Bridging Link Power Asymmetry in Mobile Whitespace Networks

Sanjib Sur and Xinyu Zhang
University of Wisconsin-Madison
Email: sur2@wisc.edu and xyzhang@ece.wisc.edu

***Abstract*—We explore the use of *TV White Space* (TVWS) wireless networks for providing robust and long range connectivity to vehicles. A key distinctive requirement of TVWS networks is the power asymmetry – the static APs are allowed to transmit at up to 4 W, while the mobile clients in vehicles are limited to only 100 mW. Our measurements reveal that the power asymmetry not only causes severe uplink blackouts but also poses significant coexistence problems, as high-power fixed nodes can easily starve the low-power mobile ones due to carrier sensing loss. To tackle these unique challenges, we propose a cross-layer design of a Direct-Sequence Spread Spectrum (DSSS) based system. We employ an adaptive DSSS mechanism that strategically configures the spreading code, so as to boost uplink coverage while maximizing throughput. We further design a traffic-aware code assignment algorithm for uplink packets to balance the requirement of throughput-intensive and latency-sensitive flows. We have implemented the design on a TVWS software-radio platform on a moving vehicle in an urban environment, and demonstrated that link asymmetry can be completely removed to support realistic application traffic, while the carrier sense loss rate at fixed nodes can be reduced by around 85%.**

I. Introduction

High bandwidth, robust Internet connectivity and long range are key requirements for vehicular networks to enable diverse set of applications, *e.g.*, improved traffic intelligence, transportation safety, infotainment and location-aware services. Many research systems such as MAR [1], WiRover [2], ViFi [3] and CaberNet [4], have strived to fulfill the vision of vehicular networking, using existing cellular technologies (3G/4G), and sometimes augmented by opportunistic WiFi access. However, cellular networks are costly and usually delay-prone [5]. On the other hand, WiFi has limited coverage.

The TV White Space (TVWS) spectrum on the UHF band (470 – 698 MHz), recently released for unlicensed usage in the U.S. [6], offers a lucrative wireless communication medium. The low cost and good propagation characteristics of UHF make the whitespaces attractive for vehicular networking.

However, when operating in vehicular scenarios, TVWS networks face a unique challenge: the huge transmit power asymmetry between fixed and mobile nodes. According to the FCC rules [7], any static device (*e.g.*, an AP) can transmit with maximum EIRP of 4 W, whereas, mobile devices (*e.g.* handsets or gateways on vehicles) are constrained to only 100 mW. The conservative limit for mobile devices aims to prevent harmful interference to the primary incumbents during roaming. However, the resulting $40\times$ power asymmetry severely amortizes benefits of UHF band and poses two obstacles for vehicular networks over TVWS. *First*, the power discrepancy translates into around $4\times$ of downlink/uplink range mismatch in vehicle-to-infrastructure scenarios [5]. Thus, uplink becomes the connectivity bottleneck. Uplink connectivity may be enhanced by a dense AP deployment, but the APs have to waste their downlink coverage advantage, and the infrastructure cost may become formidable. *Second*, certain static non-AP high-power transmitters may starve the low-power mobile devices. Unlicensed TVWS networks (*e.g.* IEEE 802.11af [8]) typically use CSMA/CA for channel contention. Although a low-power device may hear a high-power one's transmission and back off for it, the reverse does not always hold, which leads to the latter arbitrarily interfere ongoing transmissions from the former.

The transmit power asymmetry problem itself already exists in cellular networks, yet, the legacy solution does not readily apply to TVWS networks. WCDMA enforces centralized policy for power equalization to ensure signals from low-power transmitters are not drown [9], which can be hardly imposed on unlicensed and unmanaged TVWS devices. LTE basestations have 20 dB higher transmit power than mobile clients [9]. The resulting downlink/uplink gap is filled by making the basestation RF front-end $100\times$ more sensitive, through large form-factor antennas and high-end low-noise amplifiers. Such solutions incur huge infrastructure cost and are inappropriate for consumer-grade TVWS deployment. Even if the TVWS APs can sustain the cost, static high-power consumer devices may not, and will remain a threat to starve low-power nodes.

In this paper, we tackle the power asymmetry problem in TVWS through a cross-layer design, referred to as *adaptive DSSS code modulation*. A DSSS transmitter spreads a data symbol's energy over a sequence of N samples, called a code. The receiver aggregates the energy through matched filtering, which can theoretically improve the link SNR by N times, *i.e.*, achieving a $10\log_{10}(N)$ dB processing gain [10]. Ideally, with a sequence length $N = 40$, a mobile TVWS device can bridge the $40\times$ power gap between static nodes.

Our empirical investigation reveals unique challenges in realizing this vision in practical TVWS networks, which are addressed through two core components.

(i) Although a longer DSSS code provides higher SNR improvement, it costs more channel time. Thus, our adaptive DSSS protocol judiciously chooses the code length that ensures coverage, while minimizing throughput loss. Balancing this tradeoff requires knowledge of the processing gain of codes, which is shown to be environment dependent in our experiments. We thus design a set of run-time estimation algorithms leveraging the inherent structure of DSSS modulated packets.

(ii) Choice of code length may garner high throughput for one traffic flow, but adversely affect delay-sensitive flows. We strike a fine balance through a traffic-aware code length assignment algorithm when different traffic patterns coexist. The problem is formulated as a utility optimization problem, which we found to have optimal substructure and can be solved through a pseudo-polynomial algorithm.

To validate the mechanisms, we have prototyped adaptive DSSS on a TVWS software-radio platform that operates on a spectrum with FCC-granted experimental license. Our implementation extends the 802.11b PHY layer which uses a fixed DSSS code length. Our experiments in an outdoor vehicular environment demonstrate that, the adaptive DSSS protocol can maintain uplink connectivity whenever the downlink can be

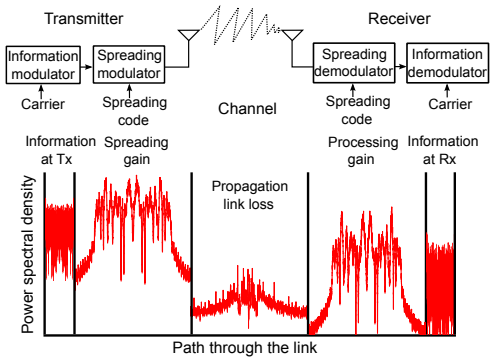

Fig. 1. Transmitter and receiver structure for DSSS communication and the evolution of power spectral density of the DSSS signal from the transmitter to the receiver.

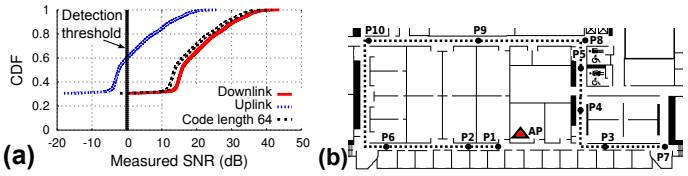

Fig. 2. (a) White space range asymmetry in Outdoor. (b) Experimental setup for indoor whitespace.

reached, albeit at the cost of reduced uplink throughput. In contrast, an OFDM uplink, even with oracle rate adaptation, can maintain the connectivity for only 43% of the time. Adaptive DSSS reduces the carrier sensing failure rate at high-power static nodes by around 85%, thus preventing them from interfering low-power mobile nodes. For delay-sensitive applications our scheme performs $2.2\times$ better than the OFDM case in terms of deadline misses, while performing equally well in terms of throughput for throughput-intensive applications.

DSSS modulation has been adopted in the early generation of WiFi (IEEE 802.11b). Our vision is that, the 802.11b's mature baseband silicon implementation can be reused and revived to complement emerging TVWS networks (*e.g.*, 802.11af) when they encounter power asymmetry. By trading channel time usage for coverage, DSSS inevitably leads to lower uplink throughput than downlink. But, our experiments demonstrate that many meaningful vehicular network applications can still be supported, given the downlink-dominated Internet traffic. In addition, once the uplink connection bottleneck is eliminated, its bit-rate can be boosted by other means, *e.g.*, opportunistically aggregating spectrum resources.

II. Background, Motivation and Feasibility Study

A. DSSS Communication Primer

The basic operation of a DSSS communication system is shown in Fig. 1. At the transmitter, each data symbol is spread by multiplying with a high-rate random sequence called *spreading code*. The nature of the high-rate spreading signal causes the power spectral density (PSD) to spread over a wider frequency range than the original data signal. The output signal, when transmitted over-the-air, experiences propagation loss, multi-path distortions, and noises.

At the receiver, a matched filter *de-spreads* the received signal by correlating it with the same spreading sequence employed by the transmitter. Such correlation boosts the power spectrum of useful information, whereas, the noise spectrum level remains the same, thus achieving an extra *processing gain*, as shown in the evolution of PSD in Fig. 1. A spreading code of length N theoretically achieves a processing gain of N times, equivalently boosting received SNR by $10\log_{10}(N)$ dB [11].

B. Power asymmetry causes uplink blackout

To understand the practical impacts of the power asymmetry issue, we measure the uplink/downlink coverage of a TVWS network in an outdoor environment (Fig. 3). The experiments run on our whitespace software-radio testbed, which implements both OFDM and DSSS modulation (Sec. V-A).

A mobile client is moved to 8 different locations denoted by, $\mathrm{P1}, \mathrm{P2}, \ldots, \mathrm{P8}$. The AP is statically placed in a 4^{th} floor room of a nearby office building with its antenna facing towards the outdoor clients. We first run OFDM BPSK (modulation scheme in 802.11af) for both types of nodes. The client transmits at 100 mW and the AP at 1 W due to hardware limitation. As FCC's power limit for static device is 4 W, we extrapolated by adding 6 dB ($\sim 10\log_{10}(4)$) to the SNR of downlink.

Fig. 2(a) shows the CDF of the detected packets. For downlink, due to occasional tall building blockages, around 25% packets are not detected and all detected packets can be decoded successfully. For uplink, more than 60% uplink packets are not even detected by the AP, with only 37% of detected packets successfully decoded. In contrast, with DSSS modulation of code length 64, the uplink performance almost matches with downlink, *i.e.*, uplink is no longer the bottleneck of network coverage. Therefore, *with traditional OFDM, power asymmetry causes severe uplink connectivity blackouts in TVWS networks, which can be potentially prevented using DSSS.*

C. Power asymmetry causes starvation of mobile clients

As discussed in Sec. I, the FCC rule [7] may cause starvation of low-power mobile clients when they coexist with non-AP high-power fixed clients. To understand the problem in detail, we place three high-power fixed clients in the outdoor scenario denoted by, $\mathrm{HP1}, \mathrm{HP2}$ and $\mathrm{HP3}$ (Fig. 3). We measure the fraction of mobile clients' packets that are not sensed by such high-power nodes. Similar to the configuration in [12], a packet is not sensible if its RSS is at the noise-floor level (0 dB SNR).

Fig. 4(a) shows the result. Take the client location $\mathrm{P4}$ as an example. The $\mathrm{HP1}$, $\mathrm{HP2}$ and $\mathrm{HP3}$ fail to sense 92%, 3% and 61% of P4's packets respectively. For client location $\mathrm{P6}$, $\mathrm{HP1}$ and $\mathrm{HP2}$ can sense all packets, while $\mathrm{HP3}$ fails on 100%.

Fig. 4(b) shows the result in more detail when the mobile client transmits from location $\mathrm{P5}$. The CDF plot shows the distribution of received SNR at the fixed clients. Note that $\mathrm{HP1}$ fails to detect 60% packets, while $\mathrm{HP2}$ and $\mathrm{HP3}$ fail on 3% and 98% packets respectively. Since all 3 fixed nodes are close to the AP, *carrier sensing failure will cause severe collisions for uplink packets from the mobile client, thus starving its transmission*. In contrast, when using DSSS code length 64, the AP and all fixed clients (solid CDF curve) are able to detect all uplink packets from $\mathrm{P5}$.

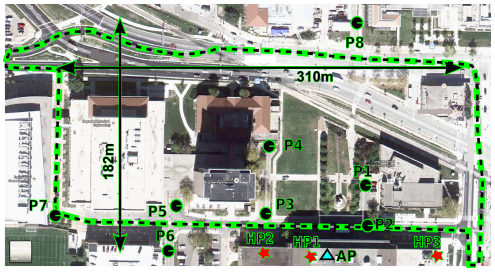

Fig. 3. Experimental setup for outdoor whitespace with 3 high power static clients. Distances are shown. Note building and tree locations.

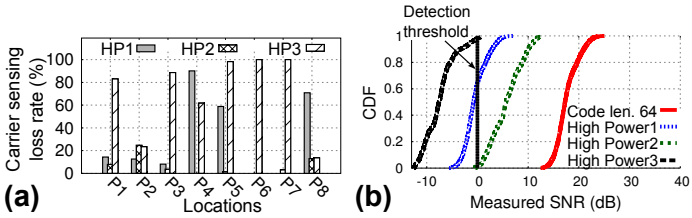

Fig. 4. (a) Carrier sensing loss rate at three high power clients from all 8 experimental locations. (b) Distribution of SNR at high power fixed clients from location P5.

III. The Need for Adapting Code Length

A. Balancing coverage and throughput

The above field experiments indicate that, the two consequences of power asymmetry, *i.e.*, uplink/downlink range mismatch and low-power mobile node starvation, can be almost avoided using a long DSSS code sequence (*e.g.* length 64). However, a longer DSSS sequence length costs longer channel time for each data symbol inside a packet, and therefore, the achievable throughput is proportionally reduced.

To understand this tradeoff in real settings, we use our testbed to measure the throughput when the packet is DSSS-modulated with code length 64 and 11, respectively. We ran the experiments in two different settings, *(i)* Indoor walking, at an average speed of 1.5 m/s and following the path in Fig. 2(b) (start: P1). *(ii)* Outdoor driving, with speed between $15-35$ mph and following the path in Fig. 3 (start: P7).

Fig. 5 shows the uplink throughput variation over time. For indoor case, near the regions of P1 and P2, the code length 11 provides higher throughput owing to less channel time cost. However, in regions with weak uplink signals due to long distance and wall blockages (P6, P4, P5), its throughput is lower than that of code length 64. In certain other regions (P10, P9, P8), its throughput plummets to zero, but, the 64-bit DSSS code link can sustain the connection with throughput between $25-30$ kbps. A similar observation was made in outdoor case shown in Fig. 5(b). In summary, adapting code length is of utmost importance as, *different choice of code length can result in higher performance, depending on the channel condition.*

B. Traffic-aware code length adaptation

A strawman approach to balance between the uplink coverage and throughput is to perform just per-packet adaptation and choose the code length that maximizes the throughput. Even if this can guarantee the throughput requirement, it is not enough. We envision future TVWS networks will support a diverse range of applications, including not only throughput-intensive (*e.g.*, file downloading and content-rich web browsing), but

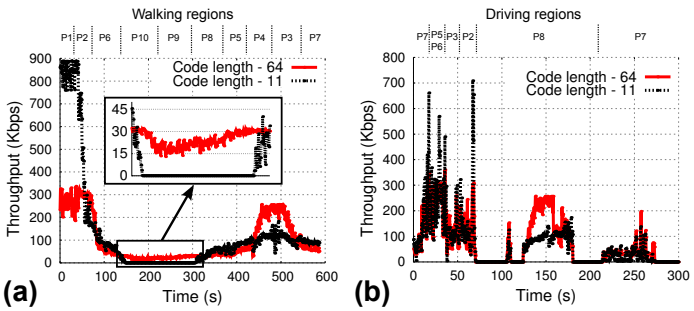

Fig. 5. Uplink throughput variation of two code lengths: (a) Indoor walking (blackout region for length 11 is zoomed in). (b) Outdoor driving.

also delay-sensitive (*e.g.*, safety warning, GPS location update) traffics. While solving the power asymmetry problem, adaptive DSSS faces a new challenge under such blended traffic patterns.

In particular, a long spreading code provisions high uplink reliability, but may adversely affect delay-sensitive packets. For instance, a 64-bit code extends a nominal packet length by 64 times, which may cause urgent packets to miss their deadlines. Certain loss-tolerant packets, such as video/audio streaming, may prefer less reliable short code in order to meet their own requirements, while saving time for others.

Therefore, when adapting code length for uplink packets' transmission, *we need to account for not only the throughput, but also short-term packet latency requirement*. Our adaptive DSSS design offers one viable approach towards this end.

IV. Adaptive DSSS Design

A. Design overview

Built on our previous feasibility and motivational studies, the adaptive DSSS protocol is designed to provide robust uplink connectivity between moving vehicles and a TVWS infrastructure node (*i.e.*, the AP). It is equally applicable to indoor TVWS networks for handheld mobile devices that suffer from the power asymmetry problem. Our design runs in conjunction with the OFDM-modulated 802.11af TVWS network standard. By default, the OFDM mode is used for both the downlink and uplink. The AP initiates the adaptive DSSS protocol for a mobile client whenever it senses the uplink SNR goes below a threshold, chosen to be the minimum SNR needed to support the lowest OFDM bit-rate.

The protocol adapts packet-level DSSS code assignment on the basis of intervals, each containing multiple packet transmissions. Intuitively, the adaptation interval is the look-ahead time within which channel is relatively stable, and thus we can schedule the code assignment for all candidate packets within the interval (Sec. IV-F). In the beginning of each interval, the client sends a *probe packet* containing a known preamble and information about candidate uplink packets (size and relative priority) within the interval. The information is modulated via the longest code (Sec. IV-B), to guarantee uplink is reachable.

Upon receiving the probe packet, the AP leverages the known preamble to estimate the processing gain of all DSSS code sequences (Sec. IV-C). Then, it runs a code adaptation algorithm to derive an *optimal configuration* across two dimensions — candidate packets in current probing interval and DSSS code sequences to be assigned to them — to maximize the

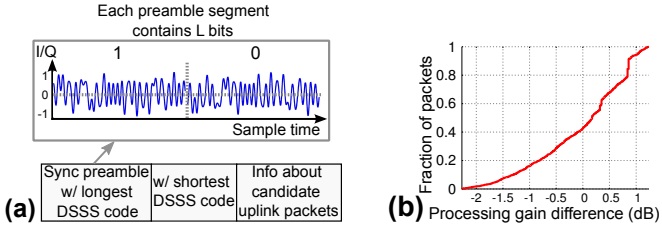

Fig. 6. (a) Adaptive DSSS probe packet structure. (b) CDF of processing gain differences of 10 pseudo-random code *w.r.t.* original Barker code.

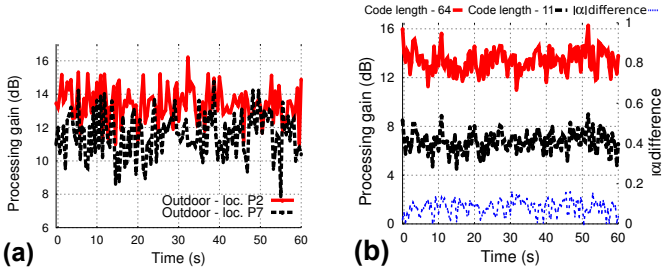

Fig. 7. Processing gain variation: (a) Same DSSS code (length 64) in two different outdoor locations. (b) Two different DSSS codes in indoor LOS.

total system utility. It executes either per-packet (Sec. IV-D) or multi-packet (Sec. IV-E) assignment, depending on the probing interval length (Sec. IV-F). The AP informs the client of the code assignment through a probe response packet.

The client's subsequent uplink packets employ the assigned codes. In case, it observes severe uplink packet losses, it infers there may be a sudden channel degradation that invalidates the optimal configuration. Thus, it terminates the current interval early by sending a new probe packet.

Below we detail the essential components in the protocol.

B. DSSS code sequences design

Theoretically, a DSSS code length of $N = 40$ can boost the uplink SNR by $40\times$, thus bridging the power asymmetry (Sec. II-A). However, the practical processing gain is lower under multipath reflections and Doppler effect, which smear symbol boundaries and reduces the despreaded signal strength. Our system chooses 64-bit as the longest code length, which has empirically proven to be able to bridge the uplink/downlink SNR gap in real environment (Sec. II-B).

A key requirement for the DSSS code sequence lies in a strong autocorrelation property, *i.e.*, a high gain of the auto-correlation peak over the sidelobes. The Barker code is a set of random sequences that satisfy ideal autocorrelation with a N-times peak gain. However, the longest known Barker code only has a sequence length of 13-bit and the 802.11b uses 11-bit code. Our system thus uses alternative pseudo-random sequences generated using *Binary Galois Field LFSR* [13], which can have arbitrary length but lower peak gain. We choose a set of 2^{th}-power sequence lengths, from 2 up to 64.

To understand the impact of the imperfect peak gain in real channel, we randomly pick 10 such sequences of length 11-bit and compare the processing gain differences *w.r.t.* to the 802.11b Barker code. The experiments run on our testbed with DSSS modulation (Sec. V-A). Fig. 6(b) shows that the 85^{th} percentile processing gain difference lies within $\pm$ 1 dB, with worse gain difference being -2.3 dB. Such difference is tolerable and can be compensated when we use a long sequence.

C. Processing gain estimation

To estimate the processing gain of different code lengths, the AP only needs to inspect the client's probe packet that contains preamble known as the *probing preamble.* The preamble comprises two small segments of symbols modulated using the longest and shortest spreading code, respectively (Fig. 6(a)). Simply put, the AP first estimates the SNR of these two code sequences based on the two segments it received. Then it predicts the processing gain of all code sequences based on the SNR estimation results. This saves significant overhead compared with a naive preamble containing all possible sequences. Below we detail these two steps separately.

1) SNR and BER estimation: Suppose the known preamble segment contains L symbols. Let P_m denote the corresponding received raw samples, which have been modulated using DSSS code c_m with code length n_m. The AP performs matched filtering of the samples with the same DSSS code and the output is given by, $MF_m = P_m * c_m$, where $*$ is the convolution operator. It runs a slicer over the matched filter output, sampling at peak values to retrieve the preamble symbols. Corresponding L peak values can be represented as,

$$\begin{aligned} Z_{m,k} &= \max_{q_k \le j < q_k + n_m} |MF_{m,j}|, \qquad \forall\ 1 \le k \le L \\ q_1 &= 1, q_{k+1} = \operatorname*{argmax}_j |MF_{m,j}| + 1, \quad \forall\ 1 \le k < L \end{aligned} \tag{1}$$

The combined signal+noise energy can be calculated by using the expectation of the peak values,

$$S_m + N_m = \frac{1}{2}\Big[\frac{1}{L}\sum_{k=1}^{L} Z_{m,k}\Big]^2 \tag{2}$$

while the noise floor is approximated as the variance:

$$N_m = \frac{1}{L-1}\sum_{k=1}^{L} Z_{m,k}^2 - \frac{1}{L(L-1)}\Big[\sum_{k=1}^{L} Z_{m,k}\Big]^2 \tag{3}$$

Then, the post-processed SNR of the segment modulated using DSSS code c_m can be calculated as, $SNR_m = S_m/N_m$.

Given the SNR of decoded symbols, the BER is simply the probability that one Gaussian random variable (corresponding to one noisy symbol) smears into the other's "region", which can be modeled by the standard Q-function:

$$\varepsilon^m = Q(\sqrt{SNR_m}) \tag{4}$$

2) Processing gain prediction: To understand how practical processing gain deviates from theory, we measure its variation over different channel conditions. Fig. 7(a) shows that for different outdoor locations, the same DSSS code length 64 achieves different processing gains. For example, location P2 (in Fig. 3) has an average processing gain of 14 dB with std. 0.9 dB over 60 seconds, whereas location P7 experiences an average of 11.9 dB with std. 1.3 dB. Fig. 7(b) shows the processing gain of two different DSSS code sequences carried by the same packets. Interestingly, the channel variation affects both of them almost following a consistent trend.

We leverage this observations to predict the processing gain. Denote the longest and shortest code sequence length as $N_{\max}$ and $N_{\min}$, respectively. The theoretical processing gain of these two sequences differs by, $G_{\max} - G_{\min} = 10\log_{10}(N_{\max}) - 10\log_{10}(N_{\min})$ dB. Using the probing preamble, the AP estimates these two sequences' SNR levels following Sec. IV-C1,

denoted as $SNR_{\max}$ and $SNR_{\min}$ respectively. Suppose channel distortion reduces the gain equally by a discounting factor α, *e.g.*, $G_{\max}(\alpha) = 10\log_{10}(\alpha N_{\max}), 0 < \alpha \leq 1$. Then, the AP can estimate α as follows:

$$\alpha = (N_{\min}/N_{\max}) \cdot 10^{(SNR_{\max} - SNR_{\min})/10} \tag{5}$$

Then, for any other spreading code with length N, it predicts the processing gain as: $G_N(\alpha) = 10\log_{10}(\alpha N)$.

Denote SNR_N as the absolute achievable SNR when using DSSS code length N. Then it can be estimated as:

$$\begin{aligned} SNR_{\text{baseline}} &= SNR_{\min} - G_{\min} \\ SNR_N &= G_N(\alpha) + SNR_{\text{baseline}} \end{aligned} \tag{6}$$

where SNR_{baseline} is the absolute SNR when no DSSS code is used, or equivalently, code length $N = 1$.

D. Per-packet code assignment

Based on the SNR predicted using Eq. (6) and model in Sec. IV-C1, a receiver can map the estimated BER ε^m to expected instantaneous throughput under a given configuration. For simplicity, we assume no error correction code is adopted. Then, the packet level throughput while using a DSSS code c_m of length n_m can be modeled as:

$$Th^m = \frac{L * (1 - \varepsilon^m)^L}{n_m \times t} \tag{7}$$

where L is the packet size and t is the original packet duration without using any DSSS code and including MAC layer overheads. This can be estimated using the model proposed in [14]. The AP can choose the DSSS code c_m that maximizes Th^m.

E. Traffic-aware multi-packet code assignment

The traffic-aware multi-packet code assignment algorithm is designed to find utility-optimal code assignment for blended traffic patterns, depending on their delay/throughput requirements. Recall that, the AP runs this algorithm by leveraging the candidate uplink packets' priority and length information collected from the clients' probe packet. We first formulate the code assignment as a utility maximization problem and solve it using a dynamic programming framework. Then, we use the solution to assign optimal DSSS codes to candidate packets.

1) Code assignment as a utility maximization problem: **Problem formulation.** Denote T as the duration of probing interval. Suppose a client has J candidate packets for the interval. The j^{th} packet has length L_j. Each packet contains possibly different priority levels denoted by, $p_1, p_2, \ldots, p_J$. We will discuss in Sec. V-D about how to design the priority levels based on the applications' delay/throughput requirements. Suppose there are M codes available denoted by, $c_1, c_2, \ldots, c_M$, which are sorted descendingly according to their length n_i ($i = 1, 2, \ldots, M$). The BER ε_j^i for a packet j while choosing the DSSS code c_i of length n_i can be estimated by Eq. (4). The corresponding instantaneous throughput is denoted by, Th_j^i and, can be calculated by leveraging Eq. (7).

Let u_j^i be the utility obtained when receiving an uplink packet j modulated using code c_i. To incorporate both priority and throughput, we model the utility as:

$$u_j^i = p_j \times Th_j^i \tag{8}$$

Let the binary variable $x_j^i \in \{0, 1\}$ indicate if packet j should be modulated and sent with DSSS code c_i of length n_i. The problem of utility-optimal code assignment can be cast as:

$$\begin{aligned} \max \quad & \sum_{j=1}^{J}\sum_{i=1}^{M} x_j^i u_j^i \\ \text{s.t.} \quad & \sum_{i=1}^{M} x_j^i \leq 1, \quad \text{and} \sum_{j=1}^{J}\sum_{i=1}^{M} x_j^i n_i t_j \leq T \end{aligned} \tag{9}$$

The first constraint means a packet can be modulated using at most 1 DSSS code. The second constraint requires the total amount of time for transmitting all the packets should not exceed the probing interval. If a packet is not assigned any code, it should be deferred to in the next interval.

Problem (9) can be reduced from multiple-choice 0-1 Knapsack problem [15, p. 425–427], and thus is NP-hard. Fortunately, we find it has inherent optimal substructures that allow for a dynamic programming solution, which we detail below.

Optimal substructure and solution. We first sort candidate packets in descending order of priority. Define $U(j, i, t)$ as the optimal utility for packets $P_l, l = 1, 2, \ldots, j$ with DSSS code up to c_i (code length n_i) and total transmission time bound t. Let $\tau_{j_1,j_2,i} = \sum_{l=j_1}^{j_2} n_i t_l$ be the amount of time needed to transmit packets within index range $[j_1, j_2]$, and using code c_i. To compute the optimal utility $U(j, i, t)$, note that only the last few packets ending at P_j may choose code c_i. Denote these packets as $P_l, l = k+1, \ldots, j$ (if $k = j$, then no packets are transmitted with code c_i). Then, we find the optimal substructure by representing $U(j, i, t)$ as the summation of the optimal utility of the first k packets using codes up to c_{i-1} within the remaining time $t - \tau_{k+1,j,i}$, and the utility obtained by transmitting packets $k+1$ to j using DSSS code c_i. Maximizing over all possible k, we obtain the recursive solution for $U(j, i, t)$ as follows,

$$U(j, i, t) = \max_{0 \leq k \leq j} \Big[U(k, i-1, t - \tau_{k+1,j,i}) + \sum_{l=k+1}^{j} u_l^i \Big] \tag{10}$$

$$q(j, i, t) = \operatorname*{argmax}_{0 \leq k \leq j} \Big[U(k, i-1, t - \tau_{k+1,j,i}) + \sum_{l=k+1}^{j} u_l^i \Big] \tag{11}$$

In Eq. (11), $q(j, i, t)$ is the memoization data structure of dynamic programming that keeps track of the best parameter k and corresponding DSSS code c_i that solves Eq. (10), which is later used for the optimal code assignment.

For the whole probing interval T, we can compute the optimal utility U^* and jointly calculate the optimal packet number j^* and optimal DSSS code index i^* as:

$$U^* = \max_{\substack{1 \leq j \leq J \\ 1 \leq i \leq M}} U(j, i, T), \quad \{j^*, i^*\} = \operatorname*{argmax}_{\substack{1 \leq j \leq J \\ 1 \leq i \leq M}} U(j, i, T) \tag{12}$$

where j^* achieves the optimal utility, indicating that packets $j > j^*$ are dropped from the current probing interval and will be deferred to in the next probing interval.

Algorithm. The dynamic programming procedure to find the optimal utility, maximum number of packets and DSSS code index within the probing interval is illustrated in Alg. 1.

Boundary conditions. To bootstrap the recursive solution to Eq. (10), we derive the boundary conditions for $U(j, i, t)$ as:

$$\begin{aligned} U(j, i, t) &= -\infty, \quad \text{if} \quad t < 0 \\ U(j, 0, t) &= -\infty, \quad \text{if} \quad j > 0, t \geq 0 \\ U(0, i, t) &= 0, \quad \text{if} \quad i \geq 0, t \geq 0 \end{aligned} \tag{13}$$

The first two equations state that, $t < 0$, or $i = 0$ and $j > 0$ is not a valid choice for the utility function $U(j, i, t)$. The valid

Algorithm 1 Packet scheduling in a probing interval

1: Compute the boundary conditions using Eq. (13).
2: **for** all j, i, t **do**
3: Compute $U(j, i, t)$ iteratively using Eq. (10).
4: Calculate and store $q(j, i, t)$ using Eq. (11).
5: **end for**
6: Find the optimal utility U^*, optimal packet number j^* and the optimal DSSS code c_{i^*} using Eq. (12).

Algorithm 2 Multi-packet DSSS code assignment

1: $t = T$, $j = j^*$, $i = i^*$, $k = q(j, i, t)$.
2: Packets $P_l, l = k+1, \ldots, j$ are assigned DSSS code c_i (if $k == j$, no packets are assigned DSSS code c_i).
3: If $k \leq 0$, go to Step 4. Otherwise, $t = t - \tau_{k+1,j,i}$, $j = k$, $i = i - 1$, $k = q(j, i, t)$, go to step 2.
4: All packets have been assigned optimal DSSS codes.

DSSS code index choices range from 1 to M and $i = 0$ is a dummy index for initialization only. The third equation models a dummy packet with index $j = 0$ and zero utility.

2) Optimal DSSS code assignment: Finally, the multi-packet code assignment algorithm assigns DSSS codes in the current probing interval T, based on the above utility-maximization solution IV-E1. This is formally described in Alg. 2. For J uplink packets, M DSSS codes and probe interval T, both the time and space complexity of the algorithm is $\mathcal{O}(JM\lceil\frac{T}{\tau_{min}}\rceil)$, where τ_{min} is the minimum transmission time required for one data packet. The algorithm is pseudo-polynomial as the time complexity is polynomial *w.r.t.* the *value of* $\lceil\frac{T}{\tau_{min}}\rceil$ but is exponential *w.r.t.* the *number of bits* required to store $\lceil\frac{T}{\tau_{min}}\rceil$ [15]. In practice, each interval T only contains tens to hundreds of packets, thus the algorithm can run very efficiently. Indeed, this is what we observed in our implementation.

Optimality of DSSS code assignment when probe interval changes. Recall a client can terminate the current interval under sudden channel condition changes. It may seem that the above code assignment algorithm will be invalid then. However, owing to the optimal substructure in the utility-maximization solution, any solution that has been executed up to this point is still optimal. Said differently, the multi-packet DSSS code allocation algorithm is independent of sudden channel condition change.

F. Adapting the probing interval

Measurement studies show that channel coherence time may vary between 10 ms. and 200 ms. in real vehicular networks depending on speed [16]. We thus adopt an *Additive Increase / Multiplicative Decrease* (AIMD) strategy to update the adaptation interval. This is formally described in Alg. 3.

More specifically, the client starts with maximum interval $T_{\max}$ and sends the probe packet to the AP. In subsequent uplink transmission, the client keeps a moving average of packet throughput Th_{win} across the interval. Let E be the expected throughput estimated during initial code assignment. When Th_{win}/E drops below a threshold σ (0.8 by default), the channel is likely to have changed remarkably compared to the beginning of the interval. Thus, the client decreases the adaptation interval by a multiplicative factor $\eta = \frac{Th_{win}}{\sigma E}$.

Algorithm 3 Update probing interval and choose between per-packet and multi-packet code assignment protocols

1: Initialize probing interval T: $T = T_{\max}$.
2: Send a probe packet to AP for multi-packet code assignment in interval T (Sec. IV-E).
3: Keep moving-average throughput Th_{win} over an interval.
4: **if** $Th_{win} \approx 0$ **then** Invalidate codes and go to step 2.
5: **else if** $Th_{win} < \sigma E$ **then** $T = \max(T \times \frac{Th_{win}}{\sigma E}, T_{\min})$;
6: **else** $T = \min(T + \beta, T_{\max})$;
7: **end if**
8: **if** $T == T_{\min}$ **then** Send a probe packet to AP for per-packet code assignment (Sec. IV-D) and go to step 3.
9: **else** Go to step 2.
10: **end if**

Otherwise, the vehicle increases the probing interval T by β (additive increase). The min. and max. interval lengths ($T_{\min}$ = 1 ms. and $T_{\max}$ = 200 ms.) are capped empirically (Sec. V).

V. Evaluation

A. Testbed and prototype implementation

We implement and evaluate the adaptive DSSS protocol on the WARP software-radio platform [17]. Each WARP board is paired with WURC (Fig. 8), a third-party RF front-end [18] that enables communication over the TVWS band.

To prototype the adaptive DSSS, we first port the GNURadio implementation of 802.11b PHY (with 11-bit Barker Code modulation) to the WARP driver. Then, we re-implement the modulation/demodulation library, which enables DSSS communication with 2^{th}-power code sequences (Sec. IV-B) and the flexibility to switch between them. We further reengineer the preamble structure for the probe packet and implement the runtime estimation algorithms (Sec. IV-C). For benchmark comparison, we also implement an OFDM modulation/demodulation layer, following the WARP 802.11g prototype in [19]. This OFDM layer uses BPSK modulation by default, but the post-decoding SNR is mapped to an optimal bit-rate using a look-up table, in order to emulate an oracle rate adaptation scheme.

We found the interface and signal demodulation latency of the WARP testbed is several hundred milliseconds per-packet, which is unsuitable for fine-time adaptation. We circumvent this limitation by continuously sending a pre-built packet comprised of the probing preamble, which drastically cut the inter-packet latency to 9 ms. AP's radio first stores all the received raw samples, and then processes the samples, demodulate the DSSS coded symbols, thus obtaining the per-packet SNR, BER and bit-rate (Sec. IV-C and IV-D) at a 9 ms. granularity. Given these per-packet statistics, we run the adaptive code assignment and interval update algorithms (Sec. IV).

For all outdoor experiments, we place the AP and clients similarly to our field-study in Sec. II. The outdoor client node runs inside a car with an omni-directional antenna on top (Fig. 8). We also run some of the benchmark experiments in a more controlled office environment (Fig. 2). All the experiments are conducted under a FCC experimental license, which allows us to use the vacant TV channels 40 and 41 (626 – 638 MHz) in our area. We limit the WARP's communication bandwidth to 10 MHz to be compatible with the 802.11af 10 MHz mode.

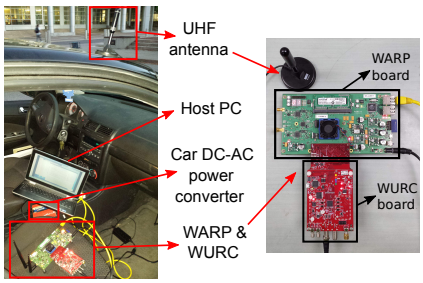

Fig. 8. Outdoor experimental setup in a car.

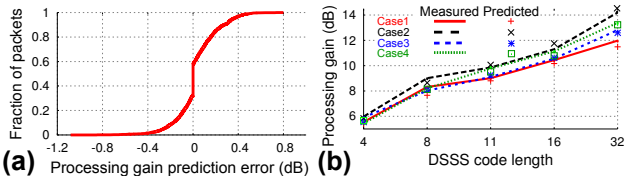

Fig. 9. (a) Accuracy of processing gain prediction. (b) An example of measured vs. predicted processing gain.

B. Micro-benchmarks

We start by evaluating the efficacy of each individual design component in the adaptive DSSS protocol.

1) Accuracy of processing gain estimation: We first place the nodes in fixed locations and evaluate the preamble-based processing gain prediction algorithm (Sec. IV-C2), in comparison with an oracle that directly uses all the code sequences to compute the processing gain of each. Each experiment runs continuously across $10,000$ transmissions, and Fig. 9(a) shows the CDF of prediction error. The 95^{th} percentile prediction error is only $\pm$ 0.8 dB with worse case estimation error of only -1.1 dB. Fig. 9(b) shows the measured and predicted processing gain across 4 different locations, which exhibits high consistency.

2) Maintaining uplink connectivity: To evaluate the capability to maintain uplink connectivity under power asymmetry, we follow similar outdoor settings as in Sec. II-B. Fig. 10 shows the measured uplink throughput of all 8 locations. For OFDM uplink with oracle bit-rate selection, when the SNR is high (P3 and P7), the throughput can be higher compared to DSSS which only uses BPSK under all code lengths. Unfortunately, for locations P1, P2, P4, P5, P6 and P8, the OFDM uplink has 0 throughput, whereas DSSS can still sustain the connection, albeit at the cost of low throughput. We emphasize again that for high-SNR uplink connections, the adaptive DSSS can switch to OFDM modulation to achieve high throughput (Sec. IV-A). Our evaluation isolates these two schemes in order to zoom in the efficacy of adaptive DSSS alone. For low-SNR cases, DSSS's achievable throughput is fundamentally limited by the Shannon's law, and can be improved by opportunistically aggregating spectrum resources.

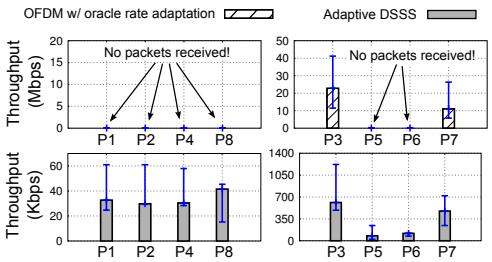

Fig. 10. Adaptive DSSS compared to OFDM in 8 outdoor locations.

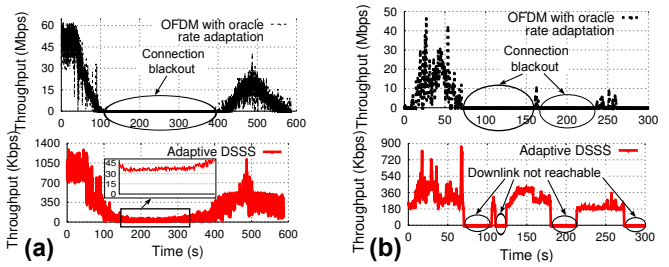

Fig. 11. Performance of adaptive DSSS scheme in uplink compared to OFDM: (a) Indoor walking. (b) Outdoor driving.

C. Performance of per packet code length adaptation

Fig. 11(a) shows the throughput variation when the client moves at walking speed indoor. Between $100^{th} - 400^{th}$ seconds, wall blockages cause a complete connection blackout for OFDM, even with oracle rate adaptation. However, the adaptive DSSS can maintain the uplink connectivity with 35 kbps throughput, which is critical for low-rate real-time and downlink-dominated traffic. The observation is similar for outdoor driving case (Fig. 11(b)). Although building blockage causes connection blackout even in adaptive DSSS case, we can see the uplink can be sustained whenever the downlink can be reached. Said differently, the adaptive DSSS successfully bridges the range gap caused by power asymmetry. In contrast, OFDM can sustain the connection only for 43% of the time.

D. Traffic-aware multi-packet code length adaptation

We run the traffic-aware code adaptation algorithm using a trace based emulation by collecting WiFi packet traces from *(i)* a FTP session downloading a 25 MB file, *(ii)* a 3-minute web browsing session, *(iii)* a 3-minute VoIP session using Google+ hangout, and *(iv)* a 3-minute youtube video streaming/downloading session. We also generate a synthetic trace for periodic GPS update to emulate location-based services. These traces contain timestamped downlink/uplink data packets, which are fed into the AP/client's outgoing queues and served by the adaptive DSSS protocol. In the traces, we found only 9.6% data (28% of packets) correspond to uplink traffic. The traces represent a real vehicular networking scenario with safety and infotainment applications running together [20].

Given R coexisting traffic patterns, we set traffic priority range to [1,R]. The real-time and delay-sensitive traffics (*i.e.* GPS update, VoIP) are statically assigned highest priority R. The priority of the non-realtime throughput-intensive (*e.g.* FTP, video downloading) traffic dynamically changes following: (1 - *currentThroughput/targetThroughput*)$\times$(R-1). We compute the current throughput via a moving average, and set the target throughput empirically according to application type.

1) Real-time traffic: We first run two real-time traffic flows (VoIP & GPS update) simultaneously between the AP and the vehicle client in outdoor driving mode. For every 10-second window in the traffic traces, we calculate the percentage of packets that miss their deadlines. For GPS update, we consider the deadline of a packet as the average of current and next packet arrival time, while for VoIP, the deadline is the next packet arrival time. Besides the oracle OFDM, we also compare with a naive DSSS scheme that randomly assigns code length to uplink packets. Note again, the downlink always uses the

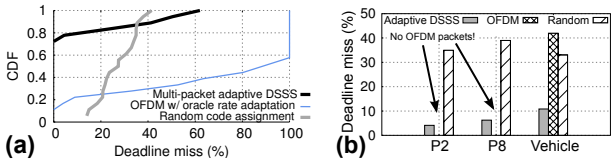

Fig. 12. (a) CDF of deadline miss percentage of real-time traffics when running in outdoor driving scenario. (b) Average deadline miss percentage of three different clients.

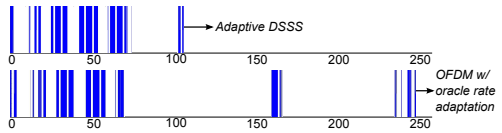

Fig. 13. Time series plot of a 80 s. video downloading in different systems for the outdoor driving scenario. The dark color shows the downloading time and the light color shows the delay.

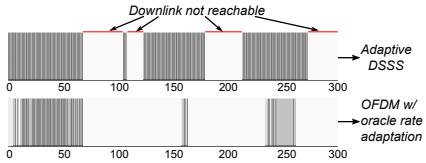

Fig. 14. Time series plot of 300 s. GPS updates for outdoor driving scenario. Dark color shows update time and light color shows the miss.

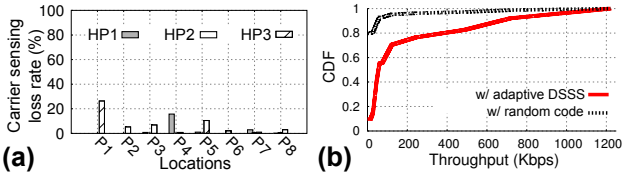

Fig. 15. (a) Carrier sensing loss rate at three high power clients after running the code adaptation algorithm. (b) CDF of throughput across all locations w/ and w/o adaptive DSSS running.

OFDM modulation as in 802.11af. To isolate the artifact caused by building blockage, we discard time period where downlink itself was not connected, thus focusing on the problems caused by power asymmetry alone.

Fig. 12(a) shows the CDF of the deadline miss percentage. The OFDM uplink experiences an average deadline miss percentage of 45%, and has 100% deadline miss for more than 41% of the time. This is mainly because the OFDM uplink is completely broken for a significant amount of time and hence the client can not even send requests or ACK towards the AP. In contrast, with adaptive DSSS, we see more than 74% of the times, it observed no deadline miss and worst case deadline miss is only 62% across all the 10-second windows. The random code assignment scheme has severe deadline misses (37% deadline miss probability for 74% of the time) as it does not allocate codes to packets in uplink queue based on their requirement. However, it still outperforms OFDM owing to the DSSS processing gain that bridges the power asymmetry.

2) Coexistence of real-time and non-real-time traffic: Fig. 13 and 14 plot the time series diagram of video downloading and GPS update when running together with all the aforementioned traffics patterns. For video downloading, we use a 480p video of playback length 80s. The adaptive DSSS can finish its download around 102s., while the OFDM with oracle rate adaptation takes about 240s. to finish because of the severe uplink blackouts. We expect the OFDM's latency will be more severe when running in an actual TVWS network stack, which involves huge connection setup overheads, such as association/re-association, TCP timeout and slow start, *etc.* Due to the limitation of our testbed, verification of such impact is left for our future work.

For the GPS traffic, the vehicle sends periodic GPS update of 20-byte packet every 500 ms. (typical in safety applications [20]). With OFDM PHY, the server got GPS updates for only 42% of the time within a 300s period. In contrast, with adaptive DSSS at uplink, we can see 92% of the GPS updates sent successfully (a 2.2× improvement!).

3) Multi-client scenario: We also ran multi-packet code adaptation in multi-client scenario with two static clients (locations P2 and P8) and the same vehicle client as before (Fig. 3). The applications are: (i) P2: VoIP, FTP and GPS update, (ii) P8: GPS update and web browsing, (iii) vehicle: video streaming and GPS update. For this experiment, we consider TVWS spectrum aggregation with 20 MHz channel which doubles link throughput in the trace driven emulation. We emulate 802.11af CSMA/CA protocol to arbitrate contention between clients.

Fig. 12(b) shows the percentage of deadline misses across all packets. For all three clients, adaptive DSSS has an average miss rate of 3.1% to 10.8%, in contrast to 34% to 39.5% for random code assignment. OFDM experiences connectivity loss for P2 and P8, and a miss rate of 41% for the vehicle client.

E. Performance in presence of high power fixed clients

Recall that, a high-power fixed client often fails to sense mobile clients (Sec. II-C). We now follow the same setup as in Sec. II-C to verify how adaptive DSSS can alleviate the problem. Fig. 15(a) shows the carrier sensing loss rate when each fixed client attempts to sense the outdoor mobile client. We observe that the adaptive DSSS reduces the loss rate by around 85% (67% for P1, $>$ 88% for others, in contrast to Fig. 4(a)). However, it does not completely eliminate the loss, partly because it occasionally resorts to a short code for efficiency, thus reducing the mobile client's signal coverage. Fortunately, when suffering from packet losses due to high-power clients' interference, as per Alg. 3, the client will resort to single packet code adaptation that uses the probe preamble (containing a long sequence) per-frame to ensure coverage. In case when the longest DSSS code cannot be sensed by the fixed clients (*e.g.*, too far away or severely blocked), the problem is similar to the traditional hidden terminals (and no longer due to power asymmetry), which we do not aim to solve in this paper.

Fig. 15(b) also shows the throughput CDF across all 8 locations and across 5 min. Throughput is calculated over 0.1s windows for each client. Random code assignment suffers from the fixed client interference sporadically, resulting in 0 median throughput. With adaptive DSSS, the median throughput is sustained at 60 kbps and 80 percentile can be 0.5 Mbps.

VI. Related Work

TV whitespace networking. The prospects of the emerging whitespace spectrum have triggered multiple standardization activities. Besides the aforementioned IEEE 802.11af, the IEEE 802.22 [21] specifies MAC/PHY mechanisms for regional area connectivity using TVWS. Several customized protocols have addressed the new challenges when extending WiFi to the whitespace. WhiteFi [22], for example, designs a lightweight

link quality sensing and bandwidth assignment scheme to tackle spectrum fragmentation and spatial/temporal variation.

Mobile networking over TVWS and ISM band. To date, TVWS networking research has mainly focused on spectrum sensing/management and other low-layer issues [6], while the real-world applications remain largely under-explored [23]. A campus-wide whitespace vehicular network has been reported in [24]. The measurement revealed that the AP/client power asymmetry leads to $4\times$ of coverage gap between downlink and uplink. Recently, Scout [5] is designed to circumvent the problem using heterogeneous link access, which binds a whitespace downlink to a cellular uplink. However, to reach a whitespace AP, an uplink packet needs to traverse the cellular and Internet infrastructure, resulting in a latency of tens to hundreds of milliseconds. Our adaptive DSSS solution overcomes this limitation through a cross-layer design — it reengineers the uplink PHY to bridge the coverage gap between the downlink.

Coexistence of power-asymmetric links. Harmful coexistence often occurs when wireless devices adopt different transmit power levels. A measurement study in [25] revealed such an issue when low-power ZigBee coexists with high power WiFi. A high-power busy-tone can alleviate the problem and protect the vulnerable weak transmitters. Similar solution has shown to be effective when 802.22 devices coexist with 802.11af [26]. Alternative solutions, such as Weeble [12], allow weak nodes to emit long preambles that can be sensed despite their low transmit power. These protocols mainly aim to reduce interference between heterogeneous networks, whereas our adaptive DSSS scheme enhances communication and connectivity between nodes in the same network.

Wireless rate adaptation. The problem of adapting code length shares some spirit with bit-rate adaptation, which has been extensively studied in 802.11 networks (*e.g.*, [27], [28]). Both entail strategic choice between different communication schemes. Yet adaptive DSSS faces several unique challenges. In particular, there does not exist a fixed mapping between a code length and link throughput, because of the channel-dependent processing gain that must be estimated on-the-fly (Sec. III). Moreover, the tradeoff between link reliability (long code length) and overhead becomes prominent especially when diverse vehicular network traffic patterns are mixed together. Our work is arguably the first to apply DSSS to solve the link asymmetry in whitespace networks, and adapt the design to counter network/traffic dynamics.

VII. Conclusion

This paper presents a protocol to bridge the huge link power asymmetry in mobile TVWS networks for providing long range and robust wireless connectivity to vehicles. Our experiments reveal that this power asymmetry causes severe uplink connectivity blackouts and starvation of mobile nodes. Through a practical cross-layer design, using simple components that are built from existing WiFi modules, the protocol ensures robust and efficient uplink communication in TVWS spectrum which is constrained by stringent rules from the FCC. In addition, we strike a fine balance through a unique traffic-aware code length assignment algorithm when heterogeneous traffic patterns coexist. Thus, our design provides a viable and effective means to realize vehicular networks over the TVWS.

Acknowledgement

The work reported in this paper was supported in part by the NSF under Grant CNS-1318292, CNS-1343363, CNS-1350039 and CNS-1404613.

References

[1] P. Rodriguez, R. Chakravorty, J. Crowcroft, J. Chesterfield, and S. Banerjee, "MAR: A Commuter Router Infrastructure For the Mobile Internet," in *Proc. of ACM MobiSys*, 2004.
[2] J. Hare, L. Hartung, and S. Banerjee, "Beyond Deployments and Testbeds: Experiences with Public Usage on Vehicular WiFi Hotspots," in *Proc. of ACM MobiSys*, 2012.
[3] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Interactive WiFi Connectivity for Moving Vehicles," in *ACM SIGCOMM*, 2008.
[4] J. Eriksson, H. Balakrishnan, and S. Madden, "Cabernet: Vehicular Content Delivery using WiFi," in *Proc. of ACM MobiCom*, 2008.
[5] T. Zhang, S. Sen, and S. Banerjee, "Enhancing Vehicular Internet Connectivity using Whitespaces, Heterogeneity, and a Scouting Radio," in *Proc. of ACM MobiSys*, 2014.
[6] M. Nekovee, "Cognitive Radio Access to TV White Spaces: Spectrum Opportunities, Commercial Applications and Remaining Technology Challenges," in *Proc. of IEEE DySPAN*, 2010.
[7] FCC, "Unlicensed operation in the TV broadcast bands, Second Memorandum Opinion and Order," September 2010.
[8] H.-S. Chen and W. Gao, "MAC and PHY proposal for 802.11af," 2010.
[9] H. Holma and A. Toskala, *WCDMA for UMTS: HSPA Evolution and LTE*. John Wiley & Sons, Inc., 2007.
[10] J. Proakis, *Digital Communications, 3^{rd} edition*. McGraw Hill, 2001.
[11] B. Muntwyler, V. Lenders, F. Legendre, and B. Plattner, "Obfuscating IEEE 802.15.4 communication using secret spreading codes," in *Proc. of IEEE Wireless On-demand Network Systems and Services (WONS)*, 2012.
[12] B. Radunović, R. Chandra, and D. Gunawardena, "Weeble: Enabling Low-Power Nodes to Coexist with High-Power Nodes in White Space Networks," in *Proc. of ACM CoNEXT*, 2012.
[13] W. J. P. Jr. and J. J. Komo, "The Autocorrelation of M-sequences Over Nonprime Finite Fields," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 24, July 1998.
[14] S.-C. Wang and A. Helmy, "BEWARE: Background Traffic-Aware Rate Adaptation for IEEE 802.11," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, August 2011.
[15] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*. The MIT Press, 2009.
[16] L. Cheng, B. Henty, D. Stancil, F. Bai, and P. Mudalige, "Mobile Vehicle-to-Vehicle Narrow-Band Channel Measurement and Characterization of the 5.9 GHz Dedicated Short Range Communication (DSRC) Frequency Band," *IEEE JSAC*, vol. 25, no. 8, 2007.
[17] A. Khattab, J. Camp, C. Hunter, P. Murphy, A. Sabharwal, and E. W. Knightly, "WARP: a Flexible Platform for Clean-Slate Wireless Medium Access Protocol Design," *ACM MC2R*, vol. 12, 2008.
[18] Volo Wireless LLC., "Wideband UHF Daughter Card (WURC)," 2014.
[19] X. Zhang and K. Shin, "Adaptive Subcarrier Nulling: Enabling partial spectrum sharing in wireless LANs," in *Proc. of IEEE ICNP*, 2011.
[20] Toyota Motor Corporation, "Entune® App Suite," 2014.
[21] IEEE, "IEEE 802.22 wireless regional area networks," 2010.
[22] P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh, "White Space Networking with Wi-Fi like Connectivity," in *Proc. of ACM SIGCOMM*, 2009.
[23] J. Wang, M. Ghosh, and K. Challapali, "Emerging Cognitive Radio Applications: A Survey," *IEEE Comm. Magazine*, vol. 49, no. 3, 2011.
[24] R. Chandra, T. Moscibroda, P. Bahl, R. Murty, G. Nychis, and X. Wang, "A Campus-Wide Testbed Over the TV White Spaces," in *MC2R*, 2011.
[25] X. Zhang and K. G. Shin, "Cooperative Carrier Signaling: Harmonizing Coexisting WPAN and WLAN Devices," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, 2013.
[26] X. Feng, Q. Zhang, and B. Li, "Enabling Co-channel Coexistence of 802.22 and 802.11af Systems in TV White Spaces," in *IEEE ICC*, 2013.
[27] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan, "Robust Rate Adaptation for 802.11 Wireless Networks," in *ACM MobiCom*, 2006.
[28] M. Vutukuru, H. Balakrishnan, and K. Jamieson, "Cross-Layer Wireless Bit Rate Adaptation," in *Proc. of ACM SIGCOMM*, 2009.