Providing Disruption QoS in an OFDM System Using Residual Idle Time Based Opportunistic Spectrum Access

Anirudha Sahoo, Michael Souryal, and Mudumbai Ranganathan
National Institute of Standards and Technology
100 Bureau Drive, Gaithersburg, MD, 20899, USA
email: {anirudha.sahoo, souryal, ranga}@nist.gov

***Abstract*—Dynamic spectrum access (DSA) is emerging as a promising technology to mitigate spectrum scarcity caused by static frequency allocation. Among the different models proposed for DSA, opportunistic spectrum access (OSA) is a promising class of solutions. In any DSA system, it is important to make sure that the performance of primary users (PUs) does not degrade significantly as a result of dynamic use of spectrum by secondary users (SUs). One of the main reasons for slow commercial adoption of OSA technology is the lack of system implementations which can demonstrate bounding interference to PUs below a given threshold. In this paper, we report a preliminary implementation of an OSA scheme which can ensure that the interference to the PU is bounded. We present our experience and challenges in building this system using the open source software toolkit GNU Radio on a software defined radio (SDR) implemented with a Universal Software Radio Peripheral (USRP).**

I. Introduction

Exclusive and static allocation of spectrum has led to artificial scarcity of spectrum. Dynamic spectrum access (DSA) is a promising technology that can address this problem. In DSA, primary users (PUs) are the licensed (or intended) users of the spectrum. In addition to the PUs, secondary users (SUs) can dynamically use the same spectrum as long as they do not cause significant interference to the PUs. There are three models proposed for DSA: *underlay*, *overlay*, and *interweave* [1]. In this paper, we propose a scheme based on the interweave model. This model is also known as *opportunistic spectrum access* (OSA), since SUs access the channel opportunistically when the PU is idle.

There are many challenges in implementing an OSA scheme. One of the most difficult and important challenges is to determine how long an SU should transmit in an idle period so that it does not cause significant interference to the PUs. This is difficult because PUs normally do not communicate with SUs. As a result, SUs do not know exactly when the idle period will end. In this paper, we implement an OSA scheme which addresses this challenge.

One of the main reasons for slow commercial adoption of OSA technology is skepticism among PU service providers about SU interference to the PU. There have been very few implementations of OSA systems which can show that the interference to the PU can be bounded. Thus, one of the most crucial factors in gaining industry adoption is to provide quality of service (QoS) with respect to disruption caused to PUs. We refer to this as *disruption QoS*. There can be different metrics to measure disruption QoS. Collision probablity with the PU has been used as a metric in [2]. Some papers have used similar but slightly different metrics. Authors in [3], [4] use *interference probability* as the performance metric, defined as the probability that a given SU transmission runs into a busy period. Sung et al. [5] use *interference violation probability* as a metric which is defined as the ratio of the average duration of an interference event to the average duration of the effective white space (effective white space is the actual white space minus the sensing overhead). Nasreddine et al. [6] define a performance parameter θ to be the ratio of conditional expected length of PU ON periods (given that an SU transmits) to the unconditional expected length of a PU ON period. A MAC based on partially observable Markov decision process (POMDP) has been proposed in [7] which maximizes SU network throughput while limiting the interference to the PU below a given threshold.

For some applications collision probability may not be an appropriate metric. For example, Huang et al. [2] propose *percentage overlapping time*, defined as the fraction of time an SU's transmission overlaps with a PU's transmission, averaged over a long time period. They cite the example of a PU hosting a voice application in which the call duration (busy period) is much longer than the duration of a packet transmitted by a SU. Hence, in this case percentage overlapping time is a more appropriate metric than collision probability.

In this paper, we consider the *Residual Idle Time Based Scheme* (RIBS) proposed in [3], which is an OSA scheme that provides disruption QoS in terms of interference probability. This scheme models PU traffic as an alternating renewal process (ARP) and the sensing instant of an SU is modeled as a random incidence into the ARP. Using the theory of random incidence into a renewal process, RIBS then computes the distribution of residual idle time. Using the residual idle time distribution it then computes the maximum duration for which the SU can transmit so that the interference probability is below a given threshold.

The study of RIBS reported in [3] was carried out in a simulation environment. In this paper, we present our experience and results of the first attempt to implement a preliminary version of RIBS on GNU Radio [8] over USRP hardware[1][9]. The primarily goal of this project is to demonstrate that using RIBS it is possible to provide disruption QoS in an orthogonal frequency division multiplexing (OFDM) based communication system on commercially available communi-

[1]Certain commercial equipment, instruments, or materials are identified in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipments identified are necessarily the best available for the purpose.

cation hardware. There were several challenges along the way to achieving this goal:

- *Modifying RIBS for OFDM based Communication*: We chose OFDM for the physical layer (PHY) because OFDM provides better protection against frequency selective fading and makes more efficient use of spectrum. Also, many modern architectures (e.g., LTE, WiMax) use OFDM as their PHY technology. The original RIBS implementation in [3] was for a single channel. We had to suitably modify RIBS to adapt it to OFDM.
- *Full duplex communication for OFDM on GNU Radio*: Although GNU Radio comes with a basic OFDM sender and receiver application, it provides only one-way communication. However, for RIBS, the sender needs to inform the receiver which subcarriers it should tune to for the next burst of packets. And once the receiver tunes to the indicated subcarriers, it sends an ACK to the sender. Thus, for our purpose, we needed full duplex communication.
- *Porting Papyrus to a new platform*: We started with a software platform called *Papyrus* [10], which had implemented a full duplex OFDM application using GNU Radio with the USRP. However, the original Papyrus implementation was for an older version of GNU Radio and an older USRP (USRP1). We had to port it to a later version of GNU Radio (version 3.6.5) and newer USRP hardware (USRP N210).
- *Robust handshaking between SU sender and receiver*: After porting Papyrus to later versions of GNU Radio (i.e., version 3.6.5) and the USRP driver, we found that the handshaking between the sender and the receiver is not very robust. As a result, the communication between sender and receiver would stop after a few bursts of packets are sent. We analyzed the handshaking protocol implemented in Papyrus and improved it to make the communication robust.

II. Overview of RIBS

RIBS is an OSA scheme which provides disruption QoS to PUs in terms of interference probability. In RIBS the PU channel occupancy is modeled as an alternating renewal process (ARP). The process alternates between idle and busy periods as per the traffic pattern of the PUs. RIBS assumes that the idle time distribution of the ARP is known.[2] The SUs sense the medium at a random time to search for white spaces (or idle time). After sensing the medium to be idle, the SU transmits for a duration based on the residual idle time distribution such that the probability of interference with the PU is below a given threshold. After a successful transmission, the SU backs off for an exponential random duration (with respect to the last sensing time) and repeats the process again. If the medium is busy, then the SU does not transmit but backs off for an exponential random duration and senses again.

Let I be the random variable representing the idle time. The SU senses the channel at a random time. Let RI be the random variable representing residual idle time. We denote with $f(\cdot)$ and $F(\cdot)$ the probability density and cumulative distribution functions of a random variable, respectively. From the theory of random incidence into a renewal process we have [11, pp. 331]:

$$f_{RI}(y) = \frac{1 - F_I(y)}{E[I]} \tag{1}$$

$$F_{RI}(y) = \int_0^y f_{RI}(z)dz \tag{2}$$

where $E[I]$ denotes the expected channel idle time.

The SU should transmit in the residual idle period for a duration y such that the probability of interference is less than a given threshold, say η. Note that interference occurs when $RI < y$. Thus, the interference probability constraint is given by

$$F_{RI}(y) \leq \eta \tag{3}$$

Note that the interference probability is measured with respect to the number of SU transmissions. In some applications, it may be more appropriate to measure the interference with respect to the number of PU transmissions. The latter metric is harder to compute theoretically, but we are working on formulating a metric with respect to PU transmission.

So, to provide disruption QoS in terms of interference probability, an SU, given the primary idle time distribution, computes the residual idle time distribution. Then it computes the maximum value of y (say y_{max}) for which (3) holds. This is the duration for which an SU transmits once it randomly senses the medium to be idle.

III. Adaptation of RIBS to OFDM System

RIBS has been implemented in a simulation environment for a single channel system [3]. However, in OFDM, a PU sender uses multiple subcarriers for transmission. In this section, we explain how RIBS is adapted to an OFDM system. We divide the available subcarriers of the OFDM system into multiple *logical channels*. Each logical channel consists of a set of non-overlapping subcarriers which are used by a PU sender and receiver pair for communication.

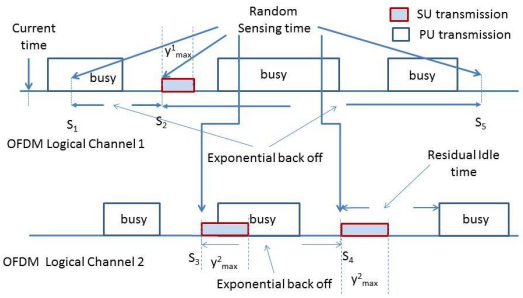

Fig. 1. OFDM Adaptation of RIBS using Logical Channels

Figure 1 shows the schematic diagram of how the start of sensing instances are determined in an OFDM system with two logical channels. As shown in the figure, the sensing times are determined based on exponential backoff. The algorithm followed by a transmitting SU is simple: For each logical channel (say, C_k), given the PU idle time distribution, the SU transmitter computes the corresponding maximum duration (y^k_{max}). Among the sensing times of all the logical channels, it picks the one nearest to the current time and waits until that time and then senses that logical channel. For example, in Figure 1, it would choose S_1. If the logical channel is idle, then it transmits for the maximum duration corresponding to

[2]Idle time distribution may be estimated based on the observation of PU traffic for a sufficiently long period.

that channel (y^k_{max}). It then computes the next sensing time such that the duration between the two sensing times is exponentially distributed. If the logical channel is found busy, then it does not transmit, but generates the next sensing schedule for the channel. This process is then repeated, selecting the logical channel with sensing time nearest to the current time and so on. Note that after transmitting for y^k_{max} duration, when the SU sets the next sensing duration, it may possibly have to perform multiple exponential backoffs and apply them cumulatively, because the first backoff duration may not be long enough to go beyond the current time.

IV. Experimental Testbed

We have built an experimental testbed which consists of two servers running centos linux 2.6.32. Each server has 12 core CPUs and 64 GB of memory. Each server is connected to two USRP N210 radios. Each USRP has one SBX daughterboard which can operate in the frequency range between 400MHz to 4400MHz. The four USRPs are connected via coaxial cable to an RF channel emulator which emulates a full mesh topology. Figure 2 shows a picture of our testbed.

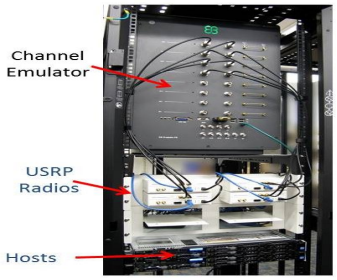

Fig. 2. Experimental Testbed

A. Experiment Setup

Two of the USRPs are set up as PU sender and receiver, whereas the other two act as SU sender and receiver. Bandwidth used for transmission is 1 MHz, which is divided into 512 subcarriers. Out of those 512 subcarriers, the middle 240 are used for communication. This is done to avoid using subcarriers on the edges of the spectrum which are subject to tapering of a lowpass filter applied to the baseband signal. A cyclic prefix of length 128 is used by both the PU and the SU. The carrier frequency is set at 795.5 MHz. The modulation scheme used on each subcarrier is binary phase shift keying. Line of sight propagation with a fixed path loss of 22 dB is used as the channel propagation model for all links (PU-PU, SU-SU, and PU-SU). The PU sender and SU sender transmit with power densities of -92.2 dBm/Hz and -97.7 dBm/Hz respectively.

The 240 utilized subcarriers are equally divided into four logical channels each having 60 consecutive subcarriers. Each logical channel is identified by its *carrier map*. The carrier map of a logical channel is represented by 240 bits in which 60 consecutive bits are 1's and all other bits are 0's. Thus, logical channel 1 has bit position 1 to 60 with 1's and remaining bits 0. Logical channel 2 has bit positions 61 to 120 with 1's and all other bits 0, and so on. Logical channel 1 is set aside as a control channel for the SU's, so it is not used for data transmission either by the PU or SU.

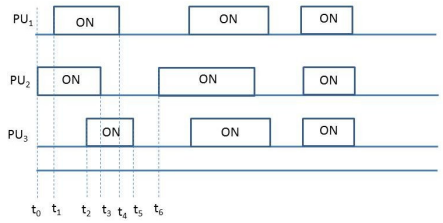

Fig. 3. Emulating three PUs using one physical PU

	tx power density (dBm/Hz)	band width (MHz)	mean ON/OFF period (sec)	carrier freq (MHz)	mean backoff duration	packet size (bytes)
PU	-92.2	1	10/10 5/5 4/4	795.5	N/A	50
SU	-97.7	1	N/A	795.5	y_max	50

TABLE I
Parameter Values used in the experiment

We emulate traffic corresponding to three PU sender-receiver pairs using only one real PU pair as follows. Each of the three PU pairs transmits with alternating ON-OFF periods. Each PU pair is assigned a logical channel (and the corresponding carrier map). We generate timestamps of ON and OFF periods for each of the three PU pairs per their respective exponential distribution parameter. To emulate these three PUs on a single physical PU, we note the time when a transition of traffic ON-OFF period occurs in any of the three PUs. Then in each of the time intervals we note which PUs are ON. The effective carrier map for the real PU is the bitwise OR of the carrier maps of the individual PUs which are ON. This is explained in Figure 3. In the figure we show the first seven transition points labeled t_0 through t_6. Between t_0 and t_1, only PU_2 is ON, hence the carrier map used by the real PU is the same as the carrier map of the logical channel used by PU_2. Between t_1 and t_2, both PU_1 and PU_2 are ON, hence the carrier map used by the real PU pair is the bitwise OR of carrier maps of the logical channels used by PU_1 and PU_2. Note that between t_5 and t_6 none of the PUs are ON and hence the real PU does not transmit in this interval.

In our experiment we assume that the PU sender is saturated, i.e., it always has packets to send. The PU sender follows the scheme described above to transmit packets such that it emulates three PU pairs each having exponential ON-OFF periods. All the parameters used in the experiments are given in Table I. The mean ON/OFF periods shown in the table are for three logical PUs. For the first logical PU, the mean ON and OFF periods are both 10 s, for the second PU they are both 5 s, and for the third they are 4 s. The mean exponential backoff parameter of a given logical channel is set to its y_{max} value.

SUs follow the algorithm described in Section III to use the white spaces opportunistically. We briefly describe how the algorithm is implemented in our testbed. Let us say the SU sender sent a set of data packets using a particular logical (data) channel (say C_k) in the previous burst and that in the next sensing schedule, the SU is supposed to sense a different logical channel (say C_l). If C_l is found idle at the next sensing time, then the SU sender sends the carrier map assigned to C_l to the SU receiver in a REQUEST message, which is sent over channel C_k. The SU receiver, upon receiving this message, sends an ACK (over channel C_k) and then sets its receive carrier map to the value carried in the REQUEST message (which is the carrier map assigned to C_l). After receiving the ACK, the sender sets its carrier map to the carrier map

inter-ference prob. bound (η)	measured inter-ference prob.	PU packet loss rate (%) with 95% confidence interval	PU packet error rate (%) with 95% confidence interval	SU throu-ghput (kb/s)
0.0	N/A	8.08 ± 0.02	4.55 ± 0.01	N/A
0.1	0.1171	14.87 ± 0.03	4.78 ± 0.01	12.91
0.2	0.2125	15.18 ± 0.03	4.61 ± 0.01	12.41
RI0.5	0.4083	17.08 ± 0.03	4.71 ± 0.01	11.33

TABLE II
EXPERIMENTAL RESULTS

assigned to C_l and starts transmitting data packets for y^l_{max} duration.[3] A simple automatic repeat request mechanism has been implemented to make this two-way handshake robust. If the handshake does not happen even after a certain number of retries (three in our experiment) then the SU sender and receiver fall back to the control channel, i.e., future handshake messages are exchanged over the control channel.

One of the main goals of the experiment is to show that using RIBS, it is possible to bound the probability of interference to the PU below a given threshold η. To show this, we ran experiments with four different configurations. In the first configuration, we set η to zero. This implies that there is no SU traffic. This configuration shows the baseline performance of the PU system. In the second and third configuration, η is set to 0.1 and 0.2 respectively. The fourth configuration (denoted as *RI0.5*) represents a naive approach of setting the SU transmission duration to 50% of the mean residual idle time. We want to show that although transmitting for a fraction (0.5, in our experiment) of mean residual idle time may appear to be fine, it yields a relatively high probability of interference. Hence, the fourth configuration helps us to show that there is a need for a more formal method (like RIBS) to provide disruption QoS to the PUs.

B. Experimental Results

The results of our experiment are presented in Table II. It can be seen from the table that the measured probability of interference in RIBS scheme is minimally higher than η, but the violation is practically insignificant. However, performance of the $RI0.5$ scheme is much worse than the RIBS scheme in terms of interference probability. Violation of interference probability constraint in the RIBS scheme can be attributed to a few factors. Our algorithm is implemented at a higher layer. Hence, there is still some latency between issuing a packet send command and the packet actually being sent out from the USRP hardware. Furthermore, while emulating three PU pairs by a single physical pair we had to insert a 50 msec delay between carrier map changes to ensure both sender and receiver have synchronized maps prior to the beginning of a data burst. This delay distorts the PU traffic distribution.

Even with this delay, we still observe a relatively high packet loss rate on the PU link. We define packet *loss* rate as the ratio of packets which are not detected by the receiver to the total number of packets sent. We traced these losses to an inherent problem with the implementation of the OFDM application (*benchmark_tx.py*) in GNU Radio 3.6.5 which loses a few packets at the beginning of every burst. We are currently investigating a solution to this problem. The packet *loss* rate increases when η increases due to higher interference. The SU throughput more or less remains the same across all the configurations. Intuitively, one would expect SU throughput to increase as η increases, but in our experiments, the mean backoff duration is chosen as y_{max}. When η increases, y_{max} also increases (see Eqn (3)), which leads to longer backoff duration. Thus, the number of SU transmission opportunities (in a given duration) decreases as η increases. Hence, the increase in throughput due to larger y_{max} is countered by the lower number of transmission opportunities. As per the parameters of our experiment, the maximum data rate achievable is 93.75 kb/s. Due to strict interefence constraint, the SU transmits conservatively. Also, without any error correction mechanism, a packet is received in error even with a single bit error. These factors lead to somewhat low SU throughput.

[3]To be more precise, sensing duration and the time taken to complete handshaking between the SU sender and receiver are deducted from y^l_{max} to get the actual duration for which SU sender transmits.

V. CONCLUSION AND FUTURE WORK

The work reported in this paper implements an opportunistic spectrum access scheme based on the theory of residual idle time distribution of an alternating renewal process. The main goal of this work is to show that it is possible to provide disruption QoS to the primary user in terms of a bounded interference probability using this scheme as validated by our experimental results on commercially available hardware.

Following are some of the features we envision implementing in the future. Rather than having different sensing schedules for different logical channels, we plan to have a common sensing schedule for all the channels. This will enable us to use all the logical channels which are idle at the time of sensing rather than using just one logical channel, which should improve SU throughput. Furthermore, we want to refine our disruption QoS metric to bound the duration of interference rather than the probability of interference, which may be more appropriate for some applications. Finally, this work generates synthetic PU traffic following an exponential distribution. We next plan to implement RIBS with realistic PU traffic and channel models.

REFERENCES

[1] M. Song, C. Xin, Y. Zhao and X. Cheng, "Dynamic Spectrum Access: From Cognitive Radio to Network Radio," *IEEE Wireless Communication*, vol. 19, no. 1, pp. 23–29., February 2012.

[2] S. Huang, X. Liu and Z. Ding, "Opportunistic Spectrum Access in Cognitive Radio Networks," in *IEEE Infocom*, pp. 2101–2109., April, 2008.

[3] M. Sharma and A. Sahoo, "A Comprehensive Methodology for Opportunistic Spectrum Access based on Residual White Space Distribution," in *Proceedings of the 4th International Conference on Cognitive Radio and Advanced Spectrum Management CogART*, October, 2011.

[4] A. Plummer Jr., M. Taghizadeh and S. Biswas, "Measurement Based Capacity Scavenging via Whitespace Modeling in Wireless Networks," in *IEEE GlobeComm*, pp. 1–7., December, 2009.

[5] K. W. Sung, S. Kim and J. Zander, "Temporal Spectrum Sharing Based on Primary User Activity Prediction," *IEEE Transactions on Wireless Communications*, vol. 9, no. 12, pp. 3848–3855, Decemebr 2010.

[6] J. Nasreddine, J. Riihijarvi, X. Li and P. Mahonen, "Exploring Opportunisitic Access Techniques Using Stochastic Models: Dynamic Spectrum Access Without Sensing," in *IEEE Military Communications Conference*, pp. 1053–1060., November 2011.

[7] Q. Zhao, L. Tong, A. Swami and Y. Chen, "Decentralized Cognitive MAC for Opportunistic Spectrum Access in Ad Hoc Networks: A POMDP Framework," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 2, pp. 589–600, April 2007.

[8] "GNU Radio." http://www.gnuradio.org/.

[9] "Ettus Research." http://www.ettus.com/.

[10] L. Yang, Z. Zhang, W. Hou, B. Y. Zhao and H. Zheng, "Papyrus: A Software Platform for Distributed Dynamic Spectrum Sharing Using SDRs," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 1, pp. 32–37., January 2011.

[11] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. John Wiley and Sons, 2nd ed., 2002.