Channel Access and Traffic Control for Dynamic-spectrum Networks with Single-transmit, Dual-receive Radios

Marwan Krunz† and David Manzi§
†Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ, 85721
krunz@ece.arizona.edu

§Raytheon Corporation, Tucson, AZ, 85756
dgmanzi@raytheon.com

Abstract

Cognitive radios have a great potential to improve spectrum utilization by enabling dynamic spectrum access. 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 is unique in that it exploits the "dual-receive" capability of radios, which is used to overcome channel access problems that are common to multi-channel designs. We conduct theoretical analysis of the protocol, and study its performance via simulations. To further improve the system throughput, we propose a framework for joint adaptive load and medium access controls. 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.

Index Terms

Multi-channel access, MAC protocols, Spectrum-dynamic networks, Cognitive radio, Single transceiver.

I. Introduction

The concept of a cognitive radio (CR) has recently triggered great interest within the research community [1]. **The term CR is often used to refer to a spectrum-agile radio that is capable of interference sensing, spectrum adaptation, and environment learning. The CR may also be capable of synchronously transmitting over multiple (not necessarily contiguous) frequency bands using technologies such as multi-carrier OFDM.**

This research was supported in part by NSF (under grants CNS-0721935 and CNS-0904681), 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 NSF.

From a spectrum-access priority standpoint, the status of users in a CR network (CRN) relative to other users (e.g., legacy systems) varies depending on the application context. The most popular context for CRN operation is that of an *opportunistic/secondary network*, whose goal is to exploit underutilized but licensed portions of the spectrum. In this case, CR users have a secondary status, and they must operate without impacting the transmissions of licensed systems. Another important context arises in military networks, whereby CR systems are not secondary users; rather, they are agile systems that exploit their dynamic spectrum access capability to hop between different frequencies (to avoid congested channels, jamming, etc.). The notion of primary vs. secondary user may not exist here, although CR users are still required to avoid certain "prohibited channels."

Architecturally, a spectrum-adaptive CRN can be organized in a centralized (access-point based) or a distributed fashion (ad hoc network). In a centralized architecture, spectrum assignment is relatively simple, as one entity (the base station or the access point) is responsible for the assignment function. An example of a centralized CRN is the network model used in the IEEE 802.22 standard [2]. Other centralized models are found in [3] [4]. For a distributed ad hoc network, a MAC protocol is needed to resolve channel contention. In this case, spectrum assignment is often intertwined with MAC design. The problem of joint spectrum assignment and MAC design is made more challenging by the presence of the multi-channel hidden terminal problem.

Numerous mechanisms for channel access and spectrum assignment in CRNs have been proposed in the literature. Representative examples are found in [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17] (see also [18], [19] and the references therein). In [5], the authors proposed a decentralized channel-sharing mechanism for CRNs based on a game-theoretic approach, addressing both cooperative and non-cooperative scenarios. However, no operational MAC protocol was presented. AS-MAC [7] is a spectrum-sharing protocol for CRNs that coexist with a GSM network. CR users select channels based on the CRN's control exchanges and GSM broadcast information. Explicit coordination with the the GSM network is required. In [14], the concept of a time-spectrum block is introduced to model spectrum reservation in a CRN. Based on this concept, the authors presented centralized and distributed CRN protocols, using a common control channel for spectrum coordination. DC-MAC [10] is a cross-layer distributed scheme for spectrum allocation/sensing. Its optimization framework is based on partially observable Markov decision processes. In [16], the authors developed a spectrum-aware MAC protocol for CRNs, called CMAC. CMAC enables opportunistic access and sharing of TV white spaces by adaptively allocating spectrum among contending users. DDMAC [17] is a spectrum-sharing protocol that aims at maximizing the CRN throughput through probabilistic channel assignment. This is done by exploiting the dependence between the signal's attenuation model and the transmission distance while considering traffic and interference conditions. Using probabilistic interference analysis, the authors in [20] presented a MAC design for CRNs that provide soft guarantees on the outage probability for primary users. Channel assignment was done according to the "best channel" greedy approach, originally used in [21], [22], [23] for a multi-channel CSMA system. Most of the above protocols assume that each CR is equipped with multiple radios, and is capable of multiple transmissions/receptions at the same time. Although it greatly simplifies the MAC design, this assumption has a significant hardware cost associated with it.

In this paper, we consider a network of CR nodes in which each device is equipped with a single half-duplex radio. The radio has a "dual-receive" capability, which allows it to receive on two channels simultaneously when not transmitting. While such a feature represents a simple hardware upgrade[1], it significantly facilitates the MAC design in a multi-channel network. In particular, if we assign a common (or coordinated) network channel for control purposes, a device that is not transmitting can tune one of its two receive chains to the control channel while simultaneously receiving data over the other chain. This way, the multi-channel hidden-terminal problem can be alleviated. However, other issues such as transmitter deafness still remain, and some new challenges arise with this new feature (details in Section II). One contribution of this paper is in providing a complete distributed contention-based MAC design that addresses these challenges.

Irrespective of how efficient the MAC design is, the overall system performance is still heavily dependent on upper-layer functions. For example, under heavy traffic, severe MAC contention and network congestion can occur, which decrease the end-to-end throughput and increase the end-to-end delay. Transport-layer protocols such as TCP address these issues by using end-to-end flow control. However, it is known that TCP performs poorly in multi-hop wireless networks (see, for example, [26], [27], and [28]). A number of schemes have been proposed to improve TCP's performance over multi-hop ad hoc networks [29], but most of them ignore the impact of MAC layer contention. Recently, the authors in [28] studied the coupling between channel 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 control functions. However, this scheme is limited to single-channel systems.

In the second part of this paper, we propose a cross-layer framework for joint adaptive load control and MAC design in multi-channel multi-hop ad hoc networks. According to this framework, the traffic loads of individual nodes are adapted using locally measurable parameters. Depending on channel and traffic conditions, two cases are considered for load control: one when the control channel is the bottleneck and the other when the data channels are the bottleneck. In the first case, a generalized queueing model is used to estimate the degree of saturation in a node's neighborhood. When data channels 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, and describes the MAC design and radio-resource allocation aspects. In Section III, we present the adaptive load control mechanism. The performance of our designs are evaluated in Section IV. Finally, Section V gives concluding remarks.

II. MAC Protocol Design

A. Network and Radio Models

We consider an ad hoc network consisting of N CR nodes. Two topological scenarios are considered: "single collision domain" and "multiple collision domains." In the first scenario, we assume that all nodes are within the transmission range of each other, whereas this assumption is not met in the second scenario. Note that even in the first scenario, hidden-terminal problems may still occur, in part due to channel heterogeneity and transmitter deafness. Nodes can obtain a list of available channels (spectrum holes) through spectrum sensing[2]. Let K be the total number of frequency channels available for CR operation. We assume that CR nodes have the following features that are relevant to the MAC design:

[1]The "dual-receive" capability is supported by several recent radios, including QUALCOMM's RFR6500 [24] and Kenwood's TH-D7A Dual-Band Handheld transceiver [25].

[2]The details of the spectrum sensing process are out of the scope of this paper.

1) Dual-receive, single-transmit: Each radio can simultaneously receive two independent bit streams on any two of the K channels, but can transmit over one channel only. The operation is half-duplex, i.e., while transmitting, the radio cannot receive/listen, even over other channels. **As discussed later, one of the receive chains is always tuned to the control channel. The other chain is tuned to whatever data channel the node is supposed to receive a packet on. This channel is decided after the receiver executes a channel selection process, described later. The switching overhead (to allow the circuitry to settle down) is usually in the order of a few tens of microseconds.**
2) Interference sensing: When not transmitting, a CR can measure the interference levels of all K channels. **We assume energy-based sensing based on the E_b/N_o ratio, which is known to have negligible switching/sensing overhead ($\sim$ tens of microseconds) [30], [31]. It is relatively cheap to implement several energy detectors at a node, allowing it to measure the E_b/N_o ratio for several channels simultaneously. In fact, our MAC design is inspired by a specific CR hardware platform, developed by Raytheon Corporation, in which the radio can sense 10 channels simultaneously in a few microseconds, then switch to 10 other channels, and so on. In practice, the maximum number of channels that are available for CR operation is in the tens (not hundreds). For example, the portion of the VHF/UHF spectrum targeted for opportunistic access (i.e., the TV white spaces) consists of no more than 70 channels (in fact, the ones targeted by the 802.22 standard is merely 36 channels in the UHF band). With such a number and using an array of energy-based sensors, it is possible to scan all K channels in no more than 0.1 millisecond. Relative to the transmission time of a packet, this sensing time is negligible.**
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 threshold (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, and modulation is reflected in the relationship between the transmission rate and SNR. In practice, this relationship takes the shape of a staircase, as shown in Figure 1.

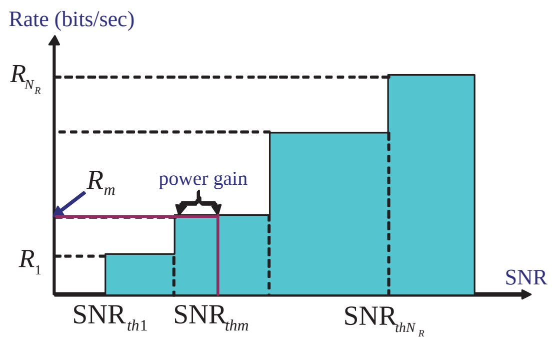


Fig. 1. Rate/SNR relationship.

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

In the rest of this section, we describe a distributed MAC protocol that exploits the aforementioned radio capabilities in a multi-hop ad hoc CRN. For now, we assume a single collision domain, i.e., a control-packet can be overheard by all the nodes in the network.

B. *Protocol Overview*

In general, the parameters of a MAC protocol can be optimized for different objectives (e.g., minimize energy consumption, maximize network throughput, etc.). Our protocol is primarily optimized for network throughput. This is done by allowing as many simultaneous transmissions as possible, with each transmission performed at the highest possible rate. Because we have a discrete set of rates, a given 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 Figure 1 as a "power gain," will be discussed in detail in Section II-D.

To achieve a 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. One of the K channels is designated for control. This channel is referred to as the control channel (CC). **In non-opportunistic CRNs (e.g., military networks), dedicating a small-bandwidth channel for control functions is quite feasible. Sometime, this channel takes the form of a common frequency hopping sequence, known to all nodes. For an opportunistic CRN, the network may initially use one of the ISM bands for control, allowing nodes to exchange their spectrum sensing results and agree on a common idle channel for control. Note that in this case, the initial CC is *not* dedicated but is known a priori. Several MAC designs for CR systems have made such an assumption (e.g., [6], [7], [8], [16], [14]). Other approaches for dynamic assignment of the CC (e.g., [32]) can also be used.**

Control information is exchanged on the CC at power $P_{\max}$ and a fixed rate R_{ctrl}. In Section II-E, 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.

C. *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). AC_i consists of channels whose network allocation vectors (NAVs) are zero according to node i. BN_i 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 to node j, it checks AC_i and BN_i. If $j \notin BN_i$ and if $AC_i \neq \emptyset$, 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 B_i, node i periodically decrements the backoff timer when the CC is idle (i.e., no carrier is sensed), and freezes it when the CC is busy. Once the timer reaches 0, if the channel is still idle, node i proceeds to transmit a request-to-send (RTS) 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. We describe the handshaking mechanism using similar terminology to that of the 802.11 channel access scheme, noting that the two schemes have fundamental differences in the data structures and functionality. In our design, the RTS contains the *AC* list of the transmitting node along with the packet size (in bytes) of the ensuing data packet (see Figure 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. **Note that the *AC* list contains the set of idle channels (the spectrum holes) only. At most, this list may consist of all K channels. The IDs of these channels can be efficiently encoded in the RTS packet. For example, one may use a range notation (e.g., "11–68") to convey a range of channels. One could also use the complement of the *AC* list (requiring a 1-bit complement field) to convey busy channels rather than idle channels (that is, if the *AC* list contains more than $K/2$ channels).**

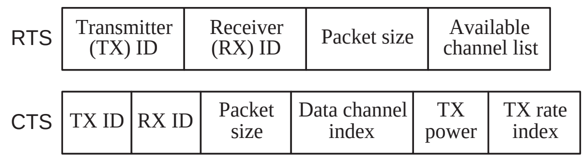


Fig. 2. RTS/CTS packet formats.

Nodes that are not transmitting 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 II-D. 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.

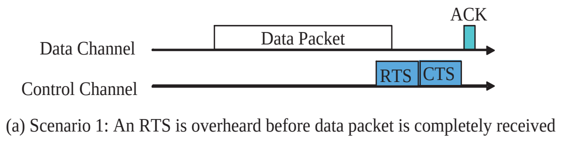
Upon receiving the CTS packet, node i first checks the physical carrier sensing flag for the selected data channel. If no carrier is detected, data transmission proceeds; otherwise, node i assumes the data channel is occupied, updates its AC list accordingly, and repeats the RTS/CTS exchange to identify another channel. Note that no backoff is performed over data channels, because contention resolution has already (largely) been dealt with over the CC. Even if the RTS/CTS exchange is successful and a data-packet transmission ensues over some 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 does not send an ACK packet, triggering a retransmission of the data packet. We set a limit $\text{DATA}_{\max}$ on the number of data-packet retransmissions. For each data-packet retransmission, the complete contention process has to be repeated. It should be noted that the conventional exponential backoff approach (i.e., doubling the contention window after each collision) commonly used in single-channel CSMA/CA protocols is not adopted in our design, as such an approach is deemed conservative for a multi-channel environment.

If 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 another packet. In this case, node i backs off following the same backoff procedure described before. A limit of $\text{RTS}_{\max}$ is imposed on the number of RTS retransmissions. If this limit is reached, the transmitter gives up and reports an error to the higher layer. The packet is dropped from the MAC-layer queue.

Upon hearing a 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. In this multi-channel scenario, the NAV consists of a table, with one entry per data channel. Each entry contains a counter for the remaining time until the ongoing data transmission and the ensuing ACK (sent over the same data channel) are completed. The *AC* list discussed above is a subset of the NAV table with counter values set to zero.

Transmitter Deafness: While receiving a data packet over a given channel, a node continues to listen 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 (or control) packet will not be able to listen to the CC, so its *AC* and *BN* tables may become outdated. This problem, which is often referred to as *transmitter deafness*, is caused by the radio's half-duplicity. To remedy this problem, when the receiver sends its ACK, it includes in it 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. This approach does not completely solve the transmitter deafness problem (the transmitter's NAV entries may be inaccurate because the transmitter and the receiver are located in different contention domains). However, this mechanism will greatly reduce the likelihood of transmitter deafness by reducing the number of unnecessary RTS attempts and data/control collisions. We show the effectiveness of this mechanism in Section IV.

Delayed ACK: Just before completing the receipt of a data packet, a node may start overhearing a control packet (see Figure 3). If this node sends its ACK packet right after the data packet reception, it becomes a deaf transmitter, and may end up with incorrect NAV entries. To avoid interrupting the reception of an overheard control packet, the receiving node defers the transmission of its ACK packet by an amount that depends on the type of the overheard control packet. If the control packet is an RTS (part (a) of Figure 3), the node will wait until the end of the next control packet (potentially, a CTS) and then send the ACK. If the overheard packet is a CTS (part (b) of Figure 3), the node will send its ACK right after completely receiving that CTS. This design significantly reduces the number of data collisions due to incomplete NAV information. To account for the worst-case ACK delay, after transmitting a data packet, the sending node sets **the timeout value of its ACK timer to the sum of two control periods (RTS and CTS), an ACK duration, and two interleaving SIFS periods. Overhearing nodes set their NAV entries similarly.**


ACK
Data Channel
Data Packet
Control Channel
RTS
CTS
(a) Scenario 1: An RTS is overheard before data packet is completely received

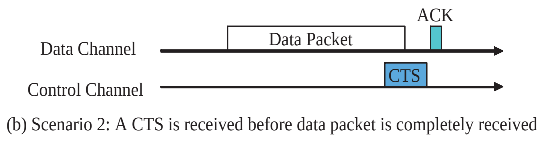

ACK
Data Channel
Data Packet
Control Channel
CTS
(b) Scenario 2: A CTS is received before data packet is completely received

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

D. Channel, Rate, and Power Assignment

After receiving an RTS packet, the receiver selects an appropriate data channel, a transmission rate, and a transmission power for the data packet.

Channel and Rate Selection: **We use a greedy approach for channel selection, which is similar to approaches used in multi-channel systems (e.g., [23], [21], [22]).** 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, 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. This approach is suitable for static or relatively low-mobility environments.

After estimating h_m, $m = 1, 2, \ldots, K-1$, the receiver 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 transmitter and receiver *AC* lists. 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 Figure 1). The solution to the above problem gives the channel m^* to use and the maximum data rate $R(m^*)$ associated with that channel.

The above channel selection process results in the best available channel based on measured interference. One of the merits of this selection criterion is its robustness to inconsistencies in the *AC* lists. Specifically, if the *AC* list of a node i is not up-to-date (e.g., as a result of missing a CTS transmission between two neighbors), it is unlikely that node i will later select a data channel, say m, that is already being used by its neighbors, even if $m \in AC_i$. The reason is that the measured interference on channel m at node i is likely to be high.

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:

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

Essentially, the above problem amounts to eliminating the slack between $P_{\max}$ and the minimum power needed to support the rate $R(m^*)$, **denoted by** $P_{\min}$. This slack was shown as "power gain" in Figure 1. $P_{\min}$ **is found by solving the following equation:**

$$\frac{h_{m^*} P_{\min}}{N_o(m^*)} = \textbf{SNR}_{th^*} \tag{7}$$

where SNR$_{th^*}$ is the SNR threshold at which the transmission rate jumps to $R(m^*)$**.**

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 $\min\{(1+\epsilon)P_{\min}, P_{\max}\}$. 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.

E. 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 all 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, where the notation $\|x\|$ indicates the size in bits of a packet x. The maximum number of control packets that the CC can support in one second is $\frac{1}{\frac{\|RTS\|+\|CTS\|}{R_{ctrl}}+t_{SIFS}}$. **A necessary (but not sufficient) condition for the stability of the system is that:**

$$\text{C1}: \frac{TN}{D} < \frac{1}{\frac{\|RTS\|+\|CTS\|}{R_{ctrl}}+t_{SIFS}}. \tag{8}$$

The second constraint is related 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 an arbitrary data channel. For a stable system, we require that:

$$\text{C2}: \frac{TN}{D} < \frac{(K-1)R_{avg}}{D}. \tag{9}$$

In some cases, C1 is tighter than C2 (for example, when R_{ctrl} is small); in other cases, data channels are the bottleneck (for example, when D is large and the CC is not fully utilized). The optimal R_{ctrl} can be found by equating the right-hand sides of (8) and (9). Ignoring the small t_{SIFS}, the optimal control rate is given by:

$$R_{ctrl}^{opt} = \frac{(K-1)R_{avg}(\|RTS\|+\|CTS\|)}{D}. \tag{10}$$

R_{ctrl}^{opt} **can be periodically (but infrequently) estimated by the network operator**[3] **and kept fixed for the whole network in between estimates, i.e., nodes do not need to determine** R_{ctrl}^{opt} **dynamically. The determination**

[3]**Although channel access and per-packet spectrum/power/rate control are done in a purely distributed fashion, other functions such as authentication, network administration, and management still require a centralized operator. This operator also performs offline and infrequent network-wide management functions, including the updating of** R_{ctrl}^{opt}**.**

of R_{ctrl}^{opt} can be based on a network-wide R_{avg}, computed by the network operator using periodic updates of the R_{avg} values (and associated total number of data packets used to determine these values), provided by various nodes. The operator may infrequently change R_{ctrl}^{opt} upon a significant change in the reported R_{avg}. If R_{ctrl}^{opt} does not match any of the available transmission rates, the operator can select the closest rate to R_{ctrl}^{opt}.

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

F. Multiple Collision Domains

The channel access mechanism discussed in Section II-C is tailored for a "single collision domain" network, where any control transmission can be heard by all nodes. Even in such a scenario, collisions over the CC may still take place, leading to inconsistencies in nodes' databases and, subsequently, to data-packet collisions. If the same access protocol were to be applied to a network with multiple collision domains, many more collisions as well as packet drops can occur. To explain, consider the simple network in Figure 4. In this example, node *A* sends its RTS over the CC to node *B*, which responds back with a CTS. Suppose that node *B* selects channel k for its data communications with *A*. Node *B* indicates such a selection in the CTS. Although node *C* overhears *A*'s RTS, it does not hear *B*'s CTS. Therefore, node *C* will not update its AC list to exclude channel k. Moreover, even though node *C* hears the RTS, it does not know if the data transmission from *A* to *B* will take place or not (e.g., node *B* may not respond back with a CTS because it is busy doing something else), so it does not add node *A* to its BN.

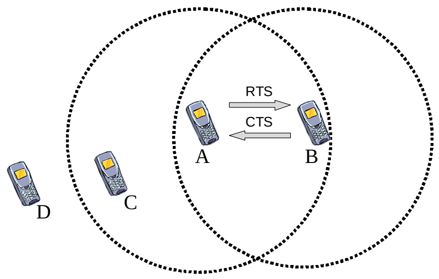
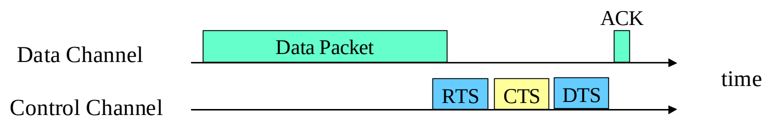


Fig. 4. Example that demonstrates the need for a 3-way handshake over the CC in the case of multiple collision domains. The two circles represent the control transmission ranges for nodes *A* and *B*, respectively.

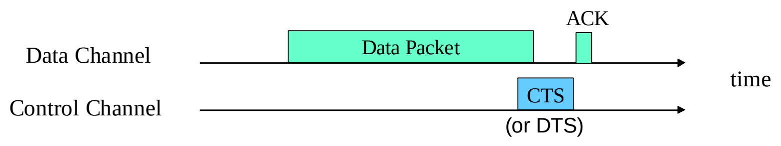
At a later time, node *C* may decide to communicate with node *A*, even though the latter is busy transmitting to *B*. Node *C* will transmit an RTS to *A*. When no reply is received from *A*, node *C* backs off over the CC and retransmits the RTS again. The process is repeated $\text{RTS}_{\max}$ times, after which node *C* gives up and drops the packet. Our simulations indicate that this problem can lead to a large number of packet drops. Another, less critical scenario is if node *C* decides to communicate with another node, say *D*. Node *C* sends its AC list in the RTS to *D*, including in it channel k. It is conceivable that node *D*, which is outside the transmission ranges of *A* and *B*, may select channel *k* for the $C \to D$ data communication. This may cause a collision at node *A*, for example if *C* transmits its data packet while *A* is receiving an ACK packet from *B*. Other scenarios may also take place and lead to collisions over a data channel.

One remedial solution to the above situations is to include a third control packet in the CC handshake. Specifically, when node *A* receives the CTS from *B*, it echoes back its content in a *decided-to-send* (DTS) packet, which *A* broadcasts to its neighbors. The purpose of the DTS is to allow the neighbors of *A* who are not neighbors of *B* to get the parameters of the upcoming data transmission (e.g., selected channel, rate, and power). After sending the DTS, node *A* proceeds to send the data packet over the selected data channel. In this case, node *C* will be able to update its AC and BN lists based on the information in the DTS. Such a 3-way handshake process has been used in other multi-channel protocols (e.g., [33], [20]).

While the extra DTS packet reduces the rate of collisions and packet drops, it also increases the load on the CC. The overall impact of this packet on network performance will be shown later in Section IV. With a DTS packet in place, the mechanism for delaying the ACK, described in Section II-C, has to be modified, as follows. If before completing the reception of a data packet, a node starts overhearing an RTS, this node will delay the transmission of its ACK packet until it overhears the corresponding CTS or DTS, whichever comes first. This situation is shown as Scenario 1 in Figure 5. Note that the node that overhears the RTS may not overhear its corresponding CTS, but is likely to overhear the DTS. Another possible scenario is that the node that is receiving a data packet starts overhearing a CTS (or DTS) packet. In this case, this node will delay its ACK transmission until the completion of this CTS/DTS (Scenario 2 in Figure 5). Because the sender of the data packet does not know which of these two scenarios (if any) will take place, it has to assume the worst-case delay for the ACK. Hence, when a node transmits a data packet, it sets its timer to the sum of the durations of three control packets, one ACK packet, and the associated interleaving SIFS durations. If the timer expires before the receipt of an ACK packet, the sender retransmits the data packet. We refer to this implementation as a 'full-wait' option.



(a) Scenario 1: RTS is overheard before data packet is completely received



(b) Scenario 2: CTS/DTS is overheard before data packet is completely received

Fig. 5. Full-wait option for delaying the ACK transmission with a 3-way (RTS/CTS/DTS) handshake.

It may be argued that the 'full-wait' option is overly conservative in that it forces the two ends of a data transmission to stay idle until other RTS/CTS/DTS activities on the CC are completed. More aggressive policies can be adopted. For example, one option, which we refer to as 'partial-wait', is to limit the wait to no more than two control packets. Specifically, if a node that is in the process of receiving a data packet overhears an RTS, this node waits until it receives the corresponding CTS. Once the CTS is overheard, the node sends its ACK. If no CTS is overheard, the node does not wait to overhear the DTS, and instead proceeds with its ACK transmission. This situation is shown in Figure 6. Of course, if the first overheard control packet is a CTS (not shown in the figure), the overhearing node sends its ACK right after overhearing this CTS. The data sender sets its timer to the sum of the durations of two control packets, one ACK packet, and the associated interleaving SIFS durations. Under the ‘partial-wait’ option, the overhearing node may end up missing some AC and BN information, but such reduction in accuracy may be compensated for by the higher degree of protocol aggressiveness.

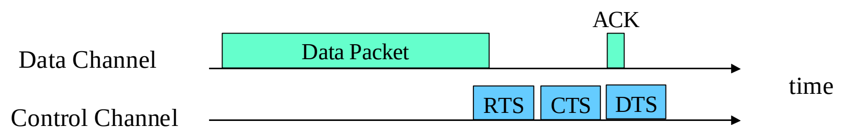


Fig. 6. Partial-wait option for delaying the ACK transmission with a 3-way handshake (scenario when RTS is overheard first).

As a third (extreme) option, we also consider the possibility of not waiting at all before sending the ACK packet, except for one control packet (which could be RTS, CTS, or DTS) that a node may start overhearing before it has completed the reception of a data packet. This option, which we refer to as ‘no-wait’, is shown in Figure 7. It is the most aggressive of the three options, but has the highest chances of causing inconsistencies in the AC and BN lists. We test the performance of these three options in Section IV.

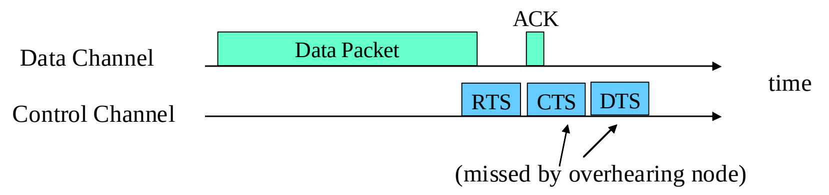


Fig. 7. No-wait option for delaying the ACK transmission with a 3-way handshake (scenario when RTS is overheard).

III. 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 layer cannot alone ensure good system performance at all times. In particular, at high traffic loads, the network becomes congested. This congestion has been extensively studied as a flow control problem (e.g., [26], [27], and [28]). Existing flow control schemes typically ignore MAC layer contention. 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., traffic is elastic). For example, video/voice users may adjust their codec parameters to achieve a variable source rate. **Adapting the traffic rate is quite feasible for real-time voice and video communications, and is often implemented in applications such as IP telephony and video broadcast over the Internet. For such adaptive applications, the default transport layer is UDP.**

If nodes keep increasing the value of T, one of the two constraints given in Section II-E 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 around the saturation point (i.e., the point at which the 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 in a distributed setting are not available to individual nodes. 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 CC rate has already been set according to the procedure in Section II-E. Each node in the network compares the current R_{ctrl} with R_{ctrl}^{opt}. If $R_{ctrl} < R_{ctrl}^{th}$, then the 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 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 Figure 8. When a node i has a packet to transmit, it senses the CC. If the CC is idle, node i start the backoff process. Once the backoff timer reaches 0, node i initializes an RTS/CTS exchange. For node i, we define the control contention delay (CCD) as the duration between the time a data packet reaches the HOL at the transmitter buffer and the time the associated CTS is successfully received at node i. CCD may involve multiple RTS/CTS transmissions. It can be estimated by individual nodes.

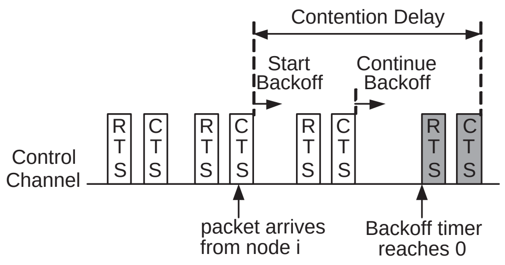


Fig. 8. Contention delay when the CC is the bottleneck.

Let τ_i be the CCD value at node i. In the absence of CC contention, $E[\tau_i] = t_{RTS} + t_{CTS} + t_{SIFS} + \bar{B}$, where $\bar{B} = (B_{\max} + B_{\min})/2$ is the average backoff duration over the CC. 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.

To estimate τ_i, we model the CC as a *virtual* single-server queue with service rate R_{ctrl}. This queue is stable when the total arrival rate is smaller than the service rate. 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 know the arrival rate in advance. Alternatively, node i estimates τ_i for each data 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 for a general queue, $q_i \approx \tau_i / E[\tau_i] - 1$, where $E[\tau_i] = t_{RTS} + t_{CTS} + t_{SIFS} + \bar{B}$. **This estimate does not depend on specific probability distributions. When the estimate is inaccurate, it leads to an unnecessary increase/decrease in the application source rate, which subsequently leads to a corresponding change in the observed CCD. This change causes a correction in the estimated q_i. In other words, the estimation process acts as a closed-loop control system with feedback.**

Note that q_i is the queue 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 estimated value of q_i at time t. The following formula is used to update the smoothed series recursively as new observations are recorded:

$$\tilde{q}_i(t) = \alpha q_i(t) + (1-\alpha)\tilde{q}_i(t-1). \tag{11}$$

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

Intuitively, a small $\tilde{q}_i$ indicates an under-utilized queue, so node i should increase its load T_i. On the other hand, if $\tilde{q}_i$ is large, node i should reduce T_i. The increase/decrease thresholds can be set similar to the DECBit protocol [34]. Specifically, if $\tilde{q}_i < 1$, node i regards the queue as under-utilized and increase its source rate; if $1 \leq \tilde{q}_i \leq 1+\delta$, node i maintains its current source rate; and if $\tilde{q}_i \geq 1+\delta$, node i decrements its rate. The thershold δ can be selected according to the traffic requirements. For example, if the traffic is delay-tolerant, then δ can be set to a large value. The details of the load control algorithm when the CC is the bottleneck are described in lines 1-16 of Algorithm 1. In this algorithm, D refers to the maximum size of the MAC data unit (MDU).

B. Data Channels Bottleneck Scenario

If data channels are the bottleneck, we model them as a K-server queueing system. We can then compute the *data contention delay* for this queueing system and use this delay to adjust the traffic loads. Alternatively, we can use the number of occupied data channels (ODC) as a measure of data-channel crowdedness. Let o_i be the ODC value at node i. 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 can also be used to predict data channel occupancy:

$$\tilde{o}_i(t) = \alpha o_i(t) + (1-\alpha)\tilde{o}_i(t-1). \tag{12}$$

Besides computing $\tilde{o}_i(t)$, we also need to set the values for three thresholds δ_1, δ_2 and δ_3, where $0 < \delta_1 < \delta_2 < \delta_3 < K-1$. These thresholds can be 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 lines 17-31 of Algorithm 1.

C. Load Control Algorithm

Each node i initializes its source rate T_i randomly from the interval $[T_i^{min}, T_i^{max}]$. Nodes may have different T_i^{min} and T_i^{max} values. The packet generation rate for node i is given by $\lambda_i = T_i/D$. The value of 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. 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. Figure 9 depicts a sample trace of $\tilde{q}_i$ and the corresponding T_i, in the case of CC bottleneck.

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 (10).
while (time<SIMTIME):
1: if $R_{ctrl} < R_{ctrl}^{opt}$ and node $i$ receives its own CTS then
2:     //Control channel bottleneck
3:     $l = l + 1$;
4:     Estimate the current CCD $\tau_i(l)$;
5:     Compute the average queue length $q_i(l) = \tau_i(l)/E[\tau_i](l) - 1$;
6:     Predict the queue length $\tilde{q}_i(l)$ using (11);
7:     if $\tilde{q}_i(l) < 1 - \delta$ then
8:         $T_i(l+1) = T_i(l) + D$;
9:     else if $\tilde{q}_i(l) \in [1-\delta, 1]$ then
10:        $T_i(l+1) = T_i(l) + D^2/T_i(l)$;
11:    else if $\tilde{q}_i(l) \in [1, 1+\delta]$ then
12:        $T_i(l+1) = T_i(l)$;
13:    else
14:        $T_i(l+1) = T_i(l)/2$;
15:    end if
16:    $T_i(l+1) = [T_i(l+1)]_{T_i^{min}}^{T_i^{max}}$
17: else if $R_{ctrl} \geq R_{ctrl}^{opt}$ and node $i$ receives an ACK then
18:    //Data channel bottleneck
19:    $l = l + 1$;
20:    Estimate the current ODC $o_i(l)$;
21:    Predict $\tilde{o}_i(l)$ using (12);
22:    if $\tilde{o}_i(l) < \delta_1$ then
23:        $T_i(l+1) = T_i(l) + D$;
24:    else if $\tilde{o}_i(l) \in [\delta_1, \delta_2]$ then
25:        $T_i(l+1) = T_i(l) + D^2/T_i(l)$;
26:    else if $\tilde{o}_i(l) \in [\delta_2, \delta_3]$ then
27:        $T_i(l+1) = T_i(l)$;
28:    else
29:        $T_i(l+1) = T_i(l)/2$;
30:    end if
31:    $T_i(l+1) = [T_i(l+1)]_{T_i^{min}}^{T_i^{max}}$
32: end if
```

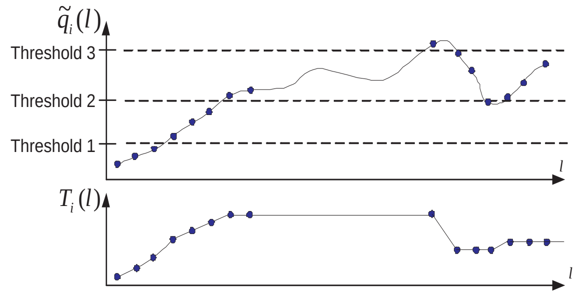


Fig. 9. A sample trace of the adaptive load control mechanism in the case of CC bottleneck.

IV. Performance Evaluation

In this section, we evaluate the performance of the proposed MAC protocol and the adaptive load control scheme. **We used CSIM (a C-based process-oriented discrete-event simulation package) to implement the Physical, MAC, and min-hop routing layers (see www.mesquite.com for details of CSIM). This gave us fine control on the working of the simulator, allowing us to account for all sources of interference in the network (including those that are outside the carrier sense range of a given node).** The main performance metrics are the end-to-end network throughput (total number of bits that are successfully received by their final destinations divided by the simulation time), the data and control collision rates (number of collided data/control packets per generated data packet), 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 i generates traffic according to a Poisson process of rate λ_i (packets/sec). The packet destination is randomly chosen from all other nodes in the network. Note that at high transmission rates, the destination node may be outside the transmission range of the source node. In this case, multi-hop transmission along the min-hop path is used. We use such a path for both data and control packets. For simplicity, we ignore the routing overhead. Unless stated otherwise, the default values for various network parameters are given in Table I. The simulations in Sections IV-A through IV-C are conducted assuming a channel access mechanism without the DTS packet. The impact of the DTS is studied under various delayed ACK options in Section IV-D.

A. *Impact of CC Transmission Rate*

We first study the system performance under different values of R_{ctrl}. We set $N_R = 4$. The different control rates and corresponding SNR$_{th}$ values are summarized in Table II. Based on the values in Tables I and II, it is easy to show that all nodes can hear each other when $R_{ctrl} = R1$, thus simulating a single-collision domain. Increasing R_{ctrl} results in a smaller transmission range for control packets, which may give rise to hidden-terminal problems. From the analysis in Section II-E, the value of R_{ctrl}^{opt} is 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.

Figure 10(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 a saturation point. **Increasing** R_{ctrl}

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
Maximum RTS retransmissions (RTS$_{\max}$)	7
Maximum Data retransmissions	3
$B_{\min}$	20 μs
$B_{\max}$	160 μs
Power inflation factor (δ)	0.2
γ	4

TABLE II

TRANSMISSION RATES FOR THE CC

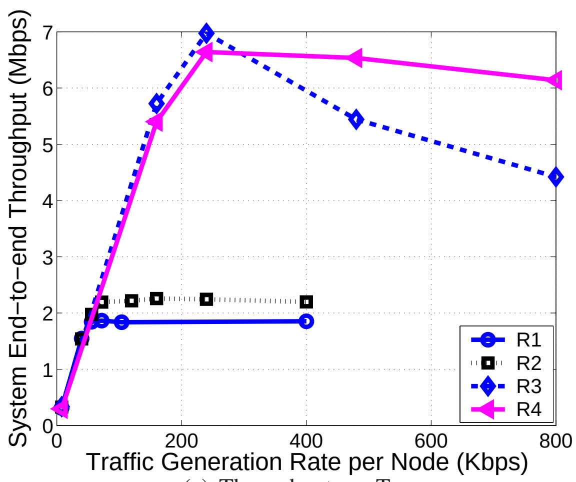
Rate Index	Rate	SNR$_{th}$
$R1$	78 Kbps	0.26
$R2$	200 Kbps	0.67
$R3$	2.5 Mbps	5.90
$R4$	5 Mbps	15.73

from $R1$ to $R3$ alleviates the CC crowdedness, and results in marked throughput improvement. At first, this may seem counter-intuitive, as a larger R_{ctrl} increases the likelihood of hidden terminals and packet collisions. However, this effect seems to be dwarfed by the additional packets that can be pushed into the network when a larger R_{ctrl} is used (as long as the CC is the bottleneck). This trend does not continue indefinitely; at some point, the network connectivity may be lost due to the shorter range of control packets associated with higher transmission rates. For $R3$ and $R4$, 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.

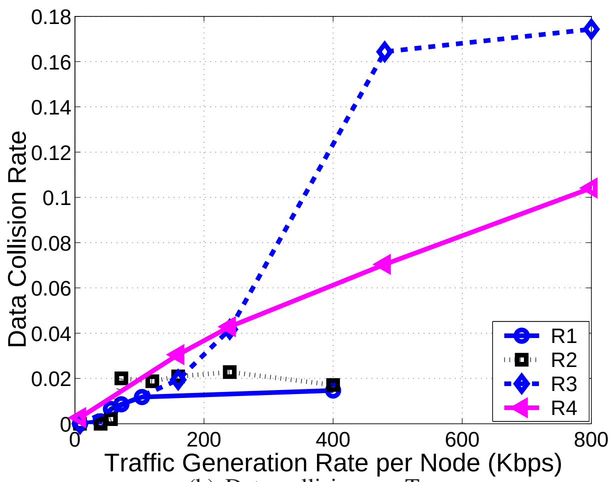
Figure 10(b) depicts the data-packet collision rate under different R_{ctrl} values. Before the saturation point, the data collision rate is relatively small. After the saturation point, this rate becomes high for $R3$ and $R4$, because data channels are saturated in this case. This also explains the drop in throughput after the saturation point in Figure 10(a).

Figure 10(c) shows the average energy consumption per successfully received one data bit. More energy is needed at low R_{ctrl}. The reason is that 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.

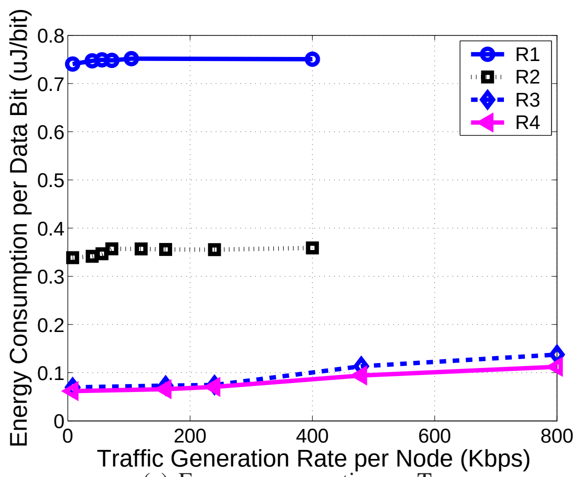
Figure 10(d) shows the end-to-end delay versus T. As expected, the end-to-end delay is relatively low (less



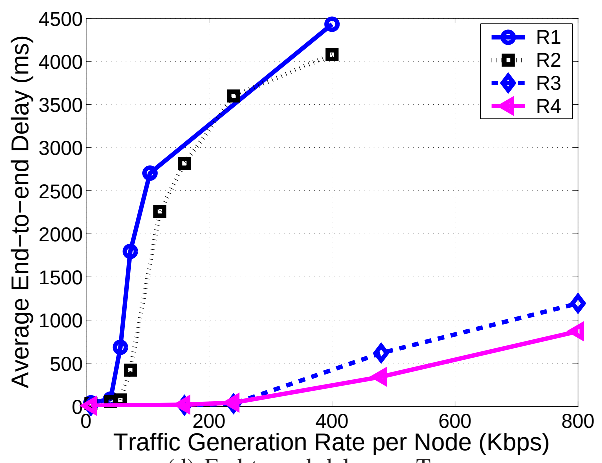
(a) Throughput vs. T.



(b) Data collision vs. T.



(c) Energy consumption vs T.



(d) End-to-end delay vs. T.

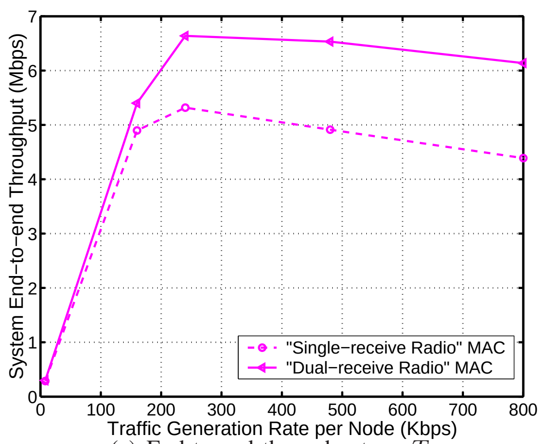
Fig. 10. Performance as a function of the traffic load for different CC transmission rates.

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 destination, the cumulative end-to-end delay is still smaller than that at low R_{ctrl}.

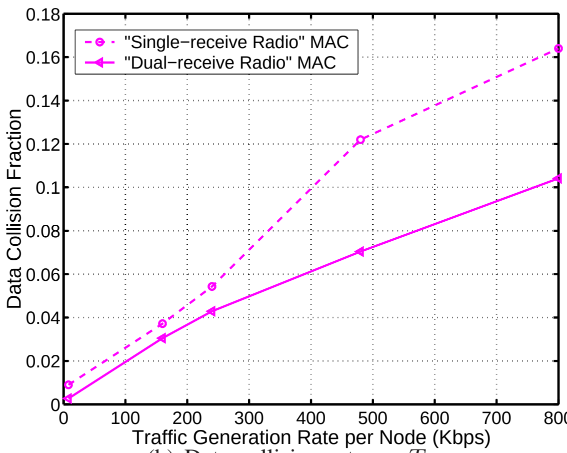
B. Performance Comparison Between Single-receive and Dual-receive Radios

In this experiment, we compare the performance of our MAC protocol under two types of radios: (1) single transceiver radios with dual-receive capability, and (2) single transceiver radios with only one receive branch. In the latter case, the radios operate the traditional 802.11-based MAC protocol but over multiple data channels. We choose $R4$ as the control rate, thus simulating a multi-hop scenario.

Figure 11(a) depicts the end-to-end throughput versus T. It shows that the MAC with single-receive radios reaches the saturation point much sooner than the "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 Figure 11(b), where the "single-receive radio" MAC shows a large number of data (and control) collisions. As mentioned before, the dual-receive capability represents a simple enhancement in the receiver design. Our multi-channel MAC design fully exploits this 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 in the design, the performance improvement due to the dual-receive radio MAC would not be significant.



(a) End-to-end throughput vs. T.

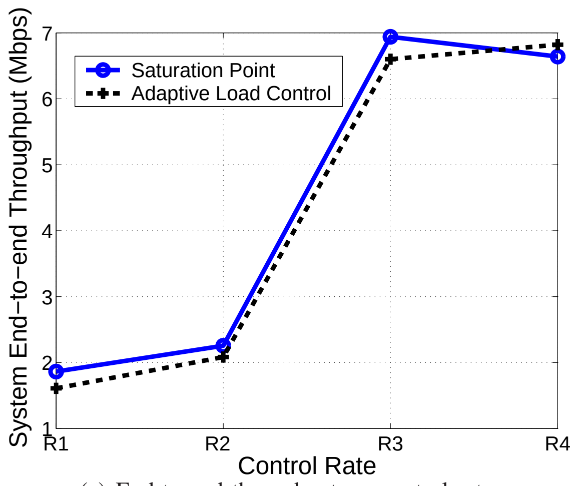


(b) Data collision rate vs. T.

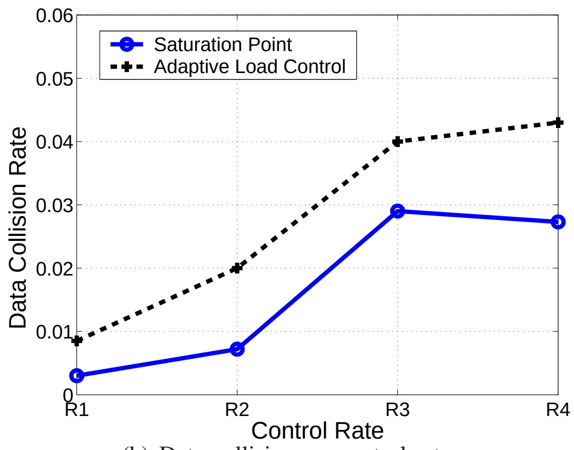
Fig. 11. Performance comparison between single- and dual-receive MAC designs.

C. Adaptive Load Control Performance

In Figure 10, 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 III, and study the effectiveness of our adaptive load control mechanism by comparing its performance with the one achieved at the saturation point, denoted by $T_{tot}^{(sat)}$. This $T_{tot}^{(sat)}$ is obtained offline by gradually increasing T for all nodes and observing the resulting end-to-end throughput. **The T value that maximizes the throughput is taken as $T_{tot}^{(sat)}$.**



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



(b) Data collision vs. control rate.

Fig. 12. Performance of the adaptive load control mechanism for different CC transmission rates.

Figure 12(a) depicts the end-to-end throughput for four CC transmission rates. Note that $T_{tot}^{(sat)}$ may not be the maximum system throughput ($T_{tot}^{(opt)}$), 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 $T_{tot}^{(opt)}$ is intractable even with a centralized controller. In fact, $T_{tot}^{(sat)}$ is a good approximation of $T_{tot}^{(opt)}$ for $R1$ and $R2$, but is smaller than $T_{tot}^{(opt)}$ for $R3$ and $R4$ because of the dropping effect at high loads, as shown in Figure 10(a). This is the reason that the throughput using load control and $R_{ctrl} = R4$ can be better than $T_{tot}^{(sat)}$. Figure 12(a) shows that the system throughput using load control is more than 90% of $T_{tot}^{(sat)}$.

Finally, Figure 12(b) shows that the data collision rates under adaptive load control mechanism are relatively low, which proves the efficiency of this mechanism.

D. *Impact of the DTS Packet*

In this section, we evaluate the performance implications of incorporating the DTS packet into the channel access process, as explained in Section II-F. We let $N = 50$, $R_{ctrl} = 2.5$ Mbps, and $K = 6$ (number of data and control channels). We also take $N_R = 8$ (the number of transmission rates), ranging from 39 kbps for the lowest rate to 5 Mbps for the highest rate. The traffic generation process is modified slightly to allow for multi-hop flows (with $R_{ctrl} = 2.5$ Mbps, most source-destination pairs are non-neighbors, necessitating multi-hop operation). Specifically, for a given randomly generated source-destination pair, we generate a stream of packets (which we refer to as a flow) from the given source to the given destination. The number of packets in this flow is sampled from an exponential distribution of mean 100 packets. The size of each packets is $D = 8000$ bits. For a given flow, the inter-packet times are exponentially distributed, with mean that is adjusted to produce a given traffic load (in bits/sec). Once a flow terminates, its source randomly generates a new flow and a new destination.

Figures 13 to 16 depict the resulting performance for the three DTS variants and for the original (no DTS) access scheme. As shown in Figure 13, when the traffic load is light (e.g., 80 kbps per node), the overhead of the DTS packet outweighs its benefit. As the load increases, the DTS-based schemes outperform the no-DTS scheme. This is especially true for the ‘partial-wait’ DTS, which gives about 18% improvement in throughput when the per-node traffic load is 800 kbps. The impact of the DTS packet on the packet drop rate is shown in Figure 14. It is very clear that the various DTS-based schemes result in a significant reduction in the packet drop rate. Nonetheless, such reduction comes at the expense of a more conservative protocol behavior (fewer transmission attempts), which hampers the overall improvement in throughput. Interestingly, while the use of the DTS packet reduces the data-packet collision rate (Figure 15), it actually increases the collision rate over the CC (see Figure 16). Note that the absolute numbers of dropped packets and data/control collisions increase with the traffic load but in a sub-linear fashion, which explains why the packet drop and collision rates (normalized with respect to the total number of generated data packets) decrease with the load.

V. 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. Specifically, the multi-channel hidden-terminal problem is alleviated by exploiting the dual-receive capability of the radios, the transmitter deafness problem is solved by using a delayed ACK mechanism, and the control channel bottleneck problem is overcome by assigning an optimal control rate. 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. In order for the system to operate around its saturation point autonomously, we also proposed a cross-layer framework for joint adaptive load control and medium access control. In this framework, the traffic loads of individual

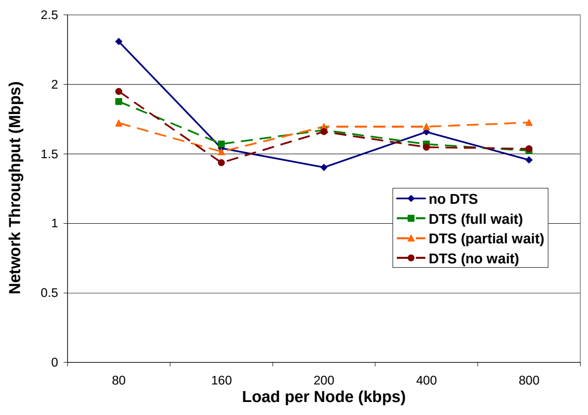


Fig. 13. Network throughput vs. traffic load.

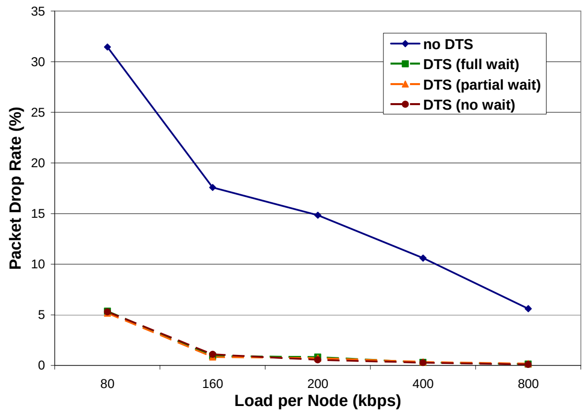


Fig. 14. Percentage of dropped packets vs. traffic load.

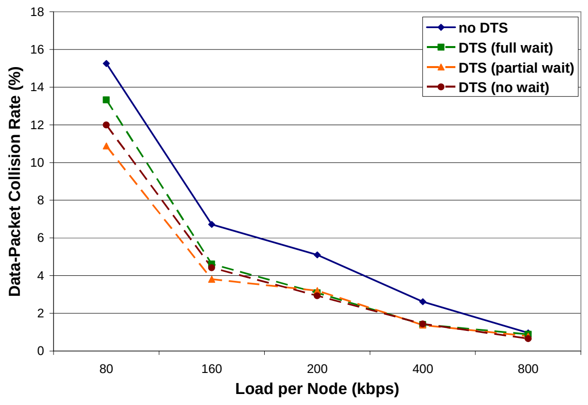


Fig. 15. Data-packet collision rate vs. traffic load.

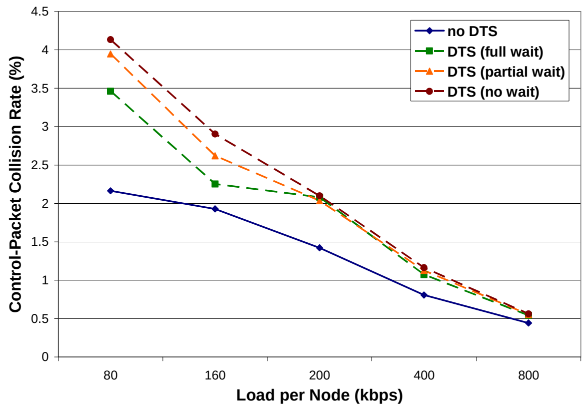


Fig. 16. Control-packet collision rate vs. traffic load.

nodes are adapted based on the values of local MAC parameters. 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. 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] Simon Haykin. Cognitive radio: Brain-empowered wireless communications. *IEEE Journal on Selected Areas in Communications*, 23(2):201–220, February 2005.

[2] *IEEE 802.22 Working Group on Wireless Regional Area Networks*. http://www.ieee802.org/22/.

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

[5] N. Nie and C. Comaniciu. Adaptive channel allocation spectrum etiquette for cognitive radio networks. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (IEEE DySPAN)*, November 2005.

[6] P. Pawelczak, R. Prasad, X. Xai, and I. Niemegeers. Cognitive radio emergency networks - requirements and design. In *Proceedings of the IEEE DySPAN Conference*, pages 601–606, November 2005.

[7] S. Sankaranarayanan, P. Papadimitratos, A. Mishra, and S. Hershey. A bandwidth sharing approach to improve licensed spectrum utilization. In *Proceedings of the IEEE DySPAN Conference*, pages 279–288, November 2005.

[8] L. Ma, X. Han, and C.-C. Shen. Dynamic open spectrum sharing MAC protocol for wireless ad hoc networks. In *Proceedings of the IEEE DySPAN Conference*, pages 203–213, November 2005.

[9] Srinivasan Krishnamurthy, Mansi Thoppian, S. Venkatesan, and Ravi 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.

[10] 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.

[11] Y. Xing, C. Mathur, M. Haleem, R. Chandramouli, and K. Subbalakshmi. Dynamic spectrum access with QoS and interference temperature constraints. *IEEE Transactions on Mobile Computing*, 6(4):423–433, 2007.

[12] Y. Xing, R. Chandramouli, S. Mangold, and S. Shankar. Dynamic spectrum access in open spectrum wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(3):626–637, 2006.

[13] Carlos Cordeiro and Kiran 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.

[14] Yuan 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.

[15] Fan Wang, Marwan Krunz, and Shuguang 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.

[16] Yuan Yuan, P. Bahl, R. Chandra, P. Chou, J. Ferrell, T. Moscibroda, S. Narlanka, and Y. Wu. Knows: Kognitive networking over white spaces. In *Proceedings of the IEEE DySPAN Conference*, pages 416–427, April 2007.

[17] H. Bany Salameh, M. Krunz, and O. Younis. Distance- and traffic-aware channel assignment in cognitive radio networks. In *Proceedings of the IEEE SECON Conference*, June 2008.

[18] H. Bany Salameh and M. Krunz. Channel access protocols for multihop opportunistic networks: Challenges and recent developments. *IEEE Network – Special Issue on Networking over Multi-hop Cognitive Networks*, 23(4):14–19, July 2009.

[19] I. Akyildiz, W.-Y. Lee, M.C. Vuran, and S. Mohanty. Next generation dynamic spectrum access cognitive radio wireless networks: A survey. *Computer Networks*, 50(13):2127–2159, 2006.

[20] Haythem Bany Salameh, Marwan Krunz, and Ossama Younis. MAC protocol for opportunistic cognitive radio networks with soft guarantees. *IEEE Transactions on Mobile Computing (TMC)*, 8(10):1339–1352, October 2009.

[21] 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.

[22] N. Jain, S. Das, and A. Nasipuri. A multichannel CSMA MAC protocol with receiver-based channel selection for multihop wireless networks. In *Proceedings of the 9th Int. Conf. on Computer Communications and Networks (IC3N)*, pages 432–439, October 2001.

[23] 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 multi-hop mobile ad hoc networks. In *Proceedings of the 2000 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '00)*, Dec. 2000.

[24] *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.

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

[26] 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.

[27] 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.

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

[29] 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.

[30] 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.

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

[32] Loukas Lazos, Sisi Liu, and Marwan Krunz. Spectrum opportunity-based control channel assignment in cognitive radio networks. In *Proceedings of the IEEE SECON Conference*, Rome, Italy, June 2009.

[33] Fan Wang, Marwan Krunz, and Ossama Younis. Throughput-oriented MAC for mobile ad hoc networks: A game-theoretic approach. *Ad Hoc Networks Journal*, 7(1):98–117, 2009.

[34] 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.