Maximizing Spatial Reuse in Dense Wireless Networks

Paper ID: 1569193805

Abstract

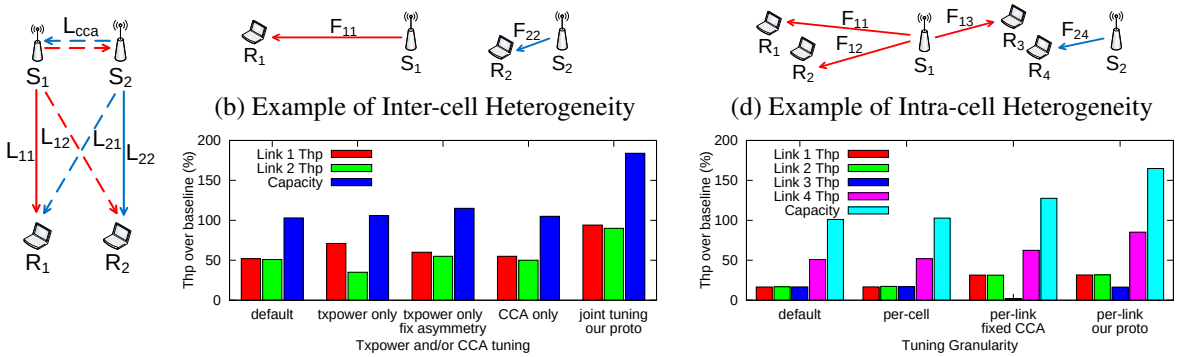
The availability of spectrum resources has not kept pace with wireless network popularity. As a result, data transfer performance is often limited by the number of devices interfering on the same frequency channel within an area. Spatial heterogeneity, which is caused by unplanned and unmanaged deployment of APs and clients, is a key characteristics of such dense and chaotic wireless networks. While various projects have exploited transmission power control and/or CCA threshold tuning, they are not capable of maximizing performance in such spatial heterogeneous networks. In this paper, we present the design of a protocol that determines appropriate power level and carrier sensing threshold for each link to maximize spatial reuse in heterogenous networks. This problem, however, is conditional NP-hard. Based on our observation of selecting the optimal configurations for a single pair of interfering wireless links, we deploy a heuristic algorithm that iteratively applies this pairwise optimization in dense wireless settings. We also describe a distributed protocol and its implementation in Linux using commercial 802.11 cards and an experimental evaluation using an 8-node testbed. Our results show that this protocol works well in practice and can improve the UDP performance over existing solutions by 40%, and 30% improvement for TCP performance.

1 Introduction

The popularity of wireless is fueling a rapid growth in the number of devices using the unlicensed frequency bands. The resulting dense and usually chaotic wireless deployments increasingly suffer from poor performance due to interference between the large number of devices sharing the same frequency band. This problem is exacerbated by the fact that the default transmit power (txpower) and carrier sense threshold (aka Clear Channel Assesment or CCA threshold) in 802.11 devices results in inefficient use of the spectrum.

Many projects [24, 25, 1, 21, 14, 15, 10, 16, 17, 23, 18, 11, 22, 19, 27, 31, 5, 29, 8, 2] have explored adjusting transmit power and/or CCA threshold to improve network capacity and/or reduce energy consumption in wireless networks. These two goals are however not always consistent so in this paper we focus on maximizing spatial reuse. Earlier solutions work well in homogenous networks, but fail to maximize the spatial reuse in chaotically deployed networks [1, 4], where spatial heterogeneity or diverse client-AP distances is a key characteristics. Figure 1(b)(d) show two examples of spatial heterogeneity. Figure 1(b) shows an example of inter-cell heterogeneity, where the two links F_{11} and F_{22} are heterogenous. Figure 1(d) shows an example of intra-cell heterogeneity, where the links F_{11},F_{12}, and F_{13} are heterogenous. As we will show in Section 2, existing solutions fail to maximize spatial reuse in these two example scenarios.

In this paper, we propose a practical, distributed protocol that simultaneously tunes power level and CCA threshold on a per-link basis, to maximize the spatial reuse. We make contributions in three areas. First, past proposals have used a range of optimization goals, e.g. minimize average packet latency, maximize battery lifetime, or minimize interference, but these goals are, at best, only indirectly related to improving spatial reuse. In contrast, our distributed protocol minimizes the number of edges in the communication conflict graph [20], which is a metric that directly maps to enabling more simultaneous transfers (i.e., increasing spatial reuse). Second, finding the optimal txpower alone for all transmissions is conditional NP-hard, and the complex interaction between txpower and CCA threshold further complicates the problem. We developed an iterative heuristic that reduces complexity considerably. We separately select ranges of txpower and CCA thresholds that optimize performance first, and then employ a separate algorithm that considers the interactions between the two parameters and chooses the best combination of values. Third, several practical issues need to be addressed before the protocol can be deployed on current commercial hardware. For example, the protocol must collect information about interference between transmitters, must calibrate RSSI and txpower levels of different nodes to ensure accuracy, and must ensure that the system is stable and incurs low overhead despite spurious measurements. We developed techniques addressing each of theses challenges, and we use a combination of experimental evaluation and simulation to show they address these practical considerations effectively. Our results show that our protocol perform well, providing a 40% improvement over exist-



(a) (c) Evaluation of (b) (e) Evaluation of (d)

Figure 1: Topologies (a) two flows in general, (b)(c)(d)(e) two example scenarios and evaluation.

ing solutions for UDP and 30% for TCP on a 8-node testbed.

The rest of the paper is organized as follows. We discuss the challenges of maximizing spatial reuse and review existing solutions in Section 2. Section 3 discrbes our algorithm for joint tuning of power levels and CCA thresholds. We describe our distributed protocol and its implementation in Sections 4 and 5. We present our evaluation using a testbed and OPNET in Section 6. We conclude and discuss some limitations of our protocol in Section 7.

2 Related Work and Challenges

Existing power control solutions can be categorized as 1) tuning txpower levels or CCA thresholds only, and 2) joint tuning of txpower levels and CCA thresholds. In this section, we review these existing solutions and present challenges for maximizing spatial reuse in heterogenous networks.

Tuning Txpower or CCA Thresholds Only We first review the protocols that only tune one of the two knobs of txpower and CCA threhold. Most work that only tunes txpower reduces the power level to the minimum necessary to decode frames at the receiver [24, 25, 1, 21, 15, 10, 16, 17, 23, 18, 22, 19, 29, 8, 2]. The minimum power is chosen for two reasons. First, it minimizes interference; however, this conclusion is based on a range-based model and, as we show in Section 3.1, this model is not accurate with existing hardware. Second, it minimizes energy consumption. In this paper, we do not focus on energy consumption. It has also been widely observed [25, 1, 15, 10, 16, 17, 23, 22, 19, 2] that using minimum power can cause unfairness or link asymmetry, and many previously mentioned systems are designed to deal with this problem. For example, recent work [22] detects such asymmetry and triggers power increase. Protocols that tune CCA thresholds only have also been proposed [27, 31]. In ECHOS [27], wireless nodes set their CCA thresholds to avoid collisions at its own receiver, but at the same time it may cause link asymmetry and exposed terminal problems. In [31], nodes set their CCA thresholds to balance the effect of exposed and hidden terminal problems.

These approaches, however, do not deal well with inter-cell heterogeneity, as shown in Figure 1(b). In this example, suppose only txpower tuning is deployed, no matter what power levels are used on S_1 and S_2, no concurrent transmission is possible because S_2 will always defer to S_1. Suppose only CCA tuning is deployed, no matter what CCA thresholds are used on S_1 and S_2, no concurrent transmission is possible because S_1 will cause collisions at R_2. The only way to allow concurrent transmissions in this scenario is to jointly tune txpower and CCA levels, which is the approach we adopt in our protocol. Note that this conclusion seems to conflict with the observations in [11], where the authors propose that tuning either txpower or CCA threshold has similar effects, but their conclusion only applies to the scenario presented in their analysis, e.g. a network where all AP are uniformly distributed (honey-grids). In contrast, we are interested in chaotically deployed networks.

We constructed the example scenario shown in Figure 1(b) on a testbed and used it to compare the performance of different approaches. We first obtained a baseline throughput by having only one of the sources transmit. We then ran experiments with both sources active, using default configuration, txpower tuning only (minimum power), txpower tuning only with link asymmetry fixed by power increase ([22]), CCA tuning only (ECHOS [27] with link asymmetry fixed), and our proposed protocol for joint power and CCA threshold selection. Note that except for our protocol, the results for other approaches are upper bounds, as we ignore

all overheads by emulating the protocol. The details of the experimental setup is described in Section 6. The results, as a percentage of baseline, are shown in Figure 1(c). We see that tuning txpower only (minimum power) suffers from slight link asymmetry, which can be fixed by increasing the power on F_{22}. The capacity for both txpower tuning only and CCA tuning only are similar to that of default, i.e. 100% and no spatial reuse. Our protocol enables concurrent transmissions in this scenario, increasing capacity to 185%.

Joint Tuning Systems that jointly tune power levels and CCA thresholds have also been proposed [11, 14], but they tune at a coarse granularity. For example, [11] assign the same CCA threshold to every node in the network, and [14] uses the same power/CCA configuration for all the nodes in the same cell. In [18], the authors conclude that coarse-grained approaches are asymptotically optimal, but spatial reuse can still be greatly limited when intra-cell heterogeneity is present.

One example of this limitation is shown in Figure 1(d). In the example, receiver R_3 is in a bad location since flow F_{13} interferes with F_{24}, but otherwise all flows can transmit simultaneously. In this example, concurrent transmissions can only happen without incurring starvation by a per-link protocol, because 1) without per-link txpower, as in [14], the same power level will be used on F_{11}, F_{12}, F_{13}, which imposes the same carrier sensing level on S_2, making S_2 unable to distinguish the three transmissions. S_2, in this case, can either use a high CCA that causes R_3 to starve, or it can use a low CCA that wastes the spatial reuse opportunities with F_{11}, F_{12}. 2) without per-link CCA, as in [11], S_1 can either use a high CCA that let F_{13} and F_{24} interfere with each other, causing F_{13} to starve, or it can use a low CCA which again wastes the spatial reuse opportunities. This example illustrutes that the performance of coarse-grained protocols is limited by the "worst" client.

We constructed this example scenario on our testbed, using the same setup as presented above. In this experiment, we ran the joint txpower and CCA tuning protocol on a per-cell [14], on a per-link basis with a fixed CCA threshold for all nodes [11], and our per-link protocol. The results are shown in Figure 1(e) (we include default configurations for reference, and again the results of the other approaches are upper bounds). The per-cell approach prevents concurrent transmissions, i.e. capacity is around 100%. While the per-link approach with fixed CCA yields higher capacity (125%), one link F_{13} suffers from starvation. Lowering the fixed CCA threshold on both senders fixes the link asymmetry, but the capacity is then almost the same as that of the per-cell approach. Our per-link protocol enables all concurrent transmissions in this scenario, yielding a capacity of 164%.

Enormous Search Space Size In summary, the goal of our solution is to maximize spatial reuse, and the approach we take is to jointly tune txpower levels and CCA thresholds on a per-link basis. The optimal solution to this problem for a cluster of nodes, however, is conditional NP-hard (conditional on Unique Games Conjecture, which this problem can be reduced to). The full proof is omitted due to space limits, but the basic idea is that these two problems have similar formats, i.e. to assign variables to find the maximum set of satisfied inequalities. Thus, the problem size is exponential in the number of transmissions.

3 Power Control and CCA Tuning

In this section, we describe an algorithm for per-link power control and CCA tuning on a collection of nodes. First, we introduce our interference model and the conflict graph data structure. As mentioned in Section 2, the problem is conditional NP-hard. In order to reduce the complexity, we use insight gained from a two flow scenario to develop an iterative heuristic that identifies ranges of txpower and CCA threshold values that optimize performance. Since this algorithm does not consider the interactions between carrier sensing and txpower levels, we then employ a separate power reallocation algorithm to chooses the best combination of values.

3.1 Modeling Interference

Many of the power control protocols mentioned in Section 2 rely implicitly or explicitly on a simple range-based model. In this model, every source is associated with a transmission and an interference range. Nodes within the transmission range of a source can decode frames from the source, and nodes within interference range will be prevented from transmitting due to carrier sense. The ranges depend on the power level each source uses. The range-based model predicts that using the minimum possible power level minimizes interference, as adopted in most power control protocols.

Another widely used model is the physical SINR model, which predicts that a packet can be successfully decoded if the signal to interference plus noise ratio exceeds some threshold. The SINR model predicts that uniformly increasing power levels increases system throughput by reducing the effects of thermal noise, though such gains are marginal in interference-dominated networks, e.g. dense 802.11 networks – exactly the opposite conclusion of the SINR model!

The right choice of model depends on whether the interface hardware exhibits the capture effect, i.e. the

ability to decode one transmission when multiple transmissions collide. In particular, it depends on how an incoming transmission is affected by a later frame with a higher signal strength. If the interface captures the stronger signal, then the SINR model is more accurate – otherwise the range-based model is a better predictor. Recent work [12], which we have confirmed in our own measurements, has shown that Atheros cards exhibit the capture effect. For Intel cards, capture effect can be indirectly supported by setting the CCA threshold to a relatively high value [30]. As the receiver threshold is tied to the CCA threshold on Intel cards, it would ignore an interfering signal that is below the CCA threshold.

Unfortunately, the SINR model is difficult to work with because each transmission interacts with all other transmissions. Thus, for deriving our protocol, we simplify the SINR model by making two assumptions: 1) interference is dominated by the strongest source (pair-wise interference assumption), and 2) we can ignore thermal noise. Measurement studies [3, 13] suggest that the pair-wise assumption is reasonable and we can ensure the second assumption by keeping power levels high enough. Note that these assumptions are used to simplify our system, but are not fundamental in our protocol.

To concisely represent the interference that in a network, researchers have used a conflict graph [20]. Each vertex in the conflict graph represents a link in the wireless network and there is an undirected edge between two links if the two links interfere, i.e. they cannot be active at the same time. Clearly the conflict graph depends on the transmit power used by the nodes. We construct the conflict graph based only on the pair-wise SINR model described above. As mentioned before, the goal of our work is to increase spatial reuse, which is beneficial for almost all practical network deployments. Since the lack of an edge in conflict graph is equivalent to enabling concurrent transmissions, our goal is equivalent to minimizing the number of edges in the conflict graph.

3.2 A Two-Flow Scenario

To gain insight into choosing the right power levels and CCA thresholds, we first consider a two-flow scenario. In Figure 1(a) S_1 transmits to R_1 and S_2 transmits to R_2. We use L_{ij} to denote the path loss from S_i to R_j $(i, j \in \{1,2\})$, and P_i, CCA_i to denote the transmit power level and CCA threshold from S_i to R_i. Note that L_{ij} can be arbitrary, i.e. we do not assume pathloss obeys the triangle inequality. Thus the SINR at receivers R_1 and R_2 are $SINR_1 = P_1 - L_{11} - P_2 + L_{21}$ and $SINR_2 = P_2 - L_{22} - P_1 + L_{12}$, respectively. Note that independent of the transmit power levels, we have $SINR_1 + SINR_2 = L_{12} + L_{21} - L_{11} - L_{22}$. Power control essentially allocates this sum between the two transmissions, i.e. increasing $SINR_1$ will decrease $SINR_2$. In order to enable concurrent transmission, we need both $SINR_1 \geq SINR_{thrsh}$ and $SINR_2 \geq SINR_{thrsh}$. Also, CCA thresholds are to be set to enable such transmissions, i.e. $CCA_1 > P_2 - L_{cca}$ and $CCA_2 > P_1 - L_{cca}$. Note that it may not be possible to satisfy the SINR constraints, indicating that concurrent transmissions are impossible. Thus in this case, txpower can be set arbitrarily and CCA thresholds should be set to $CCA_1 < P_2 - L_{cca}$ and $CCA_2 < P_1 - L_{cca}$.

Algorithm 1: Calculate power levels to allow concurrent transmissions (*power_range*)

Data: two links t, t'
Result: the range of power levels P_{min}, P_{max} on t such that t and t' can transmit together

1 $P' \leftarrow P(t')$
2 $L_{11} \leftarrow L(src(t), dst(t))$
3 $L_{12} \leftarrow L(src(t), dst(t'))$
4 $L_{21} \leftarrow L(src(t'), dst(t))$
5 $L_{22} \leftarrow L(src(t'), dst(t'))$
6 $P_{min} \leftarrow L_{12} - L_{22} - SINR_{thrsh} + P'$
7 $P_{max} \leftarrow L_{11} - L_{21} + SINR_{thrsh} + P'$

3.3 Select Power Level & CCA Threshold

In the previous section, we showed how to set the ratios of transmit power and CCA thresholds between two links to allow simultaneous transmission, if possible. However, if there are multiple links, the choice made for one link may not be compatible with the choices for other links. In fact, finding the optimal configuration is conditional NP-hard. Based on the insight from the two-flow scenario, we separate power control and CCA threshold tuning algorithms, and use a third algorithm to deal with the interactions between the two knobs. A centralized version of our algorithm is shown in Algorithm 2; our protocol uses a distributed version of this algorithm. The algorithm assumes that every node has complete knowledge of the network topology (i.e. path loss between nodes) and current configuration (i.e., power levels used by sources); we will discuss how to collect this information in the next section. Also, each link determines its configuration in turn, so our protocol does not depend on synchronization among links.

Basic Power Control Algoritm The core of our power tuning algorithm is a greedy heuristic that iteratively tunes power levels to allow more concurrent transmissions by removing edges from the conflict graph (line 2-16 in Algorithm 2). We use the following nota-

tions. $src(t)$,$dst(t)$ are the source and destination of link t while $P(t)$ is the power level currently used on that link. For any two nodes n,n', $L(n,n')$ is the path loss from n to n'. We assume that each wireless device has a range of discrete tunable power levels, e.g. -10dBm to 10dBm; in practice, power range is typically limited by noise consideration (lowerbound) and power restrictions / hardware limitation (upperbound).

In each iteration, and for each link t, the algorithm examines the power levels used on all other links and the topology to determine what power level would allow simultaneous transmission with the other links (Algorithm 1). It then picks the power level that allows the most concurrency (line 10). Usually there are multiple such power levels and we use power reallocation to choose among them, as described later. The new power level will be used if it allows more concurrent transmissions than that in the previous iteration (line 13-15). Note that after each iteration, the number of edges in the conflict graph decreases, and the algorithm converges when no more edges can be removed from the conflict graph. Also, by using this algorithm, the source that needs maximum power level will hit the power limit, and then all other sources will keep an appropriate ratio to that source.

CCA Threshold Tuning The CCA thresholds in our algorithm are set to prevent hidden terminals, while at the same time allowing possible concurrent transmissions (lines 17-23 of Algorithm 2). The criterion is that if one link interferes with another link or will be interfered with by another link (line 20), the CCA threshold is set to defer to the other link.

Note that the separation of power control and CCA tuning allows our algorithm to use other CCA tuning approaches. Though we observe in simulation that other approaches may deliver a higher network capacity, they sometimes cause starvation. Here we summarize several work. In [27], every source sets its CCA thresholds to avoid collisions at its receiver, and this local decision can cause starvation at other receivers. Wireless nodes in [31] aims to balance between exposed terminal and hidden terminal problems, and starvation is possible since hidden terminals are allowed. In [11, 5], every source uses a fixed product α (in watts) for txpower and CCA, which ensures the symmetry property [11], where either both nodes hear each other or neither can hear each other. Using this approach, setting a high α would cause some links to use too high a CCA threshold that starves other links, while setting α low may waste many spatial reuse opportunities. Note that the goal of our protocol is to maximize spatial reuse, i.e. allowing more concurrent transmissions, instead of trading fairness for higher network capacity, thus we do not use these protocols.

Algorithm 2: Select Txpower and CCA Threshold

```
1  while not stable do
2      /* v[i] counts how often concurrent transmission
       is possible with power level i for each t */
3      forall t do
4          clear(v)
5          forall t' ≠ t do
6              [P_min, P_max] ← power_range(t, t')
7              for i = P_min to P_max do v[i]++
8          end
9      end
10     Find the set of {P'_m} such that v[P'_m] is maximum
11     P_m ← power_reallocate({P'_m})
12     /* last[t] to ensure convergence: change txpower
       only when more concurrency can happen */
13     if v[P_m] > last[t] then
14         P(t) ← P_m
15         last[t] ← v[P_m]
16     end
17     forall t' ≠ t do
18         [P_min, P_max] ← power_range(t, t')
19         CCA' ← P(t') − L(src(t), src(t'))
20         if P(t) ∉ [P_min, P_max] and CCA(t) > CCA'
           then
21             CCA(t) ← CCA'
22         end
23     end
24 end
```

Power Reallocation As mentioned in Section 2, S_1 in Figure 1(d) should assign different txpower levels and CCA thresholds to F_{11}, F_{12}, F_{13}. This is done in line 11 in Algorithm 2, through a best-effort power reallocation process.

The basic idea behind power reallocation is to assign higher power levels to receivers that experience higher level of interference, e.g. R_3 in the example, while keeping the power levels within the ranges produced by the iterative algorithm. Note that the metric for the level of interference is the number of concurrent transmission opportunities not utilized due to carrier sensing if highest power is used for that link. The algorithm is executed locally on each sender, but only when the sender has multiple receivers, e.g. S_1 in the example. For example, in infrastructure networks, only APs need to execute it. To reallocate power levels, the sender first sorts all its receivers by the level of interference they experience (high to low) and then tries to improve spatial reuse by reallocating the transmit power according to the sorted order. To ensure convergence, it only picks power level from the set of $\{P'_m\}$ (line 10), even if this means that concurrent transmission opportunities cannot be increased.

4 Distributed Protocol

We now present our distributed power control and CCA tuning protocol based on the algorithm described in Section 3.

4.1 Data Collection

In order to create a local interference graph, nodes need information about the path loss on nearby links. This is obtained by having each source insert transmit power and path loss information in each packet, specifically, the power level used for the current packet, the path loss measurement to the destination, the path loss measurement to a randomly chosen third node, and a bit to indicate whether this is a high power frame (see below). The power level used for this packet will be used by the receiver to calculate the path loss. The sender's path loss measurement to the destination will be used by the receiver to help calibrate the power levels of the cards (Section 5). Finally, by including the path loss to a third node, we are effectively creating a gossiping channel that allows a source to learn the path loss of links they are not part of. This is used by the source to estimate the interference at its destinations. The overhead of extra information is less than 1% for a full-sized data packet.

Let us illustrate the data collection phase using the example in Figure 1(a), S_1 needs the following information to compute its configuration: $L_{11}, L_{12}, L_{21}, L_{22}, P_2$. P_2 can be extracted from S_2 transmissions. L_{11} can be measured when R_1 is transmitting, L_{12} can be measured when R_2 is transmitting, and L_{21} can be measured by R_2 when S_2 is transmitting and sent to S_1. Finally, L_{22} can be measured and inserted into a packet by S_2.

Each node in the system operates in promiscuous / monitor mode and upon receiving a packet, the node updates its topology information and configuration. For each of its active flows, a source node keeps several lists: T_{sd} to store current transmit power level, current CCA threshold and measured path loss for the source to the flow's destination; T_{so} to store path loss from the source to other nodes; T_{do} to store path loss from the destination to other nodes; and T_{oo} to store the power level, and path loss of other flows.

Periodic Announcement & Periodic Low CCA When a node uses a low power level, other transmitters may not be able decode its packets and collect the above information. To avoid this problem, nodes periodically (e.g. every 2 seconds) transmit announcements at the maximum power level. These announcements are piggybacked on small packets, such as beacons or ACKs, to reduce interference. A similar problem happens when a source uses a high CCA threshold and has a lot of packets to send: it will transmit a significant fraction of the time without deferring to other sources, thus preventing it from overhearing packets from other sources. Though this does not affect performance after the protocol converges, it will affect its ability to change its view of the topology, e.g. when a node moves or joins the network. In order to avoid this problem, each node periodically set its CCA threshold to low level (e.g. for 20ms every second).

Insufficient Measurements Our protocol relies on the collection of data in the network, but in some cases, not all information regarding a transmission is available, either due to protocol initialization or undecodable frames. For example, the source can hear a transmission, but the destination cannot. In this case, we take the conservative approach that assumes the signal level is just below the receiver sensitivity to avoid hidden terminals, i.e. path loss of 100dB (our system uses default txpower of 10dBm and sensitivity of -90dBm). This approach is used when only part of the topology information is collected. However, there are certain cases where one link cannot hear another interfering link at all. Note that since in our protocol, information is implicitly shared within a cell through AP, this indicates that no node in the cell can hear another link. This would lead to hidden terminal problems, and various solutions exist to detect this scenario, e.g. [28] detects packet collisions; and once detected, a two-hop broadcast may be used to fill the gap between the decodable and interference ranges.

4.2 Packet Transmission/Reception

When node S_1 is about to transmit a data or control frame to destination R_1, S_1 will first search for the power level, CCA threshold, and path loss for R_1. If such information is unavailable, it will use the default power level and CCA threshold, and it inserts an invalid value for path loss in the packet. This should only happen for the first frame between the sender and receiver, e.g. an association request in infrastructure network. After a few packet exchanges, S_1 will have initial measurements for power level, CCA threshold, and path loss, and will use those for the transmission. In addition, S_1 will also piggyback power level, path loss, and the picked entry from the T_{so} list onto the frame. The frame is then added into the device queue for transmission.

Upon receiving a packet from R_1, node S_1 measures the RSS of that packet and it extracts power level, path loss from R_1 to another node, and path loss from R_1 to its destination from the frame. The path loss from R_1 to S_1 is calculated by subtracting RSS from power level used for the packet. Then, S_1 will determine which lists

to update. If the packet is destined to S_1, or the packet is from its sender to another receiver, then it will update the T_{sd} and T_{do} lists. Otherwise, the node will update the T_{so} and T_{oo} lists. If there is an update to any of the lists, S_1 will recalculate the configurations for all its destinations.

4.3 Absolute Transmit Power

Algorithm 1 presented in Section 3 determines the *ratios* between the transmit powers for all the wireless links, i.e. uniformly shifting the power levels up or down does not affect the results. However, while absolute power levels do not affect the algorithm, the protocol generally benefits from using high power levels, since it allows nodes to learn topology more easily by overhearing packets. In practice, the power range is also limited by noise consideration (low end) and power limitations (high end). As a result, our protocol first calculates ratios and then assigns the maximum power level to the link requiring the highest power. The power for the other links are then derived using the ratios.

5 Implementation

We first present the ideal system requirements for implemening our protocol. We then present several techniques needed to implement the protocol on today's NICs: RSSI and txpower calibration, determining the CCA granularity, and smoothing RSSI readings.

5.1 Ideal System Requirements

We identify the hardware/firmware features needed to run our protocol and discuss how well they are supported in today's hardware.

Per-packet Txpower and CCA Threshold Control. Control over transmit power and CCA threshold is central to our design. To the best of our knowledge, the only publicly available card that supports this is the Intel 2915/2200 card with AP firmware and driver. Also, since our protocol works on a per-link basis, these controls need to operate on a packet granularity, at least for senders with multiple receivers. Unfortunately, the Intel 2915/2200 AP firmware does not support these functions stably on a per-packet basis. This weak support is probably a result of a lack of demand and future cards should be able to support per-packet reconfigurability.

Receiver Threshold Control. The receiver threshold is used by a radio to determine when to decode an incoming signal. If a signal is below this threshold, the radio is not activated. On Intel 2915/2200 hardware, the receiver threshold is coupled with CCA threshold [30]. As a result, when a node uses a high CCA threshold, it can no longer decode many low power transmissions, limiting its ability to gather topology information in our protocol. Even worse, the node can miss critical packets, e.g. an AP may miss association requests from a new client. We need the receiver threshold to be set independently from the CCA threshold. Again, this should be possible in future cards.

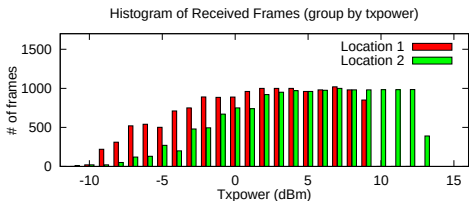
Accurate Signal Strength Measurements and Transmit Power Settings. Our protocol depends on accurate RSS measurements and precise control of the transmit power. Our experience shows that the RSSI readings from the Atheros cards and the transmit power setting of the Intel cards we used are fairly linear. However, as is confirmed by earlier work [6, 9], wireless cards are uncalibrated, both in terms of transmit power levels and RSSI readings. This is the case even for cards of the same model from the same vendor. As a result, changes in values are measured accurately but the absolute values of the readings cannot be compared across cards. Since nodes exchange both transmit power and RSSI values in our protocol, we need cards to be reasonably calibrated. Calibration by the manufacturers helps, but cards can still become uncalibrated over time due to drift. For this reason, we use automated calibration, as described in Section 5.2.

Implementation Setup. To mimic the behavior of an ideal system, we implemented our protocol in Linux using one Intel 2915 card and one Atheros 5212 card. This two-card setup is not fundamental and is only needed to work around the shortcomings of the current commercial cards. The Intel card supports transmit power and CCA tuning and it is used as the primary transmission interface. The Atheros card, with a default receiver threshold, is used to monitor ongoing traffic. By appropriately configuring the routing tables, the two interfaces are seen by applications as a single interface.

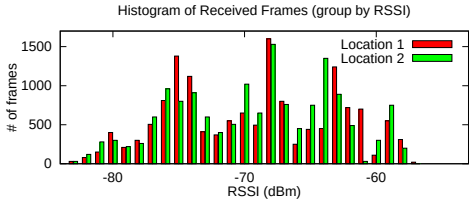
5.2 Practicality

Next, we present several mechanisms that are needed to make our protocol practical in the real world, including 1) calibration, which is fundamental to protocols that exchange RSSI readings and/or txpower levels, 2) CCA granularity, which is fundamental to protocols that tune CCA thresholds, and 3) smoothed RSSI readings, which is needed by our protocol to reduce computation costs.

Calibration Our protocol relies on precise transmit power and CCA control and accurate RSSI readings on network cards. However, previous work has observed



(a) Group by TxPower - Delta Pathloss



(b) Group by RSSI - Delta RSSI Threshold

Figure 2: Intuition of Automatic Calibration

that wireless cards are poorly calibrated [9, 6]. While the relationship between output power and transmit power value on each card is often linear, there is typically an offset; offsets of up to 20 dB have been observed in [6]. The relationship between actual RSS and RSSI readings is similar and offsets of up to 10dB have been observed in [9]. While manual calibration using a signal analyzer and generator is possible, it is not a practical solution in most deployments.

To formalize the calibration problem, let $\Delta txpow_i$ and $\Delta RSSI_i$ denote the difference between actual and reported transmit power levels and RSSI readings at node i, respectively. For our protocol, we need to calibrate the cards relative to one particular node i, i.e. solve $\Delta txpow_j - \Delta txpow_i$ and $\Delta RSSI_j - \Delta RSSI_i$ for any node j. Put it another way, after calibration, we only need to know the differences in the offsets between all nodes, and share the same unknown $\Delta txpow$ and $\Delta RSSI$. The two unknowns will not affect the result of our protocol because: 1) every pathloss will have an offset of $\Delta RSSI$, and this does not affect the calculation of SINR for either flow, 2) every txpower setting will have an offset of $\Delta txpower$, which doesn't change SINRs nor the conflict graph, 3) $\Delta RSSI$ and $\Delta txpower$ does not need to be equal because CCA thresholds are set based on observed RSS, which already includes an offset of $\Delta txpower$. Next we describe an automated calibration mechanism.

We first calibrate the RSSI readings and RSSI thresholds based on the assumption that for two cards, the difference in the receiver thresholds reflects the difference in uncalibrated RSSI readings. Intuitively, each node has its own RSSI vs. reception curve, and the idea is to calibrate one node's curve to that of others. This is done by having one node send packets with different transmit power levels and having all other nodes record the RSSI of the packets they received. In an ideal environment, the lowest RSSI reading on each receiver represents its RSSI sensitivity level, but in practice, we use the N lowest readings to improve robustness. Each node builds a histogram of these N lowest RSSI readings, and we then compare histograms pairwise with different offsets, and choose the offset with the closest distance, i.e. $argmin_k \sum_i |P_i - Q_{i+k}|$ for two histograms P and Q, which is the offset for RSSI. Figure 2 shows an example of transmit power and RSS histograms for the same device at two different locations. It shows that even though the device has a different txpower histograms at two locations, indicating that path loss to two locations are different, the RSS histograms, which reflects the RSSI thresholds, are roughly the same. Note that in order to compare the histograms, the whole SINR vs. reception curve on the receiver need to be sampled and otherwise it cannot be calibrated; there are two cases of insufficient sampling, if the receiver cannot receive any frame or it can receive frames from the lowest power levels. Since the offsets are transitive, multiple such calibration experiments can be carried out with different senders, and this process can be continued until all nodes are calibrated. This mechanism is susceptible to external interference and, thus, it should only be carried out when interference is expected to be low, e.g. early in the morning.

After the RSSI readings/thresholds are calibrated, we then calibrate the transmit power levels based on the law of channel reciprocity, i.e. the pathloss from node A to B is the same as that from B to A at each point in time. This means that nodes can detect the offset between their transmit powers by exchanging their views of the pathloss between them; the difference between the two reported pathloss values is the offset for txpower.

CCA Granularity and Txpower Spacing Measurement noise, calibration accuracy and the different ways carrier sensing are implemented [7] mean that we need to insert some slack between the RSSI level and CCA threshold for carrier sense to work reliably. For example, in the two-flow scenario in Section 3.2, while in theory setting $CCA_1 = P_1 - L_{cca} + \varepsilon$ can allow S_1 to transmit concurrently with S_2, in practice, we need to set $CCA_1 = P_1 - L_{cca} + c$, where c is greater than ε. We explore this CCA granularity issue here, focusing on identifying what CCA thresholds must be used to reliably defer or ignore competing signals.

We use Figure 1(a) to illustrate the problem. To determine how low S_1 should set its CCA threshold to correctly defer to another transmission, we let F_{22} transmit with a fixed low CCA threshold, and set the transmit power of F_{11} to interfere with F_{22}. Figure 3(Defer) shows F_{22}'s throughput for different CCA threshold settings on S_1. The throughput is plotted as a ratio to the baseline

link throughput and the offset is plotted as the difference between the average RSS and CCA threshold. The baseline is the throughput of F_{22} when S_1 uses a default low CCA threshold. It shows that with an offset -4, the ratio is close to one. This indicates that in order for S_1 to defer to transmission F_{22}, the CCA threshold on S_1 should be set 4dB lower than the interfering signal (2dB lower also works but we reserve 2dB more). To measure how high the CCA thresholds needs to be to ignore a signal, we let F_{22} transmit with a fixed high CCA threshold, and F_{11} and F_{22} are configured so that they can transmit simultaneously. The baseline here is the throughput of F_{11} when S_1 uses a very high CCA threshold. Figure 3(No Defer) shows the ratio of F_{11}'s throughput relative to the baseline for different CCA thresholds on S_1. It shows that the CCA threshold need to be 6dB higher than average RSS to achieve full spatial reuse (we again reserve 2 dB more).

A related issue is that in the power reallocation algorithm in Section 3.3, we assign different power levels to different links within a cell so that other senders can set their CCA thresholds to defer to some links and not defer to others. This indicates that we must reserve enough space between the txpower assigned to any two links, and the size of the reserved space is 10 dB in this case (difference between 6 and -4). For example in Figure 1(d), the power on F_{13} should be 10 dB higher than that on F_{12}. We carried out similar experiments on other senders and all had a granulatity of less than 10dB. Thus, we use a spacing of 10dB in our experiments. Note that this spacing is different from the granularity proposed in [26], where the authors propose that only several discrete power levels are useful depending on the level of multipath. That work is orthogonal to our solution and can be integrated into our system.

Smoothing RSSI Readings RSSI readings will always vary over time due to noise, interference and environmental changes. In our protocol, rapidly changing RSSI readings are undesirable because 1) they can cause network instability, and 2) they can trigger excessive computation and reconfiguration on each sender. A common smoothing mechanism is to take a moving average. While it is effective in dealing with the former by greatly reducing the deviation, it does not deal with the latter very well even with a large window of 1 second, which can cause serious problems in our protocol, since each such computation has a complexity of $O(n^2)$ on each node. Since most RSSI variation is on short time-scales, we take the approach of removing short-lived changes from RSSI readings, so they do not contribute to the moving average. We do this by filtering out all RSSI values that (1) deviates by more than 2dB from the current moving average and (2) is short lived. The deviating

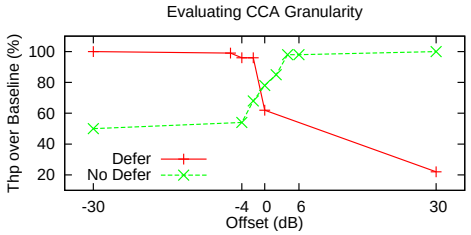


Figure 3: Determine CCA Granularity

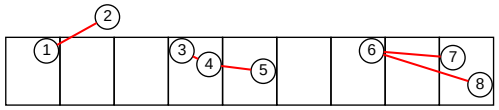
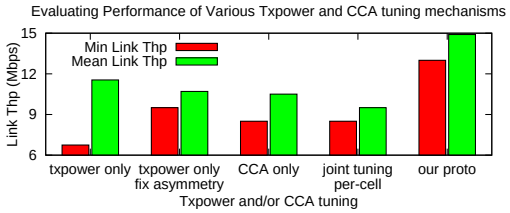
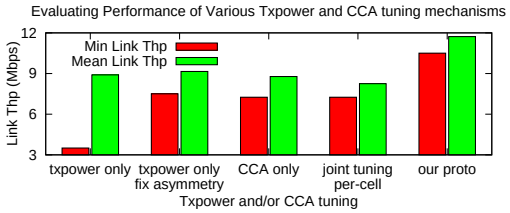


Figure 4: The topology of the 8-node testbed



(a) UDP Performance



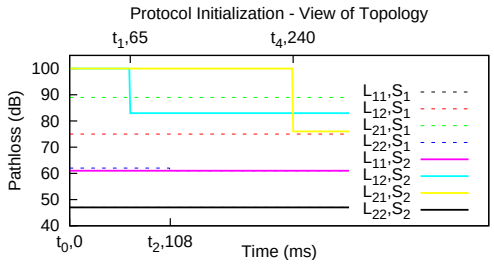
(b) TCP Performance

Figure 5: Performance evaluation on the 8-node testbed

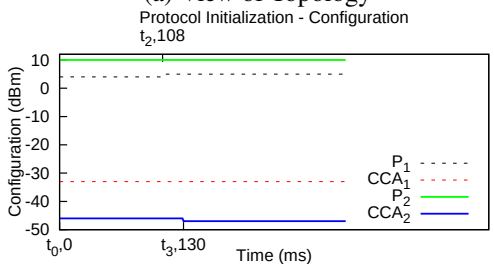
RSSI reading is considered to be short-lived, if within the averaging window, there exists a time interval K, where all RSSI values in the interval fall within 2dB of the moving average. We set the value of K to be 10% of the averaging window. Otherwise, the change is considered long-lived and it is incorporated into the average. We use a 500ms window size in our protocol as it removes most spurious readings in our experiments. In our evaluation, the filtering algorithm reduces the computation cost by more than 10x, compared to moving average.

6 Experimental Evaluation

In this section, we evaluate our protocol on a 8-node indoor testbed (an office scenario), as shown in Figure 4.



(a) View of Topology



(b) Configuration

Figure 6: Protocol Initialization

6.1 Network Capacity

First, we evalute the UDP and TCP capacity of various txpower and/or CCA threshold tuning approaches on the 8-node testbed. The power control and/or CCA tuning approaches we tested are the same as the ones presented in Section 2. Note that except for our protocol, we emulate all other approaches, so they do not incurs protocol overheads, such as protocol initialization or synchronization [14]. In this experiment, nodes 2,4,6 act as three senders and the rest 5 nodes act as receivers, and association is shown in the map. In the UDP test, all senders transmit simultaneously 20Mbps of UDP traffic using iperf for 10 seconds, and TCP manages transmit throughput by itself. Wireless data rate is manually set to 36Mbps.

The minimum and mean link throughputs of all 5 flows are shown in Figure 5. The results are consistent with those presented in Section 2: 1) our protocol improves network capacity over other protocols by at least 40% for UDP and 30% for TCP, 2) the minimum link throughput of our protocol is also much higher than that of other protocols, and, in fact, our protocol improves throughput on all links.

6.2 Protocol Behavior

We evaluate protocol behavior using a network of four nodes, S_1, R_1, S_2, R_2, that supports two UDP flows F_{11} and F_{22}.

Table 1: Convergence Time after Sudden Movement

Movement	Without Low CCA (seconds)	With Low CCA (seconds)
R_1 away from S_1	15	**0.7**
R_1 to S_1	0.3	**0.2**

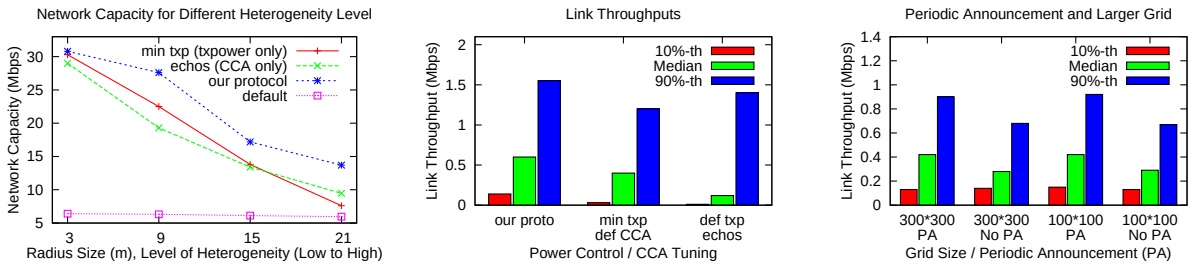
Initialization Figure 6 show the topology view on the two senders and the configuration they choose during the protocol initialization process, with the following events occuring: 1) Before t_0: AP_1 has the full topology, and decides to set the txpower to 4. AP_2 has no idea of L_{12} and L_{21} yet, thus, default pathloss of 100dB are used for them, and AP_2 thinks the two transmission can happen at the same time without any txpower change. 2) At t_1, AP_2 hears C_1, but it does not change txpower because it should use high power even with current view; 3) At t_2: AP_1 hears AP_2 and adjust its view of L_{22}, and also increases it power level, 4) At t_3, AP_2 hears AP_1 and increases the pathloss between them, which in turn decrease CCA threshold, 5) At t_4: AP_2 hears C_2 with the piggyback of pathloss C_2 to AP_1. Note that the view of the topology is quite consistent on both senders and the configurations of our algorithm is very stable, without sacrificing performance.

Mobility & Convergence We evaluate how quickly our protocol converges after a sudden node movement using two flows. We use bidirectional UDP traffic in our experiments. Table 1 shows the convergence time after moving one receiver close to or away from its sender; note that the other sender is actively transmitting. Convergence time is calculated as the time from when the earliest observed change until the protocol converged, which excludes the 500ms delay from the smoothing algorithm.

The results show that periodically lowering the CCA threshold, as described in Section 4, is necessary to deal with such changes. Without this optimization, the convergence time for moving the receiver away from the sender is very long because it requires a change from high CCA to low CCA and such change is difficult when the flows are saturating the network. Convergence time is much shorter when the receiver is moving towards the sender because a change from low CCA to high CCA is easier.

6.3 OPNET Simulation

We now use the OPNET simulator to study several properties of our protocol: network capacity, individual link throughputs (fairness), convergence, and performance in large grids. We use two node placement models. 1) Clus-



(a) Network Capacity (b) Link Throughputs (c) Periodic Announcement

Figure 7: Simulation Results

tered placement: APs (sources) are uniformly distributed, and the clients (recivers) are uniformly distributed in a disk of a chosen radius center at a randomly chosen AP, and 2) Random placement: APs and clients are uniformly distributed, and clients associate with the closest AP. Since the results for the two placement models are very similar, we only present the results for the clustered model. In our simulations, all APs operate on the same channel and use the same data rate. We simulate 10 APs and 10 clients within a $100m \times 100m$ grid, placed according to the clustered model. We vary the cluster radii from 3m to 21m. The results for each radii setting are the average of three different random topologies. The start time for traffic flows is uniformly distributed in the first 20 seconds and we run each experiment for 20 minutes. Traffic is generated using an exponential on-off process with a traffic demand for each node of about 2Mbps. We simulate our protocol, txpower only approach (using minimum power to reach receiver), and CCA tuning only approach (ECHOS [27]).

Network Capacity Figure 7(a) presents the simulation results for network capacity (total throughput across all nodes). Our protocol improves capacity by 22% to 87% over CCA tuning only approach, and 2% to 67% over power control only approach, and the improvement increases as the average radius size increases, i.e. more diversity in client-AP distance. Note that both txpower only and CCA only approaches work well when the radius is short, i.e. less spatial heterogeneity, but the performance degrades when the radius is long. We also changed the number of APs, while keeping the ratio of AP-AP distance to client-AP distance to about 4.5. The results from this setup are consistent with the above. This setup did show that the improvement of iterative approach increases with the number of APs.

Individual Link Throughputs Figure 7(b) shows the 10%-th, 50%-th (median), and 90%-th percentile of link throughput for 15m radius. The link throughputs for our protocol are consistently higher than for the others, indicating that the higher capacity of our protocol comes from spatial reuse and does not hurt fairness.

Table 2: Protocol Convergence Time (sec)

# AP	WO/ Low CCA	W/ Low CCA	Low Traffic
10	39	25	12
20	138	121	37

Convergence Table 2 shows the convergence times for our protocol with and without periodic low CCA, as described in Section 4, for different numbers of APs. The time is measured from the start of the experiment until the configuration is stable. Remember that flows are started uniformly during the first 20 seconds, so convergence may be slow during that period. It shows that the convergence time of the protocol can be reduced by periodically lowering the CCA threshold. However, the convergence time can still be more than two minutes. This is because the duration of low CCA is only 20ms in our protocol, and when the traffic demand is high, collisions can prevent nodes from successfully overhearing enough traffic to collect topology information. At lower traffic rates, the convergence time is significantly lower (last column in Table 2).

Periodic Announcement & Large Grid Figure 7(c) shows the link throughputs with and without periodic announcement, a mechanism that facilitates information collection as presented in Section 4. The simulation uses our protocol. We see that periodic announcements improve the median link throughputs by about 47%. We also ran the same experiment for a $300m \times 300m$ grid. In that case, not all nodes can hear each other so their local topology graph is not complete. Figure 7(c) shows that this only slightly reduces link throughput, showing that our protocol can also work in larger areas.

7 Conclusion

This paper presents a practical power control and CCA tuning protocol that maximizes spatial reuse in chaotic, dense wireless deployments. In our protocol, each wireless node collects information about nearby transmissions by monitoring traffic, and determine the power level and CCA threshold. This process iteratively increases the number of concurrent transmissions that can take place. We implemented the protocol in Linux and addressed several practical challenges. The experimental results from an 8-node platform and the OPNET simulator shows that our protocol works well in practice.

There are several limitations of our protocol which we left as future work: 1) data rate adaptation: A possible extension is to separate the txpower and data rate control by choosing the data rate first based on its L_{11}, which is adapted when L_{11} changes, or when there is external interference from non-compliant senders. 2) exceptional scenarios including highly mobile nodes, non-compliant nodes, and link asymmetry. We believe that an exception-handling approach, which detects such situations and falls back to more conservative behaviors may satisfactorily address many such scenarios. For example, our protocol targets nomadic mobility, where node occasionally move between stationary locations and configurations are recomputed occasionally. Frequent reconfiguration can be used to identify highly mobile nodes, and an obvious fallback configuration would be to use the default power/CCA configuration.

References

[1] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self management in chaotic wireless deployments. In *MobiCom*, 2005.

[2] Y.-J. Choi and K. Shin. Power-adjusted random access to a wireless channel. *INFOCOM*, 2008.

[3] S. M. Das, D. Koutsonikolas, Y. C. Hu, and D. Peroulis. Characterizing multi-way interference in wireless mesh networks. In *WiNTECH*, 2006.

[4] M. A. Ergin, K. Ramachandran, and M. Gruteser. Understanding the effect of access point density on wireless lan performance. In *MobiCom*, 2007.

[5] J. A. Fuemmeler, N. H. Vaidya, and V. V. Veeravalli. Selecting transmit powers and carrier sense thresholds in csma protocols for wireless ad hoc networks. In *WICON*, 2006.

[6] S. Ganu, H. Kremo, R. Howard, and I. Seskar. Addressing repeatability in wireless experiments using orbit testbed. In *TRIDENTCOM*, 2005.

[7] K. Jamieson, B. Hull, A. K. Miu, and H. Balakrishnan. Understanding the real-world performance of carrier sense. In *EWIND*, 2005.

[8] L. Jia, X. Liu, G. Noubir, and R. Rajaraman. Transmission power control for ad hoc wireless networks: throughput, energy and fairness. *WCNC*, 2005.

[9] G. Judd and P. Steenkiste. Using emulation to understand and improve wireless networks and applications. In *NSDI*, 2005.

[10] E. Jung and N. Vaidya. A power control mac protocol for ad-hoc networks. *MOBICOM*, 2002.

[11] T.-S. Kim, J. C. Hou, and H. Lim. Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks. In *MobiCom*, 2006.

[12] J. Lee, W. Kim, S.-J. Lee, D. Jo, J. Ryu, T. Kwon, and Y. Choi. An experimental study on the capture effect in 802.11a networks. In *WinTECH*, 2007.

[13] R. Maheshwari, S. Jain, and S. R. Das. A measurement study of interference modeling and scheduling in low-power wireless networks. In *SenSys*, 2008.

[14] V. Mhatre, K. Papagiannaki, and F. Baccelli. Interference mitigation through power control in high density 802.11 wlans. In *Infocom*, 2007.

[15] J. Monks, V. Bharghavan, and W.-M. Hwu. A power controlled multiple access protocol for wireless packet networks. *INFOCOM*, 2001.

[16] A. Muqattash and M. Krunz. Power controlled dual channel (pcdc) medium access protocol for wireless ad hoc networks. *INFOCOM*, 2003.

[17] A. Muqattash and M. Krunz. A single-channel solution for transmission power control in wireless ad hoc networks. In *MobiHoc*, 2004.

[18] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar. Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the compow protocol. In *in European Wireless Conference*, 2002.

[19] V. Navda, R. Kokku, S. Ganguly, and S. Das. Slotted symmetric power control in managed wlans. *Technical report, NEC Laboratories America.*

[20] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *IMC*, 2005.

[21] D. Qiao, S. Choi, A. Jain, and K. G. Shin. Miser: an optimal low-energy transmission strategy for ieee 802.11a/h. In *MobiCom*, 2003.

[22] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser. Symphony: synchronous two-phase rate and power control in 802.11 wlans. In *MobiSys*, 2008.

[23] V. Shah and S. Krishnamurthy. Handling asymmetry in power heterogeneous ad hoc networks: A cross layer approach. In *ICDCS*, 2005.

[24] A. Sheth and R. Han. Adaptive power control and selective radio activation for low-power infrastructure-mode 802.11 lans. In *ICDCSW*, 2003.

[25] A. Sheth and R. Han. Shush: reactive transmit power control for wireless mac protocols. *WICON*, July 2005.

[26] V. Shrivastava, D. Agrawal, A. Mishra, S. Banerjee, and T. Nadeem. Understanding the limitations of transmit power control for indoor wlans. In *IMC*, 2007.

[27] A. Vasan, R. Ramjee, and T. Y. C. Woo. Echos - enhanced capacity 802.11 hotspots. In *Infocom*, 2005.

[28] M. Vutukuru, K. Jamieson, and H. Balakrishnan. Harnessing exposed terminals in wireless networks. In *NSDI*, 2008.

[29] R. Zheng and R. Kravets. On-demand power management for ad hoc networks. *INFOCOM*, 2003.

[30] J. Zhu, B. Metzler, X. Guo, and Y. Liu. Adaptive csma for scalable network capacity in high-density wlan: a hardware prototyping approach. In *Infocom*, 2006.

[31] Y. Zhu, Q. Zhang, Z. Niu, and J. Zhu. On optimal physical carrier sensing: Theoretical analysis and protocol design. *INFOCOM*, 2007.