KNOWS: Kognitiv Networking Over White Spaces

Yuan Yuan, Paramvir Bahl, Ranveer Chandra, Philip A. Chou,
John Ian Ferrell, Thomas Moscibroda, Srihari Narlanka, Yunnan Wu
Microsoft Research, Redmond, USA

Abstract— **The Federal Communications Commission (FCC) has announced that it is willing to consider unlicensed operation in the TV broadcast bands. Compared to the ISM bands, this portion of the spectrum has several desirable properties for robust data communications. However, to make efficient use of this spectrum in a way that is non-disruptive to incumbents, there are a number of challenges that must be handled. For example, an unused portion of the spectrum must be found, and it is likely that its availability will vary over time. To address such challenges, we present KNOWS, a cognitive wireless networking system. KNOWS is a hardware-software platform that includes a spectrum-aware Medium Access Control (MAC) protocol and algorithms to deal with spectrum fragmentation. We describe our prototype and present evaluation results obtained from simulating our MAC protocol. We show that in common scenarios KNOWS accomplishes a remarkable 200% throughput improvement over systems that use a IEEE 802.11 based MAC protocol.**

I. Introduction

Unlicensed bands, such as the 2.4 GHz and the 5 GHz ISM bands, have become crowded in recent years due to the increasing popularity of mobile communications and wireless technologies such as Wi-Fi, Bluetooth and mesh networks. In contrast, a large portion of the licensed bands remain under-utilized or even unused over time [6]. For example, the average utilization of the licensed spectrum for television (TV) broadcast was as low as 14% in 2004 [2]. Based on these observations, the Federal Communications Commission (FCC) recently agreed to evaluate the legal operation of unlicensed devices in "white spaces", i.e., portions of the licensed TV bands that are not in active use by incumbent users, such as the TV broadcasters [1], [16]. This sub-GHz spectrum has several properties that makes it desirable for data communication. In particular, radio frequency (RF) communications can occur over longer distances and RF waves have better penetration property in the lower bands compared to the higher frequency ISM bands.

The use of TV bands by unlicensed systems poses two major challenges. First, the unlicensed systems must not interfere with ongoing TV reception. Therefore, these system must have a robust scheme for determining the white-spaces, and second, these systems must have a spectrum-aware MAC protocol that utilizes white-spaces of varying bandwidths. We present our system, called KNOWS, which detects white-spaces in the TV spectrum through collaborative sensing, and utilizes the available bandwidth even when it is fragmented.

KNOWS enables *opportunistic access and sharing* of white-spaces by adaptively allocating the spectrum among contending users. This is in contrast to existing schemes, such as IEEE 802.11 [7] and IEEE 802.16 [3], which partition the available spectrum into fixed channels, where a channel is the basic unit of spectrum provided to a wireless device. KNOWS does not use the conventional, static "channelization" approach. Instead, it employs a distributed scheme that dynamically adjusts the operating frequency, the occupancy time, and communication bandwidth, based on the instantaneously available white-spaces, the contention intensity, and the user demand. If there are few users in the system, KNOWS provides each user with a larger chunk of the bandwidth. It adaptively provides smaller chunks to all users if there are more contending nodes.

We have built a simple and effective prototype of our system, which consists of a scanner radio (or simply "scanner") and a reconfigurable radio. The scanner periodically searches for the white-spaces in the TV spectrum, and the reconfigurable radio tunes to the white-space and performs data communications. When not scanning, the scanner radio doubles up as a receiver listening for control packets on a fixed control channel. This synergistic design of the radio platform and the MAC protocol enables KNOWS to provide collaborative detection of white-spaces and adaptive spectrum allocation among contending users. To validate our design, we have implemented the KNOWS MAC protocol in QualNet [5]. Our simulation results demonstrate that in most common scenarios, KNOWS increases the system throughput by more than 200% when compared to an IEEE 802.11 based systems.

Our focus in this paper is the KNOWS hardware platform that addresses the important challenges in networking unlicensed devices in the TV bands. Within this context, we make the following three contributions:

- We describe a new hardware implementation that comprises a dual-mode scanner radio for detecting white spaces, and a reconfigurable radio for subsequent data communications. The scanner radio is an integral part of the MAC protocol as it is used for controlling access to the network.
- We introduce a new spectrum allocation scheme, called *b-SMART* that enables users to adaptively adjust the time, frequency, and bandwidth in a fine time-scale. This is in contrast to the widely-adopted spectrum allocation schemes used in IEEE 802.11 [7] and IEEE 802.16 [3], which divides the spectrum into fixed channels. Using simulations, we demonstrate the effectiveness of b-SMART in utilizing the fragmented TV spectrum.
- We present a new control-channel based MAC protocol, called *CMAC* which incorporates "virtual sensing" to arbitrate access to a fragmented spectrum. Specifically,

we enhance the RTS (request-to-send)/CTS (clear-to-send) mechanism of IEEE 802.11, which reserves airtime on a channel, to reserve empty chunks of the spectrum, which we call *resource blocks*. Furthermore, we describe a mechanism that enables all networked devices to maintain up-to-date information about spectrum usage in their neighborhood.

Our paper is organized as follows. In Section II, we present an overview and design of the KNOWS system. In Section III, we describe the implementation details of our prototype. We evaluate the performance of our MAC protocol CMAC and compare it to the IEEE 802.11 MAC protocol in Section IV. In Section V we discuss some of the important issues raised in the paper and in Section VI we discuss related work. Finally, we conclude our paper in Section VII .

II. System Architecture and Design

The goal of the KNOWS system is to enable wireless nodes to self-organize into a network without coordination from a central controller, such that it maximizes the overall spectrum utilization. This goal poses three main challenges that we address in our design.

- *Robust white space detection*: Unlicensed users need a robust way to discover the available white spaces. We note that different bandwidth chunks could be available at the sender and the receiver, therefore the goal is to use a spectrum chunk that is free for both of them.
- *Parallelism and connectivity*: There is a tradeoff between parallelism of flows and connectivity in the network. To enable parallelism, different flows should be active in different chunks of the spectrum. However, this might prevent two nodes (users) that are part of the same network from communicating with each other. One approach to solve this problem is to use schemes that have been proposed by multi-channel MACs [17], [19]. However, these approaches incur extra overhead as described in Section VI.
- *Adaptive bandwidth selection*: The amount of bandwidth assigned to a pair of communicating nodes should depend on the total available spectrum, the contention intensity, and user traffic demand. Intuitively, when there are few users, each user should be assigned a wide bandwidth for a higher data rate; when there are more users in communication range, the total spectrum should be divided to accommodate more concurrent transmissions.

KNOWS addresses the above challenges as follows. First, KNOWS uses a collaborative scanning algorithm to detect incumbent operators in the TV bands. Therefore, only those portions of the spectrum that are detected to be available at all users are used for data communication. To address the second challenge, KNOWS uses a common signalling channel (in the ISM band) to maintain connectivity among nodes, even when they are transmitting or receiving on a different spectrum chunk. Parallelism is ensured by simultaneous data communication on the reconfigurable radio. KNOWS addresses the final challenge by allowing nodes to opportunistically use available spectrum resources by reserving chunks of bandwidth at a fine time-scale. The width of an allocated chunk depends on the amount of available spectrum and the number of contending nodes.

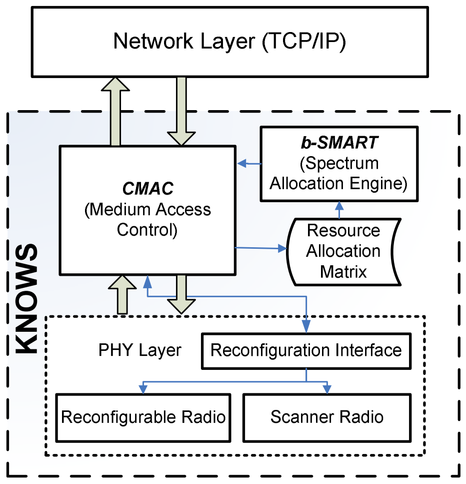


Fig. 1. The components of KNOWS, which includes the hardware, a spectrum allocation engine and a MAC protocol.

We describe the KNOWS system in detail in the rest of this section. We first present our hardware platform and then describe the MAC protocol that enables nodes to reserve portions of the spectrum. We then describe our collaborative scanning algorithm, and finally present b-SMART, a distributed algorithm that determines the amount of bandwidth to allocate to every contending node in the network.

A. System Overview

Figure 1 illustrates the architecture of KNOWS, which includes the hardware, the MAC (CMAC), and the spectrum allocation scheme (b-SMART). The hardware platform includes a dual-mode scanner and a reconfigurable radio. The scanner radio alternates between functioning as a scanner and a receiver. It scans the TV spectrum at least once every 30 minutes, as required by the FCC [16]. The scanner radio in our current platform takes less than 10 ms to scan one 6 MHz TV channel. For most of the time, the scanner radio works as a receiver and is tuned to the 902–928 MHz unlicensed ISM band, which is used as a control channel.

To enable efficient spectrum sharing, each node stores the spectrum usage information in a local data structure, which we call the *resource allocation matrix* (RAM). The spectrum allocation engine in Figure 1 uses the RAM to determine the portion of unused spectrum the node should reserve for its communication.

The RAM records spectrum usage of neighboring unlicensed users in units of *resource blocks*. We define a resource block to be the time duration and the portion of spectrum that is reserved by a node for its communication. Figure 2 depicts one snapshot of resource block allocations stored in

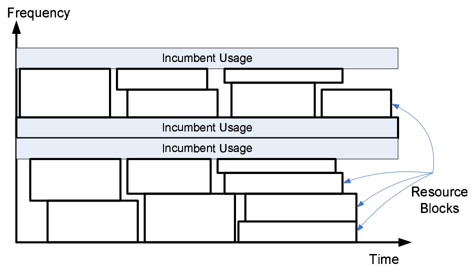


Fig. 2. A snapshot of resource block allocations stored in a Resource Allocation Matrix (RAM)

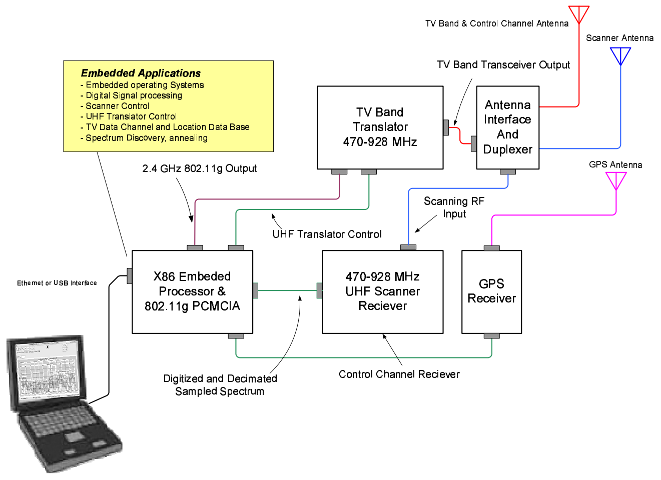

Embedded Applications
- Embedded operating Systems
- Digital Signal processing
- Scanner Control
- UHF Translator Control
- TV Data Channel and Location Data Base
- Spectrum Discovery, annealing
TV Band & Control Channel Antenna
Scanner Antenna
TV Band Transceiver Output
TV Band Translator 470-928 MHz
Antenna Interface And Duplexer
GPS Antenna
2.4 GHz 802.11g Output
Scanning RF Input
UHF Translator Control
Ethernet or USB Interface
X86 Embeded Processor & 802.11g PCMCIA
470-928 MHz UHF Scanner Reciever
GPS Receiver
Digitized and Decimated Sampled Spectrum
Control Channel Reciever

Fig. 3. Cognitive Radio Development Platform of KNOWS

a RAM. The bandwidth and time of the resource block is tuned according to the perceived contention intensity, the total available resources, and the queue length for each neighbor. The reconfigurable radio is then configured to operate in the defined resource block. It switches back to the control channel after the resource block is consumed.

Together with the scanner/receiver, CMAC builds the RAM and implements a reservation-based mechanism that regulates spectrum access. Two communicating nodes first contend for spectrum access on the control channel. Upon winning contention, a handshake is performed, which enables b-SMART at the sender and the receiver to collaboratively agree on a resource block. The reservation is announced on the control channel to inform neighboring nodes. Accordingly, nodes populate their RAM with new reservations, and garbage collect the expired ones.

B. PHY Layer: The Cognitive Radio Development Platform

Figure 3 shows the hardware platform we have built as part of the KNOWS system. The platform consists of four main function blocks, namely the reconfigurable radio, the scanner radio, the GPS receiver, and the x86 embedded processor.

The reconfigurable radio has a set of operational parameters that can be adjusted with low time overhead. The current implementation of the reconfigurable radio uses a commodity IEEE 802.11g card to generate the OFDM signals at 2.4GHz. We use a wide band frequency synthesizer to convert the received signals to the specified frequency. To control the reconfigurable radio, the interface to the MAC layer is a list of register values that specifies the operating frequency, bandwidth, and transmission power level. The operating frequency can be set from 400 to 928 MHz in 0.5 MHz steps, and the bandwidth options currently are 5, 10, 20, and 40 MHz. The narrow bandwidth options, such as 5 MHz, are provided to use white-space spectrum in between the incumbent operators. The maximum output power is 200 mW and the power level is controllable from -8 to +23 (dBm). The threshold for packet reception in the TV band is -85 (dBm). The time overhead for adjusting the radio parameters, e.g. frequency, bandwidth, and power level, is within 100 μs in the current development board.

The scanner periodically scans the spectrum and locates the vacant pockets of spectrum without incumbent signals. The scanning algorithm is prototyped in the C programming language with a Python-based interface and will be implemented on DSPs (digital signal processor) or FPGAs (field programmable gate array) in the future. The scanner measures the signal power at a frequency range with a typical resolution of 3 KHz. On average, the scanner takes at most 10 ms to scan one 6 MHz TV channel. The current setting of the DTV pilot tone detection sensitivity is -115 dBm. As required by the FCC, the TV spectrum needs to be scanned once every 30 minutes, since the TV signal arrives and leaves in a very coarse time level (several hours). Therefore, for most of the time, the scanner works as a receiver operating on the control channel. Our MAC layer can change the scanning schedule and set the frequency range to scan by configuring the registers in the scanner.

Additionally, a GPS receiver is incorporated in the hardware board for loading location information and performing time synchronization. Based on the estimated location, the node could identify the unused spectrum in case a database with TV program information was available. This is an alternative approach suggested by the FCC for detecting incumbent users. Therefore, the GPS receiver extends the flexibility of our development platform.

The x86 embedded processor controls all radios on the platform. It takes instructions from the device driver to configure the radios, and passes packets between the host computer and the development board. The driver implements our MAC design and the spectrum allocation algorithm.

C. CMAC: Spectrum Aware MAC

CMAC implements two main functions: it achieves collaborative sensing by combining scanning results in the one-hop neighborhood, and it realizes a spectrum reservation scheme using a common control channel.

When a node has packets to send it first contends for access to the control channel using CSMA/CA and random backoff mechanisms of IEEE 802.11 [7]. The pair of communicating nodes, upon winning access to the control channel, perform a three-way handshake. During the handshake process, the sender and the receiver exchange their local view of spectrum usage, decide on the spectrum block to use for the communication, and announce the reservation to their neighbors. On receiving a reservation packet, neighboring nodes store the reservation information in their local RAM structure. At the start of the reserved time period, CMAC tunes the reconfigurable radio to the selected spectrum band and initiates the exchange of packets without any backoff.

1) Handshake: CMAC uses a three-way handshake, which builds on IEEE 802.11's two way RTS (request-to-send) and CTS (clear-to-send) handshake. In the handshake process, the senders contend for spectrum access on the control channel using the random backoff mechanism of IEEE 802.11. The winning node sends a modified RTS packet to carry traffic load information and several proposed "resource blocks" to the receiver. A resource block is specified by the frequency interval $(f_0,\ f_0+\Delta f)$ and the time interval $(t_0,\ t_0+\Delta t)$. The regular control packets and our extended versions are shown in Figure 4. The modified RTS packet format incorporates the fields of queue length (1 byte) and average packet size (2 bytes) to describe the traffic load at the sender to the corresponding receiver. It also includes multiple resource blocks, each denoted by four fields: the starting frequency f_0 (1 byte), the bandwidth Δf (1 byte), the start time t_0 (4 bytes), and the duration Δt (2 bytes). The start frequency field records the offset value from the start frequency of the TV spectrum, which is 470 MHz in our system. We use 1 byte to denote frequency and bandwidth; this provides a resolution of 1 MHz. The start time and duration fields provide a timing resolution of one microsecond.

On receiving the RTS packet, the receiver chooses a resource block and informs the sender using a modified CTS packet. The extended CTS packet contains address fields of the sender and the receiver, and details of the selected resource block. We introduce a new control packet, DTS (Data Transmission reServation). The sender uses DTS to announce the spectrum reservation after receiving the CTS packet. CTS and DTS packets have the same format. Every node actively collects CTS and DTS packets to build the RAM, which is a local view of spectrum usage in frequency and time. An entry in the structure corresponds to one reservation, denoted by the source and destination addresses and the resource block. The RAM is updated each time the node receives a new CTS or DTS.

To reduce the time overhead caused by reservation nodes are allowed to make *advanced* reservations. Therefore, the handshakes are conducted in parallel with data transmissions. However, for design simplicity, each node is only allowed to have at most one valid outstanding reservation [12].

Note that the RTS packet can carry more than one resource block to convey more spectrum usage information to the receiver. However, the more information RTS carries, the higher will be the overhead on the control channel, and potentially higher loss rate. Our simulation study shows that using 1–2 resource blocks works best in most cases.

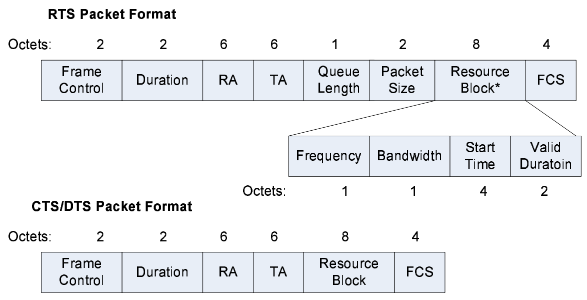


Fig. 4. RTS/CTS/DTS Packet Format

2) Data Transmission: A sender uses the reserved resource block to send data to the intended receiver. When a pair of communicating nodes switch to the selected segment of spectrum, they first perform physical-layer carrier sensing. If the selected spectrum is clear, nodes exchange packets without further back-off. Since the sender has exclusive access to the resource block, it can choose to transmit multiple packets back to back during the defined period.

Note that it is possible that after switching, the sender or the receiver find the selected band to be busy; this may happen for three reasons:

1. The selected band may suffer from interference from transmissions in adjacent frequency bands.
2. The sender or the receiver may experience deep fading in the selected band, and/or
3. Conflicting reservations may occur due to loss of control packets.

If the sender or the receiver senses the selected band to be busy, it gives up the current resource block, and switches back to the control channel. If the other node does not sense the medium to be busy, and is unable to send or receive, it will wait for a pre-defined interval before switching back to the control channel.

3) Collaborative Sensing: Ideally, there should be no communication by unlicensed users in a spectrum chunk when it is being scanned for TV signals. This increases the accuracy of a scan reported by KNOWS. To achive this goal, i.e. prevent a neighbor from sending a packet in the spectrum chunk being scanned, a node reserves the spectrum it wants to scan by sending a DTS packet with this information. Therefore, other nodes do not send a packet in that spectrum chunk, while it is being scanned by another node. During the scanning period, the reconfigurable radio of the node resides on the control channel, and collects control packets until the scanner finishes its task.

CMAC aggregates scanning results from neighbors for better detection of incumbent operators. It extends the beacon frame format used in 802.11 with a bitmap field that carries local scanning results. A bit represents the occupancy status of the corresponding TV channel, with 0 for occupied and 1 for empty. In our case, CMAC can use up to 30 TV channels from channel 21 to channel 51 (except channel 37) [16]. The bitmap field, therefore, requires 4 bytes to reflect the activities in the

UHF spectrum. Each node transmits the beacon message every beacon interval (typically 100–200 ms). A node receiving a beacon message updates the stored scanning results by applying a bit-wise AND operation.

4) Time Synchronization: CMAC performs spectrum allocation via a reservation-based mechanism; hence it is critical to ensure that the nodes have synchronized clocks. The timestamp field in the aforementioned beacon message is used for time synchronization. The timestamp field is 64 bits and offers microsecond time resolution. To synchronize the system clock, upon receiving the beacon, the node records its local time, T_L, extracts the timestamp field, T_B, estimates the transmission time of the beacon, t, and then synchronizes its system clock by adding $(T_B + t) - T_L$. To avoid cyclic synchronization, nodes only synchronize to the faster clock in the system. The accuracy of time synchronization is affected by the interrupt handler and the estimation of propagation delay, which vary across different systems. In our system, nodes make reservations only with their one-hop neighbors. Therefore, we need to focus on the clock skew among one-hop neighbors. As shown in [8], [11], the clock skew among nodes within one hop can be controlled to less than 1 μs using beacon messages.

5) Bootstrap: To join the network, a new node performs the following operations. It first uses the scanner to generate a list of unused TV channels, and at the same time tunes the reconfigurable radio to the control channel, waiting for beacon messages from other nodes for time synchronization. After scanning completes, the node combines the scanning results obtained from beacons, and sends out a beacon frame every beacon interval.

D. b-SMART: Distributed Spectrum Allocation over White Spaces

b-SMART, which is an adaptive resource allocation engine, is a core component of KNOWS. It provides the intelligence for efficiently packing communication in time, frequency, and space. The goal of b-SMART is to maximize the spectrum utilization while being fair to all users.

b-SMART is invoked at a sender node when it is not actively transmitting any packet, and when it senses the control channel to be free. The node first examines the status and the output packet queues for each of its neighbors, to decide whether it should initiate a reservation. A neighbor is considered *eligible* if it does not have an outstanding reservation, and the output packet queue for this neighbor has accumulated enough packets to amortize the control overhead, or the queue has timed out for packet aggregation (to avoid excessive delay). In order to ensure that the control channel does not become a bottleneck, our protocol ensures that (unless a queue times out), every transmission has a minimum duration length when the smallest possible bandwidth is chosen. We denote this minimum transmission duration by T_{min} and explain our choice for this parameter later in this section. To maintain fairness among neighbors we currently implement a round-robin scheduler to elect an eligible receiver.

The node proposes several candidate resource blocks to the receiver for reservation; such decisions are made based on its packet queue, the spectrum usage, and the perceived contention intensity in its vicinity. This decision triggers CMAC to initiate a handshake procedure. b-SMART at the receiver updates and finalizes the resource block parameters - frequency, bandwidth, start time, duration - based on its resource allocation matrix.

As mentioned earlier, a resource block is specified using four parameters: the starting frequency f_0, the bandwidth Δf, the starting time t_0, and the time duration Δt. Intuitively, this can be viewed as a rectangular block in the time–frequency space. The proposed method first decides the size of the block, as specified by Δt and Δf, and then decides how to place the block in the time–frequency space. A simple placement strategy is adopted in our current implementation: The block is placed at a position that results in the smallest finishing time, $t_0 + \Delta t$; ties are broken randomly.

We now focus on the problem of deciding the shape of the block, $(\Delta t, \Delta f)$. Two key guidelines are used in making this decision:

(1) We require Δt to be large enough to amortize the incurred overhead in the control channel for reservation, such that the control channel does not become the performance bottleneck; that is, $\Delta t \geq T_{min}$. The exact formula for T_{min} will be described shortly afterwards.

(2) We want Δf to be large enough to achieve a high data-rate, but in order to maintain fairness, it should not exceed B/N by too much, where B is the total available bandwidth and N is the number of concurrent transmissions in the interference range.

Determining the value of Δf poses an interesting challenge. It is fairly intuitive that when there are only few potential concurrent transmissions, each one should use a large bandwidth to avoid wasting the resource. However, when there are a number of users, it might not be as straightforward as to why we want to divide the spectrum among all concurrent users, instead of allocating the total spectrum to only one user at a time. We provide three justifications for dividing the spectrum into segments of smaller bandwidth when the number of users is large.

First, a smaller bandwidth decreases the physical-layer data rate and increases the data transmission duration. The overall spectrum efficiency, however, is increased due to the reduced percentage of signaling overhead associated with every data transmission. For example, suppose there is a fixed 100 μs overhead associated with each transmission and the actual data transmission takes 200 μs in a 1 MHz band and 100 μs in a 2 MHz band. Then two parallel transmissions in two 1 MHz bands yields an efficiency of 67%, whereas a transmission in one 2 MHz band yields an efficiency of 50%. Second, the fine control of bandwidth provides an adequate number of channels and offers high accessibility to the spectrum. The dynamic spectrum allocation, therefore, achieves better delay and jitter performance, which is preferable for TCP traffic. Finally, the reduced bandwidth allows more parallel transmissions, which reduces contention in the control channel.

We now show how we apply the two rules to determine $(\Delta t, \Delta f)$. Due to hardware limitations, the reconfigurable radio can only work with a limited number of bandwidth options, say, $b_1 < b_2 < \ldots < b_n$. Hence Δf has to be within this set of finite choices. We first examine the bandwidth option just exceeding B/N, say b_i. Then we check the length of the queue to the receiving node and estimate the transmission duration Δt when using a bandwidth of b_i for communication. If the resulting transmission duration Δt is at least T_{min}, then this pair $(\Delta t, \Delta f)$ is determined. Otherwise, b-SMART checks the next smaller bandwidth option, and so on. Note that b-SMART can apply the iteration above to generate multiple candidates of $(\Delta t, \Delta f)$ at the sender.

1) Setting T_{min}: We introduce

$$C_{max} \triangleq \frac{B}{b_1}, \tag{1}$$

where b_1 is the smallest bandwidth supported by the reconfigurable radio. The quantity C_{max} is the maximum number of parallel transmissions. We derive T_{min} as follows:

$$T_{min} = C_{max} \cdot T_o = \frac{B}{b_1} \cdot T_o, \tag{2}$$

where T_o is the time required for handshake on the control channel. From this, it is seen that supporting a wider data spectrum requires a larger T_{min}.

The reason for our setting T_{min} as shown in Equation (2) is the following. The maximum rate R_l, at which reservations are generated, is $1/T_o$. R_l is also the rate, at which the nodes leave the control channel and start transmissions. $1/T_{min}$ is the rate, at which a pair of nodes finishes transmissions and returns to the control channel. If the spectrum is fully utilized, there is a total of C_{max} parallel transmissions. Hence the returning rate R_r is C_{max}/T_{min}. To prevent the control channel from becoming a bottleneck, the rate of generating handshakes, R_l, should exceed the returning rate, R_r.

2) A low-complexity method for estimating N, the number of disjoint transmissions: As mentioned earlier, the proposed adaptive resource allocation methods hinges critically upon a parameter, N, which is the number of disjoint transmissions. This quantity depends on network topologies and user traffic patterns, and varies over time. The computation cost, however, has to be minimal, since such computation is repeated in a fine time scale (10s of ms).

We propose to use the number of valid resource blocks in the local resource allocation matrix, as an approximation of N. Note that in our system no user is allowed to have more than one outstanding reservation. A reservation that has not expired means that either the corresponding nodes are currently transmitting or waiting for their reserved block's start time to arrive. Such approximation can effectively track the number of disjoint transmissions especially when the flows are long-lived and have enough packets.

This approximation method is distributed and easy to compute. Furthermore, the method is responsive to user and traffic dynamics. For example, if a transmission created by two new nodes becomes active, other nodes learn the transmission after the corresponding nodes finish the first handshake.

Initially, when new nodes join the KNOWS system, it takes them a time period, which we call the "learning period", to learn the existence of contending transmissions. The length of the learning period depends on the number of new nodes and traffic patterns. We quantify the value of the learning period using simulations in Section IV.

3) Fairness: Since the reservations contend in the control channel via a mechanism adopted from 802.11, KNOWS preserves long-time access fairness properties similar to the ones of IEEE 802.11. Thus, for scenarios in which all nodes are within radio range of each other, KNOWS can provide flows with the same access right to the control channel as long as the number of transmissions does not exceed C_{max}. As a result, all flows will roughly receive a fair share of resource blocks. In case of $N > C_{max}$, KNOWS may suffer from a known fairness problem of the back-off protocol more than IEEE 802.11. A node that has just finished transmitting has a higher chance of recapturing the channel than other nodes, because its back-off window is smaller and hence, its probability of sending an RTS is higher. While in a single-channel IEEE 802.11 environment, there is always only one such favored transmission, there can be up to C_{max} many of them in the KNOWS system, which can aggravate the fairness problem if N becomes large.

E. Summary

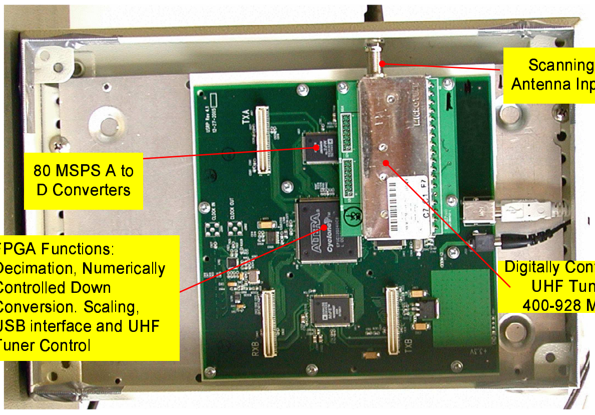
The KNOWS system consists of the hardware platform, CMAC and b-SMART, a spectrum allocation engine. Nodes constantly maintain the current spectrum usage information in a RAM. The RAM is used by b-SMART to adaptively assign resource blocks to each node in the network. CMAC conveys the reservation to the neighbors of the sender and the receiver.

III. Implementation Status

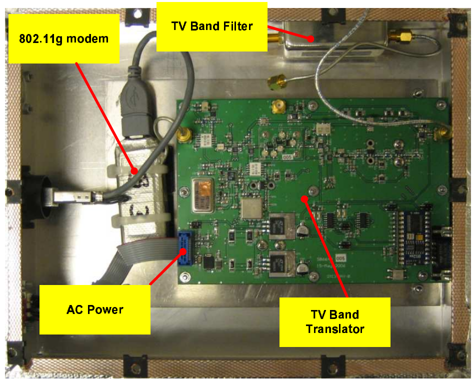
We are building a prototype of the KNOWS system, which we plan to deploy for conducting field experiments. We have currently implemented the radio design in the development boards, and conducted experiments to examine the functionality of the scanner in the San Diego area.

A. Hardware

Figure 5(a) and Figure 5(b) show the development boards for the scanner radio and the reconfigurable radio respectively. The interface to the host computer is USB. Figure 6 shows one sample output of the scanning operations conducted in the San Diego area. The scanner measures the signal strength of the spectrum. Figure 6(a) shows one detected DTV pilot tone and the peak, average and minimal signal strength in the sliding 6 MHz-wide window. Figure 6(b) illustrates the aggregated information for the whole TV bands using the accumulated peak and average values. We save all white spaces that are larger than 1 MHz and display white windows with at least 5 MHz bandwidth. Our measurement results confirm that the

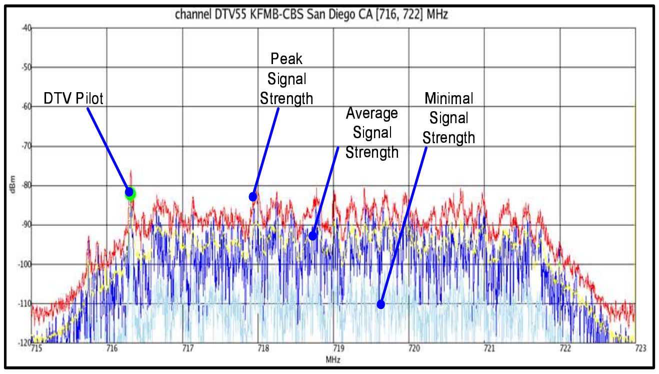

Scanning Antenna Input
80 MSPS A to D Converters
FPGA Functions: Decimation, Numerically Controlled Down Conversion. Scaling, USB interface and UHF Tuner Control
Digitally Controlled UHF Tuner 400-928 MHz

(a) Scanner Receiver Development Board

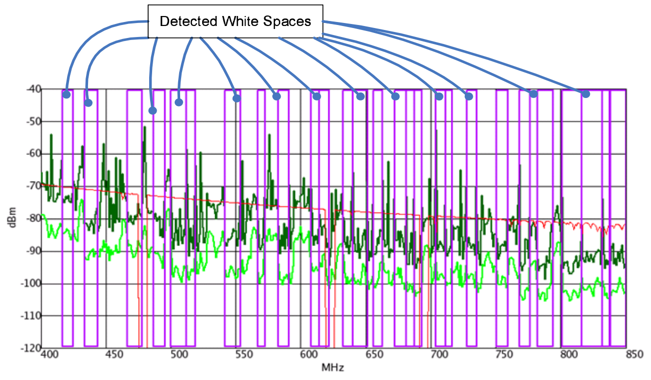


(b) Reconfigurable Radio Development Board

Fig. 5. KNOWS Radio System Development Board



(a) Scanning Results in the 6MHz-wide Window



(b) Scanning Results of the UHF Spectrum

Fig. 6. Sample Scanning Results in the San Diego Area

TV spectrum is fragmented especially in the metropolitan areas, while, in rural areas, more contiguous spectrum is available [6].

B. MAC Protocol

We have taken several steps to implement CMAC and b-SMART. First, for each chip on the board, we develop the HAL (hardware abstract layer), which provides the MAC with handles to configure the radio parameters and to control the operations of the development board. The low-layer functions, such as the scanning algorithm, are loaded in the on-board x86 processor, which runs an embedded operating system. Second, we develop a device driver to implement the control logic in CMAC, for example, 3–way handshake, data transmission, time synchronization. We plan to modify the packet format accordingly and reuse some of the source code for implementing the 802.11 MAC. Third, to avoid modifications to the network stack, CMAC maintains a queue buffer for each neighbor, and one separate queue for broadcast packets. The address of the neighbor is obtained by peeking into the IP header. The broadcast packets are sent on the control channel with a higher priority. Finally, the MAC module constructs the RAM for b-SMART to ascertain the spectrum usage information and the availability of the destination node.

IV. System Evaluation

We implement the KNOWS system in QualNet and evaluate its performance in three phases. First, we microbenchmark the throughput performance of KNOWS. We validate the formula to calculate T_{min}, and then evaluate the effectiveness of the adaptive spectrum allocation algorithm in b-SMART using both TCP and UDP flows. Second, we study the throughput performance of KNOWS in single-hop networks, and compare it against SSCH [17], which is a recently proposed multichannel MAC. Finally, we study KNOWS in more complicated scenarios, such as in multi-hop networks, and with mobility.

We modified QualNet to handle variable frequencies and bandwidths. Note that for the same physical-layer encoding scheme, the data rate is proportional to the allocated bandwidth [9]. In all our experiments, we use the bandwidth model that delivers 1.2 Mbps for each 1 MHz. The bandwidth options provided by our radio prototypes are 5 MHz, 10 MHz, 20

MHz and 40 MHz. Further, we incorporate 100 μs as the time to change the frequency, the bandwidth or the power level. We derived this value from measurements taken on our radio development board. We also set the control channel to be 5 MHz wide, which corresponds to a data rate of 6 Mbps.

A. Design Parameters

We first verify the formula for T_{min} that a backlogged client should take to transmit data packets. We place all nodes within the communication range of each other and set up disjoint UDP flows between them. Two flows are considered disjoint if they do not share either endpoint.

1) T_{min}: As described in Section II, $T_{min} = C_{max} \cdot T_o$, where T_o is the time overhead of the handshake procedure on the control channel. C_{max} denotes the maximal number of parallel transmissions on the available spectrum of the total bandwidth B. Note that $C_{max} = B/b_1$ and b_1 is 5 MHz, the smallest bandwidth offered by the reconfigurable radios. We validate the formula in Figures 7(a) and 7(b).

In the first experiment, we increase C_{max}, which corresponds to increasing the amount of available spectrum. For a given C_{max}, we simulate C_{max} flows, each carrying 6 MHz traffic to saturate one 5 MHz spectrum segment. Ideally, as we increase C_{max}, more spectrum resource is added to the system and KNOWS should deliver proportionally higher throughput.

Note that if T_{min} is set to $20ms$, the system throughput increases proportionally as C_{max} increases from 5 to 55. In this case, KNOWS is able to fully utilize the available spectrum. When T_{min} is set to 5 ms, the system throughput does not increase further after C_{max} grows over 14. This occurs because the control channel becomes the bottleneck and it fails to generate enough resource blocks to consume the available spectrum. We refer to the C_{max} at 14 as the *saturation point* C_{sat}. The T_{min} settings of 10 ms and 15 ms show a similar trend, while supporting larger value of C_{sat}, which is 25 and 36 respectively. This clearly indicates that T_{min} determines the overall bandwidth KNOWS supports.

In the next experiment, we validate our formula by plotting T_{min}/T_o versus C_{sat} for over 45 different scenarios in Figure 7(b). As predicted by our formula, both these values are nearly equal in all the scenarios. Hence $C_{sat} = T_{min}/T_o$. KNOWS sets T_{min} to support the entire available spectrum, that is $C_{sat} = C_{max}$. Based on the scanning results and b_1, the value of T_{min} can be derived.

2) b-SMART's Spectrum Allocation Algorithm: We now analyze the throughput achieved by b-SMART's adaptive spectrum allocation scheme compared to fixed spectrum allocation. We also quantify the duration of time taken by nodes to adapt to changes in spectrum usage, which we call the learning period. We consider an 80 MHz-wide contiguous spectrum. With the bandwidth options provided by our radio platform, the spectrum can be used as 2 segments of 40 MHz, 4 segments of 20 MHz, 8 segments of 10 MHz, or 16 segments of 5 MHz. We run CMAC with these fixed allocations, and compare the performance to KNOWS with the adaptive spectrum allocation.

Figure 8 shows the system throughput of UDP and TCP traffic as the number of flows, N, varies from 1 to 22. The flows are disjoint and backlogged. All nodes are within communication range of each another. The results show that any fixed allocation scheme *cannot* suit all flow populations. On the other hand, b-SMART effectively tracks the best options among the possible fixed allocations.

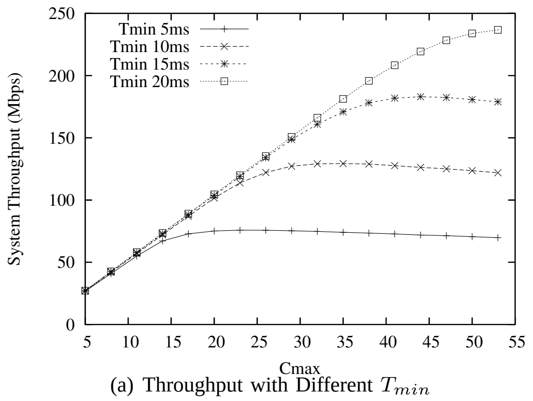
Specifically, when N is small, the allocation that offers larger bandwidth achieves higher throughput because the overall spectrum is better utilized. For example, when there is only one flow in KNOWS, the fixed allocation with 40 MHz segments achieves the best performance. As the number of flows increases over 16, fixed allocation with 5 MHz bandwidth achieves the highest throughput, which is approximately 21% and 40% greater than the one with 40 MHz using UDP and TCP flows, respectively. The key reason is that with enough flows, the smaller bandwidth, corresponding to the lower data rate, prolongs the data transmission duration and thus reduces the ratio of signaling overhead (e.g. inter-frame spaces, and acknowledgements) in each spectrum segment. Consequently, the overall efficiency is improved. Moveover, the smaller bandwidth options offer the flexibility for coping with fragmentation in the licensed spectrum and exploiting isolated small segments in the spectrum.

Note that with TCP flows, the proposed spectrum allocation algorithm is not exactly accurate at the bandwidth transition point, such as when the number of flows is 4 or 8. The reason stems from the property of TCP traffic, which introduces short ACK packets. Our approximation counts all valid reservations, including those for TCP ACKs. The algorithm makes a conservative decision on the bandwidth, consequently reduces the spectrum utilization in line with the system throughput. However, since the duration of spectrum usage for ACKs is rather small, the approximation is effective in most cases.

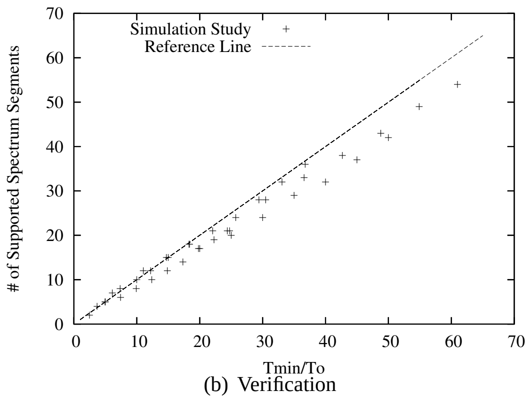
Figure 9 quantifies the learning period of KNOWS, which is the time taken by nodes to recognize one another and adjust their bandwidth accordingly. The duration of the learning period depends on the number of joining nodes, and the traffic type. In general, with backlogged UDP flows, the learning period is short and it takes less than 10 ms to accommodate 20 flows. It takes longer to learn TCP flows, approximately 80 ms for learning 20 flows. The reason is that the handshake protocol and the slow start mechanism used by TCP generates a small amount of packets when the flows initiate transmissions. To learn the contention intensity, our approximation requires flows to use the resource block with T_{min} duration. This usually happens after slow-start and when the congestion window of TCP increases beyond a threshold. As a result, KNOWS takes a longer period to discover contending TCP flows.

B. Benchmarking Throughput Improvements

We now quantify the throughput achieved by KNOWS. We place all nodes in communication range of each other, and set the flows to be always backlogged. The total vacant spectrum is set to 80 MHz wide, which is approximately half of the

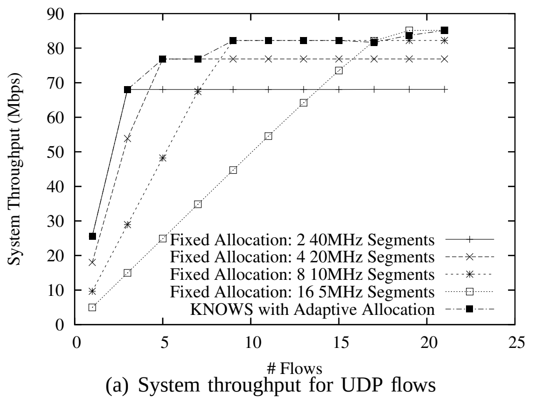


(a) Throughput with Different T_{min}

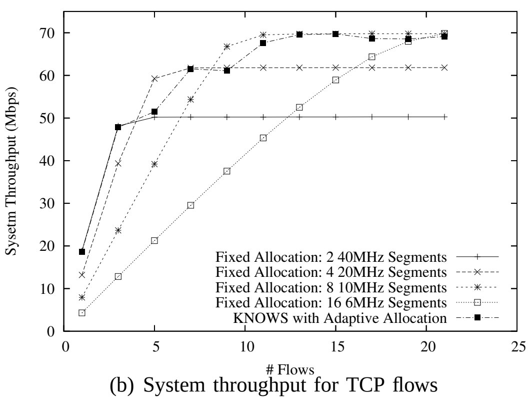


(b) Verification

Fig. 7. $T_{min} = C_{max} * T_o$

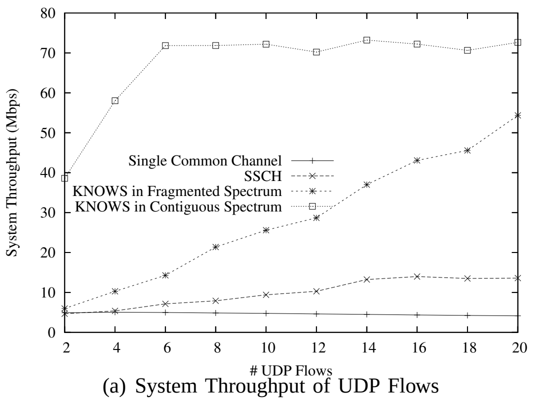


(a) System throughput for UDP flows

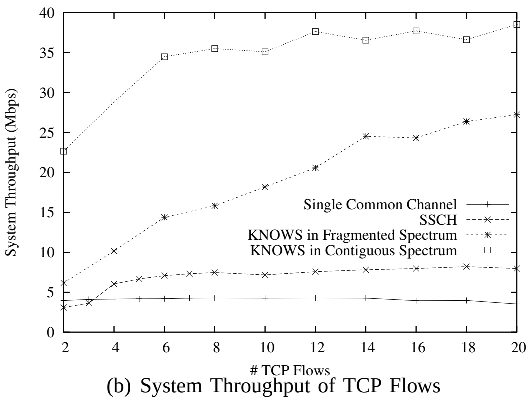


(b) System throughput for TCP flows

Fig. 8. Total throughput of a system using KNOWS for UDP and TCP flows.



(a) System Throughput of UDP Flows



(b) System Throughput of TCP Flows

Fig. 11. Throughput Performance with Non-Disjoint Flows

entire UHF spectrum. We run KNOWS in both fragmented and contiguous spectrum. In the first case, the spectrum is fragmented by incumbent TV signals and all the vacant bands are one TV channel wide, i.e. 6 MHz. The contiguous case offers 80 MHz spectrum without any overlapping incumbent operation. For comparison, we simulate SSCH, which is designed to utilize multiple pre-defined channels with *one* transceiver.[1] SSCH assigns a pseudo random seed to each node. The node switches across channels based on the hopping sequence generated by the random seeds. By synchronizing one or more random seeds, SSCH ensures two nodes to meet on certain channel(s) and exchange packets. In this case, nodes in SSCH hop across the vacant TV channels. As a reference,

[1]We do not know of any MAC protocol that is designed for our radio model of one transceiver and one receiver.

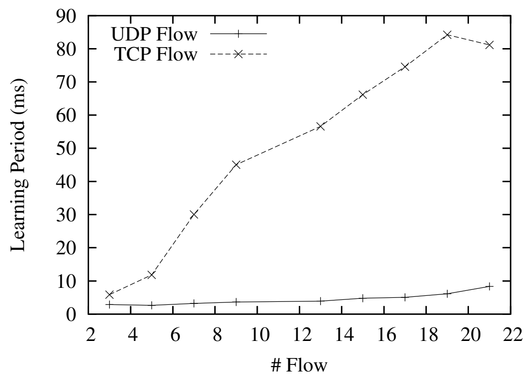


Fig. 9. Time for KNOWS to learn of the neioghbors' spectrum usage and traffic patterns

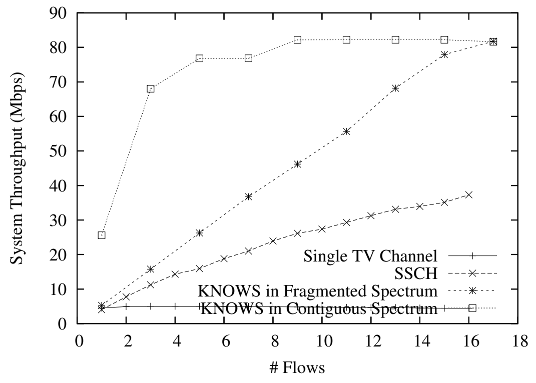


Fig. 10. System Throughput with Disjoint UDP Flows

we also run 802.11 MAC in one common TV channel.

1) Disjoint UDP Flows: We first study the throughput as the number disjoint UDP flows increases from 1 to 16. Figure 10 shows the system throughput. KNOWS utilizes all vacant TV channels in the fragmented spectrum when increasing the number of flows in the system. Within contiguous spectrum, nodes in KNOWS adjust their bandwidth based on the experienced contention intensity. For example, when there is only one flow in the system, KNOWS assigns it the maximum bandwidth offered by the cognitive radio, i.e. 40 MHz. As we increase the number of flows in the network, each flow uses a smaller bandwidth.

As shown in the figure, KNOWS achieves much higher throughput than SSCH. There are two primary reasons for the increased throughput. Firstly, the adaptive spectrum allocation enables nodes to tune the bandwidth based on the number of disjoint flows. On the other hand, any MAC design using fixed channels cannot adjust the bandwidth to opportunistically exploit the contiguous spectrum. Secondly, KNOWS leverages the extra receiver to perform more optimal spectrum scheduling than the randomized scheduling used by SSCH.

2) Non-Disjoint Flows: We now benchmark the throughput on increasing the number of non-disjoint flows, i.e. flows that may share the same sender or receiver. Figure 11(a) and Figure 11(b) shows the aggregate throughput of UDP and TCP traffic respectively, as the number of flows increases from 2 to 20. The source and destination is chosen randomly for each flow.

In case of contiguous spectrum, the system throughput quickly reaches the maximum throughput. This is because the adaptive spectrum allocations effectively manages the bandwidth across contending nodes. When the spectrum is fragmented, KNOWS takes more time to fully utilize the available bandwidth. This can be explained by senders or receivers shared by different flows, which in turn reduces the possible number of parallel transmissions. However, note that KNOWS still outperforms SSCH because of the reasons described earlier.

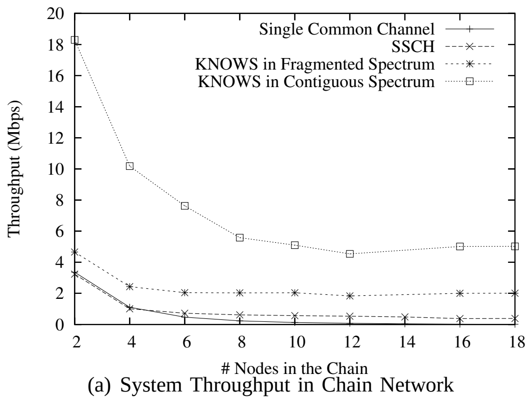
C. Multiple-hop Networks and Mobility

We now evaluate the performance of KNOWS in multi-hop networks. We first quantify the throughput improvements in a multi-hop chain network. We then consider a large-scale mesh network and compare KNOWS using fixed bandwidth allocations with using the adaptive spectrum allocation scheme. Finally, we present the impact of mobility on the performance of KNOWS in a multi-hop ad-hoc network.

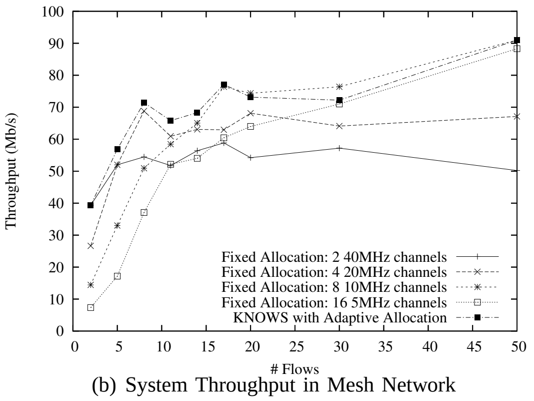
1) Chain Network: We set up a chain network where packets originate at the first node and are forwarded to the last node. To explore the ability of KNOWS in coordinating spectrum usage, we place all nodes in communication range of each other. Figure 12(a) plots the system throughput of KNOWS in the fragmented and contiguous spectrum on increasing the number of nodes in the chain from 2 to 18. KNOWS in the contiguous spectrum achieves significantly higher throughput than the fragmented case. The adaptive spectrum allocation takes effect with the contiguous spectrum, and grants bandwidth to nodes based on the instantaneous number of parallel transmissions required to opportunistically exploit the spectrum. KNOWS in a fragmented spectrum obtains much higher throughout compared to SSCH as the number of nodes in the chain increases, since KNOWS makes better spectrum allocation and scheduling decisions.

2) Mesh Network: We consider a multihop network with 100 nodes randomly placed in a 500 m x 500 m square. The nodes form a mesh network and transmit packets at 10 dbm. We randomly select source-destination pairs, and use DSR [10] to discover routes. Figure 12(b) shows the system throughput of KNOWS on increasing the number of flows in the network for different spectrum allocation schemes. Figure 8 shows that the performance in a mesh network follows a similar trend as in the single hop case. Generally, the allocation with larger bandwidths delivers high performance when there is low contention. With more contending nodes, the allocation with smaller bandwidths creates more parallel transmissions, and thus attains higher throughput. We also measured the impact of KNOWS on existing routing protocols. The average number of hops discovered by DSR using KNOWS is close to that in 802.11 MAC with the single TV channel. The hop count is 2.6 on average in this case.

3) Mobility Effect: We now study the impact of the mobility on KNOWS. We set the multi-hop network similarly but allow



(a) System Throughput in Chain Network



(b) System Throughput in Mesh Network

Fig. 12. Throughput Performance in Mutihop Network

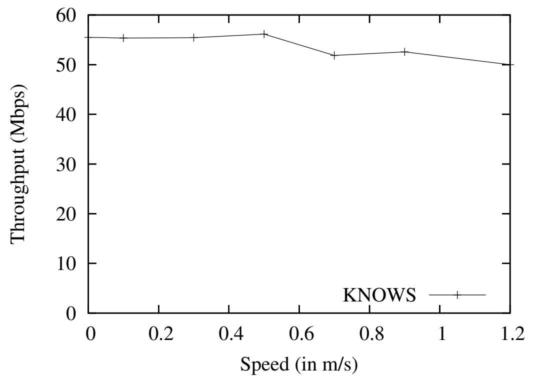


Fig. 13. Impact of Mobility on Throughput

nodes to move using the Random Waypoint Model. In this model, each node selects a random point, and moves towards it with a speed chosen randomly from an interval, $(V_{min}, V_{max}]$. Upon reaching its destination, the node moves to a new destination after it pauses for a random period between 0 and 10 seconds. We set V_{min} at 0.01 m/s and vary the V_{max} from 0.2 to 1.2 m/s. Figure 13 shows the system throughput of KNOWS with 10 flows when run for over 6 minutes. KNOWS experiences minor throughput degradation due to route maintenances and conflicts in spectrum reservations. However, the impact is not significant, even with a reasonably high mobility of 1.2 m/s.

V. Discussion

We discuss some design choices made for KNOWS and some directions for future work.

In this paper, we propose a new scheme for adaptive spectrum allocation. This scheme is different from the conventional method of spectrum allocation, which divides the available spectrum into fixed channels of equal bandwidth. For example, in IEEE 802.11a, there are 13 orthogonal channels of 20 MHz bandwidth. This fixed channelization structure is simple and incurs low implementation cost. However, such a structure creates hard boundaries for utilizing the entire spectrum. One implication is that it prevents users from bundling vacant channels to obtain higher data rates. Moreover, in the TV spectrum, the spectrum is fragmented by the incumbent signals, leaving various sizes of spectrum segments available for sharing. The adaptive spectrum allocation adopted by KNOWS deviates from this channel concept. The operating frequency and the bandwidth is adaptively determined based on local information.

We use a narrow-band control channel for disseminating spectrum usage information. In contrast to systems that use a central spectrum controller with global knowledge of user activities and spectrum allocations, KNOWS uses a distributed approach for efficient spectrum sharing. Each node constantly listens on the control channel to keep track of spectrum availability in real time. In recent work [12], we have explored the tradeoffs involved in separating control traffic from data, and use the results to set the control channel bandwidth to be 5 MHz. Our current design uses a 5 MHz band in the unlicensed ISM spectrum (902–928 MHz) as the control channel.

We note that using one fixed control channel raises security concerns. The nodes in KNOWS cannot operate in the TV spectrum if the control channel is occupied or jammed. To improve the robustness, we are investigating the use of a common hopping sequence to build the control channel. The control channel can hop across the vacant TV channels according to a negotiated sequence at a coarse-time level (several seconds). Hence, the single point of failure caused by using a single control channel can be largely reduced. In addition, the control channel is different from the frequency band used for data communications. Different bands may have different propagation proprieties, especially in terms of the transmission range. We are conducting experimental studies using our prototype radios to quantify the effect of transmission range mismatch. We expect our results to be consistent with prior work [12].

VI. Related Work

We summarize and compare prior work relevant to KNOWS mainly from the spectrum sharing perspective, which defines how the vacant spectrum should be shared among unlicensed users.

There are two different approaches for supporting spectrum sharing: centralized control and distributed coordination. In the centralized control category, IEEE 802.22 [4] is the first standardization effort to define unlicensed operations in the TV spectrum. In 802.22, a base station serves multiple Consumer Premise Equipments (CPEs) and determines the availability of a TV channel by combining scanning results from the CPEs. The base stations are allowed to combine three contiguous TV channels to generate an 18 MHz-wide operating band. Two other centralized systems are DIMSUMnet [13] and DSAP [20]. In DIMSUMnet, the spectrum brokers coordinate spectrum usage in relatively large geographic region; in DSAP [20], the centralized controller manages the spectrum access by offering long-term leases to secondary users. In contrast to the above systems, KNOWS is based on distributed coordination, and is not lease based.

Within the distributed category, several MAC protocols have been proposed to utilize the overall spectrum. However, to the best of our knowledge, all of them are based on static, evenly-divided channels. For example, SSCH [17], MMAC [19], and LCM-MAC [15] use a single radio to exploit multiple fixed channels. DCA [21], xRDT [15], HMCP [18] are proposed to use multiple channels in parallel with multiple radios. The existing MAC solutions assume a fixed channel as the default spectrum allocation unit. However, channels are not well defined in the TV bands due to the dynamic nature of white spaces. Should the bandwidth be the size of a TV channel or should it be smaller or larger? Where should we set the center frequency? These questions have motivated KNOWS to reconsider the essence of spectrum allocation. In our system, nodes adaptively utilize different frequencies and bandwidth based on spectrum availability and contention in the network.

Several MAC proposals have addressed different issues in cognitive radio networks. HD-MAC [22] maintains connectivity in a large network using a set of control channels, where each control channel manages a different local group. This is in contrast to using a global control channel. Coordination between local groups merges different groups into a connected network. In contrast, KNOWS uses a narrow channel in the unlicensed band as the common control channel. DC-MAC [23] conducts a theoretical study to derive decentralized strategies for unlicensed users to sense and access fixed channels. DOSS [14] allows nodes to use a variable bandwidth channel based only on the spectrum availability. In comparison, KNOWS provides a detailed algorithm to adapt the bandwidth at a fine-time-scale and decides the time duration considering the traffic load information.

VII. Conclusion

We have presented KNOWS, which is a sytem encompassing new hardware, an enhanced MAC protocol and spectrum sensing capabilities, for efficiently utilizing unused portions of the licensed spectrum for unlicensed operations. KNOWS co-operatively detects incumbent operators and efficiently shares the vacant spectrum among unlicensed users. Our hardware consists of a development board with a scanner/receiver radio and a reconfigurable transceiver. b-SMART maintains up to date information about the spectrum usage of all its neighbors, and stores it in a RAM. CMAC uses the RAM to dynamically decide on the portion of the spectrum to use for a given communication. CMAC also enables a spectrum reservation scheme in addition to the virtual sensing approach of IEEE 802.11. Using simulations, we have shown that KNOWS significantly increases the capacity when compared to IEEE 802.11 based systems.

References

[1] FCC, Unlicensed Operation in the TV Broadcast Bands Notice of Proposed Rulemaking (NPRM), ET Docket No. 04-186, May, 2004.

[2] Federal Communications Commission (FCC), www.fcc.gov.

[3] IEEE 802.16 WG on Broadband Wireless Access Standards, grouper.ieee.org/groups/802/16/.

[4] IEEE 802.22 WRAN WG, www.ieee802.org/22/.

[5] QualNet, http://www.qualnet.com/.

[6] Shared Spectrum Company, www.sharedspectrum.com.

[7] IEEE 802.11b/D3.0, Wireless LAN Medium Access Control(MAC) and Physical (PHY) Layer Specification: High Speed Physical Layer Extensions in the 2.4 GHz Band, 1999.

[8] J. Elson and D. Estrin. Time Synchronization for Wireless Sensor Networks. In *IEEE IPDPS 2001*.

[9] J. Proakis, Digital Communications, McGraw Hill, 2001.

[10] D. Johnson, D. Maltz, and J. Broch. DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks. In C.E. Perkins, editor, *Ad Hoc Networking*, chapter 5, pages 139–172. Addison-Wesley, 2001.

[11] K. Romer. Time Synchronization in Ad Hoc Networks. In *ACM MobiHoc 2001*.

[12] P. Kyasanur, J. Padhye, and P. Bahl. On the Efficacy of Separating Control and Data Into Different Frequency Bands. In *BROADNETS*, pages 646–655, 2005.

[13] M. M. Buddhikot, P. Kolodzy, S. Miller, K. Ryan, and J. Evans . DDIMSUMNet: New directions in wireless networking using coordinated dynamic spectrum access. In *IEEE WoWMoM05, June 2005*.

[14] L. Ma, X. Han, and C. Shen. Dynamic Open Spectrum Sharing MAC Protocol for Wireless Ad Hoc Networks. In *Dyspan 2005*.

[15] R. Maheshwari, H. Gupta, and S. Das. MAC Protocol for Multiple Channels. In *IEEE SECON 2006*.

[16] Office of Engineering and Technology. Projected Schedule for Proceeding on Unlicensed Operation in the TV Broadcast Bands, 2006.

[17] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks. In *ACM MobiCom 2004*.

[18] P. Kyasanur, J. So, C. Chereddi, and N. H. Vaidya. Multi-Channel Mesh Networks: Challenges and Protocols. In *IEEE Wireless Communications, April 2006*.

[19] J. So and N. H. Vaidya. Multi-Channel MAC for Ad Hoc Networks: Handling Multi-Channel Hidden Terminals Using a Single Transceiver. In *ACM MobiHoc 2004*.

[20] V. Brik, E. Rozner, S. Banerjee, and P. Bahl. DSAP: A Protocol for Coordinated Spectrum Access. In *IEEE Dyspan 2005*.

[21] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu. A New Multi-Channel MAC Protocol with On-Demand Channel Assignment for Mobile Ad Hoc Networks. In *International Symposium on Parallel Architectures, Algorithms and Networks (I-SPAN) 2000*.

[22] J. Zhao, H. Zheng, and G. Yang. Distributed Coordination in Dynamic Spectrum Allocation Networks. In *Dyspan 2005*.

[23] Q. Zhao, L. Tong, and A. Swami. Decentralized Cognitive MAC for Dynamic Spectrum Access. In *Dyspan 2005*.