Uncovering a Hidden Wireless Menace: Interference from 802.11x MAC Acknowledgment Frames

Wei Wang, Qiang Wang, Wai Kay Leong, Ben Leong, and Yi Li
Department of Computer Science, National University of Singapore

***Abstract*—Previous work on 802.11x (Wi-Fi) power control have focused almost exclusively on the mitigation of interference from data frames. Since Wi-Fi wireless access points are increasingly ubiquitous, it is now common for an 802.11x client to be able to receive signals from more than 10 APs simultaneously in a modern urban operating environment. At such high densities, we found that the interference from the MAC acknowledgment (ACK) frames can potentially reduce throughput by several fold. We show that by dynamically adjusting the transmission power of the ACK frames, a simple ACK power control algorithm, which we call `MinPACK`, can significantly mitigate interference and increase throughput by a median of 31%, while simultaneously improving fairness. `MinPACK` is complementary to existing data frames power control algorithms, and adapts rapidly to dynamic environments.**

I. Introduction

802.11x (Wi-Fi) networks are increasingly ubiquitous, and recent report suggests that the global Wi-Fi hotspot market will continue to grow at an annual rate of 84% in the next few years [2]. Wi-Fi traffic is expected to eventually make up 56% of total global Internet traffic by 2017 [1], and the increased Wi-Fi traffic will likely lead to even denser deployments of Wi-Fi hotspots and more co-channel interference, given the limited number of orthogonal Wi-Fi channels.

Existing work on Wi-Fi performance optimization have focused almost exclusively on minimizing the interference from *MAC data frames* by regulating the transmission power of access points (APs) [5, 15, 16] and by scheduling the transmission time carefully [14, 17]. In a recent measurement study of corporate Wi-Fi AP deployments, we found that 802.11x MAC Acknowledgment frames can sometimes cause significant interference, and as illustrated in Fig. 1, can effectively extend the interference range of an AP.

Because the main bulk of the access network traffic is downstream from the APs [7, 13], clients can generate a large number of ACK frames. Our analysis of the traces obtained from real AP density measurements suggests that the likelihood that a client will experience interference from MAC ACK frames can range from 25% to 50% in practical Wi-Fi deployments. We found that for a campus 802.11n AP deployment, ACK interference could cause a reduction of several fold when two interfering 802.11n flows compete with each other. Furthermore, the ACK interference from an 802.11a/b/g flow can cause starvation for other 802.11n flows.

MAC ACK frames are fundamentally different from MAC data frames, because they are generated automatically and do not provide any feedback or consider the channel state when transmitted. Thus, it is not straightforward to directly apply available power control algorithms on MAC ACK frames. In addition, due to its small frame size and low data rate, we found that we can reduce the power of ACK frame much more than that for data frame without adverse effect. To address the MAC ACK interference problem, we propose a simple ACK power control algorithms for clients, called `MinPACK`, and show through extensive experiments that `MinPACK` is able to achieve a median throughput improvement of 31% for 802.11a/b/g clients, and can potentially double the throughput for 802.11n clients.

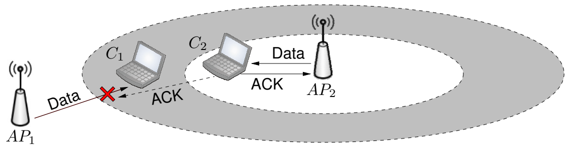


Fig. 1. Interference between adjacent Wi-Fi hotspots. MAC ACK frames of clients effectively extend the interference range of a hotspot (i.e., clients in the gray ring like C_1 will not experience interference from AP_2's data frames but can experience interference from the MAC ACK frames of AP_2's clients like C_2).

This rest of the paper is organized as follows. In Section II, we present a measurement study to motivate this work and demonstrate the severity of the ACK interference problem. In Section III, we describe the `MinPACK` power control algorithm and in Section IV, we present a comprehensive evaluation of the algorithm in practical Wi-Fi testbeds. Finally, we provide a summary of existing work on interference mitigation in Section V and conclude in Section VI.

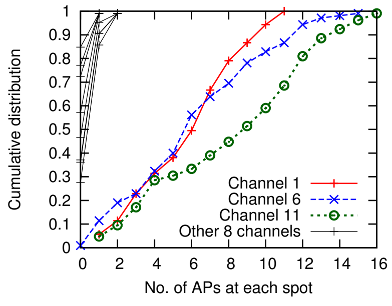
II. Motivation

In this section, we first present a measurement study on the density of APs in densely populated metropolitan areas. Next, we present a network model to estimate the likelihood of ACK interference in modern Wi-Fi AP deployments. Finally, we show with experiments how ACK interference can cause significant performance degradation and that reducing the transmission power for MAC ACK frames can mitigate this problem. For the rest of this paper, we will use "data sender" and "ACK receiver" interchangeably, and also "ACK sender" and "data receiver" interchangeably.

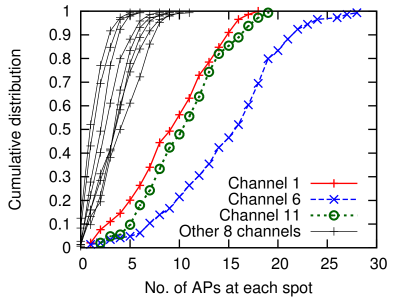
A. Measurement Study on AP Density

Prior work on AP density measurements [5] relied on the estimated physical location of APs (obtained from wardriving) and assumed a fixed interference range (e.g., 50 m) to estimate AP density. This method is not accurate as the estimated AP locations from wardriving are shown to have a median error of 32 m [11]. To achieve a higher accuracy for AP density

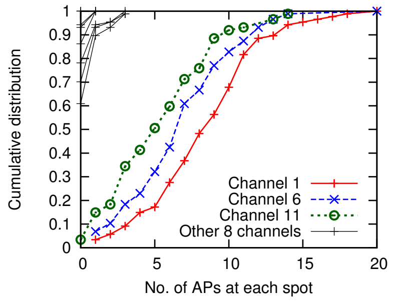
978-1-4799-4657-0/14/$31.00 © 2014 IEEE



(a) Commercial



(b) Residential



(c) Campus

Fig. 2. Number of APs observed during warwalking.

measurements, we conducted an equivalent experiment, but on foot and we call this "warwalking". We surveyed three classes of densely populated metropolitan areas: (i) a high-density residential area, (ii) a shopping mall and (iii) a university campus. Most APs in the residential area are unmanaged private hotspots while the APs on the university campus are dominated by an enterprise Wi-Fi network. The shopping mall has mix of both private hotspots and an enterprise deployment.

Throughout the warwalking exercise, we set a sniffer to repeatedly scan all available Wi-Fi channels (1 to 11), with a scan duration of 1 s for each channel. In each area, the path was about 1 km long and the walking speed was approximately 1 m/s. Since our walking speed is relatively slow compared to the range of Wi-Fi, we considered each 1 s spent in a channel to be a single "spot". We estimated the number of APs observed at each spot by counting the number of unique MAC addresses (or BSSIDs) that broadcast a beacon frame.

Precautions were taken to minimize potential measurement errors. First, as the sniffer might be able to receive beacon frames transmitted in adjacent channels, we examined the channel number embedded in the beacon frames to determine the channel an AP was on. Second, as some manufacturers support virtual APs, we consider a batch of MAC addresses to belong to the same physical AP if they do not differ more than k from each other, where is k is the largest number of consecutive MAC addresses observed in the trace. We found k to be 5 in our traces.

Fig. 2 shows the cumulative distribution of the AP count for the three scenarios. As expected, the three orthogonal channels (1, 6 and 11) had the highest frequency in all the scenarios. The residential area had the densest deployment, with a median AP count of 15 at channel 6. In other words, an AP at channel 6 in residential area is expected to have 14 neighboring APs. Although the APs in campus area are mainly from an enterprise network, we found that the median AP count for channel 1 was 8, i.e., the AP density was still surprisingly high.

B. Modeling MAC ACK Interference

In this section, we model and estimate the likelihood that a data frame to a client device will experience ACK interference from other clients connected to neighboring APs. Because whether ACK interference occurs will depend on the positions of APs and clients, we first develop a simple model for the area

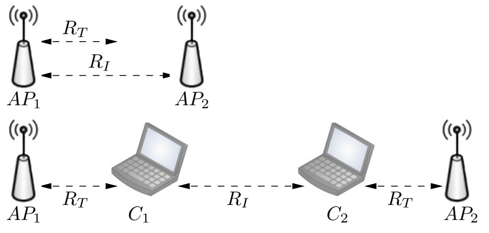


Fig. 3. Illustration of the minimum and maximum distance between two adjacent APs for ACK interference to occur.

from which the clients of neighboring cells can cause ACK interference (we call this the "interference region"). Next, we estimate the average number of clients that lie within the interference region and the probability of ACK interference in terms of the rate that clients receive data frames.

Modeling the Interference Region. For ACK interference to occur, adjacent APs cannot be too close, or too far from each other. Consider a simple case of two adjacent APs, AP_1 and AP_2 with associated clients C_1 and C_2 respectively. We assume that the APs and clients have the same transmission power and the channel is symmetric and they have the same transmission range R_T. We also assume they have the same carrier sense and interference range R_I, s.t. $R_I \geq R_T$. Fig. 3 illustrates the minimum and maximum distance between the two APs for ACK interference to occur. If the two APs are within R_I of each other, CSMA will prevent simultaneous transmission and hence there will be no ACK interference. On the other hand, if two APs are beyond $R_I + 2R_T$ from each other, ACK interference will not occur either. Hence, for ACK interference to occur, the two APs must be within a distance between R_I and $R_I + 2R_T$, and the clients are within the transmission range R_T of their associated APs.

We set up a coordinate system centered at AP_1 with AP_2 at $(u, 0)$ and C_1 at (x, y), where $R_I \leq u \leq R_I + 2R_T$, as shown in Fig. 4. The two small circles indicate the transmission range of the APs. The large circle indicates the area within which another client's transmissions will interfere with C_1. Thus, the shaded area represents the interference region from which a client C_2 that is associated with AP_2 can cause ACK interference to C_1.

Average number of clients in the interference region. If we assume that the location of the clients for an AP is

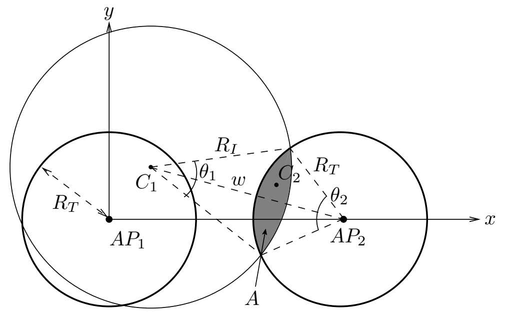


Fig. 4. Network model used in the analysis.

uniformly distributed, then $p(u, x, y)$ can be modeled as

$$p(u, x, y) = \frac{A}{\pi {R_T}^2} \tag{1}$$

where $p(u, x, y)$ is the probability that C_2 lies in the interference region, for a given u, x and y and where A is the area of the shaded intersection. Using geometric rules, we can express A as

$$A = \frac{{R_I}^2}{2}(\theta_1 - \sin\theta_1) + \frac{{R_T}^2}{2}(\theta_2 - \sin\theta_2) \tag{2}$$

where θ_1 and θ_2 are the internal angles formed by the sectors at C_1 and AP_2. By setting w to be the distance between C_1 and AP_2, we can express A in terms of u, x and y by computing θ_1 and θ_2 using the Cosine rule for a triangle with sides R_T, R_I, and w, where $w = \sqrt{(u-x)^2 + y^2}$.

Suppose AP_2 has m clients, uniformly distributed within its transmission range, we can model the number of clients in A as a binomial distribution $B(m, p(u, x, y))$. Thus, the average number of clients in A will be $m \times p(u, x, y)$. The average number of clients in the interference region $\bar{m}$, over all values of u, x and y can then be expressed as:

$$\bar{m} = \iint_S \int_{u=R_I}^{R_I+2R_T} [m \times p(u, x, y)]\, du\, dx\, dy \tag{3}$$

where S is the area of the small circle centered at AP_1.

Likelihood of ACK interference. Suppose in Fig. 4, it takes C_1 time t to receive a data frame from AP_1. If within this time t, a client C_2 in area A finishes receiving a data frame from AP_2, it will respond with an ACK which will interfere with C_1. If we assume that the rate at which C_2 receives data frames follows a Poisson distribution with rate λ, the rate of ACK transmissions from C_2 will also follow the same Poisson distribution. Let z denote the duration from the time C_1 starts receiving a data frame to the moment C_2 transmits the next ACK. z will follow an exponential distribution with rate λ. Since C_2's ACK will collide C_1's data frame if z is smaller than t, the probability of ACK interference due to C_2 (or a client in the interference region) can be expressed as:

$$P(z < t) = 1 - e^{-\lambda t} \tag{4}$$

Suppose there are on average n APs within the range of R_I and $R_I + 2R_T$ from AP_1, and each AP has m clients. Then, the probability that ACK interference is experienced at C_1 is:

$$P_{n,m} = 1 - e^{-n\bar{m}\lambda t} \tag{5}$$

Because the transmissions of data frames take time, λ is limited by $\lambda < 1/t$, where t is the transmission time of a data frame. Assuming that the data frames sent by all APs are evenly distributed among their clients, λ is further bound by $m\lambda < 1/t$. Because of carrier sense between APs, λ is also constrained by the number of APs within carrier sense range, as denoted by N_I. To find N_I, we first compute the density of AP. Since there are n APs within the range of R_I and $R_I + 2R_T$ from our AP, the AP density d can be expressed as:

$$d = \frac{n}{\pi(R_I + 2R_T)^2 - \pi {R_I}^2} = \frac{n}{4\pi R_T(R_I + R_T)} \tag{6}$$

Thus, the average number of APs within interference range of any AP is simply:

$$n_I = d \times \pi {R_I}^2 = \frac{n{R_I}^2}{4R_T(R_I + R_T)} \tag{7}$$

Hence, the final upper bound on λ is $m\lambda(n_I + 1) \leq 1/t$.

Assuming data frames are 1,500 bytes in size and sent at a typical data rate of 54 Mbps, the transmission time works out to be $t = 0.0002$ s. Under the `ns-2` model, the Wi-Fi sender implements carrier sense by detecting a high energy transmission on the channel. Hence, the interference range is typically set to twice the transmission range, i.e., $R_I = 2R_T$. However, we observed that our Atheros Wi-Fi adapter considers the channel to be busy only when it detects a Wi-Fi preamble. Thus, in practice, we find that the carrier sense range is equal to the transmission range, i.e., $R_I = R_T$. For completeness, we show both cases in Fig. 5, where we plot the probability of ACK interference against varying values of $m\lambda$ and n. We believe that most practical networks would lie somewhere between these two scenarios.

In our warwalking experiment, the median number of APs observed ranges from 5 to 15. Since we could only observe APs within our transmission range (R_T), we cannot directly observe n, the number of APs between R_I and $R_I + 2R_T$. As such, we assume that the APs are uniformly distributed and estimate the density of APs within R_T to derive the value of n. For $R_I = 2R_T$, the area enclosing the APs is 12 times that of R_T. Thus, n is estimated to be in the range from 60 to 180. For the other case where $R_I = R_T$, the area enclosing the APs is 8 times that of R_T, and thus n is estimated to be in the range from 40 to 120.

From our results in Fig. 5, we observed that there is an upper bound on the probability that seems to be largely independent of λ and n, i.e., while the probability of ACK interference generally increases with the number of APs and also the sending rate, there is a cap on the probability of ACK interference. This is due to the bound we imposed on the sending rate to model the effects of CSMA. The cap on the ACK interference probability is dependent on the interference range. When the interference and carrier sense range is large, CSMA will prevent the APs from sending at a higher rate. As such, the average probability of ACK interference is bound at about 0.25 and 0.5 for the scenarios $R_I = 2R_T$ and for $R_I = R_T$ respectively.

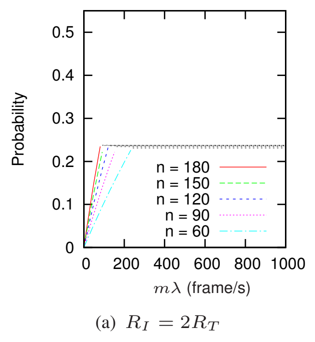
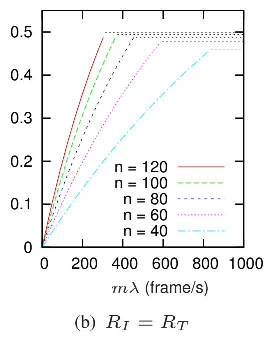


Fig. 5. Computed probability of ACK interference in our model.

We note that the maximum sending rate for the 802.11x data rate of 54 Mbps is about 4,700 frame/s. Fig. 5 suggests that it is therefore quite plausible for a single busy AP to create sufficient traffic to reach the upper bound of $m\lambda$. Thus, we can expect an ACK interference probability of between 25% and 50% in 802.11x networks.

C. Impact of MAC ACK Interference

In order to understand the impact of MAC ACK interference in practical Wi-Fi networks, we conducted a set of experiments for the scenario shown in Fig. 1 using our campus WLAN network, which comprises of Cisco 1140 series APs. We have two clients: Advantech system board with Compex WLE350NX 3x3 802.11n adapter, and ALIX system board with Compex WLM54AG 802.11a/b/g adapter. The traffic sources were Linux machines in our lab. The channels of the APs are dynamically assigned by the backend Cisco wireless LAN controller, and the two APs' channels were fixed at channel 149 for our experiments. RTS/CTS is not enforced by the APs.

For each set of experiments, we compared the throughput of both clients at the default ACK power of 20 dBm against that when one or both reduce their ACK power to 10 dBm. We also placed sniffers next to each client to count the number of ACK frames transmitted. Fig. 6 shows the results with both clients using 802.11n and Fig. 7 shows the results with one client using 802.11a and the other using 802.11n.

ACK power control mitigates ACK interference. As expected, reducing the ACK power of one client improves the throughput of the other client. For example, in Fig. 6(a), C_1's throughput increased from 16 Mbps to 30 Mbps when C_2 reduced its ACK power. When both clients reduced their ACK power, the overall throughput improved by 134%, compared to that when both were using the default power. We also observed similar improvements with both clients using 802.11a, but we omit the results here due to space constraints.

ACK power control can improve local throughput. One counter-intuitive observation in our experiments is that a client can sometimes achieve a higher throughput by reducing its own ACK power. This can be seen in Fig. 6(a) where C_1's throughput was increased when C_2 reduced its ACK power. Surprisingly, we found that C_1 could then achieve a further increase in throughput from 30 Mbps to 35 Mbps by reducing its own ACK power. The 802.11n standard allows a data sender

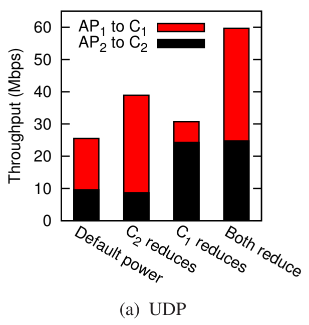
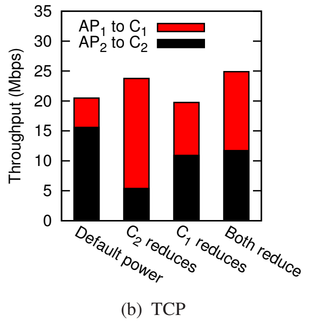


Fig. 6. Impact of ACK interference with two 802.11n links.

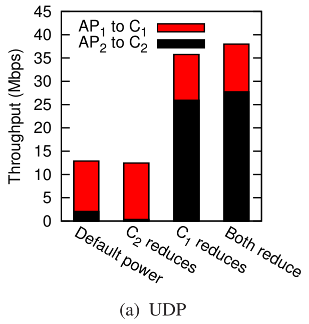
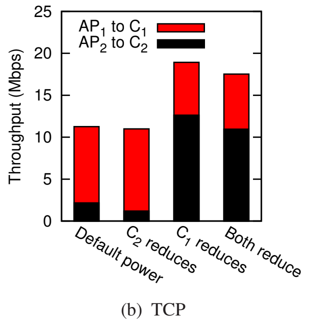


Fig. 7. Impact of ACK interference with 11a link (AP_1 to C_1) and 11n link (AP_2 to C_2).

to transmit multiple frames in a single Aggregate MPDU (A-MPDU) and to also selectively retransmit any lost frames in any order, as long as their sequence numbers lie within a transmission window of 64 frames [3]. We suspect that the reason for this is that when only C_2 reduced its ACK power, C_1's ACK interference on C_2 was so significant that AP_2 had to use a low data rate that does not support A-MPDU. Thus, C_2 would reply with one ACK frame for every data frame it received from AP_2. However, once C_1 reduced its ACK power, C_2 experienced less ACK interference and AP_2 can then switch to a higher data rate that supports A-MPDU. With A-MPDU enabled, C_2 can send one Block ACK in response to several data frames, instead of sending one ACK for each frame. We found that after C_1 reduced its ACK power, C_2 sent 58% fewer ACK frames compared to when C_1 was using the default ACK power.

ACK power control improves TCP fairness. Fig. 6(b) shows C_1 having a disproportionately small throughput as compared to C_2 when both clients were using the default ACK power. When both clients reduced their ACK power, they both achieved similar throughput and the overall throughput improved by almost 25%. (Note that we only reduced the power of MAC ACK frames but not the power of TCP ACK packets which are encapsulated as data frames.) One reason is because TCP traffic is very sensitive to packet loss, and the flow that incurred more interference (i.e., C_1) could not increase its congestion window. In other words, when two TCP flows interfere with each other, fairness is often compromised.

ACK power control prevents 802.11n from starvation. In our experiment where C_1 was using 802.11a and C_2 802.11n, C_2 was nearly starved (see Fig. 7(a)). We suspect it is because 802.11n is more vulnerable to interference than 802.11a, just like how 802.11g could be less robust against interference than 802.11b [9]. When C_1's ACK power was reduced, C_2's throughput greatly improved regardless of C_2's ACK power level. Similar results can be observed for TCP traffic, as shown in Fig. 7(b). In other words, by reducing ACK power, we are able to prevent an 802.11n client from being overwhelmed by an 802.11a client.

III. 802.11x MAC ACK Power Control

While reducing ACK power is helpful, ACK frames can be lost if we overdo it and the data sender will then wrongly interpret the loss as a loss of the data frame. This will cause the data frame to be retransmitted and in the worst case, possibly also cause the data rate to be reduced, resulting in a significant degradation in the throughput. The challenge therefore, is to reduce the ACK power without causing unnecessary ACK frame losses.

In this section, we describe the `MinPACK` algorithm. `MinPACK` is an acronym for *Minimum Power for ACK frames*. The basic idea of `MinPACK` is to gradually reduce the transmission power for the ACK frames until the point just before ACK frame losses will occur. This can be achieved if the ACK sender can accurately estimate the delivery success rate of its ACK frames.

We can try to estimate the success rate through one of two ways. A straightforward approach is to have cooperative feedback from the data sender (i.e., the ACK receiver) and a more indirect approach is to estimate the rate through passive observations. While the latter method is less accurate, we found that it is able to achieve estimation results that are comparable to the former *without requiring any modification to the 802.11 standard*. This thereby ensures compatibility with existing commercial 802.11 adapters and allows for immediate deployment in existing Wi-Fi networks.

A. Cooperative Feedback from ACK receiver

To accurately estimate the instantaneous ACK success rate (denoted by Φ), the ACK sender has to determine the number of ACK frames transmitted (denoted by n) and the number of ACK frames successfully delivered (denoted by k) in the past small time window of τ. Because ACK frames are transmitted automatically by the adapter hardware, we cannot directly count n in our implementation. Instead, since the hardware generates an ACK for every data frame successfully received, we can obtain n by counting the number of such data frames. On the other hand, the MadWiFi driver reports the status of every data frame transmitted. Thus, the ACK receiver can directly obtain k from the number of successful transmissions and send this information to the ACK sender using a small control message.

One slight complication is that both parties have to synchronize the time window τ, in which n and k are to be counted. Medium contention or the loss of control messages could cause τ on both parties to be out-of-sync. To address this problem, the range of frame sequence numbers is specified in the control messages. The ACK receiver simply counts and returns k within a specified range of frame sequence numbers.

B. Passive Estimation without Feedback

There are two drawbacks for the cooperative feedback approach. First, the control messages would incur an additional overhead on the network, and this overhead is especially high when the channel condition fluctuates rapidly, causing the frequency of control messages to increase. Second, the APs will have to be modified in order to provide clients with the feedback. If we adopt a passive approach in the estimation of Φ, clients will be able to perform ACK power control without requiring any modifications to the APs. This would make incremental deployment easier.

A data sender will repeatedly retransmit the same data frame until it receives the associated ACK frame, or until the retransmission limit is reached. Hence, by counting the number of duplicate data frames received, the ACK sender can in principle infer the number of unsuccessful ACK frames (denoted by m). Φ can thus be estimated as $\frac{n-m}{n}$.

The limitation of passive inference is that while the ACK sender is able to accurately count n, there is a risk of under-estimating m, because of the retransmission limit. When this limit is reached, a data sender will proceed to transmit the next data frame. Thus, the ACK sender may wrongly infer that the ACK frame was successfully delivered when the limit was reached.

To address this problem, the data sender can encode a bit in each data frame to indicate whether it had received the ACK for the previous data frame. This will however require the data sender to be modified like the cooperative feedback approach and also hardware modifications which we are currently unable to implement in the available adapters. Nevertheless, we show in Section IV that even without this optimization, we are able to achieve performance that is comparable to the cooperative feedback approach in practice.

C. Extension to Block ACK

The 802.11n standard allows a data sender to transmit multiple frames in a single A-MPDU, which means that instead of sending an acknowledgment for every frame, one ACK frame, called a Block ACK, is sent to acknowledge each A-MPDU. While the passive estimation method described in Section III-B is designed for conventional per-frame ACK, it is straightforward to extend it to handle Block ACK for 802.11n.

When an A-MPDU with at least one non-corrupted frame is successfully received, the hardware adapter will automatically reply with a Block ACK. As before, the ACK sender can count n by directly counting the number of such A-MPDUs received. To count m, the ACK sender has to maintain a short history of A-MPDUs received as the data sender might not immediately retransmit the unacknowledged frames. If any frame in a newly received A-MPDU matches one in the history, we can infer that the Block ACK for the corresponding A-MPDU in the history has been lost. This entire A-MPDU is then discarded from the history to prevent double counting as we have already determined the status of its Block ACK. As the

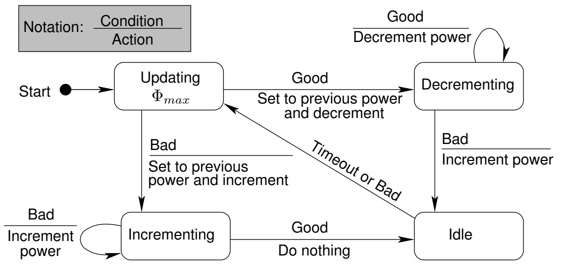


Fig. 8. State diagram of ACK power control algorithm. Condition "Good" refers to $\Phi \geq \Phi_{max} - \delta$, whereas "Bad" for otherwise.

transmission window of the data sender is 64 frames, we only need to maintain a history of A-MPDUs within this range. Our passive estimation method for Block ACK does not require any hardware modification and can be easily implemented using current 802.11n adapters.

D. `MinPACK` *Power Control Algorithm*

Ideally, the transmission power of ACK frames should be set at a level that is just slightly higher than that which will cause unnecessary ACK frame losses. To determine this level, we compare the instantaneous ACK frame delivery rate Φ to the ACK delivery rate when transmitting at maximum power Φ_{max}. Since the quality of 802.11x links varies over time, `MinPACK` needs to adapt to the dynamic environment by periodically adjusting the transmission power level. The key idea for `MinPACK` is very simple: we keep reducing the transmission power until Φ falls below Φ_{max}, and then set the power level at minimum power level required to bring Φ back to Φ_{max}. Periodically, we will probe the network to determine if the network condition has improved. Whenever Φ falls below Φ_{max}, we gradually increase the power level until Φ is close to Φ_{max}. Fig. 8 shows the state diagram of our algorithm, which has 4 states.

The algorithm starts at the maximum transmission power, which is the default value set in the adapter. We estimate the instantaneous ACK delivery rate Φ whenever we receive a new data frame. To keep Φ up to date, we use a sliding window of size τ = 200 ms in our implementation to compute Φ. We consider Φ to be "Good" and move to the next state if it is close to Φ_{max}, i.e., when $\Phi \geq \Phi_{max} - \delta$. δ introduces some hysteresis and we set it to 0.05 in our implementation. When $\Phi < \Phi_{max} - \delta$, we consider the condition to have turned "Bad" and perform the corresponding action and make the state transition as depicted in the state diagram. Because we use a sliding window of size τ to update Φ, we will wait for at least τ time between state transitions to ensure that we have a good estimate before we decide.

Handling Changes in the Link Quality. The link quality of 802.11x networks varies over time, e.g., due to weather condition [19]. To adapt to the change in link quality, we periodically try to update Φ_{max}. When a timeout occurs after `MinPACK` is in the Idle state longer than 10τ, `MinPACK` will attempt to update Φ_{max}. While a simple approach to determine Φ_{max} is to reset the transmission power to the default maximum value to make a measurement, doing so indiscriminately could potentially cause ACK interference to other clients. Thus, we introduce a heuristic to decide whether or not to update Φ_{max}. Every time Φ_{max} is measured, we also record the average RSSI value (R) of the data frames. To decide if we should update Φ_{max}, we compare the current average RSSI value (R_{curr}) with the recorded value (R_{prev}). If the latest RSSI is lower than previous one, we will update Φ_{max}. If the RSSI has improved, we will examine the current value of Φ. If Φ is close to 1, it means that nothing should be done since there is little room for improvement. Otherwise, we should update Φ_{max}. If the RSSI remains unchanged, we simply maintain the current Φ_{max}. In summary,

$$\begin{array}{rl} R_{curr} \leq R_{prev} - 2\,\text{dB} & \rightarrow \text{update } \Phi_{max} \\ R_{curr} \geq R_{prev} + 2\,\text{dB} & \rightarrow \text{consider if } \Phi \not\approx 1 \\ \text{Otherwise,} & \rightarrow \text{do nothing.} \end{array}$$

We choose 2 dB as the threshold because our measurement showed that short-term RSSI reading tends to fluctuate within 2 dB.

IV. Evaluation

A. Experiment Setup

We implemented `MinPACK` for OpenWRT (kernel 2.6.25) and Ubuntu (kernel 3.10.0) because we used two different hardware platforms for 802.11a/b/g and 802.11n respectively.

802.11a/b/g. The OpenWRT version was installed in an 802.11a/b/g urban wireless testbed deployed at a campus dormitory, with 20 ALIX boards (each with a 500 MHz CPU and 256 MB RAM). Their antennas are stuck to the wall outside the buildings [19]. The adapters used are Compex adapters, featuring the Atheros AR5414 chipset. The default transmission power is 23 dBm for both data and ACK frames and the adapters were configured to run in 802.11a mode to minimize the interference from nearby campus wireless networks. The MadWiFi (version 0.9.4) driver was used with default MAC transmission limit of 10 and running in monitor mode. We adopted the popular RRAA method [18] as the default rate adaptation algorithm of data frames.

802.11n. The Ubuntu version of `MinPACK` was deployed on the Advantech board (1.66 GHz CPU and 1 GB RAM), equipped with an 802.11n adapter. Two such boards were used as clients to associate with and monitor the campus APs. The 802.11n adapter is also from Compex (with 20 dBm transmission power), and the `ath9k` driver was used.

We modified both the MadWiFi and `ath9k` drivers to implement fine-grained ACK power control by writing the desired value to a 6-bit ACK power control register in the hardware. Compared to the conventional method of using `iwconfig` to achieve ACK power control, our register-based approach offers the following advantages: (i) finer granularity of control (about 0.5 dBm), (ii) larger range (30 dBm for our 802.11a/b/g adapter), and (iii) option to not affect the transmission power of the data frames.

Both versions were implemented using Click modular router. In our experiments, Iperf was used as the traffic generator. We measured the throughput of `MinPACK` after

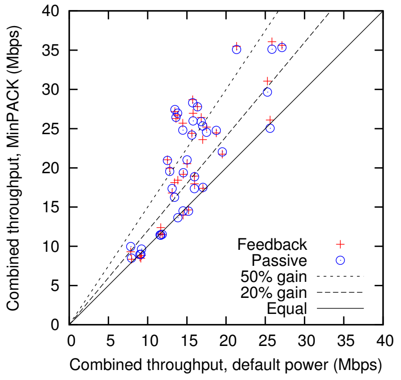


Fig. 9. Throughput gain due to MinPACK for topologies in Fig. 1.

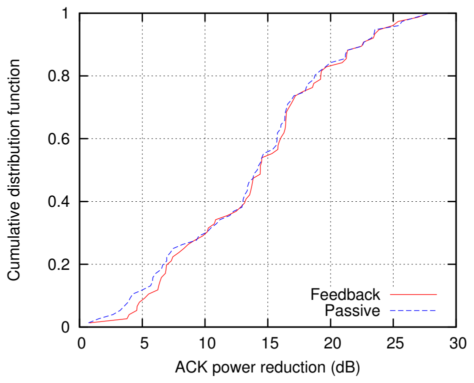


Fig. 10. Results of power reduction of ACK frames.

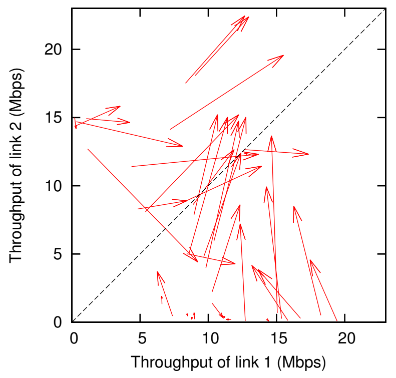


Fig. 11. Improvement in fairness due to MinPACK for topologies in Fig. 1.

the throughput stabilizes. It typically takes at most 15 s for MinPACK to converge from the default maximum power to the final power level. Each experiment lasted for 1 minute.

B. Gain Achieved

To evaluate the effectiveness of MinPACK in mitigating ACK interference for the scenario in Fig. 1, we selected 38 such topologies from our testbed at random and measured the throughput for concurrent UDP flows, for different scenarios. Both the active feedback and passive Φ estimation approaches were evaluated.

Fig. 9 shows the results of the measured throughput for the two contending links. We see that MinPACK is able to improve the total throughput for most cases. With the passive estimation method, 12 of all the 38 cases (32%) saw an increase of more than 50% in total throughput, and the median throughput gain is 31%. The largest improvement was 104%. We also plot the distribution of ACK power reduction achieved by MinPACK in Fig. 10. The median reduction in ACK interference was about 14 dB, and the maximum reduction was 28 dB. The power reduction achieved for the two different estimation methods were similar. For the rest of this paper, we will only present the results for MinPACK with the passive estimation method.

Fig. 11 shows the impact of MinPACK on the fairness in the throughput achieved by the contending link pairs. For each link pair, we draw an arrow from the point of default maximum ACK power to the operating point that MinPACK converges to. Most arrows point towards the diagonal line, indicating an improvement in the fairness between the two flows. Some arrows also point towards the top right corner. This means that MinPACK is able to achieve improvements in both fairness and efficiency for these topologies.

To evaluate the performance of MinPACK with 802.11n adapters for the scenario of Fig. 1, we used two Advantech boards as the clients and associated them with the campus WLAN APs. We first placed the clients in such a way that there was direct line-of-sight (LOS) between them. Three different scenarios were investigated. Fig. 12(a) and Fig. 12(b) show the throughput results of MinPACK for UDP and TCP traffic, respectively. As ACK interference is very strong due to direct LOS, MinPACK is able to significantly improve the overall UDP throughput as well as fairness (e.g., 196% throughput gain for position 2). The TCP throughput gain due to MinPACK is not as significant as that of UDP (e.g., 26% for position 2), but we still observe a great improvement of TCP fairness. We repeated the above experiment by moving the clients to the positions where there is no direct line-of-sight between the two clients. As shown in Fig. 12(c) and Fig. 12(d), the gain due to MinPACK is not big for non-LOS case (e.g., about 10% to 12% for UDP). That is largely because the ACK interference was already low and there is little room for improvement.

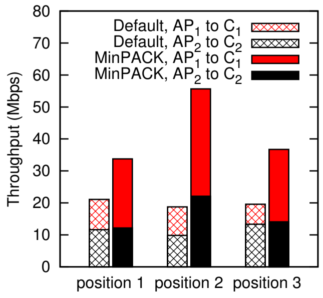
We also repeated the above experiments for the case for an 802.11a and an 802.11n client, as shown in Fig. 13. Both clients used the same default ACK power of 20 dBm. With direct LOS, MinPACK is able to significantly improve the UDP throughput of the 802.11n client, which is less robust against ACK interference than 802.11a client (see Section II-C). The total TCP throughput with MinPACK is also increased although the improvement is not large. When there is no direct LOS between the clients (Fig. 13(c) and Fig. 13(d)), MinPACK can still improve the throughput of the 802.11n client by some extent, due to the vulnerability of 802.11n to interference.

C. Power Control of Data Frames is Not Sufficient

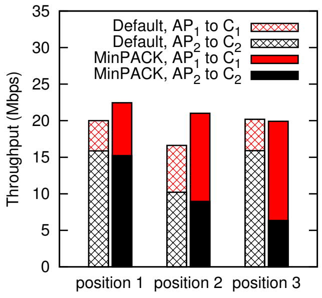
It is not uncommon for a receiver to experience both ACK interference and data interference at the same time, as illustrated in the scenario in Fig. 14(a). When AP_1 and AP_2 are transmitting at the same time, client C_1 is subject to the interference from both AP_2's data frames and client C_2's ACK frames. Conventional power control solutions [5, 15, 16] were designed to mitigate data frame interference by reducing the transmission power, but they did not consider the impact of ACK interference.

To investigate the coupling between MinPACK and the power control of data frames, we selected a sample topology of Fig. 14(a) in our testbed, and run concurrent UDP traffic. During the experiments, we gradually reduced the transmission power of AP_2's data frames, at steps of 2 dBm. This is to emulate the behavior of the power control solutions of data frames. For each power level of node AP_2's data frame, we run the experiment with MinPACK and also with the default maximum power of ACK frames.

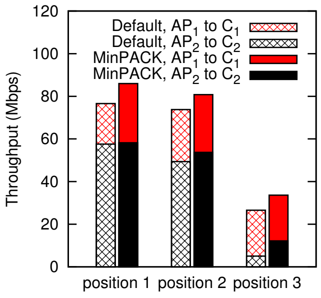
Fig. 14(b) shows the measured throughput, and we make several observations. First, when AP_2's data frames were sent



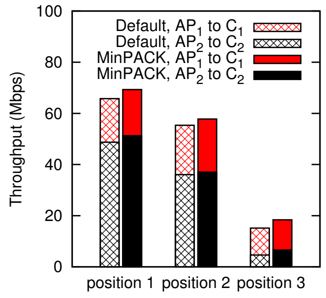
(a) UDP, LOS between C_1 and C_2.



(b) TCP, LOS.

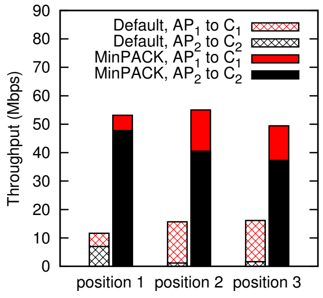


(c) UDP, non-LOS.

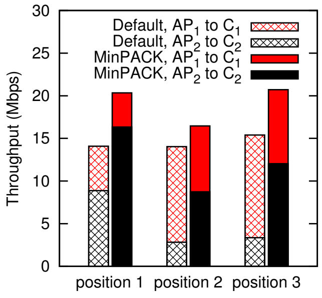


(d) TCP, non-LOS.

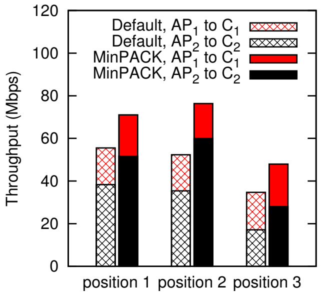
Fig. 12. Achieved throughput for 802.11n vs. 802.11n in campus WLAN.



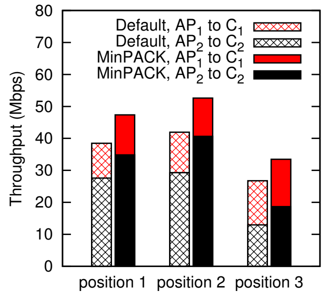
(a) UDP, LOS between C_1 and C_2.



(b) TCP, LOS.



(c) UDP, non-LOS.



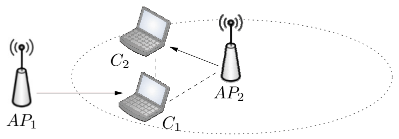
(d) TCP, non-LOS.

Fig. 13. Achieved throughput for 802.11a (C_1) vs. 802.11n (C_2) in campus WLAN.

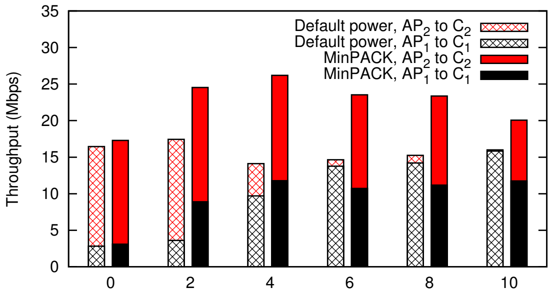
at the default maximum power, their interference on C_1 was so strong that the throughput of C_1 did not improve much even after MinPACK mitigated the ACK interference from C_2. Second, reducing the power of AP_2's data frames alone might not be sufficient to increase C_1's throughput because C_1 was still subject to the ACK interference from C_2. For example, after the power of AP_2's data frames was reduced by 2 dBm, the increase of C_1's throughput was very small when the default ACK power was used. Using MinPACK, the throughput of C_1 was greatly improved. Third, when the default ACK power was used and the power of AP_2's data frames dropped further (e.g., from 4 dBm onwards), we observed a continuous increase of C_1's throughput but a gradual reduction of C_2's throughput. The reason is likely because, as AP_2 reduced its power, C_1 could receive more data frames from AP_1. As a result, more ACK frames were transmitted from C_1, causing more interference to C_2. By mitigating the ACK interference from C_1, MinPACK is able to prevent the throughput reduction of C_2, thereby improving both the combined throughput and fairness performance.

D. Client Mobility

We also investigated the performance of MinPACK when a client is mobile. The experiment setup is similar to the scenario of Fig. 1. Client C_1 was initially located close to AP_1. It was then moved away from AP_1 towards C_2 at a speed of approximately 1 m/s for about 60 s. Fig. 15 shows the change in throughput at default maximum ACK power and with MinPACK, respectively. At maximum power, client C_1 initially had a much higher throughput than C_2. However, as C_1 was moved away from AP_1 (e.g., after 40 s), the throughput of



(a) Scenario of both ACK and data interference.



(b) Power reduction of AP_2's data frames (dBm)

Fig. 14. Effect of power control of data frames.

client C_2 increased and client C_1 was almost starved. The shift of the two clients' throughput is partly due to the continuous signal strength drop from AP_1 to C_1. Initially the signal received at C_1 was strong enough to sustain the interference from C_2's ACK but not later when it became weak. On the other hand, MinPACK achieved concurrent transmission in the first half of the trace. When the signal received at C_1 was weak (e.g., the second half), MinPACK was able to prevent the starvation of C_1 by mitigating the ACK interference from C_2. Note that, at the late stage of the trace for MinPACK, client C_2 incurred some degree of throughput drop. The reason is because the signal between AP_1 and C_1 became weak and there was smaller room for C_1 to reduce its ACK power.

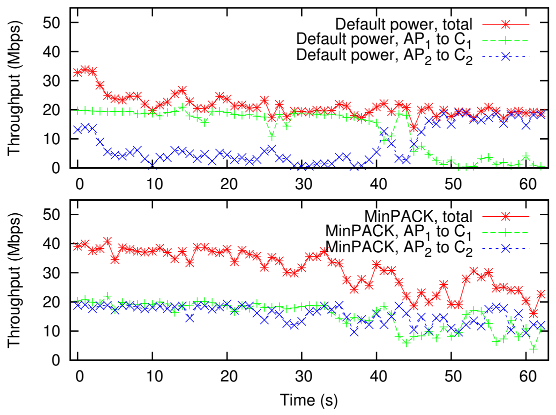


Fig. 15. Performance with mobile client.

V. Related Work

Interference mitigation in practical 802.11 networks has been an important topic in the literature. Like `MinPACK`, some previous proposals also attempt to mitigate interference by controlling transmission power, but they only work with data frames [5, 10, 12, 15, 16], and not MAC ACK frames.

Akella et al. [5] showed that the AP density in selected US cities was quite high some 10 years ago. They proposed a power control algorithm to reduce the power of data frames without affecting the maximum data rate. Ramachandran et al. [16] proposed a similar data-rate-aware power control algorithm, which considers channel asymmetry as well as client mobility. A more recent work on similar joint data rate and power control mechanism can be found in [10]. In the work by Mhatre et al. [15], transmission power and carrier sensing threshold are jointly adjusted to minimize the summed delay of clients. However, it assumes the interference level at AP is the same as that at clients, which may be not true in practice. The work by Kim et. al [12] also jointly controls the power and carrier sensing threshold, but it is based on an ideal unit-disk graph model. Unlike these earlier proposals, our `MinPACK` algorithm regulates the power of MAC ACK frames. There have been previous proposals to reduce ACK interference directly, but they require either ACK grouping [6] or successive interference cancellation [8], both of which cannot easily be implemented with current commodity hardware.

Another category of work on interference mitigation focuses on *synchronizing the transmissions of contending links*. MACA-P [4] inserts a “control phase interval” between RTS/CTS exchange and data frame so that the two competing links can negotiate their common transmission instant. However, such control phase interval (used for every transmission) introduces significant overhead. Shuffle [14] schedules the transmissions of different APs in such a way that there are more concurrent transmissions due to capture effect. Centaur [17] depends on the accurate synchronization to achieve concurrent transmissions in the exposed-node scenario, without disabling carrier sense. However, both Shuffle and Centaur require a powerful central server to achieve tight synchronization among the transmitting nodes, which is not feasible in unmanaged 802.11 networks.

VI. Conclusion

In this paper, we study the interference problem of MAC ACK frames in 802.11 networks. Existing work on interference mitigation only consider data frames, and little attention has been paid to the interference due to MAC ACK frames. To demonstrate the ACK interference problem is serious, we collect extensive warwalking traces and find that the density of 802.11 networks running at the same channel is astonishingly high. We propose a simple and effective power control algorithm, called `MinPACK`, to mitigate the interference due to ACK frames, without affecting the original throughput. Through extensive testbed and real WLAN experiments, we show that `MinPACK` is able to improve both the throughput and fairness performance. Furthermore, `MinPACK` is complementary to available power control algorithms on data frames, and is also adaptive to changing environments.

References

[1] Cisco visual networking index forecast. http://www.cisco.com/go/vni.
[2] Global Wi-Fi hotspot market 2013 report. http://tinyurl.com/otgs8vs.
[3] IEEE 802.11 standard: Wireless LAN medium access control (MAC) and physical layer (PHY) specification, Jun. 2012.
[4] A. Acharya, A. Misra, and S. Bansal. MACA-P: a MAC for concurrent transmissions in multi-hop wireless networks. In *Proc. of PerCom '03*, Mar. 2003.
[5] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in chaotic wireless deployments. In *Proc. of MobiCom '05*, Aug. 2005.
[6] J. M. Brázio, J. L. Sobrinho, L. Matraszek, and L. Byczek. The destructive effect of acknowledgment traffic in WLANs. In *Proc. of GLOBECOM '06*, Dec. 2006.
[7] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin. A first look at traffic on smartphones. In *Proc. of IMC '10*, Nov. 2010.
[8] D. Giustiniano and G. Bianchi. On the exploitation of ACK cancellation for spatial reuse in unplanned multi-hop WLANs. In *Proc. of MedHoc '06*, Jun. 2006.
[9] K. D. Huang, D. Malone, and K. R. Duffy. The 802.11 g 11 Mb/s rate is more robust than 6 Mb/s. *Wireless Communications, IEEE Transactions on*, 10(4):1015–1020, 2011.
[10] T. Huehn and C. Sengul. Practical power and rate control for WiFi. In *Proc. of ICCCN '12*, Jul. 2012.
[11] M. Kim, J. J. Fielding, and D. Kotz. Risks of using AP locations discovered through war driving. In *Pervasive Computing*, pages 67–82. Springer, 2006.
[12] T.-S. Kim, H. Lim, and J. C. Hou. Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks. In *Proc. of MobiCom '06*, Sep. 2006.
[13] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On dominant characteristics of residential broadband internet traffic. In *Proc. of IMC '09*, Nov. 2009.
[14] J. Manweiler, N. Santhapuri, S. Sen, R. Roy Choudhury, S. Nelakuditi, and K. Munagala. Order matters: transmission reordering in wireless networks. In *Proc. of MobiCom '09*, Sep. 2009.
[15] V. P. Mhatre, K. Papagiannaki, and F. Baccelli. Interference mitigation through power control in high density 802.11 WLANs. In *Proc. of INFOCOM '07*, May 2007.
[16] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser. Symphony: synchronous two-phase rate and power control in 802.11 WLANs. In *Proc. of MobiSys '08*, Jun. 2008.
[17] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra. CENTAUR: realizing the full potential of centralized WLANs through a hybrid data path. In *Proc. of MobiCom '09*, Sep. 2009.
[18] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Proc. of MobiCom '06*, Sep. 2006.
[19] G. Yu, W. Wang, J. Yong, B. Leong, and W. T. Ooi. Adaptive antenna adjustment for 3D urban wireless mesh networks. In *Proc. of SECON '13*, Jun. 2013.