UNIVERSITY of CALIFORNIA
SANTA CRUZ

SENDER- AND RECEIVER-INITIATED MULTIPLE ACCESS PROTOCOLS FOR AD-HOC NETWORKS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Asimakis E. Tzamaloukas

December 2000

The dissertation of Asimakis E. Tzamaloukas is approved:

Professor J.J. Garcia-Luna-Aceves, Chair

Professor Patrick Mantey

Professor Phokion Kolaitis

Vice Provost and Dean of Graduate Studies

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
DEC 2000		**00-12-2000 to 00-12-2000**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Sender- and Receiver-Initiated Multiple Access Protocols for Ad-Hoc Networks	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Santa Cruz,Department of Computer Engineering,Santa Cruz,CA,95064**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**157**	

Contents

List of Figures

List of Tables

Abstract

Sender- and Receiver-Initiated Multiple Access Protocols for Ad-Hoc Networks

by

Asimakis E. Tzamaloukas

This thesis focuses on the medium access control (MAC) layer. Many MAC protocols for wireless networks proposed or implemented to date are based on collision-avoidance handshakes between sender and receiver. The key objective of collision-avoidance handshakes is reducing or eliminating the collision of data packets from a source at any given receiver due to interference from packets from other sources. In the vast majority of these protocols, including the IEEE 802.11 standard, the handshake is sender initiated, in that the sender asks the receiver for permission to transmit using a short control packet, and transmits only after the receiver sends a short clear-to-send notification. There are two main objectives in this work: analyze the effect of reversing the collision-avoidance handshake as a way to improve the performance of MAC protocols under any conditions in the network, and design MAC protocols that provide correct floor acquisition without carrier sensing or code pre-assignment.

We show that receiver-initiated collision-avoidance MAC protocols not only outperform any sender-initiated ones, but also guarantee collision-free data transmission and seamless support for mobility by using simple, low-cost wireless radios. We study the effect of persistent carrier sensing in receiver as well as sender-initiated MAC protocols. We extend our work to multi-channel radios and introduce novel collision-avoidance MAC protocols that do eliminate the need for carrier sensing and code pre-assignment, and improve the utilization of the medium in the presence of unicast, multicast and broadcast traffic.

To my parents,

Sofia and Elias

Acknowledgements

There is no doubt in my mind that pursuing a doctorate degree under the supervision of J.J. Garcia-Luna-Aceves is the best thing that could ever occur in my academic career. Working with J.J. has always been fun for me. I feel honored that I had J.J. as my advisor and I would like to thank him from the bottom of my heart for believing in me throughout all these years.

I would also like to thank Professors Patrick Mantey and Phokion Kolaitis for taking the time to read and evaluate this dissertation.

I am obliged to Christos Tryfonas for his help during my first steps in this country. Chane Fullmer has been a good inspiration and support for me while looking for a thesis direction. Marcelo Spohn and Paul Tatarsky taught me many things in the programming arena and I would be always grateful to them.

The text of this dissertation is based on material that has been previously published [98, 42, 99, 101, 103, 100, 104, 102] or accepted for future publication [41]. Professor J.J. Garcia-Luna-Aceves, listed as co-author in these publications supervised the research that is presented in this dissertation.

This work was supported in part by the Defense Advanced Research Projects Agency under Grant F30602-97-0338.

Chapter 1

Introduction

Even though packet radio networks were first introduced more than three decades ago with the ALOHA system [4], their cost has been prohibitive until recently. Recent advances in VLSI technology have made it possible to have wireless, ad-hoc network installations all over the world. The distributed, self-configured capability of ad-hoc networks makes them very attractive for tactical communications as well as a way to build a *global wireless internet*. Laptop computers, personal digital assistants (PDA), pagers, and lately wearable computing devices are just a few forms of mobile communicating devices that ramp up the need for connectivity from every place the user might be.

To minimize losses of data due to collisions, data networks need a mechanism to regulate the access on the transmission medium. *Medium Access Control* (MAC) protocols control access to the shared communication medium so that it is used efficiently. Because in wireless networks the available medium (in frequency or time) is scarce, its utilization is of utmost importance; hence, a MAC protocol is a critical part of the network stack that determines to a large extent the correct and efficient operation of the wireless network.

We focus our research on new MAC protocols for wireless ad-hoc networks. Designing a new MAC protocol for ad-hoc networks must take into account the many different parameters of

radio links, which make the problem of *sharing* the medium non trivial. Because the nodes in an ad-hoc network may move, the quality of the wireless link between any two nodes constantly changes. Physical obstacles, propagation delays, and interference from other nodes are just some parameters that change constantly in an ad-hoc network. Furthermore, the applications running on different nodes impose a wide range of *needs* to be satisfied. For example, voice and video applications (in general real-time applications) give higher priority to the end-to-end delay encountered by packets than to the maximum throughput that can be achieved for a short period of time. On the other hand, a large multicast group could be more interested in minimizing the overhead of sending multiple copies of the same packet than in having the lowest delay possible to only one member of the multicast group.

There is a large body of work on the design of MAC protocols for wireless networks. Kleinrock and Tobagi [57] identified the hidden-terminal problem of carrier sensing, which makes carrier-sense multiple access (CSMA) perform as poorly as the pure ALOHA protocol when the senders of packets cannot hear one another and the vulnerability period of packets becomes twice a packet length. The BTMA (busy tone multiple access) protocol was a first attempt to solve the hidden-terminal problem by introducing a separate busy tone channel [95]. The same authors proposed SRMA (split-channel reservation multiple access) [96], which attempts to avoid collisions by introducing a control-signal handshake between the sender and the receiver. A station that needs to transmit data to a receiver first sends a request-to-send (RTS) packet to the receiver, who responds with a clear-to-send (CTS) if it receives the RTS correctly. A sender transmits a data packet only after receiving a CTS successfully. ALOHA or CSMA can be used by the senders to transmit RTSs.

Several variations of this scheme have been developed since SRMA was first proposed, including MACA [55], MACAW [13], IEEE 802.11 [3], and FAMA [36]. These examples, and most protocols based on collision-avoidance handshakes to date are sender-initiated, in that the node wanting to send a data packet first transmits a short RTS asking permission from the receiver.

In contrast, in the MACA by invitation (MACA-BI) protocol [92], the receiver polls one of its neighbors asking if it has a data packet to send. A receiver-initiated collision-avoidance strategy is attractive because it can, at least in principle, reduce the number of control packets needed to avoid collisions. However, as we show in Chapter 2, MACA-BI cannot ensure that data packets never collide with other packets in networks with hidden terminals.

Under heavy load, almost all the nodes repeatedly try to access the medium, resulting in successive collisions between RTSs that degrade the performance of the network substantially. Collision resolution schemes [38, 40, 39], have been proposed that improve the network performance under heavy load.

An alternative solution to collision-avoidance for applications that need to satisfy strict delay guarantees are the transmission-scheduling MAC protocols based on [32, 16, 17]. Due to the fact that the minimum-length scheduling problem is NP-complete, the protocols in this category perform optimally only for a certain set of network configurations. An interesting approach has been proposed recently [63, 64] in which the nodes that have a packet to send, join a transmission group. During a frame, there is a contention-based period during which the nodes try to join the group, and a contention-free period during which the nodes of the group transmit their data, collision free. The drawback of this scheme is that all the nodes need to agree on the duration of the contention-based and contention-free periods.

Lately, the unlicensed Industrial Scientific Medical (ISM) bands have been the target of many commercial, affordable radios. At this time, the 915MHz, 2.4GHz and 5.8GHz unlicensed bands are available in North America. Proposed rule-making is expected to bring another two unlicensed bands: a 5GHz single band at 59GHz, and an additional 3.5GHz in seven bands between 71GHz and 153GHz. Figure 1.1 gives a picture of the distribution of unlicensed bands in Europe and North America. The 2.4GHz band is available worldwide.

According to the FCC Part 15 regulations [1], spread spectrum technology should be used to allow coexistence of multiple networks when operating over an ISM band. The transmitted

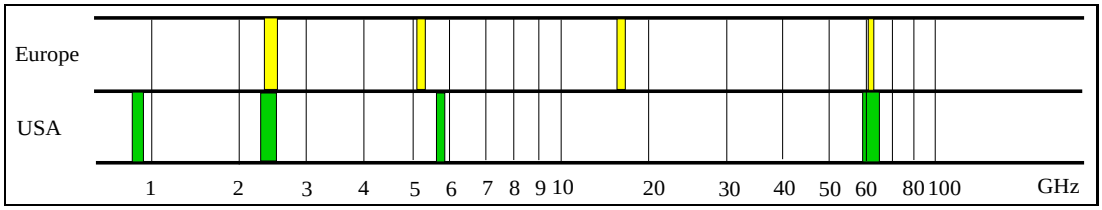


Figure 1.1: ISM bands in the world. The rectangles in gray shades correspond to the actual frequency range that a given ISM band covers

signals power and the bandwidth of the signal are limited by government regulation. Two major implementations for the underlying physical layer (PHY) have been proposed: Direct Sequence Spread Spectrum (DSSS) and Frequency Hopping Spread Spectrum (FHSS). Details about DSSS and FHSS can be found elsewhere [1] [3]. There is a large amount of work arguing in favor of one or another physical layer implementation. For the 2.4GHz band, which is the focus of many vendors recently, the IEEE 802.11 standard uses a FH physical layer with 79 non-overlapping frequency channels with 1MHz channel bandwidth. A predefined computer-generated pseudo-random list of 79 frequencies is generated by adding an offset to the base frequency, modulo 79. Since the minimum frequency distance to avoid interference is 6MHz there are at most 22 hopping patterns that can coexist at the same time without any collisions (using BFSK). Figure 1.2 shows a small subset of two non overlapping hopping patterns. The corresponding DS scheme has 11 different channels as shown in Figure 1.3. Important benefits that the SS technology provides among others are: robustness against multi-path propagation, the hidden terminal problem occurs only when the hidden nodes are close to each other, increased security, not prone to fading, capable to capture a packet even when two or more packets overlap.

As a natural evolution of single-channel medium access, several MAC protocols have been proposed and analyzed that take advantage of spreading codes for multiple access. Sousa and Silvester [88] presented and analyzed various spreading-code protocols that are sender-, receiver- or sender-receiver based, i.e., in which codes are assigned to senders, receivers, or combinations. Gerakoulis et. al. [43] used carrier sensing to propose a receiver-based, asynchronous transmissions

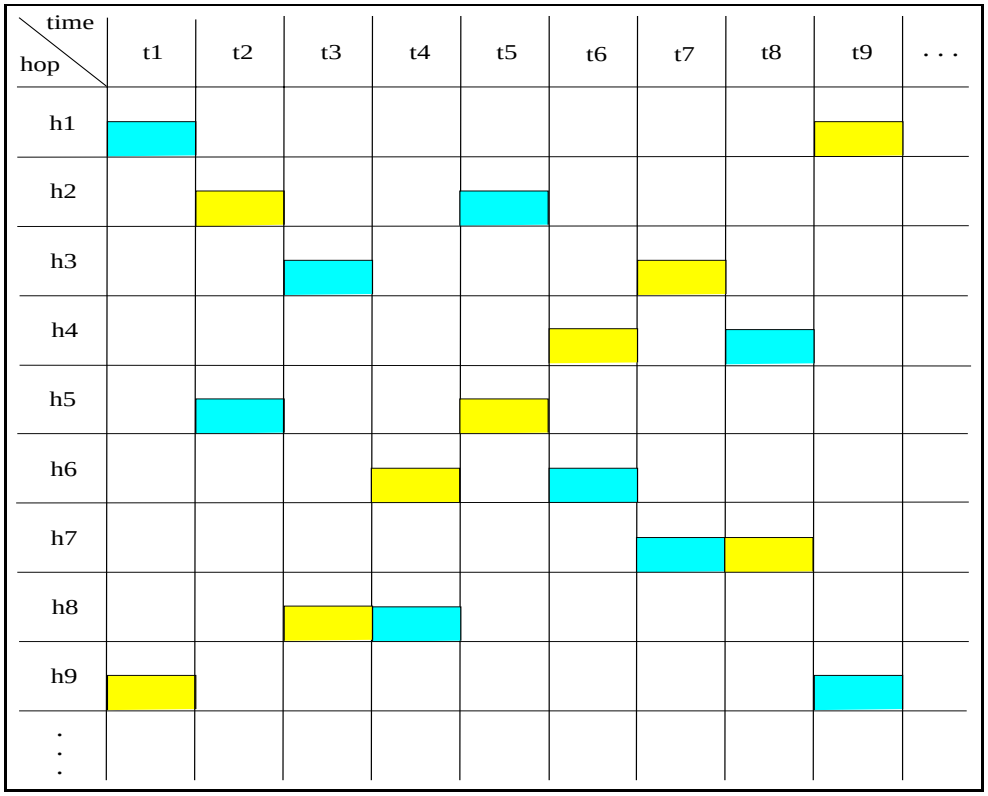


Figure 1.2: Two radio pairs exchanging data using FHSS

protocol. Jiang and Hsiao [51] proposed a receiver-based handshake protocol for CDMA (code division multiple access) networks that improved the efficiency of the network by reducing the amount of unsuccessful transmissions and unwanted interference. Several other proposals have been made to implement correct collision-avoidance in multi-hop networks without requiring nodes to use carrier sensing; these proposals rely on multiple codes assigned to senders or to receivers to eliminate the need for carrier sensing (e.g., [34, 39, 52]).

The key limitation of protocols based on predefined code assignments is that senders and receivers have to find each others' codes before communicating with one another. Most of the commercial DSSS radios today use only 11 chips per bit; therefore, CDMA is not an option. Future DSSS are expected to use 15 chips per bit, allowing two different systems to operate over the same DS frequency channels as they were defined in IEEE 802.11 [3]. On the other hand, up to 26 FHSS radios can be co-located. According to the FCC regulations, up to 15 FHSS radios can be co-

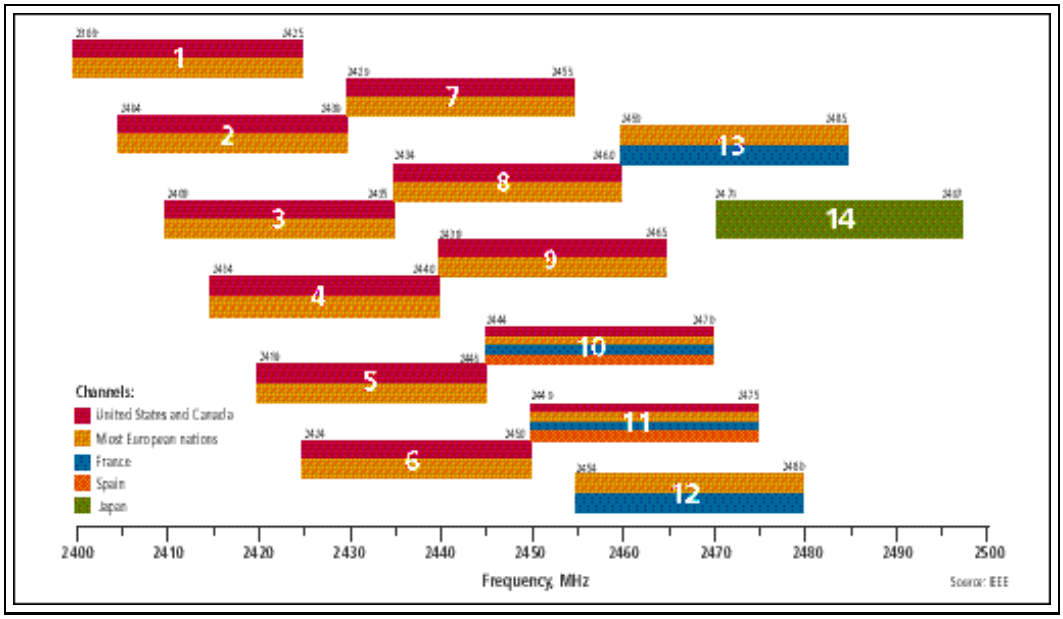


Figure 1.3: The DSSS available channels in the 2.4GHz ISM band

located with minimum interference problems. For wireless LANs (compatible with IEEE 802.11b) the number of co-located users is fixed and in most cases the network is customized towards higher data rates. In this case, DSSS is preferred at a slightly higher cost. However in ad-hoc networks built with commercial radios operating in ISM bands, code assignments do not guarantee that receivers can capture one of multiple simultaneous transmissions, and the number of users in a given area might be changing rapidly. Slow frequency hopping (with one or more packets sent per hop) is the viable way to achieve multiple orthogonal channels. Therefore, for ad-hoc networks it becomes imperative to develop MAC protocols that can take advantage of the characteristics of FHSS radios operating in ISM bands to ensure that transmissions are free of collisions due to hidden terminal interference.

This dissertation focuses on the design and analysis of novel channel access control protocols that provide superior utilization of the available bandwidth for wireless ad-hoc networks. We study the effect of reversing the collision-avoidance dialogues as a technique that can be controlled by senders, receivers, or both. As our first contribution, we show that receiver-initiated collision-

avoidance protocols can be more efficient than sender-initiated collision avoidance protocols, and prove that a receiver-initiated collision avoidance strategy works correctly. For our second contribution, we study the effect of exerting limited persistence in sender- and receiver-initiated medium access protocols and provide some insightful results. Our third contribution extends the notion of receiver-initiated medium-access to radios with multiple channels. We specify novel control packet handshakes and show that these compare favorably against other multiple-channel protocols. Our last contribution applies also in multi-channel networks in which unicast, multicast and broadcast traffic must be present. We demonstrate how the concept of packet trains can be used to maximize the use of the medium in this case.

This dissertation is organized in chapters that elaborate on each of our contributions in the field of MAC protocols. Chapter 2 presents three different Receiver-Initiated Multiple-Access (RIMA) protocols that reverse the traditional collision-avoidance handshake used in protocols like MACA [55], MACAW [13] and FAMA [35, 36]. In RIMA with simple polling (RIMA-SP), the receiver sends a Ready-to-Receive (RTR) control packet to the sender inviting any potential data packets. If the sender does not have any data packets to send then a Clear-to-Send (CTS) control packet is sent to the receiver enabling the receiver to send its data packet. In RIMA with dual-use polling (RIMA-DP), the receiver also sends an RTR to the sender. If the sender has a data packet to send then it does so after a waiting period; immediately after the receiver has completed the reception of the data packet from the sender, it sends its data packet without any additional control signals. In the case that the sender does not have a data packet to send, it responds with a CTS and the receiver sends its data packet. RIMA with broadcast polling (RIMA-BP) follows the same handshake sequence as RIMA-SP but the polling control signals are broadcasted to everybody. All the RIMA protocols are shown to provide correct collision-avoidance. The throughput of these protocols is analyzed for the case of fully-connected networks, and the results show that RIMA-DP achieves the higher throughput under any offered load conditions than any other sender or receiver initiated protocol. The results of simulation experiments are also presented that show the

performance of RIMA protocols in multi-hop networks.

Chapter 3 analyzes the effect limited persistence in carrier sensing has on the performance of sender as well as receiver-initiated MAC protocols. More specifically, we modify all RIMA protocols as well as FAMA [36] to support different degrees of persistent carrier sensing, and present a comparison with their non-persistent versions. Our analysis shows that the limited persistence version of the protocols outperforms the non-persistent version when the offered load in the network does not exceed a high threshold.

Chapter 4 applies the notion of receiver-initiated handshakes for multi-channel radios. The need of collision-avoidance MAC protocols for single-channel networks to sense the channel as an integral part of the collision-avoidance handshake limits their applicability. Some commercial radios do not provide true carrier sensing, and DSSS radios may capture none or one of multiple overlapping transmissions depending on the proximity and transmission power of the sources. Even if FHSS radios are used, carrier sensing adds to the complexity of the radio, which has already to provide coarse time synchronization at the dwell-time level. To solve that problem, we introduce two Receiver-Initiated Code Hopping (RICH) protocols that do not require carrier sensing or pre-defined code assignment. With RICH with simple polling (RICH-SP), a potential receiver sends an RTR inviting the sender to transmit a data packet. With RICH with dual-use polling (RICH-DP), both the sender and the receiver can engage in data packet exchange in a single very efficient control packet handshake. We prove, with both RICH protocols, that correct collision-avoidance is guaranteed even in the presence of hidden terminals, and that much higher throughput and shorter average delay are obtained than with other sender-initiated protocols tailored to multi-channel networks.

Chapter 5 consists of two parts: in the first part, the sender-initiated equivalent of RICH-SP (which we call Channel Hopping Multiple Access or CHMA), is presented, and it is shown to provide slightly better performance results than prior collision-avoidance MAC protocols designed for multi-channel networks. In the second part, we extend CHMA to support the transmission

of data packet trains. We call this new protocol Channel Hopping Access with Trains (CHAT), which consists of a novel control packet handshake. We show that the performance of the network is improved substantially for the case in which multicast and broadcast traffic must coexist with unicast traffic.

Chapter 6 summarizes our work and gives some suggestions for future research directions.

Chapter 2

Receiver-Initiated Multiple-Access Protocols

In most of the collision-avoidance MAC protocols to date, the sender initiates the handshake with the potential receiver by transmitting a small control packet, usually called RTS (request-to-send). However, only when a data packet is received and decoded successfully at the receiver does it contributes to the good operation of the network. For wireless, ad-hoc networks this is a very important observation, because in such networks it is not unusual that the measured throughput differs by an order of magnitude between a sending and a corresponding receiving node due to the high bit error rate and the hidden terminal effect. The importance of the receiver node in packet networks was our very first motivation towards introducing MAC protocols that are based on receiver-initiated control packet handshakes.

In this chapter, we present MAC protocols in which the receiver creates and sends to the sender an RTR (request-to-receive) control packet. Even though Talucci, Gerla and Fratta have presented a similar receiver-initiated MAC protocol called MACA by invitation (MACA-BI) [92], MACA-BI does not provide *correct collision avoidance*, i.e., prevent data packets addressed to a

given receiver from colliding with any other packets at the receiver. The key contributions of this chapter are recasting collision-avoidance dialogues as a technique that can be controlled by senders, receivers, or both; showing that receiver-initiated collision-avoidance can be even more efficient than sender-initiated collision-avoidance; and presenting a method for proving that a receiver-initiated collision avoidance strategy works correctly.

Section 2.1 introduces fundamental aspects of receiver-initiated collision-avoidance handshake, and Section 2.2 presents a number receiver-initiated collision-avoidance MAC protocols. These protocols require that nodes accomplish carrier sensing, which can be done with baseband radios and today's commercial slow frequency hopping radios, in which complete packets are sent in the same frequency hop. Section 2.3 proves that, in the absence of fading, all these protocols solve the hidden-terminal problem of CSMA, i.e., they eliminate collisions of data packets. Section 2.4 uses an analytical model to study the throughput and average delay of these protocols in fully-connected networks. We use a fully-connected network topology to discern the relative performance advantages of different protocols, because of two reasons: (a) it allows us to use a short analysis that can be applied to several protocols; and (b) our focus on protocols that provide correct collision-avoidance means that the relative performance differences in a fully-connected network are very much the same when networks with hidden terminals are considered. In particular, results presented for FAMA protocols [35, 36] indicate that, in a network with hidden terminals, the performance of a MAC protocol with correct collision-avoidance is almost identical to the performance of the same protocol in a fully-connected network if the vulnerability period of a control packet is made proportional to the length of the entire packet. This is intuitive, if a MAC protocol prevents data packets from colliding with other packets in any type of topology, hidden terminals can degrade the protocol's performance from that obtained in a fully-connected network only to the extent that control packets used to prevent data collisions are subject to additional interference caused by the fact that nodes cannot sense the transmissions of control packets by hidden sources. Our analysis shows that receiver initiated multiple access with dual-use polling (RIMA-DP) is the

most efficient approach among all the sender- and receiver-initiated MAC protocols proposed to date for single-channel networks with asynchronous transmissions. Section 2.5 presents the results of a number of simulation experiments carried out to validate our analytical results, as well as to provide additional insight on the performance of receiver-initiated collision-avoidance in networks with hidden terminals. The results of the experiments illustrate that, as predicted by our analytical model, RIMA-DP performs much better than MAC protocols based on sender-initiated collision-avoidance in networks with hidden terminals.

2.1 Receiver-Initiated Collision Avoidance

Critical design issues in receiver-initiated MAC protocols over a single channel are: (a) whether or not to use carrier sensing, (b) how to persist transmitting packets, (c) how to resolve collisions, and (d) deciding how a receiver should poll its neighbors for data packets.

Carrier sensing has been shown to increase the throughput of sender-initiated collision-avoidance tremendously [35]; furthermore, carrier sensing has also been shown to be necessary to avoid collisions of data packets in sender-initiated collision avoidance over single-channel networks in which transmissions occur in an asynchronous way, i.e., without time slotting [36].

We describe all receiver-initiated schemes assuming carrier sensing and asynchronous transmissions. To simplify the analysis of the protocols, we also assume non-persistent carrier sensing, which has been shown to provide better throughput characteristics than persistent disciplines for CSMA and CSMA/CD [83] at high loads. Furthermore, our treatment of receiver-initiated collision avoidance assumes simple back-off strategies; however, the benefits of using sophisticated back-off strategies or collision resolution algorithms has been analyzed for a number of sender-initiated MAC protocols [13, 40], and it should be clear that the same schemes could be adopted in any of the receiver-initiated approaches we address in this chapter.

In sender-initiated collision-avoidance, a node sends a request-to-send packet (RTS) when-

ever it has data to send and, in protocols using carrier sensing, the channel is free. However, deciding how to send polling packets in receiver-initiated protocols is not as immediate as sending transmission requests in sender-initiated protocols; furthermore, as we show in this chapter, the polling discipline chosen determines to a large extent the performance of the protocol. A polling rate that is too small renders low throughput and long average delays, because each sender with a packet to send is slowed down by the polling rate of the receiver. Conversely, a polling rate that is too high also renders poor performance, because the polling packets are more likely to collide with each other and no source gets polled.

The polling discipline used in a receiver-initiated MAC protocol can be characterized by three different factors:

- Whether or not the polling rate is independent of the data rate at polling nodes.
- Whether the poll is sent to a particular neighbor or to all neighbors.
- Whether the polling packet asks for permission to transmit as well.

In terms of the relationship between the polling rate and the data rate, we can categorize polling disciplines in two major classes: independent polling and data-driven polling.

With independent polling, a node polls its neighbors at a rate that is independent of the data rate of the node or the perceived data transmission rate of its neighbors. In contrast, with data-driven polling, a node attempts to poll its neighbors at a rate that is a function of the data rate with which it receives data to be sent, as well as the rate with which the node hears its neighbors send control and data packets. The specification of the MACA-BI protocol by Talucci et al. [92] assumes this type of polling. Throughout the rest of the chapter, we assume data-driven polling, because it is very difficult in a real network to determine a good independent polling rate by the receivers.

In practice, to account for data rate differences at nodes and to eliminate the possibility of a data-driven polling discipline never allowing a node to receive data, a protocol based on data-

driven polling should send a poll based on its local data to be sent or after a polling timeout elapses without the node having any packet to send to any neighbor.

The intended audience of a polling packet can be a single neighbor, a subset of neighbors, or all the neighbors of a node. A large audience for a poll packet introduces the possibility of contention of the responses to the poll, and either the collisions of responses need to be resolved, or a schedule must be provided to the poll audience instructing the neighbors when to respond to a poll.

The intent of a polling packet can be simply to ask one or more neighbors if they have data to send to the polling node, or it can both ask for data and permission to transmit in the absence of data from the polled neighbors. Intuitively, the latter approach should have better channel utilization, because data will be sent after every successful handshake, and more data per successful handshake are sent as traffic load increases even if the polled node does not have data for the polling node. We also note that a polling packet asking for data from a neighbor could allow the polled node to send data to *any* destination, not just to the polling node; however, this strategy would not work efficiently in multi-hop networks, because there is no guarantee that the recipient of a data packet who did not ask for it will receive the transmission without interference.

It is clear that polls that specify transmission schedules can address the three key functions of a polling discipline that we have just discussed. In this chapter, however, we concentrate on single-node polling and broadcast polling only. Receiver-initiated protocols based on schedules is an area of future research.

2.2 Receiver-Initiated Protocols

This section introduces new MAC protocols based on receiver initiated collision-avoidance and relates them to the taxonomy of polling disciplines presented in Section 2.1. To our knowledge, these protocols are the first based on receiver-initiated collision avoidance that eliminate the

collisions of data packets with any other control or data packets in the presence of hidden terminals.

2.2.1 Protocols with Simple Polling

MACA-BI

The original MACA-BI [92] protocol uses a ready-to-receive packet (RTR) to invite a node to send a data packet. A node is allowed to send a data packet only if it has previously received an RTR, whereas a node that receives an RTR that is destined to a different node has to back off long enough for a packet to be sent in the clear.

According to the description of MACA-BI, a polled node can send a data packet intended for the polling node or any other neighbor. In a fully-connected network, whether the data packet is sent to the polling node or not is not important, because all the nodes must back off after receiving an RTR in the clear. However, this is not the case in a network with hidden terminals.

By means of two simple examples, we can show that MACA-BI does not prevent data packets sent to a given receiver from colliding with other data packets sent concurrently in the neighborhood of the receiver. The first example illustrates the fact that, in order to avoid the transmission of data packets that the intended receiver cannot hear because of other colliding data packets, a polled node should send data packets only to the polling node. The second example illustrates the possibility that collisions of data packets at a receiver may occur because the receiver sent an RTR at approximately the same time when data meant for another receiver starts arriving.

In Fig. 2.1, nodes a and d send RTRs to nodes b and e at time t_0, respectively. This prompts the polled nodes to send data packets at time t_1; the problem in this example occurs when at least one of the polled nodes sends a data packet addressed to c, which cannot hear either packet.

In the example shown in Fig. 2.2, node a sends an RTR to b at time t_0. This RTR makes node b start sending data to node a at time t_1 which in order to provide good throughput must

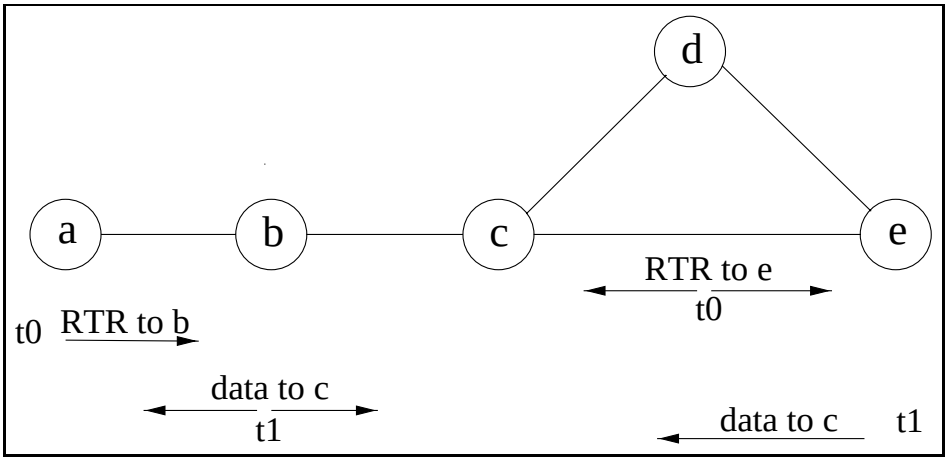


Figure 2.1: Data packets colliding in MACA-BI when packet is not sent to polling node

be larger than γ seconds, where γ is the length of an RTR. At time t_2 node c starts sending an RTR to node d. Because of carrier sensing, t_2 must be within τ seconds (maximum propagation delay) of t_1. In this example, after receiving node c's RTR, node d replies with data that must start arriving at node c at time t_3. Because the maximum propagation delay is τ, it must be true that $t_3 \leq t_2 + \gamma + 2\tau \leq t_1 + \gamma + 3\tau$. Hence, if data packets last longer than $\gamma + 3\tau$ seconds, the data packets from b and d collide at node c. In practice, data packets must be much longer than RTRs to provide good throughput, and it thus follows that MACA-BI cannot prevent all data packets from experiencing collisions.

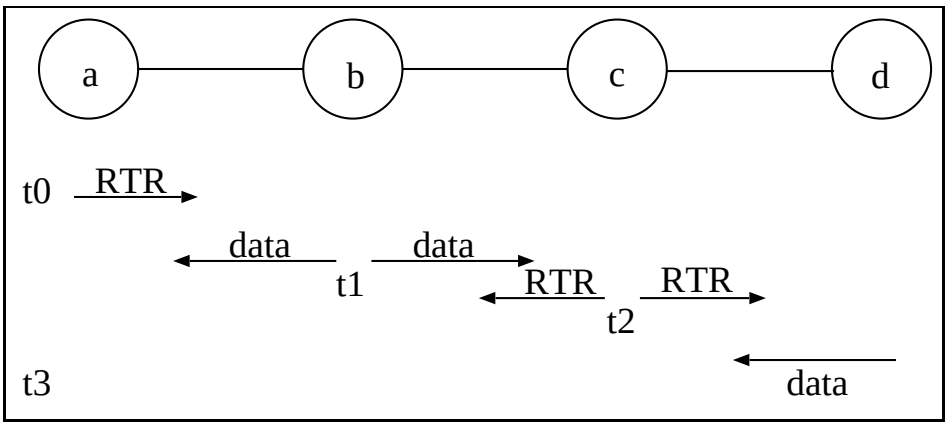


Figure 2.2: Data packets colliding in MACA-BI due to RTR not being heard

RIMA-SP

The above problems in MACA-BI went unnoticed in the specification by Talucci et al. [92]. To make the RTR-data handshake in MACA-BI collision free, the following two minor modifications are required:

- The polled node should transmit data packets only if they are addressed to the polling node.
- A new control signal is also required, which we call No-Transmission-Request (NTR), and an additional collision-avoidance waiting period of ξ seconds is required at a polled node prior to answering an RTR.

During the collision-avoidance waiting period, if any channel activity is heard, the receiver (polling node) that originated an RTR sends an NTR telling the polled node not to send any data. Otherwise, if nothing happens during the waiting period, the polled sender transmits its data, if it has any to send to the polling node.

We call the protocol resulting from modifying MACA-BI with the above two rules RIMA-SP (receiver initiated multiple access with simple polling). Fig. 2.3 illustrates the operation of RIMA-SP. The complete proof that RIMA-SP provides correct collision avoidance when $\xi = \tau$ is given in Section 2.3.

Figure 2.4 shows the specification for RIMA-SP in pseudo-code. In RIMA-SP, every node initializes itself in the START state, in which the node waits twice the maximum channel propagation delay, plus the hardware transmit-to-receive transition time (ϵ), before sending anything over the channel. This enables the node to find out if there are any ongoing transmissions. After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time that is sufficient for the node to start receiving packets in transit.

If a node x is in the PASSIVE state and senses carrier, it transitions to the REMOTE state to defer to ongoing transmissions. A node in REMOTE state must allow enough time for a

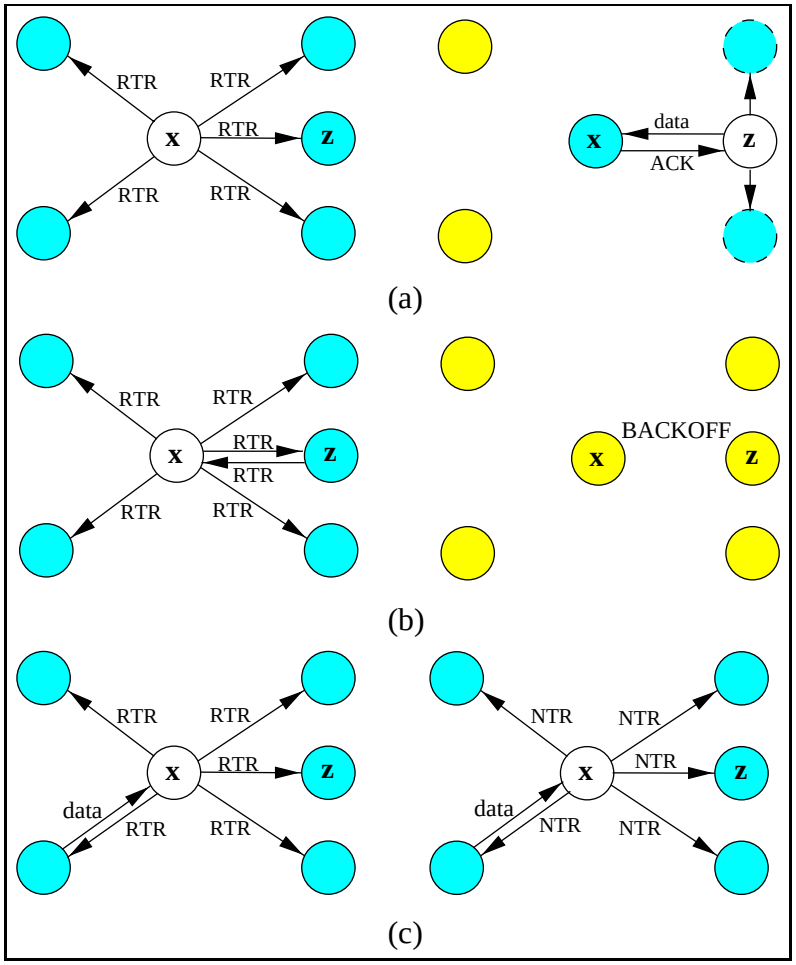


Figure 2.3: RIMA-SP illustrated

complete successful handshake to take place, before attempting to transition from remote state.

Any node in PASSIVE state that detects noise in the channel must transition to the BACKOFF state. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTR state. In the RTR state, node x uses non-persistent carrier sensing to transmit an RTR. If node x detects carrier when it attempts to send the RTR, it transitions to the BACKOFF state, which makes the node back off immediately for a sufficient amount of time to allow a complete handshake between a sender-receiver pair to occur; otherwise, x sends its RTR.

If node z receives the RTR correctly and has data for x, it waits for ξ seconds. If during the waiting period there is no activity in the channel, node z transitions to the XMIT state, where

```
Variable Definitions
    Timer = A global timer
    CD = Carrier Detected
    γ = Time to transmit an RTR or NTR packet
    δ = Time to transmit a data packet
    T_WT = The waiting time ξ
    T_PROP = Maximum propagation delay across the channel
    T_PROC = Processing time for carrier detection
    T_TXRX = Hardware transmit to receive transition time

Procedure START()
Begin
    Timer ← (2 × T_PROP) + T_TXRX;
    While (CD and Timer not expired) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
    Else call PASSIVE();
End

Procedure XMIT()
Begin
    Wait T_TXRX;
    Transmit Data Packet;
    Timer ← T_PROP + T_TXRX;
    While (Timer not expired) wait;
    If (Local Packet) Then
     call BACKOFF();
    Else call PASSIVE();
End

Procedure RTR(T_x)
Begin
    Transmit an RTR Packet;
    Timer ← T_x;
    While (CD and Timer not expired) wait;
    If (CD) Then Begin
     Receive packet;
     Do Case of (received packet type)
     Begin
      NTR:
      call BACKOFF();
      DATA:
      If (Destination ID = Local ID) Then
       pass Data Packet to upper layer;
       call PASSIVE();
      ERROR:
      call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
     End
    End
End

Procedure PASSIVE()
Begin
    While (CD and No local packet) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), FALSE);
    If (Local Packet) Then
     call RTR(2 × T_PROP + T_PROC + T_TXRX);
End

Procedure BACKOFF()
Begin
    Timer ← RANDOM(1, 10 ×γ);
    While (CD and Timer not expired) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), FALSE);
    Else call RTR(2 × T_PROP + T_PROC + T_TXRX + T_WT);
End

Procedure REMOTE(T_x, flag)
Begin
    Timer ← T_x;
    While (CD and Timer not expired) wait;
    If (Timer Expired) Then Begin
     If (Local Packet) Then
      call RTR(2 × T_PROP + T_PROC + T_TXRX + T_WT);
     Else call PASSIVE();
    End
    Else Begin
     While (CD) wait;
     Receive Packet;
     Do Case of (received packet type)
     Begin
      RTR:
      if (flag = TRUE)
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      If (Destination ID = Local ID) Then
       wait T_TXRX;
       Transmit Data packet;
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      Else
      call REMOTE((γ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
      NTR:
      call BACKOFF();
      DATA:
      If (Destination ID = Local ID) Then
       pass packet to upper layer;
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      ERROR:
       call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX));
     End
    End
End
```

Figure 2.4: RIMA-SP Specification

it transmits a data packet to x, and node x sends an acknowledgment (ACK) immediately after receiving the data packet (Fig. 2.3(a)); otherwise, node z assumes that there was a collision and transitions to the BACKOFF state to allow floor acquisition by some other node. After sending its RTR, node x senses the channel. If it detects carrier immediately after sending its RTR, node x assumes that a collision or a successful data transfer to a hidden node is taking place. Accordingly, it sends a No transmission Request (NTR) to z to stop z from sending data that would only collide at x (Fig. 2.3(b)).

When multiple RTRs are transmitted within a one-way propagation delay, a collision takes place and the nodes involved have to transition to the BACKOFF state and try again at a later time chosen at random, as shown in Fig. 2.3(b).

Node x determines that its RTR was not received correctly by z after a time period equal to the maximum round-trip delay to its neighbors plus turn-around times and processing delays at the nodes, plus the waiting period ξ. After sending its RTR, node x listens to the channel for any ongoing transmission. Because of non zero propagation delays, if node x detects carrier immediately after transmitting its RTR, it can conclude that it corresponds to a node other than z, which would take a longer time to respond due to its need to delay its data to x to account for turn-around times. [1]

The lengths of RTRs and NTRs are the same. The same argument used in [35] to show that the length of an RTS must be longer than the maximum propagation delay between two neighbors to ensure correct collision-avoidance can be used to show that RTRs and NTRs must last longer than a maximum propagation delay. In ad-hoc networks in ISM bands, propagation delays are much smaller compared with any packet that needs to be transmitted.

To reduce the probability that the same nodes compete repeatedly for the same receiver at the time of the next RTR, the RTR specifies a back-off-period unit for contention. The nodes that must enter the BACKOFF state compute a random time that is a multiple of the back-off-period unit advertised in the RTR. The simplest case consists of computing a random number of back-off-period units using a uniformly distributed random variable from 1 to d, where d is the maximum number of neighbors for a receiver. The simplest back-off-period unit is the time it takes to send a small data packet successfully.

2.2.2 Protocols with Dual-Use Polling

The collision-avoidance strategy described for RIMA-SP can be improved by increasing the probability that data will follow a successful RTR, without violating the rule that data packets should be transmitted only if they are addressed to the polling nodes. A simple way to achieve this with data-driven polling is to make an RTR entry both a request for data from the polled

[1] Our analysis assumes 0 turn-around times and 0 processing delays for simplicity.

node, and a transmission request for the polling node to send data. The RIMA-DP (receiver-initiated multiple access with dual-purpose polling) protocol does exactly this. Fig. 2.5 illustrates the modified collision avoidance handshake to permit the polling node to either receive or send data without collisions.

As Fig. 2.5(a) illustrates, a key benefit of the dual-use polling in RIMA-DP is that both polling and polled nodes can send data in a round of collision-avoidance. This is possible because the RTR makes all the neighbors of the polling node back-off, and the data from the polled node make all its neighbors back-off, which can then be used by the polling node to send its data.

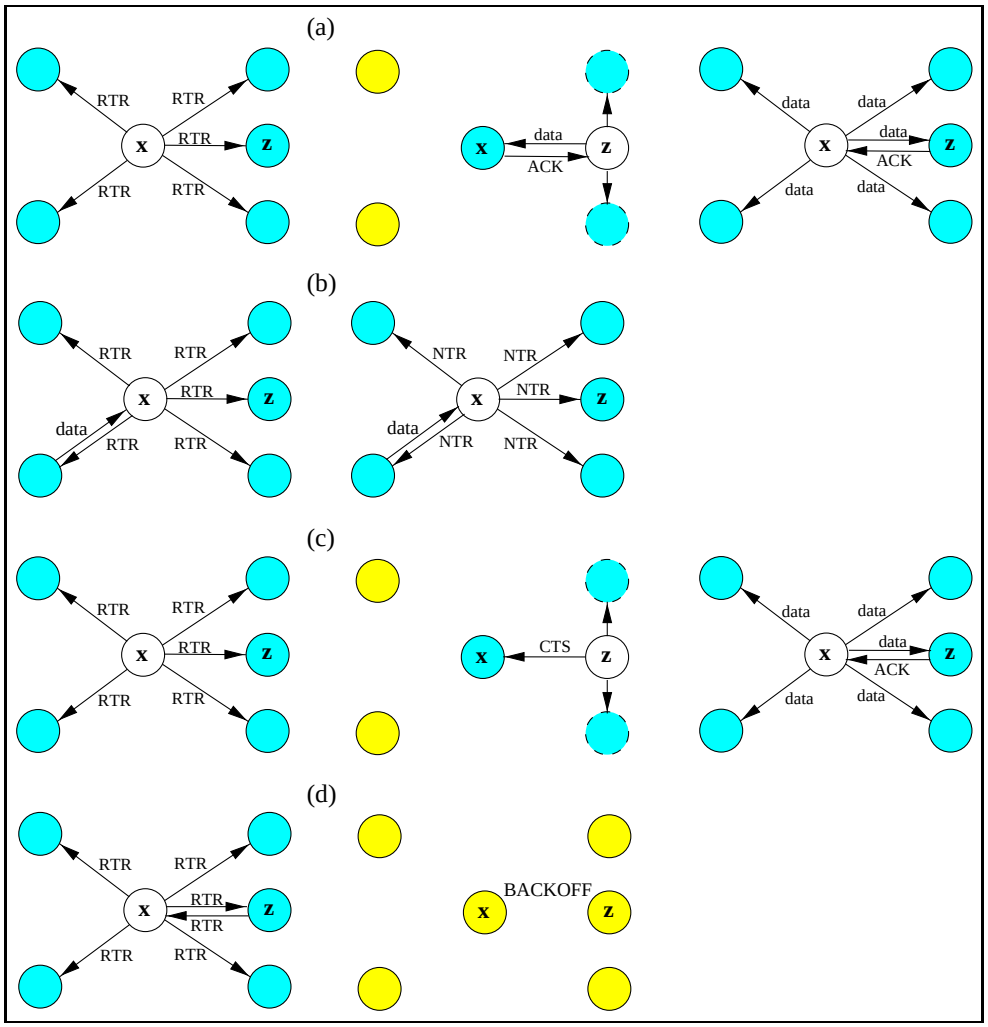


Figure 2.5: RIMA-DP illustrated

RIMA-DP gives transmission priority to the polling nodes. When a node z is polled by node x and has data for node x, z waits ξ seconds before sending a data packet. In contrast, if the polled node does not have data for x, it immediately sends a CTS (Clear-To-Send packet) to x. This permits a polling node x exposed to a neighbor sending data to hear part of that neighbor's data packet after sending its RTR; in such a case, node x can send an NTR to the polled node to cancel its RTR. Section 2.3 shows that this prevents collisions of data packets, provided that z waits for $\xi > \gamma + 7\tau$ seconds before sending any data after being polled and the length of a CTS is 2τ seconds longer than the length of an RTS. As in RIMA-SP, the lengths of RTRs and RTSs are the same.

Figure 2.6 shows the specification for RIMA-DP in pseudo-code. As with RIMA-SP, every node starts in the START state and transitions to to the PASSIVE state when it is initialized. If a node x is in the PASSIVE state and senses carrier, it transitions to the REMOTE state to defer to ongoing transmissions. A node in REMOTE state must allow enough time for a complete successful handshake to take place, before attempting to transition from remote state.

Any node in PASSIVE state that detects noise in the channel must transition to the BACKOFF state where it must allow sufficient time for complete successful handshakes to occur. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTR state. In the RTR state, node x behaves as in RIMA-SP.

If node z receives the RTR correctly and has data for x, it waits for ξ seconds before sending a data packet to x. If during the waiting period there is no activity in the channel, node z transitions to the XMIT state, where it transmits a data packet to x. Otherwise, z assumes a collision or data transfer to a hidden node and goes to the BACKOFF state. If z has no data for x, it sends a CTS to x immediately.

If node x detects carrier immediately after sending an RTR, it defers its transmission attempt and sends an NTR to the node it polled. The CTS length, which is τ seconds longer than an RTR, forces polling nodes that send RTRs at about the same time when a polled node sends

```
Variable Definitions
    Timer = A global timer
    CD = Carrier Detected
    γ = Time to transmit an RTR or CTS packet
    δ = Time to transmit an RTR or CTS packet
    T_PROP = Maximum propagation delay across the channel
    T_PROC = Processing time for carrier detection
    T_TXRX = Hardware transmit to receive transition time

Procedure START()
Begin
    Timer ← (2 × T_PROP) + T_TXRX;
    While (¬CD and Timer not expired) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
    Else call PASSIVE();
End

Procedure CTS(T_x)
Begin
    Transmit an CTS Packet;
    Timer ← T_x;
    While (¬CD and Timer not expired) wait;
    If (Timer Expired) Then call BACKOFF();
    While (CD) wait;
    Receive packet;
    Do Case of (received packet type)
    Begin
     DATA:
     If (Destination ID = Local ID) Then
      pass packet to upper layer;
      call PASSIVE();
     ERROR:
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
    End
End

Procedure RTR(T_x)
Begin
    Transmit an RTR Packet;
    Timer ← T_x;
    While (¬CD and Timer not expired) wait;
    If (¬CD and Timer Expired)
     call CTS(2 × T_PROP + T_PROC + T_TXRX);
    Else Begin
     Receive packet;
     Do Case of (received packet type)
     Begin
      CTS:
      call XMIT();
      DATA:
      If (Destination ID = Local ID) Then
       pass Data Packet to upper layer;
       call PASSIVE();
      ERROR:
      call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
     End
    End
End

Procedure PASSIVE()
Begin
    While (¬CD and No local packet) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), FALSE);
    If (Local Packet) Then
     call RTR(2 × T_PROP + T_PROC + T_TXRX);
End

Procedure XMIT()
Begin
    Wait T_TXRX;
    Transmit Data Packet;
    Timer ← T_PROP + T_TXRX;
    While (Timer not expired) wait;
    If (Local Packet) Then
     call BACKOFF();
    Else call PASSIVE();
End

Procedure BACKOFF()
Begin
    Timer ← RANDOM(1, 10 × γ);
    While (¬CD and Timer not expired) wait;
    If (CD) Then
     call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), FALSE);
    Else call RTR(2 × T_PROP + T_PROC + T_TXRX);
End

Procedure REMOTE(T_x, flag)
Begin
    Timer ← T_x;
    While (¬CD and Timer not expired) wait;
    If (Timer Expired) Then Begin
     If (Local Packet) Then
      call RTR(2 × T_PROP + T_PROC + T_TXRX);
     Else call PASSIVE();
    End
    Else Begin
     While (CD) wait;
     Receive Packet;
     Do Case of (received packet type)
     Begin
      RTR:
      if (flag = TRUE)
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      If (Destination ID = Local ID) Then
       wait T_TXRX;
       Transmit Data packet;
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      Else
      call REMOTE((γ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
      CTS:
      call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX), TRUE);
      DATA:
      If (Destination ID = Local ID) Then
       pass packet to upper layer;
       call REMOTE((2 × T_PROP + T_PROC + T_TXRX), TRUE);
      ERROR:
       call REMOTE((δ + 2 × T_PROP + T_PROC + T_TXRX));
     End
    End
End
```

Figure 2.6: RIMA-DP Specification

a CTS to detect carrier from the CTS and stop their attempt to send or receive data. Any node other than x receiving the CTS for x transitions to the BACKOFF state. When node x receives the CTS from z, it transitions to the XMIT state and transmits a data packet to z.

2.2.3 Protocols with Broadcast Polling

Contrary to the prior two approaches, an RTR can be sent to multiple neighbors. We describe a modification of RIMA-SP based on this variant, which we call RIMA-BP (Broadcast Polling).

Fig. 2.7 illustrates the receiver-initiated handshake of RIMA-BP. As it is shown in the figure, the key difference with RIMA-SP is the use of an RTS prior to the transmission of a data packet. A node broadcasts an RTR only when there is a local data packet (data-driven polling). Only after a node has received an invitation, is it allowed to send any data. Because a poll broadcast to all the neighbors of a node can cause multiple nodes to attempt sending data to the polling node, an additional control packet is needed to ensure that transmissions that collide last a short period and do not carry user data. Accordingly, a polled node sends a short RTS (Ready-To-Send packet) before sending data. Furthermore, after sending its RTS, the polled node must wait for ξ seconds to allow the polling node to send an NTR when collisions of RTSs occur at the polling node. It can be shown that RIMA-BP provides correct collision avoidance if $\xi = 4\tau$.

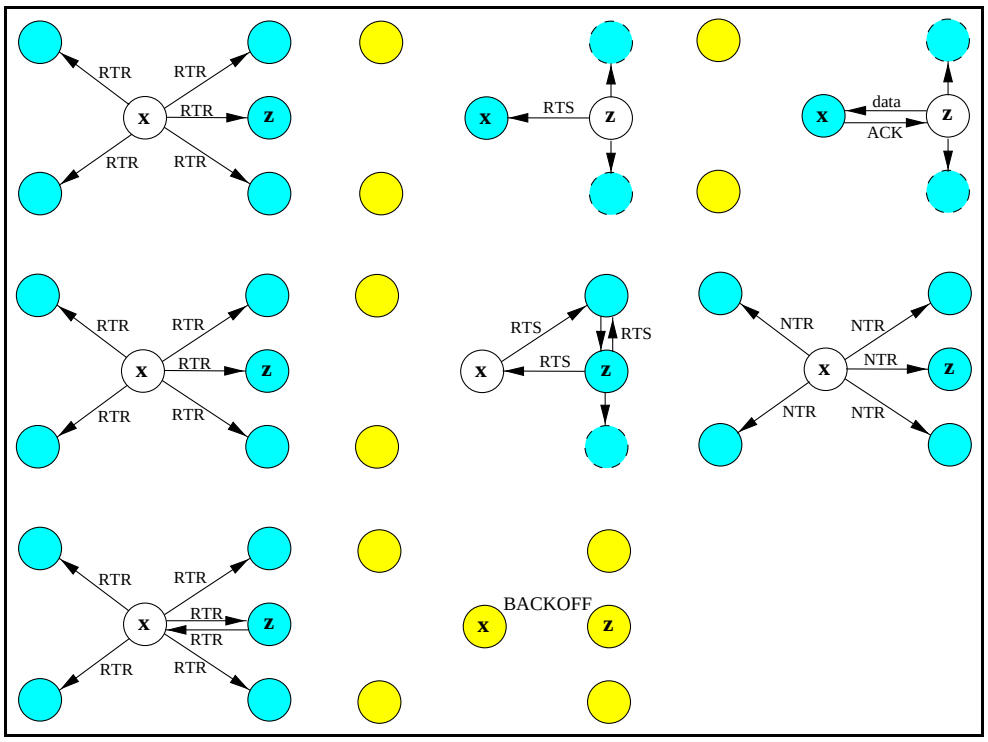


Figure 2.7: RIMA-BP illustrated

2.3 Correct Collision Avoidance in RIMA protocols

Theorems 1 and 2 below show that RIMA-SP and RIMA-DP ensure that there are no collisions between data packets and any other transmissions. A similar proof to that of Theorem 1 can be used to show that RIMA-BP provides correct collision-avoidance if $\xi = 4\tau$. The following assumptions are made to demonstrate correct collision-avoidance in RIMA protocols [36]:

A0) A node transmits an RTR that does not collide with any other transmissions with a non-zero probability.

A1) The maximum end-to-end propagation time in the channel is $\tau < \infty$.

A2) A packet sent over the channel that does not collide with other transmissions is delivered error free with a non-zero probability.

A3) All nodes execute a RIMA protocol correctly.

A4) The transmission time of an RTR and a CTS is γ, the transmission time of a data packet is δ, and the hardware transmit-to-receive transition time is zero; furthermore, $2\tau < \gamma \leq \delta < \infty$.

A5) There is no capture or fading in the channel.

A6) Any overlap of packet transmissions at a particular receiver, causes that receiver to not understand any of the packets.

The approach used to show that a collision-avoidance protocol works correctly; i.e., that it prevents data packets from colliding with any type of packets, consists of showing that, once a data packet is sent by a node, the intended receiver obtains the packet without interference. In any receiver-initiated collision-avoidance scheme, we have polling nodes and polled nodes, and we must show that any interference at the polled node prevents it from sending data, while any detected interference at a polling node that has sent an RTR forces the node to jam the polled node to prevent data from arriving and collide at the polling node.

Because interference must be detected by polling and polled nodes, the receiver-initiated collision-avoidance protocols we are describing require carrier sensing. The ability to detect carrier is applicable only to baseband radios or slow frequency hopping radios, and periods of fading disrupt any type of collision-avoidance dialogue, i.e., data packets may experience collisions in the presence of fading.

Assuming zero processing and turn-around delays is done for convenience; however, the same type of proofs, with adjusted parameters, apply for non-zero hardware delays.

Theorem 1 *RIMA-SP provides correct collision-avoidance in the presence of hidden terminals, provided that* $\xi = \tau$.

Proof: Consider a polling node A and a polled node X and assume that A sends an RTR at time t_0. If X does not receive the RTR correctly due to interference from any neighbor hidden from A, it does not send any data. Else, X waits $\xi = \tau$ seconds after receiving A's RTR before sending its data to A. Because propagation delays are positive, the earliest time when X can start sending data to A is $t_1 > t_0 + \gamma + \tau$. On the other hand, if A detects interference immediately after sending its RTR, i.e., at time $t_0 + \gamma$, it starts sending an NTR to X, and this NTR must start arriving at X no later than $t_2 \leq t_0 + \gamma + \tau$. Because $t_1 > t_2$, it follows that X cannot send data to A that can collide with any other transmission arriving at A. *Q.E.D.*

Theorem 2 *RIMA-DP provides correct collision-avoidance in the presence of hidden terminals, provided that* $\xi > \gamma + 7\tau$ *and a CTS lasts* 2τ *seconds longer than an RTR.*

Proof: Consider a polling node A and a polled node X and assume that A sends an RTR to X at time t_0. If A is exposed to a polled node Y sending data or a CTS, A must have started its RTR within τ seconds of Y's start of transmission; for otherwise A would have detected carrier caused by Y and would not have sent its RTR. Accordingly, because data packets and CTSs are at least $\gamma + 2\tau$ in length, A must detect carrier from Y's transmission immediately after sending its RTR, which forces node A to send an NTR at time $t_0 + \gamma$. Therefore, regardless of what happens at the

polled node X, the polling node A must send an NTR immediately following its RTR and back-off, and cannot send any data if there is any exposed polled node sending or requesting data.

Assume that A is not exposed to a polled node sending or requesting data, but is exposed to a polling node B. Let A poll node X and B poll node Y.

For both A and B to send their RTRs they must do so within τ seconds of each other, for otherwise one of the two would detect carrier and back-off. For X to send any packet to A (data or CTS), A's RTR must be received collision free at X. X can receive A's RTR successfully no earlier than $t_1 > t_0 + \gamma$, because propagation delays are positive. If X has no data for A, it sends a CTS immediately to X. This CTS can arrive in its entirety at A no earlier than $t_2 > t_0 + 2\gamma + 2\tau$, because a CTS lasts $\gamma + 2\tau$ and propagation delays are positive. The same is the case for the node Y polled by B. Therefore, if both X and Y send data, Y must be hidden from A and X must be hidden from B, and no data packets collide.

There are three cases to consider now. In one case one polled node sends a CTS and the other sends data, in another case both polled nodes send data, and in the last case each polled node sends a CTS.

Without loss of generality, assume that Y sends a CTS and X is ready to send data. The earliest time when X can start sending data is $t_2 > t_0 + \gamma + \xi$, because propagation delays are positive. On the other hand, the latest time when A must start receiving data sent by B, after B receives the CTS from Y, is $t_3 \leq (t_0 + \tau) + (2\gamma + 3\tau) + \tau$. The first $t_0 + \tau$ stems from the fact that B can send its RTR up to τ seconds after A starts sending its RTR. The second term $(2\gamma + 3\tau)$ corresponds to B's RTR and Y's CTS plus the corresponding maximum propagation delays, and the last term τ is the maximum propagation delay of data from B to A. Accordingly, A must detect carrier and starts sending its NTR at time t_3, and X must detect carrier from A's NTR at time $t_4 \leq t_3 + \tau \leq t_0 + 2\gamma + 7\tau$. Because $t_2 > t_0 + \gamma + \xi$ and $\xi > \gamma + 7\tau$, it follows that X cannot send its data to A and no collision occurs.

If both X and Y send data or CTSs after being polled no collision occurs with data packets, because we have shown that X must be hidden from B and Y must be hidden from A. Therefore, RIMA-DP provides collision-avoidance correctly. *Q.E.D.*

2.4 Performance Analysis

The objective of our analysis is to contrast the various polling polices introduced for RIMA protocols, and to compare them against sender-initiated collision-avoidance protocols, namely, MACA [55] and FAMA-NCS [36]. The choice of protocols was made because MACA is the simplest sender-initiated collision-avoidance protocol and FAMA-NCS is the best performing MAC protocol based on sender-initiated collision-avoidance reported to date.

2.4.1 Approximate Throughput

We analyze the throughput of receiver initiated protocols using the model first introduced by Kleinrock and Tobagi [57] for CSMA protocols and used subsequently to analyze MACA [55], FAMA [36], MACA-BI [92] and several other collision-avoidance protocols. According to this model, the following assumptions are made:

1. There are N nodes in the fully-connected network.

2. A single unslotted channel is used for all packets, and the channel introduces no errors.

3. All nodes can detect collisions perfectly.

4. The size for a data packet is δ seconds and the size of an RTR, an ACK, and an RTS is γ seconds, the size of a CTS in RIMA protocols is γ seconds, and the size of a CTS for FAMA-NCS is $\gamma + 2\tau$ [36].

5. The turn-around time ϵ is considered to be part of the duration of control and data packets.

6. The propagation delay of the channel between any two nodes is τ seconds.

For simplicity, our previous analysis of receiver-initiated MAC protocols in [42] did not take into account the effect of ACKs. The present analysis included the overhead incurred by the ACKs needed to inform the sender of the correct reception of a data packet.

To provide a fair comparison between sender-initiated and receiver-initiated protocols while preserving the tractability of the analytical model, we assume that a polled node receiving an RTR always has a data packet to send, but the probability that that packet is addressed to the polling node is $\frac{1}{N}$. Furthermore, we assume that each node sends its RTR according to a Poisson distribution with a mean rate of $\frac{\lambda}{N}$, and that (when applicable) the polling node chooses the recipient of the RTR with equal probability. This model is slightly unfair to RIMA protocols compared to MACA-BI, because the likelihood that a polled node can transmit remains constant even as the traffic load increases. To account for this, we also discuss a heavy-traffic approximation of our results, in which a polled node always has data to send to any polling node.

The corresponding assumptions for sender-initiated protocols are that a node always has packets to send, but schedules the transmission of RTSs according to a Poisson distribution with a mean rate of $\frac{\lambda}{N}$, and chooses to which neighbor to send the RTS with probability $\frac{1}{N}$. These assumptions preserve the validity of prior analytical results for FAMA and MACA [36].

Because the arrival of RTSs or RTRs to the channel is Poisson, the average channel utilization is:

$$S = \frac{\overline{U}}{\overline{B} + \overline{I}} \tag{2.1}$$

where $\overline{B}$ is the expected duration of a busy period, defined to be a period of time during which the channel is being utilized; $\overline{I}$ is the expected duration of an idle period, defined as the time interval between two consecutive busy periods; and $\overline{U}$ is the time during a busy period that the channel is used for transmitting user data successfully.

MACA-BI

The following theorem provides the throughput of MACA-BI in a fully-connected network. In a network with hidden terminals, MACA-BI's performance would degrade substantially according to two factors: (a) the probability of bad busy periods is increased by the probability that either a node sends an RTR within τ seconds of any neighbor sending a data packet, or a node receives a data packet addressed to it while it also receives other data packets; and (b) the length of a bad busy period is proportional to the length of a data packet, rather than the length of an RTR as in RIMA-SP.

Theorem 3 *The throughput for MACA-BI in a fully-connected network is given by*

$$S = \frac{\delta}{\delta + \gamma + 2\tau + \frac{1}{\lambda} + (\gamma + 2\tau)e^{\lambda\tau}} \tag{2.2}$$

Proof: Because a successfully polled node can send data to any neighbor, the probability that a successful transmission occurs equals the probability that an RTR is transmitted successfully, that is,

$$P_S = e^{-\lambda\tau} \tag{2.3}$$

The duration of every successful busy period is $2\gamma + \delta + +3\tau$, and the first and the last packet of the busy period is the successful packet of the period.

Because the network is fully connected, a failed busy period can occur only when there is a collision between RTRs, which occurs with probability $1 - P_S$.

The average duration of any busy period always consists of at least an RTR and the associated propagation delay (i.e., $\gamma + \tau$) plus the average time between the first and the last RTR of the busy period, which we denote by $\overline{Y}$ and is the same as in CSMA [95], i.e., $\overline{Y} = \tau - \frac{1-e^{-\lambda\tau}}{\lambda}$. If the busy period is successful, a data packet is also sent; therefore, the length of the average busy period in MACA-BI is given by

$$\begin{aligned}\overline{B} &= \gamma + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}(\delta + \gamma + 2\tau) \\ &= \gamma + 2\tau - \frac{1}{\lambda} + e^{-\lambda\tau}\left[\tau + \frac{1}{\lambda} + \delta\right] \end{aligned} \tag{2.4}$$

The length of the average idle period is $\frac{1}{\lambda}$, and the length of the average utilization period is

$$\overline{U} = \delta P_S = \delta e^{-\lambda\tau} \tag{2.5}$$

The theorem follows by substituting the values of $\overline{U}$, $\overline{B}$ and $\overline{I}$ in Eq. (2.1). *Q.E.D.*

The throughput of MACA-BI has been reported before by Talucci et al. [92]. However, that prior derivation did not take into account that, in computing the length of an average busy period, the first and the last RTR of a busy period is the same, and that there is a non-zero probability that a polled node has no packets to send to any node if RTRs are sent when packets arrive and arrivals are Poisson. Nevertheless, the results in Theorem 2.2 and [92] are practically the same for the model we have assumed in our analysis, in which a polled node always has something to send, even if it is not the polled node. We should also point out that our own prior analysis of MACA-BI [42] incorrectly assumed that a polled node could only transmit packets to the polling node, which is unfair to MACA-BI in a fully-connected network.

RIMA-SP

The following theorem provides the throughput of RIMA-SP in a fully-connected network. In a network with hidden terminals, the performance of RIMA-SP would degrade by the increase of the vulnerability period of RTRS from one propagation delay to essentially twice the length of the RTR, and by the need for the polling nodes to send NTRs after detecting interference.

Theorem 4 *The throughput for RIMA-SP in a fully-connected network is given by*

$$S = \frac{\delta\frac{1}{N}}{\frac{\delta+\gamma+\tau}{N} + \xi + \tau + \frac{1}{\lambda} + (\gamma + 2\tau)e^{\lambda\tau}} \tag{2.6}$$

where $\xi = \tau$.

Proof: Because of our independence assumptions, the probability that a successful transmission occurs equals the probability that an RTR is transmitted successfully, times the probability that the polled node has a data packet for the polling node at the head of its queue; that is,

$$P_S = e^{-\lambda\tau}\left(\frac{1}{N}\right) \tag{2.7}$$

The duration of every successful busy period is $2\gamma + \delta + \xi + 3\tau$. Notice that, in this case, the first and the last packet of the busy period is the successful packet of the period.

In RIMA-SP, a failed busy period can occur when there is a collision between RTRs, and when an RTR is sent in the clear but the polled sender does not have a data packet to send to the polling node. The first case occurs with probability:

$$P_{F1} = 1 - e^{-\lambda\tau} \tag{2.8}$$

The probability of the second case of a failed busy-period scenario occurring is given by

$$P_{F2} = e^{-\lambda\tau}\left(1 - \frac{1}{N}\right) \tag{2.9}$$

As it was the case for MACA-BI, any busy period always consists of at least an RTR and the associated propagation delay (i.e., $\gamma + \tau$) plus the average time between the first and the last RTR of the busy period, denoted by $\overline{Y}$. When the busy period fails due to the collision of two or more RTRs, there are no additional time components in the busy period. When the busy period is successful, $Y = 0$, of course, and additional time due to the collision-avoidance waiting time and a data packet is incurred, i.e., $\delta + \xi$. Finally, if the busy period fails because the polled node does not have a packet for the polling node, then an additional propagation delay and a collision-avoidance waiting time are incurred. Accordingly, the length of the average busy period is given by

$$\begin{aligned}\overline{B} &= \gamma + 2\tau - \frac{1-e^{-\lambda\tau}}{\lambda} + \frac{1}{N}e^{-\lambda\tau}(\delta+\gamma+\xi+2\tau) + e^{-\lambda\tau}(1-\frac{1}{N})(\xi+\tau) \\ &= \gamma + 2\tau - \frac{1}{\lambda} + e^{-\lambda\tau}\left[\xi+\tau+\frac{1}{\lambda}+\frac{\delta+\gamma+\tau}{N}\right] \end{aligned} \tag{2.10}$$

The length of the average idle period is $\frac{1}{\lambda}$, and the length of the average utilization period is

$$\overline{U} = \delta P_S = \delta e^{-\lambda\tau}\left(\frac{1}{N}\right) \tag{2.11}$$

The theorem follows by substituting the values of $\overline{U}$, $\overline{B}$ and $\overline{I}$ in Eq. (2.1). *Q.E.D.*

RIMA-DP

The following theorem provides the throughput for RIMA-DP in a fully-connected network. The performance of RIMA-DP in a network with hidden terminals would degrade by the increase in the vulnerability period of RTRs, which is one propagation delay in a fully-connected network and is twice an RTR in a network with hidden terminals.

Theorem 5 *The throughput of RIMA-DP for a fully connected network is given by*

$$S = \frac{\delta(1+\frac{1}{N})}{2\gamma+\delta+3\tau+\frac{1}{\lambda}+\frac{1}{N}(\delta+\xi)+(\gamma+2\tau)e^{\lambda\tau}} \tag{2.12}$$

where $\xi > \gamma + 7\tau$.

Proof: Because the network is fully connected, whenever an RTR is transmitted successfully a packet always follows, either from the node sending the poll or the polled node. Therefore, the probability of success, P_S, is equal to the probability with which an RTR is transmitted successfully. Because all nodes are connected, an RTR from node w is successful if there are no other RTRs transmitted within τ seconds from the start of the RTR. After this vulnerability period of τ seconds, all the nodes detect the carrier signal and act appropriately. Accordingly,

$$P_S = e^{-\lambda\tau} \tag{2.13}$$

The probability, P_{S1}, with which the polled node has data to send to the polling node is equal to the probability that an RTR is sent in the clear, times the probability that the polled node has a packet to send to the polling node, that is:

$$P_{S1} = e^{-\lambda\tau}\left(\frac{1}{N}\right) \tag{2.14}$$

The second case of a successful busy period happens when the polled sender does not have a packet to send and therefore it sends a CTS back to the sender of the RTR enabling the node to send a data packet. The probability, P_{S2}, with which this scenario occurs is equal to the probability that an RTR is sent in the clear, times the probability that the polled node has no data packet for the polling node, that is,

$$P_{S2} = e^{-\lambda\tau}\left(1 - \frac{1}{N}\right) \tag{2.15}$$

As it was the case with RIMA-SP, the length of an average busy period always includes an RTR and a propagation delay, plus the average time between the first and the last RTR of the busy period. When the busy period fails, there are no additional components in it. With probability P_{S1}, a successful busy period case contains two data packets, one from the polled node followed by one from the polling node, two ACKs, plus the associated propagation delays and the collision-avoidance waiting period of ξ seconds. With probability P_{S2}, a successful busy period contains a single data packet from the polling node, an ACK, plus a CTS from the polled node and the associated propagation delays. It follows that the duration of the average busy period is given by

$$\begin{aligned}
\overline{B} &= \gamma + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}\left[\frac{1}{N}(2\delta + \xi + 2\gamma + 3\tau) + (1 - \frac{1}{N})(2\gamma + \delta + 3\tau)\right] \\
&= \gamma + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}\left[\frac{1}{N}(\delta + \xi) + 2\gamma + \delta + 3\tau\right]
\end{aligned} \tag{2.16}$$

Because inter-arrival times for RTRs are exponentially distributed, it follows that $\overline{I} = \frac{1}{\lambda}$. The average utilization time at node w is the proportion of time in which useful data are sent, consequently,

$$\begin{aligned} \overline{U} &= P_{S1}(2\delta) + P_{S2}(\delta) \\ &= e^{-\lambda\tau}\frac{2\delta}{N} + e^{-\lambda\tau}(1-\frac{1}{N})\delta = e^{-\lambda\tau}\delta(1+\frac{1}{N}) \end{aligned} \tag{2.17}$$

Eq. (2.12) follows from substituting $\overline{B}$, $\overline{I}$ and $\overline{U}$ into Eq. (2.1). *Q.E.D.*

RIMA-BP

Theorem 6 *The throughput of RIMA-BP is given by*

$$S = \frac{\delta}{\delta - \xi + \tau + \left(\frac{N}{N-1}\right)^{N-1}\left[\frac{1}{\lambda} + \gamma + \xi + 2\tau + e^{\lambda\tau}(\gamma + 2\tau)\right]} \tag{2.18}$$

where $\xi = 4\tau$.

Proof: Given our independence assumptions, the probability of success, P_S, equals the probability with which an RTR is transmitted successfully, times the probability with which an RTS is transmitted successfully. Because all nodes are connected, an RTR from node w is successful if there are no other RTRs transmitted within τ seconds, where τ is the time needed for all the nodes connected to detect the carrier signal. After this vulnerability period of τ seconds, all the nodes detect the carrier signal and act appropriately. Because the arrivals of RTRs to the channel follow the Poisson distribution with rate λ, we can write:

$$P_{S\,RTR} = e^{-\lambda\tau} \tag{2.19}$$

The probability that only one of the nodes that receive a successful RTR transmits an RTS is equal to the probability that only one neighbor has a packet ready for the polling node.

Because at each neighbor this is the case with probability $\frac{1}{N}$, and because each node has $N-1$ neighbors, this can be expressed as follows:

$$P_{S_{RTS}} = (N-1)\left(\frac{1}{N}\right)\left(1-\frac{1}{N}\right)^{N-2} \tag{2.20}$$

Therefore, the probability with which a packet is transmitted successfully is

$$\begin{aligned} P_S &= P_{S_{RTR}} P_{S_{RTS}} \\ &= e^{-\lambda\tau}(N-1)\left(\frac{1}{N}\right)\left(1-\frac{1}{N}\right)^{N-2} \end{aligned} \tag{2.21}$$

There are three ways in which a busy period can be unsuccessful; i.e., contain no data packet. First, the RTRs sent in the busy period may collide with one another, which occurs with probability $1-e^{-\lambda\tau}$ because all nodes can hear one another. A busy period can also fail if a single RTR is sent in the clear but none of the polled nodes has a packet to send to the polling node; the probability with which this scenario takes place is equal to:

$$P_{F_2} = e^{-\lambda\tau}\left(1-\frac{1}{N}\right)^{N-1} \tag{2.22}$$

The last case of a failed busy period is when an RTR is successful, but more than one RTSs are sent in response. In this case, the polling node sends an NTR immediately after detecting the collision. The probability with which this type of busy period occurs is:

$$P_{F_3} = e^{-\lambda\tau}\left[1-2\left(1-\frac{1}{N}\right)^{N-1}\right] \tag{2.23}$$

The average length of a busy period always includes the length on an RTR, a propagation delay, and the average time between the first and the last RTR in the busy period, which is the same as in the previous proofs of this section. When RTRs collide, the busy period has no additional components.

With probability P_s, the busy period also includes an RTS from a polled node, a collision-avoidance waiting time at the polling node, the data packet from the polled node, the ACK from the polling node, plus the associated propagation delays. With probability P_{F2}, the busy period also contains a waiting time of 2τ after which the polling node detects no RTSs. With probability P_{F3}, the busy period also contains the length of the RTSs that collide and its propagation delay, a collision-avoidance waiting time at the polled nodes, and a propagation delay with which the polling node starts sensing the collision. Accordingly, the duration of an average busy period is

$$\begin{aligned}
\overline{B} &= \gamma + 2\tau - \frac{1-e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}(1-\frac{1}{N})^{N-1}(2\tau) \\
&+ e^{-\lambda\tau}(1-\frac{1}{N})^{N-1}(2\gamma+\delta+\xi+3\tau) \\
&+ e^{-\lambda\tau}(1-2(1-\frac{1}{N})^{N-1})(\gamma+\xi+2\tau)
\end{aligned} \tag{2.24}$$

According to the RIMA-BP specification the channel will be idle after every data transmission for a period of $\tau+\epsilon$ seconds. In addition, the channel is idle for a time period equal to the inter-arrival rate of RTRs, so $\overline{I} = \frac{1}{\lambda}$.

The average utilization time at node w is the proportion of time in which useful data are sent. Consequently,

$$\begin{aligned}
\overline{U} &= \delta P_S = \delta e^{-\lambda\tau}\,(N-1)\left(\frac{1}{N}\right)\left(1-\frac{1}{N}\right)^{N-2} \\
&= \delta e^{-\lambda\tau}\left(1-\frac{1}{N}\right)^{N-1}
\end{aligned} \tag{2.25}$$

Substituting the equations for $\overline{U}$, $\overline{I}$ and $\overline{B}$ into Eq. (2.1) we obtain Eq. (2.18). *Q.E.D.*

Numerical Results

To compare the various RIMA protocols with MACA, FAMA-NCS, and MACA-BI, we introduce the variables in Table 2.1. We assume a fully-connected network topology with a propagation delay of $1\mu s$; we used 500 byte data packets; a length of 20 bytes for RTRs, CTSs, ACKs, and

$a = \frac{\tau}{\delta}$(normalized propagation delay)
$b = \frac{\gamma}{\delta}$(normalized control packets)
$G = \lambda \times \delta$(Offered Load, normalized to data packets)

Table 2.1: Normalized variables

NTRs for the various RIMA protocols; CTSs of length $\gamma + \tau$ for FAMA-NCS; a channel data rate of 1 Mb/s; and zero preamble and processing overhead for convenience. Figs. 2.8, 2.9 and 2.10 plot the throughput of MACA, FAMA-NCS, MACA-BI, RIMA-SP, RIMA-DP, and RIMA-BP against the average offered load when the network consists of 5, 10, and 50 nodes, respectively.

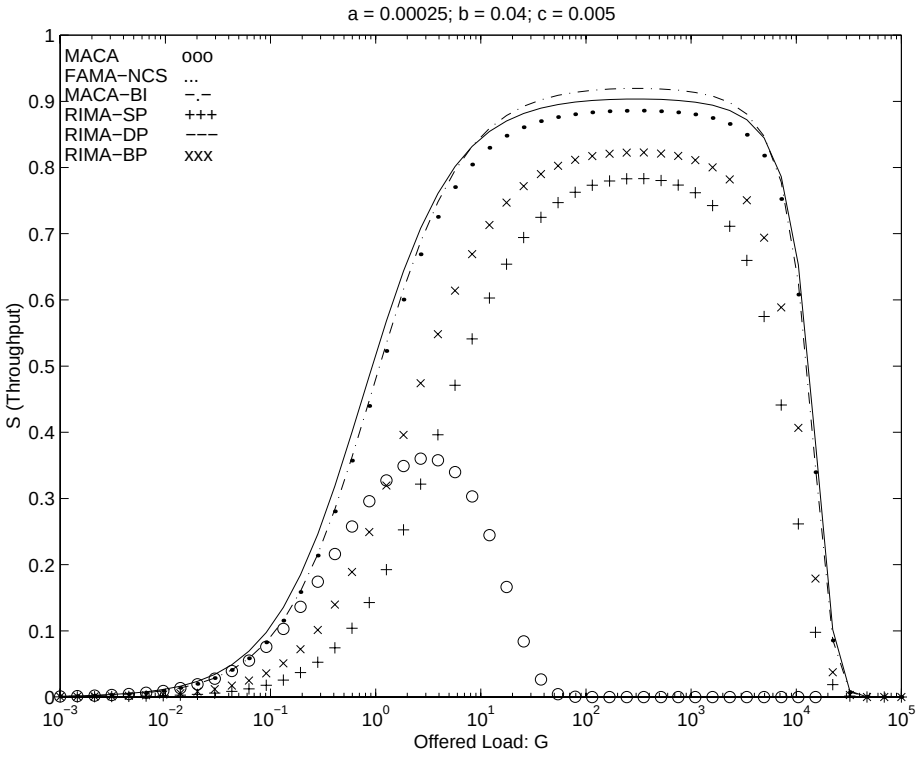


Figure 2.8: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets; network of 5 nodes

The performance attained by RIMA-DP is much better than the performance of the other MAC protocols that provide correct collision avoidance (FAMA-NCS, RIMA-SP, and RIMA-BP). This should be expected, because RIMA-DP permits one or two packets to be sent with each successful handshake, while the other protocols allow just one packet per handshake.

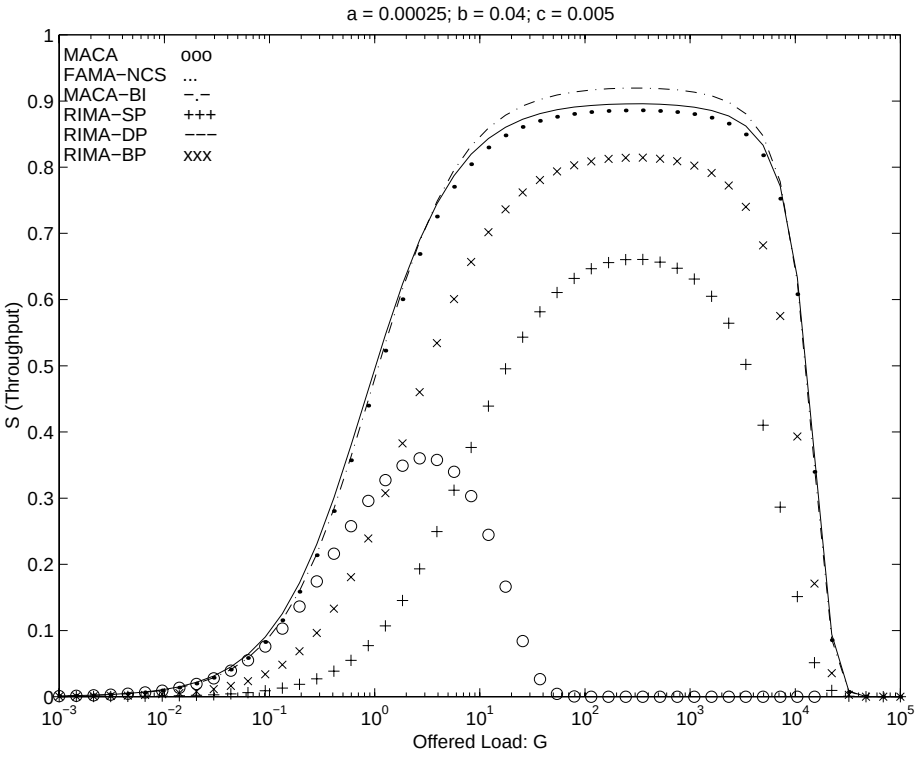


Figure 2.9: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets; network of 10 nodes

As Figs. 2.8 to 2.10 illustrate, the throughput of RIMA-SP degrades as the size of a node neighborhood increases. Even though our model is only a rough approximation of the impact of the number of neighbors a node has; This illustrates the fact that simple polling is inherently limited compared to dual-use polling, because at light and moderate loads there is a non-zero probability that the polled node has no data to send to the polling node.

It is also interesting to observe that the throughput of RIMA-BP is independent of the number of nodes and is always lower than RIMA-DP. There are two reasons for this behavior: a node receiving a broadcast poll can only transmit packets to the polling node, and multiple responses (RTSs) to the poll are likely to be sent, incurring wasted busy periods.

Figs. 2.8 to 2.10 also illustrate that carrier sensing is needed to provide high throughput in addition to correct collision-avoidance. MACA's poor performance is due to the long durations of busy periods in which collisions occur, which are bounded by a maximum round trip delay and a

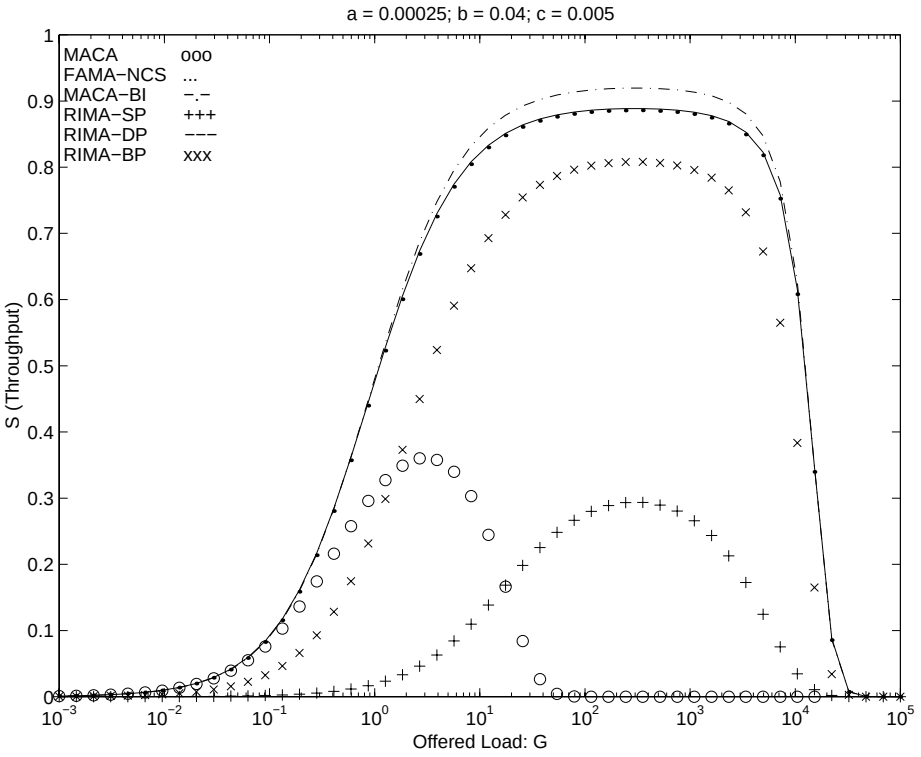


Figure 2.10: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets; network of 50 nodes

control packet length with carrier sensing. In fairness to MACA and variants of collision avoidance protocols that do not use carrier sensing, it should be emphasized once more that, with the COTS radios available today, carrier sensing is possible only with FHSS radios in ISM bands, with which entire packets are sent in a single frequency hop. In contrast, collision-avoidance without carrier sensing can be applied to FHSS and DSSS radios. However, given the performance advantage of collision avoidance using carrier sensing, FHSS radios appear more attractive than DSSS radios for ad-hoc networks.

In Figs. 2.8 to 2.10, MACA-BI achieves the maximum throughput among all the protocols. The reason for this is that a polled node can transmit a data packet to any node, not just the polling node; however, as we have shown in Section 2.2, a polled node should transmit only if it has data meant for the polling node in order to avoid sending data to a neighbor that cannot receive the data in the clear. Nevertheless, the good performance of MACA-BI reported by Talucci et al. [92]

indicates that a receiver-initiated collision-avoidance protocol should perform very well when nodes have traffic to send to most of their neighbors. To provide a fairer comparison between MACA-BI and RIMA protocols without having to consider a more complex model involving hidden terminals, we can use a heavy-traffic approximation consisting of assuming that a polled node always has data to send to any polling node. This approximation is actually not far from reality in large networks in which a node always has packets in its transmission queue meant for different destinations and has to distribute them among its various neighbors. With this approximation, the probability that a successful RTR generates two data packets in RIMA-DP is 1, and the probability that an RTR is not answered with data in RIMA-SP is 0; Fig. 2.11 shows the corresponding results. As could be expected, under the heavy-traffic assumption, RIMA-DP achieves the best throughput under any average load, and RIMA-SP exhibits essentially the same throughput as MACA-BI. Both RIMA-DP and RIMA-SP achieve higher throughput than FAMA-NCS in this case as well.

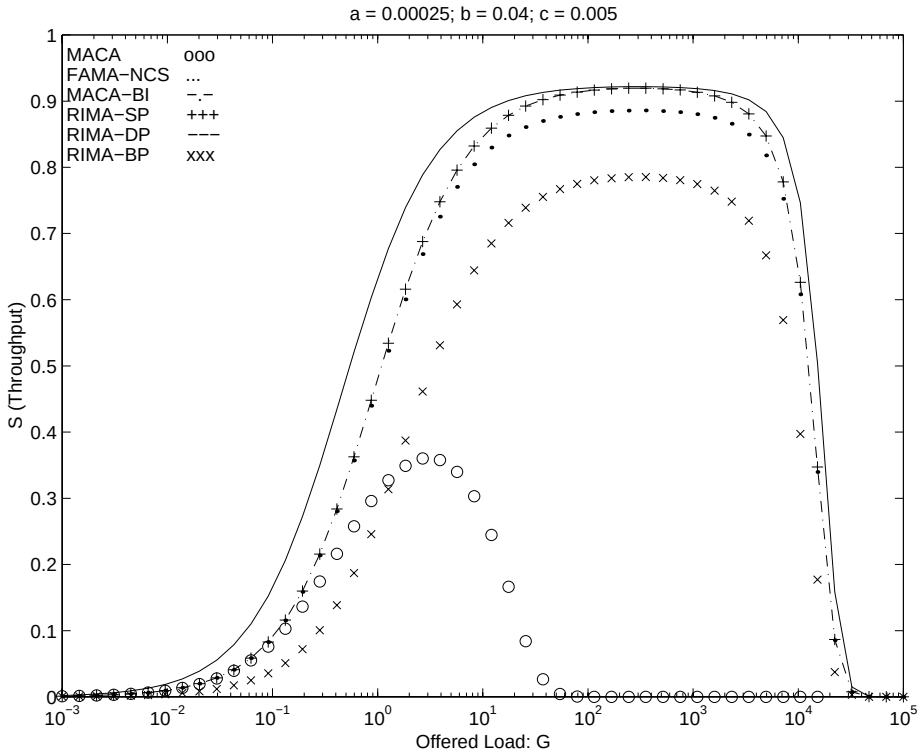


Figure 2.11: Heavy-traffic approximation: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets; network of 50 nodes

Our analysis shows a number of interesting results. By making collision-avoidance a joint effort by sender and receiver (as we do in RIMA-DP), a much better performance is obtained than what can be achieved with FAMA-NCS; this should be expected, because dual-use polling doubles the opportunity for collision-free data to be sent. Protocols based on simple polling (RIMA-SP) perform much better than MACA, but their performance degrades with the number of neighbors that a node has. RIMA-SP exhibits lower performance than FAMA and RIMA-DP, because polled nodes can send data packets only to polling nodes to avoid collisions; therefore, at low and moderate loads, there is a non-zero probability that a polled node has nothing to send to the polling node. An interesting result of the analysis is that undirected polling (RIMA-BP) always has lower performance than dual-purpose polling, which should be expected, but is more attractive than simple polling when the node neighborhood is large.

2.4.2 Average Delay

Given that RIMA-DP achieves the best throughput among the MAC protocols based receiver-initiated collision-avoidance, we obtain the average delay for RIMA-DP only. To calculate the average delay that a packet ready for transmission experiences with a RIMA-DP protocol, we have to know the transitions a node follows according to the packet received as well as the probability with which each of those transitions occurs. Because the inter-arrival times of packets have a Poisson distribution, we can calculate the average delay as a Markov process. We assume that there are four states in our model: ARRIVE, BACKOFF, ATTEMPT and COMPLETE, with the transitions and probabilities shown in Fig. 2.12. The ARRIVE state is visited when a packet first arrives at a node. Once a node has a packet ready to be transmitted, the channel is busy for the node with probability P_B, in which case the node has to transition to the BACKOFF state. Given that packets are arriving according to a Poisson process, the average partial transmission period is equal to $\frac{T_P}{2}$, where T_P is the duration of an average transmission period. On the other hand, the channel is clear for a node with a packet ready for transmission with probability $(1 - P_B)$, in

which case the node transitions to the ATTEMPT without any delay and tries to send the packet. With probability P_S the transmitting node is successful and the packet is transmitted within one successful transmission period or T_S seconds. This is shown in Fig. 2.12 with the transition from the ATTEMPT to the COMPLETE state where the packet is assumed to be delivered. With probability $(1 - P_S)$, the transmitting node fails and a failed transition period of length T_F seconds occurs, after which the node transitions to the BACKOFF state.

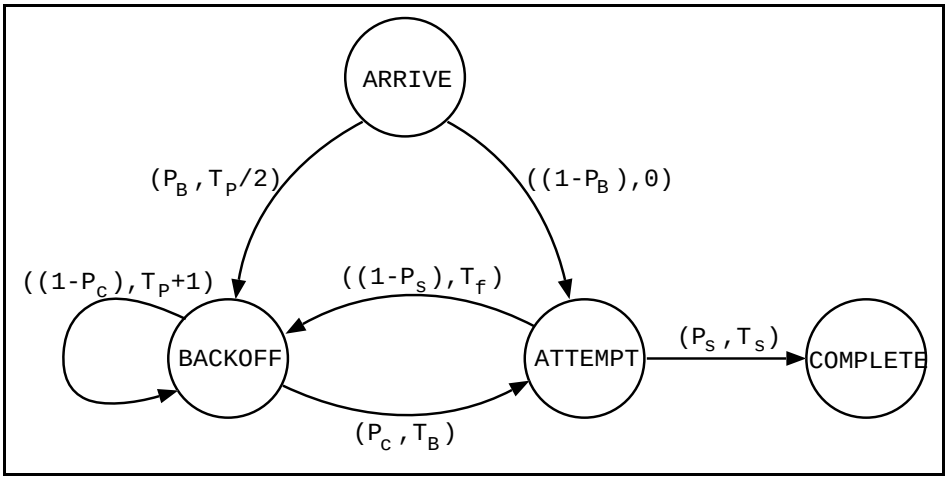


Figure 2.12: State diagram for average-delay computation

The purpose of the BACKOFF state is to express the random waiting time that a node waits before contending for the channel again. We assume that the average waiting period is T_B seconds. After T_B seconds, the node finds the channel clear with probability P_C and transitions to the ATTEMPT state in such a case. Else, the channel is found busy with probability $(1 - P_C)$ and the node has to remain in the BACKOFF state for a length of time equal to the average transmission period plus the average idle period.

Let $\overline{D}$ be the average delay for a packet to transition from the ARRIVED state to the COMPLETE state. If $E(A)$ is the expected delay every time a node is in the ATTEMPT state, and $E(B)$ is the expected delay every time a node is in the BACKOFF state, $\overline{D}$ equals

$$\begin{aligned}\overline{D} &= T_P + T_S - T_F - T_B - \overline{I} + P_B\left[T_B - \overline{I} - \frac{T_P}{2}\right] \\ &+ \frac{T_B + T_F - T_P - \overline{I}}{P_S} + \frac{(P_B - 1)\left(T_P + \overline{I}\right)}{P_C} + \frac{T_P + \overline{I}}{P_C P_S}\end{aligned} \tag{2.26}$$

Substituting the appropriate values in Eq.(2.27), we can calculate the average delay for RIMA-DP.

As we have shown in Section 2.4, $T_{S1} = 3\gamma + 2\delta + \xi + 4\tau = 4\gamma + 2\delta + 11\tau$, P_{S1} is given in Eq. 2.14, $T_{S2} = 3\gamma + \delta + 4\tau$, and P_{S2} is given in Eq. 2.15. Therefore, T_S in RIMA-DP is equal to

$$T_S = \frac{1}{2}\left[(3\gamma + \delta + 4\tau) + (\frac{1}{N})(\delta + \gamma + 7\tau)\right] e^{-\lambda\tau} \tag{2.27}$$

Eq. 2.13 gives P_S, and we have already calculated $\overline{I}$, and $\overline{U}$ for RIMA-DP. Therefore, P_B is equal to

$$P_B = \frac{\gamma + 2\tau - \frac{1-e^{-\lambda\tau}}{\lambda} + \left[(3\gamma + \delta + 4\tau) + (\frac{1}{N})(\delta + \gamma + 7\tau)\right] e^{-\lambda\tau}}{\frac{1}{\lambda} + \left[(3\gamma + \delta + 4\tau) + (\frac{1}{N})(\delta + \gamma + 7\tau)\right] e^{-\lambda\tau}} \tag{2.28}$$

The probability P_C is equal to the probability that there is no arrival during the waiting period, that is,

$$P_C = e^{-\lambda T_B} \tag{2.29}$$

For simplicity, we assume that $T_F = \gamma + \epsilon + 2\tau$ when a node fails to acquire exclusive use of the channel. The average transmission period T_P is once again equal to an average busy period or $T_P = \overline{B}$.

Substituting all the above into Eq.(2.27), we find that the average delay for a packet for RIMA-DP is equal to

$$\overline{D}_{RIMA-DP} =$$

$$\overline{B} + \frac{1}{2}\left[(3\gamma + \delta + 4\tau) + (\frac{1}{N})(\delta + \gamma + 7\tau)\right] e^{-\lambda\tau}$$
$$-\gamma - 2\tau - \epsilon - T_B - \overline{I} + P_B(T_B - \overline{I} - \frac{\overline{B}}{2})$$
$$+T_B + \gamma + \epsilon + 2\tau - \overline{B} - \overline{I}e^{\lambda\tau}$$
$$+\left[(P_B - 1)(\overline{B} + \overline{I}) + \left(\overline{B} + \overline{I}\right) e^{\lambda\tau}\right] e^{\lambda T_B} \quad (2.30)$$

For FAMA-NCS [36] the average busy period is $\overline{B} = \gamma + 2\tau - \frac{1-e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}(2\gamma + \delta + 4\tau)$. The duration for a successful busy period is $T_S = 3\gamma + \delta + 4\tau$. The duration for a failed busy period is $T_F = \gamma + \tau + \epsilon$. The average time spend in the BACKOFF state before a node tries to acquire the floor again is $T_B = 5\gamma$. The probability of success is $P_S = e^{-\lambda\tau}$. Lastly, $P_C = e^{-\lambda T_B}$ and by substituting these values into Eq. (2.27) we can obtain the average delay for FAMA-NCS.

The average delay for FAMA-NCS and RIMA-DP is shown in Fig. 2.13. We assume $\tau = 0.0001$, $\gamma = 0.02$, $\epsilon = 0.002$, and $T_B = 5 * \gamma$ (the size of an RTR). As we can see, RIMA-DP under any load of traffic achieves equal or lower average delays than FAMA-NCS.

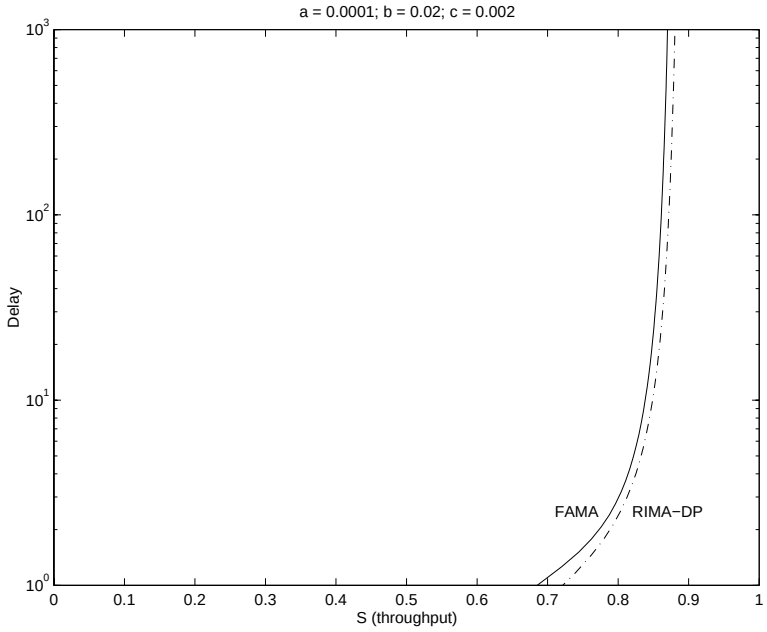


Figure 2.13: Delay performance of FAMA-NCS and RIMA-DP

2.5 Simulation Results

We validated our analytical results by performing a number of simulation experiments. Our goal is to investigate the performance of RIMA-DP under different network topologies and to show how it compares against CSMA and FAMA-NCS [36]. We used the OPNET simulation tool to implement the protocols. Table 2.2 presents the results for RIMA-DP, FAMA-NCS, and MACAW; the results for MACAW are taken from [13].

For the simulation experiments, we used a single channel capable radios that can only receive or transmit data at any given time at a data at a rate of 1Mbps. Nodes are assumed to be approximately one mile away from each other, giving a maximum propagation delay of 5 microseconds. We included a transmit-to-receive turnaround time of 20 microseconds, and the ramp-up and ramp-down of radios time was set to 5 microseconds. Figure 2.14 shows the various topologies used in the experiments, which are the same as those used in the past to show that FAMA-NCS performs better than other sender-initiated collision-avoidance protocols [36].

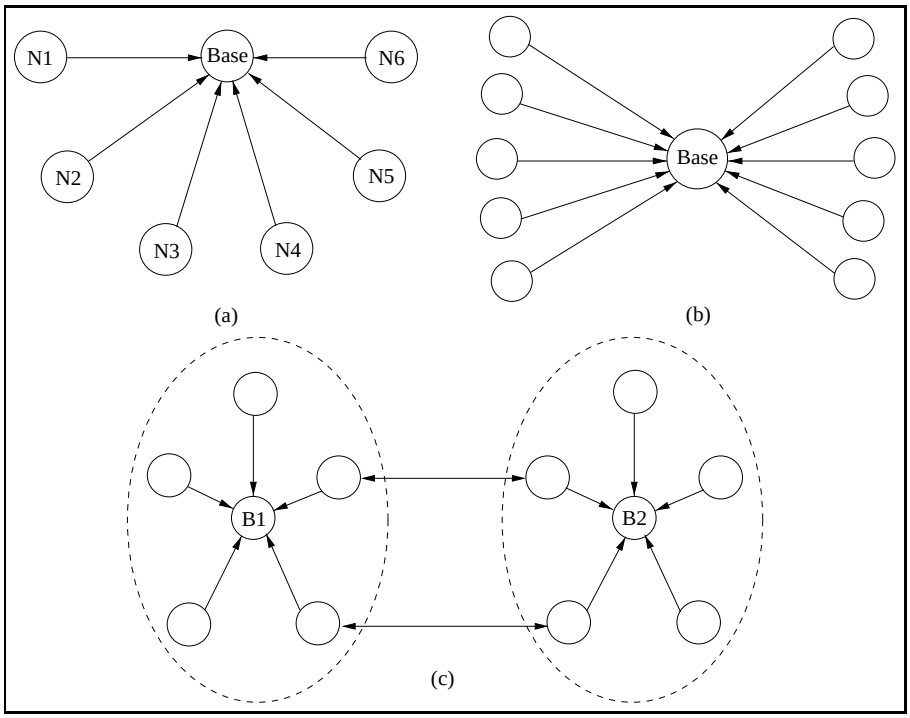


Figure 2.14: Simulation topologies used to calculate the throughput and delay of RIMA-DP

Network	RIMA-DP	FAMA-NCS	MACAW
(a)	.83	.76	.63
(b)	.58	.58	.49
(c)(B1)	.76	.74	.45
(c)(B2)	.76	.74	.39

Table 2.2: Maximum throughput results for various network topologies

Topology (a) of Figure 2.14 shows a fully-connected network in which all the traffic produced from nodes $N1$ to $N6$ is directed to the base station. Topology (b) has two groups of five nodes that can hear each other node in the same group but are hidden from all the nodes in the other group. Again, traffic is generated from all the nodes in each group with destination the central base station $B1$. Topology (c) has two separate base stations $B1$ and $B2$ with a group of five nodes sending traffic to each one of them respectively. All the nodes in each group are hidden from the nodes in the other group with the exception of two nodes in each group that can interfere with their peers. Topology (d) shows a multi-hop network of eight nodes. The lines between the nodes show the connectivity in the network whereas the lines with arrows denote unidirectional flows of traffic. A node is generating traffic that at least three other nodes will receive at any given time whereas there are always at least two other nodes that are hidden.

Data packets are generated according to a Poisson distribution and the data packet size is assumed to be constant equal to 512 bytes, which render a capacity of approximately 244 data packets per second, without including any overhead. Table 2.2 shows the maximum throughput achieved by RIMA-DP, FAMA-NCS and MACAW as reported by Bharghavan et al. [13]. For the fully-connected topology (Figure 2.14 (a)), RIMA-DP achieves a maximum throughput of 83% while FAMA-NCS achieves a 76% throughput. RIMA-DP achieves a maximum throughput larger than or equal to that of FAMA-NCS in all the topologies we considered. These results confirm the results obtained in, and predictions derived from, our analytical model.

2.6 Conclusions

We have presented the first treatment of collision-avoidance based, at the receiver instead of the sender, that demonstrates the required features of such handshakes in order to eliminate the possibility of data packets colliding with any other packets at the intended receiver in single-channel networks with hidden terminals. Our simple comparative analysis of throughput of receiver-initiated multiple access protocols shows that a receiver-initiated collision-avoidance strategy can be made more efficient than any of the sender-initiated strategies used and proposed to date. Especially the version of RIMA protocols that is based on dual-purpose polling (RIMA-DP) is always better than any other sender-initiated collision-avoidance schemes.

Although we have analyzed RIMA protocols in fully-connected networks only, the importance of our analysis is in showing which type of collision-avoidance handshake should be investigated further. Because RIMA protocols provide correct collision-avoidance in any topology, the relative performance differences among these protocols apply also to networks with hidden terminals. It is clear from our results that strategies in which dual-purpose polls are used and in which polls are directed to specific neighbors are the ones that should be implemented. Simulation experiments using OPNET confirm the results of our simple analytical model; the results of the simulations show that RIMA-DP achieves a higher maximum throughput than FAMA-NCS or MACAW in networks with hidden terminals.

The receiver-initiated collision-avoidance protocols described in this chapter assume the ability of radios to sense the channel, which is not always possible; accordingly, developing correct collision-avoidance strategies that do not rely on carrier sensing is an important area that needs to be addressed. Chapter 4, addresses this issue.

Chapter 3

Exerting Adequate Persistence in Collision Avoidance Protocols

The vast majority of comparative performance analysis to date for both sender- and receiver-initiated collision-avoidance protocols [35, 36, 42, 41, 92] assumes non-persistent channel access for the transmission of collision-avoidance control packets. With a non-persistence approach to collision-avoidance, a node senses the channel before transmitting collision-avoidance control packets. If the channel is sensed idle, the node transmits its control packet; otherwise, the node backs off for a random amount of time and attempts to transmit at that later time. In this chapter, we are interested in proposing and analyzing MAC protocols that make use of persistent carrier sensing as a mechanism that can drastically improve the utilization of the medium under certain load conditions.

The use of persistence in MAC protocols has been reported for CSMA [57, 87] and CSMA/CD [94]. A variation of the CSMA protocol, CSMA with collision avoidance (CSMA/CA) and 1-persistent carrier sensing is the medium access protocol used by *Ethernet* systems found in the vast majority of local area networks all over the world. The persistence strategies reported in

the literature consist of a node with a packet to send, that senses the channel being busy, to persist with certain probability in sending its packet as soon as the channel is sensed idle again. As traffic load increases in the channel, the likelihood that many nodes will try to transmit immediately at the end of an ongoing transmission increases substantially, which makes traditional persistent CSMA and CSMA/CD unattractive for networks without light traffic loads.

We introduce a new persistence strategy aimed at collision-avoidance MAC protocols that limits the contention among nodes that receive packets, to send at the time the channel is busy. The limited persistence mechanism we introduce is very simple and consists of establishing a time bound on how long a node can persist transmitting once it has a packet to send and sensed the channel busy. More specifically, when a node receives a packet to send (control packet or data packet depending on the protocol), it senses the channel. If the channel is sensed to be idle and no other node is known to have the right to transmit, the node transmits its packet; otherwise, the node persists trying to send its packet for a *persistence time* of η seconds, which by design is much smaller than a data packet and, in the case of collision-avoidance protocols, is proportional to the transmission time of a control packet. If the channel becomes idle before the persistence time of the node elapses, the node transmits its packet if no other node has the right to transmit; otherwise, the node backs off for a random amount of time and attempts to transmit at that later time.

The objective of introducing a limited window of persistence in collision-avoidance protocols is twofold. Protocol performance is improved at light loads by allowing some degree of persistence, because it reduces the number of times in which a single node with a packet to send after sensing the channel busy must back off for a relatively long time. At the same time, limiting the amount of time any node can persist transmitting after detecting a busy channel improves performance relative to traditional persistent strategies, because it reduces the amount of contention at the time the channel becomes idle. The main contribution of this chapter consists of showing that persistence in collision-avoidance can be beneficial to the performance of the system, provided

that the adequate amount of persistence is applied.

Section 3.1 describes sender- and receiver-initiated protocols with limited persistence; we modify FAMA-NCS [36] and RIMA protocols [42] to operate with limited persistence, because they have been shown to be the best performing sender-initiated and receiver-initiated collision-avoidance protocols with non-persistent carrier sensing. Section 3.2 uses an analytical model to study the throughput of these protocols in fully-connected networks and compares the performance of the protocols with non-persistence and limited persistent carrier sensing. We use a fully-connected network topology to discern the relative performance advantages of different protocols, because of two reasons: (a) it allows us to use a short analysis that can be applied to several protocols, and (b) our focus on protocols that provide correct collision-avoidance means that the relative performance differences in a fully-connected network are very much the same when networks with hidden terminals are considered.

3.1 Limited Persistence Collision Avoidance Protocols

Carrier sensing has been shown to increase the throughput of sender-initiated and receiver-initiated collision-avoidance and to be necessary to avoid collisions of data packets with other packets at the receivers [35, 42] in single-channel networks. The rest of this section describes sender- and receiver-initiated collision-avoidance protocols with limited-persistence carrier sensing (LCS). The proofs that these protocols support correct collision avoidance in the presence of hidden terminals are essentially the same as those published for the non-persistent versions of the protocols.

3.1.1 Sender-Initiated Protocols

In sender-initiated collision-avoidance we describe a variant of FAMA-NCS, which is based on non-persistent carrier sensing. This variant is called FAMA-LCS (limited-persistence carrier sensing), and its operation on a fully-connected network is depicted in Fig. 3.1.

In FAMA-LCS, the sender of a packet transmits a short Request-To-Send (RTS) packet asking the receiver permission to transmit. To send its RTS, the sender uses LCS. More specifically, if the sender senses the channel to be idle and no other node has the floor (right to transmit), the sender transmits its RTS. Alternatively, if the sender senses a busy channel, it persists trying to transmit its RTS for a persistence time of η seconds equal to or smaller than the transmission time of an RTS (γ). If the channel becomes idle before the persistence time elapses, the sender transmits its packet, unless another node has the right to transmit on the channel. Otherwise, the sender backs off for a random amount of time and attempts to transmit its RTS at a later time.

Once an RTS is sent, the receiver responds to a correctly addressed RTS with a Clear-To-Send (CTS) packet. The sender transmits its data packet upon reception of the CTS, and the receiver sends an acknowledgment to a correctly received data packet.

As in FAMA-NCS, the length of a CTS in FAMA-LCS equals the length of an RTS plus at least a maximum round-trip delay in order to ensure correct collision-avoidance [36].

As Fig. 3.1 illustrates, a successful RTS can occur when a node receives a packet to send when the channel is idle, as well as when the channel is busy, provided that the channel becomes available within η seconds from the arrival of the packet to be sent. Similarly, an RTS can fail when multiple RTSs are sent within τ seconds of one another when the channel is idle, or when multiple RTSs are scheduled for transmission within the last η seconds of a transmission period, after which the channel becomes available.

3.1.2 Receiver-Initiated Protocols

In receiver-initiated collision-avoidance protocols, the receivers poll the senders for packets to be sent. We assume that this polling is data driven, in which a node attempts to poll its neighbors at a rate that is a function of the data rate with which it receives data to be sent, as well as the rate with which the node hears its neighbors send control and data packets. We present variants of RIMA protocols that incorporate LCS and differ on the type of polling packets sent by the

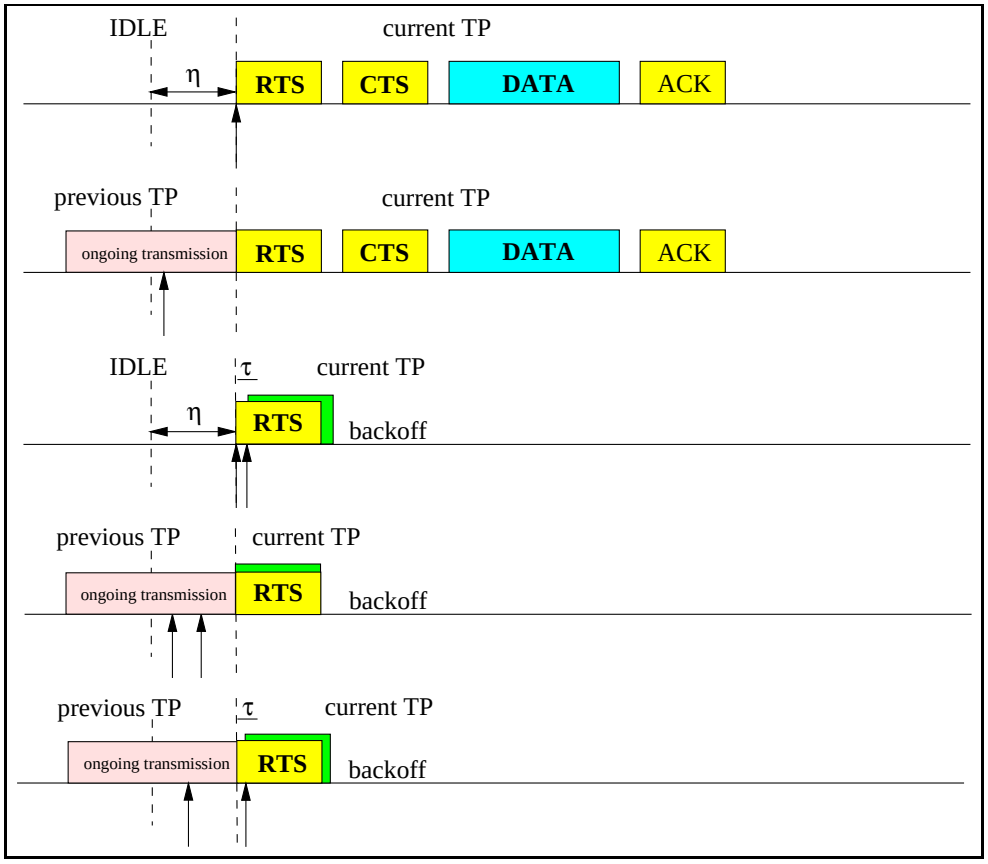


Figure 3.1: FAMA-LCS illustrated

receivers.

RIMA with Simple Polling

In the RIMA-SP (simple polling) protocol [42], the receiver sends a ready-to-receive (RTR) packet to a particular sender. If the polled node has data to send, it waits for a *collision-avoidance period* of length ξ that allows the polling node to abort the transaction after detecting noise in the channel by sending a no-transmission-request (NTR) packet. If the polled node perceives the channel idle during the collision-avoidance period, it transmits its data packet to the polling node if it has any packets intended for it.

We modify RIMA-SP by making the polling node use LCS for the transmission of its RTRs. We call the resulting variant RIMA-SPL (simple polling with limited persistence). In

RIMA-SPL, the polling node senses the channel before sending its RTR, and transmits the RTR if the channel is idle and no other node has gained control of the channel. If the polling node senses a busy channel, it persists for a time lasting η seconds, which is equal to or smaller than the length of an RTR (γ). If the polling node senses that the channel becomes idle and no other node attains control of the channel before the persistence time elapses, it transmits its RTR. Otherwise, it backs off for a random amount of time and tries to send its RTR at a later time. Fig. 3.2 illustrates the operation of RIMA-SPL for a fully-connected network. Like FAMA-LCS, a successful RTR occurs when the channel is idle or becomes idle in less than η seconds from the time that a local packet has arrived. However, since RTRs are not always followed by data in RIMA-SPL we can have the two additional failed periods of Fig. 3.2(c).

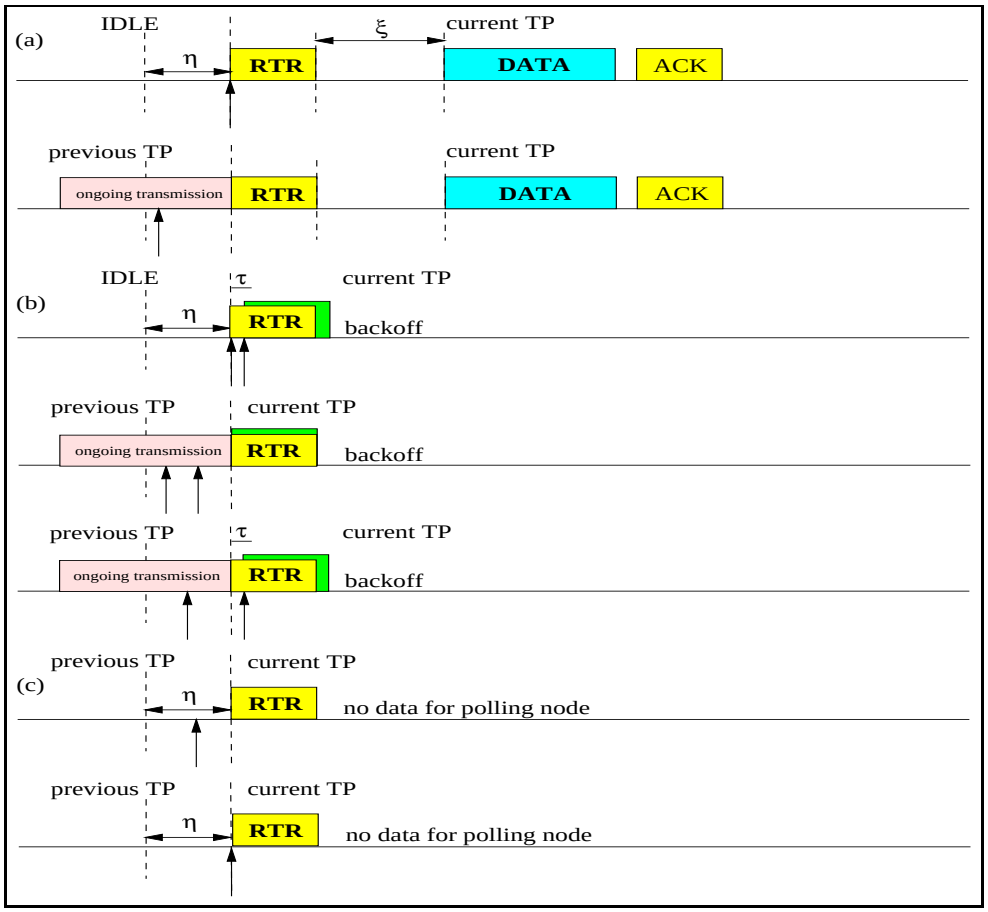


Figure 3.2: RIMA-SP with limited persistence carrier sensing

In RIMA-SPL, every node initializes itself in the START state, in which the node waits twice the maximum channel propagation delay, plus the hardware transmit-to-receive transition time (ϵ), before sending anything over the channel. This enables the node to find out if there are any ongoing transmissions. After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time that is sufficient for the node to start receiving packets in transit.

If a node x is in the PASSIVE state and senses carrier, it transitions to the REMOTE state to defer to ongoing transmissions. A node in REMOTE state must allow enough time for a complete successful handshake to take place, before attempting to transition from remote state.

Any node in PASSIVE state that detects activity in the channel must transition to the BACKOFF state. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTR state. In the RTR state, node x uses LCS to transmit an RTR. If node x detects carrier when it attempts to send the RTR, it starts a persistence timer lasting η seconds. If the channel remains busy during the η seconds or the channel becomes idle but another node gains the right to use the channel, the node transitions to the BACKOFF state. This step makes the node back off immediately for a sufficient amount of time to allow a complete handshake between a sender-receiver pair to occur; otherwise, x sends its RTR. If the node detects an idle channel and no node gains control of the channel before the η seconds of the persistence timer expire, the node transmits its RTR.

If node z receives the RTR correctly and has data for x, it waits for ξ seconds. If during the waiting period there is no activity in the channel, node z transitions to the XMIT state, where it transmits a data packet to x and node x sends an acknowledgment (ACK) immediately after receiving the data packet (Fig. 3.2(a)); otherwise, node z assumes that there was a collision and transitions to the BACKOFF state to allow floor acquisition by some other node. After sending its RTR, node x senses the channel. If it detects carrier immediately after sending its RTR, node x assumes that a collision or a successful data transfer to a hidden node is taking place. Accordingly,

it sends a No Transmission Request (NTR) to z to stop z from sending data that would only collide at x. This scenario can only occur in a multi-hop network topology.

When multiple RTRs are transmitted within a one-way propagation delay a collision takes place, and the nodes involved have to transition to the BACKOFF state and try again at a later time chosen at random, as shown in Fig. 3.2(b).

Node x determines that its RTR was not received correctly by z after a time period equal to the maximum round-trip delay to its neighbors plus turn-around times and processing delays at the nodes, plus the waiting period ξ. After sending its RTR, node x listens to the channel for any ongoing transmission. Because of non zero propagation delays, if node x detects carrier immediately after transmitting its RTR, it can conclude that it corresponds to a node other than z, which would take a longer time to respond due to its need to delay its data to x to account for turn-around times. [1]

The lengths of RTRs and NTRs are the same. The same argument used in [35] to show that the length of an RTS must be longer than the maximum propagation delay between two neighbors to ensure correct collision-avoidance can be used to show that RTRs and NTRs must last longer than a maximum propagation delay. In ad-hoc networks in ISM bands, propagation delays are much smaller compared with any packet that needs to be transmitted.

To reduce the probability that the same nodes compete repeatedly for the same receiver at the time of the next RTR, the RTR specifies a back off period unit for contention. The nodes that must enter the BACKOFF state compute a random time that is a multiple of the back off-period unit advertised in the RTR. The simplest case consists of computing a random number of back off-period units using a uniformly distributed random variable from 1 to d, where d is the maximum number of neighbors for a receiver. The simplest back off-period unit is the time it takes to send a small data packet successfully.

[1] Our analysis assumes 0 turn-around times and 0 processing delays for simplicity.

RIMA with Dual-Use Polling

RIMA-DP (dual-purpose polling) [42] improves over RIMA-SP by making the RTR into a request for data from the polled node, as well as a transmission request for the polling node to send data. We refer to the variant of RIMA-DP with limited-persistence carrier sensing by RIMA-DPL (dual-purpose polling with limited persistence). Fig. 3.3 illustrates the operation of this protocol. With RIMA-DPL, a successful RTR can be followed from one or two data packet transmission as shown in Fig. 3.3(a).

A key benefit of the dual-use polling in RIMA-DP and RIMA-DPL is that both polling and polled nodes can send data in a round of collision avoidance. This is possible because the RTR makes all the neighbors of the polling node back off, and the data from the polled node make all its neighbors back off, which can then be used by the polling node to send its data.

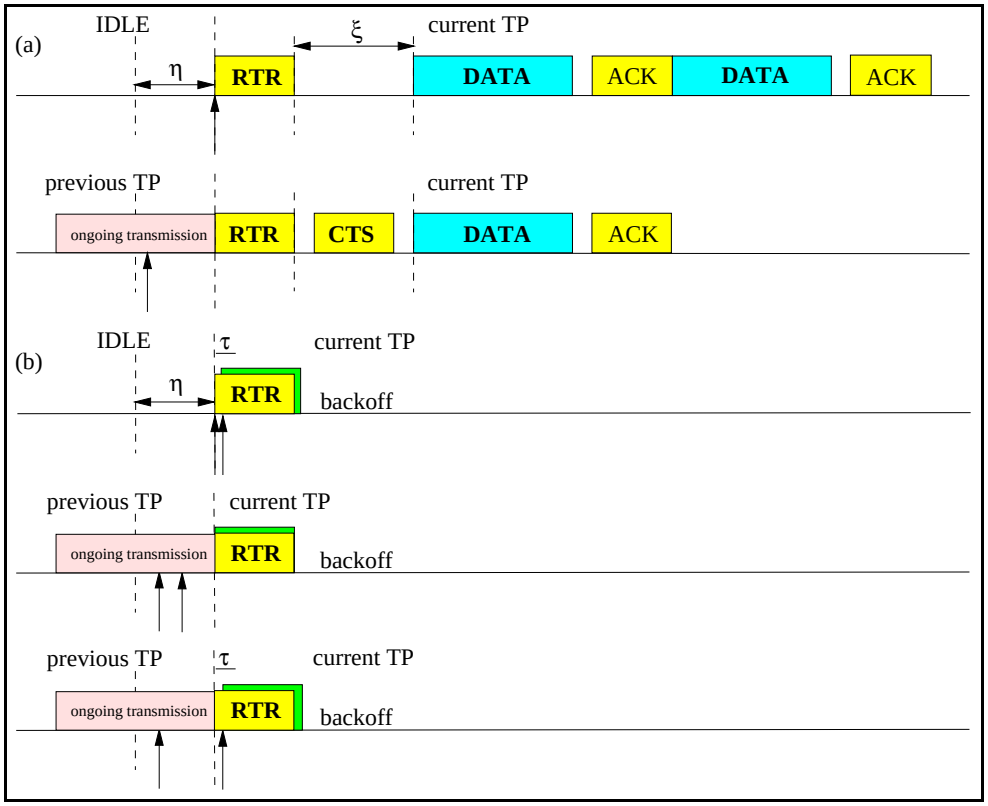


Figure 3.3: RIMA-DP with limited persistence carrier sensing

In RIMA-DPL, a sender with an RTR to be sent senses the channel, and transmits the RTR if the channel is idle and no other node has control of the channel. If the channel is busy, the sender (polling node) persists trying to send the RTR for a persistence time of length η seconds that is smaller than or equal to the length of an RTR.

RIMA-DPL gives transmission priority to the polling nodes. When a node z is polled by node x and has data for node x, z waits for a collision-avoidance period of ξ seconds before sending a data packet. In contrast, if the polled node does not have data for x, it immediately sends a CTS (Clear-To-Send packet) to x. This permits a polling node x exposed to a neighbor sending data to hear part of that neighbor's data packet after sending its RTR; in such a case, node x can send an NTR to the polled node to cancel its RTR. In [42] it is proven that RIMA-DP and consequently RIMA-DPL prevents collisions of data packets with any types of packets, provided that z waits for $\xi > \gamma + 7\tau$ seconds before sending any data after being polled and the length of a CTS is 2τ seconds longer than the length of an RTS. The lengths of RTRs and RTSs are the same.

Every node starts in the START state and transitions to the PASSIVE state when it is initialized. If a node x is in the PASSIVE state and senses carrier, it transitions to the REMOTE state to defer to ongoing transmissions. A node in REMOTE state must allow enough time for a complete successful handshake to take place, before attempting to transition from remote state.

Any node in PASSIVE state that detects activity in the channel must transition to the BACKOFF state where it must allow sufficient time for complete successful handshakes to occur. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTR state. In the RTR state, node x behaves as in RIMA-SPL.

If node z receives the RTR correctly and has data for x, it waits for ξ seconds before sending a data packet to x. If during the waiting period there is no activity in the channel, node z transitions to the XMIT state, where it transmits a data packet to x. Otherwise, z assumes a collision or data transfer to a hidden node and goes to the BACKOFF state. If z has no data for x, it sends a CTS to x immediately.

If node x detects carrier immediately after sending an RTR, it defers its transmission attempt and sends an NTR to the node it polled. The CTS length, which is τ seconds longer than an RTR, forces polling nodes that send RTRs at about the same time when a polled node sends a CTS to detect carrier from the CTS and stop their attempt to send or receive data. Any node other than x receiving the CTS for x transitions to the BACKOFF state. When node x receives the CTS from z, it transitions to the XMIT state and transmits a data packet to z.

3.2 Performance Analysis

The objective of our analysis is to compare the performance of receiver- and sender-initiated protocols with limited persistence carrier sensing and non-persistent carrier sensing. The objective of the model we use is to analyze the effect of persistence on the throughput of the system. Because the protocols we analyze ensure correct collision-avoidance in the presence of hidden terminals [36, 42], the relative differences in performance among these protocols are the same with and without hidden terminals [36]; accordingly, to simplify our model, we assume a fully-connected network.

3.2.1 Modeling Assumptions

Our modeling of limited-persistent carrier-sensing MAC protocols is based on the model first introduced by Sohraby et al. [87] which requires the following assumptions to be made:

1. There are N nodes in the fully-connected network.

2. A single unslotted channel is used for all packets, and the channel introduces no errors.

3. All nodes can detect collisions perfectly.

4. The size for a data packet is δ seconds and the size of an RTR, an ACK, and an RTS is γ seconds, the size of a CTS in RIMA protocols is γ seconds, and the size of a CTS for

FAMA-NCS is $\gamma + 2\tau$ [36].

5. The turn-around time is considered to be part of the duration of control and data packets.

6. The propagation delay of the channel between any two nodes is τ seconds.

7. The collision-avoidance interval used in RIMA protocols is ξ seconds.

8. The persistence timer in RIMA protocols is η seconds.

To provide a fair comparison between sender-initiated and receiver-initiated protocols while preserving the tractability of the analytical model, we assume that a polled node receiving an RTR always has a data packet to send, but the probability that that packet is addressed to the polling node is $\frac{1}{N}$. Furthermore, we assume that each node sends its RTR according to a Poisson distribution with a mean rate of $\frac{\lambda}{N}$, and that (when applicable) the polling node chooses the recipient of the RTR with equal probability.

The corresponding assumptions for sender-initiated protocols are that a node always has packets to send, but schedules the transmission of RTSs according to a Poisson distribution with a mean rate of $\frac{\lambda}{N}$, and chooses to which neighbor to send the RTS with probability $\frac{1}{N}$. These assumptions preserve the validity of prior analytical results for FAMA and sender-initiated collision-avoidance MAC protocols [36].

3.2.2 Throughput Analysis

As Fig. 3.4 illustrates, under steady-state operation, the utilization of the channel consists of cycles of idle periods followed by busy periods.

A busy period consists of a sequence of one or more transmission periods, and each transmission period starts with the transmission of either one or multiple control packets (RTRs for RIMA and RTSs for FAMA). We define a transmission period of type 1 (TP 1 in Fig. 3.4) to be a transmission period that starts with a single RTR or RTS. We also define a transmission period of

type 2 (TP 2 in Fig. 3.4) to be a transmission period that starts with the simultaneous transmission of two or more RTRs or RTSs. For convenience, we refer to idle periods as transmission periods of type 0 (TP 0 in Fig. 3.4).

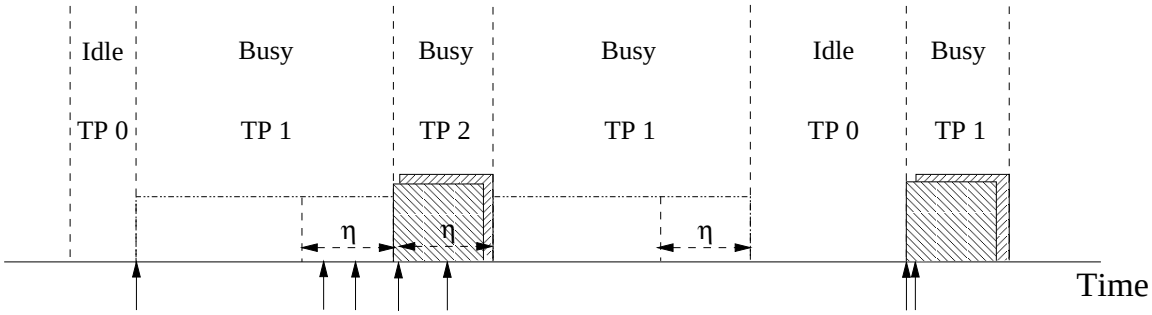


Figure 3.4: 1-persistent transmission periods

Because the arrivals of RTRs or RTSs follow a Poisson distribution, a transmission period following a transmission period of type 0 is always of type 1. Because a node persists trying to transmit an RTS or RTR for η seconds after detecting a busy channel, the type of transmission period that follows a transmission period of type 1 or 2 is defined solely by the number of RTS or RTR arrivals that occur during the last η seconds of the current transmission period.

Following the analysis by Sohraby et al. [87], we define the state of the system at the beginning of a transmission period to be the type of that transmission period. Because the type of transmission period reached depends only on the number of arrivals in the prior transmission period, the three possible states of the system and the possible transitions between them, correspond to a three-state Markov chain embedded at the beginning of the transmission periods. Fig. 3.5 illustrates this Markov chain.

No packets are transmitted during a type-0 transmission period. In contrast, during a type-2 transmission period, multiple RTRs or RTSs collide. A type-1 transmission period can be successful or unsuccessful, depending on the number of arrivals that occur during the vulnerability period of the transmission period. Furthermore, for the case of RIMA protocols, the length of a type-1 transmission period further depends on the availability of a packet for the polling node at

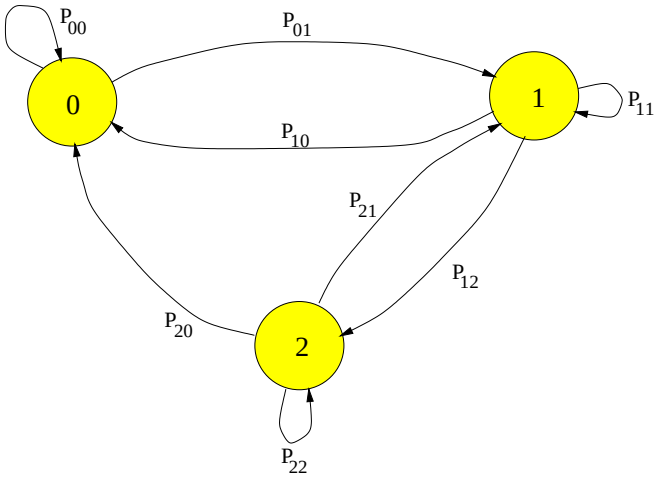


Figure 3.5: State transitions of collision-avoidance protocols with limited persistence

the polled node. The vulnerability period of a transmission period is equal to the propagation delay of τ seconds needed for all nodes to detect the transmission of RTRs or RTSs that start the transmission period.

The transition probability from state i to state j is denoted by P_{ij}. We denote by π_i ($i = 0, 1, 2$) the stationary probability of being in state i; i.e., that the system is in a type-i transmission period, and by T_i the average duration of the random variable that represents the length of a type-i transmission period. From renewal theory, we can define the throughput of the network by:

$$S = \frac{\pi_1 U}{\sum_{i=0}^{2} \pi_i T_i} \tag{3.1}$$

where U is the average time during which data packets are sent in a successful transmission period.

To compute S we need to compute the state probabilities and the average duration of transmission periods. From the Markov state diagram we have the following four equations:

$$\pi_0 = \pi_1 P_{10} + \pi_2 P_{20}$$

$$\pi_1 P_{12} = \pi_2 (P_{21} + P_{20})$$

$$\pi_0 + \pi_1 + \pi_2 = 1$$

$$P_{10} + P_{11} + P_{12} = 1 \tag{3.2}$$

The probability of transitioning to a type-0 transmission period from a type-1 or type-2 transmission period equals the probability that no RTR or RTS arrives during the last η seconds of the transmission period. The probability of transitioning to a type-1 transmission period from a type-1 or type-2 transmission period equals the probability that a single arrival of RTR or an RTS takes place during the last η seconds of the transmission period. Similarly, the probability of transitioning from a type-1 or type-2 transmission period to a type-2 transmission period equals the probability that two or more RTRs or RTSs arrive during the last η seconds of the prior transmission period. Accordingly, we have

$$P_{1j} = P_{2j} \qquad\qquad j = 0,\ 1,\ 2 \tag{3.3}$$

The state probabilities can then be obtained from Eqs. (3.2) and (3.3) to be [87]:

$$\pi_0 = \frac{P_{10}}{1 + P_{10}} \tag{3.4}$$

$$\pi_1 = \frac{P_{10} + P_{11}}{1 + P_{10}} \tag{3.5}$$

$$\pi_2 = \frac{1 - P_{10} - P_{11}}{1 + P_{10}} \tag{3.6}$$

To compute the transition probabilities P_{11} and P_{10}, let Y denote the random variable representing the arrival time of the last RTR or RTS that arrives during the vulnerability period of a transmission period. Conditioning on $Y = y$, a transition from a type-1 transmission period to a type-0 transmission period occurs if there are no arrivals of RTRs or RTSs in the time interval spanning the last η seconds of the type-1 transmission period and the first y seconds of the type-0

transmission period. Given that the arrival of RTRs or RTSs is Poisson with parameter λ, the probability of this event equals $e^{-\lambda(\eta+y)}$. Unconditioning, we have [87]:

$$P_{10} = (1+\lambda\tau)e^{-\lambda(\eta+\tau)} \tag{3.7}$$

Following the same approach, we find that

$$P_{11} = \lambda e^{-\lambda(\eta+\tau)}[\eta + \lambda\tau(\eta+\tau/2)] \tag{3.8}$$

Substituting Eqs. (3.7), (3.8), and (3.4) to (3.6) in Eq. (3.1), we obtain that the throughput of the system equals

$$\begin{aligned} S &= \frac{U(P_{10}+P_{11})}{T_0P_{10} + T_1(P_{10}+P_{11}) + T_2(1-P_{10}-P_{11})} \\ &= \frac{U(1+\lambda\tau+\lambda[\eta+\lambda\tau(\eta+\tau/2)])}{T_0(1+\lambda\tau) + T_1(1+\lambda\tau+\lambda[\eta+\lambda\tau(\eta+\tau/2)]) + T_2(e^{\lambda(\eta+\tau)}-1-\lambda\tau-\lambda[\eta+\lambda\tau(\eta+\tau/2)])} \end{aligned} \tag{3.9}$$

The throughput achieved by each collision-avoidance protocol can now be obtained as a function of the rate of arrival of RTRs or RTSs in the system by obtaining the values of U, T_0, T_1, and T_2 for each protocol.

The throughput of collision-avoidance protocols is specified in Theorems 1 to 3 below, making use of the following definitions:

$$A = e^{\lambda\tau}(\gamma + 2\tau - 1/\lambda)$$

$$B = 1 + \lambda\tau + \lambda[\eta + \lambda\tau(\eta+\tau/2)]$$

Theorem 7 *The throughput for FAMA-LCS in a fully-connected network is given by*

$$S = \frac{\delta B}{e^{\lambda\tau}(\frac{1}{\lambda}+\tau) + (A+\frac{1}{\lambda}+2\gamma+\delta+5\tau)B + (A+\frac{1}{\lambda})(e^{\lambda(\eta+\tau)}-B)} \tag{3.10}$$

Proof: Because the arrival of RTSs is Poisson with parameter λ, type-0 transmission periods are exponentially distributed and $T_0 = \frac{1}{\lambda}$. A type-2 transmission period consists of multiple RTSs

starting at the beginning of the period, and can also contain additional RTSs that arrive to the channel within the vulnerability period of the period; therefore, $T_2 = \gamma + \tau + \overline{Y}$ seconds. The average value of Y is the same as in CSMA and equals [95] $\overline{Y} = \tau - \frac{1-e^{-\lambda\tau}}{\lambda}$. Therefore,

$$T_2 = \gamma + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} \tag{3.11}$$

A type-1 transmission period always contains an RTS and the associated propagation delay. If no RTSs arrive within τ seconds from the start of the transmission period, the period is successful and includes in addition a CTS, a data packet, an ACK, and the associated propagation delays. A type-1 transmission period is successful when no RTSs arrive within τ seconds from the start of the period, which also means that the first and the last RTS that arrives within τ seconds of its start are the same. Because RTS arrivals are Poisson with parameter λ and a CTS lasts $\gamma + 2\tau$, we obtain:

$$T_1 = \gamma + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + e^{-\lambda\tau}(2\gamma + \delta + 5\tau) \tag{3.12}$$

The average utilization period in FAMA-LCS always lasts δ seconds. Therefore, because a type-1 transmission period succeeds with probability $e^{\lambda\tau}$, we have $U = e^{-\lambda\tau}\delta$ and the theorem follows by substituting the average values of transmission and utilization periods in Eq. 3.10. *Q.E.D.*

Theorem 8 *The throughput for RIMA-SPL with $\xi = \tau$ in a fully-connected network is given by*

$$S = \frac{\delta B \frac{1}{N}}{(A + \frac{1}{\lambda} + \frac{1}{N}(\gamma + \delta + 2\tau))B + e^{\lambda\tau}(\frac{1}{\lambda} + \tau) + (A + \frac{1}{\lambda})(e^{\lambda(\eta+\tau)} - B)} \tag{3.13}$$

Proof: Because the arrival of RTRs is Poisson with parameter λ, type-0 transmission periods are exponentially distributed and $T_0 = \frac{1}{\lambda}$. The average length of type-2 transmission periods is the same as in FAMA-LCS and given in Eq. (3.11), given that RTRs and RTSs last γ seconds.

A type-1 transmission period in RIMA-SPL always contains an RTR, the associated propagation delay, and the collision-avoidance waiting time, all of which lasts $\gamma + \tau + \xi$. When no RTRs

arrive within τ seconds from the start of the transmission period, the period is successful if the polled node has a packet ready for the polling node; this happens with probability $e^{\lambda\tau}/N$. In this case the period also includes a data packet, an ACK, and two propagation delays. Therefore,

$$T_1 = \gamma + \xi + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + \frac{e^{-\lambda\tau}}{N}(\delta + \gamma + 2\tau) \tag{3.14}$$

The average utilization period of RIMA-SPL always lasts δ seconds. An RTR succeeds in obtaining a data packet from a polled node if no other RTRs are sent within τ seconds of its start time and the polled node has data to send to the polling node; this probability equals $e^{-\lambda\tau}/N$. Therefore, $U = \delta e^{-\lambda\tau}/N$. The theorem follows by substituting the average values of transmission and utilization periods in Eq. 3.10. *Q.E.D.*

Theorem 9 *The throughput of RIMA-DPL with $\xi > \gamma + 7\tau$ in a fully connected network is given by*

$$S = \frac{(2e^{-\lambda\tau} - \frac{1}{N})\delta B}{e^{\lambda\tau}(\frac{1}{\lambda} + \tau) + (A + \frac{1}{\lambda})(e^{\lambda(\eta+\tau)} - B) + (A + \frac{1}{\lambda} + e^{\lambda\tau}(2\delta + 2\gamma + 3\tau) - \frac{1}{N}(\delta + 2\gamma + 8\tau))B} \tag{3.15}$$

Proof: As in RIMA-SPL, the average length of type-0 transmission periods is $T_0 = \frac{1}{\lambda}$ and the average length of type-2 transmission periods is given by Eq. (3.11).

A type-1 transmission period in RIMA-SPL always contains an RTR, the associated propagation delay, and the collision-avoidance waiting time, all of which lasts $\gamma+\tau+\xi$. If no RTRs arrive within τ seconds from the start of the transmission period, there are two mutually exclusive cases to consider. If the polled node has no data packet to send to the polling node, the transmission period also includes a CTS, a data packet, and the associated propagation delays. Alternatively, if the polled node has data to send to the polling node, the period also includes a collision-avoidance interval, two data packets, two ACKs, and three propagation delays given that the polling node sends its ACK and a packet in sequence. Therefore, we obtain

$a = \frac{\tau}{\delta}$(normalized propagation delay)
$b = \frac{\gamma}{\delta}$(normalized control packets)
$h = \frac{\eta}{\delta}$(normalized persistence duration)
$G = \lambda \times \delta$(Offered Load, normalized to data packets)

Table 3.1: Normalized variables

$$T_1 = \gamma + \xi + 2\tau - \frac{1 - e^{-\lambda\tau}}{\lambda} + \frac{e^{-\lambda\tau}}{N}(\delta + \gamma + 2\tau) + (1 - \frac{e^{-\lambda\tau}}{N})(2\delta + 2\gamma + \xi + 3\tau) \tag{3.16}$$

Given that a successful type-1 transmission period contains one data packet when the polled node has not data to send and two data packets when it does, the length of the average utilization period in RIMA-DPL equals

$$U = \frac{e^{-\lambda\tau}}{N}(\delta) + (1 - \frac{e^{-\lambda\tau}}{N})(2\delta) \tag{3.17}$$

Eq. (3.15) is obtained by substituting the average values of transmission and utilization periods in Eq. 3.10. *Q.E.D.*

3.2.3 Performance Comparison

To compare the limited-persistence collision-avoidance protocols introduced in this chapter, we introduce the variables listed in Table 3.1. We assume a fully-connected network topology with a propagation delay of $1\mu s$, the channel data rate of 1 Mbps, and preamble and processing overhead are ignored for convenience. Data packets are assumed to consist of 500 bytes, and the RTRs, ACKs, and RTSs used in all protocols consist of 20 bytes. For the case of RIMA-DPL, a CTS is also 20 bytes, and for FAMA-LCS a CTS lasts one round-trip longer than an RTS.

Figs. 3.6, 3.7 and 3.8 plot the throughput of FAMA-LCS, RIMA-SPL, and RIMA-DPL, against the average offered load of RTSs or RTRs when the network consists of 10 nodes. The figures also show the non-persistent variants of the collision-avoidance protocols. The three figures illustrate that proper amounts of limited persistence make all the collision-avoidance schemes more

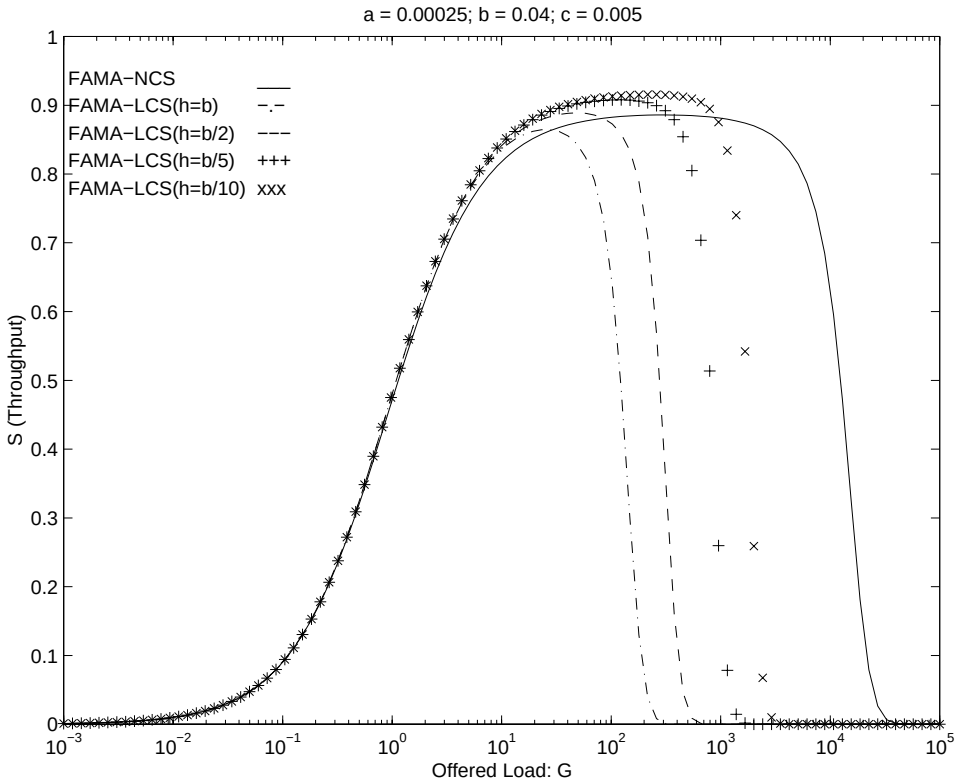


Figure 3.6: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets for FAMA-LCS with different persistence intervals; network of 10 nodes

efficient. In all cases, a persistence time of just a fraction (e.g., one half) of the transmission time of a control packet (RTR or RTS) gives the best results. As should be expected, all the protocols analyzed achieve higher throughput at light loads and sustained throughput up to moderate average loads. More marked improvements are obtained in receiver-initiated collision-avoidance strategies than in sender-initiated strategies. The best results are obtained with RIMA-DPL, in which case the throughput at light average offered loads is markedly higher than in the non-persistence strategy.

The reason why limited persistence improves the efficiency of collision-avoidance protocols is that it tends to eliminate idle-time periods in the channel at light average loads, because stations are allowed to persist in their attempt to acquire control of the channel after detecting an ongoing transmission. Furthermore, throughput remains higher than with non-persistence at moderate offered loads, because only a fraction of those RTSs or RTRs that become ready for transmission

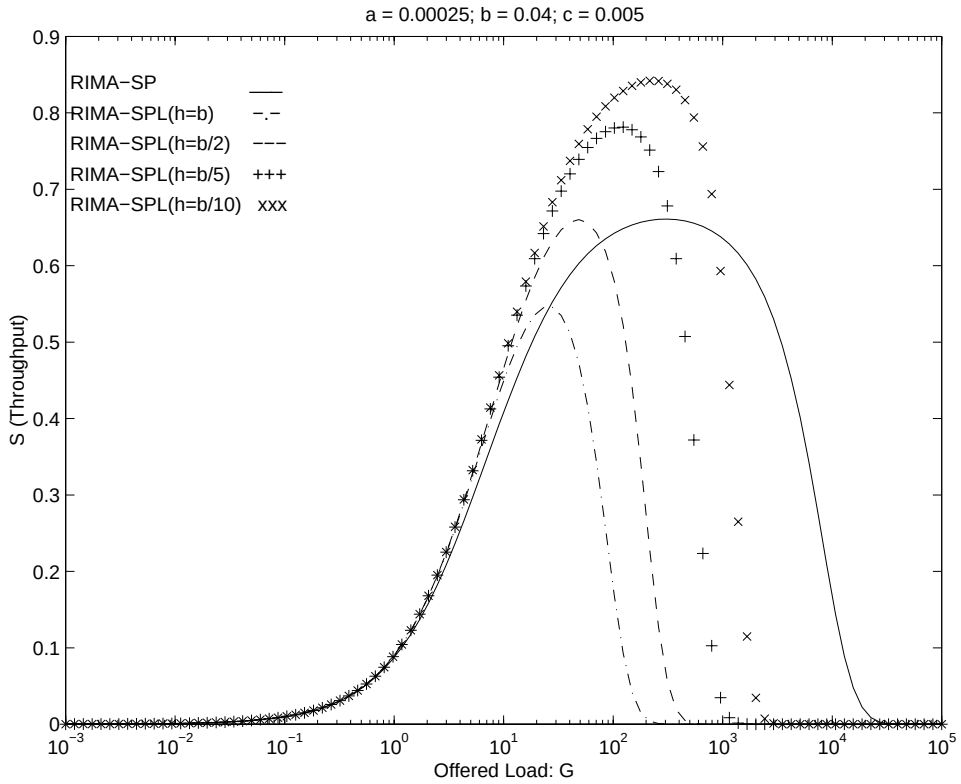


Figure 3.7: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets for RIMA-SPL with different persistence intervals; network of 10 nodes

while the channel is busy are allowed to contend for the channel when the channel becomes idle.

Given that all contention-based protocols, including collision-avoidance protocols, should operate in regions where offered traffic loads are light to moderate, our analysis shows that introducing limited persistence, together with back off strategies that reduce the average offered load as congestion starts to mount, is the right approach to making collision-avoidance protocols more efficient.

3.3 Conclusions

We introduced a set of new sender- and receiver-initiated MAC protocols that use limited persistence carrier sensing. Limited persistence consists of allowing a station that detects a busy channel when it receives a packet to send to persist in engaging in a collision-avoidance dialogue

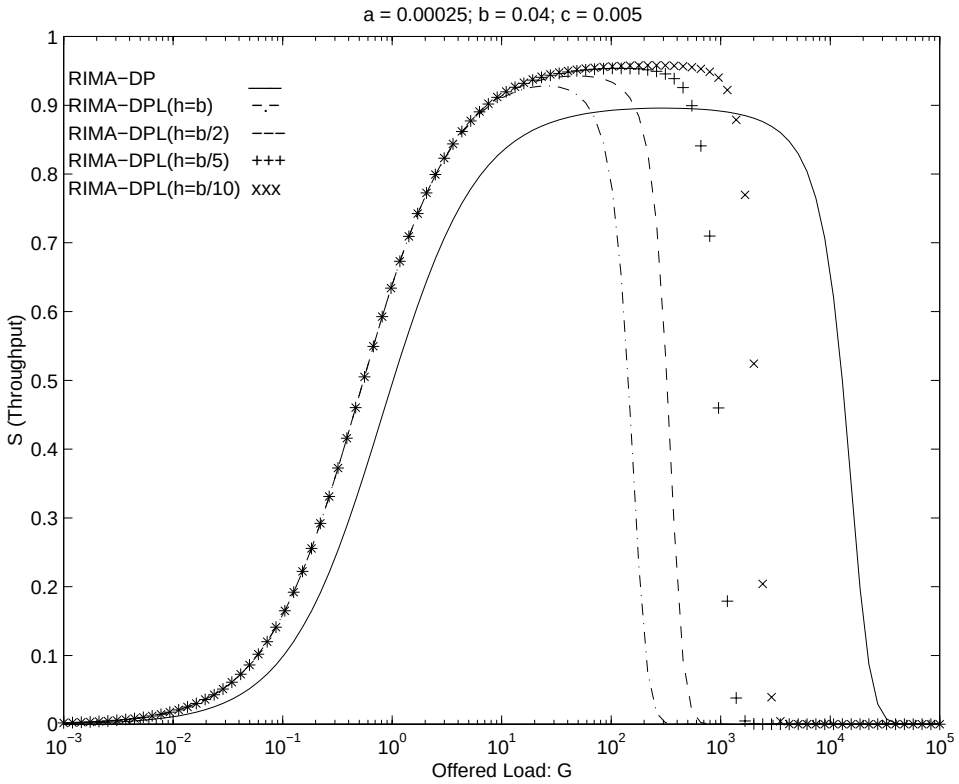


Figure 3.8: Throughput vs. offered load for 1Mbit/sec channel and 500 Byte data packets for RIMA-DPL with different persistence intervals; network of 10 nodes

for a limited amount of time only. The station is forced to back off for a random amount of time if the channel is busy or another station acquires control of the channel at the end of the persistence time.

Our analysis of limited persistence is based on earlier work on 1-persistent CSMA by Sohraby et al. [87]. Although this analysis assumes a fully-connected network, our results can be extrapolated to networks with hidden terminals, because RIMA and FAMA protocols provide correct collision-avoidance, which means that the relative performance differences among these protocols observed in the analysis apply also to networks with hidden terminals.

Our analysis results show that a small persistence time of only a fraction of the transmission time of a collision-avoidance control packet (RTR or RTS) suffices to provide much higher throughput at high to moderate average offered loads. The performance improvement observed

with limited persistence stems from reducing idle time in the channel due nodes backing off for large periods of time, and limiting the number of nodes that can contend for channel control after the channel becomes idle.

Chapter 4

Receiver-Initiated Channel-Hopping Protocols

Most of the MAC protocols to date have made the implicit assumption that there is a single-channel available and that only two nodes within range can exchange packets successfully at any given time. As the technology for radios that support multiple channels has become more economically feasible, a number of new MACs have been introduced to take advantage of more than one co-located physical channels. Even though code division multiple access techniques were proposed in the literature decades ago, it was not until a few years ago that the first commercial, multi-channel radios were introduced to the public. In 1993, the Telecommunications Industry Association (TIA) establish a standard under the name IS-95 for CDMA medium access techniques. The free access to certain ISM frequency bands was a good incentive for many companies to come up with low-cost, multi-channel radios that can reach speeds comparable with those of many wire-line LAN installations.

The need for collision-avoidance MAC protocols for single-channel networks to sense the channel as an integral part of the collision-avoidance handshake [36] limits their applicability.

Some commercial radios do not provide true carrier sensing, and direct sequence spread-spectrum (DSSS) radios may capture none or one of multiple overlapping transmissions in a non-deterministic manner, depending on the proximity and transmission power of the sources. Even if frequency-hopping spread-spectrum (FHSS) radios are used, carrier sensing adds to the complexity of the radio, which must already provide coarse time synchronization at the dwell-time level. On the other hand, using one or more busy tones to indicate when a receiver is busy [46] requires, in essence, a second transceiver, which is not economically attractive.

Section 4.1 introduces a novel suite of protocols that are based on a receiver-initiated channel-hopping operation that do not require code assignments or carrier sensing. RICH protocols require all nodes in a network to follow a common channel-hopping sequence, which is a requirement that can be easily met in practice. A channel can be defined to be a frequency channel, a spreading code, or a combination of these and other techniques (i.e. hybrid CDMA). However, with commercial radios operating in ISM bands, a channel should be viewed as a frequency channel or a hopping sequence. At any given time, all nodes that are not sending or receiving data listen on the common channel hop. To send data, nodes engage in a receiver-initiated dialogue over the frequency channel in which they are at the time they require to send data; those nodes that succeed in a collision-avoidance handshake remain in the same frequency channel for the duration of their data transfer, and the rest of the nodes continue to follow the common channel hopping sequence.

Section 4.2 proves that, in the absence of fading, RICH protocols solve the hidden-terminal problem, i.e., they eliminate collisions of data packets, without the need for carrier sensing or code assignments. As such, the RICH protocol family is the first approach reported to date that can accomplish correct collision avoidance without carrier sensing or predefined code assignments. Section 4.3 analyzes the throughput of the two RICH protocols: first we consider the case in which a single data packet is sent with every successful collision-avoidance handshake, and then we extend our analysis for the scenario in which two data packets can be exchanged between the sender and the receiver in a single control packet handshake. We compare both RICH protocols with

the MACA-CT protocol [52], which uses MACA collision-avoidance handshakes over a common channel and a transmitter-oriented data channel assigned to avoid collisions of data packets. We chose MACA-CT for our comparison, because it is the best representative of collision-avoidance solutions that eliminates the need for carrier sensing at the expense of requiring unique channel (code) assignments. Section 4.4 calculates the system delay for the RICH protocols. Section 4.5 presents a set of simulation experiments used to understand the performance of RICH protocols in realistic scenarios. Section 4.6 presents our conclusions.

4.1 Receiver-Initiated Channel-Hopping Collision Avoidance

4.1.1 Basic Concepts in Channel Hopping

The RICH protocols are based on three basic observations. First, reversing the collision-avoidance handshake (i.e., making the receiver in charge of avoiding collisions), improves the throughput of the network. Second, hidden-terminal interference can be eliminated by the assignment of channels or codes to senders or receivers in a way that no two senders or receivers share the same code if they are two hops away from one another. Third, with commercial frequency-hopping radios operating in ISM bands, radios have to synchronize in time so that all radios hop to different frequency channels at approximately the same time.

To eliminate hidden-terminal interference, RICH protocols exploit the fact that the nodes of a frequency-hopping network must agree on when to hop. A common frequency-hopping sequence is assumed by all the nodes (i.e., a common channel), so that nodes listen on the same channel at the same time, unless instructed otherwise. Nodes then carry out a receiver-initiated collision-avoidance handshake to determine which sender-receiver pair should remain in the present hop in order to exchange data, while all other nodes that are not engaged in data exchange continue hopping on the common hopping sequence. Because the collision-avoidance handshake ensures that the receiver of a successful handshake cannot receive packets that suffer from hidden-terminal

interference, and because all nodes not able to exchange data must hop to the next frequency channel, RICH protocols eliminate the need for carrier sensing and code assignment by simply allowing the sender and receiver of the handshake to remain on the same frequency channel in which they succeeded in their handshake.

The dwell time at a frequency channel in the RICH protocols need be only as long as it takes for a handshake to take place; as it will be clear, this time need only be long enough to transmit a pair of MAC addresses, a CRC, and framing. On the other hand, according to FCC regulations, a frequency-hop radio can remain in the same frequency for up to 400msec, which at a data rate of 1 Mbps is ample time to transmit entire data packets and packet trains. Hence, RICH protocols can be implemented by allowing a sender-receiver pair to communicate in the same frequency channel for a period of time that must be the smaller of 400msec and the time elapsed before the same frequency channel is used again in the common hopping sequence. Alternatively, a few orthogonal frequency-hopping sequences can be defined (e.g., 10, which is smaller than the number of simultaneous orthogonal frequency hops around a receiver in the 2.4 GHz band) for each frequency channel of the common hopping sequence.

4.1.2 A Protocol with Simple Polling

The first RICH protocol that we introduce is based on simple polling by the receiver and is called RICH-SP (RICH with Simple Polling). The idea of simple polling was first introduced in MACA-BI [92] for single-channel networks and modified in RIMA [42] for correct collision-avoidance over single-channel networks.

All the nodes follow a common channel-hopping sequence and each hop lasts the amount of time needed for nodes to receive a collision-avoidance control packet from a neighbor. A node attempts to poll its neighbors at a rate that is a function of the data rate with which it receives data to be sent, as well as the rate with which the node hears its neighbors send control and data packets. A node ready to poll any of its neighbors sends a ready-to-receive (RTR) control packet

over the current frequency channel specifying the address of the intended sender and the polling node's address. If the RTR is received successfully by the polled node, that node starts sending data to the polling node immediately and over the same frequency channel, and all other nodes hop to the next frequency channel. In practice, the dwell time in a frequency channel needs to be only long enough to allow an RTR to be received by a polled node. When the transmission of data is completed, then sender and receiver re-synchronize to the current frequency channel. If either multiple RTRs are sent during the same channel hop, or the polled node has no data to send to the polling node, the polling node does not receive any data a round-trip time after sending its RTR and must rejoin the rest of the network at the current channel hop. To permit the polling node to determine quickly that no data packet is to be expected, the polled node can transmit a short preamble packet in front of the data packet. To simplify our description, in the rest of this chapter we simply assume that a node is able to detect that no data packet is arriving.

Fig. 4.1 illustrates the operation of RICH-SP for the case in which sender-receiver pairs exchange data over a single frequency channel. In the figure, all the nodes start at time $t1$ from frequency $h1$. At time $t2$ the system is at frequency $h2$ and so on. At time $t1$ node x sends an RTR to node y and node y responds with data over the same channel. Notice that, there is a probability of $\frac{1}{N-1}$ that node y has data for x, where N is the number of nodes in the network. While x and y, stay in $h1$ until y has finished sending its data, all the other nodes hop to $h2$. At time $t2$ another node z sends an RTR to node w, but now it is the case that w does not have a data packet for z; therefore, w sends a CTS enabling z to send any data to w. At time $t4$ node z starts sending its data to w. Again, nodes z and w stay in $h2$ until z finishes sending its data, while the other nodes hop to $h3$. At time $t3$, node a sends an RTR to node b but node b is busy transmitting data to another node (uni-directional radios). Therefore, node b does not receive the RTR and at time $t4$ there is silence. In this case, node a continues to hop with the other nodes to frequency $h4$. At time $t4$ nodes c and d send an RTR and therefore a collision occurs. Both nodes have to back off and try to send an RTR at a later time.

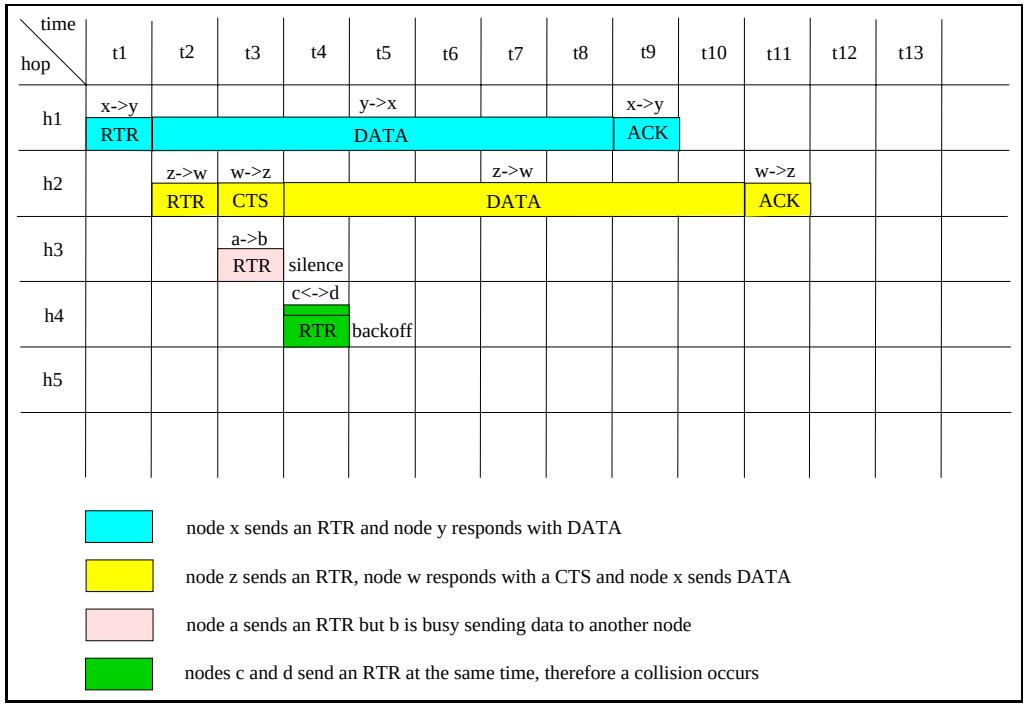


Figure 4.1: RICH-SP illustrated

Figure 4.2 shows the specification for RICH-SP in pseudo-code. After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time equal to a dwell time (time spent in one frequency channel). If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTR state. In the RTR state, the node sends an RTR packet with the destination address of the node that is the target destination, in this case z.

If node z receives the RTR correctly and has data for x, node z transitions to the XMIT state, where it transmits a data packet to x in the same frequency channel; otherwise, if node z cannot decode the RTR correctly, it perceives noise or silence, depending on the radio being used in that frequency and continues to hop with the rest of the nodes in the common hopping sequence. After sending its RTR, node x waits until the beginning of the next hop. At this time, if a preamble is not detected node x transitions to a new frequency channel according to the common hopping sequence; otherwise, x remains in the same frequency channel until (a) either a data packet arrives

```
Variable Definitions
    Global_Hop = A global counter that indicates the system-wide current
    frequency channel; at the beginning this variable is initialized to 0 that corresponds
    to the base frequency for the ISM band with no offset
    Node_Hop = A per-node counter that indicates the current frequency channel;
    when a node comes up this variable is initialized to the current value of the
    Global_Hop variable
    Timer = A global timer
    Freq_Num = A constant that represents the number of available frequencies
    channels in an ISM band
    Hop_Duration = The time spend in a single frequency
    PD = Packet Detected
    δ = Time to transmit a data packet


Procedure START()
Begin
    Read(Timer);
    Global_Hop ← Timer div Hop_Duration;
    Node_Hop ← Global_Hop;
    If (PD) Then
     Node_Hop ← Node_Hop + 1;
     call REMOTE(Timer);
    Else call PASSIVE(Timer);
End

Procedure PASSIVE(Tx)
Begin
    Node_Hop ← Global_Hop;
    If (Local Packet) Then
     call RTR();
    Else If (PD) Then
     Node_Hop ← Node_Hop +1;
     call REMOTE(Timer);
    Else
     Node_Hop ← Node_Hop +1;
End

Procedure RTR()
Begin
    Transmit an RTR Packet;
    Node_Hop ← Node_Hop +1;
    Do Case of (received packet type)
     RTS:
     call XMIT();
     NOISE:
     call BACKOFF();
End

Procedure XMIT()
Begin
    Transmit Data Packet;
    If (Local Packet) Then
     call BACKOFF();
    Else call PASSIVE(Timer);
End

Procedure BACKOFF()
Begin
    Timer ← Timer + RANDOM(1, 10 × Hop_Duration);
    If (PD) Then
     call REMOTE(Timer);
    Else call RTR();
End

Procedure REMOTE(Tx)
Begin
    Receive Packet;
    Do Case of (received packet type)
    Begin
     RTR:
     If (Destination ID = Local ID) Then
      Transmit RTS packet;
      call REMOTE(Timer);
     Else
     Node_Hop ← Node_Hop + 1;
     RTS:
     call XMIT();
     DATA:
     If (Destination ID = Local ID) Then
      pass packet to upper layer;
      call REMOTE(Timer);
     NOISE:
     call REMOTE();
    End
End
```

Figure 4.2: RICH-SP Specification

with the duration of it being part of its header, or (b) a Clear To Sent (CTS) packet arrives allowing x to sent a data packet at the same unique frequency channel.

When multiple RTRs are transmitted within a one-way propagation delay a collision takes place and the nodes involved have to transition to the BACKOFF state and try again at a later time chosen at random. After sending its RTR, node x waits for a response in the new frequency base. Node x determines that its RTR was not received correctly by z after a time period equal to one hop. If that is the case, node x will synchronize with the other nodes at a frequency channel that can be determined easily since node x is aware of the base frequency channel that the whole system is hopping at, from the initialization that took place at the beginning of the *hop cycle*.

To reduce the probability that the same nodes compete repeatedly for the same receiver at the time of the next RTR, the RTR specifies a back-off-period unit for contention. The nodes that must enter the BACKOFF state compute a random time that is a multiple of the back-off-

period unit advertised in the RTR. The simplest case consists of computing a random number of back-off-period units using a uniformly distributed random variable from 1 to d, where d is the maximum number of neighbors for a receiver. The simplest back-off-period unit is the time it takes to send a small data packet successfully.

4.1.3 A Protocol with Dual-Use Polling

The second RICH protocol that we introduce is based on dual-use polling and is called RICH-DP (RICH with Dual-use Polling). In this case the handshake originally presented in chapter 2 for RIMA-DP is extended and applied to multi-channel radio networks.

Fig. 4.3 illustrates the operation of RICH-DP for the case in which sender-receiver pairs exchange data over a single frequency channel. In the figure, all the nodes start at time $t1$ from frequency $h1$. At time $t2$ the system is at frequency $h2$ and so on. At time $t1$ node x sends an RTR to node y and node y responds with data over the same channel. Notice that, there is a probability of $\frac{1}{N-1}$ were N is the number of nodes in the network, that node y has data for x. At time $t9$ node y receives an ACK from node x and at time $t10$ node x is now enabled to transmit its own data packet to the polled node y. While x and y, stay in $h1$ until y has finished sending its data, all the other nodes hop to $h2$. At time $t2$ another node z sends an RTR to node w, but now it is the case that w does not have a data packet for z; therefore, w sends a CTS enabling z to send any data to w. At time $t4$ node z starts sending its data to w. Again, nodes z and w stay in $h2$ until z finishes sending its data, while the other nodes hop to $h3$. At time $t3$, node a sends an RTR to node b but node b is busy transmitting data to another node (uni-directional radios). Therefore, node b does not receive the RTR and at time $t4$ there is silence. In this case, node a continues to hop with the other nodes to frequency $h4$. At time $t4$ nodes c and d send an RTR and therefore a collision occurs. Both nodes have to back off and try to send an RTR at a later time. Notice that a successfully received data packet is always followed by an acknowledgment (ACK) from the destination node to the source node.

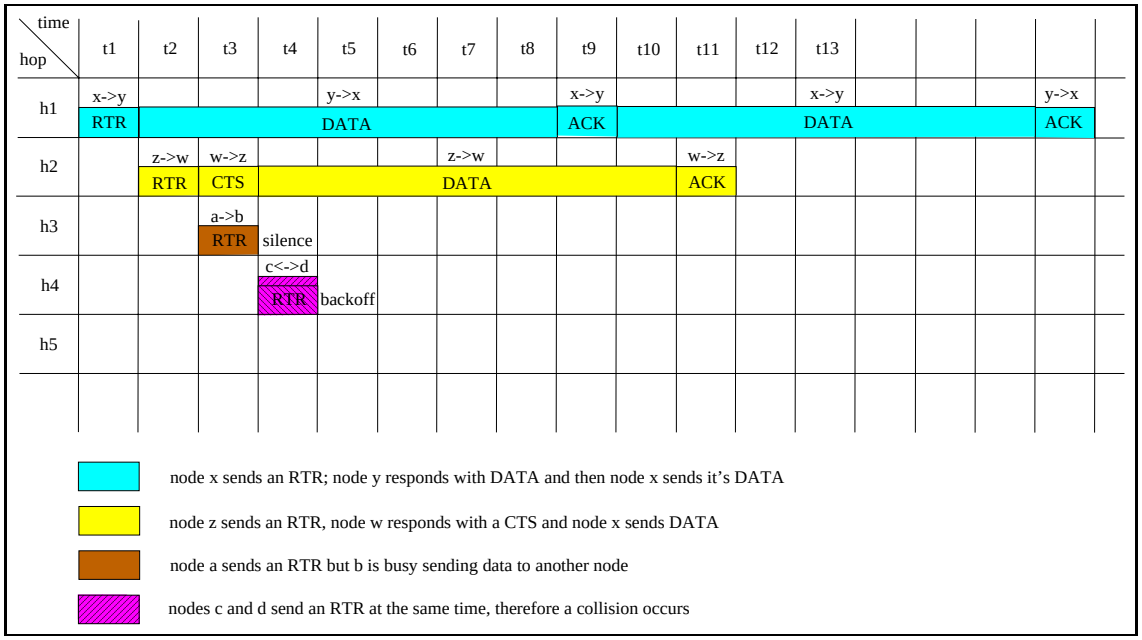


Figure 4.3: RICH-DP illustrated

After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time equal to a dwell time (time spent in one frequency channel). If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor y, it transitions to the RTR state. In the RTR state, the node sends an RTR packet with the destination address of the node that is the target destination, in this case y.

If node y receives the RTR correctly and has data for x, node y transitions to the XMIT state, where it transmits a data packet to x in the same frequency channel; otherwise, if node y cannot decode the RTR correctly, it perceives activity or silence, depending on the radio being used in that frequency and continues to hop with the rest of the nodes in the common hopping sequence. After sending its RTR, node x waits until the beginning of the next hop. At this time, if a preamble is not detected node x transitions to a new frequency channel according to the common hopping sequence; otherwise, x remains in the same frequency channel until (a) either a data packet arrives with the duration of it being part of its header, or (b) a Clear To Sent (CTS) packet arrives

allowing x to sent a data packet at the same unique frequency channel.

Although the 400msec allowed per dwell time by the FCC is a long time to transmit data in ISM bands, it may be desirable to allow nodes sending data to hop over multiple frequency channels, because staying at the same frequency channel for a long period of time does not take advantage of many inherent advantages that come with frequency hopping. For example, frequency channel can continue to work efficiently even in the presence of narrow-band jamming, is resilient against fading and erasures, and minimizes the multi-path propagation problem. However, in order to realize these benefits, the rate with which the nodes in the network hop from one frequency to another should not be below a certain threshold.

In order to keep all the advantages of a frequency hopping modulation while avoiding the need for code assignments, a receiving frequency-hopping sequence must be defined that is guaranteed to be free of interference from other data transmissions for at least a few dwell times, which could be up to the time when the same frequency channel occurs in the common hopping sequence used for handshakes. The key difference here is that in the CTS sent back to the source from the destination, the base frequency of the destination (which is part of the CTS) is used by the sender to discover the unique hopping pattern to be used to exchange data. Notice that sending data in this way requires packet trains consisting of packets with lengths equal to the size that can be accommodated in a single hop. Clearly, there is correlation between a non-perfect channel (due to fading, multi-path propagation, cross-channel interference) and the selected frequency hopping pattern but this is beyond the scope of this dissertation.

When multiple RTRs are transmitted within a one-way propagation delay a collision takes place and the nodes involved have to transition to the BACKOFF state and try again at a later time chosen at random. After sending its RTR, node x waits for a response in the new frequency base. Node x determines that its RTR was not received correctly by z after a time period equal to one hop. If that is the case, node x will synchronize with the other nodes at a frequency channel that can be determined easily since node x is aware of the base frequency channel that the whole

system is hopping at, from the initialization that took place at the beginning of the *hop cycle*.

To reduce the probability that the same nodes compete repeatedly for the same receiver at the time of the next RTR, the RTR specifies a back-off-period unit for contention. The nodes that must enter the BACKOFF state compute a random time that is a multiple of the back-off-period unit advertised in the RTR. The simplest case consists of computing a random number of back-off-period units using a uniformly distributed random variable from 1 to d, where d is the maximum number of neighbors for a receiver. The simplest back-off-period unit is the time it takes to send a small data packet successfully.

4.2 Correct Collision Avoidance in RICH Protocols

Theorems 10 and 11 below show that RICH-SP and RICH-DP protocols ensure that there are no collisions between data packets and any other transmissions under the following assumptions [36]:

A0) A node transmits an RTR that does not collide with any other transmissions with a non-zero probability.

A1) The maximum end-to-end propagation time in the channel is $\tau < \infty$.

A2) A packet sent over the channel that does not collide with other transmissions is delivered error free with a non-zero probability.

A3) All nodes execute a given RICH protocol correctly.

A4) The transmission time of an RTR and a CTS is γ, the transmission time of a data packet is δ, and the hardware transmit-to-receive transition time is zero; furthermore, $2\tau < \gamma \leq \delta < \infty$.

A5) The dwell time in each frequency is equal to the time needed to transmit an RTR (or CTS) plus the maximum end-to-end propagation time.

A6) There is no capture, erasure, or fading in the channel.

A7) Any overlap of packet transmissions at a particular receiver, causes that receiver to not understand any of the packets.

A8) Any frequency hopping pattern depends solely in the base frequency channel used and the probability that two or more distinct hopping sequences will collide is zero. For simplicity we assume that data packets are exchanged over a single frequency channel, rather than over a hopping sequence.

With the commercially available spread spectrum radios today, periods of deep fading (*erasures*) disrupt any type of collision avoidance dialogue, i.e., data packets may experience collisions in the presence of fading. However, with frequency hopping radios, the higher the rate with which a radio hops from one frequency to another the less the probability that an erasure will occur. Even though fast frequency hopping would be ideal to avoid erasures, it is not commercially available. However, because the dwell time used in RICH protocols needs to include only two MAC addresses, a CRC, and framing bits, the effect of erasures should be negligible.

Assuming zero processing and turn-around delays is done for convenience; however, the same type of proofs, with adjusted parameters, apply for non-zero hardware delays.

The approach used to show that a collision-avoidance protocol works correctly, i.e., that it prevents data packets from colliding with any type of packets, consists of showing that, once a data packet is sent by a node, the intended receiver obtains the packet without interference from any other source. The intuition suggesting why this is possible is shown in Fig. 4.4, which illustrates that pairs of nodes can exchange data over a given frequency h_i while the other nodes move on with the common hopping sequence or are exchanging data over a different hop.

Theorem 10 *RICH-SP provides correct collision-avoidance in the presence of hidden terminals when the time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence.*

Proof: Consider a polling node A and a polled node X and assume that A sends an RTR at time t_0. After sending its RTR, node A remains in frequency channel H for a period of time that is long enough to detect a CTS or the presence or absence of a data packet. We denote by h the dwell time in a particular hop. If X does not receive the RTR correctly due to interference from any neighbor hidden from A, it does not send any data. Else, X receives A's RTR at time $t_1 = t_0 + h$ and remains in the same frequency channel H where the RTR was received. At time $t_1' > t_0 + h$, if node X has a local data packet for A, then it starts sending its data to A; otherwise, X sends a CTS to A enabling A to send its data packet. Both nodes A and X remain in frequency channel H, that never collides with the common hopping sequence since we made the assumption that the time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence (Fig. 4.4). *Q.E.D.*

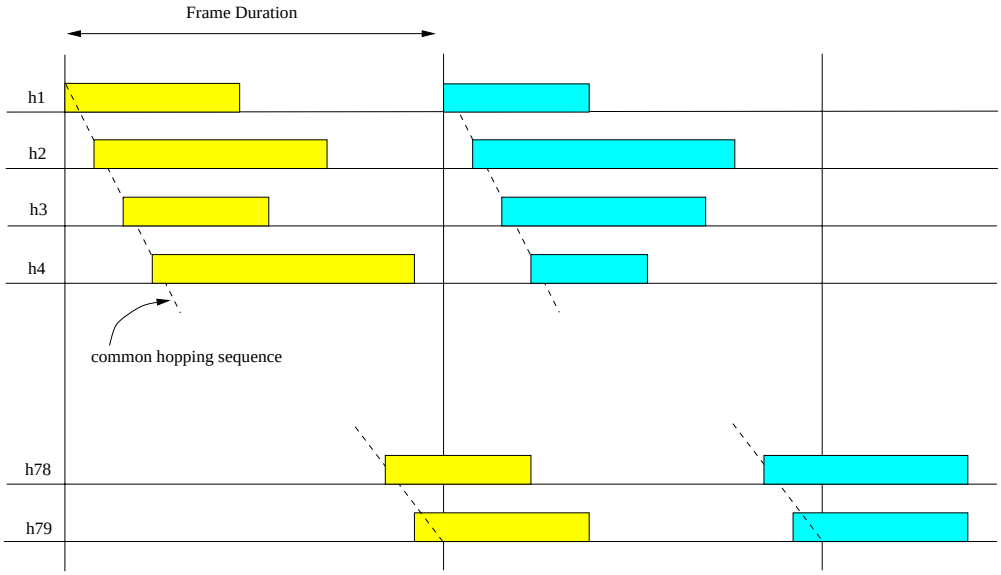


Figure 4.4: RICH provides correct collision-avoidance since there are no conflicts between the common frequency hopping sequence and ongoing DATA packet transmissions

Theorem 11 *RICH-DP provides correct collision-avoidance in the presence of hidden terminals when the time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence.*

Proof: Consider a polling node A and a polled node X and assume that A sends an RTR at time t_0. After sending its RTR, node A remains in frequency channel H for a period of time that is long enough to detect a CTS or the presence or absence of a data packet. We denote by h the dwell time in a particular hop. If X does not receive the RTR correctly due to interference from any neighbor hidden from A, it does not send any data. Else, X receives A's RTR at time $t_1 = t_0 + h$ and remains in the same frequency channel H where the RTR was received. At time $t_1' > t_0 + h$, if node X has a local data packet for A, then it starts sending its data to A; otherwise, X sends a CTS to A enabling A to send its data packet. Both nodes A and X remain in frequency channel H, that never collides with the common hopping sequence since we made the assumption that the time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence (Fig. 4.4). *Q.E.D.*

4.3 Approximate Throughput Analysis

The objective of our analysis is to calculate the throughput achieved with RICH-SP and RICH-DP and compare our results against sender -initiated CDMA protocols, i.e., MACA-CT [52]. The choice of MACA-CT was made because we want to show how RICH-DP performs against the best performing CDMA protocol reported to date for ad hoc networks in which receivers can detect at most one transmissions at a time. Our analysis shows a number of interesting results. By making collision-avoidance a joint effort by sender and receiver, a much better performance is obtained than what can be achieved with MACA-CT; this should be expected, because the vulnerability period with both RICH protocols is half the one with MACA-CT, and dual-use polling doubles the opportunity for collision-free data to be exchanged over a single control packet handshake.

4.3.1 Assumptions

We analyze the throughput of RICH-SP using the model first introduced by Sousa and Silvester [88] for CDMA protocols. We extend the analytical framework used to date to evaluate the throughput of RICH-DP. We calculate the throughput and average delay for RICH-SP and RICH-DP with a discrete-time Markov chain. The following assumptions are made:

1. There are N nodes in the fully-connected network.

2. A single unslotted channel is used for all packets, and the channel introduces no errors (no capture or fading).

3. At any given time slot, at most one RTR can be successfully transmitted.

4. Since there is an upper limit in the number of transmissions that can co-exist at the same time in an ISM radio band when using FHSS, we can have up to m pairs of nodes that exchange data at the same time.

5. The data packet length distribution is geometrically distributed with parameter q; therefore, the probability of a data packet with length l is, $P[L = l] = (1 - q)q^{l-1}$ and the average packet length, measured in mini-packets per slot is, $\overline{L} = \frac{1}{1-q}$.

6. The size for an RTR and a CTS plus a maximum end-to-end propagation is equal to h, where h is the dwell time in a particular hop; the size for a data packet is always a multiple of h.

A polled node has a packet addressed to the polling node with probability $\frac{1}{N-1}$ (i.e. uniform distribution). Furthermore, we assume that each node sends its RTR according to a Poisson distribution with a mean rate of $\frac{\lambda}{N-1}$, and that (when applicable) the polling node chooses the recipient of the RTR with equal probability.

4.3.2 Receiver-Initiated Channel-Hopping protocols

To make a fair comparison with MACA-CT, we use the same average packet length, $\overline{L}$, for both protocols. However, since in MACA-CT a slot is equal to the size of an RTS plus a CTS plus the corresponding propagation time needed, the duration of a slot size, h, for the RICH protocols is equal to half the size of the slots used in MACA-CT. Consequently, the average packet length for MACA-CT will be equal to $\frac{1}{2(1-q)}$.

At any given slot, a node can be: (a) idle, (b) transmitting an RTR or a CTS control signal, and (c) sending a series of consecutive (in time) slots with segments of the data packet. The possible scenarios that can occur in the RICH protocols are:

- node x sends an RTR to node y and y sends its data packet to x with probability $\frac{1}{N-1}$
- node x sends an RTR to node y but y does not have any data for x, therefore y sends a CTS to x and x sends its data to y
- node x sends an RTR at the same time that node y sends an RTR, therefore a collision occurs
- node x sends an RTR but node y is already tuned in a different hopping pattern, therefore node x does not hear anything in the next hop

Notice that for RICH-DP only after the end of a data packet another data packet might follow immediately without any additional control packets. This scenario can be treated as a single data packet exchange where two parameters are treated differently from the case of a single data packet transmission: (a) the probability that such an event occurs (two data packets can be exchanged) depends on the probability that the polled node has a data packet for the polling node, and (b) the aggregate length of the transmission depends on the length of the two data packets.

At any given time the system state can be described by the number of communicating pairs of nodes (Fig. 4.5). Because all the nodes that transmit an RTR that is not received at time slot $t-1$ are available at slot t, the system state at any given time slot t is independent from

the number of nodes that transmit an unanswered RTR. Accordingly, we need to calculate the transition probabilities of this Markov chain under the assumptions presented above. A transition in the Markov chain from one state to another occurs when: (a) at least one member from the set of nodes exchanging data packets, finish transmitting data, and (b) the nodes that participate in the handshake either succeed or fail transmitting an RTR. To calculate the transition probability from the current state we need to know the number of nodes that will finish transmitting data and the number of nodes that succeed or fail transmitting an RTR.

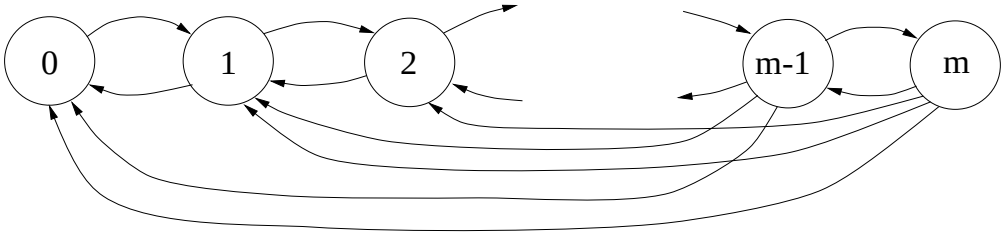


Figure 4.5: Markov Chain defining the average number of communicating pairs

We will use $B(n,p,k)$ in the following to represent a geometric distribution; that is:

$$B(n,p,k) = \left(\begin{array}{c} n \\ \\ k \end{array} \right) p^k (1-p)^{n-k} \tag{4.1}$$

Let $P_{k,l}$ be the transition probability in the Markov chain from state k (where k pairs of nodes exchange data) in slot $t-1$, to state l (where l pairs of nodes exchange data) in slot t. We condition on the number i of communicating pairs of nodes that finish sending or receiving data packets at the beginning of slot t. The system is at state l in time slot $t-1$ and therefore the number of nodes that are available to receive or transmit is equal to $N' = N - 2(l-i)$. If the transition to state l is made, then let x' be the number of nodes which transmit an RTR at the beginning of time slot t. Furthermore, $l' = l - (k-i)$ pairs of nodes will become busy exchanging data packets and $n' = x' - l'$ nodes will transmit an RTR packet that will not be received. Due to the assumption that only one RTR can be successful at a given time slot, a transition from state k to state l is possible only if $l' = 1$ or $l' = 0$.

We denote with Φ the event that a transition from k to l occurs, with ΦI being the event that exactly one transmission occurs and it is addressed to an idle node, and with ΦB being the event that exactly one transmission occurs and it is addressed to a busy terminal. Then, the transition probabilities can be calculated as follows [88]

$$\begin{aligned} p_{k,l} &= \sum_{i=0}^{k} \left(\begin{array}{c} i \text{ pairs} \\ \\ \text{become idle} \end{array} \right) \cdot [P[\Phi \cap \Phi I] + P[\Phi \cap \Phi B] + P[\Phi \cap (0 \text{ or } > 1 \text{ transmissions})]] \\ &= \sum_{i=0}^{k} B(k, 1-q, i) \cdot [\delta(m'-1)\delta(n')B(N', p, 1) \left(\frac{N'-1}{N-1} \right) \\ &+ \delta(m')\delta(n'-1)B(N', p, 1) \left(\frac{N-N'}{N-1} \right) + \delta(m')(1-\delta(n'-1))B(N', p, n')] \end{aligned} \tag{4.2}$$

where $B(n, p, k)$ is given from Eq. 4.1, $\delta(0) = 1$, and $\delta(x) = 0$ if $x \neq 0$.

4.3.3 RICH-SP

From Eq. 4.2 we can easily derive the following equation for RICH-SP:

$$\begin{aligned} p_{k,l} = q^{l-1}(1-q)^{k-1} \quad & \left\{ \left(\begin{array}{c} k \\ \\ l-1 \end{array} \right) (1-q)p(1-p)^{M+1} \frac{M^2+3M+2}{N-1} \right. \\ & \left. - \left(\begin{array}{c} k \\ \\ l \end{array} \right) qp(1-p)^{M-1} \frac{M^2-M}{N-1} + \left(\begin{array}{c} k \\ \\ l \end{array} \right) q \right\} \end{aligned} \tag{4.3}$$

where $M = N - 2l$. To calculate the average throughput we need to know the steady-state probabilities that correspond to each one of the states of the Markov chain (Fig. 4.5). From the transition probability equation, we can solve a linear system of equations with as many unknowns as the number of states in the Markov chain to calculate the steady-state probabilities. If PS_l is the steady state probability for state l, then the average throughput S is equal to the number of data packets transmitted at the same frequency channel; that is

$$S = \sum_{l=0}^{\frac{N}{2}} l \cdot PS_l \tag{4.4}$$

Figure 4.6 shows the throughput achieved by RICH-SP and MACA-CT versus the probability of transmission p for various numbers of nodes in the network. Because the slot duration in RICH-SP is half the one in MACA-CT, the probability of transmission at a given slot is $\frac{p}{2}$. The maximum throughput of RICH-SP is always higher than MACA-CT because the duration for the exchange of the control signals is half the size of the one used in MACA-CT and consequently the vulnerability period in RICH-SP is half the time spent in MACA-CT. Since no data will be ever send with RICH-SP to a busy terminal, nodes in RICH-SP are immediately available to try again, something that is not the case in C-T [88]. Therefore, at any given time slot, the number of nodes available to transmit an RTR is maximized while the contention period is minimized!

Figure 4.7 shows the throughput against the probability of transmission p for a fixed number of nodes ($N = 12$) with the average packet length L being the parameter. As it is obvious, RICH-SP has a higher throughput than MACA-CT regardless of the size of the data packet. The general conclusion that can be drawn in this case is that, higher throughput can be achieved with a longer average packet length. However, notice that we have made the assumption of a perfect channel. In a realistic environment, by increasing the length of the transmitted packet we also increase the probability that errors will occur. Furthermore, when the number of co-located nodes is high, the interference from adjacent frequency channels is more likely to introduce errors in the transmission of data packets. It has been shown [44] that there is no improvement in the throughput achieved by increasing the length of the data packet after a certain threshold in a non-perfect channel for other spread spectrum protocols. The same should be expected for RICH-SP.

4.3.4 RICH-DP

We can calculate the transition probabilities for RICH-DP from Eq. 4.2 in a similar way as follows

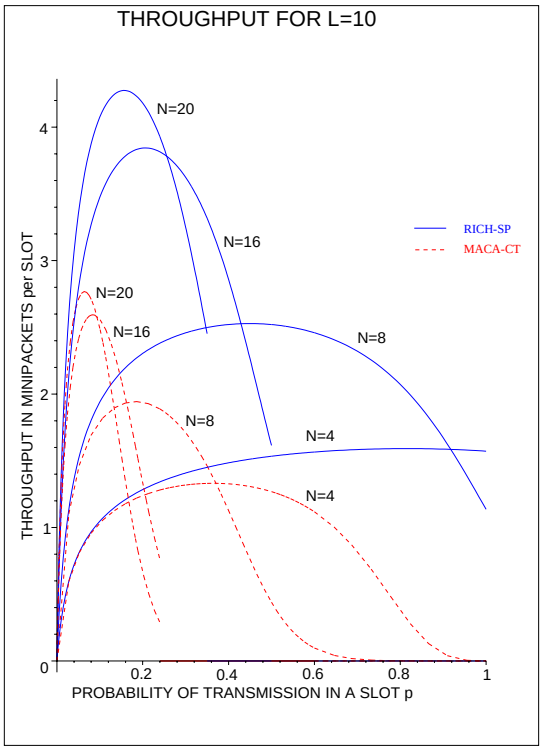


Figure 4.6: Throughput versus transmission probability for MACA-CT and RICH-SP for a fixed average packet length $\overline{L} = 10$

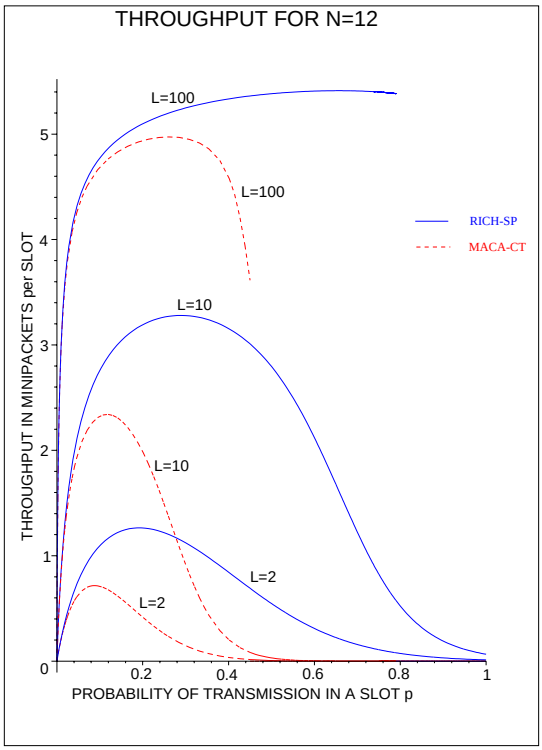


Figure 4.7: Throughput versus transmission probability for MACA-CT and RICH-SP for a fixed number of nodes $N = 12$

$$
\begin{aligned}
p_{k,l} \quad = \quad & \sum_{i=0}^{k} \begin{pmatrix} i \text{ pairs} \\ \\ \text{idle} \end{pmatrix} \cdot \{P[\Phi \cap \Phi I] + P[\Phi \cap \Phi B] + P[\Phi \cap (0 \text{ or } > 1 \text{ transmission})]\} = \\
& \sum_{i=0}^{k} \begin{pmatrix} i \text{ pairs} \\ \\ \text{idle} \end{pmatrix} \cdot \Bigg\{ \delta(m'-1)\delta(n')B(N',p,1) \left(\frac{N'-1}{N-1}\right) \\
& \qquad + \delta(l')\delta(n'-1)B(N',p,1) \left(\frac{N-N'}{N-1}\right) \\
& \qquad + \delta(l')(1-\delta(n'-1))B(N',p,n') \Bigg\} \qquad (4.5)
\end{aligned}
$$

where $\delta(0) = 1$ and $\delta(x) = 0$ if $x \neq 0$.

The number i of pairs of nodes that become idle at any given time slot t, is dependent of the number of nodes that are exchanging only one data packet, as well as the number of nodes

that are exchanging two data packets. Because all the nodes are independent sources of packets with identical geometrically distributed packet lengths, the length of data transmitted is equal to a negative binomial distribution when two nodes exchange data packets at the same busy period. The probability that a data packet has length l is equal to $P[L = l] = (1 - q)q^{l-1}$. If two data packets are to be sent, then the average length will be $\overline{L} = \frac{2}{1-q}$. We denote with q' the parameter of the binomial distribution when two packets are transmitted.

To calculate the probability that i pairs of nodes become idle in a given time slot t, we assume that out of those i pairs that become idle, x are idle after exchanging only one data packet and the remaining $i - x$ pairs are idle after exchanging two data packets. Figure 4.8 shows the sets of nodes at the beginning of every time slot. Notice that, set A contains pairs of nodes that exchange only one data packet, and pairs of nodes that exchange two data packets.

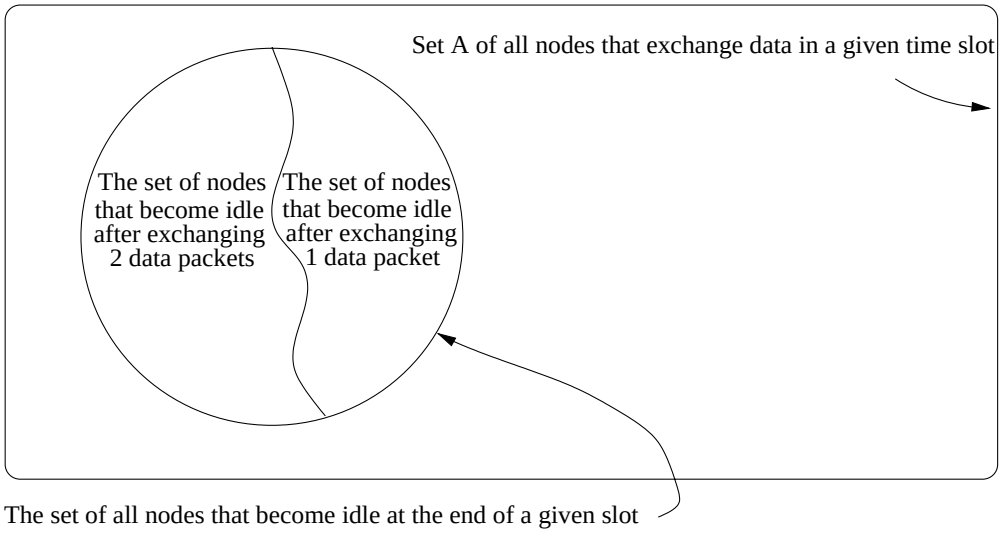


Figure 4.8: The various sets of nodes when RICH-DP is deployed

Because we have mutually exclusive events, we can make use of the multinomial probability law. If B_1, B_2, B_3 are the three partitions of the sample space, then let q be the probability that a pair of nodes becomes idle after exchanging one data packet, and q' be the probability that a pair of nodes becomes idle after exchanging two data packets. Then,

$$Pr\left(\begin{array}{c} i \text{ pairs} \\ \\ \text{become idle} \end{array}\right) = \frac{k!}{x!(i-x)!(k-i)!}\left(\frac{x}{i}q\right)^x\left(\frac{i-x}{i}q'\right)^{i-x}\left(1-\frac{xq+(i-x)q'}{i}\right)^{k-i} \quad (4.6)$$

Substituting Eq. 4.6 into Eq. 4.5 we have

$$\begin{aligned} p_{k,l} = & \sum_{x=0,k-l\neq\{0,-1\}}^{x=k-l+1}\left\{\frac{k!}{x!(k-l+1-x)!(l-1)!}\left(\frac{x}{k-l+1}q\right)^x\left(\frac{k-l+1-x}{k-l+1}q'\right)^{k-l+1-x}\right. \\ & \left.\left(1-\frac{xq+(k-l+1-x)q'}{k-l+1}\right)^{l-1}B(N',p,1)\frac{N'-1}{N-1}\right\}+ \\ & \sum_{x=0,k-l\neq\{0,-1\}}^{x=k-l}\left\{\frac{k!}{x!(k-l-x)!(l)!}\left(\frac{x}{k-l}q\right)^x\left(\frac{k-l-x}{k-l+1}q'\right)^{k-l-x}\right. \\ & \left.\left(1-\frac{xq+(k-l-x)q'}{k-l}\right)^{l}B(N',p,1)\frac{N-N'}{N-1}\right\}+ \\ & \sum_{x=0,k-l\neq\{0,-1\}}^{x=k-l}\left\{\frac{k!}{x!(k-l-x)!(l)!}\left(\frac{x}{k-l}q\right)^x\left(\frac{k-l-x}{k-l+1}q'\right)^{k-l-x}\right. \\ & \left.\left(1-\frac{xq+(k-l-x)q'}{k-l}\right)^{l}(1-B(N',p,1))\right\} \end{aligned} \quad (4.7)$$

To calculate the average throughput we need to know the steady-state probabilities that correspond to each one of the states of the Markov chain (Fig. 4.5). Given the transition probabilities (Eq. 4.7), we can solve a linear system of equations with as many unknowns as the number of states in the Markov chain to calculate the steady-state probabilities. If PS_l is the steady state probability for state l, then the average throughput S is equal to the number of data packets transmitted at the same frequency channel; that is

$$S = \sum_{l=0}^{\frac{N}{2}} l \cdot PS_l \quad (4.8)$$

Figure 4.9 shows the throughput achieved by RICH-DP and MACA-CT versus the probability of transmission p for various numbers of nodes in the network. Just as with RICH-SP, the slot duration in RICH-DP is half the one in MACA-CT and therefore the probability of transmission

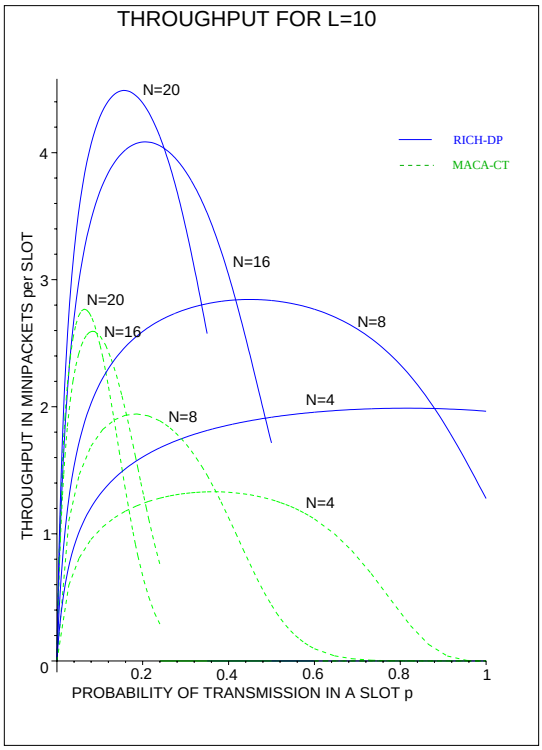


Figure 4.9: Throughput versus transmission probability for MACA-CT and RICH-DP for a fixed average packet length $\overline{L} = 10$

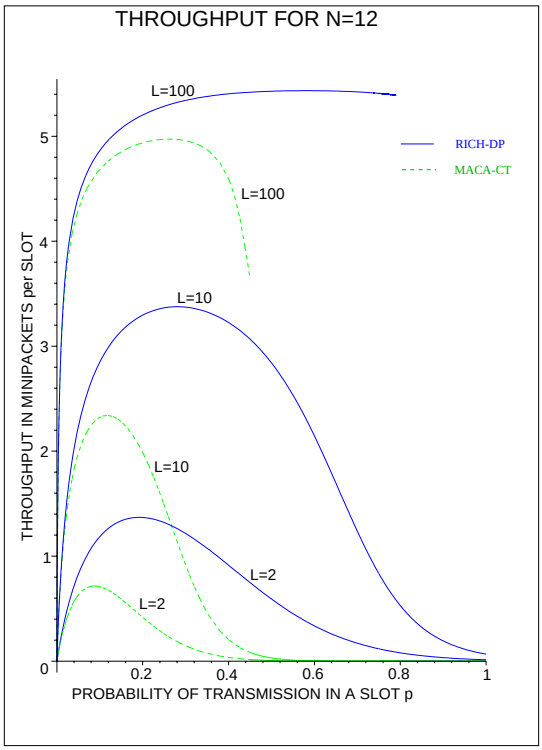


Figure 4.10: Throughput versus transmission probability for MACA-CT and RICH-DP for a fixed number of nodes $N = 12$

at a given slot is $\frac{p}{2}$. The maximum throughput of RICH-DP is always higher than MACA-CT because of two factors: (a) the vulnerability period in RICH-DP is half the one in MACA-CT, and (b) the polling and polled nodes can engage in exchange of two data packets over the course of a single control packet handshake.

Figure 4.10 shows the throughput against the probability of transmission p for a fixed number of nodes ($N = 12$) with the average packet length L being the parameter. Clearly, RICH-DP has similar behavior with RICH-SP demonstrating higher throughput than MACA-CT regardless of the size of the data packet. The longer the average length of the data packet, the higher the effective throughput that can be achieved.

Due to the fact that with RICH-DP two data packets can be exchanged over a single handshake, under certain network conditions we expect the performance to be better with RICH-

DP rather than RICH-SP. Figure 4.11 verifies our conjecture for the case of a fixed average packet length with two ad-hoc networks of 8 and 12 nodes respectively.

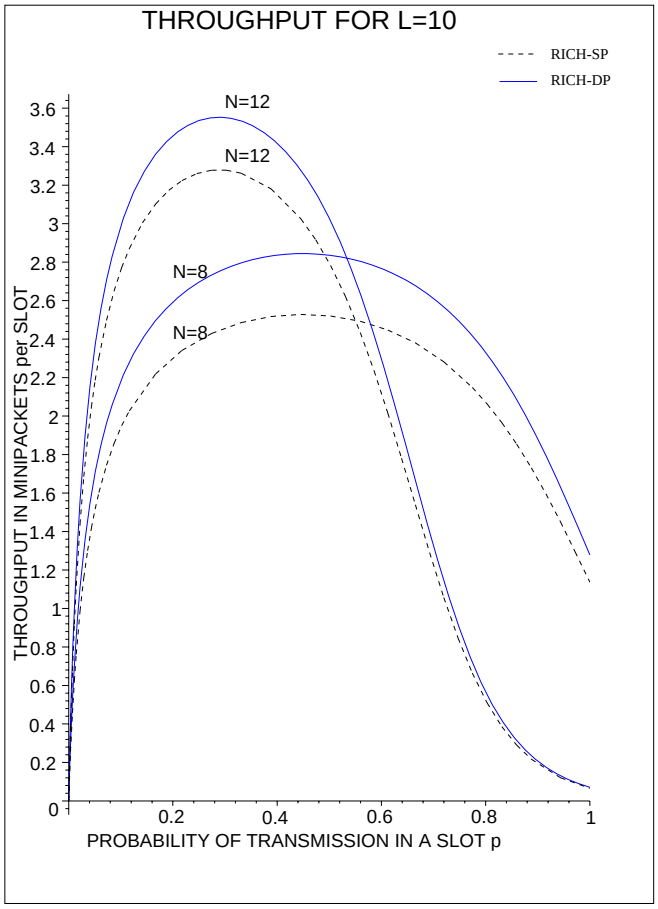


Figure 4.11: Throughput versus transmission probability for RICH-SP and RICH-DP for a fixed average packet length $\overline{L} = 10$

4.4 Delay Analysis

To calculate the average delay for the RICH protocols we need to first define a retransmission policy. We assume that the arrival process is Bernoulli with probability p for every node. Because we have a queue of maximum size equal to one packet, if a packet is waiting in the queue then there are no further new packet arrivals, and the waiting packet is retransmitted in the next slot with probability p. If a node has a packet waiting to be sent, but a packet from some other

user is received, then the waiting packet is discarded and when the handshake is completed the given node becomes idle and generates a new packet with probability p. All the assumptions that were presented in section 4.3 are valid in the following derivation as well.

We use Little's theorem to calculate the average delay. We define the system delay D as the time that it takes for a new arriving packet that is waiting in the queue to be transmitted and successfully received by the intended receiver. If $\overline{m}$ is the average number of pairs of nodes that simultaneously exchange data packets, and $\overline{B}$ is the average number of blocked users (due to collision of RTSs or RTSs that are not received), then at any given time the average number of packets in the system will be equal to $\overline{m} + \overline{B}$. We can calculate $\overline{m}$ and $\overline{B}$ as follows

$$\overline{m} = \sum_{m=0}^{\lfloor \frac{N}{2} \rfloor} m P_m \tag{4.9}$$

$$\overline{B} = \sum_{m=0}^{\lfloor \frac{N}{2} \rfloor} p(N-2m) \left(1 - \frac{N-m-1}{N-1}\right) P_m \tag{4.10}$$

The average delay normalized to a packet length is derived by applying Little's theorem as follows

$$\overline{D} = \frac{\overline{m} + \overline{B}}{S} \tag{4.11}$$

Since the mean transmission time for a packet is equal to $\frac{1}{1-q}$ the actual system delay should include the transmission time for the data packet. That is

$$D = \frac{\overline{D}}{(1-q)} \tag{4.12}$$

In Figure 4.12 we can see the numerical results obtained for the normalized delay performance of MACA-CT and RICH-SP. It is clear that RICH-SP offers the smallest delay at any load. Furthermore, the system delay with RICH-SP remains almost the same up to $p > 0.6$ whereas with MACA-CT the delay increases exponentially when $p > 0.4$. This is to be expected, because collisions between control packets increase as the offered load increases, and minimizing the length

of the collision-avoidance handshakes that are susceptible to collisions becomes crucial. Indeed, with RICH-SP, only RTRs can collide and therefore their vulnerability periods are half the vulnerability period in MACA-CT. It is obvious from the same figure the normalized delay can be reduced noticeably by increasing the packet length.

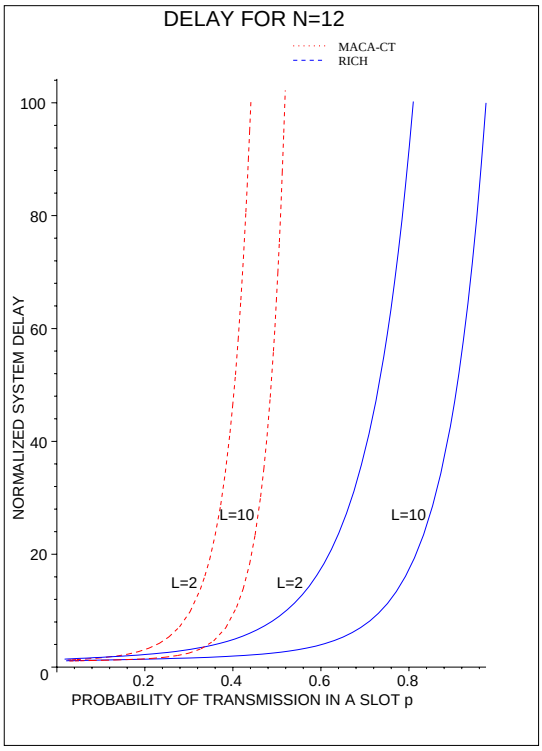


Figure 4.12: Normalized system delay versus transmission probability for MACA-CT and RICH-SP for a fixed number of nodes $N = 12$

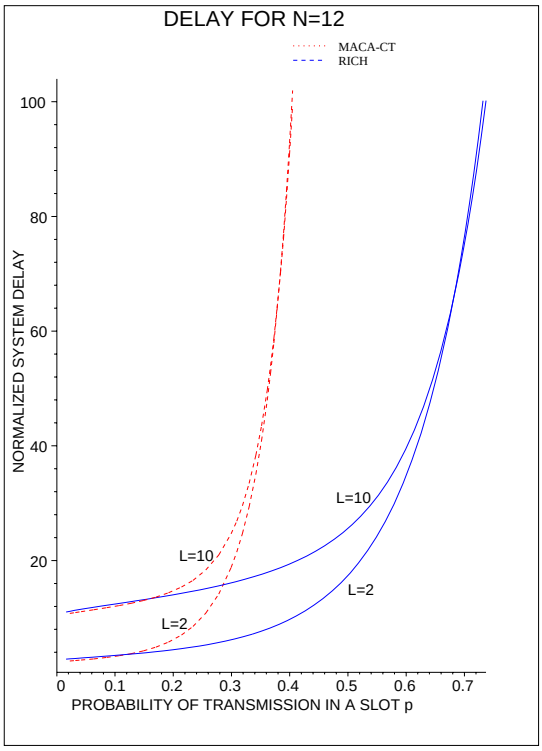


Figure 4.13: Actual system delay versus transmission probability for MACA-CT and RICH-SP for a fixed number of nodes $N = 12$

In Figure 4.13 the actual system delay that includes the packet transmission time for MACA-CT and RICH-SP is shown. In this figure, contrary to what happened with the normalized system delay, we notice that by increasing the packet length we do not achieve smaller delays. However, this is to be expected since the transmission time is the dominating delay in this case.

In Figures 4.14 and 4.15 we can see the corresponding numerical results obtained for the normalized and actual delay performance of MACA-CT and RICH-DP. Clearly the same conclusions described previously for RICH-SP apply in this case as well. Moreover, with RICH-DP the system

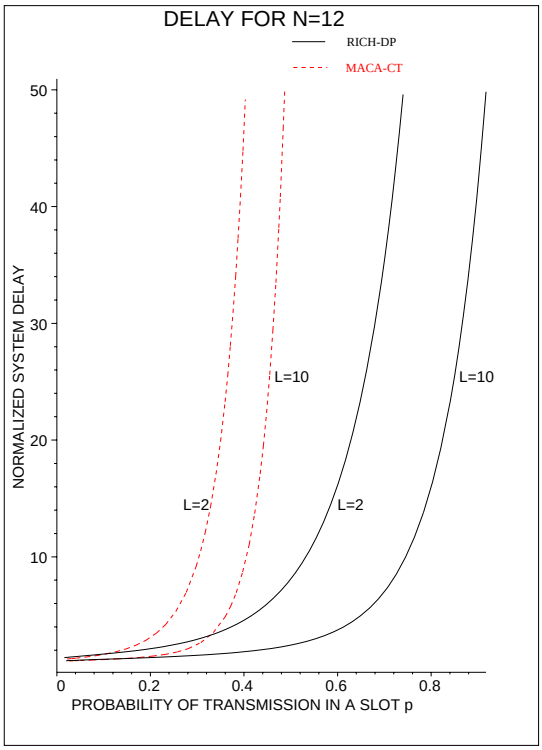


Figure 4.14: Normalized system delay versus transmission probability for MACA-CT and RICH-DP for a fixed number of nodes $N = 12$

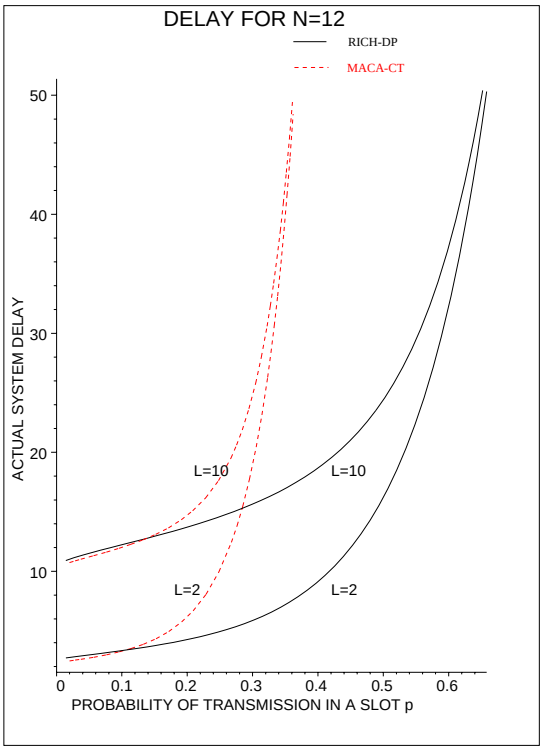


Figure 4.15: Actual system delay versus transmission probability for MACA-CT and RICH-DP for a fixed number of nodes $N = 12$

delay (normalized or actual) is always less than the one with RICH-SP as can be seen in Figures 4.16 and 4.17.

4.5 Simulation Results

We validated our analytical results by performing a number of simulation experiments. Our goal was to investigate the performance of the RICH protocols under different network topologies and to show how the results compare against the analytical results presented previously. Since we have shown analytically that both RICH protocols exhibit the same behavior we only consider RICH-DP in our simulation experiments. We used the OPNET simulation tool to implement MACA-CT and RICH-DP.

For the simulation experiments, we used a multiple-channel capable radio that approxi-

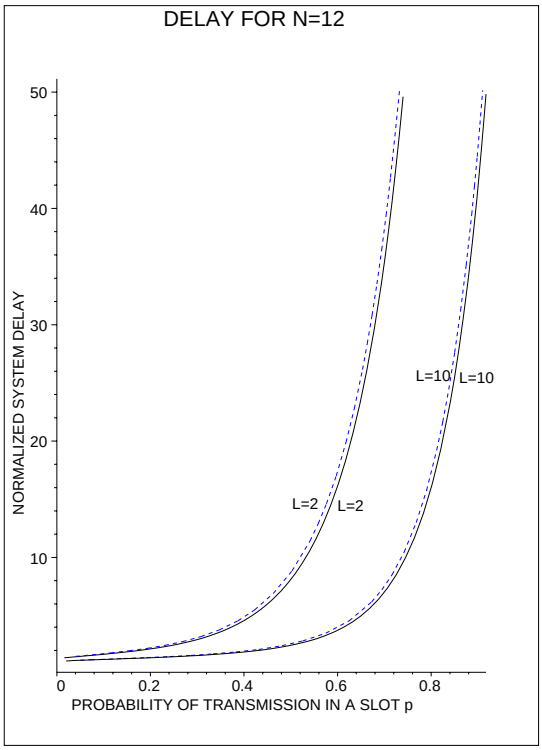


Figure 4.16: Normalized system delay versus transmission probability for RICH-SP and RICH-DP for a fixed number of nodes $N = 12$

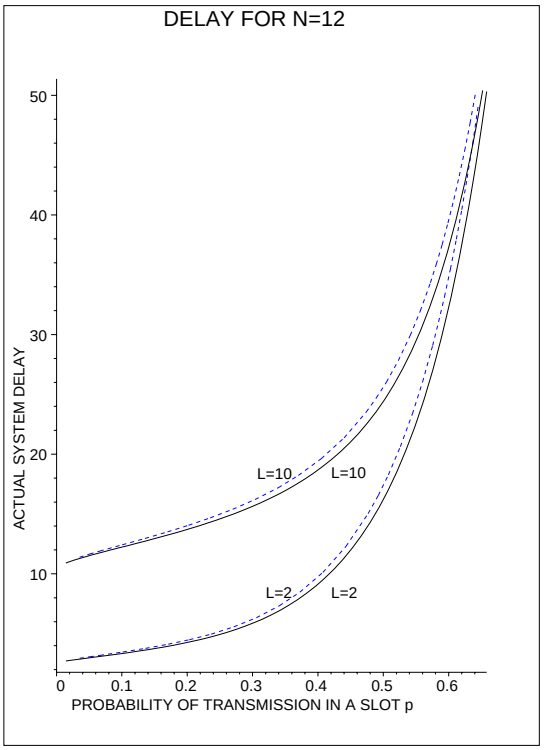


Figure 4.17: Actual system delay versus transmission probability for RICH-SP and RICH-DP for a fixed number of nodes $N = 12$

mates a commercially available frequency hopping radio operating over the 2.4GHz ISM band. By using the external model access (EMA) capability of the OPNET simulation tool, we produced a radio model with 79 frequency channels of bandwidth 1MHz and maximum data rate of 1Mbps. Because all the commercially available radios are half-duplex, the simulated radio cannot receive or transmit data at the same time. The simulation model for the physical layer was derived from the standard, high-fidelity, 13-pipeline stages model that is embedded in the simulation tool [49]. To be compatible with the analysis, we chose not to include any modifications in the physical layer that would simulate delay or power capture phenomena.

Nodes are assumed to be approximately one mile away from each other, giving a maximum propagation delay of 5 microseconds. We included an overhead of 24 microseconds to account for receive-to-transmit turn-around time, the necessary framing (preamble) bits, and guard-bands.

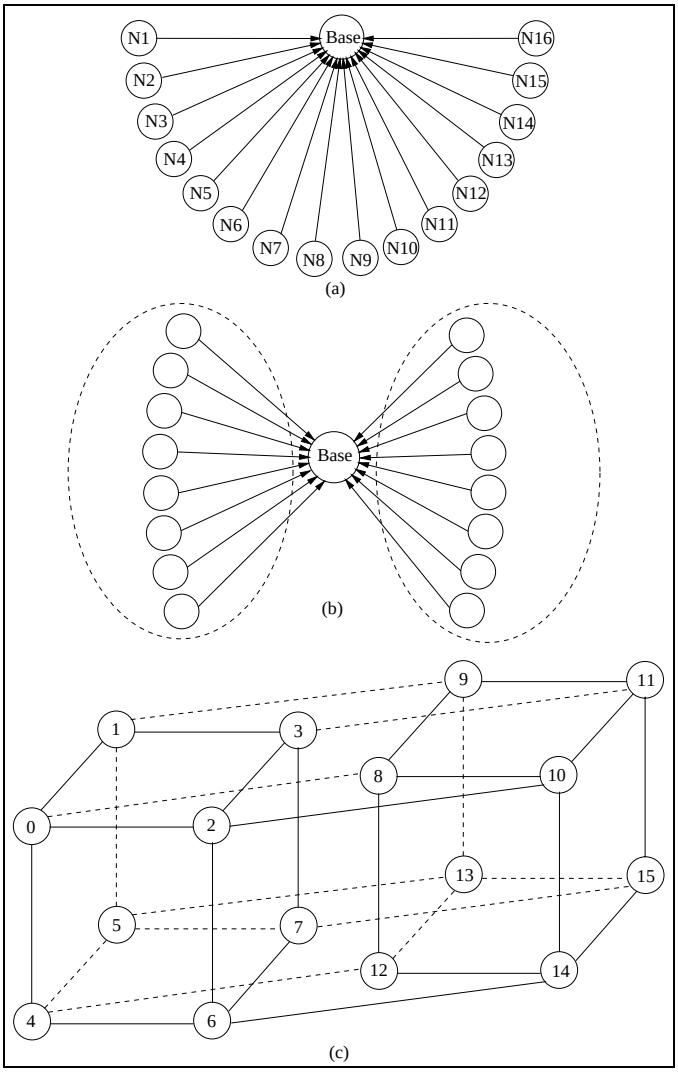


Figure 4.18: Various network topologies used in the simulations

Because the size of an RTR is equal to 96 bits, we chose our slots to be equal to 120 microseconds. When two control packets collide they back-off for an amount of time that is exponentially distributed up to the size of a data packet. Clearly, there are many different back-off strategies that can be applied to help improve the performance of RICH-DP (or MACA-CT for that matter), but this is not the focus of this chapter. If a node fails to initiate a handshake after seven retransmissions, the data packet is dropped from the head of the queue.

Figure 4.18 shows the various topologies used in the experiments. Figure 4.18(a) shows a

fully-connected network in which all the traffic produced from nodes $N1$ to $N16$ is directed to the base station, *Base*. Figure 4.18(b) shows two groups of eight nodes that can hear each other node in the same group but are hidden from all the nodes in the other group. Again, traffic is generated from all the nodes in each group with destination the central base station *Base*. In Figure 4.18(c) a multihop network of sixteen nodes in a four dimensional hypercube configuration is depicted. The lines between the nodes show the connectivity in the network. A node is generating traffic that four other nodes will receive at any given time, and there are always at least three other nodes that are hidden. These topologies were chosen for two reasons: to compare with similar topologies used in prior work on collision-avoidance [36], and to test the performance of the protocols under widely different conditions.

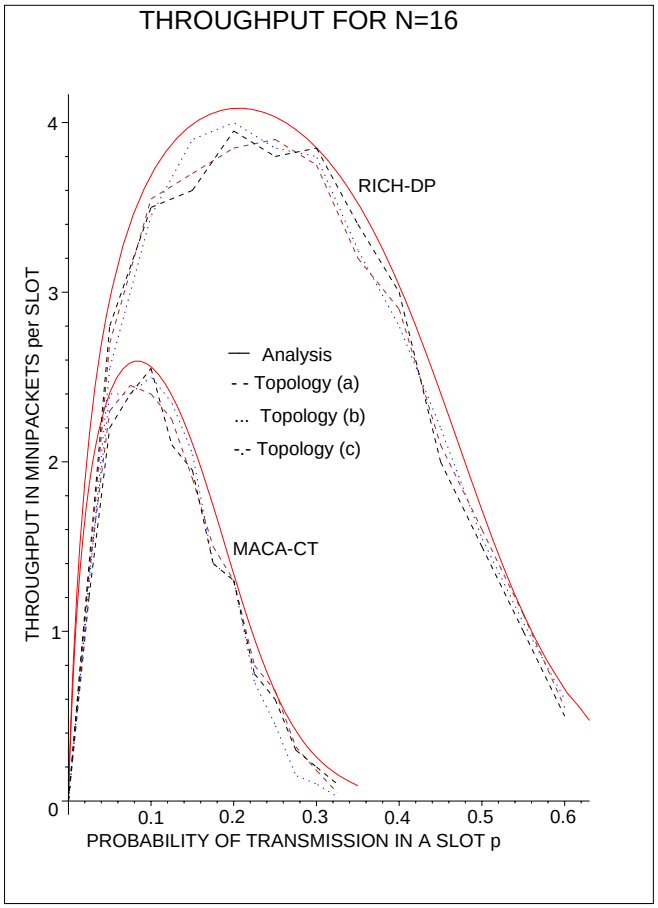


Figure 4.19: Aggregate throughput for RICH-DP versus MACA-CT for the topologies of Fig. 4.18; the number of nodes is N = 16 and the average packet length is $\overline{L} = 10$ or approximately 150 bytes

Data packets are generated according to a Poisson distribution and the data packet size is assumed to be constant equal to 150 bytes, which equals to approximately 10 slots (i.e. $L = 10$) of 120 bits each. According to our analytical model, for the same number of nodes in the network the number of packets transmitted per slot remains the same. To demonstrate that the performance of RICH-DP does not depend on the selected network topology, we collected simulation results for all three topologies shown in Figure 4.18. Figure 4.19 shows the throughput measured for MACA-CT and RICH-DP versus the results found with analytical methods and described in section 4.3. It is clear that the effective throughput is fairly independent of the exact network topology since for all three configurations our simulation results are within a 10% difference from the results obtained from the analysis, which is very reassuring. Some difference is expected, because the simulated radio model includes extra overhead bits for a more accurate representation of the physical effects that take place when a packet is sent or received (i.e. framing bits, padding bits). The two factors that contribute to performance that is independent of network topology, is that any node in any of the networks has more available channels than neighbors competing for them, and RICH-DP provides correct collision avoidance in the presence of hidden terminals [103] [100].

Using all three network topologies, a number of statistics were recorded to help understand the various effects that take place when a commercially available frequency hopping radio operates. For example, when the nodes in the network produce packets at a data rate higher than the available channel bandwidth, the size of the packets waiting in the queue to be serviced grows rapidly. As can be seen in Figures 4.20 to 4.22 for the network topology in Figure 4.18(a), when the data rate is low, all the packets are received by the base station and the end-to-end medium access delay remains almost constant (Fig. 4.20). However, when the data rate is higher than the radio can deliver, packets are lost (after exceeding the available amount of retransmissions) and the delay increases rapidly (Fig. 4.21). A collision resolution mechanism could be applied in the future to guarantee delay bounds for certain kind of applications (i.e. voice). There are many examples of such a mechanism in the literature (i.e. [40]).

4.6 Conclusions

We have specified a family of receiver-initiated collision-avoidance protocols that correctly eliminate hidden-terminal interference without the need for carrier sensing or the assignment of unique codes to network nodes, both of which are difficult to accomplish in ad-hoc networks based on commercial radios operating in ISM bands. We proved that RICH-SP and RICH-DP do eliminate hidden-terminal interference, and compared their throughput against MACA-CT, which is a recent example of collision-avoidance protocols that do not require carrier sensing but need code pre-assignments to operate correctly. For this comparison, we used the same analysis method introduced by Sousa and Silvester for code-hopping protocols [88] for RICH-SP and we introduced a novel analytical framework for the characterization of RICH-DP. Our results showed that both RICH protocols achieve higher throughput than MACA-CT, without the need for any code assignments. Our delay analysis verified that RICH protocols can drastically reduce the actual and normalized system delay, even under medium-to-high offered load. Various simulation scenarios were developed to verify the analysis and investigate other aspects of the RICH protocols (i.e. queue build-up).

Although we have assumed a perfect physical channel (i.e. no errors are introduced), the effects from interference should remain fairly minimal when using slow frequency hopping, especially when the number of co-located users is under the FCC recommended threshold. Future extensions of RICH could very easily modify the polling function so that certain frequency channels that demonstrate high bit error rate are omitted during the hopping cycle procedure. This is of particular importance since the frequency hopping medium access technique is vulnerable to narrow-band jamming. For instance, a common microwave operating over the 2.4GHz ISM band can produce enough interference to increase the bit error rate by several orders of magnitude for a nearby radio in the same band. The polling function could also be the driving factor in designing an extension of the RICH protocols that provides a differentiation of services or a "power aware" medium access that selectively checks certain frequencies for potential indications of any neighbor

activity. We believe, that the value of the RICH protocols as presented in this chapter lies in their simple and efficient design. It should be very interesting to see these protocols operating on a real ad-hoc network sometime in the future.

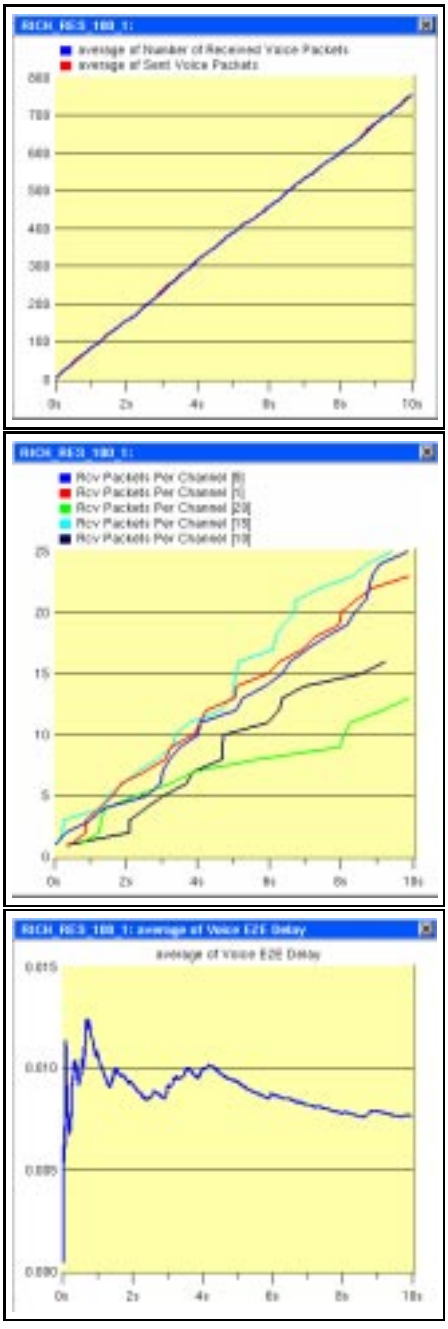


Figure 4.20: Packets send with an aggregate node data rate that is less than the available channel bandwidth

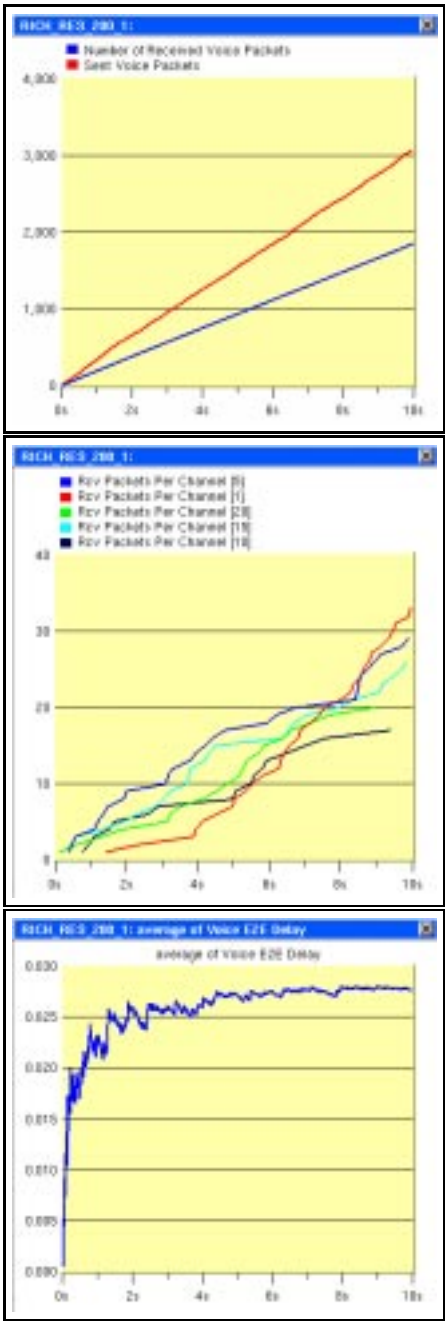


Figure 4.21: Packets send with an aggregate node data rate that is higher than the available channel bandwidth

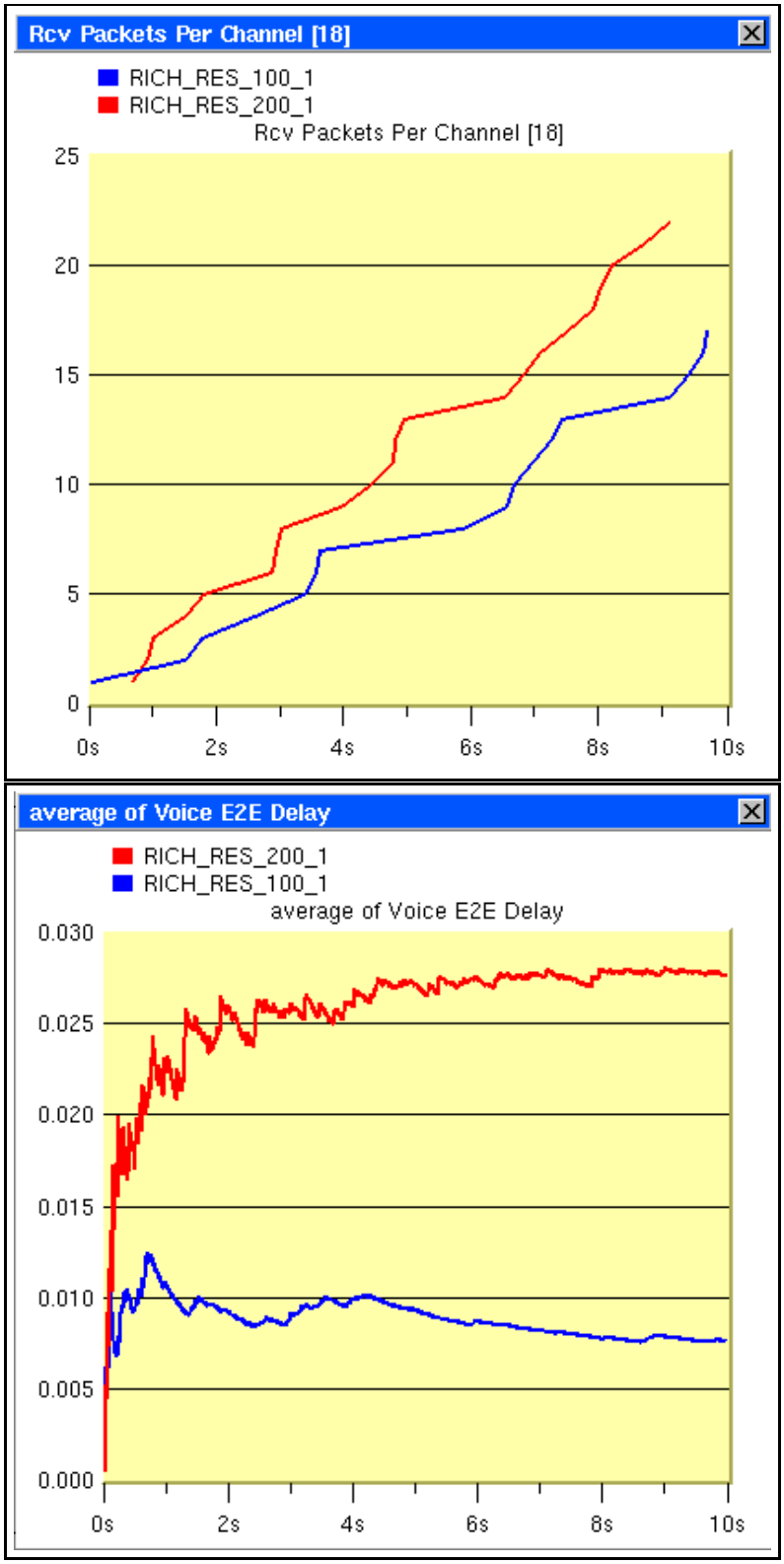


Figure 4.22: Difference between an aggregate arrival rate that is less, or more than, the available channel bandwidth

Chapter 5

Sender-Initiated Channel-Hopping Protocols

This chapter consists of two major sections. In the first section we present a sender-initiated, mutli-channel MAC protocol that we call Channel-Hopping Multiple Access (CHMA). CHMA has similar control packet handshakes as RICH-SP and therefore will be briefly presented. In the second section of the chapter, our motivation is to design a MAC protocol that supports multicast, broadcast, and unicast traffic efficiently. We call the new protocol Channel-Hopping with Access Trains (CHAT).

5.1 Channel-Hopping Multiple Access

As we have seen in the previous chapter, RICH-SP is a very efficient multi-channel MAC protocol that is based on a receiver-initiated handshake and does not require carrier sensing or any pre-assignment of codes for correct collision-avoidance. However, we can easily reverse the control packet handshake used in RICH-SP and produce a sender-initiated MAC protocol that would be based on the same principles of operation as RICH-SP. Our goal with this sender-initiated approach

is twofold: to verify that we indeed outperform other sender-initiated MAC protocols, and to build a basis for a MAC protocol that can support the notion of packet trains as presented in the literature for single-channel networks [35].

Section 5.1.1 describes CHMA, a new MAC protocol that operates over any spread spectrum modulation and does not require code assignments or carrier sensing. According to CHMA, all nodes in a network are required to follow a common channel-hopping sequence. A channel can be defined to be a frequency channel, a spreading code, or a combination of both. At any given time, all nodes that are not sending or receiving data listen on the common frequency channel. To send data, nodes engage in a sender-initiated dialogue over the frequency channel in which they are at the time they require to send data. Those nodes that succeed in a collision-avoidance handshake remain in the same frequency channel for the duration of their data transfer, while the rest of the nodes continue to follow the common channel hopping sequence.

Section 5.1.4 proves that, in the absence of fading, CHMA provides correct collision-avoidance in a multi-hop network. Section 5.1.5 analyzes the throughput in unslotted, multi-hop networks with CHMA. We compare CHMA with the MACA-CT protocol [52], which uses MACA collision-avoidance handshakes over a common channel and a transmitter-oriented data channel assigned to avoid collisions of data packets; we chose MACA-CT for our comparison, because it is essentially the same concept as that used in CHMA and is a good representative of collision-avoidance solutions that eliminate the need for carrier sensing at the expense of requiring unique channel assignments. Section 5.1.6 calculates the system delay in multi-hop networks for CHMA as well as MACA-CT. Section 5.1.7 presents our conclusions.

5.1.1 Sender-Initiated Channel-Hopping

Hidden-terminal interference can be eliminated by the assignment of channels or codes to senders or receivers in a way that no two senders or receivers share the same code if they are within a two-hop neighborhood. With commercial frequency-hopping radios operating in ISM bands, radios

have to synchronize in time so that all radios hop to different frequency channels at approximately the same time.

CHMA exploits the fact that the nodes of a frequency-hopping network must agree on when to hop to eliminate hidden-terminal interference. A common frequency-hopping sequence is assumed by all the nodes (i.e., a common channel), so that nodes simultaneously use the same frequency channel pattern to listen, unless instructed otherwise. Nodes then carry out a sender-initiated collision-avoidance handshake to determine which sender-receiver pair should remain in the same frequency in order to exchange data, while all other nodes that are not engaged in data exchange continue hopping on the common hopping sequence. Because the collision-avoidance handshake ensures that the receiver of a successful handshake will not receive packets that suffer from hidden-terminal interference, and because all nodes not able to exchange data must hop to the next frequency channel, CHMA eliminates the need for carrier sensing and code assignment by simply allowing the sender and receiver of the handshake to remain on the same frequency channel in which they succeeded in their handshake. The dwell time for a frequency channel in CHMA need be only as long enough to transmit a pair of MAC addresses, a CRC and framing. *Synchronous* frequency hopping was used since it has been proven to provide better throughput and lower interference [105, 60]. With CHMA, nodes in an ad hoc network can transmit unicast packets without experiencing hidden-terminal interference.

5.1.2 CHMA

The basic operation of CHMA is shown in Fig. 5.1. All the nodes follow a common channel-hopping sequence and each hop lasts the amount of time needed for nodes to receive a collision-avoidance control packet from a neighbor. A node that has a local data packet to send to any of its neighbors sends a ready-to-send (RTS) control packet over the current channel hop specifying the address of the intended receiver and its own address. All the nodes hop to the next frequency channel, and if the RTS is received successfully by the intended receiver, it sends a

clear-to-send (CTS) to the source node over the same common channel hop. At that time, the two given nodes will proceed to exchange data over the same frequency channel whereas all the other nodes hop immediately to the next frequency. In practice, the dwell time in a frequency channel needs to be only long enough to allow an RTS to be received by a destination node. When the transmission of data is completed, then sender and receiver re-synchronize to the current common channel hop. If either multiple RTSs are sent during the same channel hop, or the destination node does not receive the RTS (because it is already engaged in another handshake), no CTS is send to the source node. Consequently, the source node does not hear anything a round-trip time after sending its RTS and must rejoin the rest of the network at the current channel hop.

In Fig. 5.1, all the nodes start at time $t1$ from frequency $h1$. At time $t2$ the system is at frequency $h2$ and so on. At time $t1$ node x sends an RTS to node y. At time $t2$ all the nodes hop to frequency $h2$. Node y immediately responds with a CTS which is received by node x before the beginning of $t3$ time slot. Upon reception of a collision free CTS, node x will remain at the same frequency along with y to transmit its data. While x and y, stay in $h2$ until x has finished sending its data, all the other nodes continue to $h3$. At time $t3$, node a sends an RTS to node b but node b is busy transmitting data to another node (notice that we only consider uni-directional radios). Therefore, node b does not receive the RTS and at time $t4$ there is silence. In this case node a has to back off and therefore continues to hop with the other nodes to frequency $h4$. At time $t4$ node c sends an RTS to node k and d sends an RTS to node l within τ seconds. Since nodes c, d, k, l are in the same neighborhood a collision occurs. Both nodes c and d have to back off and try to send an RTS at a later time.

After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time equal to one frequency channel. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTS state. In the RTS state, the node sends an RTS packet with the destination address of the node that is the target destination, in this case z. In

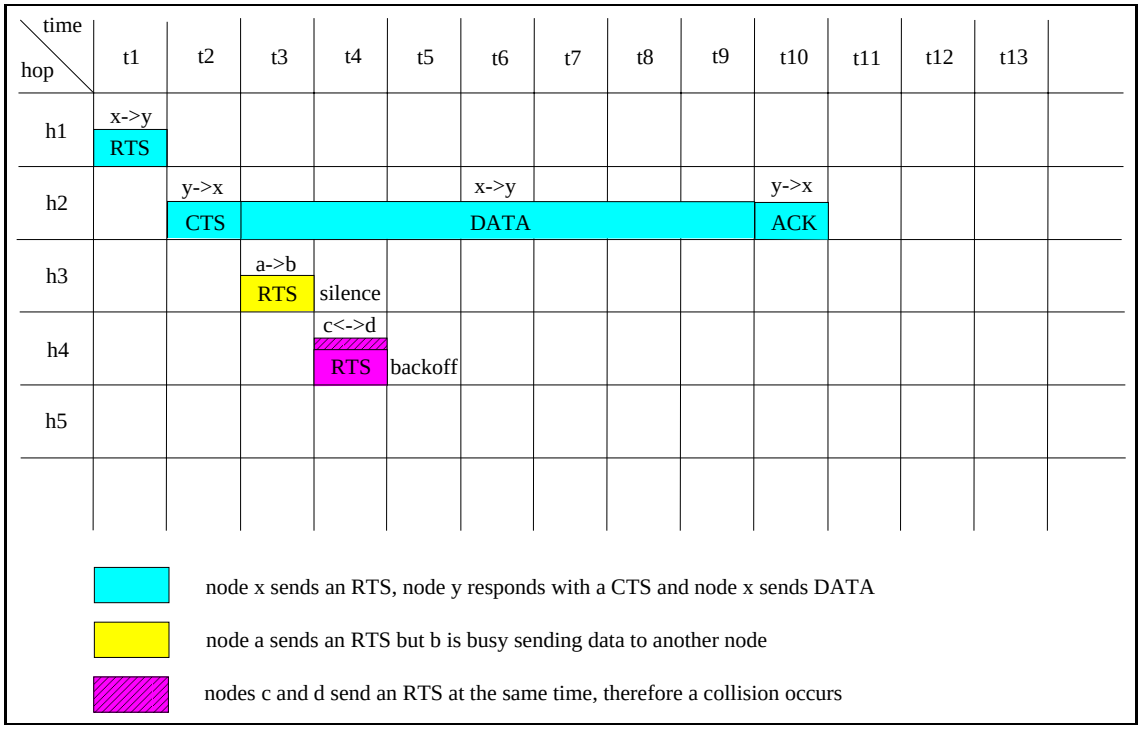


Figure 5.1: CHMA illustrated

the RTS packet sent to z, it specifies the different base frequency in which x will be available to establish a different hop sequence with z to receive any potential data from z.

If node z receives the RTS correctly and has data for x, node z transitions to the XMIT state, where it hops to a new base frequency and transmits a CTS packet to x in the frequency channel pattern that corresponds to that particular and unique hop pattern; otherwise, node z receives noise in that frequency and continues to hop with the rest of the nodes in the same base frequency that it was given at the beginning of the *hop cycle*. After sending its RTS, node x jumps to the new base frequency where it waits until the end of the next hop. At this time, there are two possible cases: (a) either there is silence (noise) in the channel and therefore node x realizes that there was a collision or the RTS was not received and resumes hopping with the rest of the nodes, only to try again at a random later back-off time, or (b) a CTS packet arrives allowing x to sent a data packet at the same unique frequency channel.

5.1.3 MACA-CT

The key difference between CHMA and MACA-CT is that, in MACA-CT, the control packet handshake occurs in the common channel, and then only the data are sent in a private per-node channel (Fig. 5.2). Since the CTS is now sent in the common channel, there is a possibility that a hidden node will transmit an RTS at the same time of the CTS, resulting in a collision. It is obvious that the vulnerability period in this case is double the one in CHMA.

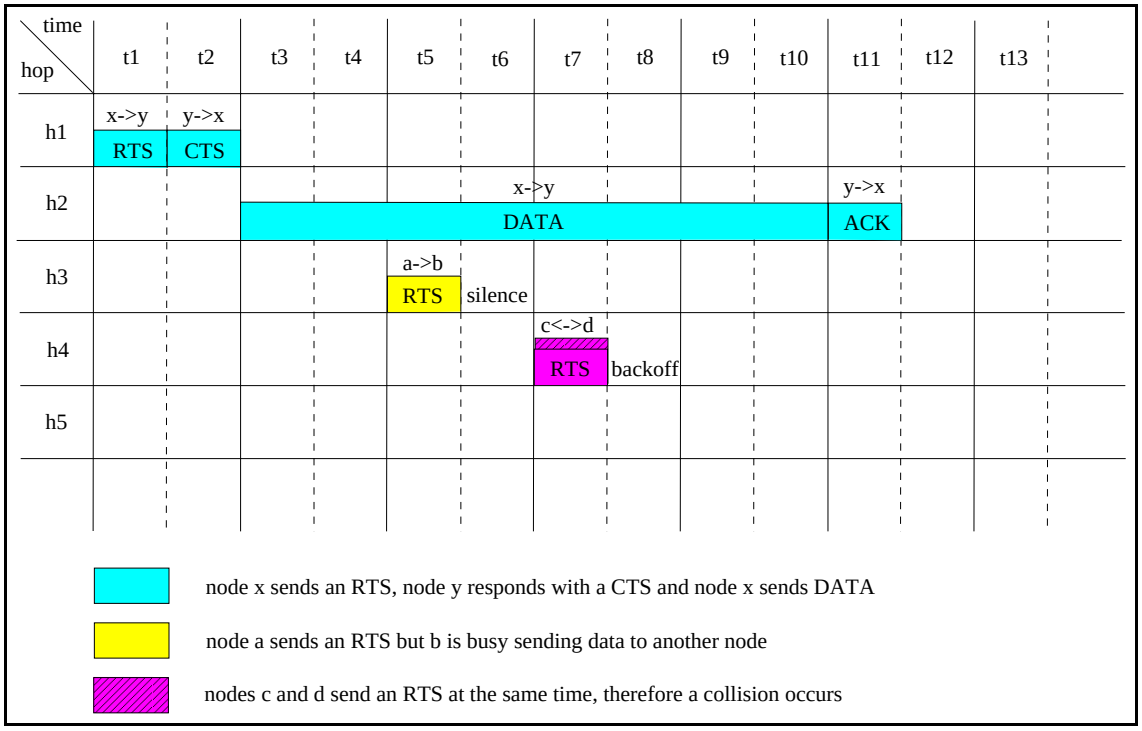


Figure 5.2: MACA-CT illustrated

5.1.4 Correct Collision Avoidance

In [52] it is shown that MACA-CT provides correct collision-avoidance in the presence of hidden terminals. Theorem 12 below shows that CHMA also guarantees that there are no collisions between data packets and any other transmissions. The following assumptions are made to demonstrate correct collision-avoidance [36]:

A0) A node transmits an RTS that does not collide with any other transmissions with a non-zero probability.

A1) The maximum end-to-end propagation time in the channel is $\tau < \infty$.

A2) The transmission time of an RTS and a CTS is γ, the transmission time of a data packet is δ, and the hardware transmit-to-receive transition time is zero; furthermore, $2\tau < \gamma \leq \delta < \infty$. The dwell time in each frequency is equal to the time needed to transmit an RTS (or CTS) plus the maximum end-to-end propagation time.

A3) There is no capture or fading in the channel. Moreover, any overlap of packet transmissions at a particular receiver, causes that receiver to not understand any of the packets (worst case scenario).

The approach used to show that a collision-avoidance protocol works correctly; i.e., that it prevents data packets from colliding with any type of packets, consists of showing that, once a data packet is sent by a node, the intended receiver obtains the packet without interference. Assuming zero processing and turn-around delays is done for convenience; however, the same type of proofs, with adjusted parameters, apply for non-zero hardware delays.

Theorem 12 *CHMA provides correct collision-avoidance in the presence of hidden terminals when the maximum number of nodes within any 2-hop distance in the network is less than the orthogonal channels available.*

Proof: The proof that CHMA provides correct collision-avoidance is the same as the one used for RICH-SP in Theorem 10. *Q.E.D.*

5.1.5 Approximate Throughput Analysis

We can apply the same throughput analysis techniques for CHMA as the ones presented in Chapter 4 for RICH-SP. The key difference between the two protocols lies on the fact that with

CHMA a sender-initiated control packet handshake occurs whereas with RICH-SP the receiver sends a polling signal to the sender soliciting potential data packets. Because of this difference, with CHMA the utilization of the medium should be expected to be slightly higher since there is always a data packet to be transmitted after a successful control packet exchange.

Fig. 5.3 shows the throughput achieved versus the probability of transmission p for various numbers of nodes in the network. Since the slot duration in CHMA is half the one in MACA-CT the probability of transmission at a given slot is $\frac{p}{2}$. Because the vulnerability period in CHMA is half the time spend in MACA-CT the maximum throughput is always higher for CHMA. Due to the small vulnerability period with CHMA, even for high probability of transmission, i.e. $p > 0.5$, the sustained throughput is high. Since no data will be ever sent to a busy terminal with CHMA, nodes in CHMA are immediately available to try again, something that is not the case in C-T [88]. Therefore, at any given time slot the number of nodes available to transmit an RTS is maximized, whereas the contention period is simultaneously minimized, providing a highly efficient combination.

Fig. 5.4 shows the throughput against the probability of transmission p for a fixed number of nodes ($N = 12$) with the average length packet L being the parameter. As it is obvious, CHMA again has a higher throughput than MACA-CT regardless of the size of the data packet. The general conclusion that can be drawn in this case is that with a longer average packet length, higher throughput can be achieved. However, in a realistic environment, by increasing the length of the transmitted packet we also increase the probability that errors will occur. Furthermore, when the number of co-located nodes is high the interference from adjacent frequency channels is more likely to introduce errors in the transmission of data packets.

5.1.6 Delay Analysis

The delay analysis for CHMA also follows the same steps as the ones presented in Chapter 4 for RICH-SP. The difference in the delay experienced between CHMA and RICH-SP is due to

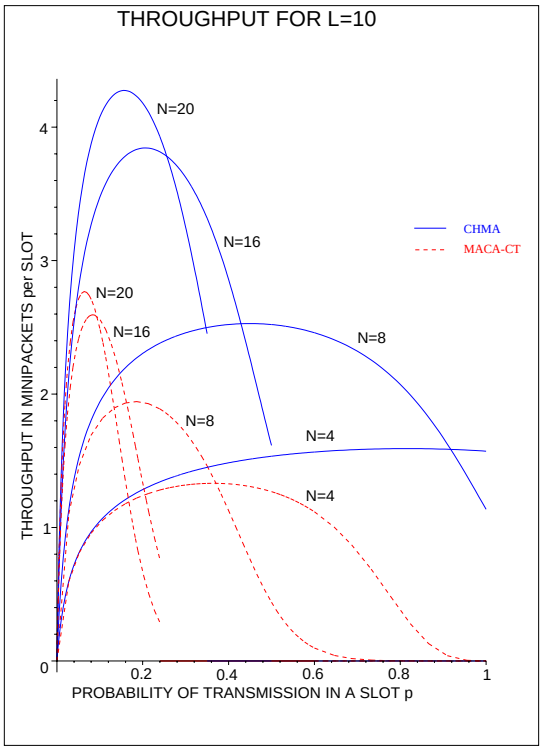


Figure 5.3: Throughput versus transmission probability for MACA-CT and CHMA for a fixed average packet length $\overline{L} = 10$

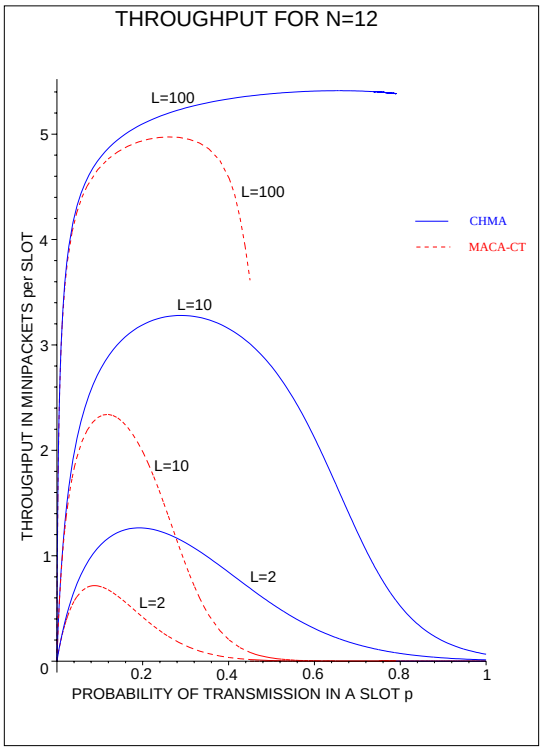


Figure 5.4: Throughput versus transmission probability for MACA-CT and CHMA for a fixed number of nodes $N = 12$

the different nature of the control packet handshake mechanisms that are employed by the two protocols.

In Fig. 5.5 we can see the numerical results obtain for the normalized delay performance of CHMA and MACA-CT. For light load ($p < 0.2$) we notice that both protocols have the same system delay. However, the system delay with CHMA remains almost the same until $p > 0.6$ whereas with MACA-CT it increases exponentially when $p > 0.4$. This is to be expected since when the load is high in the network the collisions between the control packets increase the delay. In this case, it is crucial to minimize the length of the handshakes that are susceptible to collisions. Indeed, with CHMA only RTSs can collide and therefore the vulnerability period is half the one in MACA-CT. Notice that, in general by increasing the packet length we reduce the normalized delay noticeably.

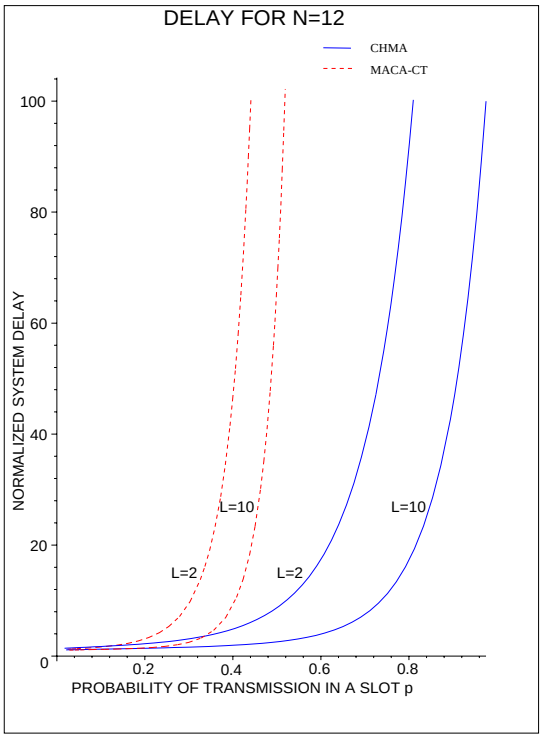


Figure 5.5: Normalized system delay versus transmission probability for MACA-CT and CHMA for a fixed number of nodes $N = 12$

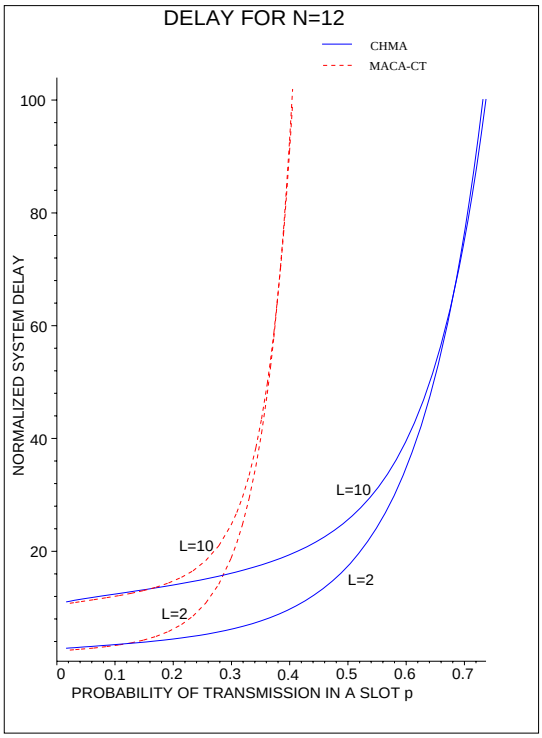


Figure 5.6: Actual system delay versus transmission probability for MACA-CT and CHMA for a fixed number of nodes $N = 12$

The actual system delay should include the transmission time for the data packet. Therefore, $D = \overline{D}/(1-q)$. In Fig. 5.6 the actual system delay that includes the packet transmission time is shown. In this figure, contrary to what happened with the normalized system delay, we notice that by increasing the packet length we do not achieve smaller delays. However, this is to be expected since the transmission time is the dominating delay in this case.

5.1.7 Conclusions

Our focus in this section was to present a sender-initiated collision-avoidance channel access protocol that guarantees correct floor acquisition without using carrier sensing or code assignment. CHMA is a very simple channel access protocol that can be used in any wireless radio with minimal effort. By using the same analytical methods presented in Chapter 4 we verified that CHMA outperforms MACA-CT [52] just like RICH-SP. In the next section, we introduce the

notion of data packet trains for multi-channel networks, which serves as the baseline for supporting multicast and broadcast traffic efficiently.

5.2 Channel-Hopping with Access Trains

This section introduces the Channel Hopping Access with Trains (CHAT) protocol. CHAT is the first MAC protocol based on collision-avoidance that (a) does not require carrier sensing to eliminate hidden terminal interference, and (b) guarantees that unicast and broadcast data packets can be transmitted without collisions. Section 5.2.1 describes the operation of CHAT in detail. Section 5.2.2 proves that, in the absence of fading, CHAT protocol provides correct collision-avoidance in a multi-hop network, i.e., it eliminates collisions of data packets, without the need for carrier sensing or code assignments. Section 5.2.3 presents the results of simulation experiments used to compare the throughput achieved with CHAT against CHMA and MACA-CT. Section 5.2.4 presents our conclusions.

5.2.1 CHAT

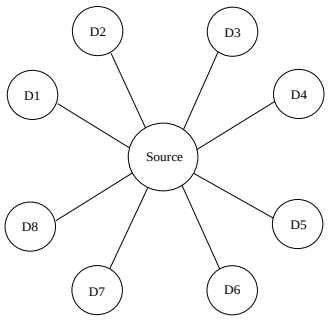
CHAT enhances the control handshake introduced in CHMA [101] to allow collision-free transmissions of packet trains, multicast packets, and broadcast packets. The basic operation for CHAT is shown in Fig. 5.1. All the nodes follow a common channel-hopping sequence and each hop lasts the amount of time needed for nodes to receive a collision-avoidance control packet from a neighbor. A node that has local data packets for any of its neighbors transmits a ready-to-send (RTS) control packet over the current frequency channel specifying its own address, and a bit vector of 32 bits. Each bit in the bit vector specifies a neighbor node.

If a node is already knows its position in the bit vector then after receiving an RTS, either (a) the corresponding bit is set and the node remains in the same channel hop, or (b) the bit is not set and the node moves on to the next channel hop. If a node does not know of its position

in the bit vector, by default it remains in the same channel hop. In the following time slot, nodes that either know or need to know if they are intended receivers of packets remain in the same frequency channel. The source node transmits a specialized RTS (SRTS) that has variable length and contains a list where each entry holds the following information fields: (a) the receiver node address, (b) a receiver number that is the index of the receiver node in the bit vector, and (c) a counter that represents the number of data packets in the packet train intended for a given node. After the SRTS is received by all the nodes in the same channel hop, each node compares the set of destination addresses with its own address. If a match is found, a CTS is sent back at a time that is equal to the current time plus the offset of the match from the beginning of the list times a slot duration. After a CTS is received successfully the source node transmits its data packet over the same channel hop.

For example, assume that at time t, node $Source$ transmits an RTS with a simplified 8-bit vector 01001100 as shown in Figure 5.7. Assuming one-to-one mapping between the names of the nodes and their position in the bit vector, there are three data packets for nodes $D2$, $D5$ and $D6$. If h is the hop duration, then at time $t+h$, lets assume that nodes $D1$, $D2$, $D4$, and $D5$ remain in the same channel hop, whereas node $D6$ is already busy exchanging data with some other node and therefore does not receive the RTS from $Source$. Immediately, node $Source$ transmits an SRTS with the format: $[addr(D2), 2, 1][addr(D5), 5, 1][addr(D6), 6, 1]$. Nodes $D1$ and $D4$ realize that there is no data packet to be received from $Source$ and synchronize with the rest of the nodes in a different channel. Node $D2$ receives the SRTS and since it is the first entry in the list, transmits a CTS right after the end of the SRTS. Likewise, node $D5$, transmits a CTS h seconds after the end of the SRTS since it was second in the list received in the SRTS packet and there was only one data packet in the packet train for node $D2$. On the other hand, node $D6$ never receives the SRTS and therefore no CTS is sent to node $Source$ in response. In this way, a number of CTSs is transmitted to the source collision free. After the source has received an indication (a CTS or silence) from all the nodes included in the list, it transmits all the corresponding data packets for

which a CTS was received. Notice that if there are more than one data packets for node $D2$ then still only one CTS will be sent back to node *Source*. When node *Source* is ready to send the data packets, the information kept in the SRTS list is used to determine when each packet should be transmitted. Only after all the packets for node $D2$ have been transmitted, node *Source* sends data to node $D5$.



Time	Node	Packet	Packet Content			
t	Source	RTS	source address: addr(Source) bit vector: 01001100			
t+h	Source	SRTS	destination address: bit vector index: number of packets:	addr(D2) 2 1	addr(D5) 5 1	addr(D6) 6 1
t+2h	D2	CTS	source address: addr(D2) destination address: addr(Source)			
t+3h	D5	CTS	source address: addr(D5) destination address: addr(Source)			
t+4h	D6	---	idle slot			

Figure 5.7: Node *Source* transmits a data packet train to nodes $D2$ and $D5$

CHAT has a clear advantage against all prior MAC protocols based on collision-avoidance when broadcast traffic is considered. In particular, with CHAT a broadcast packet is simply a unicast packet with all the bits in the bit vector of the RTS set. If all nodes return successfully a CTS then just one data packet is broadcast to all of them in a single handshake. When one or more nodes do not reply with a CTS the source node still sends a broadcast data packet to those nodes that have successfully replied with a CTS. The nodes that did not receive the broadcast data packet are saved in a list and another retransmission to send the same broadcast data packet is attempted from the source node at a later time. In contrast, in collision-avoidance protocols, a broadcast data packet has to be serviced as a number of unicast packets, one for each of the neighbors of a given source node. In this case, the source node has to succeed in a number of control packet handshakes with all of it's neighbors before the broadcast data packet is transmitted.

In practice, the dwell time in a frequency channel needs to be only long enough to allow an RTS to be received by a destination node. Our selection of a 32 bit vector stems from the fact that, with commercially available frequency-hopping spread-spectrum radios, the number of co-located

nodes must be kept below 15 to avoid excessive cross-channel interference. At the same time, the incurred overhead is kept to a minimum for both bandwidth and processing speed. Because the slot duration is fixed, we cannot have a variable length RTS with multiple destinations. By introducing a bit vector in the RTS followed by variable length SRTS packet, we guarantee the robust performance of the protocol even in the case of a very dense, heavily loaded ad hoc network, where the number of packets waiting to be serviced is large. In such a case, a variable length RTS packet would have a long list with all the packets waiting to be serviced, increasing the vulnerability period for the control packet handshake.

When the transmission of data is completed, then sender and receiver re-synchronize to the current common channel hop. If either multiple RTSs are sent during the same channel hop, or the destination node does not receive the RTS (because it is already engaged in another handshake), no CTS is sent to the source node. Consequently, the source node does not hear anything after sending its SRTS and must rejoin the rest of the network.

In Fig. 5.8, all the nodes start at time $t1$ from frequency $h1$. At time $t1$ the system is at frequency $h1$ and so on. Node x sends an RTS to node y at time $t1$. All the nodes but x and y hop to frequency $h2$ at time $t2$. Node x sends a SRTS and node y responds with a CTS at time $t3$. Upon reception of a collision free CTS, node x will remain at the same frequency along with y to transmit its data. While x and y, stay in $h1$ until x has finished sending its data, all the other nodes continue to $h2$. Similarly, at time $t2$ node k has local data packets for nodes l, m, and n. A SRTS is send with a list of the addresses of l, m, and n at time $t3$. At time $t4$, only nodes k, l, and n remain in frequency $h2$. Immediately, node l that is the first on the list of the received SRTS, sends a CTS. At time $t5$, there is no CTS received from node m possibly due to the fact that m was involved in an other transmission while node k transmitted the RTS. Finally, at time $t6$ node n also responds with a CTS. At time $t7$, node k transmits the first data packet in it's queue to node l, whereas at time $t14$ a second data packet for node n is sent. That is, 2 unicast data packets are send to nodes l and n in the same packet train. After any data packet exchange is completed the

recipient node has to synchronize with the current network hopping sequence. The sending node synchronizes with the rest of the system only when all the packets that consist the current packet train are transmitted.

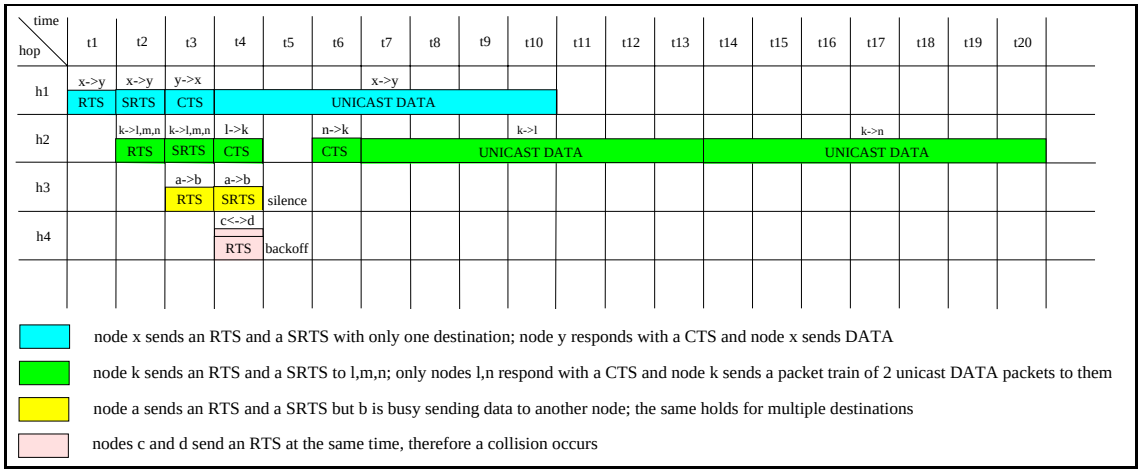


Figure 5.8: CHAT with unicast only data packet exchange illustrated

At time $t3$ and in frequency $h3$, node a sends an RTS to node b but node b is busy transmitting data to another node (notice that we only consider uni-directional radios). Therefore, node b does not receive the RTS and at time $t4$ there is silence. In this case, node a has to back off and therefore continues to hop with the other nodes to frequency $h4$. At time $t4$ and in frequency $h4$ node c sends an RTS to node k and d sends an RTS to node l within τ seconds. Since nodes c, d, k, l are in the same neighborhood a collision occurs. Both nodes c and d have to back off and try to send an RTS at a later time.

In Fig. 5.9, we see how CHAT can handle broadcast as well as unicast traffic at the same time. We assume that nodes l, m, n are the only three neighbors of node k, and at time $t2$ node k has a broadcast data packet to sent. First an RTS and a SRTS control packets are sent just as with any unicast data packet. When at least one node replies with a CTS, node k transmits it's broadcast data packet. In this example, all three nodes reply with a CTS and therefore the broadcast transmission is completed in just one handshake. If one or more nodes are already engaged in some other packet exchange then one or more CTSs will not be sent back

and the transmitting node has to try again at a later time. Obviously, a local list of all nodes that have already received the broadcast data packet is kept at the transmitting node to avoid any duplications.

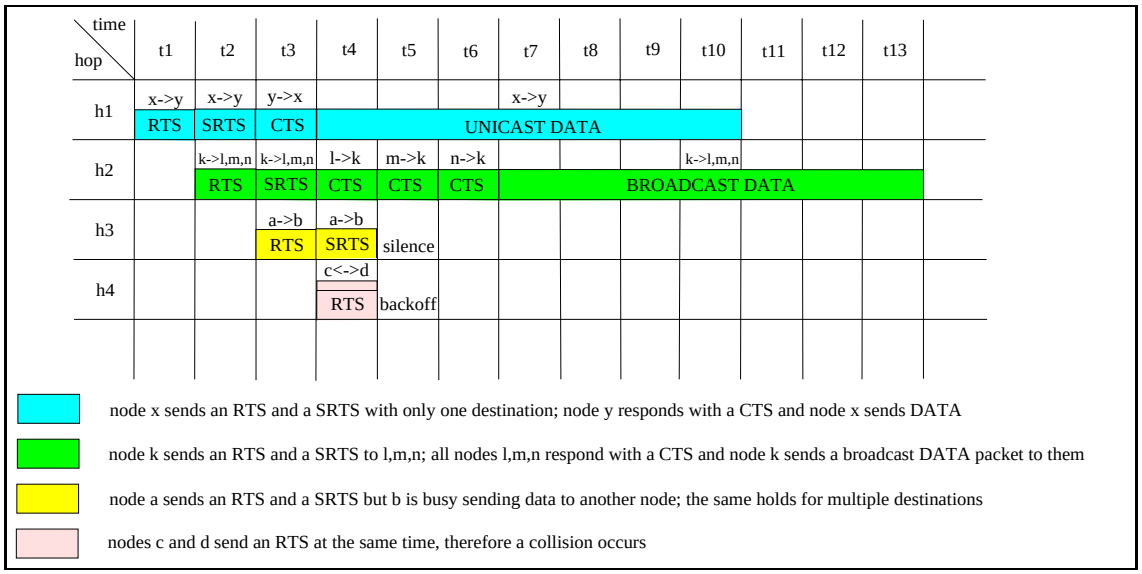


Figure 5.9: CHAT with unicast and broadcast data packet exchange illustrated

After a node is properly initialized, it transitions to the PASSIVE state. In all the states, before transmitting anything to the channel, a node must listen to the channel for a period of time equal to one frequency channel. If node x is in PASSIVE state and obtains an outgoing packet to send to neighbor z, it transitions to the RTS state. In the RTS state, the node sends an RTS packet, followed by a SRTS packet with a list of the destination addresses of the nodes that are the target destination, in this case just z. If node z receives the SRTS correctly it transitions to the XMIT state, where it transmits a CTS packet to x; otherwise, node z receives noise in that frequency and continues to hop with the rest of the nodes. After node x sends its SRTS there are two possible cases: (a) either there is silence (noise) in the channel and therefore node x realizes that there was a collision or the RTS was not received and resumes hopping with the rest of the nodes only to try again at a random later back-off time, or (b) a Clear To Sent (CTS) packet arrives allowing x to sent a data packet at the same unique frequency channel.

Although 400msec is a long time to transmit data in ISM bands, it may be desirable to allow nodes sending data to hop over multiple frequency channels to permit data exchanges lasting longer than 400msec. In addition by staying at the same frequency for a long period of time annihilates many inherent advantages that come with a frequency hopping modulation. For example, frequency hopping is the primary technology used in today's commercial radios for numerous reasons. Just to name a few, frequency hopping (a) can continue to work efficiently even in the presence of narrow-band jamming, (b) is resilient against fading and erasures, (c) minimizes the multi-path propagation problem, (d) provides increased security. However, in order to realize these benefits the rate with which the nodes in the network hop from one frequency to another should not be below a certain threshold.

To keep all the advantages from a frequency hopping modulation and at the same time avoid code assignment, a frequency-hopping sequence is needed, guaranteed to be free of interference from other data transmissions for at least a few dwell times, which could be up to the time when the same frequency channel occurs in the common hopping sequence used for handshakes. Even though we only consider the case where the data packets are sent over the same frequency channel, CHAT can be easily modified to sent portions of the same data packet in different frequency channels. The key difference here is that in the RTS sent from the source to the destination, a base frequency for the packet exchange is chosen according to the FCC frequency hopping tables.

When multiple RTSs are transmitted within a one-way propagation delay a collision takes place and the nodes involved have to transition to the BACKOFF state and try again at a later time chosen at random. After sending its RTS, node x waits for a response in the new frequency base. Node x determines that its RTS was not received correctly by z after a time period equal to one hop. If that is the case, node x will synchronize with the other nodes at a frequency channel that can be determined easily since node x is aware of the base frequency channel that the whole system is hopping at, from the initialization that took place at the beginning of the *hop cycle*.

To reduce the probability that the same nodes compete repeatedly for the same receiver

at the time of the next RTS, the RTS specifies a back-off-period unit for contention. The nodes that must enter the BACKOFF state compute a random time that is a multiple of the back-off-period unit advertised in the RTS. The simplest case consists of computing a random number of back-off-period units using a uniformly distributed random variable from 1 to d, where d is the maximum number of neighbors for a receiver. The simplest back-off-period unit is the time it takes to send a small data packet successfully.

5.2.2 Correct Collision Avoidance

Theorem 13 below shows that CHAT ensures that there are no collisions between data packets and any other transmissions. The following assumptions are made to demonstrate correct collision-avoidance [36]:

A0) A node transmits an RTS that does not collide with any other transmissions with a non-zero probability.

A1) The maximum end-to-end propagation time in the channel is $\tau < \infty$.

A2) A packet sent over the channel that does not collide with other transmissions is delivered error free with a non-zero probability.

A3) All nodes execute CHAT correctly.

A4) The transmission time of an RTS and a CTS is γ, the transmission time of a SRTS is variable, the transmission time of a data packet is δ, and the hardware transmit-to-receive transition time is zero; furthermore, $2\tau < \gamma \leq \delta < \infty$.

A5) The dwell time in each frequency is equal to the time needed to transmit an RTS (or CTS) plus the maximum end-to-end propagation time.

A6) There is no capture, erasure, or fading in the channel.

A7) Any overlap of packet transmissions at a particular receiver, causes that receiver to not understand any of the packets (worst case scenario).

The approach used to show that a collision-avoidance protocol works correctly, i.e., that it prevents data packets from colliding with any type of packets, consists of showing that, once a data packet is sent by a node, the intended receiver obtains the packet without interference.

With the commercially available spread spectrum radios today, periods of deep fading (*erasures*) disrupt any type of collision avoidance dialogue, i.e., data packets may experience collisions in the presence of fading. However, with frequency hopping radios the higher the rate with which a radio hops from one frequency to another the less the probability that an erasure will occur. Even though fast frequency hopping would be ideal to avoid erasures it is not a wide available technology yet. Instead, CHAT is specifically designed to operate over slow frequency hopping radios but since the dwell time need to be minimal (two MAC addresses, CRC, and framing bits) the effect of erasures should be negligible and therefore not considered any further.

Assuming zero processing and turn-around delays is done for convenience; however, the same type of proofs, with adjusted parameters, apply for non-zero hardware delays.

Theorem 13 *CHAT provides correct collision-avoidance in the presence of hidden terminals when the time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence.*

Proof: Consider a transmitting node A and a receiving node X and assume that A sends an RTS at time t_0. We denote with h the dwell time in a particular hop. If X does not receive the RTS correctly due to interference from any neighbor hidden from A, it does not transit to the particular base frequency in which A is waiting to transmit it's data and consequently no data are sent. Else, X receives A's RTS at time $t_1 = t_0 + h$ and transits to the particular base frequency specified in the RTS from A. At time $t_1' > t_0 + h$, node X has received a SRTS from A and responds with a CTS. Node A is then enabled to transmit it's data packet. Both nodes A and X hop in

the same hopping pattern that never collides with any other hopping pattern since we have made the assumption that time spent exchanging data is shorter than the time elapsed before the same frequency channel is reused in the common hopping sequence. Clearly, the size of a data packet train must be restricted to the maximum number of data packets that can be transmitted before the same frequency channel occurs again in the common hopping sequence. *Q.E.D.*

5.2.3 Performance Comparison

A number of simulation experiments is presented to investigate the performance of CHAT under different network topologies and to show how the results compare against CHMA and MACA-CT. We used the OPNET simulation tool [49] to implement all three protocols considered in our experiments.

For the simulation experiments, we used a multi-channel capable radio that approximates a commercially available frequency hopping radio operating over the 2.4GHz ISM band. By using the external model access (EMA) capability of the OPNET simulation tool, we produced a radio model with 79 frequency channels with 1Mhz bandwidth and maximum data rate of 1Mbps. Because all the commercially available radios are half duplex, the simulated radio can only receive or transmit data at the same time. The simulation model for the physical layer was derived from the standard, high-fidelity, 13-pipeline stages model that is embedded in the simulation tool. To be compatible with the analysis, we chose not to include any modifications in the physical layer that would simulate delay or power capture phenomena.

Nodes are assumed to be approximately one mile away from each other, giving a maximum propagation delay of 5 microseconds. We included an overhead of 24 microseconds to account for receive-to-transmit turn-around time, the necessary framing (preamble) bits, and guard-bands. Because the size of an RTS is equal to 96 bits, we chose our slots to be equal to 120 microseconds or 120 bits since our radios have a data rate of 1Mbps. When two control packets collide they back-off for an amount of time that is exponentially distributed up to the size of a data packet. Clearly,

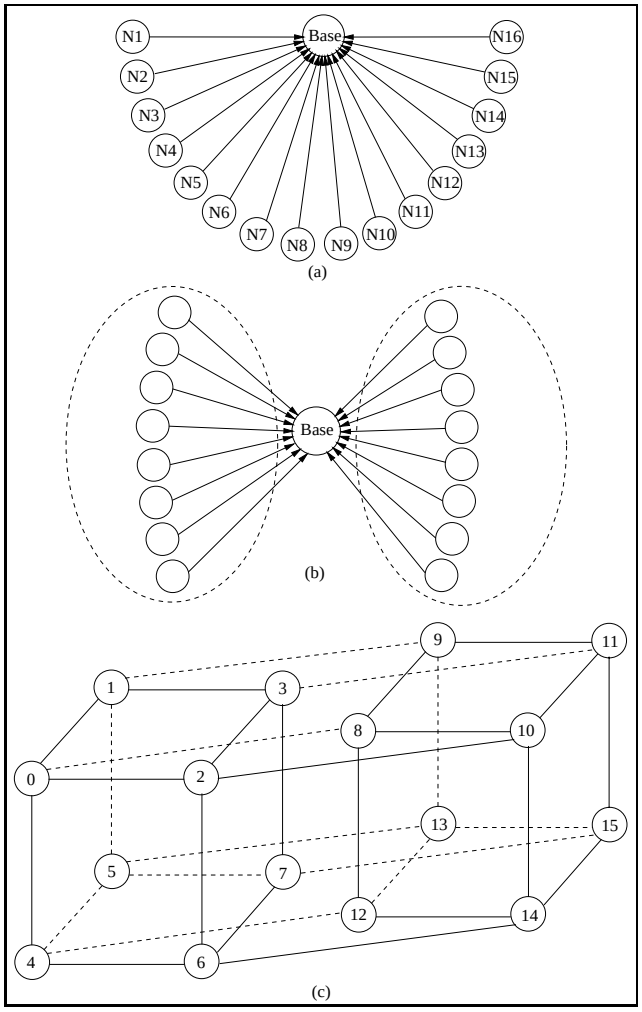


Figure 5.10: Various network topologies used in the simulations

there are many different back-off strategies that can be applied to help improve the performance of CHAT, CHMA and MACA-CT for that matter, but this is not the focus of this chapter. If a node fails to initiate a handshake after seven retransmissions, the data packet is dropped from the head of the queue.

Figure 5.10 shows the various topologies used in the experiments. Figure 5.10(a) shows a base station network in which all the traffic produced from nodes $N1$ to $N16$ is directed to the central node, $Base$. Figure 5.10(b) shows two groups of eight nodes that can hear each other node in the same group but are hidden from all the nodes in the other group. Again, traffic is generated

from all the nodes in each group with destination the central base station *Base*. In Figure 5.10(c) a multi-hop network of sixteen nodes in a four dimensional hypercube configuration is depicted. The lines between the nodes show the connectivity in the network. A node is generating traffic that four other nodes will receive at any given time whereas there are always at least three other nodes that are hidden. These topologies were chosen for two reasons: to compare with similar topologies used in prior work on collision-avoidance [36], and to test the performance of the protocols under widely different conditions. Notice also that even though with topologies 5.10(a) and 5.10(b) a packet train consists of two or more data packets with the same destination (the base station), with topology 5.10(c) in a given packet train there might be two or more data packets each with a different destination address. In this case, CHAT can take advantage of its special handshake mechanism to serve broadcast traffic by transmitting just one packet to a number of nodes at the same time.

Data packets are generated according to a Poisson distribution and the data packet size is assumed to be constant equal to 150 bytes, which equals to approximately 10 slots (i.e. $L = 10$) of 120 bits each. The simulated radio model includes extra overhead bits for a more accurate representation of the physical effects that take place when a packet is sent or received (i.e. framing bits, padding bits). To demonstrate that the performance of any channel-hopping protocol does not depend on the selected network topology, we collected simulation results for all three topologies shown in Figure 5.10. Figures 5.11, 5.12, and 5.13 show the throughput results measured for CHAT, CHMA and MACA-CT for the three network topologies shown in Fig. 5.10 for unicast traffic only. As also shown with analysis in [101] CHMA outperforms MACA-CT under any offered load by minimizing the critical vulnerability period during which collisions between the control packets can occur. In addition, CHAT exploits the fact that more than one data packets can be transmitted in just a simple control packet handshake to further improve the utilization of the medium. Especially under medium to heavy offered load CHAT seems to outperform CHMA by more than 10%. However, when the data rate is higher than what the radio can deliver, packets

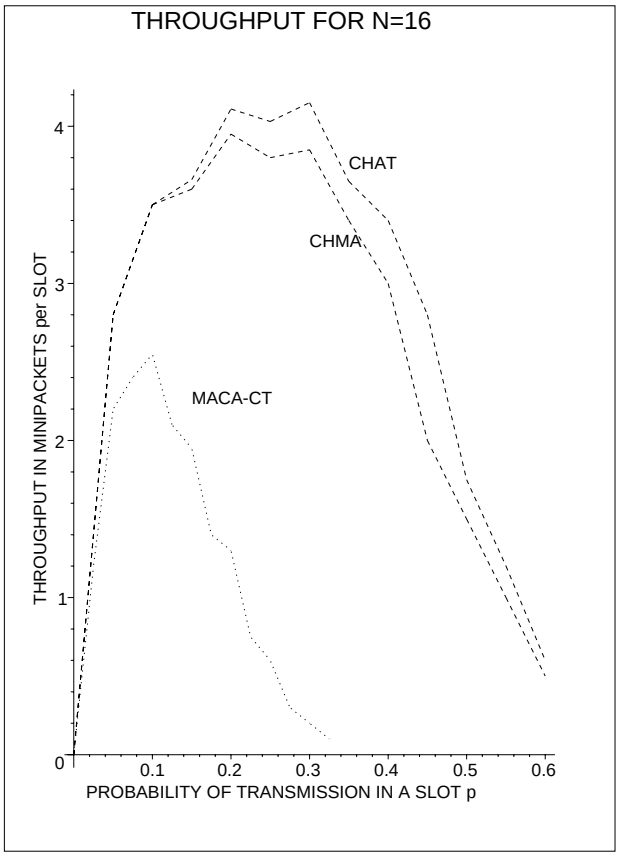


Figure 5.11: Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 5.10(a); the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

are lost (after exceeding the available amount of retransmissions) and the performance of CHAT degrades to that of CHMA. A collision resolution mechanism could be applied in the future to guarantee delay bounds for certain kind of applications (i.e. voice). There are many examples of such a mechanism in the literature (i.e. [40]).

From Figures 5.11, 5.12, and 5.13 it is obvious that the effective throughput is fairly independent of the exact network topology since for all three configurations our simulation results are within a 5% difference one from the other for all channel-hopping protocols. This difference is expected, since we are using random arrivals and have not applied any output evaluation techniques (i.e. batch mean intervals, long averages). The two factors that contribute to performance that is network topology independent, is that any node in all three networks has more available channels

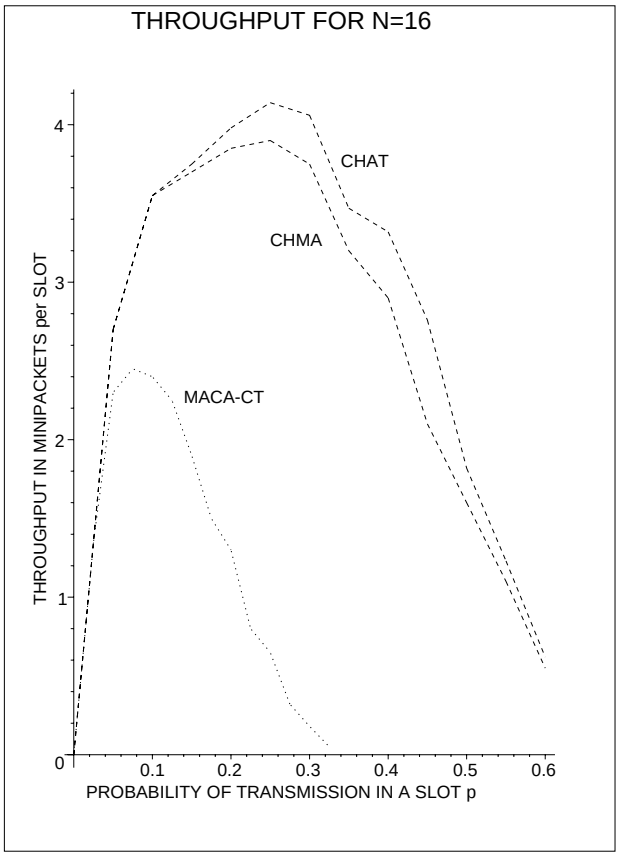


Figure 5.12: Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 5.10(b); the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

than neighbors competing for them, and all protocols considered provide correct collision-avoidance in the presence of hidden terminals [101, 52].

To examine the benefits of CHAT when we have broadcast or a mix of broadcast and unicast traffic a set of simulations was performed with the network shown in Figure 5.10(c). First we assumed only broadcast traffic and then we created a mix of broadcast and unicast traffic with equal probability (i.e. 50% of the traffic is broadcast and 50% is unicast). The retransmission policy for broadcast packets is the same as the one mentioned before for unicast only traffic. Since a broadcast packet is not successfully transmitted unless all neighbors have received a copy of it, in the presence of broadcast traffic we calculate the throughput as a single packet exchange. That is, even though a number of acknowledgments is returned for a single broadcast data packet only one

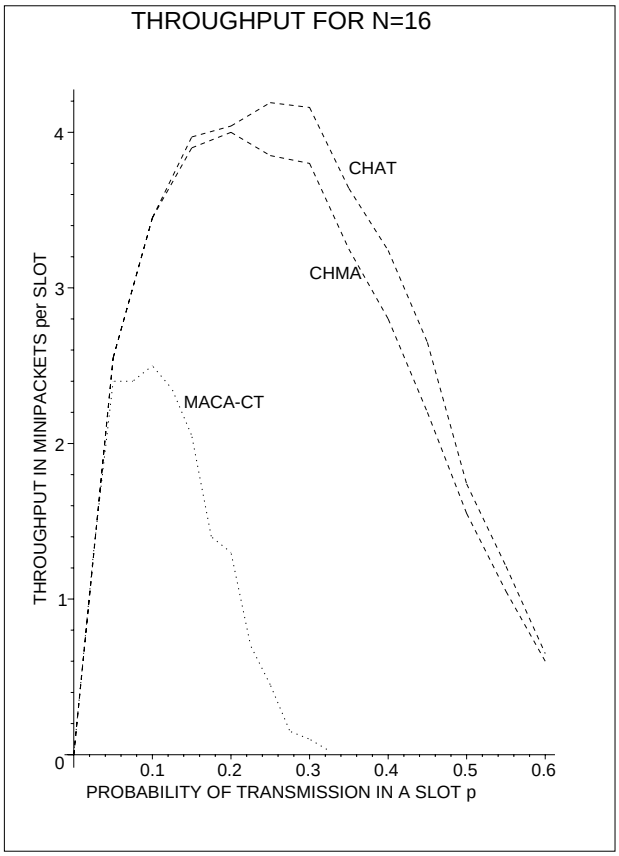


Figure 5.13: Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 5.10(c); the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

of them contributes to the calculation of the effective throughput in the network. The throughput results for broadcast only traffic are shown in Figure 5.14. Clearly, the throughput for MACA-CT and CHMA is much lower than the corresponding one presented previously with only unicast traffic. Even though there is a penalty with CHAT as well, the difference is considerably smaller than MACA-CT and CHMA. Notice that when broadcast traffic is present with MACA-CT and CHMA a number of unicast packets equal to the number of neighbors of a given node has to be successfully transmitted before the broadcast transmission is completed. On the other hand with CHAT a node can broadcast a data packet with much less control packet handshakes. Especially under light to medium loads where most of the neighbors of a given node are idle, a broadcast packet is sent with just a few attempts leading to throughput that is almost equal to the case of

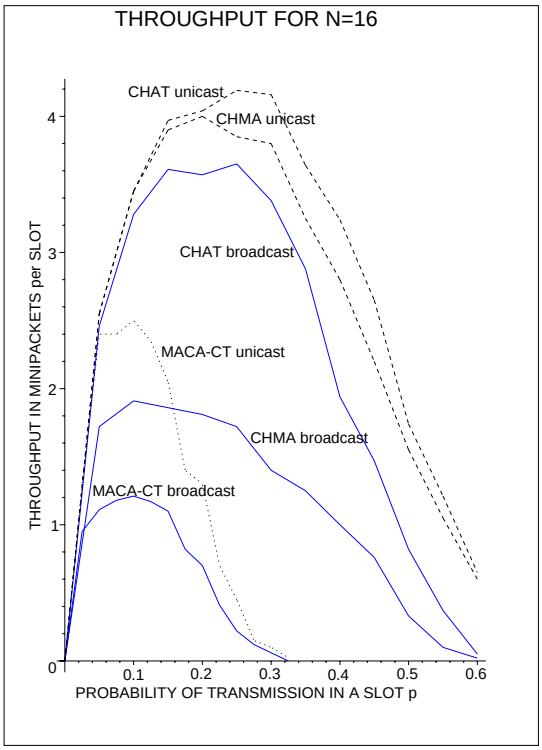


Figure 5.14: Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 5.10(c); broadcast only traffic is compared against unicast only; the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

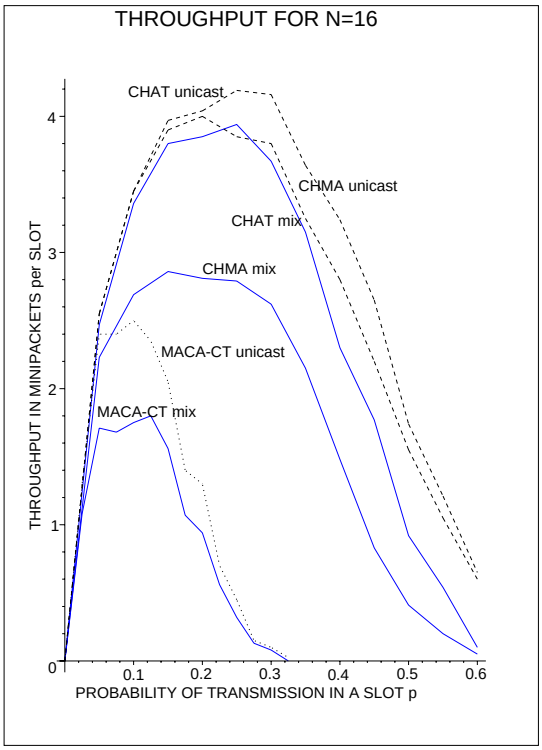


Figure 5.15: Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 5.10(c); a mix of broadcast and unicast traffic is compared against unicast only; the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

unicast only traffic.

Similar conclusions can be drawn when a mix of broadcast and unicast traffic is generated. As can be seen in Figure 5.15 in this case the throughput for MACA-CT and CHMA is also reduced much more than CHAT but since half of the traffic is unicast the difference is not as big as in the case of broadcast only traffic. When there is a mix of broadcast and unicast traffic, CHAT can combine in the same packet train broadcast as well as unicast data packets. The capability of CHAT to support efficiently broadcast traffic is a key feature of CHAT, because the routing control and many applications running on ad hoc networks are based on broadcast packet delivery.

5.2.4 Conclusions

We introduced a multi-channel access control protocol that can easily operate on commercially available, wireless radios operating in an ISM band. CHAT uses the concept of packet train data transmissions to minimize the number of control packets needed to establish a collision-free packet exchange for unicast, multicast, and broadcast traffic. We compared the throughput achieved with CHAT against CHMA and MACA-CT [52], which is a recent example of collision-avoidance protocols that do not require carrier sensing but need code assignment to operate correctly. For this comparison, we performed a large number of simulation experiments with various network topologies, and showed that CHAT achieves higher throughput than CHMA and MACA-CT for unicast and broadcast traffic, without the need for carrier sensing or any code assignments.

Our goal with CHAT was to serve multicast and broadcast traffic efficiently. A key feature of contention-based MAC protocols is that it is not trivial to support persistent reservations. When a node needs to transmit a packet to multiple destinations the traditional approach of a random access MAC protocol is to try and establish a sequence of multiple point-to-point connections with each and every neighboring node so that the broadcast packet finally reaches all possible destinations reliably. However, this leads to a very poor utilization of the medium, because a very large number of control packets needs to be exchanged in order for the broadcast packet to reach all possible destinations. With CHAT, we are able to unicast and broadcast traffic. By using just one extra control packet CHAT specifies a unique node or a set of nodes that are the recipients of a given data packet. By doing so, CHAT reduces drastically the number of handshakes that need to take place in order to reliably multicast or broadcast a data packet. Because ad-hoc networks must exchange routing information among neighboring nodes, and because of the increasing importance of multicast and broadcast applications in ad-hoc networks, CHAT is very well suited as a MAC protocol for such networks.

In the case of a non perfect physical channel the performance of CHAT as well as any

other protocol that supports multicast and broadcast traffic requests will be drastically deteriorated. Clearly, when the quality of the wireless link is very low, it becomes very difficult to achieve reliable broadcast or multicast transmissions due to the large number of retransmissions that take place. However, because CHAT uses far less control packets to exchange multicast or broadcast data, its' performance deteriorates much more gracefully providing always a superior approach to any other sender-initiated contention-based MAC protocol.

Chapter 6

Conclusion

6.1 Contributions

The main focus in this dissertation was the design and analysis of new medium access control (MAC) protocols for wireless ad-hoc networks. MAC protocols based on collision-avoidance handshakes have become very popular in wireless LANs and ad-hoc networks. Until recently all such MAC protocols were sender-initiated, in that the node that wants to transmit a data packet sends a short RTS asking permission from the receiver before transmitting a data packet. In addition, most MAC protocols require carrier sensing to avoid data packet collisions and maintain high throughput. Our motivation in this dissertation was twofold: to reverse the collision-avoidance handshake as a mechanism that can improve the utilization of the medium, and to provide correct collision-avoidance without the need for carrier sensing in multi-hop, multi-channel wireless networks.

Our first step towards receiver-initiated MAC protocols was a set of three different single-channel protocols that we called RIMA protocols. RIMA protocols are based on non-persistent carrier sensing to ensure their correct operation. The receiver polls one of its neighbors asking if it has a data packet to send. A receiver-initiated collision-avoidance strategy is attractive because it can, at least in principle, reduce the number of control packets needed to avoid collisions. With

the RIMA protocols, we introduced for the first time a receiver-initiated contention-based MAC mechanism that provides correct collision-avoidance in the presence of hidden terminals. We showed that by making the polling mechanism of RIMA a joint decision of the receiver and the sender RIMA outperforms any other sender-initiated MAC protocol proposed to date.

As our next step we studied the effect of applying limited persistence carrier sensing in FAMA [36] and in all three RIMA protocols. Our motivation for this work was the realization that under certain load conditions the utilization of the medium can be improved if some kind of persistent carrier sensing is used. We designed a variation of FAMA and the RIMA protocols to include limited persistence, and we proved analytically that indeed under a wide range of offered load we can gain significant benefits with persistent carrier sensing.

Moving on to our third step, we extended our receiver-initiated notion to multi-channel networks. Our main motivation in this case was not only to make good use of the inherent advantages of a receiver-initiated handshake mechanism but also to come up with a scheme that can be applied to commercial, off-the-self radios that operate in one of the available ISM bands. Two main obstacles have traditionally made difficult the seamless application of MAC protocols in the wireless arena. First, the need for collision-avoidance MAC protocols for single-channel networks to sense the channel as an integral part of the collision-avoidance handshake limits their applicability. Almost all of the commercial radios do not provide true carrier sensing, as in a wire-line network. The alternative solution of using busy tones to indicate when a receiver is busy [46] requires, in essence, a second transceiver, which is not economically attractive. Second, most of the multi-channel radios need to have a pre-assigned code (or frequency or channel) to each and every radio that is present and active in a given network configuration. Obviously, this can be a major drawback, especially in the case of large-scale, dense, ad-hoc networks that might be changing their configuration continuously. Our RICH protocols do not require carrier sensing or the pre-assignment of codes and as such this is the first time that a multi-channel MAC protocol can be applied as-is in a commercial radio. We proved that RICH protocols provide correct floor

acquisition and outperform any other contention-based sender- or receiver-initiated, multi-channel MAC protocol.

As our last step we designed a contention-based MAC protocol that can support efficiently unicast, multicast, and broadcast traffic. We began our effort by introducing CHMA, which is the equivalent of RICH-SP with a sender-initiated handshake. Based on CHMA, we designed CHAT, which is the first contention-based, multi-channel MAC protocol that supports the notion of sending a train of data packets to the same or different recipients over a single control packet exchange. Because of the inherent non-persistent nature of the contention-based MAC protocols, it is trivial to transmit multiple data packets with a single RTS-CTS handshake. However, we were able to cluster multiple request in a highly efficient way using packet trains. With CHAT, a data packet can be sent as a unicast transmission to just one recipient, but can also be sent to a group of recipients by setting a single field in the control packets used to acquire the channel. CHAT is very attractive for ad hoc networks, because routing update packets in such networks are sent much more efficiently using broadcast transmissions.

6.2 Future Work

In Chapter 2 we have presented an analytical approach for the evaluation of the performance of RIMA protocols for fully-connected networks. For multi-hop networks, we expect that our results should be equal or better to the ones presented for FAMA protocols [34] since all RIMA protocols provide correct collision-avoidance in the presence of hidden terminals.

Because our focus in this dissertation was on the MAC layer we have made the implicit assumption of a perfect physical channel. However, various physical phenomena adversely affect the nature of the underlying medium and consequently the behavior and performance of the MAC protocol. For example, frequency operation range, background noise, cross-channel interference, fading, erasures, multi-path propagation, delay or power capture are just some of the factors that

can affect the good operation of the MAC protocol. In the case of RIMA protocols, we expect that our protocols will behave in a similar way with any other IEEE 802.11-like protocol unless special provisions are made so that the MAC protocol can react to different conditions from the quality measured in the physical layer. However, for RICH protocols, we believe that there is a direct correlation between the dwell time that a given node remains in the same frequency channel and the probability that a channel error occurs. Our conjecture stems from the same assumption made when comparing slow frequency hopping radios with fast frequency hopping ones. A quantitative approach that shows the correlation between the dwell time and the performance of our MAC protocols due to channels errors is definitely of high priority for future research in this area.

With the increasing popularity of wireless, ad-hoc networks, many new applications have appeared that impose different requirements in the MAC layer for preferential service. Real-time audio and video applications require fixed upper delay bounds whereas file transfer applications prefer higher-capacity channels. Clearly what is referred to a differentiation of services (it can also be found as classes of service *CoS*, or quality of service *QoS*) is a crucial factor that constitutes an important research area. In particular, with our receiver-initiated approach, supporting *Qos* can be accomplished by controlling the polling rate of nodes based on the service to be provided. For instance, we can arrange that a certain application be of higher preference (priority) and therefore our polling control packets are sent at a higher rate. Clearly, a link layer scheduling protocol can also be used on top of our receiver-initiated protocols to provide fair allocation of the medium or preferential service to prime peers. *QoS* provisioning on top of receiver initiated MAC protocols is a promising area for future work.

In the same context, depending on the definition of the polling function at the receiver, we can also address the issue of conserving power. Although new battery technologies are constantly evolving producing smaller but more powerful products, wireless radios can drain even a heavy and awkward battery in less than 30 minutes. Due to the fact that a radio transceiver when not transmitting a data packet remains by default on a receiving (listening) status, large amounts of

energy can be wasted even when there is no activity in the channel for long periods. Designing and analyzing receiver initiated MAC protocols that take into account power presentation is also a area for future research, because our receiver-initiated approach can realize a two-sided advantage: first, our single-channel or multi-channel MAC protocol uses the least number of control packets to achieve collision-avoidance; second, it is fairly easy to configure a radio that uses our receiver-initiated MAC protocols so that the polling function decides the activity of the transceiver based on the users request. For instance, if a user knows that there is very slight probability that somebody else might be trying to establish a communication, it can set the polling function so that the radio is activated every 5 minutes for just a few seconds to check whether there is any activity and if so respond, and otherwise continue to be on a *energy saving* mode of operation.

Bibliography

[1] *FCC Rules and Regulations, Part15.247 and 15.249.* October 1997.

[2] *WINGS and SPARROW Projects.* http://www.cse.ucsc.edu/research/ccrg/, Santa Cruz, CA, 1998.

[3] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.* IEEE Standard 802.11, June 1999.

[4] N. Abramson. The ALOHA Systems - another alternative for computer communications. In *AFIPS Conference Proceedings 1970 Fall Joint Computer Conference*, pages 281–5, 1970.

[5] N. Abramson. Multiple Access in Wireless Digital Networks. *Proceedings of the IEEE*, 82(9):1360–69, September 1994.

[6] N. Amitay. Resource Auction Multiple Access (RAMA): Efficient Method for Fast Resource Assignment in Decentralized Wireless PCS. *Electronic Letters*, 28:799–801, April 1992.

[7] N. Amitay. Distributed switching and control with fast resource assignment/handoff for personal communications systems. *IEEE journal on Selected Areas in Communications*, 11:842–9, August 1993.

[8] D. J. Baker and A. Ephremides. The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm. *IEEE Transactions on Communications*, 29(11):1694–1701, November 1981.

[9] D. J. Baker, A. Ephremides, and J. Flynn. The Design and Simulation of a Mobile Radio Network with Distributed Control. *IEEE Journal on Selected Areas of Communications*, 2(1), January 1984.

[10] S. Bellini and P. Borgonovo. On the throughput of an ALOHA channel with variable length packets. *IEEE Transactions on Communications*, 28:1932–35, November 1973.

[11] A. Bertossi and M. A. Bonuccelli. Code Assignment for Hidden Terminal Interference Avoidance in Multihop Packet Radio Networks. *IEEE/ACM Transactions on Networking*, 3(4):441–9, August 1995.

[12] D. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall, Englewood Cliffs, NJ, 2nd edition, 1992.

[13] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wireless LAN's. In *Proceedings ACM SIGCOMM*, pages 212–25, London, UK, August 1994.

[14] J. I. Capetanakis. Tree Algorithms for Packet Broadcast Channels. *IEEE Transactions on Information Theory*, 25:505–15, September 1979.

[15] K.-C. Chen. Medium Access Control of Wireless LANs for Mobile Computing. *IEEE Networks*, 8:50–63, August 1994.

[16] I. Chlamtac and A. Farago. Making Transmission Schedules Immune to Topology Changes in Multi-Hop Packet Radio Networks. *IEEE/ACM Transactions on Networking*, 2(1):23–29, February 1994.

[17] I. Chlamtac, A. Farago, and H. Zhang. Time-Spread Multiple-Access (TSMA) Protocols for Multihop Mobile Radio Networks. *IEEE/ACM Transactions on Networking*, 5(6):804–12, December 1997.

[18] I. Chlamtac and W. R. Franta. A Multiaccess Protocol for Multihop Radio Networks. *IEEE Transactions on Communications*, 33(10):1067–75, October 1985.

[19] I. Chlamtac and S. Kutten. A Spatial-Reuse TDMA/FDMA for Mobile Multi-Hop Radio Networks. In *Proceedings IEEE INFOCOM*, pages 389–94, Washington, DC, March 1985.

[20] I. Chlamtac and A. Lerner. A Link Allocation Protocol for Mobile Multihop Networks. In *Proceedings IEEE GLOBECOM*, pages 238–42, New Orleans, LA, December 1985.

[21] I. Chlamtac and A. Lerner. Link Allocation in Mobile Radio Networks with Noisy Channel. In *Proceedings IEEE INFOCOM*, pages 641–8, Miami, Fl, April 1986.

[22] I. Chlamtac and A. Lerner. Fair Algorithms for Maximal Link Activation in Multihop Radio Networks. *IEEE Transactions on Communications*, 35(7):739–46, July 1987.

[23] I. Chlamtac and S. S. Pinter. Distributed Nodes Organization Algorithm for Channel Access in a Multihop Dynamic Radio Network. *IEEE Transactions on Computers*, 36(6):728–37, June 1987.

[24] A. M. Chou and V. O. K. Li. Fair Spatial TDMA Channel Access Protocols for Multihop Radio Networks. In *Proceedings IEEE INFOCOM*, pages 1064–73, April 1991.

[25] Israel Cidon and Moshe Sidi. Distributed Assignment Algorithms for Multihop Packet Radio Networks. *IEEE Transactions on Computers*, 38(10):1353–61, October 1989.

[26] A. Colvin. CSMA with Collision Avoidance. *Computer Communications*, 6(5):227–35, 1983.

[27] G.R. Cooper and C.D. McGillem. *Modern Communications and Spread Spectrum*. McGraw-Hill, 1986.

[28] W. Crowther and et al. A system for broadcast communications: reservation ALOHA. In *Proceedings 6th Hawaii International System Science Conference*, January 1973.

[29] D. H. Davis and S. A. Gronemeyer. Performance of slotted ALOHA random access with delay capture and randomized time of arrival. *IEEE Transactions on Communications*, 28(5):703–10, May 1980.

[30] R. C. Dixon. *Spread Spectrum Systems.* Wiley, 1976.

[31] A. Ephremides, J. E. Wieselthier, and Dennis J. Baker. A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling. *Proceedings of the IEEE*, 75(1):56–73, January 1987.

[32] A. Ephrreamides and T. Truong. Scheduling Broadcasts in Multihop Radio Networks. *IEEE Transactions on Communications*, 38(4):465–60, April 1990.

[33] N. Figueira and J. Pasquale. Remote-queuing multiple access (RQMA): Providing Quality of Service for wireless communications. In *Proceedings IEEE INFOCOM*, April 1998.

[34] C. L. Fullmer. *Collision Avoidance Techniques for Packet radio Networks.* Ph.D. Thesis, UC Santa Cruz, 1998.

[35] C. L. Fullmer and J. J. Garcia-Luna-Aceves. Floor Acquisition Multiple Access for Packet-Radio Networks. In *Proceedings ACM SIGCOMM*, Cambridge, MA, September 1995.

[36] C. L. Fullmer and J J. Garcia-Luna-Aceves. Solutions to Hidden Terminal Problems in Wireless Networks. In *Proceedings ACM SIGCOMM*, Cannes, France, September 1997.

[37] N. Funabiki and Y. Takefuji. A Parallel Algorithm for Broadcast Scheduling Problems in Packet Radio networks. *IEEE Transactions on Communications*, 41(6):828–31, June 1993.

[38] R. G. Gallager. Conflict Resolution in Random Access Broadcast Networks. In *Proceedings AFOSR Workshop Commun. Theory Appl.*, Provincetown, MA, September 1978.

[39] R. Garces. *CARMA: Collision Avoidance and Resolution Multiple Access.* Ph.D. Thesis, UC Santa Cruz, 1999.

[40] R. Garces and J. J. Garcia-Luna-Aceves. Floor Acquisition Multiple Access with collision resolution. In *Proceedings ACM/IEEE MobiCom*, New York, November 1996.

[41] J. J. Garcia-Luna-Aceves and A. Tzamaloukas. Receiver Initiated Collision-Avoidance Protocols for Wireless Networks. In *To appear in the ACM journal Wireless Networks (WINET), Special Issue from best papers of MobiCom '99 on ad-hoc networks, 2001.*

[42] J. J. Garcia-Luna-Aceves and A. Tzamaloukas. Reversing the Collision-Avoidance Handshake in Wireless Networks. In *Proceedings ACM/IEEE MobiCom '99*, Seattle, Washington, August 1999.

[43] P. D. Gerakoulis, T. N. Saadawi, and D. L. Schilling. A Channel Access Protocol for Embedding CSMA on Spread Spectrum Packet Radio Networks. *Proceedings of IEEE ICC*, 1988.

[44] E. Geraniotis and M. Pursley. Error Probability for Direct Sequence Spread Spectrum Multiple Access Communication - Part II:Approximations. *IEEE Transactions on Communications*, COM-30, May 1982.

[45] D. J. Goodman, R. Valenzuela, K. T. Gayliard, and B. Ramamurthi. Packet Reservation Multiple Access for Local Wireless Communications. *IEEE Transactions on Communications*, 37:885–90, August 1989.

[46] Z. Haas and J. Deng. Dual Busy Tone Multiple Access (DBTMA): a New Medium Access Control for Packet Radio Networks. In *IEEE 1998 International Conference on Universal Personal Communications*, Florence, Italy, Octomber 1998.

[47] B. Hajek and G. Sakaki. Link Scheduling in Polynomial Time. *IEEE Transactions on Infomation Theory*, 34(9):910–7, Sepetember 1988.

[48] L. Hu. Distributed Code Assignments for CDMA Packet Radio Networks. *IEEE/ACM Transactions on Networking*, 1(6):668–77, December 1993.

[49] MIL3 Incorporation. External Model Access. *External Interfaces*.

[50] Special issue on packet radio networks. *Proceedings of the IEEE*, 66(11), November 1978.

[51] S. Jiang and T. Hsiao. Performance Evaluation of a Receiver-Based Handshake Protocol for CDMA Networks. *IEEE Transactions on Communications*, 43, 1995.

[52] M. Joa-Ng and I. Lu. Spread Spectrum Medium Access Protocol with Collision Avoidance in Mobile Ad-Hoc Wireless networks. In *Proceedings IEEE INFOCOM 99*, San Francisco, California, April 1999.

[53] J.-H. Ju and O. K. Li. An Optimal Topology-Transparent Scheduling Method in Multihop Packet Radio Networks. *IEEE/ACM Transactions on Networking*, 6:298–306, June 1998.

[54] R. E. Kahn, J. Gronemeyer, J. Burchfiel, and R. C. Kunzelman. Advances in Packet Radio Networks. *Proceedings of IEEE*, 66, November 1978.

[55] P. Karn. MACA - a New Channel Access Method for Packet Radio. In *Proceedings ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, New York, April 1990.

[56] L. Kleinrock and S. S. Lam. Packet Switching in Radio Channels: new conflict-free multiple access scheme. *Proceedings of the IEEE*, 75(1):156–67, January 1987.

[57] L. Kleinrock and F. A. Tobagi. Packet Switching in Radio Channels: Part I - Carrier Sense Multiple-Access Modes and their Throughput-Delay Characteristics. *IEEE Transactions on communications*, 23(12):1400–16, December 1975.

[58] S. S. Lam. Packet Radio Networks - A Performance Analysis of the R-ALOHA Protocol. *IEEE Transactions on Computers*, 29(7):596–603, July 1980.

[59] B. M. Leiner, D. L. Nielson, and Tobagi F. A. *Proceedings of the IEEE*, 75(1), January 1987.

[60] R. Lupas and S. Verdu. Near-Far Resistance of Multiuser Detectors in Asynchronous Channels. *IEEE Transactions on Communications*, IT-38(4):496–508, April 1990.

[61] T. Makansi. Transmitted-Oriented Code Assignment for Multihop Packet Radio. *IEEE Transactions on Communications*, 35(12):1379–82, December 1987.

[62] J. L. Massey. Collision resolution algorithm and random access communications. In *Multiuser Communication Systems*, pages 73–137, Edited by G. Longo, Spriger-Verlag, New York, 1981.

[63] A. Muir and J. J. Garcia-Luna-Aceves. Supporting real-time multimedia traffic in a wireless lan. In *Proceedings SPIE Multimedia Computing and Networking 1997*, San Jose, CA, February 1997.

[64] A. Muir and J. J. Garcia-Luna-Aceves. A channel access protocol for multihop wireless networks with multiple channels. In *Proceedings IEEE ICC '98*, Atlanta, GE, June 1998.

[65] S. Nanda. Analysis of packet reservation multiple access: voice and data integration for wireless networks. In *Proceedings IEEE GLOBECOM*, pages 1984–8, 1990.

[66] S. Nanda, D. J. Goodman, and U. Timor. Performance of PRMA: A packet voice protocol for cellular systems. *IEEE Transactions on Vehicular Technology*, 40:584–98, July 1991.

[67] R. Nelson and L. Kleinrock. Spatial TDMA: a Collision Free Multi-Hop Channel Access Protocol. *IEEE Transactions on Communications*, 33(9):934–44, Sepetember 1985.

[68] R. G. Ogier. A decomposition method for optimal link scheduling. In *Proceedings 24th Allerton Conference*, pages 822–823, Monticello, IL, October 1986.

[69] L. Pond and O. K. Li. A Distributed Time-Slot Assignment Protocol for Mobile Multi-Hop Broadcast Packet Radio Networks. In *Proceedings IEEE MILCOM*, pages 3.6.1–2.6.5, October 1989.

[70] M. J. Post, P. E. Sarachik, and A. S. Kershenbaum. A biased greedy algorithm for scheduling and multi-hop radio networks. In *Proceedings Conference Information Science Systems*, pages 564–672, Baltimore, MD, 1985.

[71] M. J. Post, P. E. Sarachik, and A. S. Kershenbaum. A distributed evolutionary algorithm for reorganizing network communications. In *Proceedings IEEE MILCOM*, pages 133–139, October 1985.

[72] M. B Pursley. Performance Evaluation for Phase Coded Spread Spectrum Multiple Access Communication - Part I: System Analysis. *IEEE Transactions on Communications*, COM-25, August 1977.

[73] M. B. Pursley. Frequency Hopping Transmission for Satellite Packet Switching and Terestial Packet Radio Networks. *IEEE Transactions Information Theory*, IT-32:652–667, May 1986.

[74] M. B. Pursley. The Role of Spread Spectrum in Packet Radio Networks. *Proceedings of the IEEE*, 75(1):116–134, January 1987.

[75] H. Qi and R. Wyrwas. Markov Analysis for PRMA Performance Study. In *Proceedings IEEE VTC*, pages 1184–8, 1994.

[76] X. Qiu and O. K. Li. Dynamic Reservation Multiple Access (DRMA): a new Multiple Access Scheme for Personal Communication System (PCS). *Wireless Networks*, 2:117–28, 1996.

[77] X. Qiu and O. K. Li. On capacity of packet reservation multiple access with capture in personal communications systems. *IEEE Transactions on Vehicular Technology*, 45:666–75, October 1996.

[78] X. Qiu and O. K. Li. A unified Performance Model for Reservation-Type Multiple-Access Schemes. *IEEE Transactions on Vehicular Technology*, 47:173–89, February 1998.

[79] J. Raju and J. J. Garcia-Luna-Aceves. Distributed Assignment of Codes for Multihop Packet Radio Networks. In *Procceedings of IEEE MILCOM '97*, Monterey, California, November 1997.

[80] S. Ramanathan and E. L. Lloyd. Scheduling Algorithms for Multihop Radio Networks. *IEEE/ACM Transactions on Networking*, 1(2):166–77, April 1993.

[81] R. Ramaswami and K. K. Parhi. Distributed scheduling of broadcast in a radio network. In *Proceedings IEEE INFOCOM*, pages 497–504, 1989.

[82] R. Rom. Collision Detection in Radio Channels. pages 235–49, Computer Science Press, April 1986.

[83] R. Rom and M. Sidi. *Multiple Access Protocols Performance and Analysis*. Springer-Verlag, 1990.

[84] I. Rubin and S. Shambayati. Performance evaluation of a reservation multiple access scheme for packetized wireless systems with call control. In *Proceedings IEEE GLOBECOM*, pages 16–20, 1992.

[85] N. Shacham. Stochastic Models for Multihop Packet Radio Networks. In *Stochastic analysis of computer and communication system*, pages 733–65, Edited by H. Takagi, North-Holland, 1990.

[86] N. Shacham and J. B. King. Architectures and Performance of Multichannel Multihop Packet Radio Networks. *IEEE Journal on Selected Ares in Communications*, 5(6), July 1987.

[87] K. Sohraby, M. L. Molle, and A. N. Venetsanopoulos. Comments on throughput analysis for persistent csma systems. *IEEE Transactions on Communications*, COM-31:123–129, January 1987.

[88] E. S. Sousa and J. A. Silvester. Spreading Code Protocols for Distributed Spread Spectrum Packet Radio Networks. *IEEE Transactions on Communications*, 36, March 1988.

[89] D. S. Stevens and M. H. Ammar. Evaluation of Slot Allocation Strategies for TDMA Protocols Packet Radio Networks. In *Proceedings IEEE INFOCOM*, 1990.

[90] H. Takagi and L. Kleinrock. Output Processes in Contention Packet Broadcasting Systems. *IEEE Transactions on Communications*, 33(11):1191–9, November 1985.

[91] F. Talucci and M. Gerla. MACA-BI (MACA by invitation). A Wireless MAC Protocol for High Speed Ad Hoc Networking. In *Proceedings IEEE ICUPC*, 1997.

[92] F. Talucci, M. Gerla, and L. Fratta. MACA-BI (MACA by invitation) - A Receiver Oriented Access Protocol for Wireless Multihop Networks. In *Proceedings IEEE PIMRC*, 1997.

[93] S. Tasaka. Stability and performance of the R-ALOHA packet broadcast system. *IEEE Transactions on Computers*, 32(8):717–26, August 1983.

[94] F. A. Tobagi. Multiaccess protocols in Packet Communications Systems. *IEEE Transactions on Communications*, 28(4):468–488, April 1980.

[95] F. A. Tobagi and L. Kleinrock. Packet Switching in Radio Channels: Part II - the Hidden Terminal Problem in Carrier Sense Multiple-Access Modes and the Busy-Tone Solution. *IEEE Transactions on communications*, 23(12):1417–33, December 1975.

[96] F. A. Tobagi and L. Kleinrock. Packet Switching in Radio Channels: Part III - Polling and (dynamic) Split Channel Reservation Multiple Access. *IEEE Transactions on Computers*, 24(7):832–45, August 1976.

[97] F. A. Tobagi and L. Kleinrock. The Effect of Acknowledgment Traffic on the Capacity of Packet-Switched Radio Channels. *IEEE Transactions on Communications*, 26(6):815–26, June 1978.

[98] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. Poll-before-Data Multiple Access. In *Proceedings IEEE International Communications Conference (ICC '99)*, Vancouver, Canada, June 1999.

[99] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. The Effect of Exerting Adequate Persistence in Collision Avoidance Protocols. In *Proceedings IEEE Mobile Multimedia Communications (MoMuC '99)*, San Diego, California, November 1999.

[100] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. A Channel-Hopping Protocol for Ad-Hoc Networks. In *Proceedings IEEE International Conference on Computer Communication and Network (IC3N '00)*, Las Vegas, Nevada, October 2000.

[101] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. Channel-Hopping Multiple Access. In *Proceedings IEEE International Communications Conference (ICC '00)*, New Orleans, Louisiana, June 2000.

[102] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. Channel-Hopping Multiple Access with Packet Trains for Ad Hoc Networks. In *Proceedings IEEE Mobile Multimedia Communications (MoMuC '00)*, Tokyo, Japan, October 2000.

[103] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. Receiver-Initiated Channel-Hopping for Ad-Hoc Networks. In *Proceedings IEEE Wireless Communications Networking Conference (WCNC '00)*, Chicago, Illinois, September 2000.

[104] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. A Receiver-Initiated Collision-Avoidance Protocol for Multi-Channel Networks. In *Proceedings IEEE INFOCOM '01*, Anchorage, Alaska, April 2001.

[105] M. K. Varanasi and B. Aazhang. Multistage Detection in Asynchronous Code-Division Multiple-Access Communications. *IEEE Transactions on Communications*, COM-38(4):509–519, April 1990.

[106] C. Wu and V. O. K. Li. Receiver-initiated busy-tone multiple access in packet radio networks. In *ACM SIGCOMM Workshop*, Stowe, VT, August 1987.

[107] W. Xu and G. Campell. A Distributed Queuing Random Access Protocol for a Broadcast Channel. In *Proceedings ACM SIGCOMM*, San Francisco, CA, September 1993.

[108] Y. Ye, C.-J. Hou, and C.-C. Han. QGMA: A new MAC protocol for supporting QoS in wireless local area networks. In *Proceedings IEEE ICNP*, pages 339–48, Austin,TX, 1998.

[109] C. Zhu and M. S. Corson. A Five Phase Reservation Protocol (FPRP) for Mobile Ad Hoc Networks. In *Proceedings IEEE INFOCOM*, October 1998.