Multi-Channel Spectrum-Agile MAC Protocol with Adaptive Load Control

Fan Wang and Marwan Krunz
Department of Electrical & Computer Engineering
The University of Arizona, Tucson, AZ 85721
E-mail:{wangfan,krunz}@ece.arizona.edu

***Abstract*— Spectrum-agile radios, also known as cognitive radios, have a great potential to improve spectrum utilization by enabling dynamic access to the spectrum. A key challenge in operating these radios is how to implement an efficient medium access control (MAC) mechanism that adaptively and efficiently allocates transmission powers and spectrum according to the surrounding environment. In this work, we propose a distributed MAC protocol for operating spectrum-agile radios in a multi-hop ad hoc network. Our protocol differs from previous designs in that it exploits the "dual-receive" capability of radios, thus overcoming various channel access problems that are common to multi-channel designs. We conduct theoretical analysis of the protocol, and study its performance via simulations. We show significant improvement in the system throughput under the proposed MAC design. To maximize this throughput, we propose a cross-layer framework for joint adaptive load and medium access controls. In this framework, the loads of individual nodes are adapted based on the values of local MAC parameters. Simulation results show that the proposed scheme achieves more than 90% of the maximum (global) system throughput that is achieved at saturation, while guaranteeing low collision rates.**

I. Introduction

The concept of spectrum-agile radios, also known as cognitive radios (CRs) [9], has triggered great interest within the research community. Operating these radios as a peer-to-peer (ad hoc) wireless network involves various coordination issues, such as spectrum sensing (e.g., [8]), spectrum sharing and access coordinations (e.g., [20], [18], [3], [16], [11], [10], [17], [6]). In this paper, we focus on designing an efficient and adaptive access control scheme that supports dynamic channel selection and power/rate allocation in a multi-hop cognitive radio network.

Existing work on this topic can be classified according to their architectures (centralized or decentralized). The IEEE 802.22 working group is in the process of standardizing a centralized MAC protocol that enables spectrum reuse by CRs operating on the TV broadcast bands. Centralized architectures were also proposed in [3] [18], whereby a central entity controls spectrum coordination and access. For an ad hoc network, it is desirable to have a distributed MAC protocol that allows CR users to individually sense and control their spectrum. A number of decentralized MAC protocols for CRNs were proposed (e.g., [11], [20], [6], and [16]). Most of these protocols assume that each CR is equipped with multiple radios and is capable of multiple transmissions/receptions at the same time. This assumption comes at the cost of extra hardware, although it greatly simplifies the task of MAC design. Issues such as hidden terminals, exposed terminals, and connectivity are relatively easy to address with multi-transmission/reception capabilities.

In our paper, we assume that each CR is equipped with a single half-duplex radio. We exploit the "dual-receive" feature of the radio and study its impacts on the MAC design. By dual-receive, we refer to the radio's ability to receive on two channels simultaneously when not transmitting. Although this feature represents a simple enhancement on physical layer and receiver design[1], it makes the MAC design much easier. In particular, if we assume a common (or coordinated) control channel, a terminal that is not transmitting any data can tune one of its two receive branches to the control channel while receiving data over the other receive branch. In this way, the multi-channel hidden-terminal problem can be alleviated. However, other issues such as transmitter deafness still remain, and some new challenges such as the control channel bottleneck problem arise with this new feature (details in Section III). One contribution of this paper is in providing a complete distributed contention-based MAC design to overcome all the above issues.

The effectiveness of the proposed MAC still relies on upper-layer designs. For example, in the case of heavy traffic, severe MAC contention and network congestion can occur, which will lead to decreased end-to-end throughput and increased end-to-end delay. Transport layer protocols such as TCP have been used to address these issues by providing the means to flow control. However, it has been shown in many papers that wireless multi-hop ad hoc networks perform poorly under TCP (e.g., [12], [7], and [19]). A number of schemes have been proposed to improve TCP's performance over multi-hop ad hoc networks (see [5] for a survey). Most of these schemes ignore the impact of MAC layer contention on the traffic flow. Most recently, the authors in [19] studied the coupling between medium contention and network congestion, and proposed a novel cross-layer flow control and medium access scheme that utilizes information from MAC frames to conduct flow

This research was supported in part by NSF (grant CNS-0721935), Raytheon, and Connection One (an I/UCRC NSF/industry/university consortium). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

[1]The "dual-receive capability" is readily supported by many recent radios, e.g., QUALCOMM's RFR6500 radio [2] and Kenwood's TH-D7A Dual-Band Handheld transceiver [1].

978-1-4244-4439-7/09/$25.00

control functions. However, this scheme is limited to single-channel systems and only considers the IEEE 802.11 DCF MAC function.

In the second part of this paper, we propose a cross-layer framework for joint adaptive load control and medium access in multi-channel multi-hop ad hoc networks. According to this framework, the traffic loads of individual nodes are adapted using local MAC parameters. Based on the channel conditions, two alternatives are provided: one is used when the control channel is the bottleneck; and the other one is applied when data channels are the bottleneck. When the control channel is the bottleneck, we use queueing model to estimate the degree of saturation in the neighborhood. When data channels (DCs) are the bottleneck, their occupancy is used to measure the degree of saturation. The traffic load is adapted accordingly. Simulation results show that the proposed scheme achieves more than 90% of the maximum system throughput, with low data collision rate and end-to-end delay.

The rest of the paper is organized as follows. Section II introduces our system model, including the radio capabilities and the network model. The MAC design and analysis are presented in Section III. In Section IV, we present the adaptive load control mechanism. The performance of the proposed MAC protocol and the adaptive load control algorithm are evaluated in Section V. Finally, Section VI gives concluding remarks.

II. System Model

We consider an ad hoc network consisting of N CR nodes. CRs may not be necessarily within the transmission range of each other, i.e., hidden-terminal problems can occur, both due to distance and to channel heterogeneity. Nodes can obtain a list of available channels (spectrum holes) via spectrum sensing[2]. Let K be the total number of channels. We assume that these channels are orthogonal, and transmissions can be done over any one of the K channels.

CRs have the following features that are relevant to the MAC design:

1) Dual-receive, single-transmit: Each radio can simultaneously receive two independent bit streams on any two of the K channels, and can transmit on only one channel only. The operation is half-duplex, i.e., while transmitting, the radio cannot receive/listen, even over other channels.
2) Spectrum sensing: When not transmitting, a CR can measure the levels of interference (e.g., E_b/N_o) of all channels simultaneously. This can be done, e.g., via a sequential partial sensing approach with negligible switching/sensing overhead [4] [15].
3) Rate adaptation: Each CR is capable of implementing a variety of modulation schemes and waveforms. Altogether, the system can provide N_R different information rates, each with its corresponding SNR$_{th}$.
4) Rate-SNR relationship: The radios use forward error correction (FEC) together with spreading to reduce the impact of interference. The combined impact of FEC, spreading, modulation, and etc., is reflected in the relationship between the transmission rate and SNR. In practice, this relationship takes the shape of a staircase, as shown in Fig. 1.

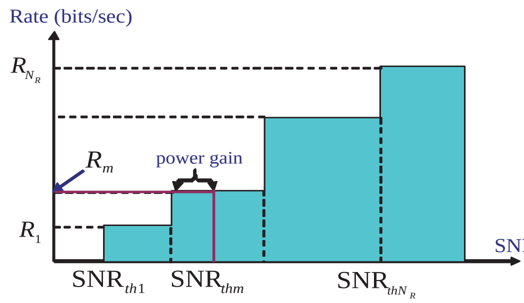


Fig. 1. Rate/SNR relationship.

5) Power control: Each CR supports fine-grain power control. The transmission power is limited to P_{max}.

[2]The spectrum sensing and its hardware implementation details are out of the scope of this paper.

III. MAC Protocol Design

A. Protocol Overview

Our MAC protocol is optimized primarily for network throughput. This is done by allowing as many simultaneous transmissions as possible, with each transmission performed at the highest possible rate. Because of the discrete nature of rate adaptation (see Fig. 1), the same bit rate can be achieved using different transmission powers. Hence, in our design, we first target throughput maximization as the primary objective, followed by energy minimization (via power control) as a secondary objective. Energy reduction, which is reflected in Fig. 1 as a "power gain," will be discussed in detail in Section III-C.

To achieve a completely distributed design, we adopt a random channel access approach that is a variant of CSMA/CA. Fundamentally, such an approach requires nodes to exchange control information. We dedicate one of the K channels for control. This channel is referred to as the control channel (CC). Control information is exchanged on CC at power P_{max} and a fixed rate R_{ctrl} (a similar approach was also used in other multi-channel MAC protocols, such as [13] and [16]). In Section III-D, we discuss how R_{ctrl} is determined. We assume no data packets can be transmitted over the CC and no multiple transmissions can take place over the same data channel in the same neighborhood.

B. Operational Details

To reduce the likelihood of CR collisions, each CR node i maintains a list of available channels (AC_i) and a list of busy nodes (BN_i). The AC_i list consists of channels whose network allocation vectors (NAVs) are zero according to node i. The BN_i list consists of the IDs of nodes that are currently busy transmitting/receiving data packets in the neighborhood of node i. When node i has a data packet to transmit, it checks its AC_i and BN_i lists. If the intended receiver of the data packet is not in BN_i and if AC_i is nonempty, node i contends over the CC using a variant of CSMA/CA. Specifically, for its first transmission or following a successful data transmission, node i selects a random backoff duration B_i that is uniformly distributed between B_{min} and B_{max}. B_{min} needs to be larger than the SIFS duration t_{SIFS} (a mandated small duration between any two control packets), and B_{max} is generally much smaller than the minimum contention window (CW_{min}), typically used in the classic 802.11 protocol. After selecting the backoff duration B_i, Node i periodically decrements the backoff timer when the CC is idle (i.e., no carrier is sensed), and freezes the timer when the CC is busy. When the timer reaches 0 and the channel is still idle, the radio is permitted to transmit its request-to-send (RTS) control packet over the CC.

While the backoff timer is frozen (CC is busy), node i continues to listen to the CC. If node i overhears a clear-to-send (CTS) packet from any other node, it updates AC_i and BN_i. If AC_i is still nonempty and the intended receiver is not in BN_i, node i continues to decrement its timer; otherwise, it freezes its timer until a new data channel becomes available or until the intended receiver finishes its data transmission/reception. While backing off, if node i overhears an RTS packet, it freezes its timer for the duration of that RTS plus t_{SIFS}. This ensures that node i will not attempt to transmit a control packet in the period between the RTS and the subsequent CTS (which node i may not be able to hear).

If node i captures the CC, it starts an RTS/CTS exchange with the intended receiver. Note that our handshaking mechanism is different form the one used in 802.11 scheme. Although we reuse some of the 802.11 terminologies, our control packets have different structures and functions than those of the 802.11 scheme. The RTS contains the *AC* list of the transmitting node along with the packet size (in bytes) of the ensuing data packet (see Fig. 2). The packet size is used along with the data-channel transmission rate (specified in the CTS) to determine the duration of the ensuing data packet.

RTS	Transmitter (TX) ID	Receiver (RX) ID	Packet size	Available channel list

CTS	TX ID	RX ID	Packet size	Data channel index	TX power	TX rate index

Fig. 2. RTS/CTS packet format.

Nodes that are not transmitting will always have one of their two receive “branches” tuned to the CC. So upon receiving an RTS packet from node i, the intended receiver, say node j, uses AC_i along with its own AC_j to determine the appropriate channel and rate to be used for the subsequent data packet (as discussed in Section III-C). The selected channel must belong to both AC_i and AC_j. The receiver will then send a CTS packet, containing the indices of the selected channel and rate. If no channel is available, the receiver will respond with a negative CTS (NCTS) packet.

If a transmitting node i does not hear back a CTS or NCTS from the intended receiver within a specified duration, it concludes that its RTS must have collided with another control packet or that the intended receiver itself is busy transmitting a control or a data packet. In this case, node i backs off following the same backoff procedure described before.

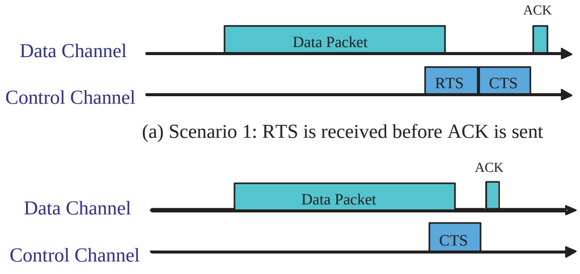
Even if the RTS/CTS exchange was successful and a data-packet transmission ensued over some chosen data channel, a collision over the data channel is still possible due to inconsistencies between the *AC* tables of different nodes and also due to unaccounted for interference from outside the network. If that happens, the receiver will not send the ACK packet, triggering a retransmission of the data packet.

If the sending node successfully receives the CTS packet, it will start its data transmission over the specified data channel. Upon hearing the CTS packet, all neighboring nodes that are listening to the CC, i.e., not in the process of transmitting packets, will update their *BN* lists by adding the IDs of the transmitter and receiver nodes of the upcoming transmission (obtained from the overheard CTS packet), along with the remaining time for the transmission (which includes the ACK duration). These nodes will also update the NAV entry that corresponds to the assigned channel.

Transmitter Deafness: While receiving a data packet over a given data channel, a node still listens to other RTS/CTS exchanges taking place over the CC, and can update its *AC* and *BN* lists accordingly. However, a node that is transmitting a data packet will not be able to listen to the CC, so its *AC* and *BN* tables may become outdated. We refer to this problem as *transmitter deafness*, which is primarily caused by the half-duplex nature of the radios. To remedy this problem, when the receiver sends its ACK, it includes in this ACK any changes in the *AC* and *BN* lists that may have occurred during the transmission of the data packet. The transmitter uses this information to update its own tables.

Delayed ACK: Just before completing the receipt of a data packet, a node may start overhearing a control packet over the CC (see Fig. 3). Sending the ACK packet directly after the data packet may result in incorrect NAV entries at both the receiver and transmitter, and may lead to subsequent data collisions. To avoid interrupting the reception of the control packet, the receiving node defers the transmission of its ACK packet. The amount of delay depends on the type of the overheard control packet. If this packet is a CTS (part (b) of Fig. 3), the node will start its ACK transmission on the data channel right after completely receiving that CTS. If the control packet is an RTS, the node will wait until the end of the next control packet (potentially, a CTS) and then send the ACK. This design significantly reduces the number of data collisions due to incomplete NAV information. To account for the worst-case ACK delay, following the transmission of the data packet, the transmitting node sets its ACK timer to the duration of two control packets plus an ACK duration. A time diagram of the RTS-CTS-DATA-ACK exchange is depicted in Fig. 4.

The mechanism of updating of the NAV using ACK may not completely solve the “transmitter deafness” problem. The transmitter’s NAV updates may be inaccurate because the transmitter and the receiver are located in different contention domains. However, this mechanism will greatly reduce the



(a) Scenario 1: RTS is received before ACK is sent

(b) Scenario 2: CTS is received before ACK is sent

Fig. 3. Two scenarios that motivate delaying the ACK packet.

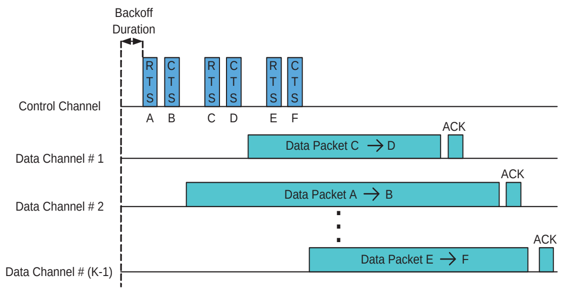


Fig. 4. RTS-CTS-DATA-ACK exchange.

likelihood of the "transmitter deafness" by reducing the number of unnecessary RTS attempts and data/control collisions. We show the effectiveness of this mechanism in Section V.

C. Channel, Rate, and Power Assignment

After receiving an RTS packet that includes the sender's *AC* list, the receiver needs to choose an appropriate data channel, a transmission rate, and a transmission power for the data packet.

Channel and Rate Selection: Based on the received power of the RTS, denoted by P_r, the receiver estimates the transmitter-receiver gain (h_c) over the CC: $h_c = P_r/P_{max}$. From h_c, the receiver estimates the channel gains over all data channels. Let h_m, f_m, and λ_m denote the channel gain, the carrier frequency, and the carrier wavelength of the mth data channel, respectively. Let f_c denotes the carrier frequency of the CC. We consider a simplified path-loss model of the form:

$$P_r = P_t A(\frac{d_0}{d})^\gamma \tag{1}$$

where d_0 is the close-in distance, γ is the path-loss exponent (between 2 and 4), and A is a unitless constant that can be determined via measurements or estimated assuming omni-directional antennas and a free-space propagation environment:

$$A = (\frac{\lambda_m}{4\pi d_0})^2 \tag{2}$$

Given the above, the channel gain over channel m is given by:

$$h_m = \frac{P_r}{P_t} = \frac{1}{{f_m}^2}\left(\frac{c}{4\pi d_0}\right)^2\left(\frac{d_0}{d}\right)^\gamma. \tag{3}$$

For a fixed distance d, we have:

$$h_m/h_c = (f_c/f_m)^2, m = 1, 2, \ldots, K-1. \tag{4}$$

Note that the above derivation is generally valid when the transmission distance $d > d_0$ (d_0 is typically is the range 1-10 meters for indoor and 10-100 meters for outdoor environments).

Another approach for estimating h_m is to rely on direct channel-gain measurements, taken over channel m. Specifically, the receiving node may use the transmission and reception powers of previously received data packets over channel m to estimate the channel gain between itself and a given transmitter.

After estimating h_m for all $m = 1, 2, \ldots, K-1$, the receiver then estimates the interference-plus-noise power $N_0(m)$ over each data channel m. It is assumed that the hardware is capable of providing such estimates for all channels.

For a given data packet, let S be the intersection of the *AC* lists of the transmitter and receiver. The goal of the receiver is to select the channel that provides the highest possible data rate. The data rate, however, is dependent on the received power. So, the receiver first assumes that the transmitter will use its maximum transmission power P_{max}, and accordingly determines the optimal channel m^*:

$$\begin{aligned} & m^* = \text{arg max}\ _{m\in S} R(m) \\ \text{where} & \\ & R(m) = f(\text{SNR}(m)) \\ & \text{SNR}(m) = \tfrac{h_m P_{max}}{N_0(m)}. \end{aligned} \tag{5}$$

In (5), f is the function that represents the rate-SNR relationship (see Fig. 1). The solution to the above problem gives the channel m^* to use and the maximum data rate $R(m^*)$ associated with that channel.

Power Control: The above channel/rate assignment is done assuming the transmitter uses P_{max}. It is possible to reduce this power while maintaining the same channel/rate assignment. This is done by solving the following problem:

$$\begin{aligned} & \text{minimize}\ P(m^*) \\ \text{s.t.} \quad & f(\tfrac{h_{m^*} P(m^*)}{N_0(m)}) = R(m^*). \end{aligned} \tag{6}$$

Essentially, the above problem amounts to eliminating the slack between P_{max} and the minimum power needed to support the rate $R(m^*)$. In practice, the minimum required transmission power needs to be inflated by a small fraction, say ϵ, to account for out-of-network interference (this fraction is often called the *link margin*). As a result, the final transmission power for the data packet is set to $\min\left((1+\epsilon)P(m^*), P_{max}\right)$. The receiver includes in the CTS packet the index of the selected channel m^*, the selected rate (or its index), and the selected transmission power.

D. Channel Capacity Constraints

Let λ be the packet generation rate of a node (in packets/sec), and let D be the size of a data packet (in bits). The traffic load generated by each node is $T = \lambda D$ bits/sec. For now, we assume an extreme case when all the nodes are within the transmission range of each other. We later relax this assumption. By considering the stability of the transmitter queue at each node along with the theoretical capacities of the control and data channels, the following two constraints can be established on the relationship among T, D and N. The first constraint comes from the capacity limit of the CC. Specifically, the average packet generation rate of the whole network is $N\lambda = NT/D$. Each data packet transmission requires at least two control packets (RTS and CTS), hence occupying the CC for $(\|RTS\| + \|CTS\|)/R_{ctrl} + t_{SIFS}$ seconds. The maximum number of control packets that the CC can support in one second is $\frac{1}{\frac{\|RTS\|+\|CTS\|}{R_{ctrl}}+t_{SIFS}}$. To guarantee a stable system, we must have:

$$\text{C1}: \frac{TN}{D} < \frac{1}{\frac{\|RTS\|+\|CTS\|}{R_{ctrl}} + t_{SIFS}}. \tag{7}$$

The second constraint is due to the capacity of the $K-1$ data channels. On average, this capacity is $(K-1)R_{avg}/D$, where R_{avg} is the average transmission rate (in bps) over a data channel. To guarantee a stable system, we require:

$$\text{C2}: \frac{TN}{D} < \frac{(K-1)R_{avg}}{D}. \tag{8}$$

Sometimes, C1 is tighter than C2 when, for example, R_{ctrl} is small; in other cases, data channels are the bottleneck when, for example, D is large and the CC is not fully utilized. The optimal R_{ctrl} can be found by equating the right-hand sides of (7) and (8). By ignoring the small t_{SIFS}, the optimal control rate is given by:

$$R_{ctrl}^{opt} = \frac{(K-1)R_{avg}(\|RTS\| + \|CTS\|)}{D}. \tag{9}$$

In practice, CRs may not be able to operate exactly at rate R_{ctrl}^{opt} because only a limited number (N_R) of rates are available. In such a case, the following procedure is needed for CRs to find the best available rate for the CC. Let $R_1, R_2, \ldots,$ and R_{N_R} be the available rates in an increasing order. When a CR joins the network, it assumes that R_{ctrl} is the nth lowest rate (initially $n = 1$) and approximates R_{avg} using $R_{avg} = (\sum_{i=n}^{N_R} R_i)/(N_R - n + 1)$. It then computes R_{ctrl}^{opt} using (9), and compares the difference between R_n and R_{ctrl}^{opt}, $r_n = |R_n - R_{ctrl}^{opt}|$. The node repeats this procedure for all $n = 1, 2, \ldots, N_R$, and selects R_{ctrl} that corresponds to the minimum r_n.

In multi-hop network scenarios, CR nodes may not be in the same neighborhood. A CR node i can estimate the number of neighboring CRs, denoted by N_i. This N_i can be used to replace N in (7) and (8). However, the optimal control rate computed in (9) remains the same because it is not a function of N. Note that R_{ctrl}^{opt} is a function of the control packet size, data packet size, average data rate, and number of data channels. These parameters are available to each node that joins the network, so nodes can individually determine R_{ctrl}^{opt}.

In Section V, we study the system performance under different values of R_{ctrl}, and illustrate the improvement of system performance by using R_{ctrl}^{opt}.

IV. Adaptive Load Control

In the previous section, we discussed an adaptive MAC design for a multi-channel CR network, and derived the optimal transmission rate for the CC. However, the MAC design alone may not be capable of ensuring good system performance at all times. For instance, at high traffic loads, the system may still experience high network congestion. This congestion has been extensively studied as a flow control problem (e.g., [12], [7], and [19]). Existing flow control schemes typically ignore the impact of MAC layer contention on the traffic flow. In this section, we propose a cross-layer framework for joint adaptive load control and channel access.

We assume that nodes can adjust their traffic loads (i.e., elastic traffic). For example, video/voice users may adjust their codec rates. If nodes keep increasing the value of T, one of the two constraints given in Section III-D will eventually be violated, leading to high collisions and low throughput. The purpose of adaptive load control is to optimize the system performance (i.e., throughput) by operating the network just right at the saturation point (i.e., load at which system performance can no longer by improved). However, the load T that corresponds to the saturation point depends on various factors, including network density and channel conditions, which are generally not available in a distributed ad hoc network. This motivates us to propose a *distributed adaptive load control scheme*, in which nodes adjust their source rates on-the-fly based on local MAC-level information. Our control scheme assumes that the control rate has already been set according to the procedure in Section III-D. Each node in the network compares R_{ctrl} with R_{ctrl}^{opt}. If $R_{ctrl} < R_{ctrl}^{th}$, then CC is the bottleneck; otherwise, the data channels are the bottleneck. If the control channel is the bottleneck, queueing analysis is used to estimate the contention delay, and consequently the degree of saturation in the neighborhood. If the data channels are the bottleneck, data channel (DC) occupancy is used to measure the degree of saturation. The traffic load is adapted accordingly.

A. Control Channel Bottleneck Scenario

If the CC is the bottleneck, once the system reaches the corresponding saturation point, nodes with packets to transmit (or relay) will experience long contention delays on the CC, as illustrated in Fig. 5. When node i has a packet to transmit, it senses the CC and starts backoff when the channel is idle. When the backoff timer B_i reaches 0, node i initializes an RTS/CTS exchange. For node i, we define the control contention delay (CCD) τ_i as the duration between the time the data packet reaches the HOL at the transmitter buffer, and the time the CTS is successfully received at node i. CCD can be estimated by every individual node for each of its transmitted RTS packets.

Note that in the absence of CC contention, the expected value of CCD is $t_{RTS} + t_{CTS} + t_{SIFS} + \bar{B}$, where $\bar{B} = (B_{max} + B_{min})/2$ denotes the average backoff duration of a given node. If there is a large number of contending nodes in node i's neighborhood, then τ_i is likely to be large. Thus τ_i is a measure of the CC crowdedness in the neighborhood.

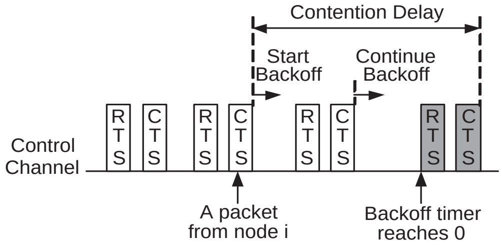


Fig. 5. Contention delay when the CC is the bottleneck.

To estimate τ_i, we model the CCD using a G/M/1 queue. The CC is modelled as a virtual single-server queue (with no "actual" buffer), whose service rate is R_{ctrl}. The system is stable when the total arrival rate is smaller than the service rate, and the saturation point corresponds to the situation when the total arrival rate is close to the service rate. However, there is no way for an individual node to estimate the total arrival rate. The only estimate available is the CCD τ_i for each packet, which can be interpreted as queueing plus service times. From τ_i, node i can estimate the number of customers (neighboring CR nodes) that are serviced ahead of node i. Let this number be denoted by q_i. Then, $q_i \approx \tau_i/\bar{\tau}_i - 1$, where $\bar{\tau}_i = t_{RTS} + t_{CTS} + t_{SIFS} + \bar{B}$.

Note that q_i is the queueing length for the current packet and is a measure of the CC crowdedness. Node i also needs to predict the channel conditions for the next packet. In the prediction, recent data should get more weight than past data. The simple exponential smoothing (SES) model is suitable for this situation. Let α denote the smoothing constant (a value between 0 and 1), and let $\tilde{q}_i(t)$ denote the value of the predicted series at time t. The following formula is used to update the smoothed series recursively as new observations are recorded:

$$\tilde{q}_i(t) = \alpha q_i(t) + (1-\alpha)\tilde{q}_i(t-1). \tag{10}$$

If $\alpha = 1$, the SES model is equivalent to a random walk model; if $\alpha = 0$, the SES model gives the mean estimate. The value of α can be easily optimized by minimizing the mean square error between the predicted value and the observations.

Intuitively, a small $\tilde{q}_i$ indicates an under-utilized queue, so node i should increase its traffic load T_i. On the other hand, if $\tilde{q}_i$ is large, node i should reduce T_i. The thresholds can be set similar to the DECBit protocol [14], i.e., if $q_i < 1$, node i will regard the queue as under-utilized and increase its source rate; if $1 \leq q_i \leq 1+\delta$, node i will keep its current source rate; if $q_i \geq 1+\delta$, node i will decrement its load. δ is selected according to the traffic requirements. If the traffic is delay-tolerant, then δ can be set to a large value; otherwise, δ is set small.

The load control algorithm when the CC is the bottleneck is summarized in Algorithm 1.

B. Data Channels Bottleneck Scenario

In the case of data channel bottleneck, the data channels can be modelled as a G/M/(K-1) queueing system, and a *data contention delay* can be used to adjust the traffic loads. An alternative to this is to use the number of occupied data channels (ODC) (o_i) directly as a measure of the data-channel crowdedness. Intuitively, if o_i is close to $K-1$, it means most of the data channels are busy, and node i should decrement T_i, and vice versa. The SES model is also used to predict the data channel occupancy:

$$\tilde{o}_i(t) = \alpha o_i(t) + (1-\alpha)\tilde{o}_i(t-1) \tag{11}$$

After $\tilde{o}_i(t)$ is computed, we set three thresholds δ_1, δ_2 and δ_3, with $0 < \delta_1 < \delta_2 < \delta_3 < (K-1)$. These thresholds are selected according to the QoS requirements of the traffic. Each node adjusts its traffic load by comparing $\tilde{o}_i(t)$ with these thresholds. The adaptive load control algorithm when data channels are the bottleneck is shown in Algorithm 1.

C. Load Control Algorithm

Each node i initializes its source rate T_i, which is randomly selected from the interval $[T_i^{min}, T_i^{max}]$. Nodes may have different T_i^{min} and T_i^{max} values. Let D be the size of the MAC data unit (MDU). The packet generation rate of node i is given by $\lambda_i = T_i/D$. When a packet is generated by node i, in the case of CC bottleneck, node i measures τ_i and computes the queue length $q_i(t)$, from which it predicts the queue length using (10). In the case of data channel bottleneck, node i estimates o_i and predicts $\tilde{o}_i(t)$ using (11).

The packet generation rate λ_i is adjusted similar to the TCP congestion control mechanism, which is known to be stable. To elaborate, when the CC is the bottleneck, for any attempted packet with $\tilde{q}_i < 1-\delta$, node i regards the queue as under-utilized and increments λ_i: $\lambda_i = \lambda_i + 1$ ($T_i = T_i + D$), similar to the "slow start" phase in TCP; if $\tilde{q}_i \in [1-\delta, 1]$, node i increases λ_i as $\lambda_i = \lambda_i + 1/\lambda_i$ ($T_i = T_i + D^2/T_i$), similar to the "congestion avoidance" phase in TCP; if $\tilde{q}_i \in [1, 1+\delta]$, node i keeps the current λ_i; finally, if $\tilde{q}_i > 1+\delta$, node i considers the network to be congested, and cuts λ_i by half, i.e., $\lambda_i = \lambda_i/2$ ($T_i = T_i/2$). The new value of T_i is then projected to the interval $[T_i^{min}, T_i^{max}]$. When data channels are the bottleneck, a similar procedure is applied by comparing $\tilde{o}_i(t)$ with δ_1, δ_2 and δ_3. Fig. 6 depicts a sample trace of $\tilde{q}_i$ and the corresponding T_i, in the case of CC bottleneck.

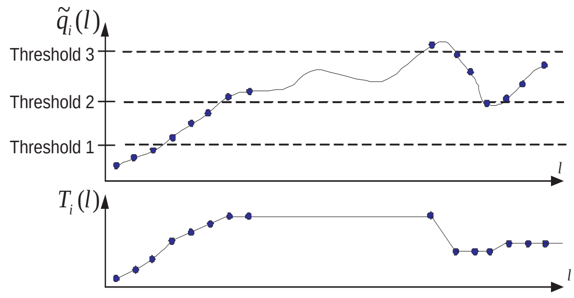


Fig. 6. A sample trace of the adaptive load control mechanism in the case of CC bottleneck.

V. Performance Evaluation

In this section, we evaluate the performance of the proposed MAC protocol and the adaptive load control scheme. At

Algorithm 1 Adaptive load control at node i

Initialize $T_i(0) \in [T_i^{min}, T_i^{max}]$, and iteration count $l = 0$.
Compute R_{ctrl}^{opt} using (9).
while (time<SIMTIME):

if $R_{ctrl} < R_{ctrl}^{opt}$ and node i receives its own CTS **then**
  //Control channel bottleneck
  $l = l + 1$;
  Estimate the current CCD $\tau_i(l)$;
  Compute the queue length $q_i(l) = \tau_i(l)/\bar{\tau}_i(l) - 1$;
  Predict the queue length $\tilde{q}_i(l)$ using (10);
  if $\tilde{q}_i(l) < 1 - \delta$ **then**
    $T_i(l+1) = T_i(l) + D$;
  else if $\tilde{q}_i(l) \in [1-\delta, 1]$ **then**
    $T_i(l+1) = T_i(l) + D^2/T_i(l)$;
  else if $\tilde{q}_i(l) \in [1, 1+\delta]$ **then**
    $T_i(l+1) = T_i(l)$;
  else
    $T_i(l+1) = T_i(l)/2$;
  end if
  $T_i(l+1) = [T_i(l+1)]_{T_i^{min}}^{T_i^{max}}$
else if $R_{ctrl} \geq R_{ctrl}^{opt}$ and node i receives an ACK **then**
  //Data channel bottleneck
  $l = l + 1$;
  Estimate the current ODC $o_i(l)$;
  Predict $\tilde{o}_i(l)$ using (11);
  if $\tilde{o}_i(l) < \delta_1$ **then**
    $T_i(l+1) = T_i(l) + D$;
  else if $\tilde{o}_i(l) \in [\delta_1, \delta_2]$ **then**
    $T_i(l+1) = T_i(l) + D^2/T_i(l)$;
  else if $\tilde{o}_i(l) \in [\delta_2, \delta_3]$ **then**
    $T_i(l+1) = T_i(l)$;
  else
    $T_i(l+1) = T_i(l)/2$;
  end if
  $T_i(l+1) = [T_i(l+1)]_{T_i^{min}}^{T_i^{max}}$
end if

the physical layer, each radio measures the total noise-plus-interference, and checks the SNR of the received signal against the corresponding SNR_{th} to determine whether the packet can be correctly received. The main performance metrics of interest are the end-to-end network throughput, the data and control collision rates, and the average energy consumption per successfully received data bit (E_b). Here, E_b includes the energy consumed in transmitting control packets. It is a more meaningful metric than the average transmission power per bit, as the latter does not account for the transmission time of the bit (i.e., rate adaptation).

In the simulated network, N nodes are randomly distributed in a square area of length 2000 meters. Each node generates traffic according to a Poisson process of rate λ (packets/sec). The packet destination is randomly chosen from all other nodes in the network. Note that at high transmission rates, the target destination may be outside the transmission range of the source node. In this case, multi-hop transmission along the shortest path is used. We implement a simple min-hop Bellman-Ford routing algorithm and use it as a basis for multi-hop communications (for both data and control packets). For simplicity, we ignore the routing overhead. The parameters used in the simulations are summarized in Table I.

TABLE I
SIMULATION PARAMETERS

Number of nodes (N)	40
Number of channels (K)	10
Data packet size (D)	8000 bits
P_{max}	17 dBm
N_0	-111 dBm
Control packet size	120 bits
SIFS duration	10 μs
B_{min}	20 μs
B_{max}	160 μs
Power inflation factor (δ)	0.2
Path-loss exponent	2

TABLE II
TRANSMISSION RATES FOR THE CC

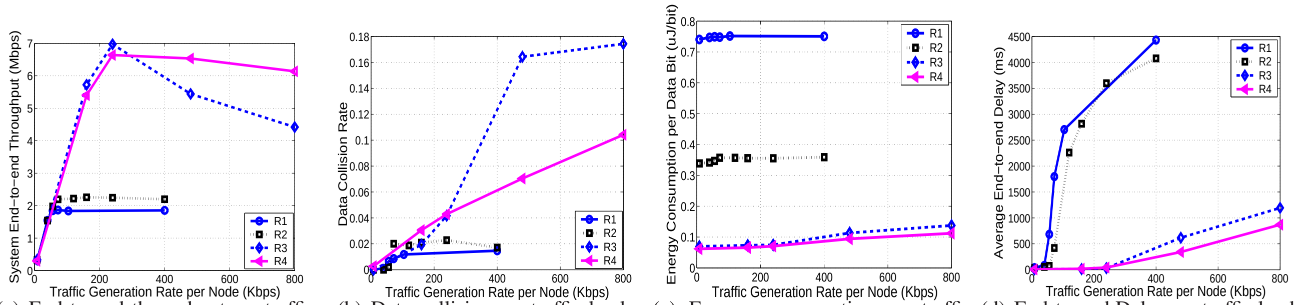
Rate Index	Rate	SNR_{th}
$R1$	78 Kbps	0.26
$R2$	200 Kbps	0.67
$R3$	2.5 Mbps	5.90
$R4$	5 Mbps	15.73

A. Impact of CC Transmission Rate

We first study the system performance under different values of R_{ctrl}. We set $N_R = 4$. The values of different control rates and their corresponding SNR_{th} are summarized in Table II. Based on the values in Table I and II, it is easy to show that all nodes can hear from each other when $R_{ctrl} = R1$, thus simulating a single-hop network. Increasing R_{ctrl} results in a smaller transmission range for control packets, and may give rise to hidden-terminal problems. From the analysis in Section III-D, R_{ctrl}^{opt} is a value between $R3$ and $R4$. So when $R_{ctrl} = R1$ or $R2$, the CC is the bottleneck; when $R_{ctrl} = R3$ or $R4$, data channels are the bottleneck.

Fig. 7(a) depicts the end-to-end throughput versus T under different R_{ctrl} values. For all cases, when T is small, the system is able to serve the incoming traffic, and the throughput grows almost linearly with T. As T increases, the performance becomes bounded by C1 (or C2), i.e., the control (or data) channel(s) is (are) no longer capable of coping with the traffic demand, and the system eventually reaches the saturation point. For the lowest three control rates, increasing R_{ctrl} alleviates the CC crowdedness, and thus delays the process of reaching the saturation point. For the largest two control rates, the throughput reaches its peak value at some $T = T_{opt}$, and then starts to drop with a further increase in T. This behavior can be explained by the fact that at large values of R_{ctrl} and T, the network starts to experience a large number of data collisions due to the contention nature of the MAC protocol.

Fig. 7(b) depicts the data-packet collision rate under different R_{ctrl} values. Before the saturation point, the data collision rate is relatively small. After the saturation point, this rate



(a) End-to-end throughput vs. traffic load. (b) Data collision vs. traffic load. (c) Energy consumption vs traffic load. (d) End-to-end Delay vs. traffic load.

Fig. 7. Performance under difference CC transmission rates.

becomes high for $R3$ and $R4$, because data channels are saturated in this case. This also explains the drop in throughput after the saturation point in Fig. 7(a). The control-packet collision rate follows a similar trend, and is omitted here for brevity.

Fig. 7(c) shows the average energy consumption for successfully transmitting one data bit. More energy is needed for low R_{ctrl}. In other words, using high R_{ctrl} induces short-range and multi-hop transmissions, and is more energy-efficient, as corroborated by several previous studies. Also note that the average energy consumption slightly increases after the saturation point because of the increase in the number of collisions.

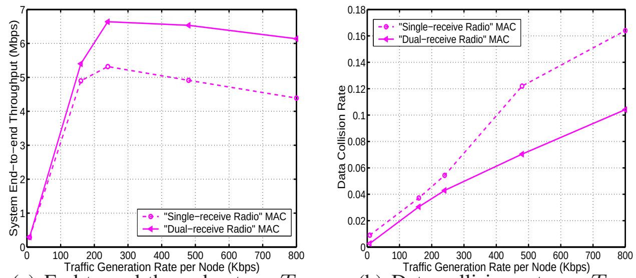
Fig. 7(d) shows the end-to-end delay versus T. As expected, the end-to-end delay is relatively low (less than 100 ms) before the system saturates, which is generally sufficient for most real-time applications. After the saturation point, the end-to-end delay greatly increases. Note that although higher R_{ctrl} results in more hops to reach the same destination, the cumulative end-to-end delay is still smaller than that of the low R_{ctrl} cases.

B. Performance Comparison Between "Single-receive" and Dual-receive Radio MAC Protocol

In this experiment, we compare the system performance of our MAC protocol under two types of radios: (1) single transceiver radios but with dual-receive capability, and (2) single transceiver radios with only single receive branch. In the latter case, the radios operate the traditional 802.11-based MAC protocol. We choose R_4 as the control rate, thus simulating a multi-hop scenario.

Fig. 8(a) depicts the system end-to-end throughput versus T. It shows that the MAC with single-receive radios reaches the saturation point much sooner than our "dual-receive radio" MAC, indicating a lower system throughput for most traffic generation rates. This throughput degradation can be up to 30 percent at high traffic generation rates, which can be explained in Fig. 8(b), where the "single-receive radio" MAC shows a large number of data (and control) collisions. As mentioned before, the performance improvement of our MAC protocol is due to a simple enhancement in the receiver design.

Note that our multi-channel MAC design fully exploits the dual-receive capability. For example, it overcomes the "transmitter deafness" problem by implementing a delayed ACK mechanism to update the transmitter's NAV. Without such a scheme and other features of the design, the performance improvement due to the dual-receive radio MAC would not be significant.



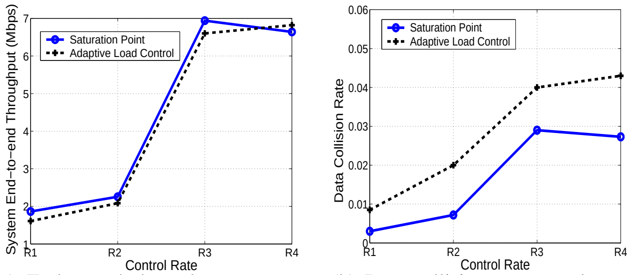
(a) End-to-end throughput vs. T. (b) Data collision rate vs. T.

Fig. 8. Performance comparison between single- and dual-receive MAC designs.

C. Adaptive Load Control Performance

In Fig. 7, we showed that the performance of our proposed MAC is optimized at a given saturation load. If there exists a central entity that controls the traffic loads, then the system can be easily operated at the saturation point. However, in a distributed environment, the optimal value of T is generally unknown to individual nodes. In this section, we allow nodes to adapt their loads on-the-fly based on the adaptive load control mechanism discussed in Section IV, and show the effectiveness of our adaptive load control mechanism by comparing its performance with achieved at the saturation point (obtained by gradually increasing T for all nodes, and observing the resulting end-to-end throughput).

Fig. 9(a) depicts the system end-to-end throughput using adaptive load control and the throughput achieved at the saturation points (Ψ^s). Note that Ψ^s may not be the maximum system throughput (Ψ^{opt}), because we assumed T to be the same for all nodes in the last section. When nodes individually vary their loads, the problem of finding Ψ^{opt} is intractable even with a centralized controller. In fact, Ψ^s is a good approximation of Ψ^{opt} for $R1$ and $R2$, but is smaller than Ψ^{opt} for $R3$ and $R4$ because of the dropping effect at high loads, as shown in Fig. 7(a). This is the reason that the throughput using load control under $R_{ctrl} = R4$ can perform slightly



(a) End-to-end throughput vs. control rate.

(b) Data collision vs. control rate.

Fig. 9. Performance of the adaptive load control mechanism.

better that Ψ^s. Fig. 9(a) shows that the system throughput using load control is more than 90% of Ψ^s.

Finally, Fig. 9(b) shows that the data collision rates under adaptive load control mechanism are relatively low, which proves the efficiency of this mechanism.

VI. Conclusions

In this paper, we proposed a distributed multi-channel MAC protocol for multi-hop ad hoc networks. This protocol overcomes various channel access problems that are common to multi-channel designs. From the simulation results, we showed that the system can efficiently serve all incoming traffic before saturation, and the system performance is maximized at the saturation point. We also proposed a cross-layer framework for joint adaptive load control and medium access control. Simulation results showed that the proposed scheme achieves more than 90% of the system throughput that is achieved at saturation, while guaranteeing low collision rates.

References

[1] Kenwood TH-D7A dual-band handheld transceiver. http://www.kenwoodusa.com/Communications/Amateur-Radio/Portables/TH-D7A(G).

[2] *Qualcomm Announces Sampling of the Industry's First Single-Chip Receive Diversity Device for Increased CDMA2000 Network Capacity*. http://www.qualcomm.com/press/releases/2005/050504-rfr6500.html.

[3] V. Brik, E. Rozner, S. Banerjee, and P. Bahl. DSAP: a protocol for coordinated spectrum access. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (IEEE DySPAN)*, pages 611 – 614, Nov. 2005.

[4] D. Cabric, S. Mishra, and R. Brodersen. Implementation issues in spectrum sensing for cognitive radios. *Proceedings of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, pages 772 – 776, Nov. 2004.

[5] X. Chen, H. Zhai, J. Wang, and Y. Fang. A survey on improving TCP performance over wireless networks. *Resource Management in Wireless Networking*, pages 657 – 695. Kluwer Academic Publishers, 2005.

[6] C. Cordeiro and K. Challapali. C-MAC: A cognitive MAC protocol for multi-channel wireless networks. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (IEEE DySPAN)*, pages 147 – 157, April 2007.

[7] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla. Impact of multihop wireless channel on TCP throughput and loss. In *Proceedings of the IEEE INFOCOM Conference*, March 2003.

[8] M. Gandetto and C. Regazzoni. Spectrum sensing: A distributed approach for cognitive terminals. *IEEE Journal on Selected Areas in Communications*, 25(3):546 – 557, April 2007.

[9] S. Haykin. Cognitive radio: Brain-empowered wireless communications. *IEEE Journal on Selected Areas in Communications*, 23(2):201–220, February 2005.

[10] Y. T. Hou, Y. Shi, and H. D. Sherali. Optimal spectrum sharing for multi-hop software defined radio networks. In *Proceedings of the IEEE INFOCOM Conference*, Anchorage, AL, May 2007.

[11] S. Krishnamurthy, M. Thoppian, S. Venkatesan, and R. Prakash. Control channel based MAC-layer configuration, routing and situation awareness for cognitive radio networks. In *Proceedings of the Military Communications Conference (MILCOM)*, Oct. 2005.

[12] J. Li, C. Blake, D. Couto, H. Lee, and R. Morris. Capacity of ad hoc wireless network. In *Proceedings of the ACM MobiCom Conference*, July 2001.

[13] A. Nasipuri, J. Zhuang, and S. R. Das. A multichannel CSMA MAC protocol for multihop wireless networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1402 – 1406, Sep. 1999.

[14] K. K. Ramakrishnan and R. Jain. A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer. In *Proceedings of the ACM SIGCOMM Conference*, Aug. 1988.

[15] A. Sabharwal, A. Khoshnevis, and E. Knightly. Opportunistic spectral usage: Bounds and a multi-band CSMA/CA protocol. *IEEE/ACM Transactions on Networking*, 15(3):533 – 545, June 2007.

[16] F. Wang, M. Krunz, and S. Cui. Price-based spectrum management in cognitive radio networks. IEEE Journal on Selected Topics in Signal Processing (JSTSP) - Special Issue on Signal Processing and Networking for Dynamic Spectrum Access, 2(1):74–87, Feb. 2008.

[17] Y. Yuan, P. Bahl, R. Chandra, T. Moscibroda, S. Narlanka, and Y. Wu. Allocating dynamic time-spectrum blocks for cognitive radio networks. In *Proceedings of the ACM MobiHoc Conference*, Sep. 2007.

[18] S. A. Zekavat and X. Li. Ultimate dynamic spectrum allocation via user-central wireless systems. *Journal of Communications*, 1(1):60–67, April 2006.

[19] H. Zhai and Y. Fang. Distributed flow control and medium access in multihop ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(11):1503 – 1514, Nov. 2006.

[20] 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*, 25(3):589–600, April 2007.