Scheduling Algorithms for Multi-Carrier Wireless Data Systems

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

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

ABSTRACT

We consider the problem of scheduling wireless data in systems such as 802.16 (WIMAX). Each scheduling decision involves constructing a frame of one or more time slots. Within each time slot *multiple* carriers must be assigned to users. The important aspect of our problem is that a scheduler knows the channel rates across all users and all carriers whenever a scheduling decision is made. Hence there is no need to treat each carrier in complete isolation. This gives a potential for enhancing performance by allocating multiple carriers simultaneously.

We analyze this problem in a situation where finite queues are fed by a data arrival process. We generalize the well-known *MaxWeight* algorithm for the single-carrier setting to accommodate a number of natural optimization problems in the multi-carrier setting. We state the hardness of these problems and present simple algorithmic solutions with provable performance bounds. We also validate our algorithms via numerical examples.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; F.2.m [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms

Algorithms, Design, Theory

Keywords

Scheduling, multiple carriers, stability, queue performance, Max Weight, wireless communication, WIMAX

1. INTRODUCTION

The advent of wireless data systems has led to renewed interest in scheduling data in multiuser systems. In recent years a large body of work has looked at the problem of scheduling over time-varying user-dependent channels in a cellular wireless system. (See Figure 1.) This work examines a number of different models. For example, in *the finite-queue model* (e.g. [26, 27]) the aim is to keep the system stable assuming the queues are fed by an exogenous arrival process. Alternatively, in *the infinitely-backlogged model* (e.g. [28, 25, 17]) the aim is to maximize the system utility assuming the queues are permanently backlogged. Other work examines the difference between models where the channel rates are governed by some stationary stochastic process and models where a worst-case adversarial channel process is assumed, e.g. [2, 7]. However, most of the previous work looks at a situation with a single wireless carrier in which we can make a scheduling decision on a time slot by time slot basis. Some wireless systems however, have multiple carriers in which we can assign different carriers to different users. Examples include multi-carrier CDMA systems and also systems such as IEEE 802.16 (WIMAX), EV-DO Revision C and the Long-Term Evolution (LTE) of UMTS that use an OFDMA (orthogonal frequency division multiple access) physical layer in which different "tones" can be assigned to different users at each time. Another feature of most OFDMA-based systems is that we cannot schedule each time slot in isolation. Time is divided into frames of multiple time slots and we must populate the entire next frame whenever a frame ends. (See Figure 2.)

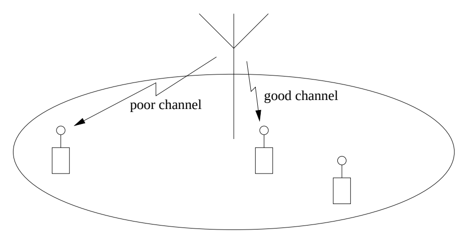

Figure 1: A cellular wireless system.

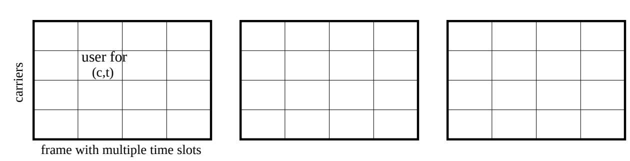

Figure 2: Schedule for a multi-carrier and frame-based wireless system.

MobiCom'07, September 9–14, 2007, Montréal, Québec, Canada.

In this paper we study the problem of scheduling in multi-carrier and frame-based systems. We focus on the issue of multiple carriers since as we argue later multiple timeslots per frame can be regarded as a special case of multiple carriers.

A straightforward approach for multiple carriers is to schedule carriers one by one independently by using an existing scheduling algorithm for each carrier in turn. Under such a simple adaptation, it is unclear *a priori* if the performance of a scheduling algorithm can be directly translated from a single-carrier system to a multi-carrier system. For example, all carriers could favor the same user which could lead to an excessive amount of service to one user. A main goal of this paper is to examine how to adapt the popular algorithm known as *MaxWeight* to the case of multiple carriers. We present a number of natural analogues of *MaxWeight* in the multi-carrier setting and prove their performance bounds against different objective functions. We show the trade-off between the complexity of the variants and their performance.

Our approach is based on the natural assumption that a multi-carrier scheduler knows the channel rates across all users and all carriers whenever a scheduling decision is made. This "global" information may give a potential for enhancing performance via an optimized allocation of carriers to users. Another purpose of this paper is to investigate the benefits of jointly allocating multiple carriers versus the isolated local optimization of each carrier.

1.1 The Model

We consider a single basestation transmitting data to a set of N wireless users. The basestation can transmit on a set of C carriers. At each time step, multiple carriers may be assigned to the same user; each carrier however can be assigned to at most one user. We use the indicator variable $x(i,c,t)$ to indicate whether or not carrier c is assigned to user i at time t. Due to the wireless nature of the channel, the channel rate depends on the user and the time slot. The channel rate can also depend on the carrier since the radio propagation environment may be different for different carriers, e.g. when carriers are on different frequencies. Hence, we use $r(i,c,t)$ to denote the channel rate of carrier c for user i at time step t. If $x(i,c,t) = 1$, then data of size $r(i,c,t)$ can be transmitted to user i at time t on carrier c.

Our goal is to *schedule* the system, i.e. to choose the values $x(i,c,t)$ in the most advantageous way. In a frame-based system such as 802.16, time is divided into frames of length T. We focus on the case that the frame-size is one time slot since we can convert our arguments to the situation with larger frames by treating the carriers in subsequent time slots as "new carriers". (Note that since the schedule for a frame needs to be constructed before the frame starts, and we need to know the channel rate for all tuples (i,c,t), this only makes sense under the assumption that the channel rate does not vary significantly over the duration of the frame. However, we believe that this is a natural assumption since in order for efficient frame-based scheduling to be possible, the frame size would need to chosen so that this assumption holds.)

1.2 Problems and Results

We first consider a finite-queue model with an external arrival process. Let $a_i(t)$ be the amount of data that arrives for user i in time slot t. One objective is to keep the system stable, i.e. to keep queues bounded whenever this is achievable. In the single carrier situation, the well studied *MaxWeight* algorithm always serves the user that maximizes $Q_i^s(t)r(i,t)$ at each time step t. Here $Q_i^s(t)$ denotes the queue size of user i at the beginning of time slot t. The *MaxWeight* algorithm is known to have desirable stability properties. The proof relies on showing that if the queue sizes are large then *MaxWeight* creates a negative drift in the Lyapunov function $\sum_i (Q_i^s(t))^2$.

We consider a number of ways to emulate the *Max-Weight* algorithm in multi-carrier systems. Let $\mu(i,t) = \sum_c r(i,c,t) \cdot x(i,c,t)$ be the amount of service user i receives at time t. We consider three objective functions when scheduling time slot t.

$$\max \quad \sum_i Q_i^s(t)\mu(i,t); \tag{1}$$

$$\max \quad \sum_i Q_i^s(t) \min\{Q_i^s(t), \mu(i,t)\}; \tag{2}$$

$$\max \quad \sum_i (Q_i^s(t))^2 - (Q_i^e(t))^2. \tag{3}$$

Objective (1) is the simplest analogue of *MaxWeight* which maximizes $\sum_i Q_i^s(t)r(i,t)x(i,t)$ for the single-carrier case. However, optimizing (1) has the potential shortcoming of assigning more service to a user than it can actually use.[1] Hence, although this effect may not change the stability properties of *MaxWeight*, it can lead to much larger queue sizes (and hence packet delays) than are necessary. Objective (2) offers a natural fix by replacing $\mu(i,t)$ with $\min\{Q_i^s(t), \mu(i,t)\}$ in the objective function. Objective (3) explicitly maximizes the negative drift of the Lyapunov function, where $Q_i^s(t)$ and $Q_i^e(t)$ denote the queue size of user i at the beginning and at the end of time slot t. Both objectives (2) and (3) are more sensitive to maintaining small queues than objective (1). We propose five variations of the *MaxWeight* algorithm, and refer to them as *MaxWeight-Alg1* through *MaxWeight-Alg5* (or *Alg1* through *Alg5* in short). In this paper we prove a collection of results regarding these five algorithms in relation to the three objectives.

1. We describe *MaxWeight-Alg1* that assigns each carrier c to the user that maximizes $Q_i^s(t)r(i,c,t)$. This carrier-by-carrier algorithm optimizes objective (1).
2. Somewhat surprisingly, both objectives (2) and (3) are NP-hard to optimize. Furthermore they cannot be approximated[2] to within a factor of $1-\delta$ for some constant δ.
3. Since we cannot hope for optimum solutions to objectives (2) and (3) we focus on developing *approximation algorithms*. We present *MaxWeight-Alg2* and *MaxWeight-Alg3* that are variations of *MaxWeight-Alg1*. They provide a $\frac{1}{2}$-approximation and a $\frac{1}{3}$-approximation for objectives (2) and (3) respectively.

[1] Although this problem exists for the single-carrier setting, we believe that it is particularly acute for multi-carrier systems where the amount of service that can be assigned in a single time slot is relatively large.

[2] If *opt* is the optimal value of a maximization problem, we say an algorithm is an α-approximation algorithm if it always returns a solution whose value is at least αopt. If for every algorithm, there are instances for which the algorithm cannot guarantee an α approximation then we say that the problem cannot be approximated to within a factor α.

4. We present *MaxWeight-Alg4* which, unlike *Alg1*, *Alg2* and *Alg3*, operates by considering each user in turn and choosing an optimal set of carriers for that user. This user-by-user algorithm gives an alternative $\frac{1}{2}$-approximation for objective (2). *Alg2* and *Alg4* are special instances of a powerful algorithm for maximizing a submodular function over a matroid.
5. We show that there are scenarios for which *Max-Weight-Alg2*, *Alg 3* and *Alg4* achieve at most a $\frac{1}{2}+\varepsilon$ fraction of their respective optimal objective values.
6. We present a more complex algorithm *MaxWeight-Alg5* that is based on a recent algorithm for the Generalized Assignment Problem [12]. It improves the approximation ratio for objective (2) to $1-\frac{1}{e}-\varepsilon$ for any $\varepsilon > 0$. Although we believe that *Alg5* is not simple enough to be practical for wireless systems, we feel that it does provide theoretical insight into the multi-carrier scheduling problem.
7. We show that the stability properties of the single-carrier *MaxWeight* algorithm also apply to the multi-carrier algorithms *MaxWeight-Alg1* through *Alg5*.
8. We show how to adapt our algorithms to the case in which the weights are more general than simple queue sizes. This is important if we wish to address objectives such as fairness in addition to maintaining small queues. In particular we define multi-carrier analogues of well-known weight-based algorithms such as the Proportional Fair algorithm [28, 15].[3]
9. We present simulation results to show that although *MaxWeight-Alg2*, *Alg3* and *Alg4* may not optimize objectives (2) and (3), they still significantly outperform *MaxWeight-Alg1* due to the fact that they are trying to optimize better objectives. The reason for the improved performance is that *MaxWeight-Alg1* will often try to assign more service to a user than it can actually use. This behavior does not occur for algorithms *MaxWeight-Alg2*, *Alg3* and *Alg4*. We also observe that although *Alg2*, *Alg3* and *Alg4* have similar approximation ratios for a single time slot in isolation, their performance can be dramatically different over longer timescales.

1.3 Related work

The *MaxWeight* algorithm was first shown to perform well in wireless networks by Tassiulas and Ephremides [26, 27]. Other papers that study Max-Weight include [4, 3, 21]. Two algorithms that are similar to *MaxWeight* are *MaxDelay* [4, 3] and *Exp* [22, 23]. *MaxDelay* allocates service to user $\arg\max_i \Delta_i(t) r_i(t)$ where $\Delta_i(t)$ denotes the Head-of-Line delay for user i at time t. Exp is a more complex algorithm that provides more control over the relative delays that the users experience.

The above algorithms were designed for the case in which the finite queues are fed by an arrival process. For the case in which the queues are infinitely backlogged and we wish to maximize a system utility function the Proportional Fair algorithm was introduced by [28, 15] and studied in [25, 17, 1]. It was shown in [2] that Proportional Fair does not work so well when the queues are fed by an arrival process. In particular it can cause the queues to be unstable. Algorithms for optimizing utility functions subject to fairness requirements and constraints on minimum/maximum throughput have been studied by [18, 19, 5]. Algorithms that combine the goals of system utility maximization and queue stability have recently been presented by [24, 10, 20]. In [6] it was shown that unless these problems are studied jointly, system oscillations can occur. We remark that all of this previous work on wireless scheduling has looked at a single carrier in isolation.

[3] The single carrier Proportional Fair algorithm always serves the user that maximizes $(1/R_i(t))r(i,t)$, where $R_i(t)$ is an estimate of the average service rate provided to user i.

2. CANDIDATE ALGORITHMS

In this section we define a number of algorithms that aim to emulate the *MaxWeight* algorithm in the multi-carrier scenario. We focus on constructing a schedule for a single timeslot. It is not difficult to adapt our techniques to a frame with multiple timeslots by simply considering each (time slot, carrier) pair as a separate carrier. As mentioned in the Introduction, the reason that the problem differs from the single carrier problem is that we know the channel rates for all the carriers and users at the beginning of the time step and hence there is a potential benefit to jointly optimizing the allocation of carriers.

We utilize the following notation. For convenience, the dependence on t is omitted.

$$\begin{aligned} Q_i^s &= \text{queue size for user } i \text{ at the beginning of time } t \\ &\quad\ \text{which includes the arrival } a_i(t) \text{ for time } t. \\ Q_i^e &= \text{queue size for user } i \text{ at the end of time } t \\ r(i,c) &= \text{rate for user } i, \text{ carrier } c \text{ during time } t \\ \mu_i &= \text{total service to user } i \text{ during time } t \end{aligned}$$

$$Q_i^c = \text{queue size for user } i \text{ after carrier } c \text{ is assigned}$$

Two equations that relate the above quantities are:

$$\begin{aligned} Q_i^{c+1} &= \max\{0, Q_i^c - r(i,c)\} \\ Q_i^e &= \max\{0, Q_i^s - \mu_i\} \end{aligned}$$

Recall the objectives (1), (2) and (3) defined in Section 1. We analyze the following five algorithms with respect to these three objectives. The first three algorithms go through the carriers in order. At time t carrier c serves user $\hat{i}$ defined below.

- *MaxWeight-Alg1:* $\hat{i} = \text{argmax}_i Q_i^s r(i,c)$, where argmax means $\hat{i}$ is the index that maximizes $Q_i^s r(i,c)$.
- *MaxWeight-Alg2:* $\hat{i} = \text{argmax}_i Q_i^s \min\{r(i,c), Q_i^c\}$.
- *MaxWeight-Alg3:* $\hat{i} = \text{argmax}_i Q_i^c \min\{r(i,c), Q_i^c\}$.
- *MaxWeight-Alg4* does not locally optimize each carrier in isolation. It considers each user one by one and finds the best carrier(s) for the user. However, this assignment can be modified if there is more benefit for a carrier to serve a later user. We defer the detailed description of this algorithm to Section 5
- *MaxWeight-Alg5* begins by approximately solving a relaxation of an integer linear program for objective (2) followed by rounding the fractional approximate solution. We defer the detailed description to Section 6.

We conclude this section with two simple results. The following theorem follows directly from the definition of *MaxWeight-Alg1*.

THEOREM 1. *MaxWeight-Alg1 optimizes objective (1).*

Our second result shows that objectives (2) and (3) are related.

LEMMA 2. *Any α-approximation algorithm for objective (2) provides a $\frac{\alpha}{2}$-approximation for objective (3).*

PROOF. Since $0 \le Q_i^e \le Q_i^s$ we have,

$$\begin{aligned} Q_i^s(Q_i^s - Q_i^e) &\le (Q_i^s + Q_i^e)(Q_i^s - Q_i^e) \\ &= (Q_i^s)^2 - (Q_i^e)^2 \\ &\le 2Q_i^s(Q_i^s - Q_i^e). \end{aligned}$$

In addition, $\min\{Q_i^s(t), \mu(i,t)\} = Q_i^s(t) - Q_i^e(t)$. Therefore objectives (2) and (3) are always within a factor of two of each other. □

3. HARDNESS OF OBJECTIVES (2) AND (3)

As we discussed in the Introduction, optimizing objective (1) is not ideal since it could lead to more service being allocated to a user than it is able to use and hence the queue sizes (and packet delays) may become larger than necessary. Hence it would be preferable to use objectives (2) and (3). In this section we show that unfortunately, we cannot hope for an efficient algorithm that optimizes objectives (2) and (3) in general.

THEOREM 3. *For some $\delta > 0$, there is no $(1-\delta)$-approximation algorithm for objectives (2) and (3) unless P=NP.*

PROOF. We use a reduction from the 3-bounded 3-dimensional matching problem. In this problem we are given a set $T \subseteq X \times Y \times Z$ where $|X| = |Y| = |Z| = n$. A 3-dimensional matching M is a subset $M \subseteq T$ such that no elements in M agree in any coordinate. In a 3-bounded instance each element in $X \cup Y \cup Z$ appears at most three times in T. The goal is to find a matching M of maximum cardinality. Kann [16] showed that there exists an ε such that it is NP-hard to decide whether the maximum size matching equals n or is at most $(1-\varepsilon)n$.

We now convert this into an instance of our problem. We use a reduction similar to that of [9] for a problem known as the Generalized Assignment Problem. For each hyperedge $e \in T$ we are given a user i_e. For each element $w \in X \cup Y \cup Z$ we have a carrier c_w. We call these $3n$ carriers *regular* carriers. We set the channel rate $r(i_e, c_w) = 1$ if w is a component of e, and $r(i_e, c_w) = 0$ otherwise. We have another set of $|T| - n$ dummy carriers c' for which $r(i, c') = 2 + \varepsilon$ for all users i. Let $Q_i^s = 3$ for all users i.

Given a scheduling solution, we partition the users into three sets A, B and C. Each user in A is assigned 3 regular carriers only. Note that the users in A correspond to a 3-dimensional matching and hence $|A| \le n$. Each user in B is assigned 1 dummy carrier and possibly 1 regular carrier; each user in C is assigned 1 or 2 regular carriers only. To see that A, B and C form a partition of the users that receive service, we observe that there is no benefit to assigning 1 dummy carrier and $x \ge 2$ regular carriers to a user since $Q_i^s = 3$. There is also no benefit to assigning 2 dummy carriers to a user since there is more benefit to reassigning one dummy carrier to a user with $x \le 2$ regular carriers. Such a user always exists since the number of users in $T - A$ is at least $|T| - n$, the number of dummy carriers. Therefore, $|B| = |T| - n$ and $|C| = n - |A|$.

Consider the $3n - 3|A|$ regular carriers not assigned to users in A. With respect to objectives (2) and (3), there is more benefit to assigning them to users in C than assigning them to users in B. However, we can assign at most 2 regular carriers to each user in C. Hence at least $3n - 3|A| - 2|C| = n - |A|$ regular carriers are assigned to users in B. Therefore, objectives (2) and (3) can be upper bounded as follows.[4]

$$\begin{aligned} & OBJ2 \\ =\ & \sum_i Q_i^s \min\{Q_i^s, \mu_i\} \\ \le\ & 3 \cdot (3|A| + (2+\varepsilon)|B| + (1-\varepsilon)(n - |A|) + 2|C|) \\ =\ & 3 \cdot (\varepsilon|A| + (2+\varepsilon)(|T| - n) + (3-\varepsilon)n) \qquad (4) \end{aligned}$$

and

$$\begin{aligned} & OBJ3 \\ =\ & \sum_i (Q_i^s)^2 - (Q_i^e)^2 \\ \le\ & 9|A| + (9 - (1-\varepsilon)^2)(|B| - (n - |A|)) \\ & \quad + 9(n - |A|) + 8|C| \\ \le\ & (2\varepsilon - \varepsilon^2)|A| + (9 - (1-\varepsilon)^2)|T| \\ & \quad + (2(1-\varepsilon)^2 - 1)n \qquad (5) \end{aligned}$$

We now consider two cases. If the size of the maximum 3-dimensional matching is indeed n, then $|A| = n$ and $|C| = 0$. In this case the upper bounds on objectives (2) and (3) that we have just derived are actually tight, i.e. $OBJ2_{|A|=n}$ equals (4) and $OBJ3_{|A|=n}$ equals (5). If the maximum 3-dimensional matching has size at most $(1-\varepsilon)n$, then $|A| \le (1-\varepsilon)n$. For both objectives, the drop in value is at least $\varepsilon^2 n$, i.e. $OBJ2_{|A|=n} - OBJ2_{|A| \le (1-\varepsilon)n} \ge \varepsilon^2 n$ and $OBJ3_{|A|=n} - OBJ3_{|A| \le (1-\varepsilon)n} \ge \varepsilon^2 n$.

We note that $|T| \le 3n$ since the matching instance is 3-bounded. Therefore, both objectives (2) and (3) are at most $27n$. This means that the relative difference in the objective values between the two cases is at least $\varepsilon^2/27$. By setting $\delta = \varepsilon^2/27$, we obtain our result. □

4. APPROXIMATION RATIOS OF *ALG2* AND *ALG3*

In this section we show that algorithms *MaxWeight-Alg2* and *Alg3* are constant-factor approximations for objectives (2) and (3). The hardness results of Section 3 implies that for these objectives, constant-factor approximation algorithms are the best that we can hope for. Moreover, in Section 9 we present simulation results to show that although these algorithms may not optimize objectives (2) and (3), they still signficantly outperform *MaxWeight-Alg1* due to the fact that *MaxWeight-Alg1* will often try to assign more service to a user than it can actually use.

[4]In this section we use OBJ2 and OBJ3 to denote objectives (2) and (3). We also use the subscript $|A| = n$ to denote the case in which the maximum size matching has size n and the subscript $|A| \le (1-\varepsilon)n$ to denote the case in which the maximum size matching has size at most $(1-\varepsilon)n$.

THEOREM 4. *MaxWeight-Alg2 is a $\frac{1}{2}$-approximation algorithm for objective (2). By Lemma 2, this immediately implies that it is a $\frac{1}{4}$-approximation algorithm for objective (3).*

PROOF. We show that *MaxWeight-Alg2* is a special case of the greedy algorithm for maximizing a nondecreasing submodular function over a matroid. In order to clarify this relationship we first define the following terms.

- Consider a ground set Ω and let I be a set of subsets of Ω. The set I is a *matroid* if,
  - $\emptyset \in I$.
  - If $A \in I$ and $B \subseteq A$ then $B \in I$.
  - If $A, B \in I$ and $|A| > |B|$, then there exists an element $a \in A \backslash B$ such that $B \cup \{a\} \in I$.
- A special case of a matroid is a *partition matroid*. We say that a matroid is a partition matroid if there is a partition of Ω into components $\Gamma_1, \Gamma_2, \ldots$ such that $A \in I$ if and only if $|A \cap \Gamma_k| \leq 1$, for all k.
- Let $f(\cdot)$ be a function on sets in I.
  - It is a *submodular* function on I if for all a, A, B such that $(A \cup \{a\}) \in I$ and $B \subseteq A$,
  $$f(A \cup \{a\}) - f(A) \leq f(B \cup \{a\}) - f(B).$$
  - It is a *non-decreasing* submodular function if in addition $f(\emptyset) = 0$ and for all a, A such that $(A \cup \{a\}) \in I$,
  $$f(A \cup \{a\}) - f(A) \geq 0.$$
- The Greedy algorithm for maximizing a nondecreasing submodular function over a matroid works as follows.
  - Initially let $A = \emptyset$.
  - Repeat the following procedure for as long as possible. Let $a = \arg\max_{a \in \Omega\ A \cup \{a\} \in I} f(A \cup \{a\}) - f(A)$. Set $A \leftarrow A \cup \{a\}$.

  For partition matroids the algorithm can be simplified. At step k, instead of considering all elements in Ω, we only need to find $a = \arg\max_{a \in \Gamma_k} f(A \cup \{a\}) - f(A)$.
- Fisher, Nemhauser and Wolsey [11] proved the following property of the above Greedy algorithm.

  LEMMA 5. *The Greedy Algorithm gives a $\frac{1}{2}$- approximation to the problem of maximizing a non-decreasing submodular function over a matroid.*

We now show that *MaxWeight-Alg2* is special case of the above Greedy Algorithm for partition matroids. The ground set $\Omega = \{(i, c) : 1 \leq i \leq N, 1 \leq c \leq C\}$. A subset $A \in I$ if and only if there is at most one element in A for each carrier. In other words, A defines a valid schedule. Let $\Gamma_c = \{(i, c) : 1 \leq i \leq N\}$. This clearly defines a partition matroid. The function $f(\cdot)$ is defined by,

$$f(A) = \sum_i Q_i^s \nu_i(A),$$

where $\nu_i(A) = \min\{\sum_{c:(i,c) \in A} r(i, c), Q_i^s\}$. Note that if $B \subseteq A$ then $\nu_i(B) \leq \nu_i(A)$. From this it is easy to see that for any element (i, c) where $A \cup \{(i, c)\}$ forms a valid schedule, $f(A \cup \{(i, c)\}) - f(A) \leq f(B \cup \{(i, c)\}) - f(B)$, i.e. the function $f(\cdot)$ is submodular. Moreover, it is clear that function $f(\cdot)$ corresponds directly to objective (2). Hence when we try to optimize objective (2) we are trying to find an assignment (which corresponds to an element of a partition matroid) that maximizes a submodular function. Recall that *MaxWeight-Alg2* goes through each carrier in turn and assigns it to the user that maximizes the increase in the objective. Hence *MaxWeight-Alg2* corresponds to the Greedy algorithm and so by the result of Fisher et al. it is a $\frac{1}{2}$-approximation algorithm for objective (2). $\square$

We now provide an analysis of algorithm *MaxWeight-Alg3*.

THEOREM 6. MaxWeight-Alg3 *is a $\frac{1}{3}$-approximation for objective (3).*

PROOF. Recall that Q_i^c is the queue size for user i after algorithm *MaxWeight-Alg3* has assigned carrier c. Let $\hat{Q}_i^c$ be the analogous queue size for the optimum algorithm, which we denote OPT. Consider any carrier c. Suppose that OPT assign carrier c to user k and *MaxWeight-Alg3* assigns it to user j. We first show that the gain obtained by algorithm *MaxWeight-Alg3* due to carrier c is at least one half of the gain obtained by OPT due to carrier c that is never obtained by *MaxWeight-Alg3*. More precisely, we show

$$\begin{aligned} & (Q_j^{c-1})^2 - (Q_j^c)^2 \\ \geq\ & \frac{1}{2}((\min\{\hat{Q}_k^{c-1}, Q_k^e\})^2 - (\min\{\hat{Q}_k^c, Q_k^e\})^2). \end{aligned} \quad (6)$$

Suppose that $\hat{Q}_k^c \leq Q_k^e$. We have,

$$\begin{aligned} & (Q_j^{c-1})^2 - (Q_j^c)^2 \\ \geq\ & Q_j^{c-1}(Q_j^{c-1} - Q_j^c) \\ \geq\ & Q_k^{c-1}\min\{Q_k^{c-1}, r(i,c)\} && \text{by defn of } Alg3 \\ \geq\ & Q_k^{c-1}\min\{Q_k^{c-1}, \hat{Q}_k^{c-1} - \hat{Q}_k^c\} && \text{by choice of OPT} \\ =\ & \min\{(Q_k^{c-1})^2, Q_k^{c-1}(\hat{Q}_k^{c-1} - \hat{Q}_k^c)\} \\ \geq\ & \min\{(Q_k^e)^2, \min\{\hat{Q}_k^{c-1}, Q_k^e\}(\hat{Q}_k^{c-1} - \hat{Q}_k^c)\} \\ \geq\ & \min\{(Q_k^e)^2, \\ & \quad \min\{\hat{Q}_k^{c-1}, Q_k^e\}(\min\{\hat{Q}_k^{c-1}, Q_k^e\} - \hat{Q}_k^c)\} \\ \geq\ & \min\{(\min\{\hat{Q}_k^{c-1}, Q_k^e\})^2, \\ & \quad \frac{1}{2}((\min\{\hat{Q}_k^{c-1}, Q_k^e\})^2 - (\hat{Q}_k^c)^2)\} \\ & \qquad \text{by assumption that } \hat{Q}_k^c \leq Q_k^e \\ \geq\ & \frac{1}{2}((\min\{\hat{Q}_k^{c-1}, Q_k^e\})^2 - (\hat{Q}_k^c)^2) \end{aligned}$$

Inequality (6) also holds true for the case that $\hat{Q}_k^c > Q_k^e$ since then there is no gain obtained by OPT that is not obtained by *MaxWeight-Alg3*. Algebraically, the inequality holds trivially in this case since the right-hand-side is zero. Note that the inequality also holds when $k = j$.

Now that we have verified Inequality (6), we proceed to prove the lemma. For clarity of the rest of the proof, we let j_c be the user that carrier c serves under *Alg3* and let k_c be the user that c serves under OPT. Note that k_c can be the same as j_c. We know that $Q_i^c = Q_i^{c-1}$ for $i \neq j_c$ and

$\hat{Q}_i^c = \hat{Q}_i^{c-1}$ for $i \neq k_c$. We therefore have,

$$
\begin{aligned}
&\sum_i (Q_i^s)^2 - (Q_i^e)^2 \\
=\ & \sum_i \sum_c (Q_i^{c-1})^2 - (Q_i^c)^2 \qquad \text{Telescoping on } c \\
=\ & \sum_c (Q_{j_c}^{c-1})^2 - (Q_{j_c}^c)^2 \\
& \qquad \text{since } Q_i^c = Q_i^{c-1} \text{ for } i \neq j_c \\
\geq\ & \sum_c \frac{1}{2}((\min\{\hat{Q}_{k_c}^{c-1}, Q_{k_c}^e\})^2 - (\min\{\hat{Q}_{k_c}^c, Q_{k_c}^e\})^2) \\
& \qquad \text{by Inequality (6)} \\
=\ & \sum_c \sum_i \frac{1}{2}((\min\{\hat{Q}_i^{c-1}, Q_i^e\})^2 - (\min\{\hat{Q}_i^c, Q_i^e\})^2) \\
& \qquad \text{since } \hat{Q}_i^c = \hat{Q}_i^{c-1} \text{ for } i \neq k_c \\
=\ & \sum_i \frac{1}{2}((\min\{\hat{Q}_i^s, Q_i^e\})^2 - (\min\{\hat{Q}_i^e, Q_i^e\})^2) \\
\geq\ & \frac{1}{2}\sum_i (Q_i^e)^2 - (\hat{Q}_i^e)^2 \\
=\ & \frac{1}{2}\sum_i ((\hat{Q}_i^s)^2 - (\hat{Q}_i^e)^2) - ((Q_i^s)^2 - (Q_i^e)^2)
\end{aligned}
$$

This immediately implies,

$$\sum_i (Q_i^s)^2 - (Q_i^e)^2 \geq \frac{1}{3}\sum_i (\hat{Q}_i^s)^2 - (\hat{Q}_i^e)^2,$$

which completes the proof. □

We conclude this section by showing that our analysis of *MaxWeight-Alg2* is essentially tight and our analysis of *MaxWeight-Alg3* cannot be significantly improved.

Theorem 7. *For any constant $\varepsilon > 0$, there exists an instance on which* MaxWeight-Alg2 *and* Alg3 *achieve at most a $1/(2-\varepsilon)$ fraction of the optimal value of objectives (2) and (3).*

Proof. The example is as follows. There are 2 users, each with $Q_i^s = 1$. The channel rates are given by $r(1,c) = 1$ for $c = 1, 2$, $r(2,1) = 1-\varepsilon$, and $r(2,2) = 0$. The optimal algorithm assigns carrier 1 to user 2 and carrier 2 to user 1. Hence the optimal values of objectives (2) and (3) are $2-\varepsilon$ and $2-\varepsilon^2$ respectively. On the other hand, algorithms *MaxWeight-Alg2* and *MaxWeight-Alg3* both assign carrier 1 to user 1 since $r(1,1) > r(2,1)$. □

5. MAXWEIGHT-ALG4: ALTERNATIVE 1/2 APPROXIMATION FOR OBJECTIVE (2)

We have so far considered algorithms that greedily assign each carrier to the best user. Instead of the carrier-by-carrier approach, we propose a new algorithm *MaxWeight-Alg4* which finds an optimal set of carriers for each user in a user-by-user fashion.

5.1 Definition of MaxWeight-Alg4

MaxWeight-Alg4 operates by going through the users one-by-one and making temporary assignments for carriers to users. For each carrier c it also maintains a quantity $\beta(c)$ that measures the best allocation so far for c. (The value of $\beta(c)$ is initialized to zero.) These quantities are derived from the following knapsack-like problem which is solved for each user i. The variables in this problem are the $b(i,c)$ and they represent an assignment of $b(i,c)$ bits from carrier c to user i.

$$
\begin{aligned}
\textit{Knapsack} \qquad & \max \sum_c (\max\{0, Q_i^s b(i,c) - \beta(c)\}) \\
\text{s.t.} \qquad & \\
& b(i,c) \leq r(i,c) \quad \forall i, c \\
& \sum_c b(i,c) \leq Q_i^s \quad \forall i
\end{aligned}
$$

For each user i, *MaxWeight-Alg4* first finds a solution to this *Knapsack* problem. (We show how to do this in Section 5.3.) If under this solution we have $Q_i^s b(i,c) > \beta(c)$ for carrier c then this means that there is a benefit to be gained by removing carrier c from its temporary assignment and reassigning it to user i. In this case *MaxWeight-Alg4* updates $\beta(c)$ to $Q_i^s b(i,c)\}$ and (re)assigns c to user i.

Remark. In order to simplify the analysis we assume (without loss of generality) that in the optimum *Knapsack* solution, at most one carrier is partially assigned to user i and this carrier has the highest index among carriers assigned to user i. That is, if there exists a carrier $\hat{c}$ such that $Q_i^s b(i,\hat{c}) > \beta(\hat{c})$ and $b(i,\hat{c}) < r(i,\hat{c})$ then $\hat{c} = c' := \arg\max_c\{b(i,c) > 0\}$. To see why this is a legitimate assumption, note that if it is not the case then we can increase $b(i,\hat{c})$ and decrease $b(i,c')$ at a common rate until either carrier $\hat{c}$ is fully assigned (i.e. $b(i,\hat{c}) = r(i,\hat{c})$) or else $b(i,c') = 0$ (in which case $\arg\max_c\{b(i,c) > 0\}$ becomes a different carrier). Hence we can continue this process until our assumption is satisfied.

5.2 Analysis of MaxWeight-Alg4

Theorem 8. MaxWeight-Alg4 *is a $\frac{1}{2}$-approximation algorithm for objective (2). By Lemma 2, this immediately implies that it is a $\frac{1}{4}$-approximation algorithms for objective (3).*

Proof. We follow the same approach as the proof for Theorem 4 by showing that *MaxWeight-Alg4* is equivalent to the greedy algorithm for optimizing a non-decreasing submodular function over a partition matroid. Consider the ground set $\Omega = \{(i, S_i) : 1 \leq i \leq N, S_i \in \Lambda_i\}$, where Λ_i is the set of all assignments of carriers to user i. We partition Ω into $\Gamma_i = \{(i, S_i) : S_i \in \Lambda_i\}$. We say that subset $A \subseteq \Omega$ is in I if and only if there is at most one element in A for each user i, i.e. $|A \cap \Gamma_i| \leq 1$ each i. This clearly defines a partition matroid.

Suppose we have $(i, S_i) \in A \in I$. If $c \in S_i$ we define $b_A(i,c) = \min\{r(i,c), Q_i^s - \sum_{c'<c: c' \in S_i} b_A(i,c')\}$, otherwise we set $b_A(i,c) = 0$. Let $\beta_A(c) = \max_i Q_i^s b_A(i,c)$. We define a function $f(\cdot)$ by

$$f(A) = \sum_c \beta_A(c).$$

Lemma 9. *$f(\cdot)$ is a non-decreasing submodular function over the above partition matroid.*

Proof. Consider any $B \subseteq A \in I$ and any (k, S_k) such that user k is not assigned any carriers under A, i.e. $|A \cap \Gamma_k| = 0$. Therefore, $A' = A \cup \{(k, S_k)\}$ and $B' = B \cup$

$\{(k, S_k)\}$ are members of the partition matroid I. To see $f(A') - f(A) \le f(B') - f(B)$, we consider the following cases.

- If $\beta_B(c) = \beta_{B'}(c)$, then $\beta_A(c) = \beta_{A'}(c)$. Therefore $\beta_{B'}(c) - \beta_B(c) = \beta_{A'}(c) - \beta_A(c) = 0$.
- If $\beta_B(c) < \beta_{B'}(c)$, then $\beta_{B'}(c) = Q_k^s b_{B'}(k, c)$.
  - If $\beta_A(c) = \beta_{A'}(c)$, then $\beta_{B'}(c) - \beta_B(c) > \beta_{A'}(c) - \beta_A(c) = 0$.
  - Otherwise, $\beta_{A'}(c) = Q_k^s b_{A'}(k, c)$. We know $\beta_B(c) \le \beta_A(c) \le \beta_{A'}(c)$ since $B \subseteq A \subseteq A'$. Moreover, $\beta_{B'}(c) = \beta_{A'}(c)$ since (k, S_k) is in both A' and B'. Therefore $\beta_{B'}(c) - \beta_B(c) \ge \beta_{A'}(c) - \beta_A(c) \ge 0$.

□

LEMMA 10. *If the set A corresponds to an assignment of carriers to users, function $f(A)$ equals objective (2) for this assignment.*

PROOF. Suppose set A corresponds to an assignment of carriers of users. By the remark at the end of Section 5.1, we can assume that for each user i there is at most one carrier that is only partially utilized by user i and this carrier has the highest index among carriers that are assigned to user i. This implies that the amount of service that carrier c provides to user i can be written as $\min\{r(i, c), Q_i^s - \sum_{c' < c: c' \in S_i} b_A(i, c')\} = b_A(i, c)$. Hence the total service to user i is $\sum_c b_A(i, c) \le Q_i^s$. Moreover, since each carrier is assigned to at most one user we have $\beta_A(c) = \max_i Q_i^s b_A(i, c) = \sum_i Q_i^s b_A(i, c)$. Therefore,

$$\begin{aligned} f(A) &= \sum_c \beta_A(c) \\ &= \sum_c \max_i Q_i^s b_A(i, c) \\ &= \sum_c \sum_i Q_i^s b_A(i, c) \\ &= \sum_i Q_i^s \sum_c b_A(i, c). \end{aligned}$$

The last line is equivalent to objective (2) for assignment A since the total amount of service to user i is $\sum_c b_A(i, c)$. □

We are now ready to prove Theorem 8. By comparing the *Knapsack* problem with the definition of $f(\cdot)$, we can see that the aim of *MaxWeight-Alg4* is to go through the users one-by-one and assign to each user the set of carriers that maximizes the increase in $f(\cdot)$. The equivalence is not exact however because if a carrier c is reassigned from user j to user i, the service that was provided by carrier c to user j could potentially be replaced by a another carrier that remains assigned to user j. However, this difference can only increase the value of objective (2).

In other words we have shown that *MaxWeight-Alg4* is strictly better than the Greedy algorithm for optimizing the non-decreasing submodular function $f(\cdot)$ over the partition matroid. Hence by the result of Fisher et al. *MaxWeight-Alg4* is a $\frac{1}{2}$-approximation algorithm for function $f(\cdot)$. By Lemma 10 and the fact that *MaxWeight-Alg4* produces a valid assignment of carriers, it is also a $\frac{1}{2}$-approximation algorithm for objective (2).

We now show that this analysis is tight.

THEOREM 11. *For any constant $\varepsilon > 0$, there exists an instance on which* MaxWeight-Alg4 *achieves at most a $1/(2-\varepsilon)$ fraction of the optimal value of objective (2).*

PROOF. We use the same example as in Theorem 7. Recall that there are 2 users, each with $Q_i^s = 1$. The channel rates are $r(1, c) = 1$ for $c = 1, 2$, $r(2, 1) = 1 - \varepsilon$ and $r(2, 2) = 0$. Recall also that the optimal value of objective (2) is $2 - \varepsilon$. *MaxWeight-Alg4* assigns carrier 1 to user 1 since it sets $b(1, 1) = 1$ and $b(2, 1) = 1 - \varepsilon$. Therefore it achieves value 1 for objective (2) which is at most a fraction $1/(2 - \varepsilon)$ from optimal. □

5.3 Solution of the Knapsack problem

In this section we describe how to solve the *Knapsack* problem that must be solved by *MaxWeight-Alg4* for each user i. The solution is similar to the standard algorithm for Knapsack problems. (See e.g. [14].) The main difficulty is due to the fact that there may be a carrier that is only partially utilized by user i. (Recall that we can assume there is at most one such carrier and it has the highest index among all the carriers assigned to user i.)

The algorithm first guesses the identity of the partially utilized carrier and denotes it by c_K. (In reality this involves trying all possible values of c_K.) For $k < K$, let $p_k = Q_i^s r(i, c_k) - \beta(c_k)$ be the *benefit* of assigning c_k to i; let $F_k(p)$ be the minimum size of Q_i^s such that we can obtain total benefit of at least p using the carriers $c_1, c_2, \dots, c_k$. We define

$$F_1(p) = \begin{cases} 0 & \text{for } p \le 0 \\ r(i, c_1) & \text{for } 0 < p \le p_1 \\ \infty & \text{for } p > p_1 \end{cases}$$

and

$$F_k(p) = \min\{F_{k-1}(p - p_k) + r(i, c_k), F_{k-1}(p)\}.$$

For the partial carrier c_K, we define

$$\begin{aligned} F_K(p) &= \min_{b: Q_i^s b > \beta(c_K)} \{F_{K-1}(p - (Q_i^s b - \beta(c_K)) + b, \\ &\qquad F_{K-1}(p)\}. \end{aligned}$$

The solution to the *Knapsack* formulation is the minimum value of p such that $F_K(p) \le Q_i^s$.

The running time of the above dynamic program is polynomial in C, the number of carriers, and $b_{\max} := \max_c\{r(i, c) - \beta(c)\}$. We can improve the running time by using the standard technique of scaling down each value of $r(i, c) - \beta(c)$ by a factor of $b_{\max}\varepsilon/C$ for some parameter ε. (See e.g. [14].) In this case we obtain a $(1 - \varepsilon)$-approximation to the *Knapsack* problem in time polynomial in C and $1/\varepsilon$. (This in turn reduces the approximation ratio of *MaxWeight-Alg4* by a factor $1 - \varepsilon$.)

6. MAXWEIGHT-ALG5: IMPROVED APPROXIMATION FOR OBJECTIVE (2)

In this section we show that for any $\varepsilon > 0$ it is actually possible to obtain a randomized $(1 - \frac{1}{e} - \varepsilon)$-approximation for objective (2). (Note that $1 - \frac{1}{e} = 0.632\ldots > \frac{1}{2}$.) The algorithm, which we call *MaxWeight-Alg5*, is based on a recent algorithm for the Generalized Assignment Problem (GAP) due to Fleischer et al. [12]. (In the GAP problem we are given a set of bins of different sizes. Each item has a bin-dependent profit and a bin-dependent size. The goal is to pack the items into bins so as to maximize the profit in such a way that no bin-size is violated.) *MaxWeight-Alg5* is somewhat complex, and so we feel that it is impractical for scheduling wireless systems. However, we include it here since we feel that it is of theoretical interest to understand what are the limits regarding the approximability of objective (2).

Let Λ_i be the set of all possible subsets of carriers that could be assigned to user i. For $S \in \Lambda_i$ let $f_i^S = Q_i^s \cdot \min\{\sum_{c\in S} r(i,c), Q_i^s\}$. For convenience we also calculate a new set of rates $r(i,c,S)$ such that $r(i,c,S) \leq r(i,c)$ and $f_i^S = Q_i^s \cdot \sum_{c\in S} r(i,c,S)$. This can clearly be done in linear time. The variable X_i^S is used to indicate whether or not subset S is assigned to user i. We could optimize objective (2) by solving the following integer program.

$$\begin{aligned}
&\textit{Alg5-IP} \qquad \max \sum_{i,S\in\Lambda_i} f_i^S X_i^S \\
&\text{s.t} \\
&\qquad \sum_{i\in U, S\in\Lambda_i : c\in S} X_i^S \leq 1 \quad \forall c \\
&\qquad \sum_{S\in\Lambda_i} X_i^S \leq 1 \quad \forall i \\
&\qquad X_i^S \in \{0,1\}.
\end{aligned}$$

Note that since we showed in Theorem 3 that optimizing objective (2) is NP hard, we cannot hope to solve the above integer program exactly. Algorithm *MaxWeight-Alg5* finds an approximate solution as follows. It first finds a solution to the linear relaxation of the above integer program in which the constraint $X_i^S \in \{0,1\}$ is replaced by $X_i^S \in [0,1]$. Note that we cannot directly use a standard linear programming algorithm for this relaxation since there are exponentially many variables. However, following [12] we can apply standard iterative Lagrangian LP algorithms (e.g. [13]) to obtain a $(1-\varepsilon)$-approximation to the linear relaxation of Alg5-IP in time polynomial in N, C and $1/\varepsilon$.

MaxWeight-Alg5 then rounds the solution to this linear relaxation by choosing a single set S_i to assign to user i. In particular, we set $S_i = S$ with probability X_S^i. We still do not have a valid assignment since a carrier c may be assigned to two different users. In this case we pick the user that gives the maximum value of $Q_i^s r(i,c,S_i)$.

THEOREM 12. *The value of the solution obtained by* MaxWeight-Alg5 *is at least a* $1 - \frac{1}{e} - \varepsilon$ *fraction of the optimal value of objective (2).*

PROOF. For each carrier c, we set (i_1, S_1) to be the user-set pair that has the highest value of $Q_i^s r(i,c,S)$. We set (i_2, S_2) to be the user-set pair that has the next highest value of $Q_i^s r(i,c,S)$ etc. The definition of our rounding algorithm means that carrier c is assigned to user i_k as part of S_k with probability at least $\prod_{k'<k}(1 - X_{i_{k'}}^{S_{k'}})X_{i_k}^{S_k}$.

The contribution of carrier c in the solution to the relaxation of *Alg5-IP* is $\sum_{i,S} Q_i^s r(i,c,S) X_i^S$. By the above argument the expected contribution in the rounded solution is at least,

$$\sum_k Q_{i_k}^s r(i_k, c, S_k) \prod_{k'<k} (1 - X_{i_{k'}}^{S_{k'}}) X_{i_k}^{S_k}.$$

Fleischer et al. [12] show, using the arithmetic/geometric mean inequality, that expressions of this form are at least,

$$(1 - \frac{1}{e}) \sum_{i,S} Q_i^s r(i,c,S) X_i^S.$$

Hence we have obtained an assignment whose value with respect to objective (2) is at least a $(1 - \frac{1}{e})$ fraction of the solution to the fractional relaxation. Since this is in turn at least a $(1-\varepsilon)$ fraction of the optimal solution to objective (2), our final approximation ratio is at least $(1-\frac{1}{e})(1-\varepsilon) > 1 - \frac{1}{e} - \varepsilon$. □

7. STABILITY OF FIVE VARIANTS OF MAXWEIGHT

In this section we consider the stability of the five variants of *MaxWeight* that we have proposed. Informally, an algorithm is said to be stable if it keeps the queue sizes bounded whenever this is achievable. More formally, we define stability as follows. Let $a_i(t)$ be the amount of data injected for user i in time slot t. We say that a system is (w,ε)-admissible if the adversary has a schedule $y(i,c,t)$ such that in any window $[t_0, t_0 + w)$, we have,

$$\sum_{t=t_0}^{t_0+w} r(i,c,t) y(i,c,t) \leq (1-\varepsilon) \sum_{t'=t}^{t+w} a_i(t) \quad \forall i.$$

We say that an algorithm is stable if it keeps the queues bounded for any (w,ε)-admissible.

The single-carrier *MaxWeight* algorithm is known to be stable as long as the channel rates for a user cannot be zero for arbitrarily long periods. (This condition holds for example when the rates are governed by a stationary stochastic process with non-zero mean.) The following theorem states that this property also holds for the five multi-carrier *MaxWeight* algorithms presented in this paper. The proof uses a standard technique (e.g. [26, 27]) of showing that the Lyapunov function $\sum_i (Q_i^s)^2$ has a negative drift when the queues are large. For this reason we defer the proof to the Appendix.

THEOREM 13. *If* $\varepsilon > 0$*, then algorithms MaxWeight-Alg1 through Alg5 are stable for any* (w,ε)*-admissible system as long as for each* i, c *the rates* $r(i,c,t)$ *cannot be zero for arbitrarily long periods.*

8. EXTENSIONS TO MORE GENERAL WEIGHT DEFINITIONS

In this section we show how to extend the definitions of algorithms *MaxWeight-Alg1* through *Alg5* in order to handle more general definitions of weight. There are many scheduling algorithms for single-carrier systems that always schedule the user that maximizes $W_i(t) r(i,t)$, for some weight W_i that is not necessarily equal to the queue size. This is important if we wish to achieve objectives such as fairness in addition to maintaining small queues. For example, the well-known Proportional Fair [28, 15] scheduler (that is used in the EV-DO Rev 0 system [8]) has this form and uses a weight that is the reciprocal of an estimate of the average service rate provided to user i. We define multi-carrier versions of our five algorithms as follows. Consider a time step t and let W_i^s be the weight at the beginning of the time step,

W_i^e be the weight at the end of the time step and W_i^c after carrier c has been assigned.

The changes to the first three algorithms are straightforward. They become,

- *MaxWeight-Alg1:*
  $\hat{i} = \text{argmax}_i W_i^s r(i,c)$.

- *MaxWeight-Alg2:*
  $\hat{i} = \text{argmax}_i W_i^s \min\{r(i,c), Q_i^c\}$.

- *MaxWeight-Alg3:*
  $\hat{i} = \text{argmax}_i W_i^c \min\{r(i,c), Q_i^c\}$.

Two adaptations are required for *MaxWeight-Alg4*. First, in the *Knapsack* problem defined in Section 5.1 we must change the objective from $\max \sum_c(\max\{0, Q_i^s b(i,c) - \beta(c)\})$ to $\max \sum_c(\max\{0, W_i^s b(i,c) - \beta(c)\})$. Second, whenever we (re)assign carrier c to user i we update $\beta(c)$ to $W_i^s b(i,c)\}$. For *MaxWeight-Alg5* a single change is required. We redefine f_i^S by $f_i^S = W_i^s \cdot \min\{\sum_{c\in S} r(i,c), Q_i^s\}$.

We can also provide new definitions of objectives (1) and (2). (We do not believe that objective (3) makes as much sense for general weights since it does not provide a clear separation between $Q_i^s(t)$ and $\mu_(i,t)$.) In particular, objective (1) becomes $\max \sum_i W_i^s(t)\mu(i,t)$ and objective (2) becomes $\max \sum_i W_i^s(t) \min\{Q_i^s(t), \mu(i,t)\}$. Each of our analyses for objectives (1) and (2) carry over directly to the case of more general weights. In particular, *MaxWeight-Alg1* optimally solves objective (1), *MaxWeight-Alg2* and *MaxWeight-Alg4* are $\frac{1}{2}$-approxi-mations for objective (2) and *MaxWeight-Alg5* is a $(1 - \frac{1}{e} - \varepsilon)$-approximation for objective (2).

9. SIMULATIONS

We analyze the performance of the first three *Max-Weight* algorithms in terms of queue sizes. We implement the algorithms in simple homegrown programs written in Python. We use a field trace that represents measured channel conditions from a third-generation wireless system as well as synthetic traces in which the channel rates fluctuate around a mean value according to 3km/h Rayleigh fading. We assume a constant rate arrival model. The number of users vary between 5 and 10, and the number of carriers vary between 4 and 8. In all cases *MaxWeight-Alg2* and *Alg3* have extremely similar performance and so we combine them onto a single plot. Both of these algorithms significantly outperform *MaxWeight-Alg1*. This is due to the fact that *MaxWeight-Alg1* is wasteful and often assigns more service to a user than it can actually use. See Figures 3 and 4 for total queue size plots under the simulated traces and Figure 5 for plots under the field trace. The figure captions offer summary statistics for the mean, 95th percentile and median queue sizes.

Recall that for a single time slot, the performance of *MaxWeight-Alg4* differs from the performance of algorithms *MaxWeight-Alg2* and *MaxWeight-Alg3* by at most a small constant factor. However, we now observe that this can lead to dramatic differences over longer timescales. In particular we present an example in which *MaxWeight-Alg4* outperforms *MaxWeight-Alg1* through *Alg3* in terms of queue sizes. In this example, we have an equal number of users and carriers. For the first half of the users the arrival rate is a parameter x and for the second half the arrival rate is a parameter $y > x$. We have channel rates $r(i,c) = x$ for

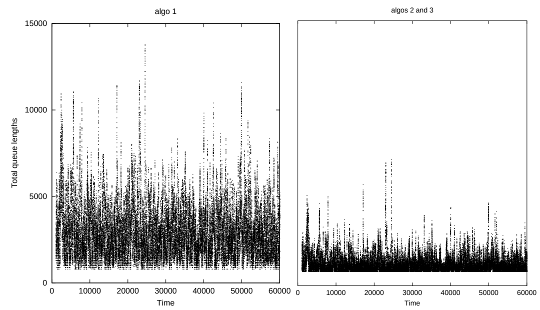

Figure 3: Simulated trace: total queue size. (Left) ***MaxWeight-Alg1*****. Mean: 3008, 95%: 5827, Median: 2744. (Right)** ***Alg2*** **and** ***Alg3*****. Mean: 1257, 95%: 2228, Median: 1104.**

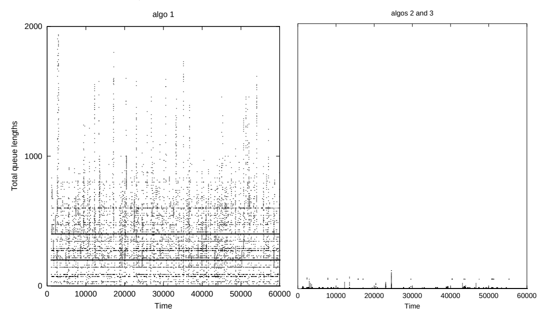

Figure 4: Simulated trace: total queue size. (Left) ***MaxWeight-Alg1*****. Mean: 151, 95%: 480, Median: 200. (Right)** ***Alg2*** **and** ***Alg3*****. Mean: 0.17, 95%: 0, Median: 0.**

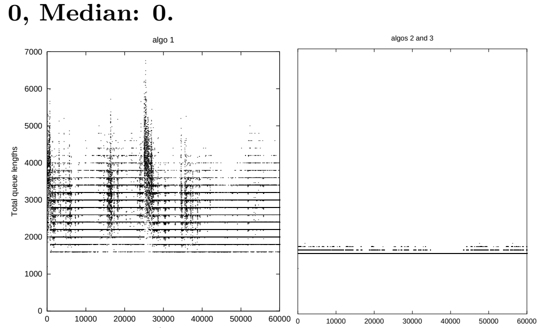

Figure 5: Field trace: total queue size. (Left) ***MaxWeight-Alg1*****. Mean: 2655, 95%: 3776, Median: 2600. (Right)** ***Alg2*** **and** ***Alg3*****. Mean: 1615, 95%: 1688, Median: 1600.**

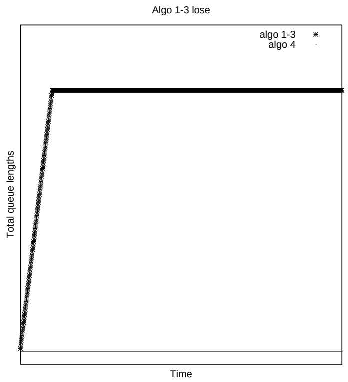

Figure 6: *MaxWeight-Alg4* outperforms *Alg1* through *Alg3*.

$c = i \le n/2$; $r(i,c) = y$ for $c = i > n/2$; and $r(i,c) = y$ for $c = i - n/2$. All other channel rates are zero. *MaxWeight-Alg4* manages to serve all arrivals during each time slot by having carrier c serve user $i = c$. *MaxWeight-Alg1* through *Alg3* let carrier $c \le n/2$ serve user $i = c+n/2$ since this gives a larger value of the objective. However, this prevents carriers $c > n/2$ from being of any use later on. Therefore, users $i \in [1, n/2]$ are initially not served by *Alg1* through *Alg3*. Once these $n/2$ users have built up large enough queues, every carrier c starts to serve user $i = c$. The queues do not grow further. However, the initial queues do not disappear. The build up equals y^2/x for each user $i \le n/2$. Figure 6 shows the total queue size during a simulation of this effect.

We can also create a similar example for which *MaxWeight-Alg1* through *Alg3* outperform *Alg4* in terms of queue lengths. The arrival rate is x for the first half of the users and y for the second half. We have channel rates $r(i,c) = y$ for $c = i \le n/2$; $r(i,c) = y$ for $c = i - n/2$; and $r(i,c) = x$ for $c = i + n/2$. *MaxWeight-Alg1* through *Alg3* can serve all arrivals during each timeslot; For *Alg4*, the carriers $c \le n/2$ serve users $c + n/2$. The curves for the resulting queue sizes are identical in shape to Figure 6 except that the labels of the two curves are flipped.

10. CONCLUSIONS

In this paper we studied a variety of scheduling algorithms for multi-carrier, frame-based wireless data systems. We presented a set of algorithms that aim to emulate the *MaxWeight* algorithm for the single carrier case.

A number of open problems remain. First, we would like to know if it is possible to improve on the $\frac{1}{3}$-approxi-mation for objective (3). In particular we would like to know if algorithm *MaxWeight-Alg3* has a better approximation ratio than $\frac{1}{3}$ since in the worst example that we can construct, the performance of *MaxWeight-Alg3* only differs from the optimum by a factor of $\frac{1}{2}$. We would also like to know if there is a simple algorithm that improves on the $\frac{1}{2}$-approximation for objective (2). We feel that *MaxWeight-Alg5* is probably too complex to be implemented in practice.

11. 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, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. CDMA data QoS scheduling on the forward link with variable channel conditions. *Bell Labs Technical Memorandum*, April 2000.

[4] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. Providing quality of service over a shared wireless link. *IEEE Communications Magazine*, February 2001.

[5] M. Andrews, L. Qian, and A. Stolyar. Optimal utility based multi-user throughput allocation subject to throughput constraints. In *Proceedings of IEEE INFOCOM '05*, 2005.

[6] M. Andrews and A. Slivkins. Oscillations with TCP-like flow control in networks of queues. In *Proceedings of IEEE INFOCOM '06*, 2006.

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

[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] C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 213–222, San Francisco, CA, January 2000.

[10] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length based scheduling and congestion control. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, March 2005.

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

[12] L. Fleischer, M. Goemans, V. Mirrokni, and M. Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 611–620, Miami, FL, 2006.

[13] N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 300 – 309, Palo Alto, CA, November 1998.

[14] D. Hochbaum. Various notions of approximations; Good, better, best, and more. In *Approximation algorithms for NP-hard problems, D. Hochbaum ed.*, pages 346–398, 1995.

[15] 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.

[16] Viggo Kann. Maximum bounded 3-dimensional

matching is max snp-complete. *Inf. Process. Lett.*, 37(1):27–35, 1991.

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

[18] X. Liu, E. Chong, and N.B.Shroff. Opportunistic transmission scheduling with resource-sharing constraints in wireless networks. *IEEE Journal on Selected Areas in Communications*, 19(10), 2001.

[19] X. Liu, E. Chong, and N.B.Shroff. A framework for opportunistic scheduling in wireless networks. *Computer Networks*, 41(4):451–474, 2003.

[20] M. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, March 2005.

[21] M. Neely, E. Modiano, and C. Rohrs. Power and server allocation in a multi-beam satellite with time varying channels. In *Proceedings of IEEE INFOCOM '02*, New York, NY, June 2002.

[22] S. Shakkottai and A. Stolyar. Scheduling algorithms for a mixture of real-time and non-real-time data in HDR. In *Proceedings of 17th International Teletraffic Congress (ITC-17)*, pages 793 – 804, Salvador da Bahia, Brazil, 2001.

[23] S. Shakkottai and A. Stolyar. Scheduling for multiple flows sharing a time-varying channel: The exponential rule. *Analytic Methods in Applied Probability*, 207:185 – 202, 2002.

[24] A. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, 50(4):401–457, 2005.

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

[26] 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.

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

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

APPENDIX

A. PROOF OF THEOREM 13

PROOF. We assume that the channel rates are bounded. Let $R^{\sup}$ be the supremum of these rates. For simplicity we also assume that the channel rates are bounded away from zero. Let $R^{\inf}$ be the infimum of these rates. By looking over larger time windows it is straightforward to adapt the argument to the case where channel rates can be zero but only for bounded periods of time. Consider the potential function $L(t) = \sum_i (Q_i^s(t))^2$. We first show that if the queues are sufficiently large then this potential function has negative drift for *MaxWeight-Alg1*. Recall that $x(i,c,t)(t) \in \{0,1\}$ indicates whether or not carrier c serves user i at time t. If $Q_i^s(t_0) \geq \sum_{t=t_0}^{t_0+w-1} \sum_c x(i,c,t) r(i,c,t)$, then

$$Q_i^s(t_0+w)^2 = \left(Q_i^s(t_0) + \sum_{t=t_0}^{t_0+w-1} a_i(t) - \sum_{t=t_0}^{t_0+w-1} \sum_c x(i,c,t) r(i,c,t) \right)^2,$$

and otherwise

$$\begin{aligned} & Q_i^s(t_0+w)^2 \\ \leq & \left(Q_i^s(t_0) + \sum_{t=t_0}^{t_0+w-1} a_i(t) \right)^2 \\ \leq & \left(\sum_{t=t_0}^{t_0+w-1} x(i,c,t) r(i,c,t) + \sum_{t=t_0}^{t_0+w-1} a_i(t) \right)^2 . \end{aligned}$$

In both cases this implies,

$$\begin{aligned} & L(t_0+w) - L(t_0) \\ =& \sum_i Q_i^s(t_0+w)^2 - \sum_i Q_i^s(t_0)^2 \\ \leq& 2\sum_i \left(\sum_t a_i(t) \right)^2 \\ &+2\sum_i \left(\sum_t \sum_c x(i,c,t) r(i,c,t) \right)^2 \\ &+2\sum_i Q_i^s(t_0) \left(\sum_t a_i(t) - \sum_t \sum_c x(i,c,t) r(i,c,t) \right) \end{aligned} \tag{7}$$

Since the arrival process (w,ε)-admissible, the total arrivals per user is upper bounded by a function of $R^{\sup}$, C, w and ε. In particular,

$$\begin{aligned} \sum_{t=t_0}^{t_0+w} a_i(t) &\leq (1-\varepsilon) \sum_{t=t_0}^{t_0+w} \sum_c r(i,c,t) y(i,c,t) \\ &\leq (1-\varepsilon) w C R^{\sup}. \end{aligned}$$

Similarly the second term can be upper bounded by a function of $R^{\sup}$, C, N and w. Let c_1 denote an upper bound of the first two terms. To bound the third term, we note that for $t_0 \leq t \leq t_0 + w$,

$$Q_i^s(t) - wCR^{\sup} \leq q_i(t_0) \leq q_i(t) + wCR^{\sup}.$$

Hence,

$$\begin{aligned} & L(t_0+w) - L(t_0) \\ \leq & c_1 + 2wCR^{\sup} \sum_i \sum_t \left(a_i(t) + \sum_c x(i,c,t) r(i,c,t) \right) \\ & +2 \sum_t \sum_i \sum_c (q_i(t)(1-\varepsilon) y(i,c,t) r(i,c,t) \\ & \qquad -q_i(t) x(i,c,t) r_(i,c,t)) \end{aligned}$$

Again, the second term of the above expression can be upper bounded by a function of $R^{\sup}$, C and w. Let c_2 be this constant. To bound the third term, let $\hat{i} = \text{argmax}_i Q_i^s(t) r(i,c,t)$ for given time step t and carrier j. The definition of max weight implies $x(\hat{i}, c, t) = 1$ and $x(i, c, t) = 0$ for all $i \neq \hat{i}$. Therefore,

$$\sum_{t,i,c} Q_i^s(t) r(i,c,t) x(i,c,t) \geq \sum_{t,i,c} Q_i^s(t) r(i,c,t) y(i,c,t).$$

Given t and c, let $k = \text{argmax}_i Q_i^s(t)$. In other words $Q_k^s(t) = Q_{\max}(t)$. Therefore, given t and c, we have

$$\sum_i Q_i^s(t) r(i,c,t) x(i,c,t) \geq Q_k^s(t) r(k,c,t) \geq Q_{\max}(t) R^{\inf}.$$

This implies,

$$\begin{aligned}
& L(t_0 + w) - L(t_0) \\
\leq \quad & c_1 + c_2 - 2 \sum_{t,i,j} \varepsilon Q_i^s(t) x(i,c,t) r(i,c,t) \\
\leq \quad & c_1 + c_2 - 2\varepsilon R^{\inf} \sum_{t,c} Q_{\max}(t) \\
= \quad & c_1 + c_2 - 2\varepsilon R^{\inf} C \sum_t Q_{\max}(t)
\end{aligned}$$

Hence when the queues are sufficiently large, i.e. larger than $\sum_{t=t_0}^{t_0+w} (c_1 + c_2)/(2\varepsilon R^{\inf} C)$, the potential function decreases. This implies stability for *MaxWeight-Alg1*.

For algorithms *MaxWeight-Alg2*, *Alg4* and *Alg5*, stability follows from a similar argument to the above and the fact that if $Q_i^s \geq C R^{\sup}$ then each of these algorithms assigns each carrier c to user $\arg\max_i Q_i^s(t) r(i,c,t)$. Hence in all cases,

$$\begin{aligned}
& \sum_{t,i,c} Q_i^s(t) r(i,c,t) x(i,c,t) \\
\geq \quad & \sum_{t,i,c} (Q_i^s(t) - C R^{sup}) r(i,c,t) y(i,c,t) \\
\geq \quad & \sum_{t,i,c} Q_i^s(t) r(i,c,t) y(i,c,t) - c_3,
\end{aligned}$$

where c_3 is a function of $R^{\sup}$, C and w.

Similarly, if $Q_i^s \geq C R^{\sup}$ then algorithm *MaxWeight-Alg3* assigns carrier c to user $\arg\max_i (Q_i^s(t) - B) r(i,c,t)$ for some $B \leq (c-1) R^{\sup}$. Therefore we once again have,

$$\sum_{t,i,c} Q_i^s(t) r(i,c,t) x(i,c,t) \geq$$
$$\sum_{t,i,c} (Q_i^s(t) - C R^{sup}) r(i,c,t) y(i,c,t).$$

□