Cell Breathing in Wireless LANs: Algorithms and Evaluation

Paramvir Bahl, Microsoft Research
Mohammad T. Hajiaghayi, CMU
Kamal Jain, Microsoft Research
Vahab Mirrokni, MIT
Lili Qiu, UT Austin
Amin Saberi, Stanford

***Abstract*— Wireless LAN administrators often have to deal with the problem of sporadic client congestion in popular locations within the network. Existing approaches that relieve congestion by balancing the traffic load are encumbered by the modifications that are required to both access points and clients. We propose *Cell Breathing*, a well known concept in cellular telephony, as a load balancing mechanism to handle client congestion in a wireless LAN. We develop power management algorithms for controlling the coverage of access points to handle dynamic changes in client workloads. We further incorporate hand-off costs and manufacturer specified power level constraints into our algorithms. Our approach does not require modification to clients or to the standard. It only changes the transmission power of beacon packets, and does *not* change the transmission power of data packets to avoid the effects of auto-rating. We analyze the worst-case bounds of the algorithms, and show they are either optimal or close to optimal. In addition, we evaluate our algorithms empirically using synthetic and real wireless LAN traces. Our results show that cell breathing significantly out-performs the commonly used fixed power scheme, and performs at par with sophisticated load balancing schemes that require changes to both the client and access points.**

***Index Terms*— Wireless LAN, power control, cell breathing, algorithms.**

I. Introduction

The proliferation of lightweight hand-held devices with built in high-speed WiFi network cards and the significant benefit of any-where any-time Internet access has spurred the deployment of wireless "hot-spot" networks [2], [3]. It is easy to find wireless local area networks (WLANs) in classrooms, offices, airports, hotels, and malls. A key challenge for organizations that deploy WLANs is capacity management, making the best use of the available resources to derive the best return on investment while satisfying client service demands.

Previous studies of public-area wireless networks have shown that client service demands are highly dynamic in terms of both time of day and location, and that client load is often distributed unevenly among wireless access points (APs) [6], [7], [36], [39]. Clients tend to localize themselves in particular areas of the network for various reasons, such as availability of favorable network connectivity, proximity to power outlets, classrooms, meeting rooms, or geographic constraints of other services (e.g., airport gate area with arriving and departing flights). A consequence of such behavior is sporadic client congestion at popular locations within the network. At any one time, a large percentage of mobile clients communicate with a small subset of the APs. These client concentrations create an unbalanced load in the network and complicate capacity planning, making it difficult to accommodate heavy, concentrated load in different parts of the network without significant, and costly, over-engineering.

The mapping between clients and the APs that service them is a critical determinant of system performance and resource usage. An AP can get seriously overloaded even when several nearby APs are lightly loaded. This is because a majority of the WiFi cards associate with the APs with the loudest beacons (i.e. the strength of the received beacon signal is highest among all neighboring APs). We call this as the basic association scheme.

One way to address this issue is to modify the client association algorithm to incorporate the APs' load in addition to the received signal strength indicator (RSSI) of the APs' beacon. A client associates with the AP that is lightly loaded and whose beacons have a highest RSSI value. This technique and its different variants have been proposed by researchers (e.g., [30], [17], [22], [5], [8]), and adopted by vendors of wireless LAN products [28], [11], [1]. The problem is that this technique requires support from both APs and clients. APs have to communicate their current load to the client, and the client AP selection algorithm has to incorporate APs load information. In practice, clients in public areas are generally heterogeneous, i.e. they use wireless cards from different vendors or wireless cards that are running older "legacy" software. Consequently, such schemes provide limited benefit.

To achieve efficient resource usage without requiring changes to client software, we propose the use of *cell breathing* technique. Cell breathing is a well known concept in cellular telephony (2G, 3G, CDMA, CDMA200 and WCDMA systems) [14], [31]. It is defined as the constant change in the geographical area covered by the cell tower. When the cell becomes heavily loaded, it shrinks, and the lightly loaded neighboring cells expand.

In this way, client traffic from the overloaded cell is redirected to neighboring cells, and consequently, the overall system is load balanced.

In WLANs, cell breathing can be implemented by *controlling the transmission power of an AP's beacon packets*. Note that we do *not* change the transmission power of data packets to avoid degrading clients' performance. More specifically, when SNR of data packets reduces, the AP may see higher data packet losses, or even adapt to a lower sending rate, both of which degrade the client's performance. In comparison, changing the transmission power of beacon packets only affect how clients associate with APs, and does not affect the loss rate or sending rate of data packets, which matches our goal well.

Our proposed power control does not require any change to client software or to the standard. Clients continue to associate with an AP with the strongest beacon. APs manage their load by adjusting the beacon packets' transmission power. In this way, the AP's coverage area is shrunk or expanded transparently, adapting to client demands and balancing the traffic load across the network. Because this approach does not require modifications to clients, its deployment cost and time is small. Moreover cell breathing is effective for both legacy clients that employ the basic association scheme and the new clients that employ load-aware association scheme. So in practice its benefits can be fully realized immediately.

Finding the appropriate power assignment at APs to automatically achieve load balancing is a challenging problem. To our knowledge, the cell breathing algorithms proposed for cellular networks are based on local heuristics, and do not provide performance guarantees [14], [31].

In this paper, we develop power control algorithms for the following two cases: (i) APs are able to adjust their power to any level (*continuous-power assignment*), and (ii) APs are able to adjust their power to only some discrete power levels (*i.e.*, *discrete-power assignment*).

To develop an algorithm for continuous-power assignment, we use a duality-based approach. The duality in linear and convex programs has proved effective for algorithm designs [41]. It has been also used in analysis and design of congestion control mechanisms in the Internet [24], [26], [27]. Our duality-based approach uses linear programming to formulate the problems, and use linear programming duality and the complementary slackness conditions to derive algorithms and prove their correctness.

More specifically, in many situations, one can see dual variables as shadow prices. In our context, the shadow prices correspond to the AP power. In other words, similar to a market mechanism in which the price determines the demand for a commodity, here we can change the demands assigned to an AP by adjusting its power. The challenge is to adjust the powers of all neighboring APs at the same time in such a way that their loads remain balanced.

Interestingly, when client demands are homogeneous (i.e., all clients have the same demand), we can always compute such a power assignment - we can set the powers of all APs in such a way that after all the clients choose their AP based on RSSI, either all the clients can be served by the APs or all the APs are fully utilized. For heterogeneous demands (i.e., clients can have different demands), we apply the same approach, and prove that it can completely satisfy at least $N - K$ clients, where N is the number of clients, and K is the number of APs (Note that K is often much smaller than N in practical scenarios).

We further develop a primal-dual combinatorial algorithm based on the matching theory, which is applicable to a more general setting. In this case, we only assume that received power is proportional to the transmission power, but do not assume any relationship between the received power and the distance. (In fact, our algorithm does not even require the knowledge of the distance between APs and clients.) The algorithm is described in the Appendix. It is based on the insight that our problem has similarity to market equilibrium problem[13], [21], [18]. Based on the same insight, we can apply the ideas of the auction-based distributed algorithms for computing market equilibria [18], and develop a distributed algorithm for our problem.

For discrete-power assignment, we develop a greedy algorithm. The high level idea of our algorithm is as follows. We start by setting the powers of all APs to the highest value, and then we choose the best power configuration resulting from iteratively decreasing the power of overloaded APs. This approach is intuitive and easy to implement. Moreover, *it only requires knowledge of APs' load*, which is easy to obtain. We show that if there exists a power assignment such that each AP has capacity to accommodate the demands assigned to it, our algorithm can find the solution in a polynomial time.

In addition, we consider two extensions to the above algorithms: (1) dynamic adjustment of APs' power in response to changes in clients' load while limiting the number of hand-offs, and (2) minimizing APs' transmission power to reduce interference.

To sum up, the key contributions of our research are as follow:

- We describe four algorithms for continuous-power assignment. The first three algorithms assume that APs can estimate the received power at the clients. Among the three algorithms, we prove the first two are optimal (i.e., maximizes the total satisfied client demand) for homogeneous demands, and the third is close to optimal for heterogeneous demands. Our fourth algorithm, described in the Appendix, is designed for a more general case, where the only assumption about received power is that it is proportional to the transmission power.

- We describe a greedy algorithm for discrete power assignment, and prove its optimality under a certain condition. The algorithm only requires APs' load as input.
- We extend the algorithms to handle dynamically changing client demands while limiting the number of hand-offs. We also consider minimizing APs' power to reduce interference.
- We evaluate the algorithms using both synthetic and real WLAN traces.

Our results show the algorithms are effective for improving throughput. Under high load, the improvement is up to 50% for uniform client distributions, and up to an order of magnitude for nonuniform distribution of clients' locations.

The remaining paper is organized as follow. In Section II, we review related work. We formulate the power control problem in wireless LAN in Section III. In Section IV, we present algorithms for continuous power assignments, and analyze their worst-case bounds for both homogeneous and heterogeneous client demands. We describe a discrete power assignment algorithm in Section V. In Section VI, we develop a dynamic power control algorithm that adapts to changes in client demands while limiting the number of hand-offs. In Section VII, we consider minimizing APs' power. We describe our evaluation methodology in Section VIII, and present performance results in Section IX. Finally we conclude in Section X.

II. Related Work

Several researchers have studied the usage characteristics of wireless LANs in different environments, including a university campus [39], [25], a large corporation [44], and a conference [6]. All these studies report that the client load is unevenly distributed across AP. The imbalance in client load distribution results in inefficient resource utilization and poor performance.

As suggested in several previous work (e.g., [30], [22], [17], [5], [8]), one approach to addressing the load imbalance issue is to incorporate APs' load into the association scheme. For example, Papanikos and Logothetis [30] determine client and AP association based on RSSI and the number of clients associated with each AP. The authors in [22], [17] propose that APs maintain a measurement of their load, and broadcast beacons containing this load to clients in the cell. New clients receive beacons from multiple APs, and use this information to associate with the least loaded AP [22], [17]. Balachandran *et al.* suggest clients associate with the AP that can accommodate its minimum bandwidth requirement. When multiple such APs are available, the AP with highest RSSI is chosen. Bejerano *et al.* develop network-wide max-min fair bandwidth allocation algorithms. In their scheme, each client deploys an appropriate client software to monitor the wireless channel quality it experiences from all its nearby APs. The client then reports the information to a network control center, which determines client and AP association. Their algorithms are the first that provide worst-case guarantees on the quality of the bandwidth allocation.

All the above work assume that clients deploy the appropriate module for AP selection. This requirement is hard to realize in practice, since wireless cards at clients are heterogeneous, and may not support such cooperation. Moreover, the required modules for AP selection may differ from one network to another network (e.g., some networks may require clients to report the information to a centralized server for determining association, while other networks require clients to make selection by themselves). The different requirements posed by different wireless networks make the deployment even harder.

As stated in the previous section, we propose an AP-centric approach. When an AP becomes heavily loaded, it shrinks its coverage by reducing the transmission power of its beacon packets. This forces redirection of some traffic to a neighboring cell that is lightly load, thereby achieving load balancing. Different from the previous cell-breathing work in cellular network, which use heuristics and does not give worst-case performance guarantees, we prove our power control algorithms are optimal for homogeneous demands and close to optimal for heterogeneous demands. In addition, our algorithm can adapt to changing client demands while limiting the number of clients required to switch to different APs.

The concept of cell breathing originates from cellular networks. To our knowledge, the cell breathing algorithms proposed for cellular networks are based on local heuristics, and do not provide performance guarantees [14], [31]. More recently, Sang *et al.* [33] propose a cross-layer framework that coordinates packet-level scheduling, call-level cell-site selection and handoff, and system-level load balancing. One of the components in their framework is cell breathing. Different from our work, which uses power control at the physical layer, their cell breathing is performed at MAC layer by having a congested cell allocate less time slots to the mobiles at the cell boundary. Such TDMA-based scheme is not applicable to IEEE 802.11 DCF (distributed coordinated function), which is the focus of our work. Moreover under their cell breathing scheme, the mobiles are still required to perform load-aware cell-site selection and handoff. In comparison, our cell breathing scheme completely removes the need of client-side modifications (i.e., clients can simply associate with the access point based only on signal strength).

There are significant research work on power control for ad hoc networks (e.g., [32], [23], [42], [37]). Ad hoc networks are significantly different from infrastructure wireless networks, and these schemes do not apply to our scenarios.

There is a close relationship between our power assignment problems and the market equilibria [13], [21], [18]. We use the insight to develop some of the algorithms.

III. Problem formulation

We propose an AP-centric approach to transparently balance load across different APs. The main challenge in this approach is to find appropriate transmission power for each AP such that the total client demand that APs can serve is maximized when clients use the basic association scheme (i.e., associate with the AP with highest RSSI). In order to formally specify the power control problem, we first introduce the following notations.

- K: The number of APs
- N: The number of clients
- C_i: The capacity of AP i
- $d(i, j)$: The distance between AP i and client j
- α: signal attenuation factor
- P_i: AP i's power
- D_i: client i's demand
- $P_r(i, j)$: received power from AP i at client j
- L_i: The total load served at AP i

Based on the notations listed above, we now formulate the power control problem as follow. Given K, N, C_i, D_i, and $d(i, j)$, our goal is to find the transmission power for each AP i, denoted as P_i, to maximize system throughput (i.e., maximizing $\sum_i L_i$) given that client j is assigned to AP i when $P_r(i, j) > P_r(i', j)$ for all $i' \in \{1, 2, ..., K\}$, and $L_i = min(C_i, \sum D_j)$ for all clients j that are mapped to AP i. The last equation reflects the fact that the maximum client demand the AP i can service, L_i, is bounded by its capacity and the total client demands that are assigned to it. Note that when there are multiple APs with similar RSSI, the client is randomly assigned to one of them.

IV. Maximizing Throughput for Continuous Power

In this section, we present power control algorithms for the cases when APs can adjust their power to any value (i.e., continuous power). The algorithms in this section require APs to estimate the received power at different clients. In Appendix, we extend the algorithm to a more general case that does not require the knowledge of the distance between APs and clients. It only assumes the received power at any location is proportional to the transmission power, which holds in general even under obstruction. Moreover, the discrete-power assignment algorithm presented in Section V requires even less information – only APs' load information is needed.

We estimate the received power at the clients as follow. The received power, $P_r(i, j)$, is a function of transmission power, $P(i)$, and the distance between the client and AP, $d(i, j)$. The function depends on the wireless propagation model in use. We use the following function:

$$P_r(i, j) = a * P_i / d(i, j)^\alpha \quad (1)$$

where a is a constant. It is easy to see that this power function can incorporate both free-space and two-ray ground reflection models.

In addition, we can also incorporate other wireless propagation models as follow. When the wireless propagation does not follow Equation 1 (e.g., under obstruction), we can approximate the actual wireless propagation by introducing virtual distance, where the virtual distance follows Equation 1. More specifically, APs collect the measurement of transmission power and received power, and then approximate the actual wireless propagation by finding $d'(i, j)$ (virtual distance), α' (virtual attenuation factor), and a' that fit the model $P_r(i, j) = a' * P_i / d'(i, j)^{\alpha'}$, where $P_r(i, j)$ and P_i are from the measurement. Then we apply our power assignment to the virtual distances and virtual attenuation factor. Conceptually, this is similar to Internet distance embedding (e.g., GNP [29]), which embeds a complicated Internet space onto a simple geometric space. In our case, we embed a complicated space, which describes actual wireless propagation, onto a simpler space that follows Equation 1.

A. Maximizing Throughput for Homogeneous Demands

First we design power control algorithms for homogeneous client demands. Without loss of generality, we consider each client has one unit of demand. (Since client demands are homogeneous, we can always scale client demands and AP capacity to make the client demand to be one unit.) We first find a mapping of clients to APs such that either all clients' demands are satisfied or the total capacity of all APs are exhausted. It is easy to see that such a mapping maximizes our objective – the total satisfied demand, since the total satisfied demand cannot exceed the total client demand or APs' capacity. Then we prove that there exists a set of powers that enforces this assignment, when each client selects its AP based on RSSI. Next we derive two algorithms to find the set of powers that enforces this assignment. The first algorithm is based on solving a linear program. The second algorithm is combinatorial, and has a better running time.

1) Finding the Mapping.: We develop a polynomial-time algorithm to find a mapping of clients to APs such that either clients' demands are satisfied or the total capacity of all APs are exhausted. We call this mapping a perfect assignment. We prove that there exists a set of powers for APs that enforces this assignment under homogeneous client demands. Our proof uses linear programming duality and complementary slackness conditions.

The algorithm to find the assignment is as follows:

FindAssignment Algorithm

1) Given an instance of the power control problem as specified in Section III, we can construct the following weighted bipartite graph $G(A, C, E)$, where A is the set of APs, and C is the set of clients. There is an edge between each AP i and each client j. The weight of the edge from AP i to client j is equal to $w_{ij} = \alpha \ln(d(i, j))$.
2) Find the minimum weight bipartite matching in G, where the capacity of every client is 1, and the capacity of an AP i, is C_i. In other words, among all the maximal assignments of clients to access points in which a client can be assigned to at most one AP and an AP i can be matched to at most C_i clients, find the matching with the minimum weight that covers either all the clients or all the APs.

Note that the minimum weighted perfect matching problem (even in general graphs) can be solved in polynomial time [15], [16]. For bipartite graphs, simple primal-dual algorithms are known for this problem (see e.g. [43]). In addition, the integrality gap of the natural linear programming formulation is one, which means that we can find the optimal solution by solving a linear program (see e.g. [35]). Below we prove that there exists a set of powers that enforce the assignment obtained by the above algorithm.

Theorem 1: There exists a set of powers that enforces the assignment obtained by **FindAssignment** algorithm.

Proof: First assume that the perfect matching covers all clients. We will consider the other case where the perfect matching covers all APs later. We can formalize the minimum weighted perfect matching problem as follows:

$$\begin{aligned}
&\text{minimize} && && \sum_{i\in C, j\in A} w_{ij}x_{ij} && (2)\\
&\text{subject to} && \forall i \in C && \sum_{j\in A} x_{ij} = 1 &&\\
& && \forall j \in A && \sum_{i\in C} x_{ij} \le C_j &&\\
& && \forall i \in C, j \in A && x_{ij} \ge 0 &&
\end{aligned}$$

In the above linear program, $x_{ij} = 1$ indicates that client i is assigned to AP j in the matching. The first constraint indicates that each client is assigned to at most one AP. The second constraint shows that AP j is assigned to at most C_j clients. Our objective is to minimize the weight of the resulting matching.

Since there exists an assignment that covers all clients, this linear program has a feasible solution. As mentioned before, it is known that in bipartite graphs the integrality gap of the above Linear Program is one. Thus, there is an optimum solution with 0-1 variables.

The dual of the above linear program is as follows:

$$\begin{aligned}
&\text{maximize} && && \sum_{i\in C} \lambda_i + \sum_{j\in A} C_j\pi_j && (3)\\
&\text{subject to} && \forall i \in C, j \in A && \lambda_i + \pi_j \le w_{ij} &&\\
& && \forall j \in A && \pi_j \ge 0 &&
\end{aligned}$$

Let $(x^*_{ij} | i \in A, j \in C)$ denote the optimal solution to the primal program, and $(\lambda^*_i, \pi^*_j | i \in A, j \in C)$ denote the optimal solution to the dual program. We claim that by setting $\log(P_j) = \pi_j$, the resulting assignment of clients to APs corresponds to the assignment of x^*_{ij}'s. In other words, by setting $P_j = e^{\pi_j}$, client i will be assigned to AP j for which $\frac{P_j}{d^\alpha_{ij}}$ is maximized, and this assignment is consistent to the mapping as specified by x^*_{ij}.

To prove the above claim, we first make an observation that by setting $\log(P_j) = \pi_j$, client i will be assigned to AP j for which $\frac{P_j}{d^\alpha_{ij}}$ is maximized. This is equivalent to that client i is assigned to the AP j for which $\ln \frac{P_j}{d^\alpha_{ij}} = \pi_j - w_{ij}$ is maximized (or equivalently, $w_{ij} - \pi_j$ is minimized). Then we show this assignment is consistent to the assignment specified by x^*_{ij}. From the dual program, it is clear that $\lambda_i = \min_{j\in A}(w_{ij} - \pi_j)$. From complementary slackness conditions, $x_{ij} > 0$ if and only if $\lambda_i + \pi_j = w_{ij}$. Thus, after this power assignment client i is assigned to AP j if and only if $x_{ij} > 0$. Therefore the power assignment realizes the minimum weighted matching assignment.

Next we consider the other case, where there exists an assignment that fills all APs' capacity, but does not satisfy all clients' demands. We can use the following linear program to specify the minimum weighted perfect matching problem:

$$\begin{aligned}
&\text{minimize} && && \sum_{i\in C, j\in A} w_{ij}x_{ij} && (4)\\
&\text{subject to} && \forall i \in C && \sum_{j\in A} x_{ij} \le 1 &&
\end{aligned}$$

$$\forall j \in A \qquad \sum_{i\in C} x_{ij} = C_j$$
$$\forall i \in C, j \in A \quad x_{ij} \geq 0$$

The rest of the proof is similar to the first case. We can write the dual of the above linear program, and find the optimal set of powers to fill the capacity of all APs. Again using complementary slackness conditions, we can prove that this power assignment realizes the minimum weighted matching assignment. ■

2) Finding the Power Assignment.: The previous section described how to assign clients to APs to achieve maximum throughput. Below we develop two power assignment algorithms that enforce the client-to-AP assignment derived above.

a) An Algorithm Based on the Linear Programming: Below is the algorithm to compute power assignment for APs using a linear program. The proof of its correctness is essentially in the proof of Theorem 1.

FindPowers1 Algorithm

1) Solve the following linear program (Linear Program 3).

$$\begin{array}{lll} \text{maximize} & & \sum_{i\in C}\lambda_i + \sum_{j\in A} C_j \pi_j \\ \text{subject to} & \forall i \in C, j \in A & \lambda_i + \pi_j \leq w_{ij} \\ & \forall j \in A & \pi_j \geq 0 \end{array} \quad (5)$$

2) Let $(\{\lambda_i^* | i \in C\}, \{\pi_j^* | j \in A\})$ be the optimal solution to the above linear program.
3) Set $P_j = e^{\pi_j^*}$ for all APs j.
4) Scale all powers by the same factor such that $P_j \geq M_j$ where M_j is the minimum power at which AP j can reach all the clients that it has to serve.

b) A Combinatorial Algorithm: Next we design a combinatorial algorithm to find the power assignment that enforces the client-to-AP assignment derived in Section IV-A.1. Assume that we are given the client to AP assignment x_{ij}'s for $i \in C, j \in A$ of clients to APs. Let P_j denote the set of powers AP j use to realize the given assignment. If $x_{ij} = 1$, $\frac{P_j}{d_{ij}^\alpha} \geq \frac{P_k}{d_{ik}^\alpha}$ for any AP k. By setting $\pi_k^* = -\ln(P_k)$ and $w_{ik} = -\alpha \ln(d_{ik})$, we know that $x_{ij} = 1$ if and only if $-\pi_j^* + w_{ij} \geq -\pi_k^* + w_{ik}$ for all clients i and APs j and k. For an AP $j \in A$, let f_j be the farthest client in C that is connected to j ($x_{f_j j} = 1$). Let M_j be the minimum transmission power at which an AP j can reach client f_j and let $m_j = -\ln(M_j)$ The power of AP j should be no less than M_j, i.e., $\pi_j^* \leq m_j$. Thus, a set of powers results in the desirable assignment if and only if it satisfies the following inequalities:

$$\forall i \in C, j \in A : x_{ij} = 1,\ \forall k \in A \quad -\pi_j^* + \pi_k^* \geq -w_{ij} + w_{ik}$$
$$\forall j \in A \qquad \pi_j^* \leq m_j$$

We note that the above set of inequalities correspond to a polytope on which we can optimize any linear function as a linear program. For example, if we want to find a set of powers such that the sum of the logarithm of power is minimized, we can solve the following linear program:

$$\begin{array}{rl} \text{maximize} & \sum_{j\in A} \pi_j^* \\ \text{subject to} & \\ \forall i \in C, j \in A : x_{ij} = 1, \forall k \in A & \pi_j^* - \pi_k^* \leq w_{ij} - w_{ik} \\ \forall j \in A & \pi_j^* \leq m_j \end{array} \quad (6)$$

The above linear program can be solved combinatorially using the shortest path algorithm. This is more efficient than solving a linear program. For example, Dijkstra's algorithm can find the shortest paths in $O(|V|^2)$, where $|V|$ is the number of vertices in the graph. We convert this problem into finding the shortest paths as follow. We construct a directed graph $D(A \cup \{r\}, E)$, where A is the set of APs, r is an extra root vertex, and E is the set of edges between them. The length of edges in graph D are as follows: there is an edge from each vertex $j \in A$ to r with length $l_{jr} = m_j$. If client $i \in C$ is assigned to AP $j \in A$, we put an edge from $j \in A$ to k in graph D of length $l_{jk} = \min_{i\in C: x_{ij}=1}(w_{ij} - w_{ik})$. Let p_j be the shortest path from vertex j to r in graph D. In fact, the inequalities in the program are the triangle inequalities for the shortest path to the root r. Thus, it is not hard to see that p_j's satisfy all inequalities of Linear Program 6.

Hence, we have the following combinatorial algorithm.

FindPowers2 Algorithm

1) Given an instance of the power control problem as specified in Section III, we can construct the following weighted bipartite graph $G(A, C, E)$, where A is the set of APs, C is the set of clients, and E is the set of edges between them. There is an edge between each AP i and each client j. The weight of the edge from AP i to client j is equal to $w_{ij} = \alpha \ln(d(i, j))$.
2) Find the minimum weight bipartite matching in G, where the capacity of every client is 1, and the capacity of an AP i is C_i. In other words, among all the maximal assignments of clients to access points in which a client can be assigned to at most one AP and an AP i can be matched to at most C_i clients, find the one with minimum total weight.
3) Construct a directed graph $D(A \cup \{r\}, E)$. For two APs j and k, set $l_{jk} = \min_{i\in C: x_{ij}^*=1}(w_{ij} - w_{ik})$. For an edge jr from AP j to r, set $l_{jr} = m_j$.
4) Set p_j as the shortest path from AP j to r in graph D.
5) Set the power of AP j, $P_j = e^{-p_j}$.

The Algorithm **FindPowers2** outputs a set of powers that enforces the most efficient assignment. The proof of correctness of this algorithm is from Theorem 1, and the fact that the shortest paths to vertex r in graph D is a feasible solution to Linear Program 6. We will give a formal of proof of this fact in the proof of Theorem VII in Section 3.

The main advantage of Algorithm **FindPowers2** over Algorithm **FindPowers1** is that this algorithm is combinatorial, and has a better running time. Moreover, as we will show in Section VII, Algorithm **FindPowers2** can also be applied to optimize the sum of the logarithms of powers of APs, while maximizing throughput.

3) Multiple Preferred APs: In the previous sections, we proved that when each client selects the AP with the maximum RSSI, the set of powers from Algorithms **FindPowers1** and **FindPowers2** maximizes throughput. However, for a given set of powers, it is possible for a client to have multiple APs with similar RSSI. We call all these APs as this client's preferred APs. In such a case, a client will randomly choose among these preferred APs, and the performance may degrade, since the client may choose an AP other than the one in the assignment derived above. To handle this case, we use Algorithm **FindPowers2** to enforce stronger inequalities, i.e., instead of the inequality $\pi^*_j - \pi^*_k \le w_{ij} - w_{ik}$, we can put the inequality $\pi^*_j - \pi^*_k \le w_{ij} - w_{ik} - \beta$, where $\beta > 0$ is a given threshold (β represents the smallest signal strength difference a client can sense). The advantage of these stronger inequalities is that it ensures each client has a unique preferred AP, and the performance degradation caused by random tie breaking is avoided. We note that this change to the linear program may make it infeasible due to stronger inequalities. But this is a useful heuristic, which we will use in our evaluation, to find a set of powers that yield a unique assignment. When the stronger inequalities cannot be satisfied, we then use the random tie breaking for assigning a client that has multiple preferred APs.

B. Maximizing Throughput for Heterogeneous Demands

In this section, we develop a power control algorithm for heterogeneous client demands. We consider two cases: splittable and unsplittable demands. Under unsplittable demands, we gain the benefit of satisfying a demand only if we satisfy this demand completely. This setting is motivated by real-time services, e.g., the video streaming. In these services, if the demand cannot be completely satisfied, it is better not to service the demand, because the video requires certain bandwidth to achieve an acceptable performance. In the case of splittable demands, the throughput from a demand is proportional to the fraction of the demand that is provided by APs. The main application of this setting is in the best-effort services such as web browsing. In these settings, we can derive a benefit even if we cannot transfer files at a desirable data rate.

1) Unsplittable Heterogeneous Demands: It is not hard to see that under unsplittable demands the problem of maximizing throughput is NP-complete, since the assignment problem is a multiple knapsack problem [10]. In fact, a polynomial time $1+\epsilon$-approximation (PTAS) is known for the multiple knapsack problem [10]. We observe that the power assignment problem to maximize throughput for the unsplittable heterogeneous demands is APX-hard. The proof of this fact is via a reduction from the generalized assignment problem(GAP) [10], where each item can be assigned to a subset of bins (and not to all of them). We can reduce an instance of GAP to the power assignment problem by putting very large distances between the items and bins that cannot hold these items. The details of this reduction is omitted in the interest of brevity.

Here, we present an algorithm based on linear programming. This algorithm solves the problem approximately when the number of clients is much larger than the number of APs. Let D_i denote the demand from client i. The linear program formulation in Section IV-A changes to:

$$\begin{aligned} \text{minimize} \quad & & \sum_{i\in C, j\in A} w_{ij} x_{ij} \qquad (7)\\ \text{subject to} \quad & \forall i \in C & \sum_{j\in A} x_{ij} = 1\\ & \forall j \in A & \sum_{i\in C} D_i x_{ij} \le C_j\\ & \forall i \in C, j \in A & x_{ij} \ge 0 \end{aligned}$$

The dual program becomes:

$$\begin{aligned} \text{maximize} \quad & & \sum_{i\in C} \lambda_i + \sum_{j\in A} C_j \pi^*_j \qquad (8)\\ \text{subject to} \quad & \forall i \in C, j \in A & D_i\lambda_i + \pi^*_j \le w_{ij}\\ & \forall i \in C & \lambda_i \ge 0 \end{aligned}$$

We can show that when the number of clients is much more than the number of APs, Linear Program 7 has solutions in which most of the x_{ij}'s are either 0 or 1. These solutions are simply the corner points of the polyhedra. We call them extreme point solutions. They are also called basic feasible solutions. We use the following algorithm to find power assignment.

FindPowers3 Algorithm for heterogeneous demands

1) Find the optimum extreme point solution x^*_{ij} to the Linear Program (7), and its corresponding dual

optimum λ_i^* and π_j^* to the dual Linear Program (8).
2) Set $P_j := e^{\pi_j}$.
3) Connect every client i to the AP j for which $x_{ij}^* = 1$ if such j exists. Otherwise do not serve i.
4) Scale all powers by the same factor such that $P_j \geq M_j$, where M_j is the minimum power by which AP j can reach all the clients that it has to serve.

As we noted before, unlike the Linear Program 2, the primal Linear Program (7) does not always have an integral (0 or 1) solution. In other words, it might be the case that for some i and j, $0 < x_{ij}^* < 1$. We will say that client j is assigned integrally if $x_{ij}^* = 1$ for some i. Otherwise, we will say that it is fractionally set.

The following facts are implied by the theory of linear programming. The proof can be found in [34].

Lemma 1: The extreme point optimum solution to the primal program $\mathbf{x}^*$ assigns at least $N-K$ clients to APs integrally, where N is the number of clients, and K is the number of APs.

Proof: Let r denote the number of variables in the primal Linear Program 7. An extreme point solution is defined by the constraints in the linear program, where the inequality constraints are changed to equality constraints. Among these r independent variables, at least $r-K-N$ should be of type $x_{ij} \leq 0$. Their corresponding variables will be zero due to the last constraint in Linear Program 7. Therefore the number of non-zero x_{ij}^*'s are at most N + K. Let α and β denote the number of clients that are assigned integrally and fractionally, respectively. We have $\alpha + \beta = N$ and $\alpha + 2\beta \leq N + K$ (since for each client assigned fractionally, there are at least two non-zero x_{ij}^*'s). Therefore $\alpha \geq N - K$. ∎

In most cases, the number of clients is much larger than the number of APs. In that case even by dropping the clients that are assigned fractionally by the above program, the total satisfied demand is still close to the optimal.

The proof of the next lemma is similar to that of Theorem 1, and follows from the complementary slackness conditions.

Lemma 2: The assignment of clients to the APs defined by the optimum primal solution $\mathbf{x}^*$ can be achieved by setting the power of APs according to $P_j = e^{\pi_j^*}$. In other words, the optimal primal solution assigns the clients i only to the AP j for which the ratio $\frac{P_j}{d_{ij}^{\alpha}}$ is maximized.

Proof: From the dual program, it is clear that $D_i\lambda_i^* = \min_{j\in A}(w_{ij} - \pi_j^*)$. From complementary slackness conditions, $x_{ij}^* > 0$ if and only if $D_i\lambda_i^* + \pi_j^* = w_{ij}$. This means that $x_{ij}^* > 0$ if and only if $w_{ij} - \pi_j^*$ is minimized (or equivalently, $\pi_j^* - w_{ij}$ is maximized). Since $\pi_j^* = \ln P_j$ and $w_{ij} = \alpha \ln d_{ij}$, $x_{ij}^* > 0$ if and only if $\frac{P_j}{d_{ij}^{\alpha}}$ is maximized. ∎

2) Splittable Heterogeneous Demands: The algorithms for splittable heterogeneous demands is similar to that of the homogeneous demands.

Here we give two ways to solve this problem. The first algorithm is to split the demands into small uniform demands and use Algorithm **FindPowers2**. The second algorithm is based on solving the Linear Programs 7 and 8. As we noted in the previous section, the primal Linear Program 7 does not always have an integral (0 or 1) solution. However, as we allow splitting the demands, the fractional solution to Linear Program 7 is a valid solution. Therefore, we can use the optimal solution to the dual Linear Program 8 to enforce the most efficient assignment of clients to APs. The proof of correctness of this algorithm follows from the proof of Theorem 1.

V. Maximizing Throughput for a Discrete Set of Powers

In this section, we consider a variation of the problem in which the powers of APs can only take certain discrete values. This problem is motivated by the fact that APs from many vendors have only a handful power levels (e.g., Cisco Aironet [11]). In this case, the solution of our linear programming is not directly applicable because the power values computed by the linear program could be arbitrary fractional numbers. One approach to remedy this issue is to round the solution of our linear program to the closest discrete values that APs can take. However rounding may introduce significant performance degradation. In this section, we present an algorithm that finds the power assignments in a more direct way.

Assume that the power of an AP $a \in A$ can be set to one of the values from the set $\{P_1^a, P_2^a, ..., P_h^a\}$, where $P_1^a \geq P_2^a \geq \ldots \geq P_h^a = 0$. Our algorithm starts by setting the power of all APs to the maximum power level, P_1^a; then it tries to improve the solution in every step as follow.

FindPowers4 Algorithm for discrete powers
1) Assign the maximum power P_1^a to each AP a.
2) *while* there exists an AP a of power $P_{i_a}^a$, $1 \leq i_a \leq h$, such that the AP cannot accommodate all the demands assigned to it, we change the power of AP a to $P_{i_a+1}^a$.
3) Among all power configurations generated in the above step, choose the one that yields the highest throughput.

It is easy to see that the above algorithm is very efficient: the number of iterations in the *while* loop is at most hK. Therefore the algorithm has a polynomial running time. Next we prove the optimality of the algorithm under a certain condition, which is formally specified in Theorem 2.

Theorem 2: If there exists a power assignment such that each AP a has capacity to accommodate all the demands assigned to it, Algorithm **FindPowers4** finds such an assignment in polynomial time.

Proof: Let F be the feasible (optimal) power assignment. Suppose for $1 \leq i_a \leq h$, the power of AP a in F is $P^a_{i_a}$, and the Algorithm **FindPowers4** assigns AP a with power $P^a_{i'_a}$. It is easy to see that if $i'_a \leq i_a$, we find a power assignment in which all clients' demands are served without overloading APs (since the algorithm terminates at non-zero power only when it finds a solution in which all client demands are satisfied). Next we prove $i'_a \leq i_a$ holds. Suppose by contradiction, during the *while* loop, there is an AP a to which for the first time we assign a power $P^a_{i'_a}$ for $i'_a = i_a + 1$. Since the powers of all other APs are at least the power in the optimal power assignment, the total demands of clients that prefer AP a can be at most the total demands assigned to a in F. This cannot be more than its capacity according to the definition of F. Therefore it is a contradiction. ■

Note that the above theorem holds even in the case where the demands are heterogeneous and unsplittable.

We are assuming that for any power assignment to APs, every client has a unique preferred AP. When a client has multiple preferred APs (i.e., RSSI from multiple APs are equal or similar to each other), the client has a well-defined deterministic rule for breaking the tie. This tie-breaking rule could be different for different clients. This is a necessary condition, because sometimes it is impossible to set the powers of APs so that every client observes different signal strengths from different APs. Even if such a power assignment exists, it is NP-hard to find it. The proof of this fact is in our technical report [4].

VI. Dynamic Power Assignment

So far we examine how to control power to optimize throughput based on given client demands. When clients' demands are continuously changing, it is often desirable to find an assignment without requiring many clients to handoff to different APs, since the overhead of handoff is non-negligible. In this section, we develop a dynamic algorithm for this purpose.

We assume that a client will not switch to a different AP, unless its RSSI from a new AP is improved by a threshold. We define a client i to be happy if it is connected to an AP j, and the RSSI from j is at least $1/\gamma * max(RSSI_a)$ for all $a \in A$, where $max(RSSI_a)$ denotes the maximum RSSI received from all APs, and γ is larger than 1.

Our algorithm starts with the existing assignment of clients to APs, and finds a number of changes to the existing assignment so that all the clients are happy after the changes. We use the auction algorithms introduced in [9] to achieve this.

1) Start with the current power assignment and current mapping of clients to APs.
2) Repeat the following procedure until either all the clients are happy or all the APs are completely utilized:
   a) If a client i is not happy, it tries to find an AP j, for which $\pi_j - w_{ij}$ is maximized. It sends an association request to AP j.
   b) If an AP j receives an association request from a client i, it accepts the request when it has capacity. Otherwise, it sorts the clients that are connected or requested to connect in the decreasing order of their $\lambda_i - w_{ij}$. Let k be the highest index such that clients $1, 2, \cdots, k$ can be served by AP j. j accepts these clients, and sets its power to $\lambda_k - w_{kj} - \epsilon$.

At the end of the algorithm, it might be the case that the powers of all APs are decreased several times. We can re-normalize by multiplying all the power values by a constant δ. Clearly, this will not affect the assignment of clients to APs.

The main advantage of the above algorithm is that it tries to only make local adjustments to the existing connections. Moreover, since the changes in the powers are powers of γ, the algorithm converges to the right solution very quickly. Refer to [9] for a detailed analysis of auction algorithms.

VII. Power Optimization

In the previous section, we developed power control algorithms that maximize system throughput. In this section, we study how to simultaneously maximize system throughput and minimize APs' power. Power minimization is helpful to reduce interference among different APs. For ease of explanation, we consider homogeneous client demands. The same approach can be applied to splittable heterogeneous client demands.

First, we consider the problem of optimizing the power for a given mapping of clients to APs. In this case, we can write Linear Program 6, and optimize the power given the assignment of APs to clients. In the following theorem, we prove that the shortest paths to vertex r in graph D of Algorithm FindPowers2 are in fact the optimal solution to Linear Program 6. This in turn gives a combinatorial algorithm to optimize the sum of logarithms of powers for a given assignment.

Theorem 3: Let $(p_j | j \in A)$ be the length of the shortest path from vertex j to vertex r in graph D of Algorithm FindPowers2. Then p_j's are the optimal solution to the Linear Program 6.

Proof: Since p_k is the shortest path from k to r, $p_k \leq p_j + l_{kj}$ for any $j \in A$. So $p_j - p_k \leq w_{ij} - w_{ik}$, and the vector $(p_j | j \in A)$ is a feasible solution for the Linear Program 6. In order to show that this vector is the optimal solution to the Linear Program 6, we prove that for any feasible solution $(p'_j|j \in A)$, $p'_j \le p_j$ for any $j \in A$. We prove this by induction on the number of edges on the shortest path from j to r. If the number of edges on the shortest path to r is equal to 1, then $p'_j \le m_j = p_j$. Assume that $p_j \ge p'_j$ for all node j with the shortest path of size at most t edges between j and r, we prove that for a node k with the shortest path of $k+1$ edges from j to r. Since p_k is the shortest path, there exists a vertex k' for which $p_k = p_{k'} + l_{kk'}$. The size of the path from k' to r is at most t, thus $p'_{k'} \le p_{k'}$. As $(p'_j|j \in A)$ is a feasible solution, we know that $p'_k - p'_{k'} \le w_{ik} - w_{ik'}$ for any $i \in C$ for which $x_{ik'} = 1$. Thus $p'_k - p'_{k'} \le l_{kk'}$. Using these inequalities we get $p'_k \le p'_{k'} + l_{kk'} \le p_k + l_{kk'} = p_k$. This proves the induction step. ∎

A few comments follow. First, our power minimization is conditioned on maximizing throughput. This is achieved by ensuring the client-to-AP assignment is the same as that derived from Section IV-A or Section IV-B.

Second, we can use a similar approach to minimize the sum of APs' powers (while maximizing system throughput). This is done by minimizing the convex function $\sum_{j\in A} P_j = \sum_{j\in A} e^{-\pi_j^*}$ instead of minimizing $\sum_{j\in A} -\pi_j^*$ in Linear Program 6 using *interior point methods* (see e.g., [19]).

Finally, we note that minimizing the power while maximizing the system throughput is sometimes hard. We prove this by showing that finding an assignment of all clients to APs with minimum total power and maximum throughput is APX-hard. Refer to our technical report [4] for the proof.

VIII. Evaluation Methodologies

We evaluate the combinations of three AP power control schemes with two client association schemes.

- Basic power control: all APs are assigned the same fixed power.
- Continuous power control: the APs' power is determined by our power control algorithm, described in Section IV-B, for continuous power assignment.
- Discrete power control: the APs' power is determined by our power control algorithm, described in Section V, for discrete power assignment. The discrete power levels are based on Cisco Aironet 350 series [11]. It has the following 6 power levels: 20 dBm, 17 dBm, 15 dBm, 13 dBm, 7 dBm, and 0 dBm.
- Basic client association: a client associates with the AP that has the highest RSSI.
- Smart client association: a client associates with the AP with the maximum available capacity among all the APs whose RSSI exceeds its received sensitivity threshold.

Table VIII summarizes the five approaches that we compare, and their notations.

Name	AP	Client
basic/basic	fixed power assignment	RSSI-based association
basic/smart	fixed power assignment	load sensitive association
cont./basic	smart power assignment	RSSI-based association
discrete/basic	discrete power assignment	RSSI-based association
cont./smart	continuous power assignment	load sensitive association

TABLE I

The five schemes that we evaluate.

We use the total throughput as the performance metric. It represents the total amount of client demand that can be serviced. A higher throughput indicates a more efficient resource utilization, and hence is preferred.

We use both synthetic traces and real traces for our evaluation. Evaluation using synthetic traces gives us intuition about how the performance benefit varies with different parameters. We examine the impact of the following parameters:

- Total offered load: the ratio between the total client demand and the sum of all APs' capacity.
- The number of APs
- The distribution of client locations

We use two types of distributions to generate client locations: uniform distribution and normal distribution. When a normal distribution is used, we generate clients' x and y coordinates such that they each follows a normal distribution with the mean at the center of the area. Normal distribution reflects the case where clients are more concentrated in certain area. We vary the standard deviation in the normal distribution to generate different degrees of spatial locality.

In addition, we also use real traces to estimate the performance benefit of our power control schemes in a realistic environment. Table VIII shows the traces that we use in our evaluation. These traces cover a diverse set of environments: university campus, conference, and a large corporate. (Dartmouth traces span many campus buildings, and we use the traces from three buildings labeled as AcadBldg10, SocBldg4, LibBldg2 in their traces. We report the performance results for LibBldg2, and the results for the other two buildings are similar.)

We use the traces in the following way. All the traces record the amount of traffic generated from each client.

Location	Time	# APs	# clients
UCSD SIGCOMM traces [40]	Aug. 29-31, 2001	4	195
Dartmouth traces [12]	Mar. 2001	≈ 20/bldg.	variable
Stanford campus traces [38]	Sept. 1999	12	74
IBM traces [20]	Aug. 2002	variable	variable

TABLE II

Four traces used in our evaluation

For every 5-minute time interval, we compute the average data rate for each client, and use it as the client demand. In order to examine the impact of different load conditions, we also scale the traffic so that the total offered load varies from 25% to 100%. During the scaling, we try to maintain the relative data rate from different clients; for cases when a client's demand after scaling exceeds an AP's capacity, we split the demand into multiple clients, each assigned with at most 2Mbps. Dartmouth and UCSD traces both record the APs' locations, so we use them for placing the APs. For the other two traces, we randomly place the APs in a 500m*500m area. Since none of the traces record clients' location, we have to synthetically generate clients' location. As before, we use both uniform and normal distribution for placing the clients. Therefore the use of real traces mainly allow us to explore how realistic traffic distributions among different clients affect the performance of cell breathing.

Unless otherwise specified, we use the signal attenuation factor $\alpha = 4$, and AP's capacity is 5 Mbps, which approximates the data rate in 802.11b after taking into account of the MAC overhead.

IX. Evaluation Results

In this section, we present our evaluation results using both synthetic and real traffic traces.

A. Homogeneous client demand

First, we evaluate the different schemes using homogeneous client demand. In our evaluation, we randomly place clients and APs in a 500m*500m area, and all clients generate 1Mbps traffic.

Figure 1 shows the total throughput as we vary the offered load. We make the following observations. First, our power control schemes (both discrete and continuous assignments) out-performs the common practice (basic/basic), which uses a fixed power and lets the clients select APs based on RSSI. The performance of our approaches is close to that of using the smart AP selection (basic/smart and cont./smart), which serve as the upper bound. Second, the continuous assignment yields better performance than the discrete assignment, since the latter has more limited power choices. (Note that it is not guaranteed that there exists a discrete power assignment that results in maximum throughput.) Third, the cont./smart overlaps with basic/smart, which suggests that the AP power control scheme does not interfere with the smart AP selection implemented at the clients. Finally we observe that the performance benefit of the smart AP selection and our power control schemes tends to increase with the offered load. This is consistent with our expectation, since load balancing is more useful for high load situations.

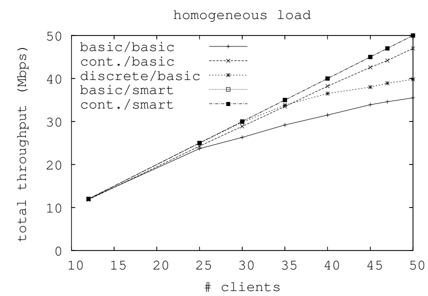


Fig. 1. Performance comparison under varying offered load, where 10 APs are deployed, and each client generates 1Mbps demand.

Figure 2 shows the total throughput as we vary the number of APs deployed in the area. The benefit of load balancing achieved using either power control or the smart AP selection increases with the number of APs. This is because when the number of APs increases, it is more likely to have a lightly loaded AP nearby to absorb some load from overloaded APs. In addition, the curves of continuous and discrete assignments overlap, both of which are close to the performance of the smart AP selection.

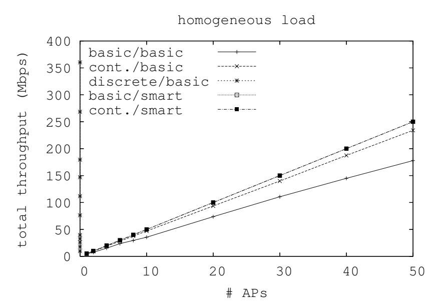


Fig. 2. Performance comparison for a varying number of APs, where the offered load is 1, and each client generates 1Mbps demand.

Next we examine the impact of the distribution of client locations. Figure 3 shows the performance as we vary the standard deviation (in a normal distribution), which is used to generate client locations. Note that a smaller standard deviation indicates a stronger spatial locality in the client load. As we can see, for small deviations (i.e., most of clients are concentrated in a certain area), the throughput under the basic scheme is much lower than the sum of APs' capacity. This indicates inefficient resource utilization. In comparison, the load balancing via continuous power assignment improves throughput by up to a factor of 9. The performance benefit of discrete power control is lower, but still significant: it often doubles the throughput in such cases. When the clients are more evenly distributed, the performance benefit of load balance reduces, since in such cases APs' load is already evenly distributed even under the basic scheme. Finally, as before, the smart AP selection works equally well with and without the power control at the APs. Therefore in the remaining evaluation, when clients apply the smart AP selection, we only consider APs' using a fixed power (since the performance of APs' using power control is similar).

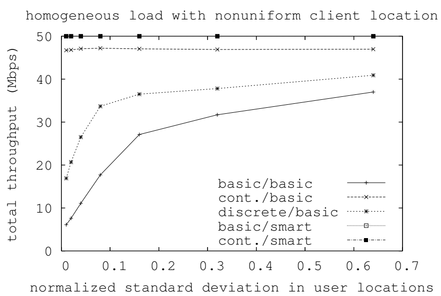


Fig. 3. Varying sigma (10 APs, offered load=1, demand per client=1).

B. Real WLAN traces

In this section, we present the performance results based on real WLAN traces.

Figure 4 shows time series plots of the performance results for the four WLAN traces. In all cases, our continuous power control algorithm achieves similar throughput as the smart AP selection scheme, and significantly outperforms the basic scheme. The discrete power assignment performs slightly worse than the continuous power assignment due to limited flexibility in power selection. However its throughput is still considerably better than that of the basic scheme.

To examine the impact of different offered load, we scale all clients' traffic by a factor. As shown in Figure 5, when the network is lightly loaded, all the schemes yield comparable performance; when the network is heavily loaded, the three load balancing schemes achieve significantly higher throughput than the basic scheme, by up to 50%. In addition, the performance difference between the continuous and discrete power assignments increases as the load increases. This is because during a high load, the number of good power assignments is fewer, which makes the discrete assignment harder to find them due to limited power choices.

We further study how the distribution of client locations affect the performance. Figure 6 summarizes the results. The performance benefit of power control scheme is significant, by up to an order of magnitude of throughput improvement. The improvement is larger when clients are unevenly distributed (i.e., small standard deviation). This is for the same reason as described in Section IX-A.

C. Summary

To summarize, in this section we evaluate our power control algorithms using both synthetic and real WLAN traces. Our results show that our power control can significantly out-perform the popular fixed power schemes, and perform comparably to the smart AP selection that require cooperation between clients and APs. Moreover, the performance benefit is highest for an uneven spatial distribution of client load. Such scenarios are quite common in practice because clients tend to localize themselves in particular areas (e.g., classrooms, meeting rooms, airport gate area with departing flights). These results demonstrate the effectiveness of the cell breathing approach for handling sporadic congestion and improving resource utilization.

X. Conclusion

We have developed a set of load balancing algorithms for handling sporadic client congestion in a wireless LAN. Our algorithms provide capacity where it is needed, and when it is needed. Consequently, more clients are satisfied and the overall utilization of the network is improved.

Existing solutions for handling congestion fall short since they either result in inefficient utilization of resources and poor performance, or require changes to the client software, which is hard to realize in practice. Our proposal, cell breathing, achieves load balancing by dynamically reconfiguring cell boundaries. It does not require changes to the client software or the standard, thereby making it rapidly deployable. Cell breathing is implemented by adjusting the power at each AP in the network. We show that our power control algorithms work for both homogeneous and heterogeneous client demands. In addition, the dynamic version of the algorithm can adapt to changes in client demands by maximizing the total satisfied demand while limiting the number of clients that switch APs.

We demonstrate the effectiveness of cell breathing, and show that it significantly out-performs the popular fixed power schemes and perform comparably to the sophisticated load balancing techniques where the client and the APs are required to cooperate with one another. Under high load, we show that with cell breathing the throughput improves by up to 50% for uniform distributions of client locations, and by up to an order of magnitude for non-uniform client distributions.

References

[1] Agere syste: Firmware update for orinoco pc cards v7.28 - spring 2001 release, April 2001.

[2] J. Ala-Laurila, H. Haverinen, J. Mikkonen, and J. Rinnemaa. Wireless lan architecture for mobile operators. pages 159–176, 2002.

[3] P. Bahl, A. Balachandran, A. Miu, W. Russel, G. M. Voelker, and Y.-M. Wang. Pawns: Satisfying the need for ubiquitous secure connectivity and location services. 9(1), February 2002.

[4] P. Bahl, M. T. Hajiaghayi, K. Jain, V. Mirrokni, L. Qiu, and A. Saberi. Cell breathing in a wireless lans: Algorithms and evaluation. *Microsoft Technical Report*, 2005.

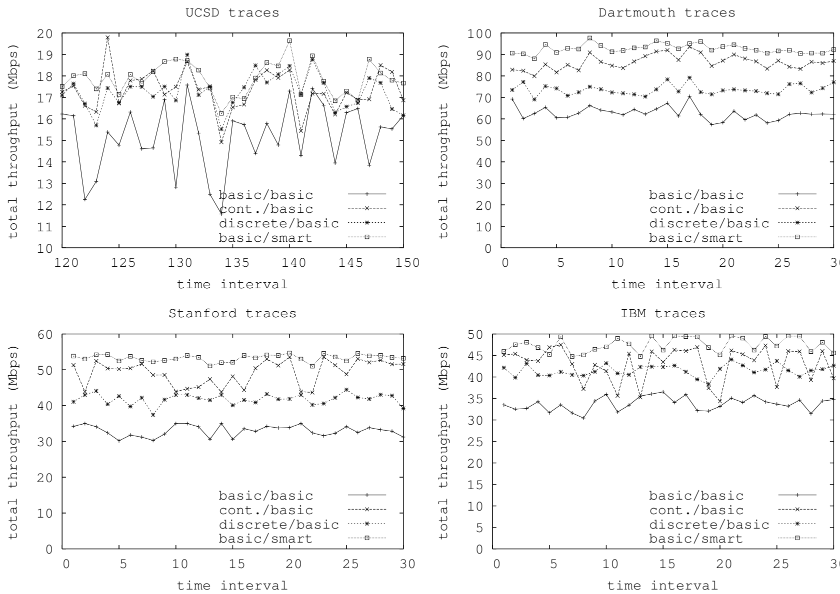


Fig. 4. Time series plots of performance results using real WLAN traces for uniform client locations. The offered load after scaling is 1.

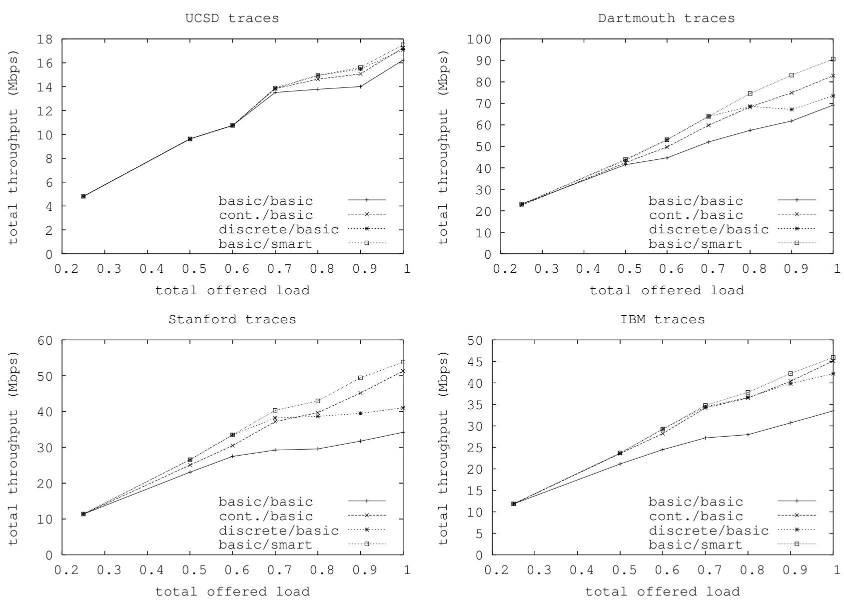


Fig. 5. Performance results using different WLAN traces under various offered load and a uniform distribution of client locations.

[5] A. Balachandran, P. Bahl, and G. M. Voelker. Hot-spot congestion relief in public-area wireless networks. In *SIGCOMM Comput. Commun. Review*, 2002.

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

[7] M. Balazinska and P. Castro. Characterizing mobility and network usage in a corporate wireless local-area network. In *Proc. of ACM MOBISYS*, May 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*, 2004.

[9] D. P. Bertsekas. Linear network optimization: Algorithms and codes. *M.I.T. Press, Cambridge, MA*, 1991.

[10] C. Chekuri and S. Khanna. A ptas for the mulitiple knapsack problem. In *SODA*, pages 213–222, 2000.

[11] Cisco aironet 350 series. http://www.cisco.com/warp/public/cc/pd/witc/ao350ap/.

[12] Dartmouth campus-wide wireless traces. http://www.cs.dartmouth.edu/ campus/.

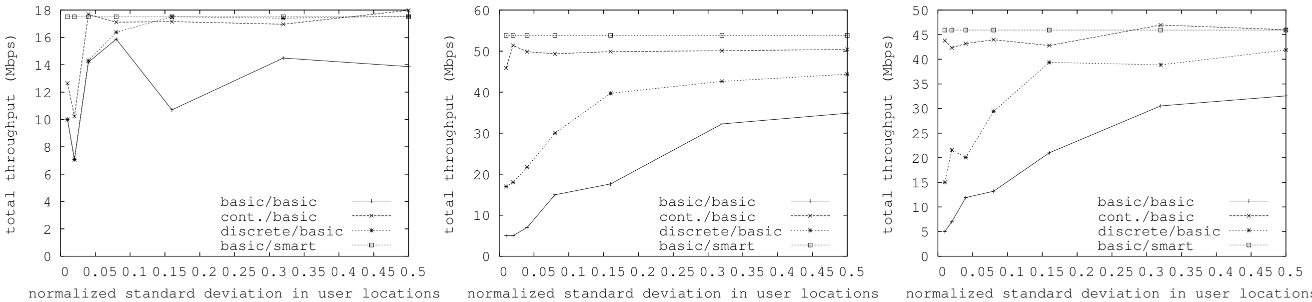
[13] N. Devanur, C. Papadimitriou, A. Saberi, and V. Vazirani. Market equilibrium via a primal-dual-type algorithm. In *FOCS*, 2002.

[14] L. Du, J. Bigham, and L. Cuthbert. A bubble oscillation algorithm for distributed geographic load balancing in mobile networks, March 2004.

[15] J. Edmonds. Maximum matching and a polyhedron with $0,1$-vertices. *J. Res. Nat. Bur. Standards Sect. B*, 69B:125–130, 1965.

[16] J. Edmonds. Paths, trees, and flowers. *Canad. J. Math.*, 17:449–467, 1965.

[17] Y. Fukuda, T. Abe, and Y. Oie. Decentralized access point selection architecture for wireless lans – deployability and robustness



(a) UCSD traces (b) Stanford traces (c) IBM traces

Fig. 6. Performance results using different WLAN traces for nonuniform client locations. We vary the standard deviation in a normal distribution used to generate the client locations

for wireless lans. In *Proc. of IEEE VTC*, September 2004.

[18] R. Garg and S. Kapoor. Auction algorithms for market equilibrium. In *STOC*, pages 511–518, 2004.

[19] H. Hindi. A tutorial on convex optimization. In *Proceedings of American Control Conference*, 2004.

[20] Wireless traces at ibm corporation. http://nms.lcs.mit.edu/ mbalazin/wireless/.

[21] K. Jain. A polynomial-time algorithm for computing an arrow-debreu market equilibrium for linear utilities. In *FOCS*, 2004.

[22] G. Judd and P. Steenkiste. Fixing 802.11 access point selection. In *ACM SIGCOMM Poster*, August 2002.

[23] V. Kawadia and P. R. Kumar. Power control and clustering in ad hoc networks. In *Proc. of IEEE INFOCOM*, Apr. 2003.

[24] F. Kelly. Mathematical modelling of the internet. *Mathamatics Unlimited - 2001 and Beyond, Springer-Verlag*, pages 685–702, 2001.

[25] D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *Proc. of ACM MOBICOM '2002*, Sept. 2002.

[26] S. Low and D. Lapsley. Optimization flow control, i: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking, Vol 7*, pages 861–875, 1999.

[27] S. Low, L. Peterson, and L. Wang. Understanding vegas: A duality model. *Journal of ACM, 49(2)*, pages 207–235, 2002.

[28] Prosafe 802.11g wireless access point - model wg302. http://www.netgear.com/products/details/WG302.php/.

[29] T. S. E. Ng and H. Zhang. Predicting Internet network distance with coordinates-based approaches. In *Proc. of IEEE INFOCOM*, Jun. 2002.

[30] I. Papanikos and M. Logothetis. A study on dynamic load balance for ieee 802.11b wireless lan. In *Proc. COMCON*, 2001.

[31] J. Qiu and J. Mark. A dynamic load sharing algorithm through power control in cellular cdma, September 1998.

[32] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proc. of IEEE INFOCOM*, Apr. 2000.

[33] A. Sang, X. Wang, M. Madihian, and R. Gitlin. Coordinated load balancing, handoff/cell-site selection, and scheduling in multi-cell packet data systems. In *Proc. of ACM MOBICOM*, Sept. 2004.

[34] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, New York, 1986.

[35] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003.

[36] D. Schwab and R. Bunt. Characterizing the use of a campus wireless network. In *Proc. of IEEE INFOCOM*, March 2004.

[37] S. Singh, M. Woo, and C. S. Raghavendra. Power aware routing in mobile ad hoc networks. In *Proc. of ACM MOBICOM*, Sept. 1998.

[38] Wireless lan traces from stanford university. http://mosquitonet.stanford.edu/software.html.

[39] D. Tang and M. Baker. Analysis of a local-area wireless network. In *Proc. of ACM Mobicom*, Aug. 2000.

[40] Wireless lan traces from acm sigcomm'01. http://ramp.ucsd.edu/pawn/sigcomm-trace/.

[41] V. V. Vazirani. *Approximation algorithms*. Springer-Verlag, Berlin, 2001.

[42] R. Wattenhofer, L. Li, P. Bahl, and Y. M. Wang. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In *Proc. of IEEE INFOCOM*, 2001.

[43] D. B. West. *Introduction to graph theory*. Prentice Hall Inc., Upper Saddle River, NJ, 1996.

[44] Wireless security auditor (WSA). http://www.research.ibm.com/gsal/wsa/.

APPENDIX

Appendix

A general algorithm for continuous-power assignment based on primal-dual

Now we describe a continuous power assignment algorithm for a more general received power function. Our only assumption is that the received power is proportional to the transmission power, which holds in general even under obstruction. We do not assume any relationship between the received power and the distance.

We observe that there is similarity between our problem and market equilibrium problem [13], [21], [18]. Market equilibrium has two popular settings. The one relevant to our problem is called Fisher setting. In the Fisher setting, there are two kinds of entities: *sellers* and *buyers*. Sellers want to sell a set of goods they have. Buyers want to buy a set of goods that they can afford and provides the maximum happiness, called *utility* in economics. Buyers naturally put a demand on goods from the various sellers depending upon the prices of the goods each seller set. The classical market equilibrium theorem says that under certain mild conditions sellers can set the prices so that the demand of their products is exactly equal to the supplies they have.

Our situation is quite analogous. Instead of buyers we have clients; instead of sellers we have APs; instead of a supply of goods we have a capacity on each AP; Instead of prices we have power levels at each AP. Since we have a simple setting: each client connects to the AP that gives the best reception, we discuss the simplest setting of market equilibrium, Fisher setting with linear utilities (i.e., each buyer's utility for a set of goods is a linear function).

There has been numerous work on market equilibrium problem with linear utilities. There are three kinds of algorithms currently known: (i) convex programming based [21], (ii) primal-dual based [13], and (iii) auction based [18]. The first kind of algorithms, theoretically has provided the fastest known running time and mathematically has been able to provide various properties of the market equilibrium. The drawback is that these algorithms need the input upfront. Auction based algorithms, on the other hand, are truly distributed. In between are the primal dual algorithms: they are not distributed, but still do not require the input upfront.

Here we describe a primal-dual type algorithm. The idea is inspired by [13], but note that there are specific difference between [13] and this. One major difference is the loop invariant. [13] makes sure that all the demands subsume the supplies. From that point onwards, the algorithm keeps trying to increase the prices and reduce the demands so that the demands still subsume the supplies; but not strictly, in other words total demand is equal to the total supplies.

Clearly if we have more capacity on the APs, demand cannot subsume the supply. If we have less capacity on the APs, the equilibrium does not even exist. If we have the total capacity equal to the total number of clients, we get the solution when we get the loop invariant for the first time. That is, if demand can subsumes supply, then the only way in this case is that demand is equal to the supply. So we cannot follow the loop invariant technique of [13].

Instead we start with an arbitrary assignment of positive powers to each AP. Suppose P is the power assignment vector. We define the *equality* graph as follow: one side of the equality graph include all the clients, and the other side include all the APs. Suppose we have n clients, and the total capacity on the AP's is at least n. Let j denote the client index, and i denote the AP index. We put an equality edge between i and j when i provides the best reception to j. Note that there can be more than one AP that provides the best reception to a client, but there is always at least one AP that provides the best reception to a client.

Theorem 4: If P is the equilibrium power, the equality graph has a complete matching for the clients, i.e., the size of the maximum matching is n. This means that the total throughput is maximized.

Next we prove the above theorem. Define the deficiency of a power assignment as the minimum number of clients remain unserved. In other words, the deficiency is n minus the size of the maximum matching in the equality subgraph. Suppose S is a set of clients. Define the neighborhood capacity of S as the total capacities of all those APs that have at least one edge from S. Suppose the neighborhood capacity of some set S is $|S| - k$. The deficiency of the power assignment is at least k. In fact, in every matching at least k clients from S itself remain unmatched. A well known fundamental theorem in the graph theory says that the converse is also true.

The following lemma can be proved in more than one way, and is a well known fundamental theorem in the graph theory. A special case of this theorem, where $k = 0$, is called Hall's theorem.

Lemma 3: If the deficiency is k, there exist a set S of the clients such that the neighborhood capacity of S is $|S| - k$.

This lemma clearly implies that in fact k unmatched clients belong to S. We take the smallest such S. By using the submodularity of the deficiency function or the supermodularity of the neighborhood capacity function, one can prove that there exists a unique such S. The intuition behind taking the smallest S is that we want to corner the unmatched k clients as much as possible so that we can do something for them.

Since S has k unmatched clients, and the neighborhood capacity of S is exactly k less than the clients in S, all the neighbor capacity will be assigned to S, and S still needs some more neighborhood capacity. In this case, we take all the APs not in the neighborhood of S, and start raising power on them. We do not raise powers arbitrarily. Instead we do it in a systematic fashion. We multiply the power of every AP not in the neighborhood of S by a variable x. We initialize $x = 1$. We start increasing the value of x gradually. The following facts can be easily proved by our power model of received powers.

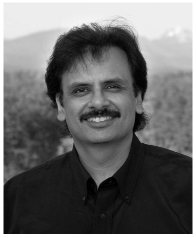
- All the edges from the complement of S to the neighborhood of S do not remain equality edges, so we remove them. Note that these edges are not needed in the first place.
- All other equality edges remain equality edges.
- Eventually some edge from S to the complement of the neighborhood set of S will be eventually added into the set of equality edges. At this point, we stop increasing x. We call it a phase.

The following lemma is self evident.

Lemma 4: After a phase, exactly one of the following two events will happen.

- The size of the smallest set with deficiency k has increased. In fact, the new smallest set with deficiency k contains S.
- The deficiency of the new power assignment has decreased. We call it an iteration.

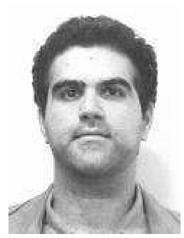
The algorithm terminates when there is no deficient set. Clearly the number of iterations in this algorithm is at most the number of clients, and in each iteration the number of phases is at most the number of clients. Hence the algorithm terminates in time $O(n^2)$ number of matching computations.



Paramvir (Victor) Bahl is a Principal Researcher and founding Manager of the Networking Research Group in Microsoft Research. He is responsible for directing research activities that push the state-of-art in computer networking. He and his group build proof-of-concept systems, engage with academia, publish papers in prestigious conference, publish software for the research community, and work with product groups to influence Microsofts products. His personal research interests are in wireless systems and mobile networking. In addition to building several seminal systems, he has authored over 75 papers and 65 patent applications (of which 22 have issued). Dr. Bahl is the founder and past-Chair of ACM SIGMOBILE; the founder and past Editor-in-Chief of ACM MC2R , and the founder and steering committee chair of ACM/USENIX MobiSys Conference; He serves or has served on the steering committees of IEEE DySpan, ACM SenSys, ACM MobiSys, ACM MobiCom, IEEE ISWC, IEEE COMSWARE and on the Program Committee of over 60 international conferences and workshops. He has served on the board of six IEEE and ACM journals and on several National Science Foundation and National Research Council panels. He received Digital's prestigious Doctoral Engineering Award in 1995 and SIGMOBILE's Distinguished Service Award in 2001. In November 2003, he became an ACM Fellow for "Contributions to wireless communication systems, and leadership in the mobile computing and communications community". Prior to joining Microsoft, he was with Digital (1998-97) where he shipped several seminal multimedia products. He received his PhD from the University of Massachusetts Amherst in 1997.

MohammadTaghi Hajiaghayi received the Bachelor's degree in Computer Engineering from Sharif University of Technology in 2000. He received the Master's degree in Computer Science from the University of Waterloo in 2001. From September 2001 to May 2005 , he was a Ph.D. candidate in Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. During his Ph.D. studies, he also worked at the IBM T.J. Watson Research Center (Department of Mathematical Sciences) and at the Microsoft Research Center (Theory group). Currently, Dr. Hajiaghayi is a Postdoctoral Fellow in the School of Computer Science at Carnegie Mellon University with ALADDIN project. Before this from June 2005 to December 2005, he was a Postdoctoral Associate in MIT Computer Science and Artificial Intelligence Laboratory (CSAIL). His research interests are algorithmic graph theory, combinatorial optimizations and approximation algorithms, distributed and mobile computing, computational geometry and embeddings, game theory and combinatorial auctions, and random structures and algorithms.

Kamal Jain is a senior researcher in Microsoft Research. He earned a doctorate degree from Georgia Institute of Technology in the multidisciplinary area of Algorithms, Combinatorics and Optimization in 2000. He is also an associate editor of Operation Research for Optimization and a vice chair for organizing the cluster on Network and Combinatorial Optimization for Informs 2006 and 2007. In his research carrier he has won several awards of excellence including the Optimization Prize. He has written research articles in several different areas of Computer Science, Electrical Engineering and Economics. Most notables of them are the algorithms for Networking problems, Facility Location problems, Network Coding problems and Cost sharing problems. Recently he has been working on auctions.



Sayyed Vahab Mirrokni is a postdoctoral researcher in the theory of computation group at Microsoft Research. He received his B.S. from Sharif University in 2001 and his PhD from Computer Science and Artificial Intelligence Laboratory and the Mathematics Department at the Massachusetts Institute of Technology in 2005. He is the co-winner of the best student paper award in the Symposium on Discrete Algorithms (SODA'05). He has worked at[LINK] Bell-Laboratories, Microsoft Research, T. J.Watson Research Center, and amazon.com. He has written research articles in several different areas of Computer Science, Discrete Mathematics and Economics. His research interests are in combinatorial optimization, algorithmic aspects of economics, mobile computing, and combinatorics.

Lili Qiu is an Assistant Professor in the Department of Computer Sciences at the University of Texas at Austin. Her research interests are in Internet and wireless networking, with special focuses on Internet measurement, overlay routing, content distribution, and wireless network performance and management. She is a recipient of NSF career award. Prior to joining UT, she spent four years as a researcher at Microsoft Research. She received her Ph.D. in Computer Science from Cornell University in 2001.

Amin Saberi received his PhD in the area of algorithms, combinatorics and optimization from Georgia Institute of Technology in 2004. He joined Stanford University as an assistant professor after a post-doc in Microsoft Research. He is interested in algorithms especially algorithmic game theory, approximation algorithms and their applications in the context of the Internet and World Wide Web.