White Space Networking with Wi-Fi like Connectivity

Paramvir Bahl[†], Ranveer Chandra[†], Thomas Moscibroda[†], Rohan Murty[‡], Matt Welsh[‡]
[†]Microsoft Research, Redmond, WA [‡]Harvard University, Cambridge, MA
{bahl, ranveer, moscitho}@microsoft.com {rohan, mdw}@eecs.harvard.edu

ABSTRACT

Networking over UHF white spaces is fundamentally different from conventional Wi-Fi along three axes: spatial variation, temporal variation, and fragmentation of the UHF spectrum. Each of these differences gives rise to new challenges for implementing a wireless network in this band. We present the design and implementation of WhiteFi, the first Wi-Fi like system constructed on top of UHF white spaces. WhiteFi incorporates a new adaptive spectrum assignment algorithm to handle spectrum variation and fragmentation, and proposes a low overhead protocol to handle temporal variation. WhiteFi builds on a simple technique, called SIFT, that reduces the time to detect transmissions in variable channel width systems by analyzing raw signals in the time domain. We provide an extensive system evaluation in terms of a prototype implementation and detailed experimental and simulation results.

Categories and Subject Descriptors:
C.2.1 [Computer-Communication Network]: Wireless communication

General Terms: Algorithms, Design, Experimentation

Keywords: white spaces, channel width, Wi-Fi, dynamic spectrum access, cognitive radios

1. INTRODUCTION

The unused portions of the UHF spectrum, popularly referred to as "white spaces", represent a new frontier for wireless networks, offering the potential for substantial bandwidth and long transmission ranges. These white spaces include, but are not limited to, 180 MHz of available bandwidth from channel 21 (512 MHz) to 51 (698 MHz), with the exception of channel 37. On November 4, 2008, the FCC issued a historic ruling permitting the use of unlicensed devices in these white spaces [10]. In its ruling the FCC imposed an important requirement that white space wireless devices must not interfere with incumbents, including TV broadcasts and wireless microphone transmissions. This landmark ruling was a result of extensive tests performed by the FCC on white space hardware prototypes that were submitted by Adaptrum, Microsoft, Phillips and Motorola. These prototypes demonstrated feasible solutions for an accurate and agile sensing of incumbent signals [9].

SIGCOMM'09, August 17–21, 2009, Barcelona, Spain.

Most of the prior research in UHF white spaces has focused on accurately detecting the presence of incumbent RF signals [14, 17, 18]. Recently, researchers have mentioned that they are beginning to look at the problem of establishing a wireless link between white space devices [8,12]. Our research pushes the state-of-art to the next level by going beyond a single link. We identify the challenges of forming a UHF white space network and show how to overcome them by presenting techniques, algorithms, and protocols backed up by extensive evaluation over a prototype network as well as in simulations. We focus primarily on the problem of setting up a Wi-Fi like network consisting of an Access Point (AP) with multiple associated clients. We leave the case of evaluating multiple APs with multiple clients as follow-on work. Our solutions are complementary to the ongoing work in the IEEE 802.22 Working Group [1], as we discuss in Section 7.

To appreciate the networking problem, it is important to understand the differences between white spaces and the popular ISM bands where Wi-Fi devices operate. First, in both bands there is spatial variation in spectrum availability, but the impact of this variation is higher in white spaces than in ISM bands. This is because the FCC ruling requires non-interference with wireless transmissions of primary users (incumbents) (Section 2.1). Second, since the incumbents can operate in any portion of the white spaces, the network must be designed to handle spectrum fragmentation, with the possibility of each fragment being of different width. A UHF channel is narrow (6 MHz wide in the US), and prior research has shown that aggregating contiguous channels improves throughput [15, 21]. Consequently, the network must support variable width channels (Section 2.2). Third, RF transmissions in white spaces are subject to temporal variations because wireless microphones can become active at any time without warning. Our experiments show that even a single packet transmission causes audible interference during wireless microphone transmissions. Consequently, both the AP and its clients must disconnect and then rapidly reconnect using a different available channel (Section 2.3).

We have built WhiteFi, a UHF white space wireless network that adaptively configures itself to operate in the most efficient part of the available white spaces. In the following sections we describe three major innovations that allowed us to overcome the challenges in networking white space devices. Briefly, our contributions are:

- A novel spectrum assignment algorithm for managing variable bandwidth communications. Our algorithm is unique in the way it addresses the dual challenges of spatial variation of available spectrum and spectrum fragmentation. We introduce a new metric that leverages the available airtime measurements from each available UHF channel to predict the available airtime when using multiple channels.

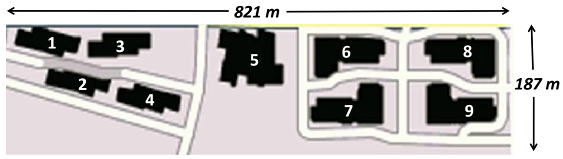

Figure 1: Map of building locations where UHF spectrum was measured.

- A novel AP discovery mechanism. These APs could be using any available channel width and could be operating in any portion of the 180 MHz wide white spaces. We have designed a new technique called SIFT, which is short for *Signal Interpretation before Fourier Transform*. SIFT analyzes incoming signals in the time domain to detect transmissions over different channel widths without changing the channel width of the wireless card. It thus overcomes the core limitation with previous approaches [15] that can only detect packets sent at the same channel width.
- A novel method for handling disconnections. Unexpected disconnections are a direct result of the temporal variations described above. We leverage SIFT to significantly reduce the time to discover APs that have switched to a different part of the spectrum and we have designed a new signaling mechanism that allows clients to signal disconnections to the AP without interfering with ongoing transmissions over wireless microphones.

We have implemented WhiteFi on the KNOWS [20] platform, a hardware prototype for white space networking. This platform incorporates a Wi-Fi card, a UHF band converter, and a software-defined radio (SDR) [5]. We use the KNOWS platform to extensively evaluate the quality and performance of our innovations and design in WhiteFi. To the best of our knowledge, WhiteFi is the first network prototype that demonstrates the feasbility of Wi-Fi like networking over UHF white spaces.

2. CHARACTERIZING WHITE SPACES

In this section, we discuss the differences between the UHF white space spectrum and the ISM bands where current Wi-Fi systems operate. To understand the differences, we performed a set of real-world measurements in the UHF bands in several different settings to characterize spatial and temporal variation. We also analyzed publicly-available TV spectrum allocation data to understand the distribution of UHF spectrum usage in urban, suburban, and rural settings in the United States. We describe how each of these characteristics has a substantial impact on the design of a UHF white space wireless network.

2.1 Spatial Variation

Television stations represent the largest incumbent use of the UHF spectrum. Across a wide area, the set of occupied TV channels depends on the location of TV transmitters as well as the number of stations operating in an area. However, spatial variation exists on smaller scales as well, based on obstructions and construction material. Wireless microphones, which are used in settings ranging from small-scale lecture rooms to large-scale music and sporting events, have typical transmission ranges of a few hundred meters [16]. For these reasons, we expect significant spatial variation in spectrum availability for wireless network communications.

To quantify this variation, we performed measurements of the UHF spectrum inside 9 buildings on our campus spanning an area of approximately 0.9 km $\times$ 0.2 km, as shown in Figure 1. Note this entire area could be covered by a single suitably positioned UHF white space AP, since communication ranges are expected to exceed 1 km [2]. We computed the Hamming distance, defined as the number of channels available at one location but unavailable at another, across all pairwise buildings. Our results showed that the median number of channels available at one point but unavailable at another is close to 7. This statistic reveals significant variation in spectrum availability within nearby buildings. While most incumbents detected in these measurements were TV channels, we also found a few wireless microphones.

The implication of this spatial variation for a white space wireless network is that an AP (a home wireless router for example) must not naively select channel(s) to operate on based solely on its own local observation of spectrum availability. The AP must take into account the availability of spectrum at its clients as well.

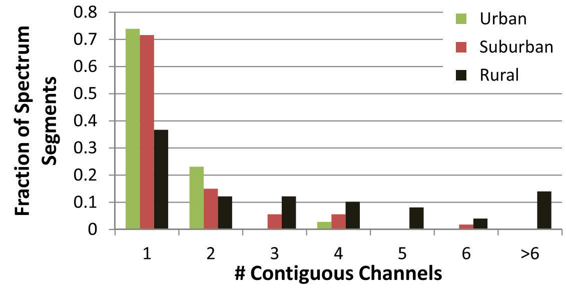

Figure 2: Expected spectrum fragmentation after the US DTV transition in June 2009. Rural and suburban regions exhibit a much lower degree of fragmentation and more contiguous spectrum than urban areas.

2.2 Spectrum fragmentation

While the ISM bands are a contiguous chunk of spectrum, UHF white spaces are fragmented due to the presence of incumbents. The size of each fragment can vary from 1 channel to several channels. The amount of fragmentation in the UHF bands depends to a large extent on the density of TV stations, which varies considerably with population density. Rural (and suburban) areas, are likely to have larger chunks of available UHF spectrum than urban areas. In addition, the US digital television transition [7], scheduled to be completed in June 2009, will open up much more of the UHF spectrum, as a number of analog TV stations will stop operating in these bands.

To quantify the spectrum fragmentation after the DTV transition, we analyzed TV station data from TV Fool [4], a website that uses sophisticated signal and terrain modeling to estimate the availability of TV channels at a given latitude and longitude. Based on this dataset, we estimate UHF spectrum fragmentation in 3 settings: urban (top 10 populated cities), suburban (10 fastest growing suburbs based on the 2007 Forbes list) and rural (10 random towns in the US with a population less than 6000). Figure 2 shows a histogram of the contiguous spectrum widths that will be available in each of these settings. As we see in the figure, in all 3 settings there is at least one locale in which there is a fragment of 4 contiguous channels available, that is, 24 MHz of spectrum. In rural areas fragments of up to 16 channels are expected.

A consequence of this fragmentation is that radios need to tune the spectrum that they occupy to fit within available fragments. This implies the need for radios to use variable channel widths [15] or channel bonding. Compared to Wi-Fi, the use of variable channel widths introduces two new challenges. First, it makes channel assignment more challenging, since APs now occupy a range of channels, rather than just one. Second, it increases the the time taken for nodes to discover APs. This is due to a limitation of techniques that can achieve variable channel widths on Wi-Fi cards [15]. Using this technique a radio can only decode packets that are sent at the *same* channel width and *same* center frequency. An expensive switch of the PLL clock frequency is required to decode packets at other channel widths. In Section 4 we show how WhiteFi overcomes these problems.

2.3 Temporal Variation

Finally, the UHF white spaces also suffer from temporal variation, in particular due to the widespread use of wireless microphones (mics) – from lecture rooms in campuses to musicians at home, and from sporting events to churches. We performed measurements of the UHF spectrum in two settings: the campus setting described earlier and a University dormitory, over several days. We used the prototype described in Section 3 to determine the incumbents. In both cases, we detected the use of wireless mics at different times of day and for different durations.

Wireless mics can be turned on at any time. Since in its initial ruling the FCC requires that white space devices avoid interfering with mic transmissions, both clients and APs should detect the presence of a mic on a channel and move away from that channel. Furthermore, if only a client or an AP detects a mic, each must have a means of informing the other of the channel switch without inducing interference.

Unfortunately, simple solutions to this problem are not feasible in practice. For example, one approach is for an AP to avoid using channels where wireless mics might be used. However, simply blacklisting known wireless mic channels is overly conservative and makes inefficient use of the spectrum, since mics tend to be used intermittently, for limited durations, and on any UHF white space channel. A more sophisticated approach would build a historical database of mic usage patterns that APs can query to determine the channels that are used at any instant in time. However, our measurements show that mic use is highly unpredictable. For example, although use of wireless mics in campus lecture rooms might follow a predictable schedule, each room tends to be over-provisioned with multiple mics on different channels, and A/V operators choose only a few of those mics for an event. Furthermore, it is impractical to predict many other uses of wireless mics, such as for special events, musical performances, or rehearsals; each again may use a multitude of mics on various channels.

The second possibility, which is being considered by the IEEE 802.22 working group [1], involves an explicit channel renegotiation protocol between clients and APs when they detect a wireless mic. This approach assumes that control messages will not induce audible interference on the wireless mic. To test this assumption, we performed an experiment by placing a wireless mic receiver along with our prototype WhiteFi device (Section 3) in an anechoic chamber. We measured the audio quality of recorded speech transmitted over the wireless mic with and without UHF transmissions. For UHF transmissions, we sent 70-byte packets every 100 ms on the same UHF channel as the mic. The transmission power level was -30 dBm, which is below the FCC-permitted maximum of 40 mW (16dBm). We found that the perceived audio quality degrades with data transmissions. The Mean Opinion Score (MOS) of the received audio, computed using Perceptual Evaluation of Speech Quality (PESQ), decreased by 0.9 during the UHF packet transmissions. Other researchers have shown that a MOS reduction of only 0.1 is noticeable by the human ear [22]. We note that these results may be worse than what one would observe in practice. In our experiments the antenna of the data transmitter and the mic receiver were within a few feet (in the same anechoic chamber). We are actively working on acquiring an experimental license

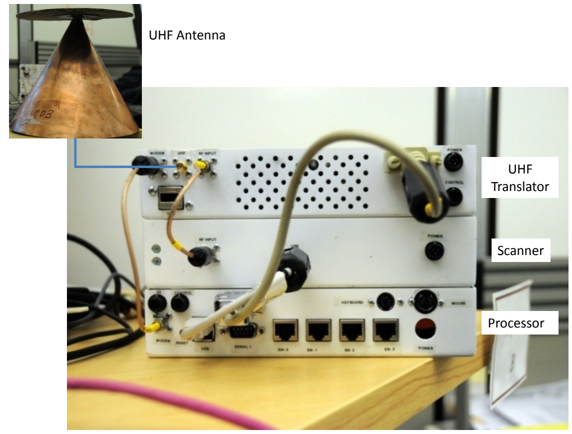

Figure 3: Photograph of the KNOWS hardware prototype.

from the FCC and repeating these experiments in more realistic and normal settings.

One could argue that interference in the beginning of a mic recording might not be cause for concern. However, when clients are mobile, a mic may be sensed only in the middle of a recording. Furthermore, such a naive approach relies on all nodes in the vicinity of a mic detecting its appearance at the same time. If not all detect the mic synchronously, then each node transmits one after the other, thereby inducing further interference with the mic.

These results demonstrate the need for a protocol that can signal the presence of a wireless mic to the network without interfering with the mic. We present such a protocol in Section 4.3.

3. KNOWS HARDWARE PLATFORM

UHF white space networking currently requires specialized hardware support, and several hardware prototypes have been reported in the literature [8, 12, 20]. All these devices have a transceiver radio and a separate scanner radio. The need for a separate scanner stems from the requirement to quickly and accurately detect the presence of primary users.

To support WhiteFi, we developed the KNOWS hardware prototype as shown in Figure 3. As this platform is described in more detail in prior work [20], we briefly recount the main features in this section. The hardware consists of three components: a PC, a scanner, and a UHF translator. The PC is used both to control the scanner and to transmit and receive packets over the UHF bands. The PC comes equipped with a standard 2.4 GHz Wi-Fi card, the antenna port of which is connected to the UHF translator, which downconverts the outgoing 2.4 GHz signal to the 512–698 MHz band. Incoming signals are likewise upconverted and passed to the Wi-Fi card. The center frequency of the UHF translator is set from the PC via a serial control interface. To ensure that the outgoing signal fits within a 6 MHz UHF channel, we use the technique presented in [15] of changing the PLL clock frequency to reduce the Wi-Fi transmission bandwidth to 5 MHz.

The scanner samples the UHF spectrum to detect the presence of TV broadcasts and wireless microphone signals. It is implemented using the USRP [5] software-defined radio board coupled with a 50–800 MHz TVRX receiver-only daughter board. The scanner scans UHF TV channels 21–51 in 6 MHz increments. Due to the USRP bandwidth constraint [6], the frequency span for each scan is 8 MHz. We perform the FFT on the PC, and using the feature detection algorithms described in [20], our scanner is able to detect TV signals at signal strengths as low as -114 dBm, and wireless microphones at -110 dBm. We note that this is much below the TV

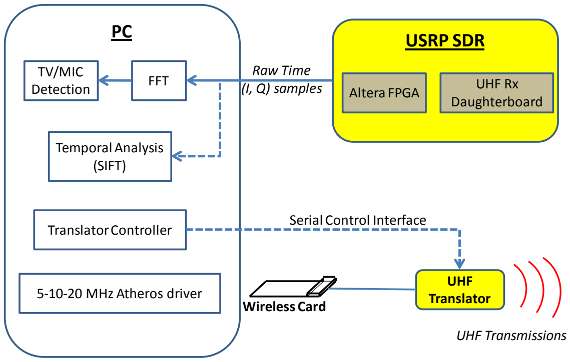

Figure 4: Functional block diagram of the KNOWS platform.

decoding threshold of -85 dBm. This 30 dB detection buffer is required to solve the classic hidden terminal problem, in which a TV is within transmission range of the TV tower but the transmitting device is not.

Sensing of incumbents, especially microphones, is an actively researched problem. Recent proposals use energy detection to detect the primary users [17, 18]. However, such an approach is prone to false positives, especially given the extremely low detection thresholds that have been set forth in the FCC report [9]. False positives reduce the amount of available white spaces to form a white space network. Furthermore, false alarms might cause WhiteFi to vacate the channels (Section 4.3). This switching overhead might affect the performance of associated clients.

Several solutions attempt to avoid false positives. One approach proposed from Motorola requires microphones to beacon at high power when they are being used [13]. Researchers from Berkeley have proposed collaborative sensing to improve sensing accuracy [14]. The FCC is looking at the use of a geo-location database to regulate and inform clients about the presence of primary users [9].

In this paper, we do not address the problem of accurate incumbent detection, which remains an active research area. Instead, we focus on the networking challenges that arise assuming a reasonably accurate incumbent detection technique. We expect WhiteFi to benefit from future advances in incumbent detection.

To enable efficient networking over white spaces, our platform has two key features unavailable in previous systems [8, 12, 20]:

Variable Channel Widths: Existing systems can only use one UHF channel, even when multiple contiguous UHF channels are unoccupied. This is because the bandwidth of the outgoing signal is fixed to be 5 MHz. To support multiple contiguous channels, we modified the Atheros Wi-Fi driver using the techniques presented in [15] to transmit and receive signals of bandwidth 5, 10 and 20 MHz. As we show in Section 5, this provides substantially greater throughput than single-channel systems 5.

Signal Inspection before Fourier Transform (SIFT): Existing systems detect signatures of primary users in the frequency domain, after performing a Fast Fourier Transform (FFT) on the time series signal. However, such scanners cannot detect data transmissions for two reasons. First, in contrast to TV and microphone transmissions, data transmissions are intermittent. Therefore, it is difficult to distinguish intermittent data from noise using prior detection techniques. Second, data transmissions in our system can be sent over multiple channel widths. Unless the entire signal is received, including all subcarriers, data packets cannot be decoded. To address these concerns, we propose SIFT, which processes raw signals in the time domain and extracts data information from them. We describe this technique in detail in Section 4.2.1.

4. WhiteFi DESIGN

In this section, we describe the WhiteFi design in detail. WhiteFi is an implementation of a Wi-Fi like protocol on top of the UHF white spaces that addresses the key challenges described earlier. We design our system on the hardware described in the previous section, with one transceiver and one scanner. Also, we focus on systems with a single data rate (since rate adaptation itself is an open problem in white spaces).

Our network architecture is based on three key components. First, WhiteFi incorporates a novel spectrum assignment algorithm that is able to handle spatial variation of the spectrum as well as spectrum fragmentation. Second, WhiteFi uses an efficient, time-domain signal analysis technique, called *SIFT (Signal Interpretation before Fourier Transform)*, that allows clients to rapidly discover APs transmitting on a range of channel widths. Third, WhiteFi provides a *chirping* protocol that permits a client to indicate a sudden disconnection from the AP due to a channel conflict with an incumbent, such as a wireless microphone, without interfering with the primary user.

In the following, we use the term *channel* to represent a range of the UHF spectrum on which a WhiteFi AP or client communicates. A channel is represented as a tuple (F, W), where F is the center frequency, and W is the width of the channel. In our current implementation, W can be either 5 MHz, 10 MHz, or 20 MHz, but our hardware is generally capable of using more channel width options. In contrast, the term *UHF channel* indicates one of the 30 segments of the UHF spectrum, which are each 6 MHz wide. Note that in our current hardware implementation, channels are always centered at a UHF channel's center frequency. Hence, a 5 MHz WhiteFi channel can fit within a single UHF channel, a 10 MHz channel spans 3 UHF channels, and a 20 MHz channel spans 5 UHF channels.

4.1 Spectrum Assignment

As shown in Section 2, the problem of selecting an appropriate transmission channel is significantly harder in white spaces than in regular Wi-Fi. Because of temporal and spatial variability in spectrum availability, the AP must pick a channel that is free for all its clients. Moreover, fragmentation leads to different-sized spans of available white spaces, so the AP also has to decide on the best possible channel width to use as well. Always using the widest channel that is available for all clients may not be the right solution, since there could be significant background traffic (from other APs) on some of the underlying UHF channels. Similarly, always picking the narrowest channel width (i.e., a single UHF channel) may be wasteful if there are wider channels available.

These challenges motivate an *adaptive* spectrum-assignment algorithm that periodically reevaluates the assignment based on white space availability at the AP and clients. The algorithm also has to be *client-aware* since the AP cannot simply rely on its own observation of the available UHF channels. This requires clients to share information with the AP on their own observed UHF channel availability.

In prior work, SampleWidth [15] solves the channel width assignment problem for a pair of nodes. In WhiteFi, we look at the broader problem of selecting both the center frequency and channel width when there are more than two nodes.

Preliminaries: The AP and each client maintains a *spectrum map* which is a bit-vector $\{u_0, \ldots, u_k\}$ where each u_i represents whether the corresponding UHF channel is currently in use by an incumbent user (that is, a TV channel or wireless microphone). $u_i = 1$ if the channel is in use by an incumbent, and 0 otherwise. In the United States, there are 30 UHF channels represented in the spectrum map. Each node also maintains an *airtime utiliza-*

tion vector $\{A_0, \ldots, A_k\}$, where A_i represents an estimate of the airtime utilization on each UHF channel. Note that for incumbent-occupied channels, A_i is undefined. The spectrum map and airtime utilization are measured using the secondary scanning radio, using the SIFT technique described in Section 4.2.1 below.

Triggering new channel selection: An AP decides to probe for a new channel when one of two conditions occurs. The first is an *involuntary* channel switch induced by an incumbent (such as a wireless microphone) becoming active anywhere on the AP's current channel (F, W). Likewise, if a client detects an incumbent, it will disconnect from the AP and cause a channel switch to occur. This process is described in Section 4.3. The second is a *voluntary* channel switch, which is triggered when the AP detects a performance drop on its current channel. The AP periodically probes for a new potential channel as well, in case another portion of the spectrum has opened up since its last probe that could yield higher performance. Of course, an AP also performs channel selection when booting up.

Channel probing: To probe for a new potential channel, the AP must have information about the spectrum map and airtime utilization observed at each of the clients. Clients periodically transmit this information to the AP as part of a control message. When bootstrapping, the AP will not have any clients and will perform channel selection without client input.[1]

The first step is to take the bitwise OR of the clients' and AP's spectrum maps, $\mathbf{u}^\star$, to determine the set of UHF channels available at *all* of the nodes.

The second step is to consider each possible channel (F, W) in the available white spaces, and estimate the aggregate bandwidth that the AP and clients would receive if selecting that channel. The challenge is that the probed channel (F, W) might overlap partially or completely with channels occupied by other APs; we do not disallow channel overlaps between APs. For this reason, estimating aggregate bandwidth based on airtime utilization measurements in UHF white spaces is harder than, say, in the 2.4 GHz Wi-Fi band.

For a given 6 MHz UHF channel c and a node n, we define $\rho_n(c)$ as the *expected share* of c that node n will receive if c is contained within (F, W). This is a function of the busy airtime A_c^n on the channel c as measured at n, as well as the estimate of the number of other access points operating on c, which we denote B_c^n. This value can be determined for instance by using the scanning radio and the SIFT technique (Section 4.2.1). For every node n and any UHF channel c, we define

$$\rho_n(c) = \max(1 - A_c^n, \frac{1}{B_c^n + 1}). \tag{1}$$

The intuition behind this definition is as follows. At any instant in time, the probability that a node will be able to transmit on the channel c is at least the residual airtime $1 - A_c^n$. This is a good estimate for n's expected share when the channel is mostly free. However, even when the medium is completely utilized by neighboring APs (A_c^n is 1) a node can still expect to get its "fair share" of the airtime when it is contending with them.[2] Therefore, we take the maximum of these two values as an estimate of the probability that a node will be able to use the channel c on each transmission opportunity.

Given a range of UHF channels spanned by the probed channel (F, W), therefore, we define the *multichannel airtime metric*, or $MCham_n(F, W)$, at node n as

$$MCham_n(F, W) = \frac{W}{5\text{MHz}} \cdot \prod_{c \in (F,W)} \rho_n(c) \tag{2}$$

Since $\rho_n(c)$ represents the expected share of a UHF channel c, the product of these shares across each UHF channel in (F, W) gives the expected share for the entire channel. We note that simply taking the minimum or the maximum across all channels, instead of the product, will be an underestimate since the traffic on a narrower channel contends with traffic on an overlapping wider channel [15]. The product is then scaled by the optimal capacity of the probed channel, $W/5\text{MHz}$. We use a 5 MHz channel as our reference point because it fits into one single UHF channel.

Example 1: If there is no background interference or other APs occupying any portion of (F, W), then $MCham_n(F, W)$ simply evaluates to the optimal channel capacity. That is, $MCham_n(F, W) = 1$ for W=5 MHz, 2 for W=10 MHz and, and 4 for W=20 MHz.

Example 2: Consider a channel $c = (F, 20MHz)$. Out of the 5 UHF channels that are spanned by c, let three have no background interference, one has 1 AP and airtime utilization of 0.9, and one has 1 AP with airtime utilization 0.2. $MCham_n(F, 20MHz) = 4 \cdot 0.5 \cdot 0.8 = 1.6$. That is, the metric predicts a throughput on this channel that is equivalent to roughly 1.6 times of an empty 5 MHz channel.

Channel selection: The AP evaluates $MCham(F, W)$ for each possible channel (F, W) in the available white spaces, and selects the channel that maximizes this metric. In order to also include the values measured at the clients (for upstream traffic), the AP bases its decision on an average value of all its clients $MCham_n(F, W)$ value, as well as its own value, $MCham_{AP}(F, W)$. Since most traffic in today's wireless networks is on the downlink [11], the AP weights its own $MCham$ proportionally higher. In our implementation, the AP selects a channel that maximizes: $N * MCham_{AP} + \sum_n MCham_n$ where N is the number of clients attached to the AP. However, notice that other metrics (such as metrics including fairness conditions) can easily be implemented instead of aggregate throughput.

The AP broadcasts the new channel to its clients, which upon hearing the message, switch to the new channel. (If a client misses the channel switch message, it will revert to the disconnection protocol described in Section 4.3.) In the case of a voluntary channel switch, if the measured performance of the new channel is less the previous channel, the AP will re-evaluate its channel selection, possibly switching back to the original channel. In our prototype of WhiteFi, the AP measures the aggregate throughput achieved by all clients as a measure of the effectiveness of a channel switch. To prevent frequent changes in the channel or ping-ponging across two channels, we also add hystersis to our system as done in [19].

In our evaluation in Section 5.4, we show that the *MCham* metric predicts the best possible channel to a degree of accuracy that is sufficient for the above WhiteFi spectrum assignment algorithm to achieve near-optimal throughput in a wide variety of test cases.

4.2 AP Discovery

The use of variable channel widths in WhiteFi presents a new challenge when performing AP discovery. Traditional Wi-Fi clients perform access point discovery by scanning each channel and listening for periodic beacons from APs, which are typically transmitted every 100 ms. The key difference in WhiteFi is that the AP may be using either a 5 MHz, 10 MHz, or 20 MHz channel width

[1] It is possible that the AP selects a channel that is blocked for all or some of its potential clients, a case that is handled by the disconnection mechanism described in Section 4.3.

[2] Since today's wireless networks are dominated with downlink traffic [11], and it is difficult to measure the number of interfering clients, we estimate the number of contending nodes as B_c^n, i.e. the number of interfering APs.

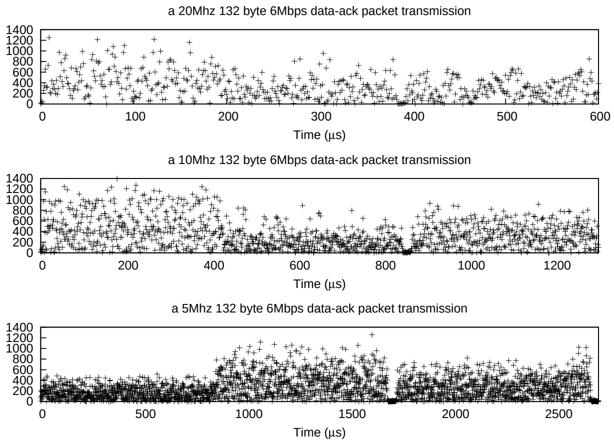

Figure 5: A time-domain view of Data-ACK frames sent at 6Mbps OFDM at different widths. The Y-axis is $\sqrt{I^2 + Q^2}$ for every sample.

for its communications, including beacon transmissions. If not performed efficiently, AP discovery time could be substantial. Given 30 UHF channels and 3 possible channel widths, there are 84 combinations to consider.[3] This approach uses the prototype's Wi-Fi card to perform AP discovery, but suffers the high cost of scanning every (F, W) channel combination.

An alternative approach would be to leverage the SDR in the node's scanner to capture a trace of the signal across a band, and then apply real time OFDM decoding, in software, on successive channel center frequencies and widths to detect an AP. However, this would incur substantial computational overhead; performing OFDM decoding in software at 802.11a PHY rates requires multiple cores of a well-provisioned server-class machine [23]. Moreover, since the SDR hardware can only sample an 8 MHz range of spectrum at a time, multiple such scans would be required.

4.2.1 SIFT: Efficient Variable-Bandwidth Signal Detection

We propose a hybrid solution that uses the SDR to sample a given 8 MHz band, but performs an efficient *time-domain analysis* of the raw signal to detect the presence of an AP and determine its channel width. This approach avoids the high overhead of decoding beacon packets in software, while making efficient use of the SDR's capabilities. Once the AP's channel (F, W) has been identified, the radio transceiver is tuned to that channel and decodes the beacon packets in hardware.

This approach, which we call *Signal Interpretation before Fourier Transform*, or *SIFT*, works as follows. For a given center frequency F, the USRP board samples a bandwidth of 1 MHz around F at 1 MSamples/sec. Each sample represents 1.024 μs of raw RF signal as an (I, Q) pair; the signal amplitude is computed as $\sqrt{I^2 + Q^2}$. The USRP delivers blocks of 2048 samples at a time to the PC.

SIFT uses a simple detection algorithm that determines packet widths based on signal amplitudes. To accurately detect the beginning and end of a packet transmission, we compute a moving average over a sliding window of the signal amplitude values. We do not use instantaneous values, since the signal amplitude might fall to very low values even in the middle of the packet transmission (Figure 5). The start of a packet transmission is detected when this average increases beyond a certain threshold. Similarly, when the average falls below the threshold, the algorithm marks it as an end of a packet. In our current implementation this threshold is fixed at a low value. We are actively working on techniques to dynamically adjust the threshold based on background noise levels.

A key question is, how do we determine the size of this sliding window? Since the 802.11 SIFS duration determines the time between the end of a data packet and the start of the subsequent acknowledgement, both of which we want to detect accurately, we limit the size of the sliding window to less than the *minimum* possible SIFS value in our system. As prior work has shown [15], SIFS values change across different channel widths and the lowest SIFS value in our system is for a 20 MHz transmission, which is 10μs or 10 samples. Hence, we choose a window size of 5 samples. Once the algorithm determines the start and end time of a packet, the duration of the packet is known. From this we also glean information about the interval between a data packet and its acknowledgement.

Both the packet duration and the SIFS interval are inversely proportional to the channel width. This information can be used to infer the channel width on which the packet was transmitted. For example, by matching the delay between the data and its acknowledgement packet, and the duration of the acknowledgement packet, we can determine the channel width of the unicast transmission.

The reason this technique works is twofold. First, the acknowledgement packet is the smallest MAC layer packet (14 bytes), and cannot be confused with a data transmission. Also, the duration of an acknowledgement packet at the narrowest width of 5 MHz is still much smaller than any data packet sent at 20 MHz. Second, the SIFS interval is different on every width and reduces the probability of any false positives. We use a similar technique to match against non-data packets such as beacons. We require APs to send a short packet, such as a CTS-to-self, one SIFS interval after sending a beacon packet.

We expect SIFT to have very few false positives since it matches both the ACK duration and the interval between the packet and ACK. However, in extremely noisy environments or in the presence of concurrent transmissions, SIFT might have false negatives. It could fail to accurately detect all transmissions. We note that although this will add delay to the time for discovering APs (Section 4.2.2) the discovery algorithm will continue to work as long as we can detect even a single packet.

When SIFT samples an 8 MHz band centered at a frequency F_s, it will be able to detect a WhiteFi transmitter whose channel overlaps with F_s, even though their center frequencies may not match. For example, when SIFT detects a 20 MHz WhiteFi channel at F_s, the true center frequency F_c of the WhiteFi transmitter can be anywhere in the range $F_s \pm 10$MHz. Therefore, the output of the SIFT algorithm is $(F \pm E, W)$ where F is the center frequency of the transmitter, E is an error term, and W is the transmitter's channel width (5, 10 or 20 MHz). Since W can be determined exactly by SIFT, $E = \pm W/2$.

We demonstrate the accuracy and performance of the SIFT algorithm in Section 5.1.

4.2.2 AP Discovery using SIFT

SIFT enables clients to discover APs without tuning into all possible (F, W) channel combinations. Based on the SIFT primitive, we devise two AP discovery algorithms, as described below. Throughout our discussion, N_C denotes the number of UHF channels (30 in the United States) and N_W represents the number of channel widths (three in our implementation).

Linear SIFT-Discovery Algorithm (L-SIFT): This algorithm simply scans each of the 30 UHF channels in succession, attempt-

[3] There are a total of 30 5MHz WhiteFi channels, 28 10MHz channels, and 26 20MHz channels.

Algorithm 1 J-SIFT Algorithm:

UHF channels are numbered $0, \ldots, N_C$.
$w_0, \ldots, w_{N_W}$: channel width options (5, 10, 20 MHz)
S: Set of UHF channels already scanned.

SIFT search:
$j := N_W$; $c := 0$; $S := \{\}$;
while AP not detected **and** $j \geq 0$ **do**
    $c := 0$;
    while AP not detected **and** $cur < N_C$ **do**
        if $cur \notin S$ **then**
            SIFTscan(cur);
            $S := S \cup \{cur\}$;
            if AP not detected
                $cur := cur + w_j$;
            end if
        end if
    end while
    $j := j - 1$;
end while

Determining AP's center frequency:
Let cur be channel on which SIFT detected an AP
Let W be the AP's channel width reported by SIFT
$k := 0$;
while AP beacon not decoded **do**
    Listen for AP beacons on channel
        $[cur - W + k, cur + k]$;
    $k := k + 1$;
end while

ing to detect an AP using the SIFT technique at each one. Because L-SIFT scans the spectrum from lower frequencies to higher frequencies, as soon as a transmitter is detected, its center frequency F_c is known: $F_c = F_s + E$, where F_s is the frequency that SIFT was scanning and E is the uncertainty returned by SIFT. The expected number of iterations until an AP is discovered is $N_C/2$, and the worst case is N_C (compared to roughly $N_C \cdot N_W/2$ and $N_C \cdot N_W$, respectively, by the non-SIFT baseline).

Jump SIFT-Discovery Algorithm (J-SIFT): We can improve upon the expected scan time of L-SIFT by performing a staggered search of the spectrum. Since SIFT is able to detect a WhiteFi transmitter by scanning anywhere within its band, we can improve performance by first scanning for 20 MHz WhiteFi channels (skipping over 5 UHF channels at a time), then 10 MHz channels (skipping over 3 UHF channels at a time as well as any UHF channels previously scanned), and finally for 5 MHz channels (in the remaining unscanned UHF channels).[4]

One disadvantage to J-SIFT is that the WhiteFi transmitter's center frequency is not immediately known when it is detected. Therefore, it is necessary to tune the radio to each of $F_s \pm E$ channels and attempt to decode packets to exactly determine the center frequency.

J-SIFT works as presented in Algorithm 1. It operates in two phases. First, it scans the UHF spectrum in a staggered fashion, using SIFT to detect the presence of a WhiteFi transmitter. In the second phase, it identifies the transmitter's center frequency F_c. While the worst-case discovery time of J-SIFT is the same as for L-SIFT (N_C), the expected discovery time can be shown to be $\frac{1}{N_W}(N_C + 2^{N_W-1} + (N_W - 1)/2)$. We elide the derivation due to lack of space.

In WhiteFi, we expect the average number of scans required for L-SIFT and J-SIFT to be $N_C/2$ and $(N_C + 4 + 1)/4$, respectively. That is, we expect J-SIFT to outperform L-SIFT when N_C is greater than about 10 UHF channels. For narrower white spaces, L-SIFT is more efficient. Our measurements in Section 5.2 validate these theoretical findings.

4.3 Handling Disconnections

A key challenge in WhiteFi is dealing with the sudden appearance of a primary user (such as a wireless microphone) on a channel that an AP-client pair is using for communications. Note that either the AP or the client might detect the primary user, requiring that a channel must be vacated. We call this a *disconnection.*

Our approach is as follows. The AP maintains a separate 5 MHz *backup channel* that is advertised as part of its beacon packets on its main channel. If the AP or a client detects a primary user on the main channel, the node switches to the backup channel and transmits a series of *chirps* that contain information on the white spaces available at that node.

If a client senses that a disconnection has occurred (e.g., because no data packets have been received in a given interval), it switches to the backup channel and listens for chirps, as well as transmitting its own. Access points periodically scan for chirps on the backup channel, in a manner similar to that being considered by the 802.22 working group [1]. To avoid disrupting communications with still-connected clients, chirp detection is performed using SIFT on the secondary radio, in the background. Once a chirp has been detected, the AP can switch its main radio to the backup channel and decode the contents of the chirp packet. As a further optimization, we can encode some amount of information in the time domain, such as the client's SSID, for example by setting the length of the chirp packet. (In effect, this uses SIFT to implement a low-bitrate OOK-modulated channel.) This approach avoids switching the main radio to the backup channel for clients associated with a different AP.

Once a node begins chirping, after a threshold time interval T_c, the collective white space availability advertised by each node on the backup channel is used to reassign spectrum to the AP and clients in that SSID, as described in Section 4.1. Nodes in the SSID switch to the new channel and resume communication.

There is an additional case we must consider, namely, when a node (either the AP or the client) determines that the previously-selected backup channel is occupied by another primary user. In this case, an arbitrary available channel is selected as a *secondary backup* and used for chirping. Therefore, in addition to scanning the backup channel for chirps, the AP periodically scans all channels in an attempt to reconnect with "lost" nodes. Note that chirps contend for the channel using CSMA, just like data packets; as a result, it is unproblematic for a backup channel to overlap with another AP's main channel.

An attacker can potentially hijack our system by sending fake chirps. However, the impact of this attack is limited. Once the AP's main radio switches to the backup channel, it will process the chirp packet only if it is encoded with the network's security key (similar to Wi-Fi). Therefore, the overhead of this attack is the extra time taken to switch across channels, which is known to be a few milliseconds. We realize that this overhead can be avoided by adding security features to SIFT, so that only an authorized client will cause the AP to switch its main radio. We are actively investigating this approach.

5. EVALUATING WhiteFi

In this section, we evaluate WhiteFi in detail. Using a combination of simulations and experiments on our prototype implementation, we show:

[4] Generally, if more widths are available, we would do the staggered search starting from the widest channel width.

	0.125 M	0.25 M	0.5 M	0.75 M	1 M
5 MHz	0.99	0.98	0.98	0.98	0.97
10 MHz	0.99	0.99	0.99	1.00	0.99
20 MHz	0.99	1.00	0.99	1.00	0.99

Table 1: SIFT's packet detection rate, i.e., the median number of packets detected by SIFT divided by the total sent by the wireless card. The values are measured across different widths when varying the traffic intensity from 125 Kbps to 1 Mbps.

- In Section 5.1, we demonstrate the accuracy of SIFT in detecting packets across different channel widths and when there is high signal attenuation.
- In Section 5.2, we show the effectiveness of WhiteFi's AP discovery algorithms. Our experiments show that J-SIFT improves the time to discover APs by more than 75% compared to non-SIFT based techniques.
- We demonstrate the correctness of WhiteFi's protocol in handing disconnections in Section 5.3.
- Finally, in Section 5.4, we show that WhiteFi's spectrum assignment algorithm adapts quickly to changes in network conditions. Using extensive simulations in QualNet, we also show that WhiteFi's performance is close to optimal under various conditions.

5.1 Accuracy of SIFT

We evaluate the packet detection accuracy of SIFT. We first describe our methodology and then show its accuracy when varying two parameters: channel width and signal attenuation.

Methodology: We used the following set up for our experiments. We started an iperf session from one KNOWS device, and measured the number of packets that were received at a second device using a packet sniffer. Simultaneously, we used the scanner of the second device to count the number of packets detected by SIFT. We repeated this experiment for 5, 10 and 20 MHz channel widths, and for each width, we varied the traffic intensity. All the reported numbers are over 10 runs. In every run, we sent 110 packets of size 1000 bytes each.

Accuracy across Channel Widths: Table 1 shows the fraction of the number of packets detected by SIFT when varying the rate at which packets were sent across different channel widths. As we see in the table, SIFT detects nearly all the packets for every channel width. The worst case loss across all widths and rates was 2%. An interesting observation is that the detection rate for 5 MHz was slightly worse than the detection rate at other channel widths. This was a result of the way 5 MHz packets are transmitted by our hardware in the time domain. As we see in Figure 5, the initial portion of a packet at 5 MHz channel width is sent at a lower amplitude than the rest of the packet. Consequently, our algorithm sometimes fails to accurately match the length of the detected packet and the transmitted one. However, SIFT always correctly detects the channel width of the transmitted packet, even when it mis-estimates the packet length.

In addition to detecting the appropriate width, we also use SIFT to measure the airtime utilization for WhiteFi's spectrum assignment algorithm. We show that SIFT performs as expected in Figure 6. The total time occupied by the packets doubles on halving the channel width. This stems from the observation in [15] that halving the channel width also halves the effective transmission rate. Since we send the same number of packets at a given width, the total airtime is constant, even when we change the rate of injected packets.

Accuracy with Signal Attenuation: We evaluated the accuracy of SIFT at low signal strengths by connecting two KNOWS devices

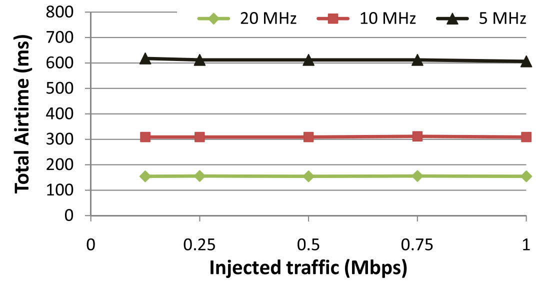

Figure 6: Accuracy of air time utilization measurement using SIFT. Error bars were within 2% of the mean.

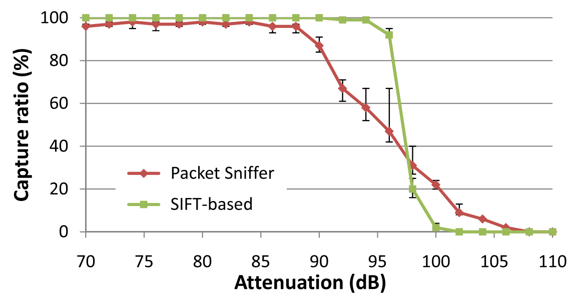

Figure 7: Discovery of APs with distance. SIFT is able to discover APs until as long as the Wi-Fi card can decode packets.

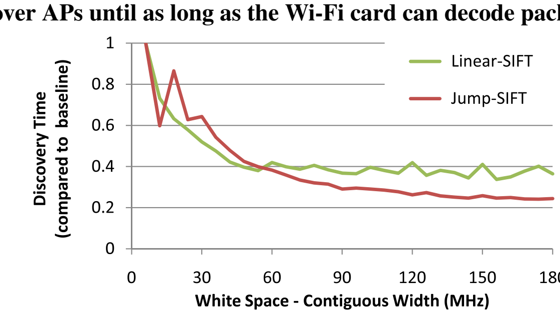

Figure 8: Reduction in discovery times using L-SIFT and J-SIFT when compared to the non-SIFT based baseline.

through a tunable RF attenuator, and performing the same experiment as above. Figure 7 shows the percentage of packets that were detected by SIFT and the packet sniffer upon varying the attenuation. At low attenuation, both SIFT and the packet sniffer perform very well. However, SIFT outperforms the packet sniffer, as it is even able to detect corrupted packets. At higher attenuation, SIFT continues to detect more packets than the sniffer until 96 dB attenuation.

Since SIFT applies a threshold to the amplitude of the incoming signal, it performs poorly beyond a certain attenuation. In our experimental setup, this occurs at 96 dB. Beyond 96 dB we see a very sharp drop in the percentage of successfully detected packets. In contrast, the reception ratio of the packet sniffer falls off more smoothly, and performs better than SIFT beyond 98 dB attenuation. However, at this attenuation the capture ratio is extremely low at around 35%. Most applications, including TCP, will perform poorly at such high loss rates. Hence, we conclude that in most commonly occuring scenarios, SIFT detects almost all packets that are successfully received by a transceiver radio.

5.2 Time to Discover APs

We now evaluate the performance of the L-SIFT and J-SIFT discovery algorithms in discovering APs. We compare them to a non-SIFT baseline that would have to scan every possible center fre-

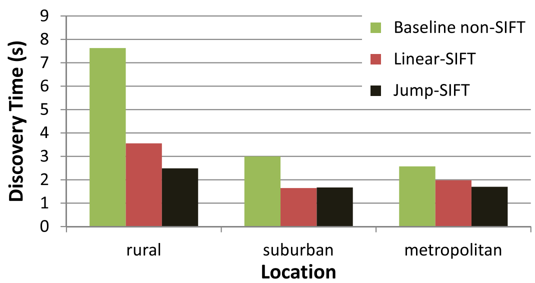

Figure 9: Time to discover one AP at various locations.

quency and width to discover the APs. In this section, we consider two scenarios. First, we show the benefit of our algorithms as a function of contiguous width. Then, we evaluate the benefits in realistic settings, i.e., in metropolitan, suburban and rural settings.

Methodology: We set up two KNOWS devices as before, and configured one as an AP and the other as a client. In the beginning of the experiment, the AP started to beacon on a randomly chosen UHF channel and channel width. We then measured the time for the client to discover the AP using L-SIFT, J-SIFT and the non-SIFT baseline. Depending on the scenario, we artificially specified the spectrum at the AP and the client. The AP did not beacon on any of the occupied channels, and the client did not scan these channels for an AP.

Contiguous Channels: In this experiment, we set the spectrum map to have only one available fragment. We varied the number of UHF channels in the fragment from 1 to 30, since 30 is the total number UHF channels that are available to portable devices. In Figure 8, we plot the total time taken by L-SIFT and J-SIFT to discover the AP as a fraction of the total time taken by the non-SIFT baseline. When there is only one available UHF channel, the time taken by all the algorithms is the same. However, when we increase the width of the available fragment of spectrum, L-SIFT and J-SIFT perform much better than the baseline. As expected, L-SIFT outperforms J-SIFT initially (for narrow white-spaces) since it does not require the "endgame" of trying to find the proper placing of the AP channel. On the other hand, as exactly predicted by our analysis in Section 4.2, J-SIFT becomes more efficient for white spaces spanning more than 10 UHF channels (60 MHz).

Realistic Settings: We also measured the time to discover an AP in metropolitan, suburban and rural areas in the US. We used the methodology described for Figure 2 to obtain the spectrum maps post-DTV transition. We randomly placed the AP on an available channel and width and repeated the experiment 10 times for every locale. As shown in Figure 9, in metro areas, where there are fewer contiguous channels, J-SIFT is 34% faster than the baseline. In rural areas (more contiguous channels), we see that J-SIFT can discover APs in less than one-third the time taken by the baseline algorithm.

5.3 Handling Disconnections

We now quantify the time taken by WhiteFi to reconnect disconnected clients. We setup a client and an AP and started a data transfer between them. Then we switched on a wireless microphone near the client. This causes the client to disconnect, and it starts chirping on the backup channel. In our experimental setup, the AP switched to the backup channel once every 3 seconds, and picks up the chirp in at most 3 seconds. Immediately, the AP uses the spectrum assignment algorithm to determine the best available channel to operate on, and the system is operational again after a lag of at most 4 seconds.

5.4 Spectrum Assignment

We now evaluate WhiteFi's spectrum assignment algorithm. For a detailed understanding of our algorithm, and to evaluate it under varied settings, we decided to use the QualNet simulator [3]. The need to use the simulator arose for two reasons. First, we were constrained by having a limited number of prototype devices, and second, we did not have an FCC license to transmit packets in the TV bands. Therefore, we evaluated our system (spectrum assignment, discovery and disconnection protocols) in a limited setting – on a testbed spanning one floor in our building, and a maximum transmit power of 1 mW.

Modifications to QualNet: We modified QualNet to support variable channel widths by appropriately scaling the OFDM symbol period, and various MAC layer parameters that were described in [15]. We also adjusted the channel noise levels based on the channel width. Furthermore, at every node, we explicitly drop packets that were sent at a different channel width. To ensure that a node appropriately contends with packets that are sent on overlapping channels of different widths, we modified the carrier sensing mechanism in QualNet such that a node spanning multiple UHF channels will transmit a packet only if no carrier is sensed on any of those channels. We also modified QualNet to support fragmented spectrum. Every node reads its initial spectrum map from a configuration file.

5.4.1 Simulation Results

We study the performance of WhiteFi's spectrum assignment algorithm under various settings. First, we microbenchmark the *MCham* metric, and show that it is a good estimate of the expected throughput on a channel. Then, using large scale experiments, we show that WhiteFi performs reasonably well under: (i) varying amounts of background traffic on the channels, (ii) large amounts of spatial variation in spectrum availability, and (iii) when there is a lot of churn in background traffic. In all these experiments, WhiteFi performs nearly as well as an optimal algorithm. In the process, we also show the need for WhiteFi to adapt both the center frequency and the channel width.

Microbenchmark Setup: To verify that *MCham* correctly predicts the channel that will lead to the best throughput, we simulate a spectrum fragment of 5 adjacent UHF channels (26-30), each having one background client/AP-pair. There is one AP with one associated client, transmitting a link-saturating UDP flow. We vary the traffic intensity of the background nodes (from 0 to 50 ms inter-packet delay) and measure the effect on the *MCham* metric and client throughput when transmitting on the 5, 10, and 20 MHz channels centered at channel 28.

Accuracy of the MCham Metric: The results in Figure 10 show that the *MCham* metric accurately predicts which channel achieves the highest throughput for any given background intensity. For example, selecting a 20 MHz channel achieves best throughput until a background traffic intensity of roughly 18 ms inter-packet delay. Similarly, the *MCham* metric predicts that roughly at this level of background traffic, 10 MHz and 20 MHz become equally good, and the narrower 10 MHz channel surpasses the wider channel thereafter. Similarly, at about 24 ms inter-packet delay, 5 MHz starts achieving the highest throughput, which is accurately predicted by the *MCham* metric. We can conclude that the *MCham* metric yields a reasonably accurate prediction of which channel width will result in the highest throughput given a certain level of background traffic.

Setup of large-scale simulations: To better understand the behavior of WhiteFi in large-scale settings, the next three simulations consider the following basic setup. We place one AP in the middle of an area, and randomly distribute clients as well as background

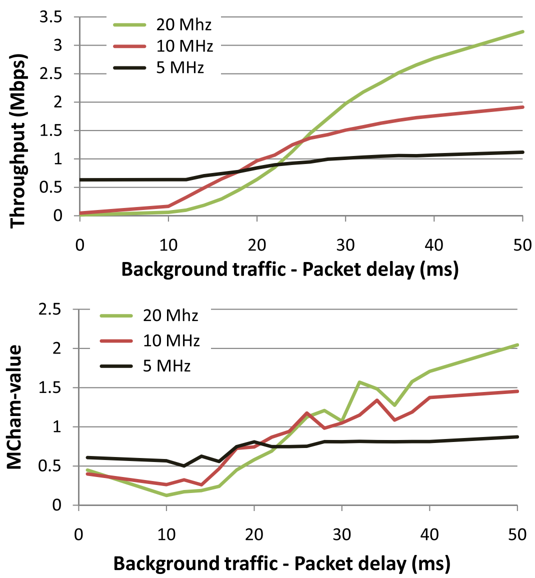

Figure 10: *MCham* value and resulting throughput of a 5, 10, and 20 MHz channel as a function of background traffic intensity. The *MCham* metric accurately predicts which channel achieves highest throughput.

AP/client-pairs within transmission range of this AP (background clients are always deployed within transmission range of their respective background AP). The AP and clients are backlogged and transmit UDP flows (up- and downstream). Background nodes transmit constant-bit-rate (CBR) traffic at a pre-specified intensity. All experiments are repeated 5 times with different random placements of nodes, and results are averaged.

An underlying spectrum map is shared across all clients (except in the experiment in which we focus on the impact of spatial variation). Specifically, the spectrum map is taken from our real measurements in Section 2. There are 17 free UHF channels, and the widest contiguous white space is 36 MHz, i.e., there are multiple possibilities of selecting even 20 MHz wide channels for the AP.

In all experiments, we measure the per-client throughput of clients/APs. We consider the following baseline algorithms for comparison with WhiteFi. OPT 5 MHz denotes the throughput achieved when statically picking the best (across all non-incumbent) UHF channels. Similarly, OPT 10 MHz and OPT 20 MHz are the algorithms that statically pick the best possible 10 and 20 MHz channel, respectively. Finally, OPT is an ideal, omniscient algorithm that for every experiment run picks the channel with maximum throughput. The goal of the WhiteFi spectrum assignment algorithm is to approach OPT as closely as possible.

Impact of Background Traffic: Figure 11 shows how WhiteFi reacts to varying degrees of background traffic. Specifically, there are X background AP/client-pairs in the system, each being randomly assigned to one of the free UHF channels, and each sending at a packet interval delay of 30 ms.

The figure shows that WhiteFi achieves close to optimal performance for varying degree of background traffic. With little or no background traffic, WhiteFi performs as well as picking the widest available channel (OPT 20 MHz), which is optimal. As the traffic increases, the throughput achieved by OPT 20 MHz drops, and OPT 10 MHz becomes better (at about 10 background AP/client-pairs). Even at this point WhiteFi performs near-optimally, which shows that WhiteFi adaptively switches to narrower channels as needed. In fact, our evaluation shows that WhiteFi is always within 14% of the optimal value throughput OPT.

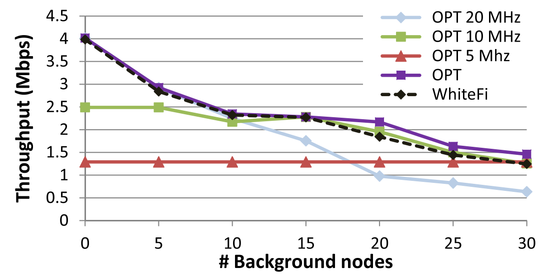

Figure 11: Impact of background traffic on throughput.

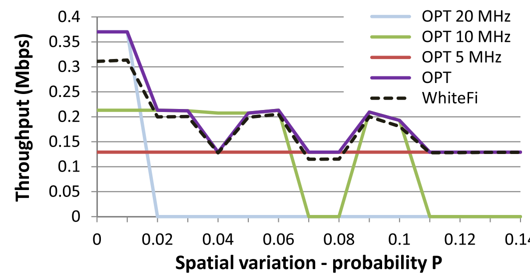

Figure 12: Impact of spatial variation on throughput.

An important observation is that due to fragmentation and background traffic, there is no single best center frequency and channel width that should be used in UHF white spaces. WhiteFi is capable of adjusting to the appropriate width and selects a near-optimal channel.

Impact of Spatial Variation: Figure 12 shows the impact of spatial variation on per-client throughput. In this experiment, there are 10 clients connected the AP, and one background client/AP-pair per UHF channel, transmitting at CBR with 30 ms inter-packet delay. Spatial variation is modeled as follows. Each client and the AP start with a common spectrum map. Then, for each client (and AP) and for each UHF channel i, we randomly flip the entry u_i with probability P. In the experiment, we vary P from 0 (no spatial variation) to 0.14 (large spatial variation).

It can be seen in the figure, spatial variation reduces achievable aggregate throughput. Because the AP needs to select a channel that is free at *all* clients, no contiguous free spectrum parts remain available for $P > 0.1$, and hence, the aggregate throughput reduces to the throughput of a single UHF channel (5 MHz). For low spatial variation, the throughput is much higher when selecting a 20 MHz wide (e.g. at $P = 0.01$) or a 10 MHz channel (e.g. at $P = 0.05$). Generally, the figure highlights the need for *adaptive* channel width in UHF white spaces: no single channel width (OPT 20 MHz, OPT 10 MHz, OPT 5 MHz) achieves close-to-optimal throughput in all cases. On the other hand, WhiteFi is near-optimal in all cases.

Impact of Churn: Finally, we want to understand the impact of churn (in terms of background traffic) on the throughput achieved by WhiteFi and the various baseline algorithms. There are a total of 34 background AP/client-pairs, two per free UHF channel. In order to model churn, we model background nodes using a simple discrete Markov chain with two states (A=active, P=passive). A background node in the active state transmits CBR traffic with 60 ms inter-packet delay. A node in the passive state does not transmit. We simulate this setting for various state transition probabilities, selecting them to cover the entire range of (1) likelihood of being in either state and (2) average state duration (see x-axis in Figure 13). The extreme cases are (i) all nodes are always in state P, (ii) nodes are in each state with equal likelihood and they remain in their current state for an average of 30 seconds, and (iii) all nodes are always in state A.

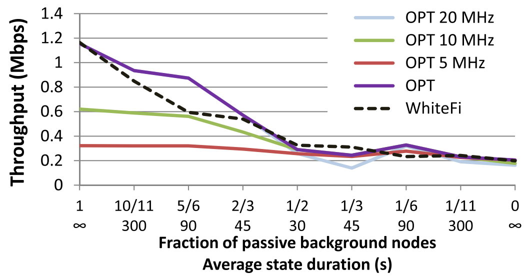

Figure 13: Impact of churn on throughput.

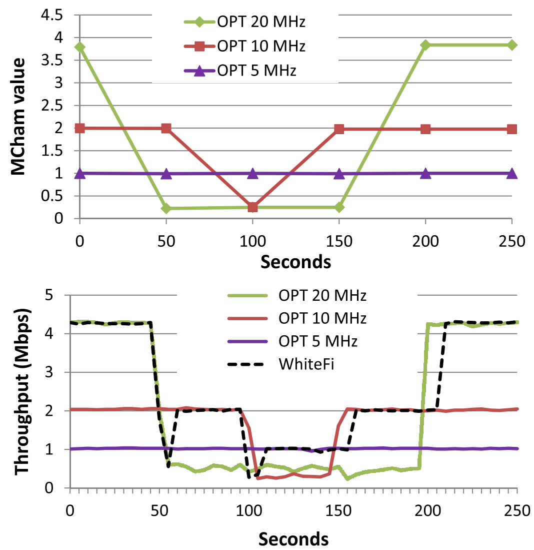

Figure 14: Experimental validation of WhiteFi's spectrum assignment algorithm on a testbed with variable background traffic. Top figure shows the MCham metric for each of the three channel widths. Bottom figure shows the throughput (averaged over 5 sec windows) for WhiteFi and OPT.

Figure 13 shows that WhiteFi performs near-optimally for varying degree of churn. For low churn and little background traffic, WhiteFi selects the widest channel. For high churn (e.g., state duration 45 seconds and passive probability $1/3$), always picking the widest channel (OPT 20 MHz) becomes the worst performing algorithm. Instead, WhiteFi is better than any static channel width choice. In fact, WhiteFi even outperforms OPT. In this experiment, this is possible because OPT is the optimal *static* channel selection throughout the entire execution of the simulation. Instead, WhiteFi is adaptive and can adjust to the current values of background traffic, changing its channel accordingly.

5.4.2 Results from our Prototype

To demonstrate the adaptability of WhiteFi's spectrum assignment algorithm, we set up an experiment with an AP and a client in our building, which is Building 5 in Figure 1. The spectrum map of our building has the following free UHF channels: 26 to 30, 33 to 35, 39 and 48. Therefore, we have fragments of size 20 MHz, 10 MHz and two channels of 5 MHz to form a network.

Every client and AP using WhiteFi spends 1 second on every UHF channel to determine the airtime utilization using SIFT, as described in Section 5.1. All nodes feed their airtime to the AP, which computes the *MCham* metric and decides on the channel to use for the network. We present the throughput of our system with time, and the corresponding *MCham* value on the different spectrum chunks in Figure 14.

Initially, when there is no background traffic, the AP and client operate on the 20 MHz spectrum chunk between channels 26 and 30. Then at time 50 seconds, we introduce background traffic on channels 26 through 29. Correspondingly, the value of the *MCham* metric for the 20 MHz fragment drops sharply, and the AP and its clients move to the 10 MHz spectrum fragment. As shown in the figure, this is also the fragment that has the best throughput. Then at time 100 seconds, we introduce background traffic on channels 33 and 34, and as before the value of the 10 MHz channel's *MCham* metric drops, and the system switches to channel 39 (any 5 MHz chunk could have been chosen). Then at times 150 and 200 seconds, we remove the background interference from channels 33 and 34, and from channels 26 through 29, respectively. Correspondingly, WhiteFi switches to the fragment with the best *MCham* value, i.e. to the 10 MHz fragment at 150 seconds, and to the 20 MHz fragment at 200 seconds. We conclude from the above experiments that WhiteFi adaptively operates on the best part of the spectrum.

6. DISCUSSION AND FUTURE WORK

White space networking provides a unique opportunity for clean-slate network design, owing to the lack of existing standards. Our decision to build the WhiteFi prototype with a Wi-Fi card was motivated by several factors. Wi-Fi is a mature, well-understood technology that is inexpensive and easily available. Several wireless card vendors we have spoken with are considering pushing some version of Wi-Fi to the IEEE standards body for white space networking. Additionally, Wi-Fi enabled us to build a prototype quickly and focus on some of the higher layer issues that are somewhat agnostic to the existing physical and MAC protocols. However, we do realize that alternative designs are possible and might be used in future networks. We discuss a few of these below.

WhiteFi leverages the technique described in [15], which requires the AP and its clients to operate over the same contiguous chunk of spectrum. An alternative technique might use a PHY layer that operates over non-contiguous spectrum chunks. The AP can then operate over the entire bandwidth, decoding signals from the different clients who may be using different OFDM subcarriers. For AP-to-client communications, the PHY layer could either suppress or send a null signal on the subcarrier that the primary user is using [21]. In theory this is a reasonable idea but it poses two practical problems. First, leakage from adjacent subcarriers causes interference to the primary user. To avoid this interference, we would require a highly accurate bandpass filter of appropriate bandwidth but to the best of our knowledge researchers are still working on developing such sharp bandpass filters. Second, and more importantly, sending data over different subcarriers to an AP is difficult to implement for uplink traffic. We are not aware of any system that can decode packets sent simultaneously from multiple clients over non-overlapping subcarriers. This is an active research area and we are investigating the practicality of such a system.

Another issue is our choice of CSMA/CA, the medium access control (MAC) protocol for Wi-Fi, in WhiteFi. The research literature has several interesting proposals for MAC protocols, which can be broadly categorized under Listen Before Transmit (LBT) and Time Division Multiple Access (TDMA). Observing what is happening in the ISM bands we made the decision that WhiteFi must be able to co-exist with other unlicensed devices. The success of LBT protocols (e.g., Wi-Fi) in the ISM bands made it a natural choice for white space networking. We also believe that an alternative TDMA like MAC (e.g., Bluetooth) will not perform well in white spaces without significant modifications. Local interference from wireless microphones around the client or the AP would impact slot scheduling and lead to poor performance. Furthermore, in UHF white spaces the clients and AP may be over a mile away, further aggravating the scheduling problem. Additional research is needed to understand these issues and is out of scope for this paper. Our initial results show that CSMA/CA is a reasonable choice for white space networking.

Prior work has proposed the use of control channels to reserve bandwidth and spectrum [12, 24]. While there are advantages to a control channel design, we believe that control channels can be compromised, thus bringing down the network. Also, control channel based solutions are prone to the range-mismatch problem [24]. We overcome these problems by not using a dedicated control channel. WhiteFi uses a backup channel in the white spaces (instead of 900 MHz spectrum as proposed by CMAC [24]) thereby avoiding the range mismatch problem. Also, WhiteFi does not use a static control channel. It dynamically adapts the backup channel to operate on spectrum that is not occupied by a primary user.

7. RELATED WORK

Prior work has mostly focussed on the problem of opportunistically forming a single link over UHF white spaces [8, 12]. This involves accurate sensing of the spectrum [14,17,18], reliable identification of incumbents, and radio agility on detecting a primary. However, to the best of our knowledge, no prior work has studied the problems of forming a Wi-Fi like network over white spaces.

WhiteFi builds upon our prior work on KNOWS [24], which uses a similar hardware platform and proposes a control channel based MAC protocol for ad hoc networks over white spaces. WhiteFi looks at the problem of forming an AP based network while reusing the Wi-Fi MAC and without using a control channel.

A complementary effort to WhiteFi is the IEEE 802.22 [1] working group's proposal for WRANs (Wireless Regional Area Networks) over UHF white spaces. It is intended to provide wireless broadband access to rural areas and neighborhoods. In contrast, WhiteFi considers a usage model similar to Wi-Fi, with one AP providing coverage to several possibly mobile users. Despite the difference in the scenarios, the techniques developed by WhiteFi, for disconnection, discovery and spectrum assignment, are also applicable in WRANs. For example, the 802.22 draft includes support for variable widths, although it does not specify how to use it.

A recent technology that enables unlicensed devices to co-exist with licensed users is SWIFT [21]. SWIFT pokes the primary user to learn about its presence. Unfortunately, this technology cannot be used over white spaces because the FCC does not allow "testing" the presence of an incumbent by "poking" at it with a transmission. Also, the incumbents of UHF white spaces do not back off.

8. CONCLUSIONS

In this paper, we have presented the design and implementation of WhiteFi, the first white space Wi-Fi like wireless network. We moved beyond the current state-of-art that considers a single link to building a real network with multiple links. In building WhiteFi we identified and described several unique challenges in operating a white space network and showed with extensive experiments how white space networks differ from ISM band Wi-Fi netwoks. WhiteFi contributes a new spectrum assignment algorithm that solves the dual challenges of spatial variation of available spectrum and spectrum fragmentation. We further described a new mechanism that quickly discovers APs operating anywhere in the 180 MHz white space, using any arbitrary channel width. We also described a new technique for handling disconnections where clients signal to the AP without interfering with ongoing wireless microphone transmissions. Underlying our solutions is a new application of a signal recognition technique called SIFT, which quickly analyzes packets in the time domain, allowing fast AP discovery and managing disconnections due to temporal variations. We demonstrated WhiteFi in the context of our custom built prototype UHF hardware and QualNet simulations. As part of ongoing work, we are deploying WhiteFi over a campus wide white space network.

Acknowledgements

We would like to thank our shepherd, Brad Karp, for his help with the final version of the paper. His detailed comments were immensely helpful in improving the presentation of the paper. We are also grateful to the anonymous SIGCOMM reviewers for their insightful comments.

9. REFERENCES

[1] IEEE 802.22 Working Group on WRANs, http://www.ieee802.org/22/.

[2] In private conversations with Adaptrum.

[3] Qualnet 4.5 simulator, http://www.scalable-networks.com.

[4] TV Fool, http://www.tvfool.com/.

[5] Universal Software Radio Peripheral, http://www.ettus.com/.

[6] USRP FAQ, http://www.gnuradio.org/trac/wiki/UsrpFAQ/Gen.

[7] FCC press release, FCC Acts to Expedite DTV Transition and Clarify DTV Build-Out Rules. November 2001.

[8] Demonstration of a Prototype Dynamic Spectrum Access System, Philips Research. In *DySPAN demo session*, 2008.

[9] FCC press release, Evaluation of the Performance of Prototype TV-Band White Space Devices. November 2008.

[10] FCC press release, FCC Adopts Rules for Unlicensed Use of Television White Spaces. November 2008.

[11] N. Ahmed, S. Banerjee, S. Keshav, A. Mishra, K. Papagiannaki, and V. Shrivastava. Interference Mitigation in Wireless LANs using Speculative Scheduling . In *MobiCom*, 2007.

[12] R. E. D. Borth and B. Oberlie. Considerations for Successful Cognitive Radio Systems in US TV White Space. In *DySpan*, 2008.

[13] G. Buchwald, S. Kuffner, M. Brown, and E. C. L. Ecklund. The Design and Operation of the IEEE 802.22.1 Disabling Beacon for the Protection of TV Whitespace Incumbents. In *DySpan*, 2008.

[14] D. Cabric, A. Tkachenko, and R. W. Bordersen. Experimental Study of Spectrum Sensing based on Energy Detection and Network Cooperation. In *ACM 1st International Workshop on Technology and Policyy for Accessing Spectrum (TAPAS)*, 2006.

[15] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl. A Case for Adapting Channel Width in Wireless Networks. In *SIGCOMM*, 2008.

[16] S. Inc. SLX series product manual.

[17] H. Kim and K. G. Shin. Fast Discovery of Spectrum Opportunities in Cognitive Radio Networks. In *DySPAN*, 2008.

[18] H. Kim and K. G. Shin. In-band Spectrum Sensing in Cognitive Radio Networks: Energy Detection or Feature Detection? In *MOBICOM*, 2008.

[19] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill. Designing High-Performance Enterprise Wireless Networks. In *NSDI*, San Francisco, CA, April 2008.

[20] S. Narlanka, R. Chandra, P. Bahl, and I. Ferrell. A Hardware Platform for Utilizing the TV Bands with a Wi-Fi Radio. In *IEEE LANMAN*, June 2007.

[21] H. Rahul, N. Kushman, D. Katabi, C. Sodini, and F. Edalat. Learning to Share: Narrowband-Friendly Wideband Wireless Networks. In *SIGCOMM*, 2008.

[22] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra. Perceptual Evaluation of Speech Quality (PESQ)-A New Method for Speech Quality Assessment of Telephone Networks and Codecs. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001.

[23] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. M. Volker. SORA: High Performance Software Radio Using General Purpose Multi-core Processors. In *NSDI*, 2009.

[24] Y. Yuan, P. Bahl, R. Chandra, P. A. Chou, I. Ferrell, T. Moscibroda, S. Narlanka, and Y. Wu. KNOWS: Kognitiv Networking Over White Spaces. In *DySPAN*, 2007.