Iterative Water-filling for Load-balancing in Wireless LAN or Microcellular Networks

Jeremy K. Chen Theodore S. Rappaport Gustavo de Veciana
Wireless Networking and Communications Group (WNCG), The University of Texas at Austin

***Abstract*— This paper presents an efficient iterative load-balancing algorithm for time and bandwidth allocation among access points (APs) and users subject to heterogeneous fairness and application requirements. The algorithm can be carried out either at a central network switch with site-specific propagation predictions, or in a decentralized manner. The algorithm converges to maximum network resource utilization from any starting point, and usually converges in 3 to 9 iterations in various network conditions including users joining, leaving, and moving within a network and various network sizes. Such a fast convergence allows real-time implementations of our algorithm. Simulation results show that our algorithm has merits over other schemes especially when users exhibit clustered patterns: Our algorithm, when assuming multiple radios at each user, achieves 48% gain of median throughput as compared with the max-min fair load-balancing scheme (also with the multi-radio assumption) while losing 14% of fairness index; we also achieve 26% gain of median throughput and 52% gain of fairness index over the Strongest-Signal-First scheme (which assumes each user has only a single radio). When only a single radio is used, our algorithm is similar to the max-min fairness scheme, and is still better than SSF with 44% gain of 25-percentile throughput and 37% gain of fairness index.**

I. Introduction

People consider increasing the capacity of WLAN or microcellular networks by increasing AP density and assigning proper non-overlapping frequency channels to APs. As the number of APs to which a user can connect increases, an algorithm that efficiently associates users to APs becomes critical for bandwidth and quality of service (QoS) management. However, the default Strongest-Signal-First (SSF) approach used in 802.11 products, in which each user chooses an AP with the strongest signal, results in unevenly distributed loads among APs and poor performance [1]. In order to better balance loads, vendors such as Cisco, Trapeze, Aruba, Meru, and Symbol have introduced central switches to have network-layer controls (e.g. load balancing and handoffs) over the AP's normal processing in physical layers today. This paper presents a load-balancing algorithm that can be carried out either in a decentralized way with some message exchange between APs and mobile users, or at a central switch with site-specific predictions (such predictions can provide the central switch with detailed RF parameters, the received signal-to-noise ratios (SNR), and estimate the achievable capacity for each wireless link; see [2]–[4] and references therein).

Several heuristic load-balancing schemes have been presented. Balachandran *et al* [1] observed that APs with load-balancing functionalities periodically send beacons with current load, captured by the number of users, bit error rates, and signal strengths. However, several measurement studies have shown that the number of users is not a good metric to determine the load [4], [5]. Balachandran *et al* proposes a better load-balancing scheme where each arriving user explicitly asks for a minimum and a maximum bound on bandwidth/throughput, and a centralized admission control is performed to associate the arriving user to an AP that is within the user's radio range and has the most available capacity [1]. The work in [1] improves the degree of load balancing by over 30% and user bandwidth allocation up to 52% in comparison with schemes with little load balancing. The work in [6] presents a decentralized load balancing algorithm that can be applied to IEEE 802.11a/b/g without modifying the standards while being transparent to end users. It was shown by example that the throughput of a station increases from 1.5 to 2 Mbps, and packet delays can be reduced from 450 to 8 ms. While the work in [1], [6] outperforms schemes with little or no load balancing, they are not shown to be optimal. To the best of our knowledge, the only work that achieves some form of optimality in load balancing is [7], which achieves max-min fair bandwidth allocation.

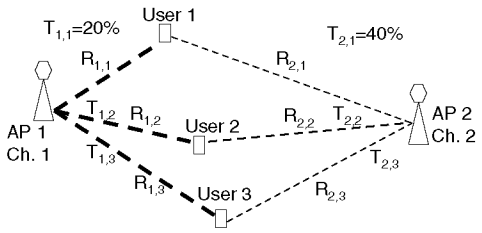


Fig. 1. A simple network with 2 APs and 3 users. Different thicknesses of dashed lines signify different available link capacity $\mathbf{R}_{a,s}$. APs 1 and 2 use disjoint channels. $\mathbf{T}_{a,s}$ denotes the time fraction allocated to user s over the RF channel of AP a.

This paper considers a network with multiple APs and users, as depicted in Fig. 1 and tries to answer a fundamental question: *which AP(s) should be connected with a particular user, and how much time should the specific AP(s) allocate to this user in order to achieve optimal network utilization subject to heterogeneous fairness and application requirements*. Section II describes the system model and notation in detail. Section III presents the formulation and an iterative algorithm for the optimal allocation of channel usage time. Simulation results are presented in Section IV.

[1]This work is supported by NSF Grant ACI-0305644.

0-7803-9392-9/06/$20.00 (c) 2006 IEEE

II. System Model and Notation

We assume a multi-radio capability that allows multiple channels to be received and decoded in parallel by each user (this model has been proposed in [7]). It is suggested in this paper that the multiple-radio assumption simplifies the computation to be efficient (the problem formulation is convex). Our approach can also be used for multi-radio APs. Our algorithm allows up to an unlimited number of radios on a user; however, 2 to 4 radios suffice in practice, since a user in an actual WLAN or microcellular network is usually surrounded by at most 4 APs.

We assume that users exhibit a *quasi-static* mobility pattern (a model that has been adopted in [7]) where users can move from place to place, but they tend to stay in the same physical places for long periods of time [5]. This model allows us to consider long-term averaged link capacities over a time scale of about 2 seconds (denoted as $T_{\rm AVG}$), which is adequate for resource re-allocation and may not be a noticeable time interval for new users who are waiting to be associated with APs. The proposed load-balancing algorithm is executed based on the predicted average capacities during every $T_{\rm AVG}$ interval. Link capacities may change in successive $T_{\rm AVG}$ intervals due to interference or changes in user applications or transmission states. The capacity $\mathbf{R}_{a,s}$ (e.g. throughput) between an AP a, and a user, s, is determined by the peak throughput for a single (unshared) user, and also determined from predicted, measured, or optimized throughput estimates based on site specific information. For the case where multiple users share a single AP over an RF channel, the throughput between the AP, a, and a user, s, is a fraction (the time fraction of channel usage) of the link capacity, that is, $\text{Throughput}_{a,s} = \mathbf{T}_{a,s}\mathbf{R}_{a,s}$, where $\mathbf{T}_{a,s}$ is the channel usage time between AP a and user s. During a $T_{\rm AVG}$ interval, even though users may join/leave the network, or RF noise sources may emit interfering signals, the effects of these transient events on link throughputs are quantized and sampled every $T_{\rm AVG}$ (e.g. block processing is used). In the beginning of every $T_{\rm AVG}$ interval, our iterative load balancing algorithm re-adjusts the time/bandwidth resource allocation over all users and APs.

The algorithm converges to optimum in merely 3 to 9 iterations regardless of network sizes, although the computation time of each *iteration* grows linearly with the number of users multiplied by the number of APs controlled by the switch. On a 2GHz Intel Pentium computer with Windows XP, each iteration in Matlab takes 30 milliseconds for a network with 36 APs and 300 users. Code implemented in assembly or C language would be much faster and is very suitable for real-time implementations of our algorithms on hardware/firmware, as contemplated in [2], [3].

With the above mentioned assumptions, the real throughput that a user experiences mainly depends on the channel usage time allocated from the APs to this user. For instance, in Fig. 1, suppose AP 1 and AP 2 allocate $\mathbf{T}_{1,1} = 20\%$ and $\mathbf{T}_{2,1} = 40\%$ of their time (over disjoint channels 1 and 2, respectively) to user 1, respectively. The total bandwidth that user 1 obtains is $b_1 = \frac{20}{100}\mathbf{R}_{1,1} + \frac{40}{100}\mathbf{R}_{2,1}$; the bandwidths of users 2 and 3 can be computed in a similar way. we consider an infinite backlog of packets (full and ready queues on every channel) for every user. Hence a user's throughput is the same as the bandwidth allocated to her. We maximize the sum utility of throughput, which means maximizing $\sum_{i=1}^{3} U_i(b_i)$ over the channel usage time in this example. If utility functions are properly chosen, users will be allocated different notions of fair allocation when the network reaches maximum sum utility [8].

We made the assumption that all APs are under the control of a network switch. However, some rogue APs or RF noise sources may emit interfering signals in the coverage area of the controlled APs. In this case, some controlled APs or overlay sensors can detect signals from rogue APs. With detected signal parameters and site specific knowledge, position location techniques can locate the rogue APs [2], [3]. Then, AP channel assignments are changed so that the APs near the rogue APs operate at orthogonal RF channels in order to eliminate most interference from rogue APs. Then, the switch will predict SNR and link capacities between users and controlled APs using site specific models for the rogue locations and transmit properties, and apply our algorithm to find the optimal resource allocation accordingly. This paper assumes the frequency band of each AP has been properly assigned [2], [3], and focuses on finding the optimal bandwidth/time allocation in a fully-controlled network.

With an assigned allocated frequency channel, each AP serves its user by time sharing. The fraction of time resource dedicated for payload transmissions between users and an AP, a, over an RF channel is denoted as $T_a^{\rm frac}$ ($0 \le T_a^{\rm frac} \le 1$) (e.g., it ranges from 59% to 88% in 802.11a). The subscript a in $T_a^{\rm frac}$ is used, since the payload time fractions may differ from AP to AP. We suppose that each user shares her utility function to all the APs that transmit signals strong enough to reach her. Then, each AP allocates its time resource (over its assigned RF channel) to users based on the information of the utility functions of all the users within its coverage area, based on site specific knowledge [2]–[4]. In this paper, utility functions are assumed to be concave, continuously differentiable, and strictly increasing [9] for simplicity of analysis. Let n and m denote the numbers of APs and of users, respectively. We use a or s as index when referring to a specific AP or user, and use j or i as dummy indices of APs or users when performing a summation. User s is said to be within the coverage of AP a if $\mathbf{R}_{a,s} > 0$; otherwise, $\mathbf{R}_{a,s} = 0$. Each entry in the rate matrix can be predicted from a site-specific prediction engine [2]–[4]. Within a unit time period, suppose AP a allocates a time fraction $\mathbf{T}_{a,s}$ (over the assigned RF channel of AP a) to user s ($0 \le \mathbf{T}_{a,s} \le 1$). The actual bandwidth that user s gets from AP a is $\mathbf{T}_{a,s}\mathbf{R}_{a,s}$.

III. Maximum Sum Utility with Time Allocation

The optimal AP-user association can be formulated as the sum utility maximization problem in (1) over time resources

from APs on different RF channels to users.

$$\max \sum_i U_i(\sum_j \mathbf{T}_{j,i}\mathbf{R}_{j,i})$$
$$\text{subject to } \sum_i \mathbf{T}_{a,i} \le T_a^{\text{frac}}, \forall a, \quad \text{over } \mathbf{T}_{a,s} \ge 0, \forall a, s. \quad (1)$$

It is hard to find a closed-form expression of the optimal channel usage time allocation for (1). Nevertheless, if the optimization is over the time resources of only a single AP (over one channel), assuming the other APs' time allocations are fixed, closed-form expressions for each AP's optimal time allocation have closed-form expressions, shown in (11) which are solutions to formulation (3). Theorem 3.1 discussed below shows that the original multiple AP problem in (1) reaches the optimum if and only if the time allocation from every AP simultaneously has the closed-form expressions as in (11). Hence, the optimization of the multiple-AP problem can be done by successively optimizing each AP's time resources, as presented in Algorithm 1 as an efficient iterative algorithm. Our derivation and proofs extend [10] to a wide class of utility functions (beyond logarithmic) for different degrees of fairness and application needs. The sole constraint in (1) means that the total channel usage time used at each AP is upper bounded. The objective is to maximize the network utility $\sum_i U_i(\sum_j \mathbf{T}_{j,i}\mathbf{R}_{j,i})$. Mo and Walrand have proposed a class of utility functions that capture different degrees of fairness and model applications with heterogeneous needs parameterized by q_i [8]:

$$U_i(b_i) = \begin{cases} (1-q_i)^{-1} b_i^{(1-q_i)}, & \text{if } q_i \ne 1 \\ \log b_i, & \text{if } q_i = 1 \end{cases}, b_i \in (0, \infty). \quad (2)$$

The parameter q_i has an index i because each user i may have a different application/fairness requirement. This family of utility functions is concave, continuously differentiable, and strictly increasing [8]. The sum of concave functions is still a concave function; hence, problem (1) is convex since a concave function is to be maximized over a convex constraint set [9]. The work in [8] shows that if $q_i \to \infty$, the formulation in (1) becomes a special case that achieves max-min fairness, as studied in [7]. Within every T_{AVG}, $\mathbf{R}$ remains constant after block processing, and the optimal sum utility and $\mathbf{T}$ will be determined accordingly.

Suppose the sum utility optimization is performed over the channel usage time resources of only AP a, $\mathbf{T}_{a,\bullet} = [\mathbf{T}_{a,1}, \mathbf{T}_{a,2}, \ldots, \mathbf{T}_{a,m}]$, assuming that the time allocations from the other APs to users are fixed. Then the formulation in (1) is reduced to

$$\max \sum_i U_i(\mathbf{T}_{a,i}\mathbf{R}_{a,i} + c_{a,i})$$
$$\text{subject to } \sum_i \mathbf{T}_{a,i} \le T_a^{\text{frac}}, \quad \text{over } \mathbf{T}_{a,s} \ge 0 \quad \forall s, \quad (3)$$
$$\text{where } c_{a,i} = \sum_{j:j\ne a} \mathbf{T}_{j,i}\mathbf{R}_{j,i} \text{ are fixed.}$$

Denote by λ_a the Lagrange multiplier for the constraint in (3). Then, the Lagrangian [9] is given by

$$L(\mathbf{T}_{a,\bullet}, \lambda_a) = \sum_i U_i(\mathbf{T}_{a,i}\mathbf{R}_{a,i} + c_{a,i}) - \lambda_a\big(\sum_i \mathbf{T}_{a,i} - T_a^{\text{frac}}\big). \quad (4)$$

Since utility functions $U_s(\cdot)$ are increasing, it is natural to exhaust the time resource for maximizing sum utility [9]; therefore, at the maximum of (3), we have $\sum_i \mathbf{T}_{a,i} = T_a^{\text{frac}}$. Then, the sufficient and necessary optimality conditions (KKT conditions) [9] for (3) can be written as:

$$\mathbf{R}_{a,s}U_s'(\mathbf{T}_{a,s}\mathbf{R}_{a,s} + c_{a,s}) = \lambda_a \quad \text{if } \mathbf{T}_{a,s} > 0, \quad \forall s \quad (5)$$
$$< \lambda_a \quad \text{if } \mathbf{T}_{a,s} = 0, \quad \forall s \quad (6)$$
$$\sum_i \mathbf{T}_{a,i} = T_a^{\text{frac}} \quad (7)$$
$$\mathbf{T}_{a,s} \ge 0, \quad \forall s; \qquad \lambda_a > 0. \quad (8)$$

It is obvious that no time is allocated to links with zero capacity (i.e. $\mathbf{T}_{a,s} = 0$ if $\mathbf{R}_{a,s} = 0$). Therefore, we focus on deriving the optimal $\mathbf{T}_{a,s}$ for $\mathbf{R}_{a,s} > 0$. For general utility functions, the optimal time fraction can be derived from (5):

$$\mathbf{T}_{a,s} = \Big\{\frac{1}{\mathbf{R}_{a,s}} U_s'^{-1}\big(\frac{\lambda_a}{\mathbf{R}_{a,s}}\big) - \frac{c_{a,s}}{\mathbf{R}_{a,s}}\Big\}^+. \quad (9)$$

While closed-form solutions of $\mathbf{T}_{a,s}$ do not exist for general utility functions, they can be obtained for the family of utility functions in (2), for which (5) becomes

$$\frac{\partial L}{\partial \mathbf{T}_{a,s}} = \frac{\mathbf{R}_{a,s}}{(\mathbf{T}_{a,s}\mathbf{R}_{a,s} + c_{a,s})^{q_s}} - \lambda_a \quad (10)$$

Equating (10) with zero gives the optimal time allocation:

$$\mathbf{T}_{a,s} = \Big\{\lambda_a^{(-\frac{1}{q_s})} \mathbf{R}_{a,s}^{(\frac{1}{q_s}-1)} - \frac{c_{a,s}}{\mathbf{R}_{a,s}}\Big\}^+ \quad (11)$$

In (9) and (11), the notation $\{x\}^+$ is needed because $\mathbf{T}_{a,s}$ is nonnegative: $\{x\}^+ = x$ if $x \ge 0$ and $\{x\}^+ = 0$ otherwise. By substituting (11) or (9) into $\sum_i \mathbf{T}_{a,i} = T_a^{\text{frac}}$ in (7), λ_a for each AP a can be numerically solved [9], [10]. In each iteration of our algorithm, finding the time resources of each AP requires solving a single-variable (λ_a) polynomial equation with m terms; hence, the time complexity of each iteration is $O(nm)$. If the parameter $q_s = 1$, the expression of $\mathbf{T}_{a,s}$ in (11) is the water-filling expression, where the constant λ_a^{-1} is known as the water-filling level [10].

Theorem 3.1: $\{\mathbf{T}_{a,s}, \forall a, s\}$ is an optimal solution to (1) if and only if $\{\mathbf{T}_{a,1}, \mathbf{T}_{a,2}, \ldots, \mathbf{T}_{a,m}\}$ is the solution in (11) for AP a with the time allocation from the other APs $\{\mathbf{T}_{j,s} : \forall j \ne a, \forall s\}$ fixed, for all $a = 1, 2, \ldots, n$. (The proof is omitted as it is a natural extension of Theorem 1 in [10].)

As described in Theorem 3.1, the time allocations from each AP to users can be solved by (11), assuming time allocations from the other APs are fixed. Hence, the optimal time allocation for the multiple-AP optimization problem (1) can be found by an iterative algorithm (see Algorithm 1).

Theorem 3.2: Algorithm 1 results in an optimal sum utility and causes $\{\mathbf{T}_{a,s}, \forall a, s\}$ to converge to an optimal time

Algorithm 1 An iterative algorithm to solve (1)

Given a rate matrix $\{\mathbf{R}_{a,s}, \forall a, s\}$.
Start with a valid time allocation $\{\mathbf{T}_{a,s}, \forall a, s\}$.
repeat
  for each AP $a = 1, 2, \ldots, n$ **do**
    Compute $\{c_{a,s}, \forall s\}$ by (3).
    Compute $\{\mathbf{T}_{a,s}, \forall s\}$ by (11) or (9).
  end for
until the sum utility converges
Output $\{\mathbf{T}_{a,s}, \forall a, s\}$.

allocation for Formulation (1). (The proof can be extended from the proof of Theorem 2 in [10]. Note that the optimum time allocation $\{\mathbf{T}_{a,s}, \forall a, s\}$ may not be unique.)

Algorithm 1 can be carried out in a decentralized manner: each AP a computes the optimal time allocation $\{\mathbf{T}_{a,s}, \forall s\}$ only for those users who are in the coverage of this AP. For the computation of each user's $\mathbf{T}_{a,s}$, a constant $c_{a,s}$ needs to be known, which in turn requires the knowledge of the bandwidth that this user s receives from APs other than AP a. In a realistic WLAN setup, a user is under the coverage of no more than 4 APs; hence, the computation of $c_{a,s}$ at each user is efficient. APs sequentially perform such decentralized computing. When the sum utility converges, a control message may be sent to APs to stop the decentralized computing.

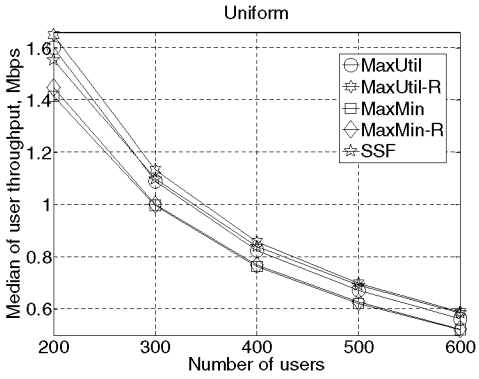
IV. Simulation Results

In this section, we compare the throughput and fairness performance of our maximum utility (denoted as *MaxUtil*) scheme with the max-min fairness scheme in [7], denoted as *MaxMin*, and the Strongest-Signal-First scheme in current 802.11 implementations. We consider a simplified scenario of free-space propagation model where no obstacles exist in the vicinity of APs. It is clear that our algorithm can utilize site specific information, which will be considered in future work. We consider different percentages (between 1% and 5%) of users joining, leaving, or moving within the network; hence, the link capacities change over time. We sample $\mathbf{R}$ for every T_{AVG}, and within this time interval, $\mathbf{R}$ is fixed. Two kinds of user distributions, namely *uniform* and *cluster* (or hotspot), are considered. First, users are uniformly distributed in a 600 meters by 600 meters square that encompasses the 36 APs. Second, we consider that a hotspot at the center attracts more people: users are distributed in a circle-shaped area centered at the middle of the APs with a radius of 250 meters. Users are randomly located on this circle based on their uniformly generated polar coordinates (the distance from the center and the polar angle are uniformly distributed between $(0, 250)$ and $(0, 2\pi)$, respectively). From the viewpoint of the Cartesian coordinate, the user density is higher near the center than near the circumference of the circle. Each point on the figures is an average over 100 independent runs. In the SSF case, each user (whose transceiver can handle only a single channel) associates with the strongest AP, and then each AP evenly distributes its time resources to the associated users. Simulations show that the number of iterations (mostly between 3 and 9) does not grow with the number of users. Our algorithm converges quickly even for large networks.

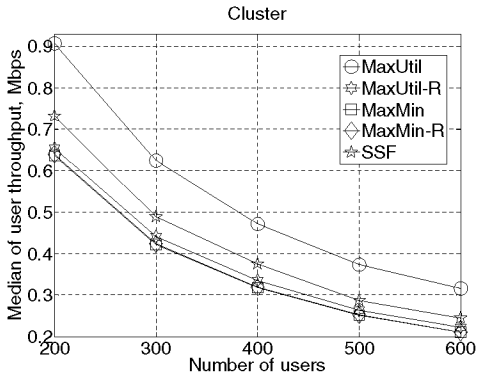
Figs. 2 and 3 show the medians and the 25-percentiles of user throughputs, respectively. Table I presents fairness indices (see [11] for this metric) for cases with 400 users; scenarios with different number of APs and users are omitted, since their fairness index values are similar to those in Table I. Both *MaxUtil* and *MaxMin* assume that each user has multiple radios. For fair comparisons with SSF, we also compute single-radio results by properly rounding multi-AP time allocation; *MaxMin-R* denotes the results produced by the rounding method in [7]. The *MaxUtil-R* results were obtained by a different rounding method: we first compute normal multiple-radio time allocation; then, if any user indeed uses multiple APs, this user simply chooses the AP that supplies her with the most bandwidth. Finally, if any AP has any time resource remained not allocated, this AP allocates the remaining time proportionally to its associated users. For example, if the rate matrix $\mathbf{R} = \begin{bmatrix} 7 & 5 & 6 & 3 \\ 4 & 1 & 4 & 4 \end{bmatrix}$ and all users' utility parameters, q, are 1, then the optimal time fraction (allowing multi radios) is $\mathbf{T} = \begin{bmatrix} 0.417 & 0.417 & 0.166 & 0 \\ 0 & 0 & 0.375 & 0.625 \end{bmatrix}$. Each user chooses only one single AP; then the time matrix becomes $\mathbf{T} = \begin{bmatrix} 0.417 & 0.417 & 0 & 0 \\ 0 & 0 & 0.375 & 0.625 \end{bmatrix}$. Then, since the first AP has time fraction (16.6%) remained, the remaining time is proportionally distributed to users 1 and 2; finally the time matrix for the single-radio case is $\mathbf{T} = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.375 & 0.625 \end{bmatrix}$.

A trade-off between throughput and fairness can be seen in multi-radio cases *MaxUtil* and *MaxMin*. Our *MaxUtil* has very good performance in *cluster* case: in Fig. 2(b), *MaxUtil* exhibits about 48% higher median throughput over *MaxMin* while sacrificing only 14% of fairness as in Table I. It is because *MaxMin* tends to achieve absolute fairness (its fairness index is almost 100% as in Table I) by sacrificing throughput (giving more time resource to users with poor link capacities). Our *MaxUtil* trades throughput with fairness; even in *uniform* case in Fig. 2(a), *MaxUtil* yields 9% higher median throughput than *MaxMin* while losing 2% of fairness as in Table I. Our algorithm, with multiple radios at each user, outperforms SSF by 26% and 52% in terms of median throughput and fairness index, respectively, as in Fig. 2(b) and Table I.

Surprisingly, the single-radio scheme *MaxUtil-R* yields worse median throughput than SSF, mainly because our rounding method (as presented in the numerical example above) makes users choose stronger APs, thereby causing unbalanced loads on APs. The rounding method in [7] may be modified to be imposed upon *MaxUtil* for better rounding performance; this is a subject for future research. Nevertheless, *MaxUtil-R* yields similar 25-percentile user throughputs as *MaxMin-R*, and is 44% and 17% higher than SSF in cluster and uniform cases, respectively (as seen in Fig. 3). Moreover, Table I indicates that SSF has poor fairness indices as compared

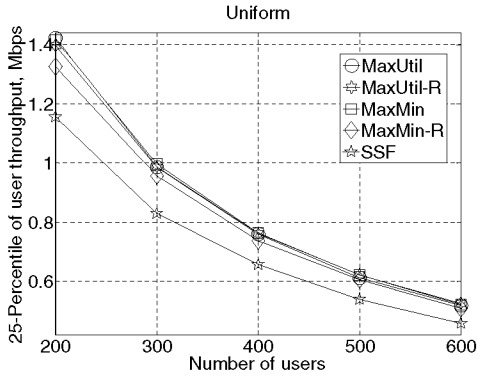


(a) Uniform user distribution

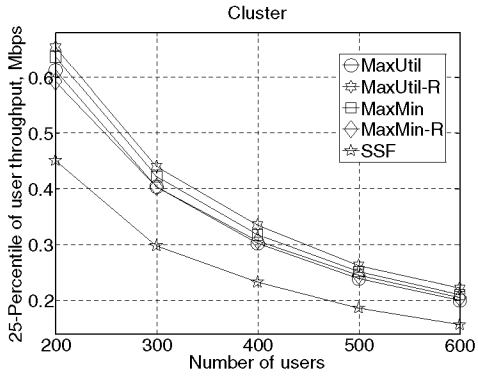


(b) Clustered user distribution

Fig. 2. The median of user throughput.



(a) Uniform user distribution



(b) Clustered user distribution

Fig. 3. The 25-percentile of user throughput.

TABLE I

FAIRNESS INDEX (CF. [11]) OF USER THROUGHPUT ALLOCATION FOR TWO KINDS OF USER DISTRIBUTIONS (CLUSTER AND UNIFORM) IN A NETWORK WITH 36 APS AND 400 USERS. (UNIT: %)

	MaxMin	MaxMin-R	MaxUtil	MaxUtil-R	SSF
Cluster	99.6	97.9	85.7	71.4	34.2
Uniform	100	99.3	98.2	95.7	85.5

with all other schemes (37% lower than *MaxUtil-R* in cluster case, for example). In summary, our method, *MaxUtil-R*, outperforms SSF in terms of 25-percentile throughput and fairness index with small sacrifice of median throughput.

V. CONCLUSIONS

We find analytical expressions for the optimal channel usage time allocation and present a fast iterative algorithm to achieve the optimum. Simulation results show that when users are clustered, our utility maximization formulation yields substantial throughput gain over both the max-min scheme in [7] and the SSF scheme, which is currently being used in WLAN products. When users are uniformly distributed in space, our max utility scheme is similar as the scheme in [7], and achieves better fairness than SSF. Regardless of the number of APs or users in a network, the convergence of the sum utility is fast in various network conditions such as users joining, leaving, or moving within the network. Therefore, the iterative algorithm has good scalability and can be implemented in real time.

REFERENCES

[1] A. Balachandran, P. Bahl, and G. M. Voelker, "Hot-spot congestion relief in public-area wireless networks," in *Proc. Fourth IEEE Workshop on Mobile Computing Systems and Applications*, June 2002, pp. 70 – 80.

[2] T. S. Rappaport and R. R. Skidmore, "System and method for predicting network performance and position location using multiple table lookups," U.S. Patent Appl., no. 20040259555, Dec. 2004.

[3] ——, "System and method for automated placement or configuration of equipment for obtaining desired network performance objectives and for security, RF tags, and bandwidth provisioning," U.S. Patent Appl., no. 20040236547, Nov. 2004.

[4] C. Na, J. K. Chen, and T. S. Rappaport, "Measured traffic statistics and throughput of IEEE 802.11b public WLAN hotspots with three different applications," *IEEE Trans. Wireless Commun.*, to appear.

[5] D. Kotz and K. Essien, "Analysis of a campus-wide wireless network," in *Proc. the Eighth Annual Int. Conf. on Mobile Computing and Networking (MobiCom)*. ACM Press, September 2002.

[6] H. Velayos, V. Aleo, and G. Karlsson, "Load balancing in overlapping wireless LAN cells," in *Proc. IEEE International Conference on Communications*, vol. 7, June 2004, pp. 3833 – 3836.

[7] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," in *Proc. ACM MobiCom*, Sept. 2004, pp. 315 – 329.

[8] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 556 – 567, Oct. 2000.

[9] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[10] W. Yu, W. Rhee, S. Boyd, and J. M. Cioffi, "Iterative water-filling for Gaussian vector multiple-access channels," *IEEE Trans. Inform. Theory*, vol. 50, no. 1, pp. 145 – 152, Jan. 2004.

[11] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," DEC, Research Report TR-301, 1984.