase in the objective. (Although this subproblem is hard in general we can use a standard bin-packing algorithm to obtain a $(1 - \varepsilon)$-approximation.) Once again it can be shown that this algorithm corresponds to the Greedy algorithm for maximizing the submodular function over this new matroid and so the value of the objective is within a factor $1/2$ of optimal.

Theorem 8: Algorithm 2 is a $1/2$-approximation for the objective $\max \sum_i w_i(t) \min\{R_i(t), q_i(t)\}$.

B. Template scheduling

We now turn to the problem of creating a template schedule. Recall that a template schedule is a schedule for a fixed number of time slots that can be repeated one or more times. This is an important problem for scenarios where we cannot afford to signal all of the schedule information to all the users in every timeslot. If we use a template schedule then we can reuse the schedule multiple times, thereby reducing the amount of signaling required. It also allows us to provide a fixed long-term service rate to each user. This type of service guarantee over static channels is sometimes required for fixed wireless applications.

Let the size of the template be T timeslots. We assume throughout this subsection that the channel conditions do not change over the length of the template since otherwise it would not be possible to create an efficient template schedule. Let $r_{i,c}$ be the rate on channel c for user i. The goal is to assign the carriers to users at *each* timeslot in the template. As before we let $x_{i,c}(t) = 1$ if carrier c is assigned to user i at timeslot t and $x_i(t) = 0$ otherwise.

Since we wish to assign a number of timeslots at once it no longer makes as much sense to talk about serving an instantaneous queuesize. We therefore consider an alternative objective of meeting some average service rate a_i.

We divide the problem into two subproblems. The goal of the first subproblem is to choose *how many* timeslots are assigned to user i on each carrier c. We denote this number by $f_{i,c}$. The goal of the second subproblem is to choose *which* timeslots are assigned to user i on each carrier c.

For the first objective we want to to choose the values $f_{i,c}$ so as to satisfy the arrival rates, i.e. we want to choose integers $f_{i,c}$ so that,

$$\begin{aligned} \sum_c f_{i,c} r_{i,c} &\geq a_i T \\ \sum_i f_{i,c} &\leq T. \end{aligned}$$

If that is not feasible then one objective to optimize would be,

$$a_i \min\{a_i, \sum_c f_{i,c} r_{i,c}\}$$

subject to

$$\sum_i f_{i,c} \leq T.$$

It is not hard to see that this optimization problem has essentially the same form as the single timeslot problem considered in the first part of Section III. This means that it is NP-hard to approximate to within a factor $1-\varepsilon$. However, by allocating the $f_{i,c}$ on a carrier-by-carrier basis or a user-by-user basis we can obtain an assignment that comes within a factor $1/2$ of optimal.

For the remainder of this section we concentrate on the second problem, that of choosing which $f_{i,c}$ slots on carrier c should be assigned to user i. We wish to do this in such a way that the service to user i is as *smooth* as possible. The aim is for the delay and jitter to be as small as possible. More formally, suppose that the choice of the $f_{i,c}$ exactly matches the data arrival rates. Let $y_{i,c}(t)$ be the number of slots allocated to user i on carrier c up to time t. This implies that $y_{i,c}(T) = f_{i,c}$. For $T+1 \leq t \leq 2T$ we define $y_{i,c}(t) = f_{i,c} + y_{i,c}(t-T)$. We define the total service to user i on carrier c up to time t by $N_{i,c}(t) = y_{i,c}(t) r_{i,c}$. We then define the total service to user i up to time t by $N_i(t) = \sum_c N_{i,c}(t)$. We define the *worst-case delay* for user i to be $E_i = \max_{1 \leq s \leq t \leq 2T}(t-s) - ((N_i(t) - N_i(s))/a_i)$. This definition of smoothness is taken from the network calculus literature (see e.g. [9]) and was also used in [3] in the context of providing smooth schedules in input-queued switches. The delay expression corresponds to the maximum delay that user i data would experience if it were injected at a constant rate a_i. Since the delay is typically larger for smaller values of a_i, when comparing delays across multiple users we sometimes normalize the delays and consider the quantity $a_i E_i$. It is easy to construct instances where for some i we must have $a_i E_i \geq 1$. (For example if we have a single carrier c and $f_{i,c} = 1$ for all i.) Hence our aim is to design scheduling algorithms for which $a_i E_i$ is close to 1, i.e. E_i is close to $a_i = T / \sum_c f_{i,c} r_{i,c}$.

Scheduling timeslots for users in a single carrier system is equivalent to a template construction problem that has been considered for a number of years. One well-known algorithm due to Itai and Rosberg [12] and Hofri and Rosberg [11] operates by spacing out tokens on a unit circle using the golden ratio and then scheduling users in the order in which their tokens appear on the unit circle. The papers [12] and [11] provide a number of smoothness properties for this approach.

We can adapt the Golden Ratio algorithm by simply using it to schedule each carrier in isolation. We show in [6] that there are values of $f_{i,c}$ for which $E_i > \max_c(2.065 - \varepsilon)T/f_{i,c}$ when using this approach. The obvious question therefore is whether we can do any better. We show in [6] that by using a slightly different single-carrier template scheduling algorithm (that is based on bipartite matching) and then applying it on each carrier separtely, we can obtain a slightly better delay of $E_i \leq \max_c 2 + (2T/f_{i,c})$.

The main result of [6] is that in many situations we can do substantially better via randomization. In particular, suppose that for each carrier c and user i we space out $f_{i,c}$ tokens evenly on a unit circle. We then randomly shift these tokens by a common amount. We next do this for each of the other carriers. Lastly, for each carrier we go around the tokens on the circle in order and assign carrier c slots to the users in the order that the carrier c tokens appear.

In [6], two probabilistic analyses are presented for the above method. The first simply looks at the worst-case spacing on each carrier and allows us to prove:

Theorem 9: With probability $1-\gamma$, the randomized template construction algorithm produces a schedule for which,

$$E_i \leq \max_c \frac{T}{f_{i,c}} + 2 + \sqrt{\frac{N}{2} \log\left(\frac{2TC}{\gamma}\right)}, \qquad (3)$$

where N is the number of users in the system and C is the number of carriers.

The second analysis looks at the aggregate service over all carriers and allows us to prove:

Theorem 10: With probability $1-\gamma$, the randomized template construction algorithm produces a schedule for which,

$$E_i \leq \frac{T}{\sum_c f_{i,c} r_{i,c}} \sqrt{\frac{1}{2} \log\left(\frac{T^2}{\gamma}\right) \sum_c \left(\frac{f_{i,c} r_{i,c} E_{i,c}}{T}\right)^2}, \quad (4)$$

where $E_{i,c}$ is the quantity inside the $\max_c\{\cdot\}$ operator in Inequality (3).

For many values of N, C and $f_{i,c}$, Inequalities (3) and (4) are significantly lower than the worst-case bound for the Golden Ratio algorithm. Numerical results in [6] confirmed that our randomized template construction algorithm can indeed give lower delay performance.

IV. Future work

In this paper we have surveyed a number of models for scheduling data in wireless networks and discussed ways in which this work can be adapted for the multi-carrier scenario. There are a number of open problems, in particular related to multi-carrier scheduling. For example, we have already mentioned that signaling overhead can be high in multicarrier systems since we need to tell the mobile users which carriers have been assigned to which users. For this reason some systems make it a requirement that all of the carriers assigned to a single user i must be consecutive, i.e. if carrier c_1 and c_2 are both assigned to user i at time t and $c_1 \leq c \leq c_2$, then carrier c must also be assigned to user i at time t. This means that we only need to signal a range of carriers to each user i, rather than an arbitrary subset of carriers. We are interested in adapting our multicarrier scheduling algorithms so that they respect that constraint.

Another complication comes from the fact that when the channel rate $r_{i,c}(t)$ is calculated, this calculation is based on the assumption that the best modulation and encoding scheme is chosen for a given signal-to-noise ratio. If a particular user is assigned multiple carriers and wants to obtain the best service rate on each carrier, it may have to utilize a different modulation and encoding scheme on the different carriers. In some systems this may be hard to achieve. We would therefore like to understand better the effect on our schemes of a situation in which each user can only use a single modulation and encoding scheme on each carrier and so the total amount of data that can be transmitted to user i at time t might be strictly less than $\sum_c r_{i,c}(t)x_{i,c}(t)$.

Lastly we are interested in studying the issues that arise when applying the tracking algorithms of Section II to multi-carrier systems.

REFERENCES

[1] R. Agrawal and V. Subramanian. Optimality of certain channel aware scheduling policies. In *Proceedings of the 40th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, October 2002.

[2] M. Andrews. Instability of the proportional fair scheduling algorithm for HDR. *IEEE Transactions on Wireless Communications*, 3(5), 2004.

[3] M. Andrews and M. Vojnovic. Scheduling reserved traffic in input-queued switches: New delay bounds via probabilistic techniques. In *Proceedings of IEEE INFOCOM '03*, 2003.

[4] M. Andrews and L. Zhang. Scheduling over a time-varying user-dependent channel with applications to high speed wireless data. *Journal of the ACM*, 52(5):809 – 834, September 2005.

[5] M. Andrews and L. Zhang. Scheduling over non-stationary wireless channels with finite rate sets. *IEEE/ACM Transactions on Networking*, 2006.

[6] M. Andrews and L. Zhang. Creating templates to achieve low delay in multi-carrier frame-based wireless data systems. In *submitted*, 2007.

[7] M. Andrews and L. Zhang. Scheduling algorithms for multi-carrier wireless data systems. In *The Thirteenth Annual International Conference on Mobile Computing and Networking*, Montreal, Canada, September 2007.

[8] P. Bender, P. Black, M. Grob, R. Padovani, and N. Sindhushayana A. Viterbi. CDMA/HDR: A bandwidth efficient high speed data service for nomadic users. *IEEE Communications Magazine*, July 2000.

[9] J. Y. Le Boudec and P. Thiran. *Network Calculus*. Springer Verlag, http://ica1www.epfl.ch/ PS_files/NetCal.htm, 2004.

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

[11] M. Hofri and Z. Rosberg. Packet delay under the golden ratio weighted TDM policy in a multiple-access channel. *IEEE Transactions on Information Theory*, 33:341 – 349, 1987.

[12] A. Itai and Z. Rosberg. A golden raio control policy for a multiple-access channel. *IEEE Transactions on Automatic Control*, 29(8):712 – 718, August 1984.

[13] A. Jalali, R. Padovani, and R. Pankaj. Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC2000-Spring*, Tokyo, Japan, May 2000.

[14] H. Kushner and P. Whiting. Asymptotic properties of proportional-fair sharing algorithms. In *40th Annual Allerton Conference on Communication, Control, and Computing*, 2002.

[15] A. Stolyar. On the asymptotic optimality of the gradient scheduling algorithm for multiuser throughput allocation. *Operations Research*, 53:12 – 25, 2005.

[16] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936 – 1948, December 1992.

[17] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 30:466 – 478, 1993.

[18] D. Tse. Multiuser diversity in wireless networks. http://www.eecs.berkeley.edu/˜dtse/stanford416.ps, 2002.