Channel Hopping Multiple Access with Packet Trains for Ad Hoc Networks

Asimakis Tzamaloukas J.J. Garcia-Luna-Aceves
Computer Engineering Department
Baskin School of Engineering
University of California, Santa Cruz, California 95064
E-mail: jamal, jj@cse.ucsc.edu

Abstract— **We present a new medium-access control protocol for ad hoc networks that does not require carrier sensing or the pre-assignment of unique codes to nodes to ensure that intended receivers receive unicast, or multicast, or broadcast data packets without interference from hidden sources. We call this new protocol channel hopping access with trains (CHAT). CHAT combines the notion of packet trains with synchronous channel hopping to improve channel utilization. We compare CHAT against two of the most efficient protocols proposed to date based on the pre-assignment of codes (MACA-CT), or channel hopping with no predefined code assignment (CHMA) via simulations. The results show that CHAT provides considerable improvement in the throughput of an ad hoc network for unicast traffic, broadcast traffic and mixed traffic consisting of both unicast and broadcast transmissions. CHAT is applicable to ad hoc networks based on commercial off-the-shelf spread spectrum radios operating in unlicensed frequency bands.**

Keywords— **Medium Access Protocols, ad-hoc networks, packet train, multichannel radio, frequency hopping spread spectrum**

I. Introduction

Medium-access control (MAC) protocols based on collision avoidance have been widely used in wireless LANs and ad hoc networks mainly due to their simplicity and good performance compared to carrier sensing multiple access (CSMA). With a collision-avoidance MAC protocol, a node 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. Several variations of this scheme have been developed since SRMA (split-channel reservation multiple access) was first proposed by Kleinrock and Tobagi [7], including IEEE 802.11 [1]. Fullmer and Garcia-Luna-Aceves [2] showed that, in order to avoid data packets from colliding with any other packets at the intended receivers in networks with a single channel, the senders had to sense the channel before sending their RTSs. More recently, receiver-initiated collision-avoidance protocols have also been proposed for single-channel networks, in which the receiver initiates the collision-avoidance handshake [3]; these receiver-initiated collision-avoidance protocols also require carrier sensing to ensure correct collision avoidance.

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. Most commercial radios do not provide true carrier sensing, and direct sequence spread-spectrum (DSSS) radios may capture none or one of multiple overlapping transmissions 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.

Sousa and Silvester [6] presented and analyzed various spreading-code protocols that are sender-, receiver- or sender-receiver based, i.e., in which codes are pre-assigned to senders, receivers, or combinations. 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, receivers or a combination of the two. MACA-CT [5] uses collision-avoidance handshakes over a common channel and a transmitter-oriented data channel pre-assigned to each node to avoid collisions of data packets; MACA-CT is a good representative of collision-avoidance solutions that eliminate the need for carrier sensing at the expense of requiring unique channel assignments.

Code assignment protocols can be implemented using DSSS or FHSS radios. However, most of the commercial DSSS radios today use only 11 chips per bit, which precludes the use of code division multiple access (CDMA). On the other hand, up to 15 FHSS radios can be co-located with minimum interference problems according to the FCC regulations. Hence, slow frequency hopping is an attractive approach to achieve multiple orthogonal channels in ad hoc networks. The limitation of protocols based on the pre-assignment of codes is that senders and receivers have to find each others' codes before communicating with one another.

We introduced the channel-hopping multiple access (CHMA) protocol [8] to eliminate the need for pre-assigning codes to nodes in order to avoid hidden terminal interference in multi-channel networks without using carrier sensing. However, a limitation of CHMA and all prior MAC protocols based on collision-avoidance handshakes is that a source cannot transmit a packet destined to multiple receivers and guarantee that all the receivers can hear the packet without experiencing hidden-terminal interfer-

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under grant F30602-97-2-0338.

Report Documentation Page

Form Approved
OMB No. 0704-0188

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

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Channel Hopping Multiple Access with Packet Trains 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**		**6**	

ence.

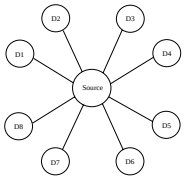
This paper introduces the Channel Hopping Access with Trains (CHAT) protocol. CHAT is the first MAC protocol based on collision avoidance that (a) does not require carrier sensing to eliminate hidden terminal interference, and (b) guarantees that unicast and broadcast data packets can be transmitted without collisions. Section II describes the operation of CHAT in detail. Section III proves that, in the absence of fading, CHAT protocol provides correct floor acquisition in a multi-hop network. Section IV presents the results of simulation experiments used to compare the throughput achieved with CHAT against CHMA and MACA-CT. Section V presents our conclusions.

II. CHAT

CHAT enhances the control handshake introduced in CHMA [8] to allow collision-free transmissions of packet trains, multicast packets, and broadcast packets. The basic operation for CHAT is shown in Fig. 2. 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 channel hop 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 familiar with 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 is not aware 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 hop. 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 1. 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. 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. On the other hand, node $D6$ never receives the SRTS and therefore no CTS is sent to node $Source$ in response. 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.



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			

Fig. 1. 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.

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

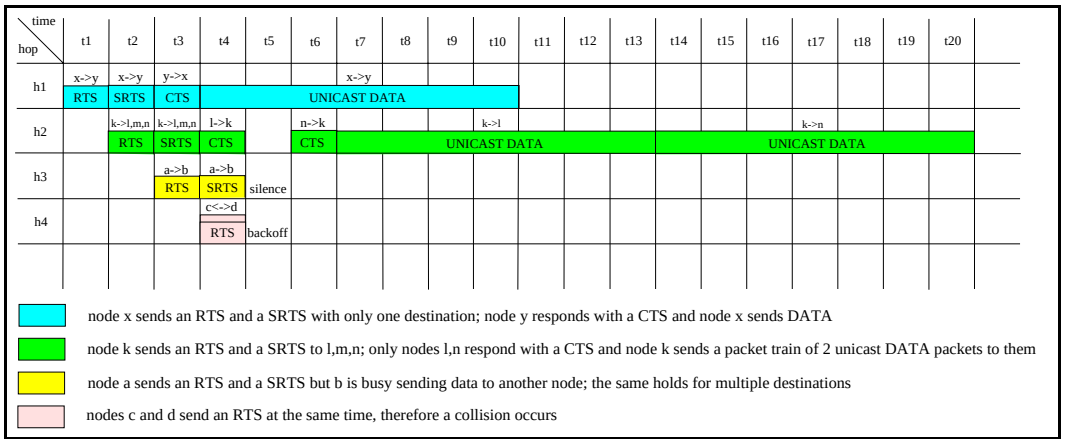


Fig. 2. CHAT with unicast only data packet exchange illustrated

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. 2, all the nodes start at time $t1$ from hop $h1$. At time $t1$ the system is at hop $h1$ 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.

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 hop $h4$. 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. 3, 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.

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 hops 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.

When multiple RTSs are transmitted within a one-way propagation delay a collision takes place and the nodes involved have to back-off. Node x determines that its RTS was not received correctly by z after a time period equal to one hop. 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 involved nodes 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.

III. Correct Collision Avoidance

Theorem 1 below shows that CHAT ensures that there are no collisions between data packets and any other trans-

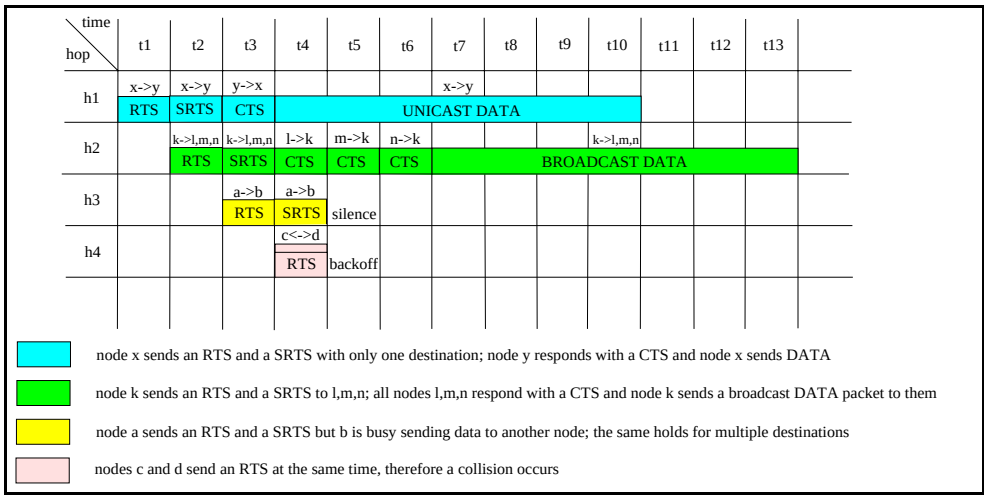


Fig. 3. CHAT with unicast and broadcast data packet exchange illustrated

missions. The following assumptions are made [2]:

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

A1) All nodes execute CHAT correctly over a perfect channel.

A2) The maximum end-to-end propagation time is τ, 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, $2\tau < \gamma \leq \delta < \infty$.

A3) The dwell time in each hop is equal to the time needed to transmit an RTS (or CTS) plus the maximum end-to-end propagation time.

Theorem 1: 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 hop 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 hop 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. □

IV. 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 [4] 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.

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. If a node fails to initiate a handshake after seven retransmissions, the data packet is dropped from the head of the queue.

Figure 4 shows the various topologies used in the experiments. Figure 4(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 4(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(c) a multi-hop network of sixteen nodes in a four di-

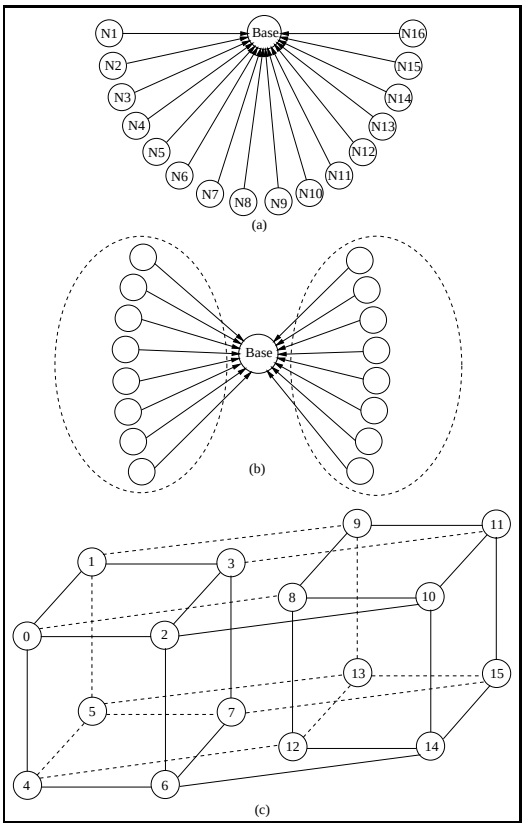


Fig. 4. Various network topologies used in the simulations

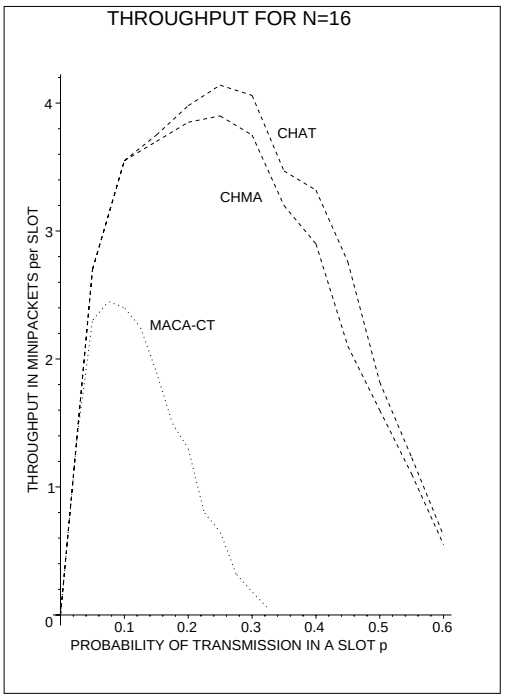


Fig. 5. Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 4(b); the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

mensional hypercube configuration is depicted. The lines between the nodes show the connectivity in the network. These topologies were chosen for two reasons: to compare with similar topologies used in prior work on collision avoidance [2], and to test the performance of the protocols under widely different conditions. Notice also that even though with topologies 4(a) and 4(b) a packet train consists of two or more data packets with the same destination (the base station), with topology 4(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 4. Figure 5 shows the throughput results measured for CHAT, CHMA and MACA-CT for the network topology shown in Fig. 4(b) for unicast traffic only. Similar, results were obtained for the other two topologies as well. As also shown with analysis in [8] 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%. When the data rate is higher than what the radio can deliver, packets are lost and the performance of CHAT degrades to that of CHMA.

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 4(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. The throughput results for broadcast only traffic are shown in Figure 6. 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

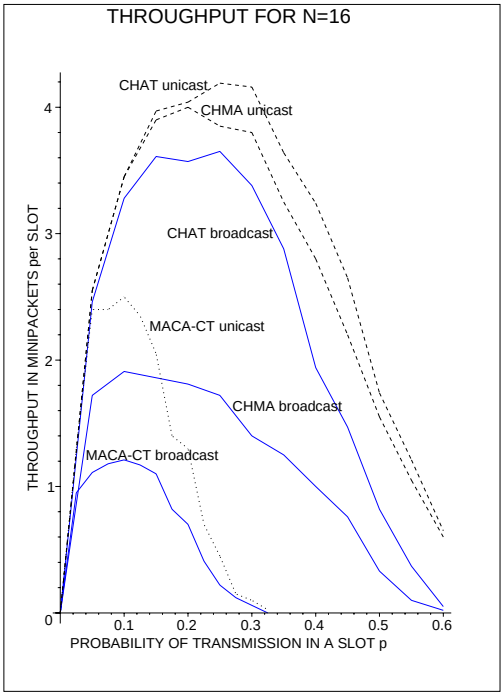


Fig. 6. Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 4(c); broadcast only traffic is compared against unicast only; the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

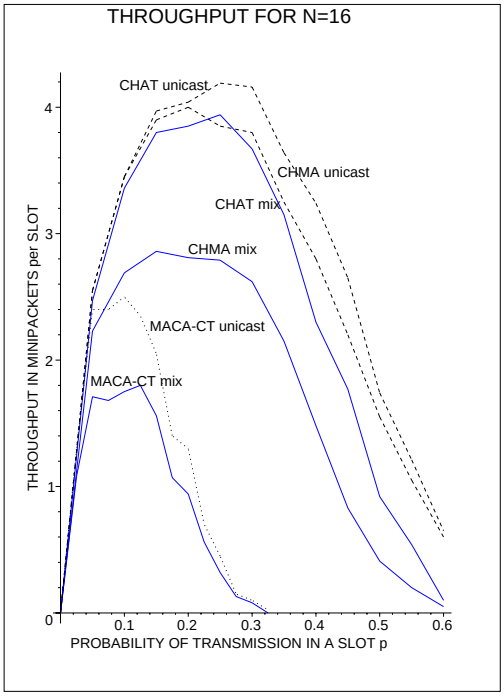


Fig. 7. Aggregate throughput for CHAT, CHMA and MACA-CT for the topology in Fig. 4(c); a mix of broadcast and unicast traffic is compared against unicast only; the number of nodes is N = 16 and the packet length is L = 10 or approximately 150 bytes

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. With CHAT a node can broadcast a data packet with 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 unicast only traffic.

Similar conclusions can be drawn when a mix of broadcast and unicast traffic is generated. As can be seen in Figure 7 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.

V. Conclusions

We introduced a channel access control protocol that can be used in any commercially available, wireless radio 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 [8] and MACA-CT [5], 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.

References

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

[2] 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.

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

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

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

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

[7] 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.

[8] 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.