This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2008 proceedings.

Proportional Fairness in Multi-rate Wireless LANs

Li (Erran) Li* Martin Pal† Yang Richard Yang§

*Bell Labs, Alcatel-Lucent †Google §Yale University

***Abstract*—In multi-rate wireless LANs, throughput-based fair bandwidth allocation can lead to drastically reduced aggregate throughput. To balance aggregate throughput while serving users in a fair manner, proportional fair or time-based fair scheduling has been proposed to apply at each access point (AP). However, since a realistic deployment of wireless LANs can consist of a network of APs, this paper considers proportional fairness in this much wider setting. Our technique is to intelligently associate users with APs to achieve optimal proportional fairness in a network of APs. We propose two approximation algorithms for periodical offline optimization. Our algorithms are the first approximation algorithms in the literature with a tight worst-case guarantee for the NP-hard problem. Our simulation results demonstrate that our algorithms can obtain an aggregate throughput which can be as much as 2.3 times more than that of the max-min fair allocation in 802.11b. While maintaining aggregate throughput, our approximation algorithms outperform the default user-AP association method in the 802.11b standard significantly in terms of fairness.**

I. Introduction

Fair and efficient medium access is a fundamental problem in wireless networks. This problem is particularly challenging to address in modern WLAN networks due to the introduction of multi-rate WLANs to accommodate users with diverse channel conditions. Recent measurement studies have shown that rate diversity is prevalent in operational WLANs [15].

In the case of a single access point supporting multi-rate, the tradeoff between fairness and efficiency is resolved by the introduction of the proportional fair scheduler. Specifically, the widely used 802.11 MAC provides equal long-term transmission opportunities. This is also referred to as throughput-based fairness. A consequence of this fairness is that nodes with lower data rates occupy the medium a larger percentage of time than those with higher data rates, leading to drastically reduced network throughput [10], [15]. In contrast, proportional fairness implements time-based fairness and provides a good tradeoff between fairness and network throughput.

However, many deployed WLANs consist of multiple APs, interconnected by a wired backbone network, providing overlapping coverage. In this much wider network-based setting, the client nodes are often distributed unevenly among the access points [6], [7], [12], and a node can be associated with one of multiple APs. In other words, a significant challenge in this more realistic setting is association control. Association control plays a major role in determining not only fairness but also efficiency and load balancing [5]. As researchers pointed out in a previous study [9], association control without consideration of fairness can lead to *non-Pareto optimal* channel capacity allocation. By a Pareto optimal allocation it means one such that there does not exist another feasible allocation where at least one node gets more bandwidth, and all others get at least the same bandwidth.

In this paper, we study how to use association control to achieve optimal proportional fairness; that is, how to associate the nodes to maximize the sum of the logarithm of the rates allocated to users. The logarithm of a rate can be viewed as a utility function of each user.

Since the objective function of proportional fairness is non-linear and non-concave, implementing optimal proportional fairness is much more challenging than the max-min fairness problem addressed in [8]. If each user can be associated with only one AP, the problem of achieving optimal proportional fairness then is NP-hard. We present two algorithms allowing tradeoffs between performance and computational speed. In our first algorithm, we solve a relaxed convex program to obtain a fractional association. We then round this solution to an integral solution. The total utility of the bandwidth allocation vector given by our algorithm is greater than the that of optimal bandwidth allocation vector scaled down by a factor of 5.828. In our second algorithm, we first discretize the nonlinear program to obtain a linear program relaxation of the problem where each user can be associated with multiple APs simultaneously. Via rounding the solution to the discretized linear program, we design an efficient approximation algorithm such that, the total utility of the bandwidth allocation vector given by our algorithm is greater than the that of optimal bandwidth allocation vector scaled down by a factor of $2 + \epsilon$.

As far as we know, the only closely related paper in the recent theory literature is [4]. Azar and Epstein [4] give algorithms for minimizing the L_p norm of the load vector in the problem of scheduling unrelated parallel machines. Similar to our work, they use the Generalized Assignment technique of Shmoys and Tardos [14] to round a relaxation of the problem formulation. *Thus, to the best of our knowledge, we are the first to present association control algorithms that provide guarantees on the quality of bandwidth allocation with respect to the optimal proportional fairness solution.*

Our approximation algorithms can be used for periodic offline optimization. The function can be implemented in a central management server (e.g., Cisco's Wireless Control System [2]) to allow enterprise wireless networks to be designed, controlled, and monitored from a centralized location.

We conduct extensive evaluations to demonstrate the effectiveness of our algorithms. We show that our approximation algorithm achieves close-to-optimal proportional bandwidth allocation (we compare with an upper bound of the optimal objective function since the problem is NP-hard), obtains an aggregate throughput which can be as much as 2.3 time more than that of the max-min fairness allocation, and outperforms the default user-AP association method in the 802.11 standard. We also find that our convex programming based algorithm runs much faster than our second discretized linear programming formulation.

The rest of this paper is organized as follows. We motivate our design in Section II. We introduce our notations and formulation in Section III. We present our two algorithms in Sections IV and V, respectively. We evaluate our algorithms in Section VI. After presenting the related work in Section VII, we conclude in Section VIII.

978-1-4244-2026-1/08/$25.00

II. Motivation

We motivate our design choices using an example network shown in Fig. 1. The network has 2 APs a and b, and 3 users indexed by 1, 2, and 3. The dashed lines show the possible links. The number besides each link is its bit rate. We assume that all users have the same priority.

A. Fairness

Two fairness definitions are widely used in network resource allocation. Proportional fairness allocates bandwidth to users in proportion to their data rates in the single AP case. It maximizes the sum of the logarithms of the allocated bandwidth of each user in the multiple-AP case. Max-min fairness, on the other hand, tries to allocate bandwidth equally among users to the extent possible. That is, a bandwidth allocation is max-min fair if there is no way to give more bandwidth to any user without decreasing the allocation of a user with less or equal bandwidth.

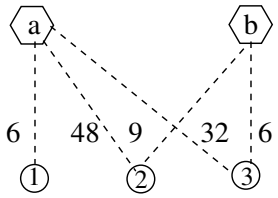


Fig. 1. An example network (the number besides each link is its bit rate).

Applying the two fairness definitions to the example network shown in Fig. 1, we can verify by exhaustive search that the following assignment of users to APs satisfies both fairness definitions: user 1 and 2 to a, and 3 to b.

However, the two fairness definitions allocate different amounts of bandwidth to different users and achieve different overall network throughput. Specifically, proportional fairness allocates 3, 24 and 6 units of bandwidth to users 1, 2 and 3 respectively by giving users 1 and 2 equal transmission time. As a comparison, max-min fairness allocates 16/3, 16/3 and 6 units to users 1, 2 and 3 respectively by giving user 1 longer transmission time (8 times more than that given to user 2). Comparing the total network throughput, we observe that max-min fairness achieves only 16.67, far less than 33, the total throughput achieved by the proportional fair allocation.

This example also can illustrate that fairness must be considered in a broader context than a per-AP basis. Without this, the bandwidth allocation can be quite inefficient. For example, consider a fixed assignment of user 1 to a, 2 and 3 to b. This allocation gives the user an allocation of 6, 4.5, and 3 which results in a total bandwidth of 13.5, far less than the optimal of 33.

Note that in an extremely large wireless LAN deployment, we may partition the network into several components. In each component, we apply our proportional fairness allocation. This may have fairness implications, but maybe a reasonable tradeoff in achieving feasible implementation.

B. Load balancing

Fairness and load balancing are traditionally considered in separation. In our framework, the resources at all access points are considered as a whole when allocating bandwidth fairly to users. With this network-wide fairness objective, load balancing is automatically taken care of.

C. Periodic offline optimization

The network has to make association decisions as users arrive. Therefore, online association algorithms are necessary. However, online decisions may become inefficient over time both in terms of aggregate throughput and fairness. Thus, our system is to combine an online algorithm with periodic offline optimization.

An online algorithm has to make a decision without knowing the future. Because of this, it can be arbitrarily unfair to certain users. For example, suppose we have two APs a, b and n users. Suppose only user 1 is able to communicate with both APs and the data rate is 48, and 24 to a and b respectively. The rest of the users are all outside the coverage of AP b, but are within the coverage of AP a. When user 1 first arrives, an online algorithm will associate the user with AP a. Then the rest of $n-1$ users arrive. Since an online algorithm will not re-associate users, the first user will get $48/n$. However, if user 1 associates with AP b instead, it will get 24. We remark that this worst case occurs for any online algorithm with a simple adversarial analysis. Suppose an online algorithm assigns the first user to AP b when it first arrives. The adversary can place the rest of $n-1$ users to AP b which result in the same situation.

Due to the inefficiency in terms of aggregate throughput and fairness of any online algorithm, our system architecture is to combine an online algorithm with periodic offline optimization.

III. Notations and Formulation

Our notation is summarized in Table I. We consider an IEEE 802.11 WLAN consisting of multiple access points (APs). We denote the set of APs as A indexed by $1, \ldots, m$. All the APs are attached to a fixed infrastructure which connects them to wired data networks such as the Internet. The backhaul capacity for each AP may be limited. Each AP has a limited coverage and it can serve only users that reside in its coverage area. The network coverage area is the union of the coverage area of each AP. We assume adequate frequency planning where interfering APs are assigned orthogonal channels. Our algorithms can be easily extended to the case where interfering APs share the same channel resource equally.

Symbol	Semantics
A	The set of all access points (APs).
m	The number of APs, i.e. $m = \|A\|$.
U	The set of all users.
n	The number of users, i.e. $n = \|U\|$.
C_i	The infrastructure link bit rate of AP i.
r_{ij}	The wireless link bit rate between AP i and user j.
p_{ij}	The fraction of time that AP i allocated to user j.
w_j	The weight (priority) of user j.
b_j	The bandwidth allocated to user j.
x_{ij}	The fractional association of user j with AP i.
x	An user-AP association matrix.
p	An user-AP time allocation matrix.
$y_{ij\tau}$	1 if AP j allocates τ time slots to i
u_{ij}	The utility user j gets from AP i.
u	An user-AP utility matrix.

TABLE I
Notations.

We denote the set of users as U indexed by $1, \ldots, n$. For each user–AP pair j, i, we assume that we know the *effective bit rate* r_{ij} of the link between j and i. The effective bit rate is measured in a longer time scale which tracks the long-term channel condition (mainly influenced by path loss and

slow fading). The effective bit rate also takes into account the overhead of retransmissions due to reception errors. We assume that a user consumes all bandwidth allocated to it by the network and always has traffic to send or receive. Furthermore, we assume that each user j has a weight w_j that specifies its priority. This weight is used to determine the channel occupancy time it entitles to have with respect to the other users.

We assume that each user communicates directly only with a single AP for an extended period of time. In other words, each user must be assigned to only one AP. Each AP, on the other hand, may serve multiple users. We formulate the problem of associating users with APs as an optimization problem. Due to the dynamic nature of the channel condition between an AP and a user, as well as the bursty nature of data traffic, rate allocation through association control should be done in a longer time scale. The alternative would entail a much higher communication overhead and adversely impact ongoing traffic flows. Therefore, our association control decisions are based on the effective bit rate of user-AP pairs, not the instantaneous rate.

In our formulation, we use two sets of variables. For each user-AP pair, we shall have a binary variable x_{ij} that is equal to 1 if client j is associated with access point i; 0 otherwise. We express the fraction of time each AP devotes to each of its users using a fractional variable p_{ij}. For each AP i and user j, if j is associated with i, then p_{ij} is a fraction between 0 and 1 that specifies how much time i spends communicating with j. If client j is not associated with i, then the fraction p_{ij} is 0. We denote the x_{ij} matrix by x and the p_{ij} matrix by p. It is easy to verify that the product p_{ij} or $x_{ij}p_{ij}$ accurately reflects the fraction of time access point i spends talking to client i. For convenience, we use $x_{ij}p_{ij}$ in our second algorithm. The bandwidth b_j allocated to each user j is given as follows. $b_j = \sum_{i\in A}(r_{ij}\, x_{ij} p_{ij}) = \sum_{i\in A}(r_{ij}\, p_{ij})$

Our goal is to construct an assignment of clients to APs in a proportional fair manner; the assignment allows each user sufficient bandwidth without unduly restricting the amount of bandwidth available to other users. Formally, we would like to maximize the weighted sum of the logarithm of the bandwidth allocation [11]:

$$f_O(x,p) = \sum_{j\in U} w_j \log b_j. \tag{1}$$

Note that, we assume no isolated users. That is, for each user j, at least one of its $r_{ij} > 0$. Therefore, the sum inside the log function is strictly greater than zero. This can be ensured easily by pruning the isolated users. We can show that the problem is NP-hard by slightly adapting the reduction in [9].

IV. A 5.828 APPROXIMATION ALGORITHM VIA CONVEX PROGRAM RELAXATION

We first present a convex program relaxation formulation of the problem. We then show how we can obtain an approximation algorithm through rounding.

A. Convex program relaxation

Since each user j is assigned to only one AP (integral association), there is exactly one non-zero p_{ij}, for $i \in A$. We now relax this constraint and obtain the following convex program formulation:

Algorithm cvapPF(U, A)

1. p_{ij} = solve_CVP($\{r_{ij}\}$, $\{w_j\}$)
2. $\forall i \in A,\ j \in U$
   set $p'_{ij} = p'_{ij}$ if (i,j) is a strong edge; 0 otherwise
   $b'_j = \sum_{i\in A} r_{ij} p'_{ij}$
   $x'_{ij} = \frac{r_{ij} p'_{ij}}{b'_j}$
   $q_{ij} = \frac{b'_j}{r_{ij}}$
3. Set up the Generalized Assignment problem using (x', q)
4. $\hat{x}$ = roundGAP(q, x')
5. $\hat{p}_{ij} = \frac{\hat{x}_{ij} b'_j}{(2+\sqrt{2}) r_{ij}}$

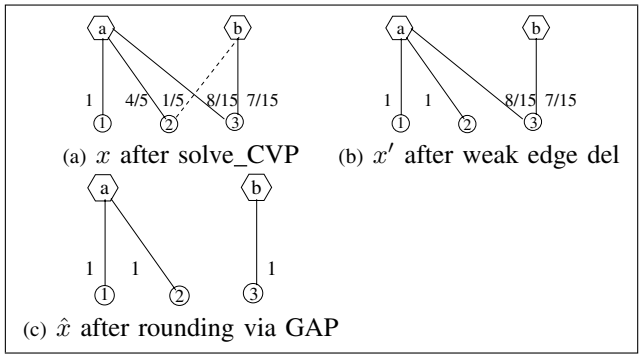
Fig. 2. A formal description of our proportional fair algorithm

$$\begin{aligned}
&\text{maximize} && \sum_{j\in U} w_j \log b_j \\
&\forall j \in U: && b_j = \sum_{i\in A} r_{ij} p_{ij} \\
&\forall i \in A: && \sum_{j\in U} p_{ij} \le 1 \\
&\forall j \in U: && \sum_{i\in A} p_{ij} \le 1 \\
&\forall i \in A, j \in U: && 0 \le p_{ij} \le 1.
\end{aligned} \tag{CVP}$$

The first constraint is an expression for the auxiliary variable b_j, the bandwidth allocated to each user j. The second constraint says that the total allocated time fraction of each AP i can not be more than 1. The third constraint says that the total fraction of time each user j communicates with all AP can not be more than 1.

Convex program can be solved to the desired precision in polynomial time. We first solve problem CVP and obtain p_{ij}, which induces a fractional assignment x_{ij}, where $x_{ij} = \frac{r_{ij} p_{ij}}{b_j}$. We can view the assignment as a bipartite graph. We now define two types of edges. An edge (i,j) is weak if $b_j > \alpha r_{ij}$; otherwise, it is called a strong edge. $\alpha = \sqrt{2}+1$ is a parameter which will be determined later. Our algorithm is summarized in Fig. 2.

We use the example in Fig. 1 to illustrate the key steps of our algorithm. Let's index AP a and b by 1 and 2. Applying cvapPF, we obtain a fraction p_{ij} as follows: $p_{11} = 1/2$, $p_{12} = 2/5$, $p_{13} = 1/10$, $p_{22} = 8/15$, $p_{23} = 7/15$. We have $b_1 = 3$, $b_2 = 24$, $b_3 = 6$. The induced x_{ij} is shown in Fig. 3-a. Note that, edge $(2,2)$ is weak. This is because $b_2 = 24 > (\sqrt{2}+1) * 9$.



(a) x after solve_CVP (b) x' after weak edge del

(c) $\hat{x}$ after rounding via GAP

Fig. 3. An example: assignments at each step.

B. Rounding via generalized assignment problem (GAP)

The fractional assignment x_{ij} yields too many assignments (edges). Our goal is to derive an integer assignment. We first delete all weak edges. This induces another fractional assignment p'_{ij}. That is,

$$p'_{ij} = \begin{array}{l} p_{ij}, \text{if (i,j) is strong;} \\ 0, \text{if (i,j) is weak.} \end{array} \quad (2)$$

Define $b'_j = \sum_{i \in A} r_{ij} p'_{ij}$. We have the following lemma:

Lemma 4.1: For a given user j, $b'_j \geq (1 - \frac{1}{\alpha}) b_j$.

Proof: Let S be a subset of APs A such that, $\forall i \in S$, (i, j) is a strong edge.

$$b_j = b'_j + \sum_{i \in A - S} r_{ij} p_{ij}.$$

By definition of weak edge,

$$\sum_{i \in A - S} r_{ij} p_{ij} \leq \frac{b_j}{\alpha} \sum_{i \in A - S} p_{ij}.$$

Since $\sum_{i \in A-S} p_{ij} \leq 1$, we have that the bandwidth contributed by all weak edges is no more than $\frac{b_j}{\alpha}$. Thus, $b'_j \geq (1 - \frac{1}{\alpha}) b_j$. ■

Define $x'_{ij} = \frac{r_{ij} p'_{ij}}{b'_j}$. By definition of x'_{ij} and b'_j, we have $\sum_{i \in A} x'_{ij} = 1$. Since $\sum_{i \in U} p'_{ij} \leq 1$, substitute p'_{ij} by $x'_{ij} \frac{b'_j}{r_{ij}}$, we have $\sum_{i \in U} \frac{b'_j}{r_{ij}} x'_{ij} \leq 1$.

Thus, we get the following assignment problem:

$$\begin{array}{ll} \forall i \in U: & \sum_{i \in A} x'_{ij} = 1; \\ \forall i \in A: & \sum_{j \in U} \frac{b'_j}{r_{ij}} x'_{ij} \leq 1. \end{array} \quad \text{(GAP-C)}$$

We have the following lemma:

Lemma 4.2: If $x'_{ij} > 0$, then $\frac{b'_j}{r_{ij}} \leq \alpha$.

Proof: By definition of x'_{ij}, $p'_{ij} > 0$. By definition of $p'_{ij} > 0$, edge (i, j) is strong. Per its definition, $\frac{b'_j}{r_{ij}} \leq \alpha$. ■

Using our example, after deleting weak edge $(2, 2)$, we obtain x'_{ij} as shown in Fig. 3-b.

Using the rounding procedure of Shmoys and Tardos [14], we get an integral solution $\hat{x_{ij}}$ such that,

$$\begin{array}{ll} \forall j \in U: & \sum_{i \in A} \hat{x}_{ij} = 1; \\ \forall i \in A: & \sum_{j \in U} \frac{b'_j}{r_{ij}} \hat{x}_{ij} \leq 1 + \alpha. \end{array} \quad (3)$$

The rounding procedure has to cleverly reassign users that are fractionally assigned to many machines each to a single machine, while not reducing the overall value of the solution obtained (we do not have an explicit objective function here; we will see one in our next algorithm). Shmoys and Tardos achieve this by carefully setting up a bipartite min-cost flow problem, whose optimum solution corresponds to a suitable integer assignment. In this process, the procedure may assign an extra job to each machine, thus exceeding the capacity of each machine by the load imposed by the extra job. For detailed description of the rounding procedure, we refer the reader to the original source [14].

Define

$$\hat{p}_{ij} = \frac{\hat{x}_{ij} b'_j}{(1 + \alpha) r_{ij}}. \quad (4)$$

Denote $\hat{b}_j$ the bandwidth allocated by $\hat{p}_{ij}$. We have the following lemma:

Lemma 4.3: $\hat{b}_j \geq \frac{1}{1+\alpha}(1 - \frac{1}{\alpha}) b_j$.

Proof: By definition, $\hat{b}_j = \sum_{i \in A} r_{ij} \hat{p}_{ij}$. By definition of $\hat{p}_{ij}$,

$$\begin{array}{rl} \hat{b}_j = & \sum_{i \in A} r_{ij} \frac{\hat{x}_{ij} b'_j}{(1+\alpha) r_{ij}} \\ = & \frac{b'_j}{1+\alpha} \sum_{i \in A} \hat{x}_{ij} \\ = & \frac{b'_j}{1+\alpha} \\ \geq & \frac{b_j}{1+\alpha}(1 - \frac{1}{\alpha}). \end{array} \quad (5)$$

■

After this step, our example yields the integral association $\hat{x}$ as shown in 3-c.

C. Analysis

We would like our integral assignment to achieve a bandwidth as close to the optimal as possible. Thus, we try to find the best parameter α. It is easy to see that $\alpha = \sqrt{2} + 1$ is the best parameter. We have the following theorem:

Theorem 4.1: Consider any optimal integral association (x^*, p^*). Then, for the solution $(\hat{x}, \hat{p})$ produced by our algorithm it holds that

$$f_O(x^*, p^*/(3 + 2\sqrt{2})) \leq f_O(\hat{x}, \hat{p}).$$

Proof:

$$\begin{array}{rl} f_O(\hat{x}, \hat{p}) = & \sum_{j \in U} w_j \log \hat{b}_j \\ \geq & \sum_{j \in U} w_j \log b_j / (3 + 2\sqrt{2}) \\ = & \sum_{j \in U} w_j \log b_j - w_j n \log(3 + 2\sqrt{2}) \\ \geq & \sum_{j \in U} w_j \log r_{ij} p^*_{ij} - w_j n \log(3 + 2\sqrt{2}) \\ = & f_O(x^*, p^*/(3 + 2\sqrt{2})). \end{array} \quad (6)$$

Note that, the second inequality is due to the fact that the convex program solution achieves a better objective function value than that of any optimal integral solution. ■

V. A $2 + \epsilon$ APPROXIMATION ALGORITHM VIA DISCRETIZED LINEAR PROGRAM RELAXATION

In this section, we first present an exact non-linear program formulation. We then discretize the non-linear formulation to obtain a linear formulation. We establish the relationship between the solutions of the two programs. We show how we can make use of the linear formulation to obtain an efficient approximation algorithm.

A. Non-linear exact formulation

Because x_{ij} takes either 0 or 1 and there is exactly one $x_{ij} = 1$ for each user j (denote such a specific AP as i'), we have $\log \sum_{i \in A}(r_{ij} x_{ij} p_{ij}) = \log(r_{i'j} p_{i'j})$; similarly, $\sum_{i \in A} x_{ij} \log(r_{ij} p_{ij}) = log(r_{i'j} p_{i'j})$. So the log of sum equals to the sum of log in our specific setting (note that in general this is not the case). Thus, the objective function in Eq. (1) is equivalent to the following function:

$$f_N(x, p) = \sum_{j \in U} \sum_{i \in A} x_{ij} \, w_j \, \log(r_{ij} \, p_{ij}). \quad (7)$$

To simplify our notation, we allow r_{ij} to be zero. When r_{ij} is 0, x_{ij} will be set to 0. Note that, by convention, $0 \, log(0)$ is

defined to be 0. In practice, when $r_{ij} = 0$, the corresponding terms in the objective function and the constraints will be pruned before running any mathematical program solver.

We now formulate the multi-AP proportionally fair association problem as the following mathematical program. The non-linear program is referred as NLP.

$$\begin{array}{lr} \text{maximize} & \sum_{j\in U}\sum_{i\in A} x_{ij}\, w_j\, \log(r_{ij}\, p_{ij}) \\ \forall j \in U: & \sum_{i\in A} x_{ij} = 1 \\ \forall i \in A: & \sum_{j\in U} x_{ij} p_{ij} \le 1 \\ \forall i \in A, j \in U: & 0 \le p_{ij} \le 1 \\ \forall i \in A, j \in U: & x_{ij} \in \{0,1\}. \end{array} \quad \text{(NLP)}$$

The first and fourth sets of constraints specify that each user must be assigned to exactly one access point. The second set of constraints expresses the fact that each AP must share its available time among its assigned users to the extent of at most 1. The third set of constraints states that no user can get more than 100% of an AP's time share.

The mathematical program (NLP) is difficult to solve because of two reasons. First, it imposes integrality constraints on the x_{ij} variables. Second, even if we disregard the integrality constraints, the problem is non-linear and non-concave.

In the case of a single AP, proportional fairness calls for the access point to divide its time among its users equally if all users have the same priority, or in proportion to their assigned priorities. Using the Lagrangian methods, we can check that our formulation satisfies this property. We state the following lemma for use by later proofs.

Lemma 5.1: If the number of APs $m = 1$, then $p_{1j} = \frac{w_j}{\sum_{j'\in U} w_{j'}}$ gives the unique optimal solution to our NLP formulation where U is the set of users associated with AP 1.

Thus, given a fixed association matrix x which determines which AP a user associates, we know how much time p_{ij} user j gets from AP i. By Lemma 5.1, it is $p_{ij} = x_{ij}w_j / \sum_{j'\in U}(x_{ij'}w_{j'})$. If w_j is the same for all j, then all users associated with a given AP i gets equal time share from i.

B. Discretized linear formulation

As the first step in solving the NLP formulation, we start by discretizing the scheduling period of each AP into D discrete intervals. To achieve a theoretical approximation guarantee, we have to pick a possibly larger D, where $D = (1+\epsilon)\frac{\sum_{j\in U} w_j}{\min\{w_j | j\in U\}}$ turns out to be sufficient. We introduce a new indicator variable $y_{ij\tau}$ that is equal to 1 if and only if user j is assigned to access point i, and access point i has allocated τ (out of D) of its time slots to user j. Note that it is τ number of slots out of D, not the τ-th slot. Here is the discretized formulation. The variable τ always ranges from 1 to D.

$$\begin{array}{lr} \text{maximize} & \sum_{i\in A}\sum_{j\in U}\sum_{\tau=1}^{D} y_{ij\tau}\, w_j\, \log(r_{ij}\, \frac{\tau}{D}) \\ \forall j \in U: & \sum_{i\in A}\sum_{\tau=1}^{D} y_{ij\tau} = 1 \\ \forall i \in A: & \sum_{j\in U}\sum_{\tau=1}^{D} y_{ij\tau}\frac{\tau}{D} \le 1 \\ \forall i \in A, j \in U: & y_{ij\tau} \in \{0,1\}. \end{array} \quad \text{(DLP)}$$

With the integral constraint on $y_{ij\tau}$ (third one in DLP formulation), the first constraint makes sure that there is one and only one $y_{ij\tau}$ equals to 1 for each user j. The second constraint ensures that the total fraction of time allocated by an AP i is no greater than 1. In the following discussion, we use f_D to denote the objective function of the DLP program. That is,

$$f_D(y) = \sum_{i\in A}\sum_{j\in U}\sum_{\tau=1}^{D} y_{ij\tau}\, w_j\, \log(r_{ij}\, \frac{\tau}{D}). \quad (8)$$

Recall that previously we have defined f_N in Eq. (7). The following lemma establishes a tight relationship between the two formulations.

Lemma 5.2: For every integral solution (x, p) to the NLP formulation, if we were to scale down all time allocations in the former formulation by a factor $1+\epsilon$, the latter formulation would yield a solution that is at least as good: $f_N(x, p/(1+\epsilon)) \le f_D(y)$.

Proof: Without loss of generality, we can assume that (x, p) is an optimal solution. By Lemma 5.1, the optimal p for any fixed integral assignment x is the weighted time fair allocation $p_{ij} = x_{ij}w_j / \sum_{j'\in U}(x_{ij'}w_{j'})$.

Define an integral solution y to the formulation (DLP) as follows. For each pair (i, j), define $y_{ijp} = 1$ and $\tau = \lfloor p_{ij}\, D \rfloor$ if $x_{ij} = 1$; $y_{ijp} = 0$, otherwise. For the Lemma to be true, all we need is $1 \ge \lfloor p_{ij}\, D \rfloor \ge \frac{p_{ij}\, D}{1+\epsilon}$, $\forall j \in U, i \in A$ if $p_{ij} > 0$. If $p_{ij} > 0$, then $p_{ij} \ge \frac{w_j}{\sum_{j'\in U} w_{j'}}$. Thus, picking $D = (1+\epsilon)\frac{\sum_{j\in U} w_j}{\min\{w_j | j\in U\}}$ suffices. ∎

For the opposite direction, we need only concern ourselves with translating a fractional solution to (DLP) to a fractional solution to (NLP).

For every i, j, define

$$x_{ij} := \sum_{\tau=1}^{D} y_{ij\tau} \quad (9)$$

$$p_{ij} := \frac{\sum_{\tau=1}^{D} \tau/D\, y_{ij\tau}}{x_{ij}} \quad (10)$$

$$u_{ij} := w_j \log r_{ij} p_{ij}. \quad (11)$$

In cases where $x_{ij} = 0$, we can define e.g. $p_{ij} = 0$ and $u_{ij} = -\infty$. We have:

Lemma 5.3: For every fractional solution y to the formulation (DLP), the preceding is a fractional solution (x, p) to the formulation (NLP) such that $f_N(x, p) \ge f_D(y)$.

Proof: It is straightforward to check that the solution (x, p) is feasible for (NLP). To see that $f_N(x, p) \ge f_D(y)$, we make use of the concavity of the logarithm function. From Jensen's inequality:

$$u_{ij} = w_j \log\left(\frac{\sum_{\tau=1}^{D} y_{ij\tau}\, \tau\, r_{ij}/D}{\sum_{\tau=1}^{D} y_{ij\tau}}\right)$$

$$\ge w_j \frac{\sum_{\tau=1}^{D} y_{ij\tau}\, \log(\tau r_{ij}/D)}{\sum_{\tau=1}^{D} y_{ij\tau}}.$$

Since $f_N(x, p) = \sum_{i,j} x_{ij}\, u_{ij}$, we have,

$$f_N(x, p) \ge \sum_{i\in A}\sum_{j\in U} x_{ij} w_j \frac{\sum_{\tau=1}^{D} y_{ij\tau}\, \log(\tau r_{ij}/D)}{\sum_{\tau=1}^{D} y_{ij\tau}}.$$

By the definition of x_{ij},

$$f_N(x, p) \ge \sum_{i\in A}\sum_{j\in U} w_j \sum_{\tau=1}^{D} y_{ij\tau}\, \log(\tau r_{ij}/D).$$

Algorithm nlapPF(U, A)

1. $D = O(n/\epsilon)$
   y = solve_relaxDLP($\{r_{ij}\}$, $\{w_j\}$, D)

2. $\forall i \in A$, $j \in U$
   $x_{ij} = \sum_{\tau=1}^{D} y_{ij\tau}$
   $p_{ij} = \frac{\sum_{\tau=1}^{D} \tau/D \, y_{ij\tau}}{x_{ij}}$
   $u_{ij} = w_j \log r_{ij} p_{ij}$

3. Set up the Generalized Assignment problem using (x, p) and u
4. $\hat{x}$ = roundGAP(p, x, u)
5. $\hat{p}_{ij} = w_j / (\sum_{j' \in U} \hat{x}_{ij'} w_{j'})$

Fig. 4. A formal description of our proportional fair algorithm

By the definition of $f_D(y)$, this finishes the proof. ■

C. Algorithm

With the formulation introduced in the preceding section, we first give an overview of our approximation algorithm referred to as nlapPF. We then discuss the details of the remaining steps of our algorithm, followed by the analysis.

1) Overview of the algorithm: Our algorithm is summarized in Fig. 4. It consists of the following five steps.

1) Set up the problem (DLP) using $D = O(n/\epsilon)$, and disregard the integrality constraints. Solve the resulting linear program, and let y denote its optimal solution.
2) Using Equations shown in Fig. 4, obtain a fractional solution (x, p) to problem (NLP).
3) Use the value p to set up an instance of the Generalized Assignment problem, to which x is a feasible fractional solution.
4) Use the rounding procedure of Shmoys and Tardos [14] to obtain an integer association vector $\hat{x}$.
5) Finally, divide up the time allocation of each AP proportional to the weight of associated users. That is, define $\hat{p}_{ij} = w_j / (\sum_{j' \in U} \hat{x}_{ij'} w_{j'})$ for each user j associated to access point i (i.e. $x_{ij} = 1$).

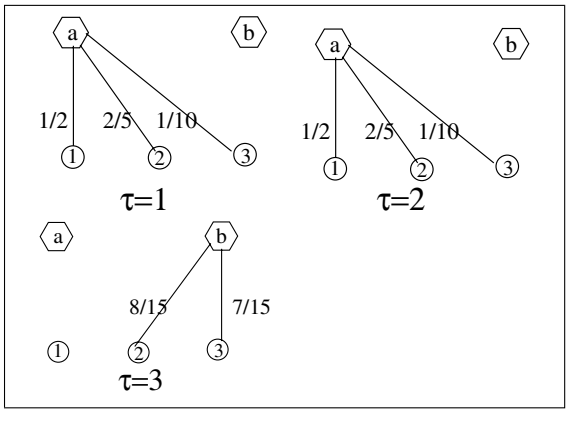


Fig. 5. Step 1: y value for $\tau = 1, 2, 3$.

To better understand these steps, we illustrate them using the example network shown in Fig. 1. Recall that we index AP a and b by 1 and 2. Assume $w_1 = w_2 = w_3 = 1$. Suppose in the first step, we pick $D = 3$. Solving the linear relaxation of DLP, we get the y value as shown in Fig. 5. That is, $y_{111} = y_{112} = 1/2$, $y_{121} = y_{122} = 2/5$, $y_{223} = 8/15$, and $y_{131} = y_{132} = 1/10$, $y_{233} = 7/15$ which means user 1 is associated with AP a and gets assigned 1/2*1+1/2*2=1.5 out of three slots, user 2 is associated with both AP a and b, and gets assigned 2/5*1+2/5*2=1.2 slot from AP a and 1.6 slots from AP b, user 3 is associated with AP a and b and gets 0.3 and 1.4 slots from each.

In Step 2, applying the first equation in this step, we obtain $x_{11} = 1$, $x_{12} = x_{22} = 4/5$ and $x_{13} = x_{23} = 1/5$; applying the second equation, we have $p_{11} = 1/2$, $p_{12} = 2/5$, $p_{22} = 8/15$, $p_{13} = 1/10$ and $p_{23} = 7/15$; applying the third equation, we have $u_{11} = \log(3)$, $u_{12} = \log(96/5)$, $u_{22} = \log(72/5)$ and $u_{13} = \log(16/5)$, $u_{23} = \log(14/5)$.

Step 3 sets up the GAP problem using the x,p,u value obtained in Step 2. In step 4, we will get a rounded solution. One such solution is $x_{11} = 1$, $x_{22} = 1$ and $x_{13} = 1$ which means user 1 and 3 is associated with AP a, and user 2 is associated with AP b. Since p is fixed in Step 4, we obtain the final $\hat{p}$ in Step 5 where $\hat{p}_{11} = \hat{p}_{13} = 1/2$, and $\hat{p}_{22} = 1$.

Our algorithm results in a bandwidth allocation of 3, 24 and 6 for users 1, 2 and 3 respectively.

2) Rounding via Generalized Assignment: We now discuss the most important remaining step in our algorithm. For this section, assume that we have a fractional solution (x, p) to the problem (NLP), obtained, for example, by applying Lemma 5.3. Our goal is to replace the fractional assignment vector x by a $\{0, 1\}$ vector $\hat{x}$ that encodes the desired association of users to APs. In the process of rounding the association vector, we shall hold the time allocation p_{ij} fixed; this allows us to consider a much simpler linear problem. Defining u_{ij} as in Equation (11), consider the following formulation:

$$\begin{array}{lrcl} \text{maximize} & \sum_{i \in A} \sum_{j \in U} x_{ij} \, u_{ij} & & \\ \forall j: & \sum_{i \in A} x_{ij} & = & 1 \\ \forall i: & \sum_{j \in U} x_{ij} \, p_{ij} & \leq & 1 \\ \forall i, j: & x_{ij} & \in & \{0, 1\}. \end{array} \quad \text{(GAP)}$$

The x_{ij} are variables, and p_{ij} and u_{ij} are constants as defined in Equations (10) and (11). Note that the formulation (GAP) is identical to (NLP) except that we are holding the time allocation p fixed. We denote the objective function as $f_G(x)$. Thus, the following lemma holds trivially.

Lemma 5.4: The vector x in a feasible (fractional) solution (x, p) to the formulation NLP is feasible for the formulation GAP. Alternatively, the feasible solution x for GAP together with its parameter vector p is a feasible solution to NLP. Furthermore, $f_N(x, p) = f_G(x)$.

The problem (GAP) is known as the Generalized Assignment Problem. It can be phrased as assigning jobs to machines, with each job being assigned to one machine, and each machine being able to handle multiple jobs. We get reward u_{ij} for assigning job j to machine i, and put load p_{ij} on that machine. The goal is to assign jobs to machines so that no machine has total load more than 1, and the total reward for the assignment is maximized. Note that by construction, we can guarantee that $p_{ij} \leq 1$.

Shmoys and Tardos [14] gave a rounding procedure for the Generalized Assignment problem. The procedure takes a fractional assignment x as input, and constructs an integer assignment vector $\hat{x}$ with the following properties.

1) $f_G(\hat{x}) \geq f_G(x)$.
2) Each user j is assigned to exactly one machine, i.e. $\sum_{i \in A} \hat{x}_{ij} = 1$.
3) The load imposed on each server is at most 2, that is, $\sum_{j \in U} \hat{x}_{ij} p_{ij} \leq 2$ for every server i.

D. Analysis

Finally we can state the following approximation guarantee for our algorithm. If we take any optimal solution (x^*, p^*), scale all bandwidths down by a factor $(2 + O(\epsilon))$, then the utility of this scaled-down optimal solution will be no greater than the utility of our solution $(\hat{x}, \hat{p})$.

Theorem 5.1: Consider any integral association (x^*, p^*) that is a feasible solution to (NLP). Then, for the solution $(\hat{x}, \hat{p})$ produced by our algorithm it holds that

$$f_N(x^*, p^*/(2 + O(\epsilon))) \leq f_N(\hat{x}, \hat{p}).$$

Proof: Consider any integral association vector (x^*, p^*). By Lemma 5.2, we know that there exists a vector y that is a feasible solution of (DLP) such that

$$f_N(x^*, p^*/(1 + \epsilon)) \leq f_D(y). \tag{12}$$

In particular, this holds if y is an optimal solution to (DLP) that we find in step 1 of our algorithm. This is because the optimal objective function value for the linear program relaxation is always greater than or equal to the optimal in the original integer program (basic linear programming theory). By Lemma 5.3, for the solution (x, p) constructed in step 2 of the algorithm we have

$$f_N(x, p) \geq f_D(y). \tag{13}$$

By the first property of the rounding algorithm, $f_G(\hat{x}) > f_G(x)$. By Lemma 5.4, we have $f_N(\hat{x}, p) = f_G(\hat{x})$ and $f_N(x, p) = f_G(x)$. Thus, we produce an integer vector $\hat{x}$ in Step 4 of the algorithm such that

$$f_N(\hat{x}, p) \geq f_N(x, p). \tag{14}$$

Combining Eqn. 12, 13 and 14, we have,

$$f_N(\hat{x}, p) \geq f_N(x^*, p^*/(1 + \epsilon)). \tag{15}$$

The solution $(\hat{x}, p)$ may not be feasible, as some of the access points may be over-scheduled by a factor of 2. However, if we scale down the time allocations p by $1/2$, we obtain a feasible solution. By noting $\sum_{i \in A} \sum_{j \in U} \hat{x}_{ij} w_j = \sum_{i \in A} \sum_{j \in U} x^*_{ij} w_j = n \sum_{j \in U} w_j$, it is matter of simple algebra to check that

$$f_N(\hat{x}, \frac{p}{2}) \geq f_N(x^*, \frac{p^*}{2(1 + \epsilon)}). \tag{16}$$

By Lemma 5.1, the final time allocation $\hat{p}$ is optimal for the given $\hat{x}$; it must be that

$$f_N(\hat{x}, \hat{p}) \geq f_N(\hat{x}, p/2). \tag{17}$$

Combining Eqn. 16 and 17 finishes the proof. ■

We remark that the inapproximibility result in [9] does not contradict to our result. The approximation factor is defined to be the ratio between the objective function value of an algorithm and the optimal objective function value. Their inapproximibility result is due to the fact that the optimal objective function value can be zero. Thus, the factor is unbounded with zero as a denominator. Our approximation ratio is defined to be the scaling factor such that when we scale the bandwidth allocation of an algorithm, we achieve a better objective function value than the optimal. The appealing feature of this definition is that it quantifies the bound of bandwidth penalty of an approximation algorithm. In other words, we need to scale the amount of bandwidth at the access points by the approximation factor in order to achieve more total utility than the optimal achieves will the current amount of bandwidth available at the APs.

VI. Evaluations

A. Methodology

We compare the performance of the following algorithms:

- Our proportional fair algorithms: cvapPF and nlapPF.
- An algorithm which computes an upper bound on proportional fairness. Since the optimal proportional fairness problem is NP-hard, we can not compute an optimal solution in polynomial time unless P=NP. We instead compare with an upper bound of any optimal solution. We obtain this upper bound as follows: we scale up the fractional solution p obtained in step 2 by a factor of $1 + \epsilon$ (equals to 1.1); it is easy to see that the total utility of this scaled solution is greater than the objective function value of any integral solution to formulation (NLP); that is, $f_N(x, p(1+\epsilon)) > f_N(x^*, p^*)$ where (x^*, p^*) is any integral optimal solution to formulation (NLP); We refer to this fractional algorithm as FracPF.
- Two algorithms based on max-min fair allocation described in [8]; we refer to their fractional and integral max-min fair algorithms as FracMM and IntMM respectively.
- A popular heuristic: Strongest-Signal-First (SSF), which is the default user-AP association method in the 802.11 standard.

For our proportional fairness algorithms, we assume a time-fair scheduling mechanism such as [15]. For max-min fair allocation, we assume a throughput-fair scheduling mechanism. For SSF, we investigate the performance under both scheduling mechanisms. The SSF method is referred to as SSF-PF and SSF-MM for the two scheduling schemes respectively.

For ease of comparison, we assume essentially the same simulation setting as the one in [8]. We use a simple wireless channel model in which the user bit rate depends only on the distance to the AP. Adopting the values commonly advertised by 802.11b vendors [3], [1], we assume that the bit rate of users within 50 meters from AP is 11 Mbps, 5.5 Mbps within 80 meters, 2 Mbps within 120 meters, and 1 Mbps within 150 meters, respectively. The maximum transmission range therefore is 150 meters. We do not assume any backhaul capacity limitation.

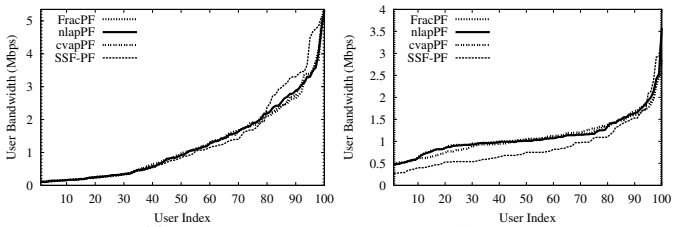
We place a total of 20 APs on a 5 by 4 grid, where the distance between two adjacent APs is set to 100 meters and we assume that an appropriate frequency planning was made.

We have conducted evaluations for two user population sizes. We use 100 to simulate a moderately loaded network and 250 a heavily loaded network. Since the results are similar qualitatively, we only present results for the 100-user case. We assume two types of user distributions: (1) users are distributed within the coverage area uniformly at random; (2) users are randomly positioned in a circle-shape hotspot with the radius of 150 meters near the center of the 20 AP network. We assume all users have the same priority. We choose the D parameter to be ten times the number of users.

We solve the discretized linear program using the CPLEX solver. All of our results presented are averaged over 10 simulation runs.

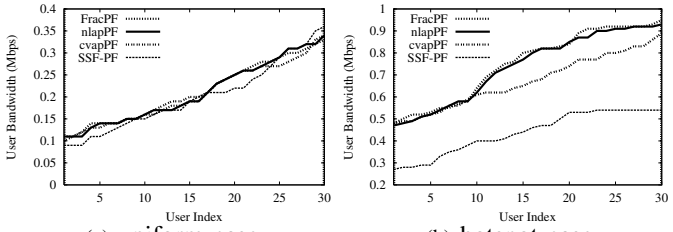
B. Performance Comparison of Proportional Fair Algorithms

We first compare the performance of proportional fairness based algorithms. We use the SSF-PF method as a baseline



(a) uniform case (b) hotspot case

Fig. 6. Comparison of proportional fair based algorithms using per-user bandwidth.



(a) uniform case (b) hotspot case

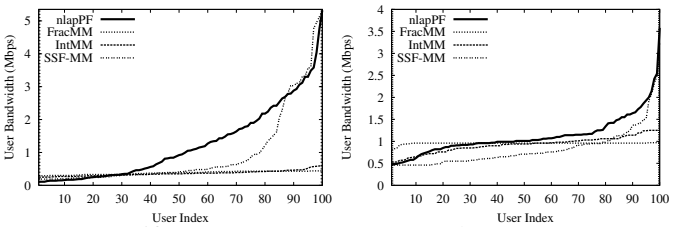
Fig. 7. Zoom in on the per-user bandwidth allocation of proportional fair algorithms.

case for comparison.

Our results are shown in Fig. 6 and 7. The Y axis is the per-user bandwidth in Mbps and the X axis is the user index. The users are sorted by their bandwidth in increasing order. The user locations are different at each run, and therefore the bandwidth of the user with the same x index actually indicates the average bandwidth of the x-th lowest bandwidth user (users allocated the x-th lowest bandwidth). We refer to the vector of per-user bandwidth in increasing order as the bandwidth allocation vector.

We make the following observations. *The bandwidth allocation vector of nlapPF and FracPF coordinate wise is very close.* Only a few coordinates differ by more than 10%. The difference is at most 22%. On the other hand, although SSF-PF performs very well for the uniform case which is expected (still many of the lowest bandwidth users get 20% less bandwidth than nlapPF), *SSF-PF can be very far from ApproxPF in the hotspot case.* For the first 48 lowest-indexed coordinates, coordinate wise, SSF-PF is between 57% and 70% of that allocated by ApproxPF. Thus, *SSF-PF can be very unfair to lots of users without considering fairness in a network-wide context.* To see the details on the differences, Fig. 7 zooms in on the low-bandwidth users. We also note that, nlapPF performs better than cvapPF in the hotspot case. Users index 12 to 30 can get as much as 20% more than cvapPF.

Note that, the per-user bandwidths of the first 36-th users in the uniform case are all smaller than the per-user bandwidth of the lowest bandwidth user in the hotspot case. This seems to be rather counter-intuitive as the users are more concentrated in the hotspot case. This observation is actually an artifact of our setting and does not in general apply in other settings. A close look at our setting shows that, in the hotspot case, all APs have users within range; in addition, since users are deployed uniformly within the coverage range of APs, there is a significant fraction of them are at the boundary with low data rate. For example, for a specific realization of the topology, we see 20 users in the uniform case whose highest data rate to APs is 1Mbps. This is because these users are at the boundary of the grid (18% of the area are more than 120 meters away from any AP); while there is none in the hotspot case because all users are within 120 meters away from at least one AP from the setup. There are 17 users whose highest data rate to APs is 2Mbps while there is only 6 of them in the hotspot case. As a consequence of this deployment setting, we see this non-intuitive result.



(a) uniform case (b) hotspot case

Fig. 8. Per-user bandwidth comparison with max-min algorithms and SSF-MM.

Besides per-user bandwidth, we also compare the algorithms using their aggregated throughput. Table II shows the results. We observe that the three proportional fair-based algorithms achieve very similar aggregate throughput. Although SSF-PF achieves a slightly higher aggregate throughput (less than 5%), this comes at a cost of reducing the bandwidth allocation of the low bandwidth users as shown in Fig. 7.

Case	FracMM	IntMM	SSF		FracPF	nlapPF	cvapPF
			PF	MM			
Uniform	38	36	132	95	125	126	126
Hotspot	96	93	89	83	112	112	110

TABLE II
AGGREGATE THROUGHPUT OF DIFFERENCE SCHEMES IN MBPS.

C. Comparison with Max-min Algorithms and SSF-MM.

We now compare nlapPF with max-min fair based algorithms: FracMM and IntMM [8] and SSF-MM.

We first look at the uniform case. From Fig. 8-(a), we can see that *max-min fairness provides the maximum throughput for the first lowest bandwidth user. However, in order to achieve this, it unduly sacrifices the throughput of other users.* Proportional fairness tries to strike a balance. The first lowest bandwidth user in our scheme gets 44% of the bandwidth it gets from the IntMM scheme. However, as shown in Table II, in terms of aggregate throughput, our scheme is about 236% times more than that of the max-min scheme. The reason for this poor performance of the max-min scheme in terms of aggregate throughput is as follows. In the uniform case, for each AP, there exists some users with very low data rate (situated at the boundary) within its coverage. Because max-min scheme tries to guarantee equal throughput among associated users, this drastically reduces aggregate throughput. Since SSF-MM does not distribute user load in any intelligent way, it is no surprising that our scheme performs much better than SSF. As shown in Table II, our scheme's aggregate throughout is about 35% more than that of SSF-MM (SSF-PF). The median per-user bandwidth value of our scheme is over 260% times that of the max-min scheme and over 2 times that of SSF-MM respectively.

We now look at the hotspot case where users are randomly positioned in a circle-shape hotspot with the radius of 150 meters near the center of the 20 AP network. Notice that in this setup different users can reach different set of cells. From Table II, we see that, because SSF-MM does not intelligently

redistribute users among nearby APs, its aggregate throughput in this case is actually even worse than the max-min scheme. The aggregate throughput of our scheme is about 20% more than that of the IntMM scheme. The lowest per-user bandwidth value of our scheme is 91% of that of the IntMM scheme. The median per-user bandwidth value of our scheme is slightly more (7%) than that of the max-min scheme, and is over 38% more than that of SSF-MM. The performance of max-min scheme in the hotspot case is much better than the uniform case. The reason is that it redistributes users to nearby APs away from the hotspot area; as most of these users situate at the boundary of nearby APs, their rate diversity is low. As a result, the aggregate throughput does not decrease too much when compared with our proportional fair based scheme ApproxPF.

D. Summary

Our simulation results demonstrate clearly that our ApproxPF algorithm achieves close to optimal proportional fairness (close to FracPF, an upper bound of optimal). We have shown that, with rate diversity, proportional fairness tends to be a better fairness measure as the max-min scheme can lead to significant reduction in aggregate throughput. The percentage of throughput gain of our proportional fair schemes over max-min schemes is precisely linked to rate diversity. The percentage is more than 226% in the uniform case, while it is 20% in the hotspot case. The rate diversity of the first case is much more than the latter. The poor performance of SSF in the hotspot case demonstrates the value of association control.

VII. Related Work

The work that is most closely related to this paper is that of [9], [8], [15]. Bu et al. [9] formulated the generalized proportional fairness problem in third generation (3G) wireless data networks. However, their formulation is specific to 3G data networks. In 3G data networks, through a signaling channel, each user feeds back its channel condition continuously to the proportional fairness scheduler at the base station it associates with. At each time slot, the scheduler at each base station schedules the user with the largest weight where the weight is the rate of the user at the current time slot divided by the average rate it has received so far. In our WLAN context, such mechanism does not exist at access points. Furthermore, their association control algorithms do not provide worst case performance bound with respect to optimal proportional fairness. Bejerano et al. [8] consider the problem of achieving network-wide fairness using association control. However, their fairness measure is max-min fairness. Max-min throughput fairness can significantly reduce aggregate throughput in multi-rate WLANs. The max-min time fairness problem they consider is intended for single-rate WLANs. We consider proportional fairness using association control. Our problem is much more challenging than [8] as the objective function is non-linear and the constraints have integral variables. Sadeghi et al. [13], Tan and Guttag in [15] consider the proportional fairness problem at each access point. They do not consider the network-wide proportional fairness problem.

VIII. Conclusion and Future Work

Wireless local area networks (WLANs) based on 802.11 standard have increasingly been deployed in enterprises, academic institutions, etc. In these deployment settings, the physical locations that access points can be deployed are typically constrained; users may not be distributed uniformly. Since each user typically communicates with a single AP for an extended period of time, it is very important to optimize network performance by intelligently distributing users among APs, i.e. association control. Network performance should not be measured in terms of aggregate throughput only. In multi-rate WLANs, without fairness consideration, some users may get starved. We consider the objective of achieving network-wide proportional fairness using association control. From the user's perspective, the network tries to provide equal channel occupancy time to the extent possible while being work-conserving. If each user has a utility function in terms of the bandwidth it gets from the network, and the utility function is concave, proportional fairness tries to maximize the total utility. As the problem is NP-hard, we design two efficient algorithms with worst performance bound. Our evaluation shows that our algorithms are very close to optimal. They can obtain much higher aggregate throughput than max-min fairness based algorithm.

For future work, we would like to study the network-wide proportional fairness problem in mesh networks. In mesh networks, our algorithm has to take routing and interference constraint into account. The problem we consider in this paper is a special case as the user-AP association takes one hop. We also believe that, our result in the paper is of interest to researchers working in network utility optimization because it is a subproblem to many of the very important open problems.

References

[1] Cisco Systems Inc: Data sheet for Cisco Aironet 1200 series.

[2] CISCO Wireless Control System(WCS). http://www.cisco.com/en/US/products/ps6305/index.html.

[3] Proxim Wireless Networks: ORINOCO AP-600 data sheet.

[4] Y. Azar and A. Epstein. Convex programming for scheduling unrelated parallel machines. In *Proc. of STOC*, pages 331–337, 2005.

[5] A. Balachandran, P. Bahl, and G. M. Voelker. Hot-spot congestion relief and service guarantees in public-area wireless networks. *SIGCOMM Comput. Commun. Rev.*, 32(1):59–59, 2002.

[6] A. Balachandran, G. M. Voelker, P. Bahl, and P. V. Rangan. Characterizing user behavior and network performance in a public wireless LAN. In *Proc. of ACM SIGMETRICS*, pages 195–205, 2002.

[7] M. Balazinska and P. Castro. Characterizing mobility and network usage in a corporate wireless local-area network. In *Proc. USENIX MobiSys*, 2003.

[8] Y. Bejerano, S.-J. Han, and L. E. Li. Fairness and load balancing in wireless LANs using association control. In *Proc. of ACM MobiCom*, pages 315–329, 2004.

[9] T. Bu, L. E. Li, and R. Ramjee. Generalized proportional fair scheduling in third generation wireless data networks. In *Proc. of IEEE INFOCOM*, 2006.

[10] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance anomaly of 802.11b. In *Proc. of IEEE INFOCOM*, 2003.

[11] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.

[12] D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *Proc. ACM MobiCom*, pages 107–118, 2002.

[13] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. W. Knightly. OAR: An opportunistic auto-rate media access protocol for ad hoc networks. *Wireless Networks*, 11:39–53, 2005.

[14] D. B. Shmoys and E. Tardos. An approx algorithm for the generalized assignment problem. *Math. Program.*, 62(3):461–474, 1993.

[15] G. Tan and J. Guttag. Time-based fairness improves performance in multi-rate wlans. In *Proc. of USENIX*, June 2004.