Comparison of Multichannel MAC Protocols

Jeonghoon Mo, *Member*, *IEEE*, Hoi-Sheung Wilson So, *Member*, *IEEE*, and Jean Walrand, *Fellow*, *IEEE*

Abstract—This paper compares, through analysis and simulation, a number of multichannel MAC protocols. We first classify these protocols into four categories based on their principles of operation: *Dedicated Control Channel*, *Common Hopping*, *Split Phase*, and *Parallel Rendezvous protocols*. We then examine the effects of the number of channels and devices, channel switching times, and traffic patterns on the throughput and delay of the protocols. Here are some of the conclusions of our study: 1) Parallel Rendezvous protocols generally perform better than Single Rendezvous protocols, 2) the Dedicated Control Channel protocol can be a good approach with its simplicity when the number of channels is high and the packets are long, and 3) the Split Phase protocol is very sensitive to the durations of the control and data phases. Our study focuses on a single collision domain.

Index Terms—Multichannel MAC, 802.11, performance.

1 Introduction

Researchers have proposed protocols that exploit the multiple channels available in 802.11 and other wireless networks to increase the capacity. These protocols are for networks in which orthogonal channels such as disjoint frequency bands are available. Using a multichannel media access control (MAC) protocol, different devices can transmit in parallel on distinct channels. The parallelism increases the throughput and can potentially reduce the delay, provided that the channel access time is not excessive. Protocols differ in how devices agree on the channel to be used for transmission and how they resolve potential contention for a channel. These choices affect the delay and throughput characteristics of the protocol.

There have been only limited studies comparing these protocols under identical operating conditions. In this paper, we compare several existing and one new multichannel MAC (MMAC) protocols. As may be expected, different protocols are preferable, depending on the operating conditions. Our objective is to contribute to the understanding of the relative merits of different designs. We first use analytical models to gain insight into the strengths and weaknesses of the various protocols. We then use simulations to obtain more accurate comparisons, assuming more realistic traffic patterns. We only consider the case of a single collision domain where all the devices can hear one another.

In Section 2, we describe the protocols that we compare in this paper. Section 3 presents simplified analytical models of the protocols. Section 4 discusses the numerical results from the analytical models and compares the performance of the protocols. Section 5 describes the simulation results of more realistic models of the protocols. Section 6 concludes the paper with some comments about the lessons of this study.

Limitations of our work. Before we move to the next section, we would like to acknowledge the limitations of our study:

- The physical channel model used in the simulation is a fixed capacity channel model with no errors. All devices have an identical capacity, which may not be true in reality since the devices can have different channel gains and interference levels. The channel quality between pairs of clients can also vary in practice.
- Our work is limited to a single collision domain and does not consider multiple domain issues such as exposed or hidden nodes. Extending the results to multiple collision domains is left as future work.

Contribution of the work. Even with these limitations, we believe that our work can be helpful by providing some insight on the MMAC design. Our major observations can be summarized as follows:

- Parallel Rendezvous protocols such as McMAC and Slotted Seeded Channel Hopping (SSCH) can perform better than Single Rendezvous protocols with one radio under a wide range of situations by eliminating the control channel bottleneck.
- The Dedicated Control Channel protocol, at the cost of two radios, outperforms other protocols when there are many channels and when packets are long. Other protocols that use only one radio fail to perfectly monitor the channels and the status of the other nodes, thereby reducing the achievable throughput. However, when the number of channels is small, the performance cost of one dedicated channel is high.
- The performance of Split Phase is very sensitive to parameters such as the duration of the control and data phases. To optimize the performance, these

- *J. Mo is with the School of Engineering, Information and Communications University, Yusong PO Box 77, Taejon, 305-600, South Korea. E-mail: jhmo@icu.ac.kr.*
- *H.-S.W. So and J. Walrand are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA 94720. E-mail: {so, wlr}@eecs.berkeley.edu.*

Manuscript received 21 Apr. 2006; revised 5 Nov. 2006; accepted 25 Jan. 2007; published online 7 May 2007.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0113-0406.
Digital Object Identifier no. 10.1109/TMC.2007.1075.

TABLE 1
Four Representative Protocols

Protocol	No. Rendezvous	No. Radios	Basic Scheme
Dedicated Control Channel	Single	2	Devices use one radio to monitor constantly the control channel.
Common Hopping	Single	1	Devices hop together quickly and stop upon agreement for transmission.
Split Phase	Single	1	Devices periodically tune to a common control channel.
McMAC	Multiple	1	Transmitter jumps to receiver's slow hopping sequence.

parameters need to be adjusted. Separating the control packets from data packets does not improve performance because, when the data channels are congested, generating more successful rendezvous is useless.

- The switching penalty is an important parameter that greatly impacts the performance of protocols such as McMAC and Common Hopping, which hop frequently.
- Receiver contention decreases the system throughput for all MAC protocols, especially the Common Hopping and Split Phase protocols. In contrast, the impact of receiver contention on the Dedicated Control Channel protocol is negligible.
- Finally, we found that, when the MAC protocols treat various packet types (for example, real-time video and file transfers) equally, allowing a sender to transmit multiple packets to the same destination after each rendezvous is highly beneficial. This possibility improves the throughput, delay, and jitter for all types of packets by using any of the four protocols.

2 Description of Multiple Channel Protocols

There are many variations on multichannel protocols. Our first step is to classify them based on their general principles of operation. We then describe the representative protocols of the different classes. We also comment on the many variations that have been proposed for such protocols.

2.1 Principles of Operation

Devices using a MMAC protocol exchange some control information to agree on the channel for transmitting data. In *Single Rendezvous* protocols, that exchange of control information occurs on a single channel at any time, even though the rendezvous channel may change over time. That single control channel can become the bottleneck under some operating conditions. When using a *parallel rendezvous* protocol, multiple devices can use different channels in parallel to exchange control information and make new agreements. This approach alleviates the rendezvous channel congestion problem but raises the challenge of ensuring that the idle transmitter and receiver visit the same rendezvous channel.

In the subsequent sections, we describe the following protocols and variations (see Table 1): Dedicated Control Channel, Common Hopping, Split Phase, and McMAC.

2.2 Dedicated Control Channel

Every device has two radios. One radio is tuned to a channel dedicated to control messages and the other radio can tune to any other channel. In principle, all devices can overhear all the agreements made by other devices, even during data exchange. This system's efficiency is limited only by the contention for the control channel and the number of available data channels.

Fig. 1a illustrates the operations of Dedicated Control Channel. In the figure, channel 0 is the control channel and channels 1, 2, and 3 are for data transmission. When device A wants to send to device B, it transmits a request-to-send (RTS) packet on the control channel. That RTS specifies the lowest numbered free channel. Upon receiving the RTS, B responds with a clear-to-send (CTS) packet on the control channel, confirming the data channel suggested by A. The RTS and CTS packets also contain a Network Allocation Vector (NAV) field, as in 802.11, to inform other devices of the duration for which the sender, the receiver, and the chosen data channel are busy. Since all devices listen to the control channel at all times, they can keep track of the busy status of other devices and channels, even during data exchange. Devices avoid busy channels when selecting a data channel.

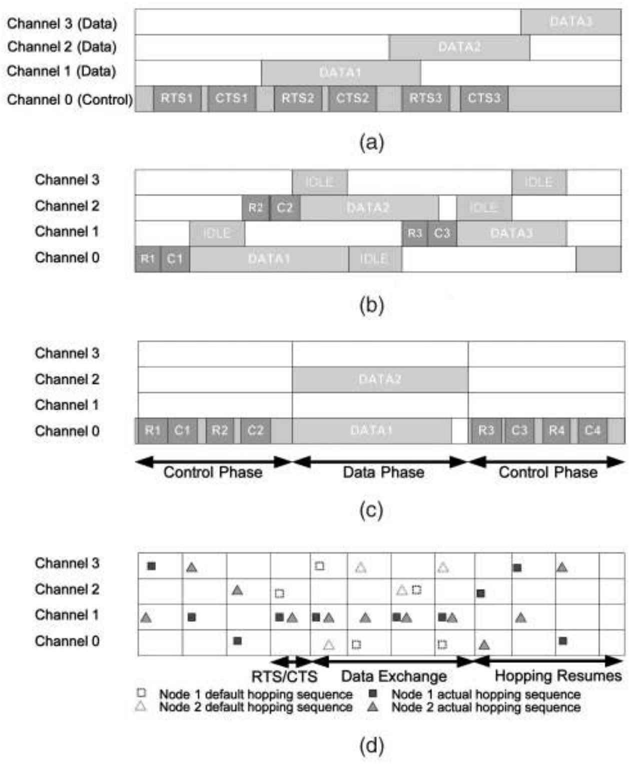


Fig. 1. Single Rendezvous MAC protocols: (a) Dedicated Control Channel Approach. (b) Common Hopping Approach. (c) Split Phase Approach. Multiple Rendezvous MAC protocol: (d) McMAC.

Examples of this approach include Dynamic Channel Allocation (DCA) [10], DCA with Power Control (DCA-PC) [11], and Dynamic Private Channel (DPC) [12].

The major advantage of Dedicated Control Channel is that it does not require time synchronization: Rendezvous always happen on the same channel. The main disadvantage of this protocol is that it requires a separate dedicated control radio and a dedicated channel, thereby increasing cost and decreasing spectral efficiency when few channels are available.

2.3 Common Hopping

In this approach, devices have only one radio. Devices not exchanging data hops through all channels synchronously. A pair of devices stop hopping as soon as they make an agreement for transmission and rejoin the common hopping pattern subsequently after transmission ends.

The Common Hopping protocol improves on Dedicated Control Channel in two respects: 1) it uses all the channels for data exchange and 2) it requires only one transceiver per device. As shown in Fig. 1b, the hopping pattern cycles through channels 0, 1, 2, and 3. When device A wants to send to device B, it sends an RTS to B on the current common channel. If B receives the RTS properly, then it returns a CTS on the same channel. Devices A and B then pause hopping and remain on the same channel during data transfer, whereas the other idle devices continue hopping. When they are finished, devices A and B rejoin the Common Hopping sequence with all the other idle devices. It is possible that the Common Hopping sequence wraps around and visits the channel that A and B are using before they finish data exchange. Idle devices sense the carrier and refrain from transmitting if it is busy.

While A and B are exchanging data, they are unaware of the busy status of the other devices. Hence, it is possible that a sender sends an RTS to a device that is currently busy on a different channel. Another issue with this approach is that devices hop more frequently. State-of-the-art integrated circuit implementations of trimode 802.11a/b/g radios require only about 30 μs for its voltage-controlled oscillator (VCO) to settle [13], but commercial off-the-shelf 802.11b transceivers require about 150 to 200 μs to switch channels [4]. Considering that an RTS in 802.11b takes only about 200-300 μs, the hopping time penalty is not negligible. The approach also requires devices to have tight synchronization. Examples of this design approach include channel hoping multiple access (CHMA) [8] and Channel Hopping multiple Access with packet Trains (CHAT) [9].

2.4 Split Phase

In this approach, devices use a single radio. Time is divided into an alternating sequence of control and data exchange phases, as shown in Fig. 1c. During a control phase, all devices tune to the control channel and attempt to make agreements for channels to be used during the following data exchange phase.

If device A has some data to send to device B, then it sends a packet to B on the control channel with the ID of the lowest numbered idle channel, say, i. Device B then returns a confirmation packet to A. At this point, A and B have agreed to use channel i in the upcoming data phase. Once committed, a device cannot accept other agreements that conflict with earlier agreements (note that, when hidden nodes are prevalent, the sender and the receiver might have very different views of which channels are free. A more sophisticated agreement protocol is then needed, as proposed in [6]).

In the second phase, devices tune to the agreed channel and start data transfer. The protocol allows multiple pairs of devices to choose the same channel because each pair might not have enough data to use up the entire data phase. As a result, the different pairs must either schedule themselves or contend during the data phase. In the analysis, we assume that at most one device pair can be assigned to each channel, so there is no need for scheduling or contention. In the simulation section, we assume random access, as suggested in MMAC [6], where multiple pairs can share the same channel during a data phase.

The advantage of this approach is that it requires only one radio per device. However, it requires time synchronization among all devices, though the synchronization can be looser than in Common Hopping because devices hop less frequently. Examples of this approach are MMAC and Multichannel Access Protocol (MAP) [2]. Their main difference is that the duration of the data phase is fixed in MMAC, whereas it is variable in MAP and depends on the agreements made during the control phase.

2.5 Parallel Rendezvous

Parallel Rendezvous protocols differ from the previous three in that multiple device pairs can make agreements simultaneously on distinct channels. The main motivation is to overcome the single control channel bottleneck. However, since there are multiple rendezvous channels, special coordination is required so that two devices can rendezvous on the same channel. One solution is for each idle device to follow a "home" hopping sequence and for the sending device to transmit on that channel to find the intended receiver. Examples of this approach include SSCH [1] and McMAC [5].

In SSCH, there are as many hopping sequences that each device can follow as there are channels. Each sequence is uniquely determined by the seed of a pseudorandom generator. Each device picks multiple (for example, four) sequences and follows them in a time-multiplexed manner. When device A wants to talk to B, A waits until it is on the same channel as B. If A frequently wants to talk to B, then A adopts one or more of B's sequences, thereby increasing the time they spend on the same channel. For this mechanism to work, the sender learns the receiver's current sequences via a seed broadcast mechanism.

In McMAC, each device picks a seed to generate a different pseudorandom hopping sequence. When a device is idle, it follows its "home" hopping sequence. Each device puts its seed in every packet that it sends, so its neighbors eventually learn its hopping sequence. For simplicity of analysis and simulation, devices are assumed to hop synchronously in this study. However, nodes are not required to align their hopping boundaries in practice. The hopping can be made less frequent than in the Common Hopping protocol to reduce the channel switching penalty and synchronization overhead. When device A has data to send to B, A flips a coin and transmits with some probability p during each time slot. If it decides to transmit, then it tunes to the current channel of B and sends an RTS. If B replies with a CTS, then both A and B stop hopping to exchange data. Data exchange normally takes place over several time slots. After the data exchange is over, A and B return to their original hopping sequence, as if no pause in hopping had happened.

SSCH and McMAC are similar in that they allow devices to rendezvous simultaneously on different channels. However, there are subtle differences. In SSCH, each node chooses four different hopping sequences and time multiplexes them to form a single hopping sequence. Nodes adapt their hopping sequences over time to the traffic but are not allowed to deviate from their hopping sequences. In McMAC, each node has one hopping sequence, which never changes. However, nodes are allowed to deviate from their default hopping sequence temporarily to accommodate sending and receiving.

In SSCH, a sender must wait until its current channel overlaps with that of the receiver before it can send data. In McMAC, the sender can temporarily deviate from its sequence to jump to the receiver's channel to send.

Since SSCH relies on several deterministic heuristics to adapt the four hopping sequences to that of receivers, it makes analysis more difficult. In contrast, McMAC relies on randomization to load balance sender/receiver pairs over different channels, which is easier to analyze. Therefore, in the rest of the paper, we focus on McMAC as an example protocol in this category. Despite these differences, the common Parallel Rendezvous feature of both protocols accounts for the major difference in performance from other Single Rendezvous protocols.

2.6 Variations on Generalized Approaches

In this paper, we compare four generalized approaches to designing multichannel MAC protocols. An actual protocol includes fine adjustments that deviate from the generalized scheme. Instead of incorporating all possible variations of the four schemes in our analysis and simulation, we briefly mention several proposed improvements and discuss their effects qualitatively.

For Dedicated Control Channel, it is possible to use the control channel for data transfer when all other channels are busy. For Split Phase, adaptation of the duration of data and control phases was proposed by Chen et al. [2]. So and Vaidya [6] suggest advertising the number of packets for each destination in the rendezvous message to achieve better load balancing across channels.

For the McMAC scheme presented, we assume that each device hops once approximately every RTS/CTS time slot. This hopping randomizes the channel for which any two devices meet and, hence, load balances traffic across different channels. However, frequent hopping imposes a more stringent requirement for the synchronization between devices. In practice, devices can hop at a slower pace and still retain most benefits of load balancing.

3 System Model and Analysis

In this section, we describe and analyze simplified models of the protocols. The objective is to numerically compare the performance of the four protocol families under identical operating conditions. The following simplifications are made for all the protocols:

1. Time is divided into small slots, with perfect synchronization at slot boundaries. Each time slot is just long enough to exchange RTS/CTS frames to make an agreement.
2. Upon making an agreement, the devices can transmit only one packet (one may think of a "packet" as the amount of data that can be transferred per channel agreement).
3. The packet lengths, which are integer multiples of slot durations, are independent and geometrically distributed with parameter q (that is, packet duration has a mean of $1/q$ slots).
4. Devices always have packets to send to all the other devices. In each time slot, an idle device attempts to transmit with probability p. The receiver of a sender is decided randomly with equal probability among possible candidates.
5. We assumed the slotted aloha model to mimic the exchange of RTS/CTS. When a device has a packet to transmit, it attempts to transmit a packet with probability p by sending an RTS. The agreement is made when only one device attempts to transmit an RTS.

The first simplification enables us to use discrete time models. The second one reduces the efficiency of all the protocols since it requires a new agreement for every packet. The third simplification is needed to construct simple Markov models. We use the fourth simplification to study the maximum throughput of the protocols.

These simplifications allow us to form a Markov chain whose state X_t is the number of communicating pairs at time t. When $X_t = k$, $2k$ devices are involved in data transfer, while the other $N - 2k$ devices are idle. The maximum number of pairs is bounded by the number of data channels M_D and half the number of devices $\lfloor N/2 \rfloor$. Accordingly, the state space of X_t is

$$\mathcal{S} := \left\{0, 1, \ldots, \min\left(\left\lfloor \frac{N}{2} \right\rfloor, M_D\right)\right\}.$$

The number M_D of data channels is equal to M for all approaches except Dedicated Control Channel, for which $M_D = M - 1$ since a channel is reserved for control.

A state transition happens when new agreements are made or existing transfers end. Let $S_k^{(i)}$ and $T_k^{(j)}$ respectively denote the probability that i new agreements are made and the probability that j transfers terminate in the next slot when the state is k. The state-transition probability p_{kl} from state k at time t to l at time $t+1$ can be expressed as follows:

$$p_{kl} = \sum_{m=(k-l)^+}^{k} S_k^{(m+l-k)} T_k^{(m)}. \quad (1)$$

In this expression, m is the number of transfers that terminate and its value is between $(k-l)^+$ and k. At least $(k-l)^+$ transfers must terminate to have l pairs in the next slot and k is the maximum number of terminating transfers.

TABLE 2
Summary of Notations

symbol	meaning of the symbol
N	number of devices
M	number of channels
M_D	number of data channels
$S_k^{(i)}$	probability that i new agreements are made in the next slot
$T_k^{(j)}$	probability that j transfers terminate in the next slot
p_{kl}	transition probability from state k to state l
C	channel capacity of each channel in bits/sec
k	subscript used to denote the number of active pairs
p	probability of any node sending an RTS during a time slot
t_s	time required for an RTS/CTS exchange
t_p	time penalty for a channel switch

Also, the probability $T_k^{(j)}$ that i transfers finish when the system is in state k is given by the following:

$$\begin{aligned} T_k^{(j)} &= Pr[j \text{ transfers terminate at time } t | X_{t-1} = k] \\ &= \binom{k}{j} q^j (1-q)^{k-j}. \end{aligned} \tag{2}$$

Equation (1) is further simplified for Single Rendezvous protocols such as Dedicated Control Channel or Common Hopping because $S_k^{(i)} = 0$ for all $i > 1$. Indeed, at most one additional pair can meet in the next slot in a Single Rendezvous protocol. Accordingly, for such protocols, the equation becomes

$$p_{kl} = T_k^{(k-l)} S_k^{(0)} + T_k^{(k-l+1)} S_k^{(1)}, \tag{3}$$

where $T_k^{(j)} = 0$ when $j < 0$.

The average utilization ρ per channel can be obtained as

$$\rho = \frac{\sum_{i \in \mathcal{S}} i \cdot \pi_i}{M}, \tag{4}$$

where π_i is the limiting probability that the system is in state i, and $\mathcal{S}$ is the state space of the Markov chain. One obtains π_i by solving the balance equations of the Markov chain. We then derive the total system throughput by multiplying ρ by the channel transmission rate and by M_D, the number of data channels. We summarize the notations in Table 2 for reference.

3.1 Dedicated Control Channel

Devices constantly monitor the control channel and keep track of which devices and data channels are idle. When a device has packets to transmit to an idle receiver, it sends an RTS message for that idle receiver on the control channel by using the control radio. If it hears the RTS, then the receiver replies to the sender with a CTS, also using the control radio. Then, both the sender and the receiver tune to the agreed channel to start transmission by using their data radios.

An agreement is made when exactly one idle device attempts to transmit an RTS on the control channel. Hence, the success probability $S_k^{(i)}$ in the next time slot, given that k pairs are communicating in the current slot, is

$$S_k^{(i)} = \begin{cases} (N-2k)p(1-p)^{(N-2k-1)} & \text{if } i = 1 \\ 1 - S_k^{(1)} & \text{if } i = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Here, $p(1-p)^{(N-2k-1)}$ is the probability that one specific device transmits an RTS with probability p in a given time slot while all other $N-2k-1$ devices do not. Since all $N-2k$ devices can try to transmit an RTS, we have the expression for $S_k^{(1)}$.

The transition probabilities (3) can be rewritten as follows:

$$p_{kl} = \begin{cases} 0 & \text{if } l > k+1 \\ T_k^{(0)} S_k^{(1)} & \text{if } l = k+1 \\ T_k^{(k-l)} S_k^{(0)} + T_k^{(k-l+1)} S_k^{(1)} & \text{if } 0 < l \leq k \\ T_k^{(k)} S_k^{(0)} & \text{if } l = 0. \end{cases} \tag{6}$$

We obtain the system throughput R_d as

$$R_d = (M-1) C \rho_d, \tag{7}$$

where C is the channel capacity and ρ_d is the data channel utilization that we calculate using (4). The subscript d refers to the Dedicated Control Channel approach.

3.2 Common Hopping

The analysis of this protocol is very similar to that of the Dedicated Control Channel, but with three differences: 1) devices do not track the status of each other, 2) some slots are busy and unavailable for control messages, and 3) the switching penalty is incurred whenever a device hops.

3.2.1 *Information about Other Devices*

Even if an RTS is sent without collision, the receiver may not respond because it might be busy on a different channel. When a device is sending or receiving, it cannot keep track of others. At best, idle devices can keep track of the agreements that others make. However, this information becomes stale once the devices become busy. We approximate this situation by assuming that the sender selects the receiver uniformly out of $N-1$ other devices and that the probability that the selected receiver is not busy is $\frac{N-2k-1}{N-1}$.

3.2.2 *Busy Slots*

The protocol can make a new agreement only when the current common hopping channel is idle. We model this effect by considering that the probability that idle devices can use a given slot to make an agreement is $(M-k)/M$ when k channels are busy.

Combining the effects described in the previous two points, the success probability $S_k^{(i)}$ becomes

$$S_k^{(i)} = \begin{cases} (N-2k)p(1-p)^{(N-2k-1)} \\ \qquad \times \frac{N-2k-1}{N-1} \times \frac{M-k}{M} & \text{if } i = 1 \\ 1 - S_k^{(1)} & \text{if } i = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

The factor $\frac{N-2k-1}{N-1}$ in (8) is the probability that the receiver is one of the $N-2k-1$ idle devices other than the transmitter among the other $N-1$ devices. The last term $\frac{M-k}{M}$ represents the chance that the common hopping channel is busy when k channels are busy.

3.2.3 *Channel Switching Penalty*

Let t_s and t_p denote the duration of one slot and the channel switching penalty, respectively. For Dedicated Control

Channel, the time slot size is t_s. However, for Common Hopping, since each idle node hops periodically and frequently, it is natural to use a slot size of $t_s' = t_s + t_p$ in both analysis and simulation. Since each time slot is slightly longer in Common Hopping, to keep the average packet length the same in real time, the packet termination probability per slot q has to be increased to $q' = q \cdot \frac{t_s+t_p}{t_s}$.

The system throughput R_c is given as follows:

$$R_c = MC\rho_c, \tag{9}$$

where C is the channel capacity and ρ_c is the utilization of the system that can be calculated using (4). The subscript c denotes Common Hopping. Note that the multiplier is M instead of $M-1$ in Dedicated Channel.

3.3 Parallel Rendezvous

We analyze McMAC [5] in this section. In McMAC, each device has a home pseudorandom hopping sequence defined by a seed. Eventually, through broadcast, every device knows the seed of the other devices and can determine their home channel at any given time.

In McMAC, a device is unaware of which devices and channels are idle. We assume that any idle device is equally likely to have any channel as its current home channel and that these home channels are independent. When a device is idle, it attempts to transmit in the next time slot with probability p. In that case, the device chooses another device at random and goes to its channel.

To make j new agreements, the following conditions must hold:

- The number A of devices that attempt to transmit should be at least j.
- The devices attempting to transmit should be on a channel without other attempts. Let O denote the number of channels where those isolated attempts take place. We call these channels "one-attempt channels."
- The channel where a device attempts to transmit should be idle. We designate by I the number of idle channels among the O one-attempt channels.
- Designate by J the number of transmitters out of I that can find a receiver that is idle and does not attempt to transmit. Then, $J = j$.

We compute the probability $S_k^{(j)}$ by conditioning on the values of O, I, and A. We can write

$$S_k^{(j)} = \sum\nolimits_{o,i,a} P[A=a] \times P[O=o|A=a] \\ \times P[I=i|O=o,A=a]P[J=j|I=i,A=a,O=o], \tag{10}$$

where:

- $P[A=a] = \binom{N-2k}{a} p^a (1-p)^{(N-2k-a)}$.
- $P[O=o|A=a]$ is the probability that o devices out of a are in a channel without other attempts, given that a devices attempt. This probability is the same as the probability that o urns out of M contain exactly one ball each after we throw a balls independently and uniformly into M urns. If we let Y_i^n be the number of urns with i balls at the nth throw, then (Y_0^n, Y_1^n) is a Markov chain and the probability distribution of Y_1^a can be computed recursively over $n = 1, \dots, a$ from the transition probabilities of that Markov chain.
- $P[I=i|O=o,A=a]$ is the probability that i channels out of o one-attempt channels are idle, which is equivalent to the probability that exactly $o-i$ out of o one-attempt channels are busy. Recall that k out of M channels are busy. We pick k busy channels out of M uniformly and also pick o one-attempt channels out of M uniformly independently of each other. The intersection of the two sets corresponds to busy one-attempt channels. The conditional distribution of I is given by

$$P[I=i|O=o,A=a] = P[I=i|O=o] = \frac{\binom{k}{o-i}\binom{M-k}{i}}{\binom{M}{k}}.$$

The first equality comes from that I and A are conditionally independent given O. The denominator is the number of ways to select k busy channels out of M. The numerator is the number of ways to select $o-i$ one-attempt channels among k busy ones and to select i one-attempt channels among $M-k$ channels.
- We approximate the probability that a sender attempting to transmit alone in an idle channel finds its receiver by $p_s = \frac{N-2k-a}{N-1}$. Moreover, we assume that the i transmitters are independently successful in finding their receivers. Let J be the number of successful senders who are alone in an idle channel and are able to find their receiver successfully. Then,

$$P[J=j|I=i,O=o,A=a] = P[J=j|I=i,A=a] \\ = \binom{i}{j} p_s^j (1-p_s)^{(i-j)}, \tag{11}$$

where p_s is defined above.

3.4 Split Phase Approach

In Split Phase, time is divided into alternate control and data phases with durations c and d, respectively. Let $R(n)$ be the throughput of the nth period. By ergodicity,

$$\lim_{n\to\infty} \frac{\sum_{i=1}^n R(n)}{n} \to R_s,$$

where R_s is the expected throughput per period of the Split Phase approach.

We designate by K_n the random number of agreements made during the nth control phase, and we define

$$\phi_i^c := P(K_n = i), \quad \text{for } i = 0,1,2,\dots,\gamma, \tag{12}$$

where $\gamma := \min(c, \lfloor \frac{N}{2} \rfloor)$. The above probability can be computed using the following recursive relationship on the duration c:

$$\phi_i^c = p_{succ}\phi_{i-1}^{c-1} + (1-p_{succ})\phi_i^{c-1}, \tag{13}$$

where p_{succ} is the probability that an agreement is made in a slot. Note that $\phi_0^c = (1-p_{succ})^c$.

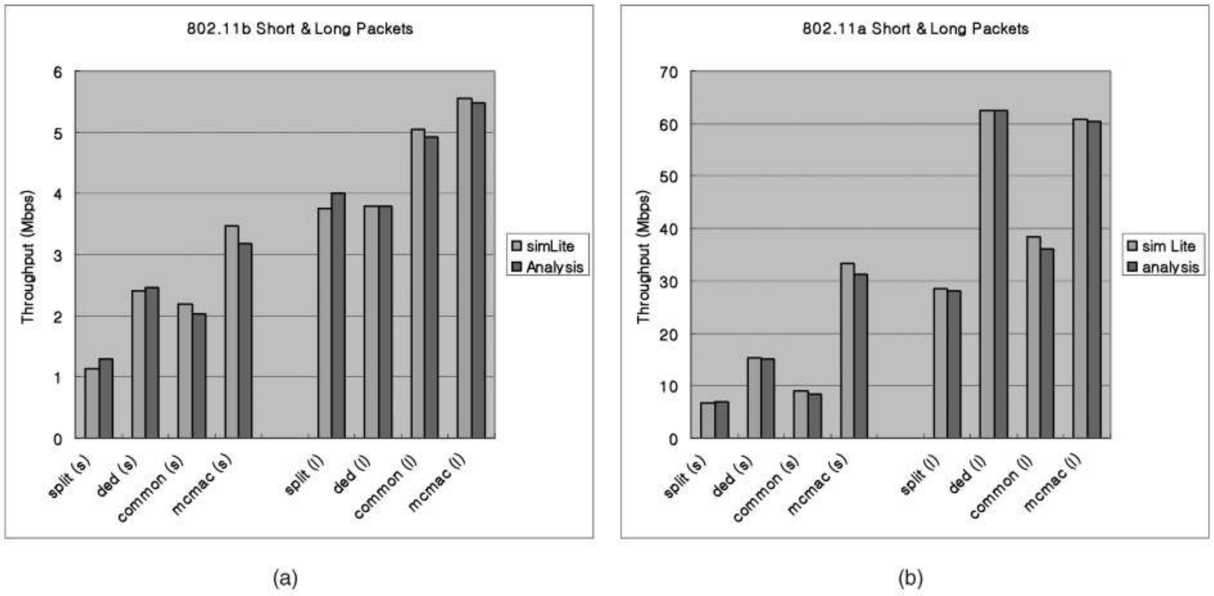


Fig. 2. Throughput predicated by the analysis versus simulation. (a) 802.11b setting. (b) 802.11a setting.

Let $R(i, d)$ denote the average throughput in the duration of d when i agreements are made in the control phase. Then, the channel utilization ρ_s can be computed using the following formula:

$$\rho_s = \frac{1}{(c+d)M} \sum_{i=1}^{\gamma} \phi_i^c R(i, d). \tag{14}$$

The system throughput R_s is

$$R_s = MC\rho_s, \tag{15}$$

where C is the channel capacity.

The throughput $R(i, d)$ when i new agreements are made in the control channel cannot be larger than Md. If we assume a perfectly even distribution of i agreements over M channels, each channel has either l or $l+1$ agreements, where $l := \lfloor \frac{i}{M} \rfloor$. Since the packet size is geometrically distributed, the channel that has l agreements can be utilized up to $\min(\sum_{j=1}^{l} Y_j, d)$ slots, where the Y_j's are geometrically distributed random variables. Therefore, the averaged throughput can be upper bounded as follows:

$$R(i, d) \leq (M-r)E\left[\min\left(\sum_{j=1}^{l} Y_j, d\right)\right] + rE\left[\min\left(\sum_{j=1}^{l+1} Y_j, d\right)\right], \tag{16}$$

where r is the number of channels with $l+1$ agreements. Thus, $(M-r)$ is the number of channels with l agreements.

The right side of (16) is an upper bound since we assume perfect alignments among l pairs or $l+1$ pairs in each channel. The real throughput can be smaller due to collision among l pairs. However, we can use the right side as an approximation when the value l is small.

If c is too short, then the multiple channels may not be used efficiently. On the other hand, if c is too long, then packets may suffer long delays and channels become underutilized. Therefore, choosing appropriate values for c and d is crucial to the performance of this scheme.

4 Numerical Results

In this section, we compare the different protocols by using the analytical models that we described so far. First, we validate the analytical model against the simulation model by using 802.11 scenarios. The details of the simulation model are described in the next section. Second, we vary the main system parameters such as the number of channels, channel switching time, number of devices, and so on to observe their effects on the system throughput.

We created two simulation scenarios: 802.11b and 802.11a. In the 802.11b scenario, there are 20 devices and three channels, and each channel has a data rate of 2 Mbps. The time required for RTS/CTS is 812 μs. In the 802.11a scenario, there are 40 devices and 12 channels with a rate of 6 Mbps each. The time required for RTS/CTS is 200 μs. The switch penalty is 100 μs in both scenarios. The packets generated have a random length with a geometric distribution, with the average length of 1 Kbyte or 10 Kbytes. One should think of the packet length as the amount of data that a device can transfer after each channel agreement since there is no queuing in our analysis. The bar charts in Fig. 2 show that the analytical models are in close agreement with the simulation results (SimLite) for different packet lengths in both scenarios. The purpose of SimLite is

1. to verify that the approximations that we made in our analysis are reasonable and
2. to get the basic understanding of parameter dependence.

Specifically, in our analysis, we only keep track of the number of busy channels and assumed that whether one channel is busy is completely independent of the other channels. SimLite keeps track of the status of different channels and nodes precisely.

4.1 802.11b Scenario

Fig. 2 shows the throughput results of four different protocols under the 802.11b scenario. The x-axis is the

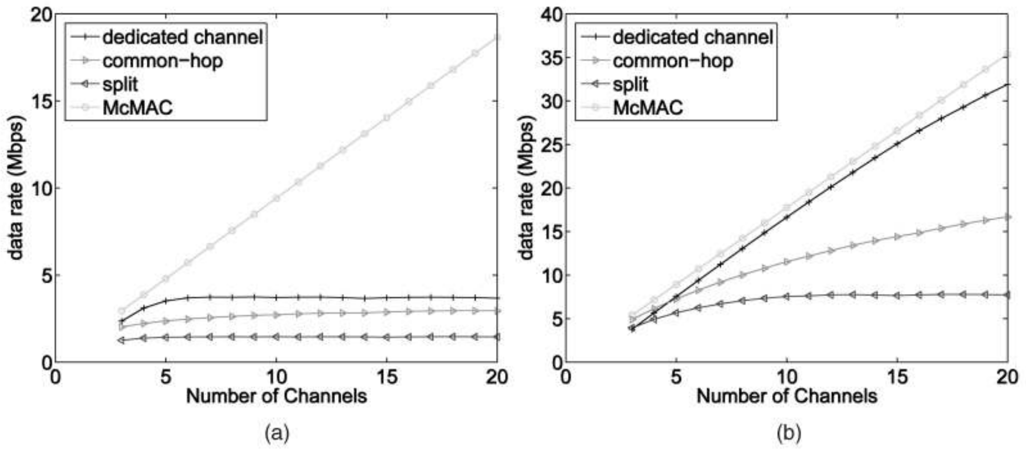


Fig. 3. Throughput versus number of channels. The average packet size is (a) 1 Kbyte and (b) 10 Kbytes.

transmission probability p of each device and the y-axis is the throughput. We make the following observations:

- *Dedicated Channel Protocol.* The protocol can only use two out of three channels for data transmission and, hence, limits the maximum throughput to 4 Mbps, which is much less than Common Hopping or McMAC. Given that constraint, however, Dedicated Control Channel achieves 3.8 Mbps.
- *Parallel Rendezvous.* Although the control channel is not saturated, McMAC still performs the best for both short and long packets because 1) it uses all three channels for data transfer and 2) it does not have any quantization overhead of Common Hopping since it can reuse a channel as soon as the previous transfer has finished. Common Hopping cannot reuse a channel until the common hopping sequence has wrapped around to it. For long packets, McMAC achieves a remarkable 5.5 Mbps out of 6 Mbps.
- *Split Phase Approach.* The throughput of Split Phase is the lowest for 1-Kbyte packets and close to the lowest for 10-Kbyte packets, even though we have optimized the control and data durations c and d to be 20 and 40 ms. This shows the inability of Split Phase to use spectrum efficiently.

4.2 802.11a Scenario

The bar chart on the right in Fig. 2 shows the throughput results of the four different protocols under the 802.11a scenario. We make the following observations:

- *Parallel Rendezvous.* McMAC achieves about 31 Mbps in the small packet size case (Fig. 2a) and 60 Mbps for the large packet size case (Fig. 2b). Note that Dedicated Control Channel performs slightly better than McMAC when packets are large. However, recall that it uses two radios, so each node has perfect knowledge of the status of other nodes and channels.
- *Control Channel Congestion.* In Single Rendezvous protocols, since there is only one control channel, a combination of short packets and a large number of channels can cause control channel congestion, as shown in Fig. 2a. The highest possible throughput for Single Rendezvous Channel protocols can be estimated as follows: On the average, one agreement is made every $e = 2.718$ slots by using the ALOHA protocol. After each agreement, a 1-Kbyte packet is transferred on the average. Since it takes 6.8 slots to transfer a 1-Kbyte packet (200 μs slots, 6 Mbps), the maximum throughput is $(6.8/e) \times 6\ \text{Mbps} \approx 15\ \text{Mbps}$. Dedicated Control Channel, Common Hopping, and Split Phase achieve 15, 9, and 8 Mbps, respectively.
- *Dedicated Channel with Long Packets.* Observe that Dedicated Control Channel achieves more than 63 Mbps when the average packet size is 10 Kbytes. With large average packet size, the control channel is no longer a bottleneck. Moreover, 1 out of 12 channels is a small overhead.

4.3 Number of Channels

Fig. 3 shows the throughput results with different numbers of channels (x-axis) and for two average packet lengths. The number of devices in each simulation is six times the number of channels. We assumed that the bandwidth per channel is 2 Mbps. The slot time is equal to 812 μs and the channel switching time is equal to 100 μs. For Split Phase, we assume the control and data durations to be 20 and 40 ms, respectively.

McMAC scales well with an increasing number of channels, whereas Single Rendezvous protocols do not. Note that, in the left plots, when the packets are small, the throughput of all the single channel protocols (Split Phase, Dedicated Control Channel, and Common Hopping) flattens out when the number of channels is greater than six because of the congestion of the control channel. With longer packets, the flattening of the curves occurs after a larger number of channels. It shows that Single Rendezvous protocols can use more channels efficiently when packets are longer. However, the bottleneck of the rendezvous channel still exists. The congestion is only delayed but *not* avoided as the number of channels increases.

The simulation results of [6] (Fig. 11) show that the performance of Split Phase exceeds that of Dedicated Control Channel as the number of channel increases. The

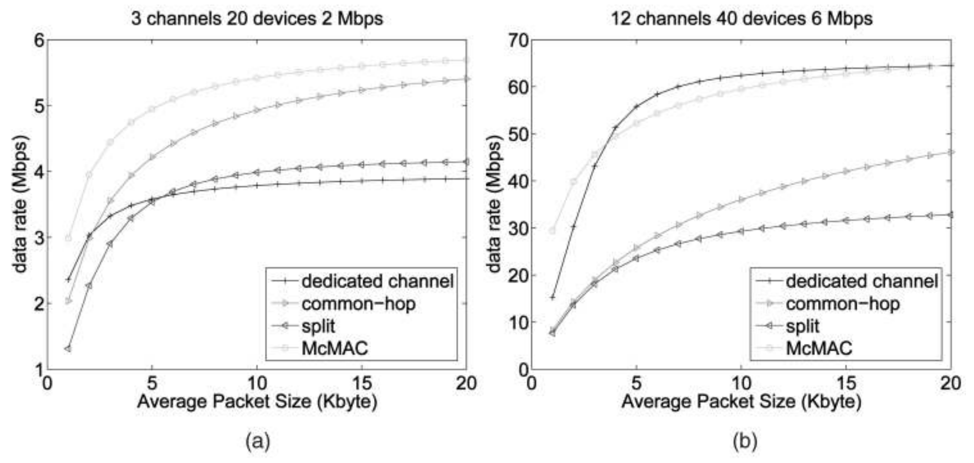


Fig. 4. Throughput versus packet size. (a) Slot time = 812 μs and (b) 200 μs. Switching time = 100 μs.

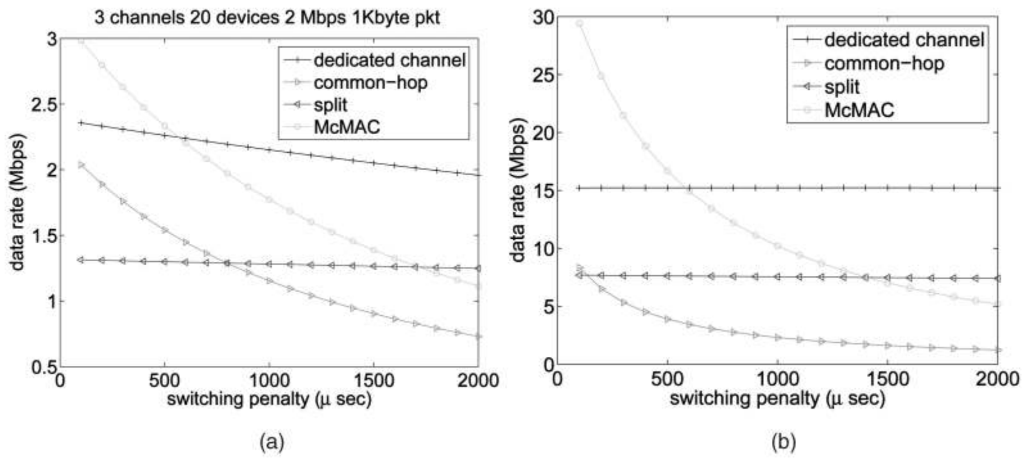


Fig. 5. Throughput versus switching penalty. (a) Slot time = 812 μs and (b) 200 μs.

authors of [6] compare MMAC, a Split Phase algorithm, with DCA, a Dedicated Control Channel approach, by using three to six channels. The aggregate throughput of MMAC with six channels was about 3.7 Mbps, whereas that of the DCA was 2.3 Mbps. Their results are opposite of our findings shown in Fig. 3. The discrepancy stems mainly from two differences. First, in the channel scaling study in [6] (Fig. 11), the authors assume that a sender can send multiple packets to the same receiver within each data phase because of queuing. Second, they assume that the traffic among different devices are disjoint such that each receiver will only communicate with one sender. When the traffic is nondisjoint, but queuing is still present, [6] shows that the performance gap between the two approaches narrows.

4.4 Average Packet Size

Fig. 4 shows the performance with different average packet sizes (x-axis) when the number of channels is three (Fig. 4a) and 12 (Fig. 4b). The throughputs of all schemes increase with the size of packets, which can be explained from the increase in utilization per agreement. Observe in Fig. 4a (three-channel case) that the Dedicated Control Channel becomes even worse than Split Phase with a packet size larger than 5 Kbytes. Dedicated Control Channel is bounded by 4 Mbps since one channel is used for control. The performance of Split Phase depends very much on the packet sizes. Its throughput increases to 3.5 Mbps with a 5-Kbyte packet from 1.2 Mbps with a 1-Kbyte packet. It is recommended that the Split Phase should transmit as much as possible when a mobile gains an access to a channel to maintain reasonable channel utilization. However, even when the packet size is large, its throughput is worse than that of the other protocols, as shown in the case of 12 channels (Fig. 4b). Another interesting observation in Fig. 4b is that the slope of the throughput of the dedicated channel is very steep. As the control channel congestion goes away with larger packet sizes, the slope increases quite fast.

4.5 Switching Penalty

Fig. 5 shows the performance as the switching penalty increases from 100 μs to 2,000 μs when the number of channels is three (Fig. 5a) and 12 (Fig. 5b), respectively (as a reference, commercial off-the-shelf 802.11 transceivers require about 150 to 200 μsec to switch channels). The throughput of Common Hopping and McMAC decrease faster, while those of Split Phase and Dedicated Control Channel are almost insensitive to the switching time. This is because the first two approaches are based on frequent hopping and incur a penalty every time they hop. When the switching penalty is high, Dedicated Control Channel and Split Phase are more efficient.

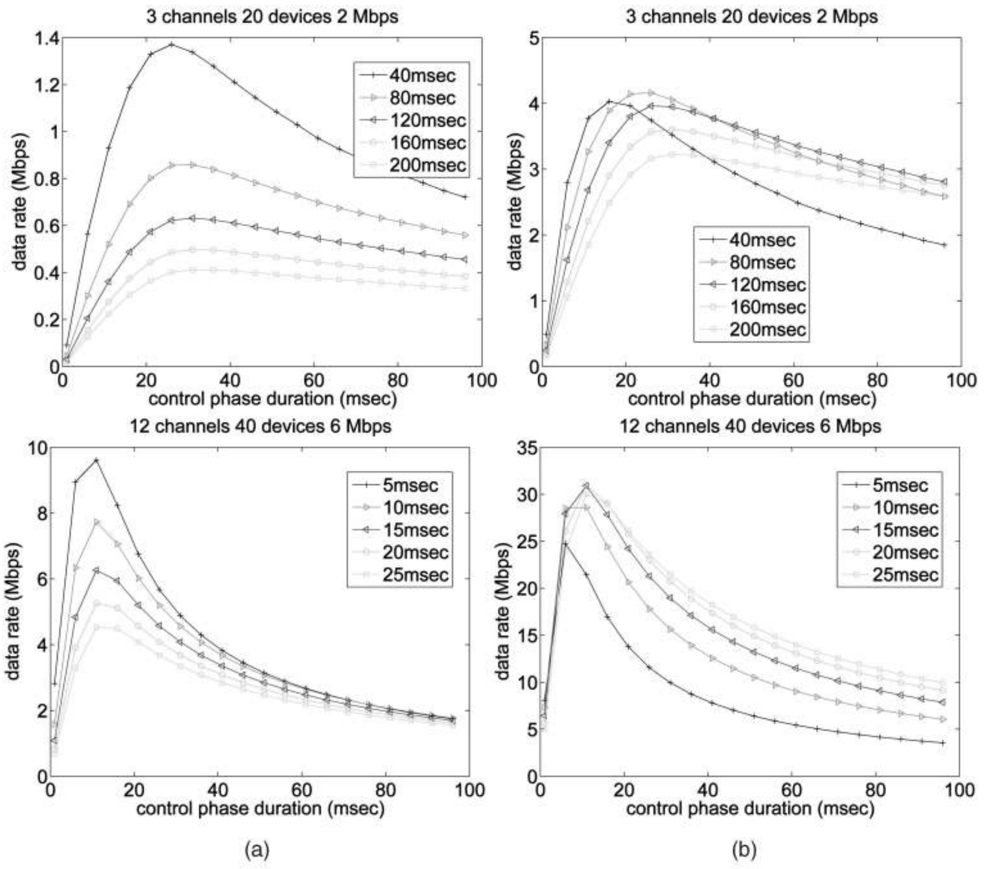


Fig. 6. Throughput versus control phase duration (x-axis) and data phase duration (different graphs). Slot time = 812 μs. Average packet size = (a) 1 Kbyte and (b) 10 Kbytes.

4.6 Split Phase

Four plots in Fig. 6 show the performance of Split Phase for different values of c, the control phase duration, and d, which is the data phase duration.

- *The performance of Split Phase is very sensitive to* c *and* d. All four plots show that there is a best value for c for any choice of d. Note that all graphs first increase then decrease. When c is too small, the devices cannot make enough agreements to use the given data phase duration. When c is too large, some of the control phase is wasted. For the three-channel cases, $c = 20$ ms and $d = 40$ ms correspond to a large throughput. For the 12-channel cases, $c = 10$ ms and $d = 10$ ms are preferable.

It is worth noting that the duration d of the data phase has little impact on the optimal control duration c. The optimal duration of the control phase is between 20 and 40 ms for the three-channel case. It is between 10 and 20 ms for the 13-channel case.

5 Simulation Results

5.1 Simulation Model

We developed a coarse-grained slotted-time simulator to compare the different protocols in more realistic settings. By coarse-grained simulation, we mean that the time scale of the simulation is in the order of one RTS/CTS transaction rather than a few microseconds. We did not model the detailed microsecond timing of short interframe space (SIFS), distributed coordination function interframe space (DIFS) durations, and so on, which is usually done in a discrete event simulation such as NS-2.

Our slotted-time simulation approach represents a compromise to model several different protocols without having to worry about the exact byte differences in the control packets of various specific protocols (more than 10 of them). Using a time-slotted simulator, we can focus on the intrinsic differences in the rendezvous mechanisms of various protocol families. More importantly, we feel that the performance differences among different families of protocols arise mainly from their rendezvous mechanisms and not from the way CSMA backoff parameters affect them. Therefore, modeling at the time scale of hundreds of microseconds is sufficient to reveal the major differences of the various protocol families (as we saw in the analysis and simulation sections). The time slot size is chosen such that it is small enough to model the overhead of control packets and data packet size differences without slowing the time-slotted simulator excessively.

Fig. 7 shows the slotted-time model that we used in the simulator. The duration of each slot, represented by one square, is as long as one RTS/CTS transaction time, that is, one agreement time. In each slot, with probability p_{succ}, a winner is chosen randomly among all the devices contending for a channel; otherwise, with probability $1 - p_{succ}$, no agreement is made. For example, in the analytical model in Section 3, p_{succ} is chosen from the slotted ALOHA model, of

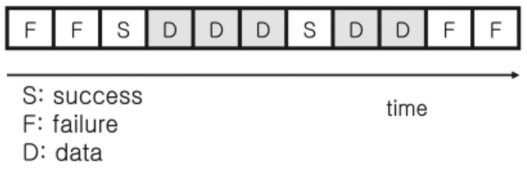


Fig. 7. Coarse-grained slotted-time simulation model.

which p_{succ} equals to $Np(1-p)^{N-1}$, where p is the transmission probability of each device and N is the number of devices. If an agreement is made (designated by "S"), then data transmission follows for a certain number of slots. Fig. 7 shows two successful rendezvous, each followed by data transfer of three and two slots, respectively.

One may argue that this model is an oversimplification of the situation since it is not clear how it represents the details of CSMA/CA, SIFS, DIFS, and the binary exponential backoff algorithm. We believe that, by selecting a suitable value of p_{succ}, we can mimic the long-term behavior of CSMA/CA. We choose p_{succ} such that the number of RTS/CTS successes in the CSMA/CA network is the same as that of the slotted model $D \cdot p_{succ}$, where D is the number of observed slots. By doing so, the simulation model can mimic the long-term behavior of rendezvous for a given duration of time. It is important to note that, since this approximation is used for all models, it results in a fair comparison.

Besides the freedom of p_{succ}, the simulation model is more realistic than the analytical models for the following reasons:

- *Arbitrary packet length distribution.* The packet length distribution can be arbitrary and does not have to be geometric. For example, we can simulate a constant-bit-rate (CBR) traffic that has a fixed packet length.
- *Arbitrary traffic model.* In the analysis, we assume that each device always has packets to transmit to all others at all times. However, the traffic model of the simulation can be Poisson, Markovian, and so forth.
- *Arbitrary receiver selection rule.* The selection of receiver is uniform among candidates in the analytical model. However, it can be arbitrary in the simulation model.
- *Queuing of packets.* In the simulation model, the queues of each device are traced exactly, while they are not in the analytical model. Therefore, all devices can transmit in the analytical model, while only those with a backlog will contend for medium and transmit in the simulation.
- *Channel status tracking.* In the analytical model, we only keep track of the number of currently busy channels. We assume that whether one particular channel is busy is independent of another. In the simulator, the status of all channels is tracked precisely in the simulation model.
- *Split Phase.* For the Split Phase protocol, to increase performance, we allow multiple pairs to share a channel in the same data phase if no more free channels are available, as in MMAC [6]. Since multiple pairs may share a channel, contention resolution is required in the data phase. Additionally, devices that did not make any channel agreement can go to the last (and usually the least utilized) channel to send data.

5.2 Sensitivity of Assumptions

Packet length distribution. To evaluate the impact of the geometric distribution assumption, we tried three different packet length distributions in the order of increasing variability: constant, geometric, and bimodal, each with the same mean. The bimodal distribution model generates a small packet with probability p and a large packet with probability $(1-p)$. Table 3 shows the sensitivity of simulation results as a function of the packet length distribution. The upper three and lower three rows correspond to 802.11b and 802.11a, respectively. Overall, we can observe a slight throughput degradation with increasing variability. However, the distribution of packet length does not impact the results much.

Impact of $\mathbf{p}_{succ}$. Since the parameter p_{succ} captures the behavior of CSMA/CA, it is important to understand its impact.

We first estimated the parameter in the 802.11b setting by using the NS-2 simulator. Fig. 8 shows that p_{succ} varies between 0.2 and 0.5 in the 802.11b environment (in the simulations, we used $p_{succ} = 1/e \approx 0.37$). To estimate p_{succ}, we position N devices randomly in a $200\,\mathrm{m} \times 200\,\mathrm{m}$ square region for $N = 10, 20, \cdots, 100$ and generate $\frac{N}{2}$ CBR flows, enough to saturate the network. We measured the total number of transmitted data packets for the duration of 200 seconds. We then used the following equation to compute p_{succ}:

$$p_{succ} = \frac{N_{succ}}{N_{ctrl\_slot}},$$

where N_{succ} is the number of successful RTS/CTS exchanges, and N_{ctrl_slot} is the number of control slots during the period of 200 seconds. To calculate N_{ctrl_slot}, we subtract the data transmission time from the 200 seconds

TABLE 3
Sensitivity of Throughput on the Packet Length Distributions (in Megabits per Second)

	Dedicated Control Channel	Split Phase	Common Hopping	McMAC
fixed (b)	1.3	1.5	1.3	2.1
exponential (b)	1.2	1.4	1.2	2.0
bimodal (b)	1.2	1.5	1.1	1.9
fixed (a)	28.6	20.3	17.4	35.4
exponential (a)	30.0	19.7	17.3	35.0
bimodal (a)	26.9	18.8	15.9	33.6

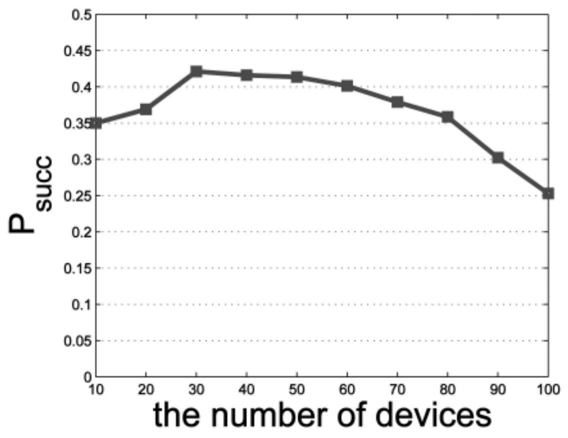


Fig. 8. Estimated p_{succ} under 802.11b setting in NS-2.

and divide the remaining time by the control slot duration, which is the RTS/CTS exchange time.

We varied p_{succ} from 0.2 to 0.8 in our simulation to check its impact. We used the 802.11a scenario with 12 channels and 40 devices. Each channel has the capacity of 6 Mbps. Table 4 shows the throughput results of different protocols in Mbps.

Observe that the value of p_{succ} has a high impact on the system throughput of all four protocols. As the value of p_{succ} increases, the throughput increases almost linearly. The implication is that the channel success probability is a critical parameter in the McMAC design. The amount of increase from 0.2 to 0.8 for each protocol is roughly 25 Mbps, 12 Mbps, 12 Mbps, and 17 Mbps, respectively. Note that the dedicated channel approach is most sensitive to the contention success probability, whereas Common Hopping and Split Phase are less sensitive than the others. It is interesting to see that the less sensitive protocols are the less efficient ones in terms of throughput. One of the reasons that Split Phase and Common Hopping are less sensitive can be that they are limited by factors other than p_{succ}. For example, the short control duration limits the performance of Split Phase.

Another interesting result is that, although all four protocols showed a throughput increasing with p_{succ}, the ordering of the throughput of the four protocols does not change with that parameter. The order is always McMAC $>$ Dedicated Control Channel $>$ Common Hopping $\approx$ Split Phase, from the highest to the lowest. The gap between the first (McMAC) and second (Dedicated Channel) reduces with increasing p_{succ} though.

5.3 Impacts of Receiver Contention

For a multichannel MAC, the ideal traffic pattern is a disjoint one, in which each sender only communicates with one receiver, and vice versa. When the traffic is non-disjoint, it is possible for a sender to pick a busy receiver, thereby wasting time and channel. To investigate the impact of receiver contention, we vary the number of simultaneous CBR connections per device in this section. Each device sets up CBR connections to δ distinct destinations. δ is known as the degree of communication. The first device chooses δ different destinations. The rest of

TABLE 4
Throughput for Different Values of p_{succ} (Mbps)

p_{succ}	Ded. Cntl Channel	Split Phase	Common Hopping	McMAC
0.2	8.6	6.9	6.0	18.7
$\frac{1}{e}$	15.6	10.3	10.2	26.2
0.6	25.1	15.8	14.8	33.2
0.8	33.9	19.6	18.6	36.5

the nodes choose up to δ destinations if they are involved in fewer than δ connections. Only destinations that have fewer than δ connections are eligible. A larger value of δ corresponds to more contention for receiver devices and, likely, lower efficiency and longer delays.

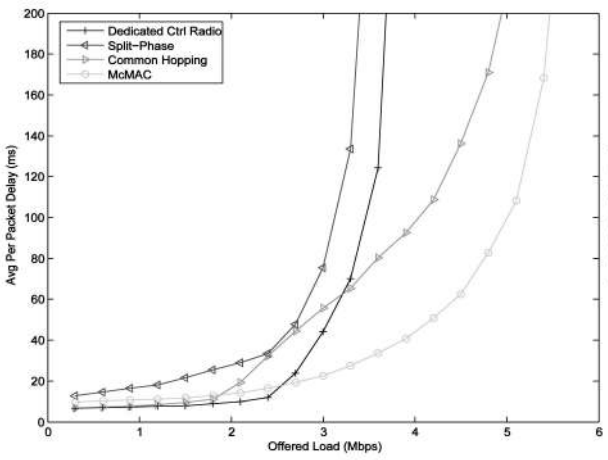
Fig. 9 shows the throughput of the various schemes as δ increases. Figs. 9a, 9c, and 9e use the 802.11b parameters, whereas Figs. 9b, 9d, and 9f correspond to the 802.11a scenario. Figs. 9a and 9b correspond to $\delta = 1$, Figs. 9c and 9d to $\delta = 2$, and Figs. 9e and 9f to $\delta = 5$. We summarize our observations as follows:

- *Throughput degradation.* Given a fixed load, as δ increases, the average amount of data sent over each pair of devices decreases. Therefore, one expects a shorter communication to take place after each channel agreement. As a result, the frequency for making channel agreements increases, thereby decreasing the throughput for all schemes.

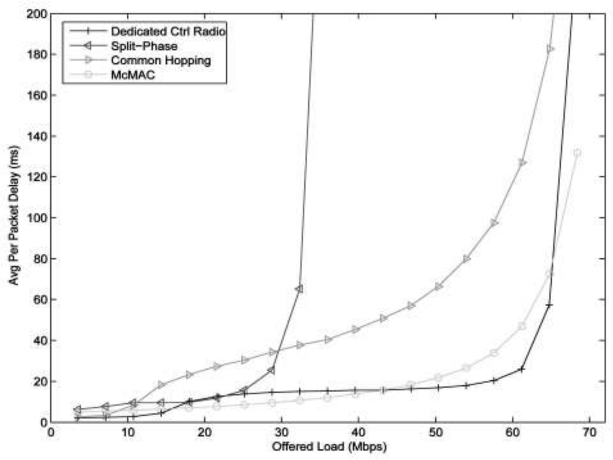
  Moreover, for Split Phase, a further decrease in the throughput, as δ increases, results because each sender has to visit more receivers to deliver its packets. When these receivers have chosen different channels during a data phase, a sender can only deliver a fraction of packets to the subset of receivers on the same channel, resulting in further loss of efficiency.

  For Common Hopping, each device cannot keep track of which other devices are busy. When $\delta = 1$, the receiver is always available when a sender successfully wins contention on the common hop since each receiver talks to only one sender. However, when δ is large, the probability that any particular receiver is available approaches $\frac{N-2k-1}{N-1}$, where k is the number of busy device pairs. This probability is small when the fraction of devices that are busy is large (that is, high utilization and a small total number of devices compared to the number of channels). As a result, the last few idle channels are more difficult to use efficiently.

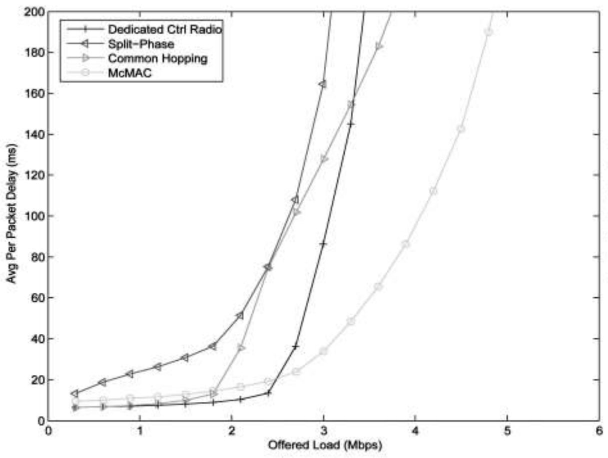
  For McMAC, the cause of degradation is similar to that of Common Hopping. However, since it allows parallel rendezvous, McMAC suffers less than Common Hopping when the channel agreement traffic increases.
- *Insensitivity of Dedicated Channel.* Looking at the graphs on the left (that is, the 802.11b scenario), we observe that Dedicated Control Channel is insensitive to the degree of communication because it knows perfectly the busy status of the channels and devices by using a second radio. Since one of the three channels is set aside for control communication, the amount of rendezvous traffic is easily absorbed by



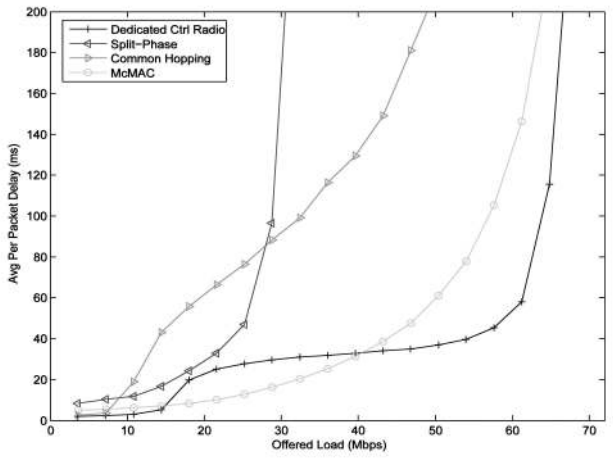
(a)



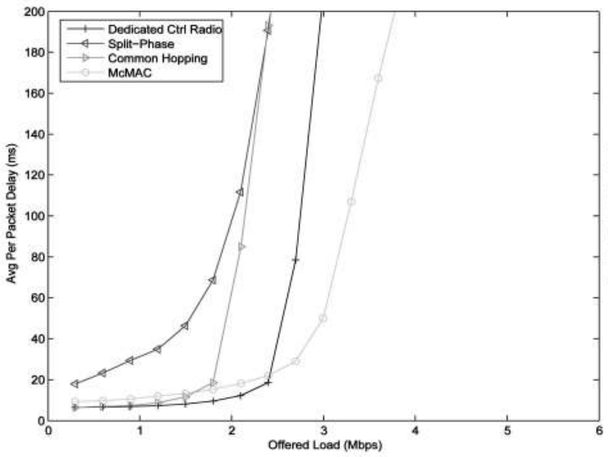
(b)



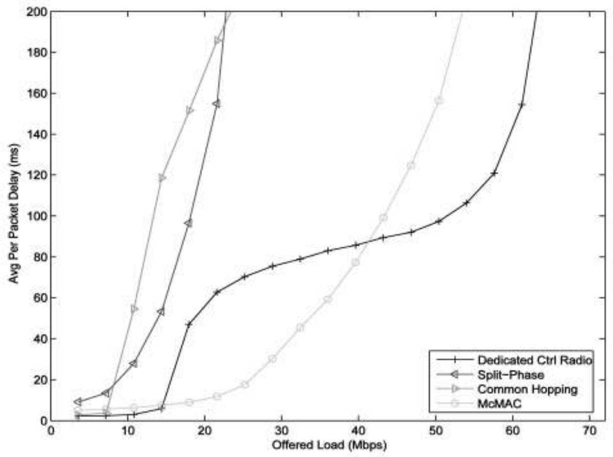
(c)



(d)



(e)



(f)

Fig. 9. Average per packet delay versus offered CBR traffic load. (a) 802.11b setting, $\delta = 1$. (b) 802.11a setting, $\delta = 1$. (c) 802.11b setting, $\delta = 2$. (d) 802.11a setting, $\delta = 2$. (e) 802.11b setting, $\delta = 5$. (f) 802.11a setting, $\delta = 5$.

the abundance of control channel capacity. As the number of available channels increases, the cost of using an extra channel for control purposes is relatively small compared to the gain achieved by knowing the status of the channels and devices. Consequently, Dedicated Control Channel achieves the highest throughput, except when $\delta = 1$.

- *Nonconvex delay curve of Dedicated Control Channel.* The delay curve is not convex under the 802.11a settings, and this requires some explanation. We also comment on the relationship between δ and the delay. The average delay in this case can be viewed as a sum of two stages. First, a pair of devices must rendezvous on the control channel. Second, they transfer one or more packets on the agreed data channel. The two stages saturate under different conditions.

Assuming $p_{succ} = 1/e$, the rendezvous process can generate at most $\frac{1}{e} \approx 0.37$ new agreement during every slot on the average. In the 802.11a scenario, each 1,024-byte packet lasts roughly seven time slots. To fully utilize the 12 channels, the rendezvous process must generate $12/7 > 1$ new channel agreements per time slot, which is not possible without sending multiple packets per rendezvous. Therefore, the control channel will saturate before the data channels do as the load increases.

The control channel saturates at a load of roughly $\frac{1}{e}/\frac{12}{7}$ or 21.46 percent. The maximum throughput reachable before it saturates is (0.2146 $\times$ 6 Mbps/channel $\times$ 12 channels) or 15.45 Mbps. At a load below 15.45 Mbps, the delay is very low because both stages are underutilized. Once the first stage starts saturating, the delay increases quickly for the first time. This explains the first jump in the delay at around 15 Mbps.

As the load increases, the number of queues waiting to be serviced in stage 1 remains constant due to the fixed traffic pattern, and the average service delay in stage 1 is relatively insensitive to the load. Since the total delay is dominated by the stage 1 delay before the second stage saturates, the delay rises very slowly. Finally, when the load approaches the capacity of the data channels, the second stage saturates and, hence, the delay increases rapidly again.

Next, we note that, when only stage 1 has saturated, the delay at any particular load is roughly proportional to δ. For instance, the delays at 40 Mbps for $\delta = 1, 2, 5$ are in the ratios of roughly 1:2:5. Once the first stage saturates, one can view it as a random scheduler that serves the $N \times \delta$ always-backlogged queue pairs at a rate of $1/e$ in an independent and identically distributed (i.i.d.) fashion. The arrival rate of the second stage is equivalent to the output rate of the first stage. The service time of the second stage represents the time required to empty the sender's queue. The average amount of data found in a queue is proportional to the delay of stage 1 and inversely proportional to δ. Since the delay of stage 1 is proportional to δ, the service time of stage 2 is roughly independent of δ.

In summary, as δ increases, stage 1 slows down proportionally, whereas stage 2 remains at about the same rate, resulting in a delay that is proportional to δ.

5.4 Impact of Limiting the Medium Occupancy Time

In this section, we evaluate the delay under a mixture of long-lived CBR flows and some random file transfers. Allowing a device to transmit until the queue becomes empty can be helpful in achieving a higher long-term throughput because more data can be transferred per channel agreement. However, the delay and/or jitter statistics experienced by CBR flows might become worse. We are interested in the effects of limiting the maximum occupancy time on throughput, delay, and jitter.

In this scenario, we randomly pair up each device with one another and add a CBR connection between them. The amount of CBR traffic of each connection is the same, adding up to 10 percent of the channel utilization. Then, we add random i.i.d. file transfers with geometrically distributed lengths among every pair. The mean file size is 10 Kbytes. The arrival rate of the files is adjusted to give a total offered load between 10 percent (that is, no file traffic) and 90 percent (that is, 80 percent file traffic). MAC protocols do not schedule the two types of packets differently. Rather, they serve the packets in a first-come, first-served order for the same destination and randomly across different destinations.

We ran the simulation experiments with two sets of network parameters. However, due to space constraints and the similarity between the two, we present only the results pertaining to the 802.11a scenario. Fig. 10 shows the effects of the medium occupancy limit on the delay and jitter and on throughput, respectively. The left column of the graphs shows the delay experienced by CBR traffic, whereas the right column shows the standard deviation of delay. Under the 802.11a settings, the different medium occupancy limits are 3.3 ms (top graphs) and 10.5 ms, respectively.

Notice that, in all cases, the increase in the medium occupancy limit reduces the delay for the CBR traffic. Furthermore, the delay standard deviation curves are almost identical to the delay curves. The curves show that, in all cases, the delay and delay jitter get smaller as one increases the medium occupancy limit. Therefore, allowing devices to occupy the medium longer benefits not only the file transfer traffic but also the CBR traffic by reducing both delay and jitter.

In every setting, Split Phase is quite insensitive to the medium occupancy limit. As we saw in Fig. 9, when the traffic is nondisjoint, the performance of Split Phase is severely reduced because a device can only communicate with a small subset of other devices during each data phase. Therefore, the small amount of traffic sent to each destination is insufficient to benefit from a longer occupancy limit.

Common Hopping improves tremendously as one increases the maximum occupancy limit, indicating that the rendezvous process is indeed a bottleneck.

McMAC improves only moderately because the reduction in rendezvous traffic has less effect on Parallel Rendezvous protocols, where rendezvous is less of a bottleneck.

In the 802.11a setting, the achievable throughput of Dedicated Control Channel increases very dramatically because the rendezvous is the bottleneck (it improves only a little in 802.11b because the rendezvous process is no longer the bottleneck).

6 Future Work

Our study has focused on a single collision domain with no hidden or exposed node. Unfortunately, these nodes can impact our results in a network with multiple collision domains. We are currently working on extending the results for the multiple domain case. Another future work is the development of a discrete-event-based simulator to more

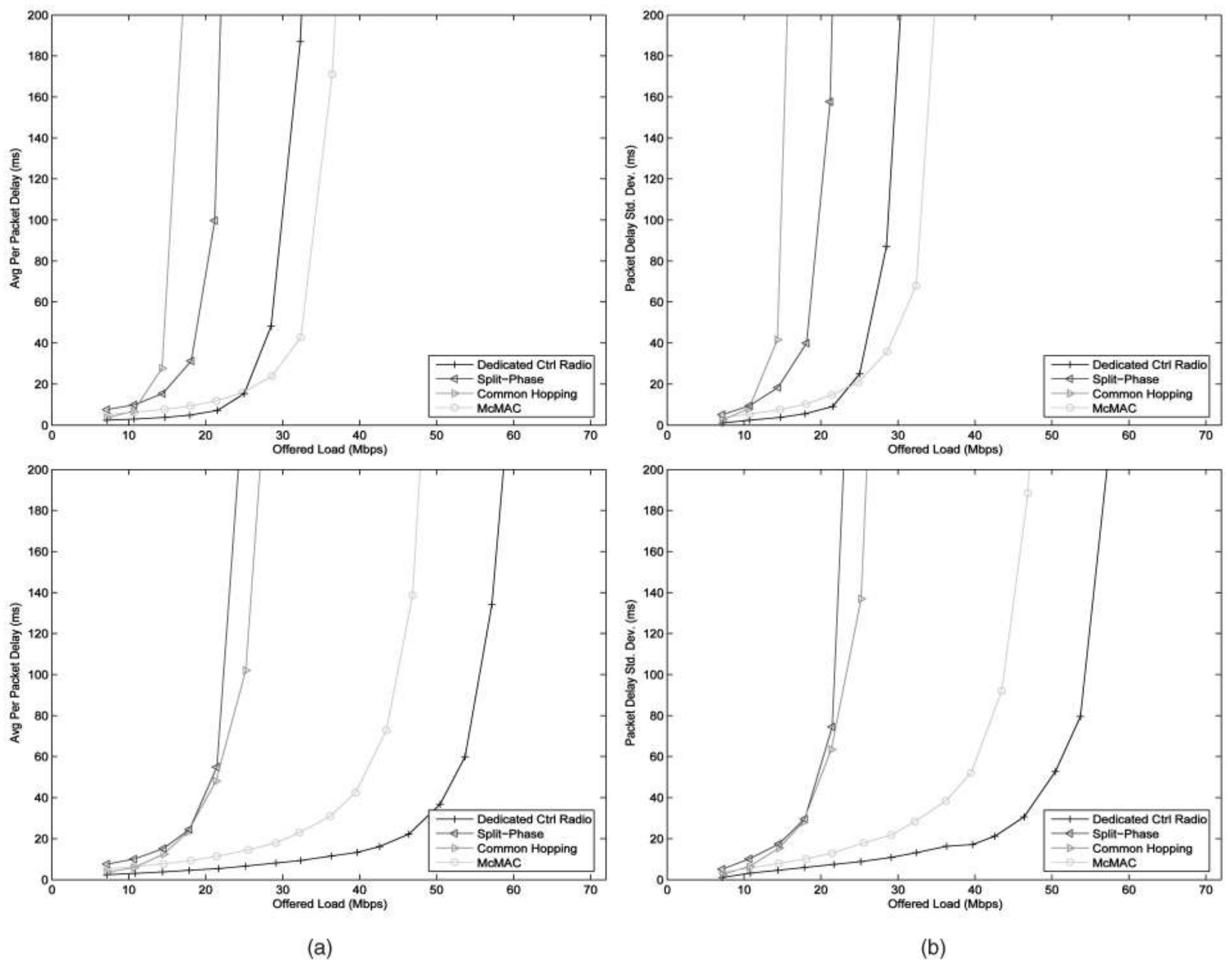


Fig. 10. The effects of medium occupancy limit on delay and delay standard deviation (jitter) versus throughput under the 802.11a scenario. The delay graphs are shown in (a) and the jitter curves are shown in (b).

TABLE 5
Summary of Comparison of Four Representative Protocols under Different Operating Conditions

Parameter Investigated	Dedicated Control Channel	Split Phase	Common Hopping	McMAC
Scalability to Many Channels	Long packets – Good Short packets – Limited	Limited	Limited	Good
Sensitivity to Packet Length (Lower is better)	Many channels – High Few channels – Low	High	High	Medium
Sensitivity to Channel Switching Time (Lower is better)	Low	Low	Very High	High
Sensitivity to Receiver Contention (Lower is better)	Low	Medium	High	Medium

accurately model the timing of RTS/CTS events. Furthermore, in the simulator, the physical-layer model of the channel can be better represented, taking into account different channel qualities among different node pairs.

SSCH is another Parallel Rendezvous MAC protocol, proposed by Bahl et al. [1]. In this study, we picked McMAC instead of SSCH in the evaluation of Parallel Rendezvous versus Single Rendezvous protocols because SSCH is more difficult to model analytically due to their hopping sequence adaptation heuristics. The comparison of the two protocols remains as future work.

7 Conclusions

In this paper, we classified various multichannel MAC protocols into four general categories—*Dedicated Control Channel*, *Common Hopping*, *Split Phase*, and *McMAC*—based on how they make an agreement. We then compared the protocols by using both analysis and simulation. We developed analytical models for the four protocols by using Markov chains, and we simulated them with a time-slotted simulator under a variety of operating conditions. Table 5 summarizes our findings.

In general, Parallel Rendezvous protocols such as McMAC perform better than Single Rendezvous protocols with one radio under a wide range of situations by eliminating the rendezvous channel bottleneck. However, under the special case when many channels are available and packets are long, Dedicated Control Channel performs the best. When traffic is nondisjoint, meaning each node needs to communicate with more than one other node, all protocols except Dedicated Control Channel suffer from a reduced throughput because they only have one radio and hence cannot monitor the status of other nodes and channels perfectly.

To note, both Split Phase and Dedicated Control Channel protocols explicitly separate control packets from data packets. However, doing so does not necessarily improve performance because generating more successful rendezvous is useless when the data channels are already congested, and vice versa. Specifically, the performance of Split Phase depends heavily on the choice of control and data phases duration.

Finally, we found that, when MAC protocols treat various packet types (for example, real-time video and file transfers) equally, allowing a sender to transmit multiple packets to the same destination after each rendezvous is highly beneficial. The throughput, delay, and jitter improve for all types of packets by using any of the four protocols. This suggests that improving the efficiency of the network by allowing each sender to use the channel longer is more beneficial than enforcing short-term fairness in reducing delays.

ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation under Grant CNS-0435478 and in part by the BrOMA ITRC program of the Korean MIC.

REFERENCES

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

[2] J. Chen, S. Sheu, and C. Yang, "A New Multichannel Access Protocol for IEEE 802.11 Ad Hoc Wireless LANs," *Proc. 14th IEEE Int'l Symp. Personal, Indoor and Mobile Radio Comm. (PIMRC '03),* vol. 3, pp. 2291-2296, Sept. 2003.

[3] F. Herzel, G. Fischer, and H. Gustat, "An Integrated CMOS RF Synthesizer for 802.11a Wireless LAN," *IEEE J. Solid-State Circuits,* vol. 18, no. 10, Oct. 2003.

[4] Maxim Integrated Products, Inc., "MAX2820, MAX2820A, MAX2821, MAX2821A 2.4GHz 802.11b Zero-IF Transceivers Data sheet rev. 04/2004," 2004.

[5] H.W. So, J. Walrand, and J. Mo, "McMAC: A Multi-Channel MAC Proposal for Ad Hoc Wireless Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC '07),* Mar. 2007.

[6] J. So and N. Vaidya, "Multi-Channel MAC for Ad Hoc Networks: Handling Multi-Channel Hidden Terminals Using a Single Transceiver," *Proc. ACM MobiHoc,* May 2004.

[7] Z. Tang and J. Garcia-Luna-Aceves, "Hop Reservation Multiple Access (HRMA) for Multichannel Packet Radio Networks," *Proc. Seventh IEEE Int'l Conf. Computer Comm. and Networks (IC3N '98),* Oct. 1998.

[8] A. Tzamaloukas and J.J. Garcia-Luna-Aceves, "Channel-Hopping Multiple Access," *Proc. IEEE Int'l Conf. Comm. (ICC '00),* June 2000.

[9] A. Tzamaloukas and J. Garcia-Luna-Aceves, "Channel-Hopping Multiple Access with Packet Trains for Ad Hoc Networks," *Proc. IEEE Device Multimedia Comm. (MoMuC '00),* Oct. 2000.

[10] S.-L. Wu, Y. Lin, Y.-C. Tseng, and J.-P. Sheu, "A New Multi-Channel MAC Protocol with On-Demand Channel Assignment for Mobile Ad Hoc Networks," *Proc. Int'l Symp. Parallel Architectures, Algorithms and Networks (ISPAN '00),* p. 232, Dec. 2000.

[11] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, C.-Y. Lin, and J.-P. Sheu, "A Multi-Channel MAC protocol with Power Control for Multi-Hop Mobile Ad Hoc Networks," *The Computer J.,* vol. 45, no. 1, pp. 101-110, 2002.

[12] W.-C. Hung, K.L.E. Law, and A. Leon-Garcia, "A Dynamic Multi-Channel MAC for Ad Hoc LAN," *Proc. 21st Biennial Symp. Comm.,* pp. 31-35, June 2002.

[13] M. Zannoth, T. Ruhlicke, and B.-U. Klepser, "A Highly Integrated Dual-Band Multimode Wireless LAN Transceiver," *IEEE J. Solid-State Circuits,* vol. 39, no. 7, pp. 1191-1195, July 2004.

Jeonghoon Mo received the BS degree from Seoul National University, Korea, and the MS and PhD degrees from the University of California, Berkeley. He is an associate professor at the School of Engineering, Information and Communications University (ICU), Korea. Before joining ICU in 2003, he was with AT&T Laboratories, Middletown, New Jersey, and with start-ups in San Jose, California. He worked for voice-over-IP (VoIP) quality assessment and performance analysis of network processor. His research interests include transport/media access control (MAC) design of communication networks, WiMax, Wi-Fi, queuing theory, optimization, and quality of service. He is also involved with the WiMax Forum. He is a member of the IEEE.

Hoi-Sheung Wilson So received the BS degree in computer science from Cornell University in 1997 and the MS and PhD degrees in computer science from the University of California, Berkeley, in 2000 and 2006, respectively. He is currently with the Siemens Technology-to-Business Center, Berkeley. His current research interests include wireless protocol design and multichannel media access control (MAC) protocol for wireless networks. He is a member of the IEEE.

Jean Walrand is a professor in the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. His research interests include stochastic processes, queuing theory, communication networks, and control systems. He is the author of *An Introduction to Queueing Networks* (Prentice Hall, 1988) and *Communication Networks: A First Course* (McGraw-Hill, 1998) and coauthor of *High-Performance Communication Networks* (Morgan Kaufmann, 2000). He is a recipient of the Lanchester Prize from the Operations Research Society of America and of the S. Rice Prize from the IEEE Communications Society. He is a fellow of the IEEE.