Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2009, Article ID 174730, 12 pages
doi:10.1155/2009/174730

Research Article

Borrowed Channel Relaying: A Novel Method to Improve Infrastructure Network Throughput

Aaron Jow and Curt Schurgers

Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093, USA

Correspondence should be addressed to Curt Schurgers, curts@ece.ucsd.edu

Received 30 April 2009; Revised 31 August 2009; Accepted 25 November 2009

Recommended by Ingrid Moerman

From a networking perspective, the chief impediment to throughput enhancement in infrastructure networks such as IEEE802.11 is the access point bottleneck: all traffic to, through, and from the network has to pass through this access point. When some clients experience poor channel conditions and therefore communicate at a lower data rate, this severely impacts the throughput of all clients in the network. Recently, multihop relaying in combination with leveraging multiple data rates was proposed to alleviate this problem. However, our experiments indicate that gains from these techniques are very small with realistic positioning of clients. Instead, we propose a novel scheme that combines relaying and multiple data rate capabilities with the concept of channel borrowing. Our protocol, BCR (Borrowed Channel Relaying), utilizes unused capacity from neighboring access points and is able to achieve network throughput gains of 20% to 60% depending on the scenario. Although we use 802.11 style networks to illustrate this concept, this general principle can be applied to any infrastructure network with receivers capable of tuning to more than one channel.

1. Introduction

Current 802.11 wireless networks are typically infrastructure-based, where an Access Point (AP) and its clients form a Basic Service Set (BSS). Virtually all traffic within these networks is between the AP and its clients. To leverage the fundamental tradeoff between channel quality and transmission speed, 802.11 devices are designed with the ability to operate at different data rates. This capability, known as rate diversity, allows an 802.11 device to use more efficient modulation schemes during better channel conditions. Unfortunately, in practical networks, rate diversity was observed to cause a detrimental effect, called the “performance anomaly” [1]. This anomaly refers to the fact that when one client throttles back its data rate, the throughput of all clients serviced by the same Access Point (AP) suffers substantially: the fast clients see their throughput decreased to a rate comparable to that of the slow client. This is illustrated in Figure 1, which shows the throughput of a fast 802.11b client versus the total number of clients in the system, for two cases: one with one slow client (1 Mbps), and one where all clients are fast (11 Mbps). Regardless of how many clients are in the system (the value on the x-axis), just one client becoming slow moves the performance of all other clients from the top line to the bottom line. The reason is that a slow link transmitting at 1 Mb/s captures the channel eleven times longer than a link using 11 Mb/s.

To alleviate this performance anomaly, relaying was proposed as an effective mechanism [2, 3]. This principle is illustrated in Figure 2 for the case of downlink traffic, that is, from the AP to a client. Instead of using a slow link between the AP and client, the use of a relay (i.e., another client acting as a forwarder node) can improve the overall throughput. The idea is that two fast transmissions (AP to relay and relay to client) can occupy less time than a single slow transmission, therefore freeing channel access time for other transmissions. Although significant benefits are attainable in principle, relaying hinges solely on the availability of relay nodes with good channels. To evaluate relaying in a more realistic setting, we performed actual field experiments using our own 802.11b platform, called

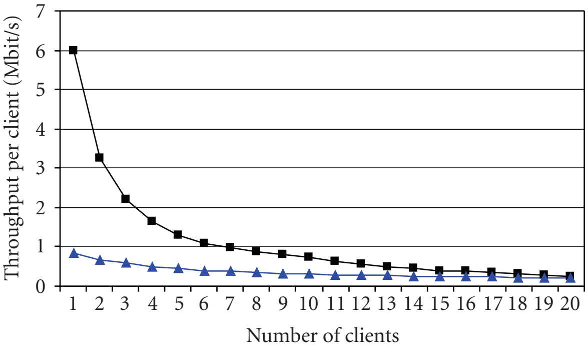

FIGURE 1: Throughput per client. Top line: All fast nodes. Bottom line: One slow node, all others fast.

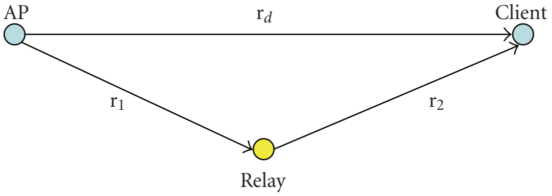

FIGURE 2: Illustration of relaying.

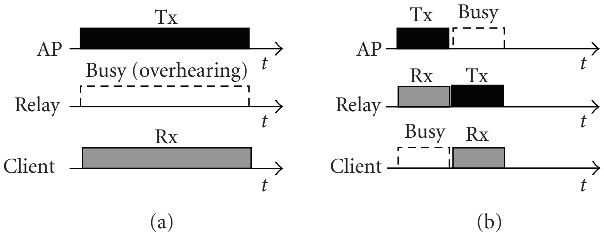

FIGURE 3: Radio Usage for Direct Transmission (a) and rDCF (b).

CalRadio [4]. Unfortunately, our versus-range experiments indicated that the benefits of relaying are minimal except for very specific scenarios where relay nodes are placed optimally as we will show in Section 5.3.

However, we have also discovered that relaying can be much more effective if cleverly combined with channel borrowing between access points. The idea is that if one Basic Service Set (BSS), a group of nodes consisting of an AP and its clients, is lightly loaded, the channel assigned to it could be used by a neighboring BSS to significantly improve the performance of relaying. This will alleviate the transmission bottleneck around the access point, which was found to be the main obstacle to throughput enhancement. In this paper, we will introduce this novel concept of relaying with channel borrowing, propose a practical protocol solution, and demonstrate its ability to achieve significant overall throughput performance enhancement.

2. Motivation

Before getting to the details of our relaying with channel borrowing, we will take a small step back to list some of the different concepts related to relaying and rate diversity that have been proposed to enhance throughput in 802.11-type networks. One of the first ideas was to leverage multihopping for spatial reuse. In a wireless network, using multiple short hops can increase the number of simultaneous transmissions if combined with power control (note that this work assumed that all nodes have a fixed data rate, so it does not exploit rate diversity) [5]. However, it was also observed that if most traffic goes to and from an access point, spatial reuse alone does not lead to an increase in network throughput [5]. The reason is that the access point remains the bottleneck, and each path is basically contending for the same resources, much the same way direct transmissions would. The first main point to realize is that infrastructure networks thus behave fundamentally differently than mesh networks: the access point is the true bottleneck and spatial reuse by itself does not alleviate this problem!

It was realized that another motivation for using multihop routes is to overcome bad channel conditions [6]. This could be thought of as a generalization of range-extension. Since in a typical wireless local area network (WLAN) scenario, devices support multiple data rates; the best rate can be chosen on each link. With such use of rate diversity, a succession of multiple fast hops can free up time on the channel compared to one slow hop. In a scenario where traffic flows to and from an access point, this approach can alleviate the observed bottleneck there, which is the throughput limiting factor in these types of networks. Specific protocols, called rDCF (relay distributed coordination function) and rPCF (relay point coordination function), have been proposed to take advantage of this situation [2, 3]. They are modifications of the 802.11 standard to include such relaying in combination with rate diversity, and basically explore the use of a single relay in a 2-hop path as was illustrated in Figure 2. However, we have found through experimental testing that while these protocols can increase network throughput in carefully constructed scenarios, the amount of throughput gain realized in a network with random node placement is low. The reason is that the probability of having two fast hops to replace a much slower one is small, given the specific dependence of rate versus distance for typical 802.11 devices. We will further illustrate this in Section 4.

Figure 3 illustrates the operation of these protocols, as well as their benefits and drawbacks. Figure 3(a) shows the scenario where the AP sends directly to the client; the link is assumed slow and communication therefore takes up a significant amount of channel time. Note that during this transmission, the relay node experiences a busy channel and would be unable to receive or send data to any other node. In Figure 3(b), we depict the principle of rDCF. By using the relay as an intermediate hop, the slow communication is replaced by two faster ones, thereby giving other nodes more time to access the channel. In practice however, the possible gains are small, as it is not very likely that two transmissions will take much less time than a single slow one. On the flip side, this figure also shows the key element we will exploit in our scheme. During the second communication, the AP sits idle, as it would interfere with the relay-client communication. However, it is also not doing anything useful. As the AP is really the bottleneck in these networks, having the AP sit idle has a direct negative impact on the overall throughput. Therefore, conceptually, we must devise a mechanism to allow the access point to remain busy most of the time by servicing another client during the second packet transmission in Figure 3(b). This is possible by changing the relay-client transmission such that it does not interfere with the AP, by using a separate channel.

The main problem with the relaying setup in rDCF and rPCF is that of the two hops in the path (AP-relay r_1 and relay-client r_2), only one of them can be active at the same time. As such, the combination of both hops, in terms of time-on-the-channel, has to outperform the direct link (r_d). For example, in Figure 2, if both r_d and r_2 are slow, relaying will not be beneficial, even if link r_1 from the AP is fast. Since the access point is the actual bottleneck, ideally one would hope to be able to exploit the fast link r_1 from the AP. Basically, due to its bottleneck status, any throughput gain for the AP translates into similar benefits to the network. The problem with rDCF and rPCF is that the second hop forces the AP to sit idle, thereby wasting valuable resources. In this paper, we solve this problem by leveraging yet another degree of freedom in 802.11 networks: the channel assignment. Current 802.11 infrastructure networks are frequency planned with static partitioning of spectrum. The total of spectrum available to the network as a whole is split up in channels. For example, there are 3 nonoverlapping channels for 802.11b in the United States. Other variants of 802.11, such as 802.11a in the 5 GHz band, have 12 nonoverlapping channels, which is even more beneficial for our protocol as we will see in this paper. Each BSS is typically assigned just one channel, so a heavily used AP can have its single channel nearly maxed out while a nonoverlapping channel assigned to a neighbor BSS sits idle. By allowing a single BSS to use more than one channel, more than one transmission can occur simultaneously, therefore improving throughput. This is the basis of the novel approach we propose in this paper: relaying and rate diversity combined with borrowing channels between BSS'. Each node (AP, relay or client) still only uses one channel at a time. However, the AP-relay links and relay-client links operate on different channels through careful coordination. In certain common scenarios, such as an AP in a busy hotel conference room next to lightly used APs in hallways and guest rooms, our protocol is likely to work well because the probability that additional channels will be available at any given time is high. In [7], some of our colleagues monitored the load on 4 APs providing service to attendees of an ACM conference at UC San Diego. They found that most of the long sessions (devices associated with an AP for greater than 10 minutes) have very low average data rates, which they conclude to be machines of users who associate with an AP and then remain idle as they pay attention to the conference sessions. They also found that load distribution at the APs was very bursty and uneven and was governed mostly by individual users and workloads rather than the number of users associated at any given time. We would like to point out that putting two transceivers on the AP, although also directly attacking the bottleneck in these systems, is not an option. The reason is that self jamming will occur if one transceiver is transmitting while the other is receiving [8]. Though they would use different channels, they operate in the same band, very close in frequency. The transmitter can easily be 50 dB or more above the received signal and even the best filters cannot handle this type of interference almost directly adjacent to the received channel. This is in contrast to systems that are designed to transmit and receive at the same time, like cellular base stations. Unlike 802.11, these systems have the benefit of operating in paired spectrum, where transmit and receive frequencies are separated by a guard band relatively large compared to the channel size to prevent self jamming. Our usage of relay nodes essentially provides the means to do more than one simultaneous transmission without special hardware, within the existing frequency band structure, while avoiding self jamming.

3. Related Work

Besides 802.11 networks, the idea of using relays to improve throughput has also been proposed for cellular networks. Most 2G cellular networks are frequency planned systems just like 802.11, with a static set of channels allocated to each cell. If one cell is heavily loaded while neighbor cells have light traffic, spectrum is not being used efficiently. Methods by which one cell can borrow unused spectrum from neighboring cells have been developed [9, 10]. Later works have looked at the use of relays to improve throughput in 3G networks, which are all based on CDMA. The general idea is to use relays to forward packets to clients with poor channel quality [6]. Mobile devices with more than one interface can be used as relay stations and the relayed data can be sent between the mobile devices and the relay stations as either an in-band (licensed) transmission or an out-of-band (unlicensed) transmission. Cooperative diversity techniques, which can take advantage of the same types of diversity that MIMO uses but do not require more than one antenna on each node, can be used. In [11], the authors argue that using more than two hops results in only marginal throughput improvement but greatly increases the complexity of the routing algorithms, and found no reason to investigate algorithms with 3 or more hops in detail. They also found that two of the relay protocols can approximately double the throughput of 3G networks. It should be noted that [6, 11] rely on having more than one wireless interface per node, while our work assumes only one interface per node.

In 802.11, each node has an equal *opportunity* to gain access to the channel in any interframe space between packets, which translates to equal throughput if the packet size used by all nodes is fixed and each transmits as fast as possible. Some protocols, however, such as the Opportunistic Auto Rate (OAR) [12], attempt to redefine fairness as each node receiving a throughput proportional to the data rate, which roughly translates to an equal amount of time on (rather than opportunity to access) the channel. However, the AP remains a system bottleneck and all the time it spends transmitting to or receiving from nodes with a slow connection is still spent sending relatively few bits per unit time.

Work has also been published about the capacity of wireless networks using multiple channels [13–15]. People have created MAC layers for 802.11 networks that use more than one channel [16]. In their simulations, traffic patterns did not necessarily reflect those of typical infrastructure networks with most of the traffic originating or terminating at a single node, and rate diversity was not considered. Mobile Ad Hoc Network (MANET) systems with multiple channels have been explored [17]. Another idea proposed is allowing traffic to pass through neighboring APs by having nodes explicitly switch channels since many nodes are in the transmission range of more than one AP [18]. This protocol was extended by allowing clients to relay and switch between various BSSes [19]. Rate diversity was not considered in either. The work in [20] takes rate diversity into account and uses a network of mesh routers to connect end users to gateways. Time is divided into superframes and each mesh node has a designated receive channel in each receive timeslot, with the remaining slots reserved for transmission. The receive pattern of each node is known to its neighbors, and a node wishing to relay a packet through a mesh node needs to transmit on the designated channel in a receive timeslot for the destination node. To help avoid conflicts, part of each superframe is reserved for downlink traffic and part for uplink traffic. Although this protocol outperformed the single channel network by up to 100%, it relies on the assumption that traffic loads across the network are steady over time. This is not the case in real networks, where bursty traffic might generate flows that are not optimal for the superframes, and the system would operate inefficiently until it could have time to adapt to the new traffic flows. Also, this protocol relies a reservation MAC, unlike our protocol which relies on the far more prevalent random backoff MAC and is therefore more suited for typical wireless networks and traffic patterns. We should emphasize that the major difference between our work and other multichannel MAC protocols is that while they focus on mesh networks with randomly generated one-to-one traffic, we focus on WiFi infrastructure networks with very different traffic patterns and a distinct bottleneck at the access point.

Other protocols require each device to have two transceivers [21]. This is the case in the vast majority of multichannel protocols. In [22], nodes are equipped with as many radios as necessary to make optimal use of the channels. The authors found that in general, nodes closer to the gateways need more radios (as they will be relaying more packets than nodes at the fringes of the networks). Besides the problem that this protocol requires special hardware, it is simply not practical to expect the nodes with the greatest number of radios to be located near the gateways in networks with mobile nodes.

Finally in [23], nodes can be equipped with as many radios as needed to optimize use of the available channels (at which point the number of simultaneous transmissions is limited by interference). Although throughput can be improved by as much as 4 to 7 times that of a single channel network, this required each node to be equipped with an average of 4 radios!

While using multiple radio interfaces works in theory, in practice the antenna location, channels used, and filters must be carefully controlled to avoid self jamming, and it is not practical because it both adds to the cost of the hardware and is not compatible with preexisting infrastructure.

4. Protocol

Our method improves upon the rDCF and rPCF [2, 3] protocols by reducing the amount of time the AP's channel is occupied during the relay process. Since the AP is the bottleneck, the worst thing we can do is force it to sit idle, which is precisely what happens in rDCF and rPCF when packets are being relayed, because each BSS in rDCF and rPCF is allowed to use only one channel. Unlike many other multichannel MAC protocols, we assume that each node has only one transceiver, which makes our protocol possible to implement on existing hardware and does not increase the cost of the equipment required to use it.

4.1. General Concept. As mentioned earlier, the typical large 802.11 network is a frequency planned system, and is installed in such a way that adjacent access points do not use overlapping channels to avoid interference. Our protocol uses the idea that a BSS can temporarily borrow an underused channel from a neighbor for the purposes of relaying packets. This allows relaying to be accomplished without forcing the AP to sit idle while the relay node is communicating with the client node. To do this, one AP would ask its neighbor permission to use the channel over the wired infrastructure. Since this process does not use the wireless channel, it introduces no wireless overhead. Because most internet traffic is in the downstream direction, we focus on relaying packets going from the AP to its clients. Since changing channels can impose a time overhead of up to 200 microseconds [24], we keep the AP tuned to its own channel at all times and require both the relay and the destination nodes to change channels so that this time overhead does not affect the bottleneck node.

4.2. Step by Step: Relaying a Packet. Once the AP identifies a packet as one that can be relayed, it selects a relay node, asks a neighboring AP for permission to use its channel, and then begins the process of relaying the packet. Figure 4 illustrates this process after a relay node has been chosen and a neighbor AP has granted permission to use a second channel.

(i) The AP sends a Relay Data (RDATA) frame to the chosen relay. This packet contains a field that allows the AP to tell the relay node which channel to use when it forwards the packet.

(ii) The relay sends a Request-to-send, Borrowed Channel (RTSBC) frame to the destination node, indicating that it intends to send a relayed packet on another channel and therefore the destination node should retune its transceiver to that channel. The

RTSBC is also overheard by the AP, and serves as the acknowledgement for the RDATA frame (indicated by the dotted line in the figure). As such, the RTSBC is sent an SIFS interval after the RDATA frame. When the AP receives the RTSBC frame, it knows that the relay is in progress and adds both the relay and destination nodes to its forbidden list (this is intended to prevent the AP from attempting to transmit to nodes that cannot hear it and is described in Section 4.4). The AP also sets a Forbidden List Timer for an amount of time longer than the relay is expected to take. If the AP does not receive a Relay ACK (RACK) within this time interval, it assumes that the relay has failed and removes both nodes from the forbidden list.

(iii) The destination node sends a Clear-to-send, Borrowed Channel (CTSBC) frame back to the relay node, acknowledging receipt of the RTSBC frame. Upon sending this packet, the destination node begins tuning its transceiver to the borrowed channel. As soon as it receives the CTSBC, the relay node also begins retuning. Similar to what happens in RTS/CTS exchanges, the CTSBC is sent out an SIFS interval after the RTSBC frame. Both the relay node and the destination node set a (Borrowed) Channel Timer when they finish retuning, for a time interval longer than the second hop is expected to take. If the second hop is not completed when this timer expires, both nodes tune back to the primary channel.

(iv) As soon as it completes the process of tuning to the borrowed channel, the relay node waits a PIFS and then sends an RTSBC frame to the destination node. This is done because any nodes using the borrowed channel (e.g., in another BSS) might be unaware that the channel has been reserved, and the RTSBC/CTSBC exchange prevents collisions. The reason we wait a PIFS is that we do not want to clobber any acknowledgements that might be sent in response to transmissions already in progress, but we want priority access to the channel after any transmissions finish (since the channel has, after all, been reserved).

(v) The destination node sends a CTSBC frame back to the relay node on the borrowed channel after an SIFS.

(vi) The relay node proceeds with the second hop transmission, sending the RDATA frame to the destination node an SIFS after receiving the CTSBC.

(vii) The destination node sends an ACK back to the relay node an SIFS after receiving the RDATA frame. As soon as it does, it begins tuning back to the primary channel.

(viii) The relay node, after receiving the ACK, also begins tuning its transceiver back to the primary channel.

(ix) As soon as it has tuned back to the primary channel, the relay node waits a PIFS and then sends a Relay ACK (RACK) back to the AP. This prevents the relay node from causing a collision with any existing transmission or its acknowledgement frame, but also gives it priority access over any new transmission.

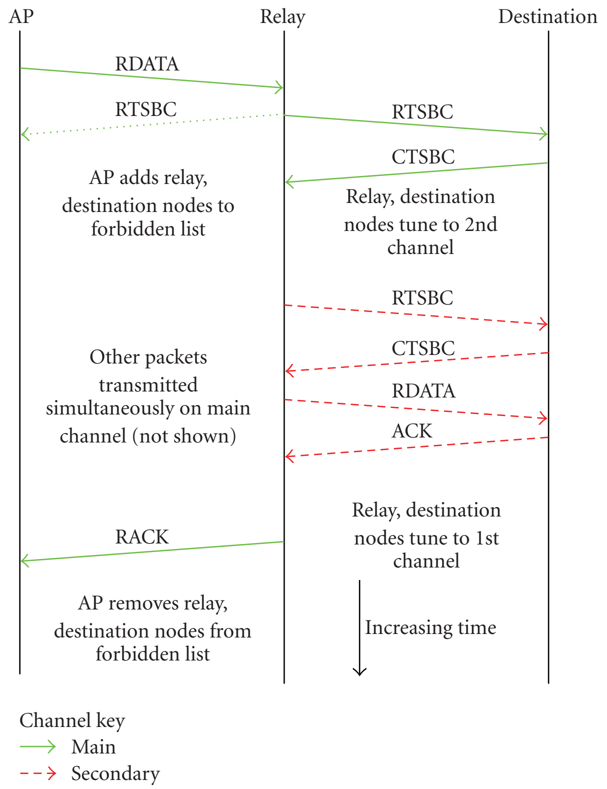
AP
Relay
Destination
RDATA
RTSBC
RTSBC
CTSBC
AP adds relay, destination nodes to forbidden list
Relay, destination nodes tune to 2nd channel
RTSBC
CTSBC
Other packets transmitted simultaneously on main channel (not shown)
RDATA
ACK
Relay, destination nodes tune to 1st channel
RACK
AP removes relay, destination nodes from forbidden list
Increasing time
Channel key
Main
Secondary

FIGURE 4: Step By Step Illustration: relaying a Packet. Primary channel transmissions as solid lines, borrowed channel transmissions as dashed lines.

(x) When the AP receives the RACK frame, it knows that both nodes are tuned to the primary channel again and removes both nodes from its forbidden list.

As already discussed in Section 4.2, implementing relaying in addition to channel borrowing required new 802.11 frames. For completeness, we list the new frames required to implement our protocol. These new frames add to the existing frames in the 802.11 specification. In order to implement our protocol, we define the following new types of 802.11 frames, using reserved subtypes in the 802.11 standard.

(1) RDATA (Relay Data): like a DATA frame, the Address 1 field contains the address of the receiver, the Address 2 field contains the address of the sender, and the Address 3 field contains the BSSID. On the first hop, for example, Address 1 holds the address of the relay and Address 2 holds the address of the AP. Likewise on the second hop, Address 1 holds the address of the relay and Address 2 holds the address of the destination node. The Address 4 field, not used in regular DATA frames unless the To DS and From DS bits are both set, is used in RDATA to hold the address of the destination node for both hops.

(2) RTSBC (Request-to-send, borrowed channel): serves the same function as a traditional RTS frame in addition to acting as a signal for the relay and client to switch channels, or to begin the second RDATA transmission. This frame has an extra field indicating the next channel the node should tune its receiver to. All nodes listen for RTSBC packets and set their NAV timers in the same way they do upon receiving an RTS frame, to avoid collisions. Note that the duration indicated in the frame on the main channel will be short, while on the secondary channel it will be long (effectively reserving the channel long enough for the RDATA frame and ACK to be sent).

(3) CTSBC (Clear-to-send, borrowed channel): contains the same information as a traditional CTS frame with the added field indicating the channel the node should tune its radio to.

(4) RACK (Relay ACK): acknowledgement frame to let the AP know that the relay was successful.

4.3. Choosing a Relay. In order to select a relay for a given transmission, the AP queries its global table of data rates between all pairs of nodes in the BSS. This information is gathered from signal strength data from each client node whenever those nodes overhear a transmission from any other node. Whenever an outgoing packet arrives at the AP's queue, the AP checks to see whether it can be relayed by going through the following steps.

(i) Is there a relay in progress? If there is, the borrowed channel is already being used and therefore is not available to relay the packet, so the packet is sent directly. If not, go on to the next step.

(ii) Is the direct link fast enough? If the direct link is slow enough that we can benefit from using a relay, go on to the next step.

(iii) Is there a suitable relay node? A node must exist that has a faster connection to the AP than the destination node, which means that sending a packet to the relay instead of directly to the destination node would take less of the AP's time. In the case more than one suitable relay node is found, the following procedure is used to select among them.

   (a) Any relay node with a faster connection to the AP (first hop rate) takes priority over potential relay nodes with slower first hop rates. This is because the AP is the bottleneck, and by choosing the fastest first hop rate, we are minimizing the time the AP spends transmitting the packet.

   (b) If there is a tie between nodes chosen in the previous stage, the node with the fastest second hop rate takes priority. This reduces the amount of time the borrowed channel must be used for relaying.

   (c) Any nodes tied after the previous steps are selected randomly with a uniform probability among each of them.

4.4. Forbidden Lists. Once the relay node is chosen, the AP begins the process of telling both the relay and client nodes to transmit the second hop packet on the borrowed channel. This poses a problem for the 802.11 protocol. Recall that if a transmission to a node goes unanswered, the 802.11 protocol assumes that the packet loss was due to network congestion and interference, and therefore increases the size of the contention window and then sets its backoff timer.

The problem is that when the relay and destination nodes are tuned to the borrowed channel, they are effectively out of range of the AP. Should a packet destined for one of these nodes reach the head of the AP's transmit queue, it will receive no response because neither of the nodes will be able to hear the transmission. The AP would incorrectly assume that the channel is congested, set its backoff timer, and hurt overall throughput.

To avoid this issue, we have the AP maintain a list of forbidden destinations. This list contains nodes that are unable to communicate with the AP because they are tuned to another channel, for example, the relay and destination nodes while they are tuned to the borrowed channel.

The transmit queue examines the forbidden list whenever a packet reaches the head of the queue. If the destination of the packet is not forbidden, the queue passes the packet to the AP for transmission. Any packets destined for forbidden destinations are ignored but maintain their position in the queue so that any packet that reached the head of the queue but was blocked receives the first priority as soon as its destination is cleared from the forbidden list.

4.5. Handling Dropped Packets. Unlike their wired counterparts, wireless networks always must be designed to operate in suboptimal propagation conditions and must be able to tolerate errors and dropped packets. For this reason, our protocol is designed to tolerate packet losses. For example, the AP should not leave the relay and destination nodes on its forbidden list forever if the relay fails, so a timer keeps track of the amount of time since the relay and destination nodes tuned to the second channel. If this time reaches a specified threshold and the AP has not received an indication that the relay has succeeded, it assumes failure and removes both nodes from the forbidden list. Likewise, the relay and destination nodes should not get stuck on the borrowed channel if the second hop fails, so both of them have a borrowed channel timer that gets set when they tune to that channel. If the second hop transmissions are not completed by the time it expires, the transmission is assumed to have failed and both nodes tune back to the primary channel. Here is how our protocol reacts to dropped packets at various stages of the relay process.

(i) Dropped RDATA frame from AP to relay: the relay will not send an RTSBC frame to the destination node. The RDATA frame will be treated like any unACKed packet, and the AP will backoff, then send the frame again.

(ii) Dropped RTSBC frame from relay to AP: the AP will not add the relay and destination nodes to its forbidden list, and may try to transmit to either node while they are tuned to the other channel. This scenario, fortunately, is unlikely because the RTSBC is sent out an SIFS after the RDATA frame.

(iii) Dropped RTSBC frame from relay to destination (primary channel): the frame is retransmitted until a maximum number of retries is reached, then the relay is aborted.

(iv) Dropped CTSBC frame from destination to relay (primary channel): the relay node would never receive the CTSBC and would just abort the relay process. The destination node, having sent the CTSBC, would tune to the secondary channel, but would never receive the second RTSBC or the RDATA. Eventually, the destination node's borrowed channel timer will expire, causing it to tune back to the primary channel.

(v) Dropped frames on secondary channel for second hop: if packets are dropped on the secondary channel for any reason, the borrowed channel timers will eventually expire, causing the nodes to tune back to the main channel.

(vi) Dropped RACK from relay to AP: if the RACK is dropped, the AP will eventually remove both the relay and destination nodes from its forbidden list when its timer expires.

4.6. Variations. It is important to note that our protocol relays one packet at a time. While this minimizes latency, since the channel switching time is 200 microseconds [24], this can be a rather large penalty to pay twice for each packet relayed (for switching the radios on the relay and destination nodes to and from the borrowed channel). Each transmission also incurs the overheads associated with the RTSBC/CTSBC/RACK exchange with the AP, as well as the associated interframe spacings. The overhead as a percentage of the total amount of data sent can be reduced, particularly for large data transfers, by relaying more than one packet at a time. This can be done by sending data packets back to back in groups, separated by an SIFS interval, as is done in the OAR protocol [12].

5. Results

5.1. CalRadio. Since the performance of our protocol can be heavily impacted by the distances over which particular data rates can be used, we first needed to make sure that we had an accurate representation of data rate versus distance data for typical 802.11b transceivers. The first experiments we ran were outdoor experiments with a device called CalRadio. CalRadio is our 802.11b research platform with a software programmable MAC developed at CalIT2 at UCSD and is pictured in Figure 5. The purpose of these experiments was to get a real world measurement of packet error rates with respect to distance [4]. Figure 6 shows the measured data from our CalRadio experiments. The maximum distances between nodes at which each data rate can be used without having significant transmission errors are summarized in Table 1. We used these distances in our simulations to ensure that they would reflect reality to the greatest extent possible. Since the maximum usable data rate is determined by the distance between two nodes, the simulated results of our protocol, as well as those of rDCF and rPCF, strongly depend on accurately measuring these distances. We would also like to note that while we use an 802.11b network along with our experimental results to demonstrate the utility of our protocol, this protocol can be used with any wireless network with multiple data rates. Some other standards, such as 802.11a and 802.11g, have data rates that drop off even more dramatically with distance, which are exactly the conditions our protocol needs to perform well, but we use 802.11b in our simulations because our real world experiments produced results for 802.11b.

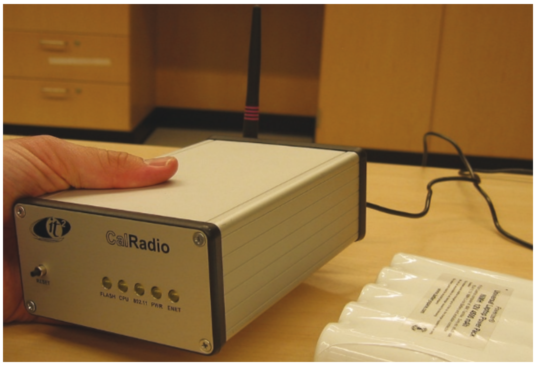

Figure 5: CalRadio external case and battery.

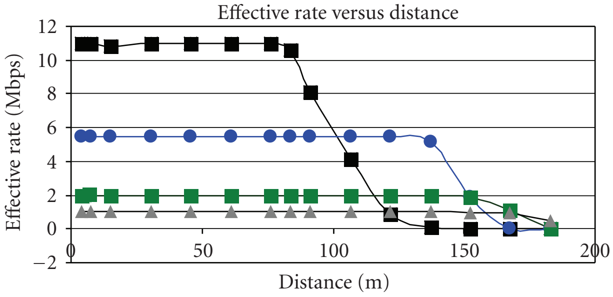

Figure 6: Effective throughput versus distance for CalRadio experiments.

Table 1: CalRadio experiment data rates versus distances.

Data rate	Maximum distance (m)
1 Mbps	164
2 Mbps	150
5.5 Mbps	130
11 Mbps	82

5.2. Matlab. One way to evaluate our protocol, assuming the vast majority of the traffic is downstream as it is in most networks (from the AP to the clients), is as a flow problem where the flow of data is from the AP “source” to the client “sinks”, both directly and through intermediate nodes (the relays). In order to be able to cast this scenario as a flow problem that can be solved with linear programming, we split each client node into two virtual nodes: a sink node, and a relay node. Each of the links between any nodes i and j has two values associated with it: $R(i, j) = R(j, i)$, the maximum data rate between nodes i and j, and $t_{ij} = t_{ji}$, the amount of time i and j communicate. t_{ij} is a real number between 0 (meaning the link is not used at all) and 1 (meaning the link is used 100% of the time). The objective is to maximize the flow from the source (AP) to the sink nodes, while still satisfying other constraints.

The constraints we use for our linear program are, in a scenario with k nodes,

(1) nonnegativity: all $t_{ij} \geq 0$. No link can be used for less than 0% of the time;

(2) capacity: all $t_{ij} \leq 1$. No link can be used for more than 100% of the time;

(3) conservation: $t_{0i} * R(0, i) = \sum_{x=1}^{k} t_{ix} * R(i, x)$ at any relay node. Since relay nodes are neither sources nor sinks, the amount of data in equals the amount of data out;

(4) radio usage: $\sum_{i=1}^{k} t_{ik} \leq 1$ for all nodes. No link can have its transceiver active for more than 100% of the time;

(5) spectrum: sum of all t_{ij} cannot exceed the number of channels. We cannot have more simultaneous transmissions than the total number of channels we are using;

(6) triangle constraint: sum of edges between any three nodes cannot exceed 1.

Constraint (6) is illustrated in Figure 7, and is something we call the *Triangle Constraint*. This is a scheduling problem that only appears when we have more than one channel. The times of the links between the nodes are valid values between 0 and 1. The total time spent on the air is 1.5 (sum of all link times in Figure 7), meaning that on average, 1.5 nodes are transmitting at any given time. This satisfies the spectrum constraint because 1.5 is less than 2 (the number of channels the BSS is using). However, this is still not schedulable. To see why, look at the bottom half of Figure 7. As we can see, the AP communicates with the relay for 50% of the time, with client 1 for 20% of the time, and client 2 for 30% of the time. The relay node additionally needs to spend 20% of its time forwarding data to client 1 and 30% of its time forwarding data to client 2. Although this keeps the total spectrum usage under 2, and no node's transceiver is on the air for more than 100% of the time, the time that the relay communicates with client 2 cannot overlap with the times that either the relay or client 2 spend communicating with other nodes. As we can see, the circled regions represent the time that the relay and client 2 have to communicate with each other. From the relay's perspective, the transmission must occur within this time interval because the relay is busy at all other times. However, the right side of the region overlaps with a time period where client 2 is busy. This leaves only 20% of the time for the relay to send packets to client 2, but since we need the two to communicate for 30% of the time, this cannot be scheduled!

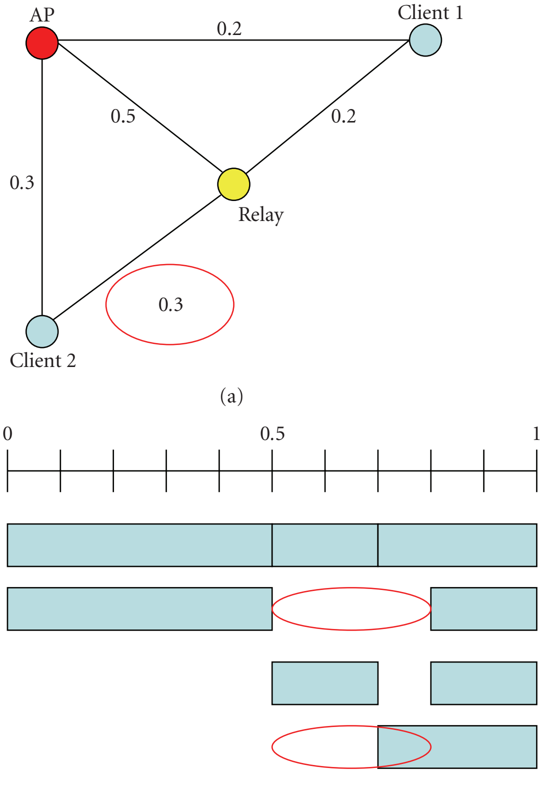

Figure 7: Illustration of triangle constraint.

We can see what the problem is by looking at the picture in a different way. Look at the triangle formed by the AP, relay, and client 2. If we consider these three nodes in isolation, it becomes clear that at any given moment in time, only two can be active (one transmitting, one receiving, and a third node that must sit idle because the other two nodes are busy). Since only one transmission (edge) can be active at any given time, in no event can the sum of the edges in any triangle add up to more than 1.0. Because this constraint applies to any three nodes that form a triangle, we call it the Triangle Constraint.

In order to find the maximum throughput gain possible by using channel borrowing, we ran simulations with Matlab's linear program solver: one set with a single channel (no borrowing) to simulate standard 802.11, and a second set with two channels and relay nodes to simulate BCR. Nodes were placed with a uniform distribution within the coverage zone of the AP and the data rate versus distance thresholds were taken directly from our experiments with CalRadio. We varied the total number of randomly placed clients in the BSS between 1 and 19. The random placement of nodes was done 250 times for each integer number of randomly placed clients. The results of all 250 runs were averaged together and, for each number of clients, these values were compared to get our estimate of the total amount of gain possible for a given number of client nodes. The top line of Figure 8

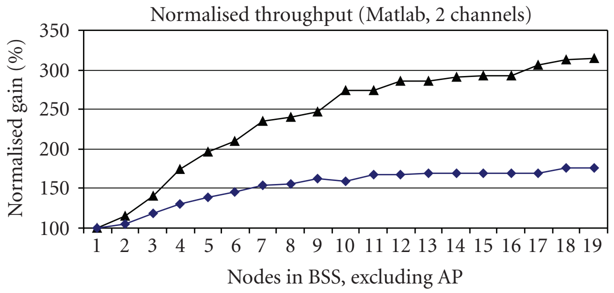

Figure 8: Matlab linear program gain results. Top: neglecting overhead, Bottom: including overhead.

Table 2: Packet lengths. All packets include 192 bit PLCP preamble and header, 32 bit FCS, and variable MAC header sent at 1 Mbps. Payload bit rates are variable.

Packet type	MAC header (bits)	Payload (bits)
DATA	192	8000
ACK	80	0
RTSBC	144	0
CTSBC	96	0
RDATA	256	8000
RACK	128	0

shows that the gain ranges from a factor of 1 (no gain) with a single client in the BSS to around 3.2 times with 19 clients, if overhead is neglected. This is the gain possible if control and management packets are not needed or take zero time, and the direct transmissions also incur no overhead (no RTS/CTS transmissions, no acknowledgement frames, etc.). While not achievable in a real network due to the extra overheads imposed by any protocol, the improvements of over 300% suggest that the channel borrowing concept has promise even after overheads reduce the amount of throughput gain that can be realized.

To get a more realistic picture of what might happen when overheads are taken into account, we ran the Matlab simulation again, accounting for overhead in both the nonrelayed (single hop) scenario and the borrowed channel, relayed packet scenario. Each packet transmitted includes 192 bits of PLCP preamble and header, 32 bits of Frame Check Sequence (error detection), and a variable length MAC header sent at 1 Mbps. The payload is sent at the data rate specified in the PLCP header, either 1 Mbps, 2 Mbps, 5.5 Mbps, or 11 Mbps. The different types of packets used, as well as the length of their MAC headers, are shown in Table 2.

To simulate the overhead associated with using our protocol, the Matlab simulation was rerun with the data rate on the links reduced to an effective rate that takes into account the length of time it takes to transmit all of our control frames as well as the time consumed by interframe spacings and channel switching times. The results are shown in the bottom line of Figure 8. As expected, the gains were lower than those when overhead is neglected, but still amount to about 50%–75% for 7 or more nodes in the BSS.

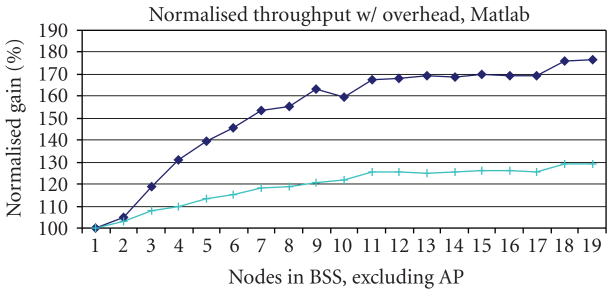

Figure 9: Matlab linear program gain results, accounting for overhead. Top: 2 channels available. Bottom: single channel available.

Table 3: NS2 Throughput Improvements for scenario shown in Figure 11.

Node	Throughput improvement
1	58.6%
2	58.1%
3	63.2%
Total	60.0%

Next, we wanted to compare the results of our BCR protocol to the rDCF and rPCF protocols which use relaying but do not allow a BSS to borrow additional channels. To do this, we ran another set of simulations in Matlab with the number of channels restricted to 1, to see how much the throughput is affected by adding the secondary channel. The results were far less impressive as shown in the bottom line of Figure 9. Although this was not a full network level simulation of rDCF and rPCF, we believe that it is safe to say that the actual performance of rDCF and rPCF would be well below that of BCR because of the large gap in theoretical throughput. Therefore, the concept of channel borrowing and relaying is worth pursuing as a method of improving 802.11 network performance.

5.3. *NS-2.*

In order to further investigate our protocol, we implemented a full network level simulation of BCR in the *NS-2* network simulator.

In each run of the simulation, we randomly placed a specified number of client nodes in the range of the AP with a uniform distribution over area, then ran both the traditional 802.11 protocol and our channel borrowing and relaying protocol, measuring the difference in total network throughput between the two. It is assumed that the AP always has access to a secondary channel from one of its neighbors whenever one is requested.

The AP transmits packets to all client nodes in the BSS as fast as it can. The results of these simulations are shown in Figure 10. The number of client nodes in the BSS was varied between 1 and 19, and each data point represents the averaged results of 250 runs with the specified number of randomly placed client nodes.

As we can see, the improvement quickly approaches 20% for around 7-8 randomly placed client nodes, a likely scenario for a crowded AP with many nodes competing for

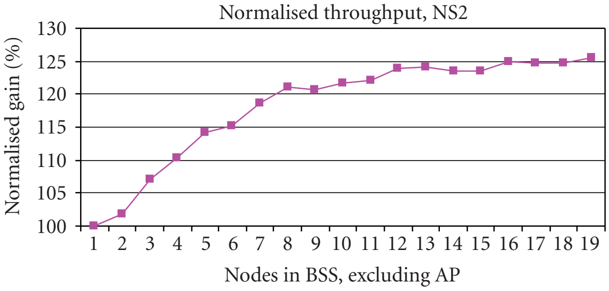

Figure 10: NS-2 normalized gain.

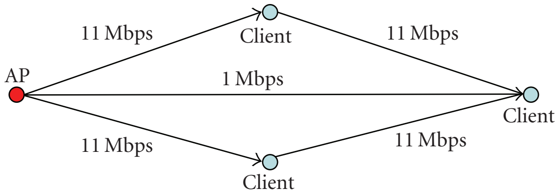

Figure 11: Ideal scenario for NS2 simulation.

time at the AP. If client nodes are numerous enough, the average gain exceeds 30%. It should be noted that these values are for random node placement. If nodes are placed optimally for relaying (e.g., maximum data rate links to both the AP and destination nodes), the actual throughput gain for those scenarios was observed to exceed 50%–60% or more. The ideal scenario for BCR is shown graphically in Figure 11. Here we have a client with the slowest possible link to the AP, and two potential relays with maximum rates on both the first and second hops. More than one client with a maximum rate connection to the AP is needed so the AP can send packets to one client while another acts as a relay and sends packets to the destination. The results from simulating this scenario are given in Table 3. However, looking back at Table 1, we can see that given the distances , the probability of finding a relay in the most ideal location is low in all except the most dense networks.

We ran a second set of NS2 simulations to see how dropped packets affect our normalized gain. For this set of experiments, we used the NS2 shadowing model with a standard deviation of 4 dB. This variation in received power causes some packets to fall below the minimum receive threshold for the data rate being used, and those packets are dropped. In our CalRadio experiments, the distance thresholds we recorded had a sufficient signal strength to come very close to a 100% transmission success rate. The receive power thresholds for each data rate were chosen so that at the maximum distance each data rate is used, the mean received power is 6 dB greater than the threshold. This causes dropped packets if the received power is weakened by shadowing by over 1.5 standard deviations from the mean. The results for this second set of experiments are shown in Figure 12. The shadowing causes some variation in the gain because packets are dropped pseudorandomly, but it is obvious that the trend is toward a gain of around 15%–25%, just as it is without the shadowing.

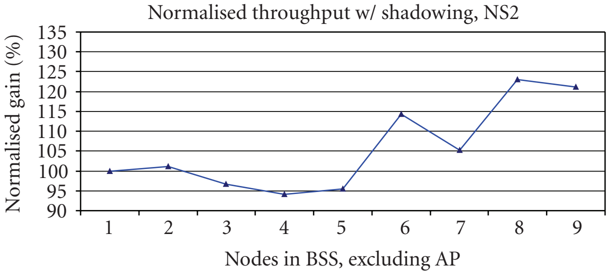

Figure 12: NS-2 normalized gain with shadowing.

We found that large numbers of packets being dropped, especially during the relay process, has an adverse effect on the gain because of the large overhead associated with sending relay packets (if a packet is dropped, we use up a lot of time on both the primary and secondary channels only to have the relay process fail). In addition, the relatively large number of packets required to accomplish a relayed transmission (7, compared to just 2 for a direct transmission) makes it more likely that something will go wrong during the process. However, as the results show, we can design our protocol to choose sufficiently conservative power thresholds for each data rate so that the probability of packets being dropped is low enough that it does not adversely affect our results.

Finally, we compared the differences between the Matlab and the NS2 results. One possible explanation for the differences is that even though we did account for overhead in Matlab, we were unable to model some types of overhead, such as the random amount of time a node backs off after a successful transmission, because linear programs require fixed amounts of flow on each link. Instead, we assumed that the channels only sit idle for the minimum amount of interframe space time between transmissions when we calculated the effective data rate on each of the links. This is not the case in *NS-2*. After the AP finishes transmitting a packet, 802.11's binary exponential backoff (BEB) mechanism requires that the transceiver sit idle in order to give other nodes a chance to access the channel. Upon receiving either an RTSBC or an RACK frame, the AP counts this as a successful transmission, and must give other nodes a chance to transmit before it sends its next packet. Since we underestimated the overhead, the improvement in throughput seen in reality will be lower than that calculated by Matlab, which does match with our results.

6. Conclusion

802.11 networks are frequency planned systems, which result in inefficiencies from static partitioning of spectrum. Existing work had addressed how to improve throughput using multiple channels and relaying with rate diversity, but we realized that these protocols may not work well if there is random node placement. Also, many existing protocols designed for multichannel MACs assume that each node can have more than one transceiver, making them incompatible with existing hardware. Building on an idea from cellular networks, we introduced a new protocol that leverages channel borrowing and relaying in combination with rate diversity to further improve throughput without custom hardware.

We used our customizable 802.11b research platform, CalRadio, to obtain data on the data rate versus distance tradeoffs in 802.11b, and used these results as inputs to our simulations. We simulated our protocol in both Matlab and *NS-2*. In *NS-2*, we saw average gains of 20–30% for random placement, and 60% for ideal placement. We believe we can do even better. Our protocol is not limited to the 802.11 family, but will work with any protocol that provides a tradeoff between signal strength and data rate. The new 802.11n protocol, for example, provides even greater rate diversity with varying signal strength, so the potential gains are even greater. Unfortunately, while our protocol cannot be used with channel bonding in the 2.4 GHz band because there is not enough spectrum to borrow even one extra 40 MHz channel pair, it is an excellent candidate to use with 802.11n in the 5 GHz band, which provides 12 nonoverlapping channels instead of just 3. Another possible improvement comes from the fact that 802.11 transceivers are generally not designed to change channels frequently because in normal operation, switching channels is an infrequent event. If this time can be cut down significantly by designing the hardware to have the ability to rapidly change channels, even more throughput gains can be made. Our transceiver in CalRadio, for example, takes 200 microseconds to tune between channels [24], and since it contains a typical 802.11 physical layer implementation, we used this channel switching overhead time as an input to our simulations. If we used radios designed to switch channels much faster, our protocol would yield even larger throughput gains. Such a task is possible, as some ultra wideband (UWB) radios can switch channels in a few nanoseconds [25].

References

[1] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 836–843, San Francisco, Calif, USA, March-April 2003.

[2] H. Zhu and G. Cao, "RDCF: a relay-enabled medium access control protocol for wireless ad hoc networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 1, pp. 12–22, Miami, Fla, USA, March 2005.

[3] H. Zhu and G. Cao, "On improving the performance of IEEE 802.11 with relay-enabled PCF," *Mobile Networks and Applications*, vol. 9, no. 4, pp. 423–434, 2004.

[4] A. Jow, C. Schurgers, and D. Palmer, "CalRadio: a portable, flexible 802.11 wireless research platform," in *Proceedings of the 1st International Workshop on System Evaluation for Mobile Platforms (MobiEval '07)*, pp. 49–54, San Juan, Puerto Rico, June 2007.

[5] H.-Y. Hsieh and R. Sivakumar, "On using peer-to-peer communication in cellular wireless data networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 1, pp. 57–72, 2004.

[6] H. Luo, R. Ramjee, P. Sinha, L. Li, and S. Lu, "UCAN: a unified cellular and ad-hoc network architecture," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MOBICOM '03)*, pp. 353–367, San Diego, Calif, USA, September 2003.

[7] A. Balachandran, G. M. Voelker, P. Bahl, and P. V. Rangan, "Characterizing user behavior and network performance in a public wireless LAN," in *Proceedings of the International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '02)*, pp. 195–205, Marina del Rey, Calif, USA, June 2002.

[8] R. L. Pickholtz, L. B. Milstein, and D. L. Schilling, "Spread spectrum for mobile communications," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 2, pp. 313–322, 1991.

[9] L. Favalli, "Sectorized channel borrowing scheme with resource pooling," in *Proceedings of the 50th IEEE Vehicular Technology Conference (VTC '99)*, vol. 5, pp. 3019–3023, Amsterdam, The Netherlands, September 1999.

[10] V. Wille, H. Multimaki, and S. Irons, "A practical approach to "channel borrowing" for microcells in GSM systems," in *Proceedings of the 48th IEEE Vehicular Technology Conference (VTC '98)*, vol. 1, pp. 144–148, Ottawa, Canada, May 1998.

[11] L. Le and E. Hossain, "Multihop cellular networks: potential gains, research challenges, and a resource allocation framework," *IEEE Communications Magazine*, vol. 45, no. 9, pp. 66–73, 2007.

[12] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, "Opportunistic media access for multirate ad hoc networks," in *Proceedings of the 8ht Annual International Conference on Mobile Computing and Networking (MOBICOM '02)*, pp. 24–35, Atlanta, Ga, USA, September 2002.

[13] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: impact of number of channels and interfaces," Tech. Rep., University of Illinois at Urbana-Champaign, 2005.

[14] W. Hung, K. Law, and A. Leon-Garcia, "A dynamic multichannel MAC for ad hoc LAN," in *Proceedings of the 21st Biennial Symposium on Communications*, April 2002.

[15] N. Jain, S. R. Das, and A. Nasipuri, "A multichannel CSMA MAC protocol with receiver-based channelselection for multihop wireless networks," in *Proceedings of the 10th International Conference on Computer Communications and Networks*, pp. 432–439, Scottsdale, Ariz, USA, 2001.

[16] J. So and N. Vaidya, "Multi-channel MAC for ad hoc networks: handling multi-channel hidden terminals using a single transceiver," in *Proceedings of the 5th International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '04)*, pp. 222–233, Tokyo, Japan, May 2004.

[17] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-L. Sheu, "A new multichannel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks," in *Proceedings of the International Symposium on Parallel Architectures, Algorithms, and Networks*, pp. 232–237, Dallas, Tex, USA, 2000.

[18] A. Balachandran, P. Bahl, and G. M. Voelker, "Hot-spot congestion relief in public-area wireless networks," in *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications*, pp. 70–80, 2002.

[19] J. So and N. H. Vaidya, "Routing and channel assignment in multichannel multi-hop wireless networks with singlenic devices," Tech. Rep., Computer Science Department, University of Illinois at Urbana-Champaign, 2004.

[20] W.-H. Tam and Y.-C. Tseng, "Joint multi-channel link layer and multi-path routing design for wireless mesh networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 2081–2089, Anchorage, Alaska, USA, May 2007.

[21] A. Raniwala, K. Gopalan, and T.-C. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, 2004.

[22] T.-Y. Lin, W.-H. Tam, K.-L. Fan, and Y.-C. Tseng, "Resource planning and packet forwarding in multi-radio, multi-mode, multi-channel, multi-rate (M4) wireless mesh networks," *Computer Communications*, vol. 31, no. 7, pp. 1329–1342, 2008.

[23] X. Wang and J. J. Garcia-Luna-Aceves, "Distributed joint channel assignment, routing and scheduling for wireless mesh networks," *Computer Communications*, vol. 31, no. 7, pp. 1436–1446, 2008.

[24] "Maxim 2.4ghz 802.11b zero-if transceivers".

[25] L. I. Williams, D. Wu, E. Staggs, and A. Yen, "Ultra-wideband radio design for multi-band OFDM 480 Mb/s wireless USB," in *DesignCon Symposium Digest*, Santa Clara, Calif, USA, January 2005.