CAM-MAC: A Cooperative Asynchronous Multi-Channel MAC Protocol for Ad Hoc Networks

Tie Luo, Mehul Motani, Vikram Srinivasan
Department of Electrical and Computer Engineering
National University of Singapore
Email: {tie, motani, elevs}@nus.edu.sg

***Abstract*— Medium access control (MAC) protocols have been studied under different contexts for several years now. In all these MAC protocols, nodes make independent decisions on when to transmit a packet and when to back-off from transmission. In this paper, we introduce the notion of node cooperation into MAC protocols. Cooperation adds a new degree of freedom which has not been explored before. Specifically we study the design of cooperative MAC protocols in an environment where each node is equipped with a single transceiver and has multiple channels to choose from. Nodes cooperate by helping each other select a free channel to use. We show that this simple idea of cooperation has several qualitative and quantitative advantages. Our cooperative asynchronous multi-channel MAC protocol (CAM-MAC) is extremely simple to implement and, unlike other multi-channel MAC protocols, is naturally asynchronous. We conduct extensive simulation experiments. We first compare CAM-MAC with IEEE 802.11b and a version of CAM-MAC with the cooperation element removed. We use this to show the value of cooperation. Our results show significant improvement in terms of number of collisions and throughput for CAM-MAC. We also compare our protocol with MMAC and SSCH and show that CAM-MAC significantly outperforms both of them.**

I. Introduction

Multiple channelization adds one more degree of freedom to wireless communications. The immediate benefit that can be reaped is an increase in spatial reuse by accommodating more simultaneous transmissions than is possible in single-channel wireless networks [1]–[4]. Thus, aggregate network throughput can potentially be increased.

However, other problems surface with the use of multiple channels. With a single channel, MAC protocols need only decide *when* it is suitable for communication to reduce the likelihood of collisions. These protocols include TDMA, ALOHA, CSMA and IEEE 802.11 families. On the other hand, when multiple channels are available, a sender-receiver pair must be synchronized to a common idle channel before attempting data transmission. This is due to a hardware limitation where a transceiver cannot be tuned to more than one channel simultaneously. Another problem is the multi-channel hidden terminal problem [2], where a rendezvous channel is reserved for nodes to negotiate a correct data channel. This happens if a node pair A-B does not overhear the negotiation between another pair C-D. A collision will result if A-B chooses the same data channel as C-D. This is induced by the same hardware limitation since a transceiver tuned to some channel is blind to all other channels.

In ad hoc networks, one solution is to make nodes more "powerful" by equipping each node with additional transceivers, so that it is aware of transactions on other channels when transmitting or receiving. This category of work includes [1], [3], [5]–[8]. However, multiple transceivers result in increased cost, size, and energy consumption of devices. All these are hostile to battery-powered devices like Personal Digital Assistants (PDAs) and wireless sensors. Therefore, a single-transceiver solution is more appealing. Unfortunately, current solutions using single transceivers rely on clock synchronization [2], [9]–[12] . This is known to be extremely hard and imposes significant overhead on a network, rendering a network not scalable [13].

We propose an asynchronous multi-channel MAC protocol using a single transceiver. The asynchronous mode of operation is made feasible through the introduction of *cooperation*.

We make a key observation as illustrated in Fig. 1. Suppose a communication session is to be established between node A and B but these nodes have insufficient knowledge of the channel usage to select a safe (collision-free) channel. This channel usage information can be potentially acquired from idle neighbors (node C, D, E) if they maintain such information. Therefore, rather than selecting channels independently, nodes C, D and E help nodes A and B in making a good decision.

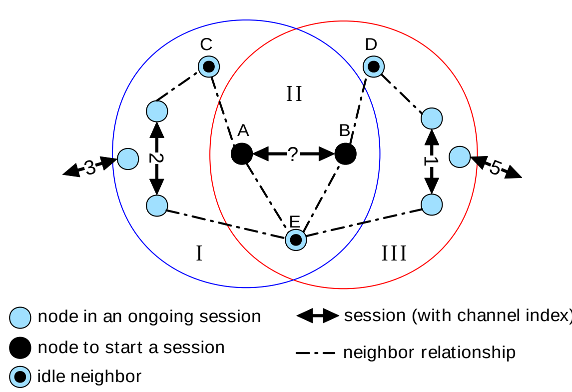


Fig. 1. The observation that leads to the cooperative solution.

Note that idle nodes naturally obtain channel usage information by overhearing transmissions in their vicinity. What is only needed is an appropriate and efficient *cooperation mechanism* to facilitate information sharing among nodes.

In this paper, we introduce cooperation into multi-channel ad hoc networks as another degree of freedom to gain the knowledge of channel conditions, thereby improving system performance. We show that by virtue of cooperation, throughput increases by more than two fold in most cases. To the best of our knowledge, this is the first multi-channel single-transceiver MAC protocol without any requirement on clock synchronization. We name our protocol Cooperative Asynchronous Multi-channel MAC (CAM-MAC).

Our main contributions are:

- Introduction of cooperation into multi-channel MAC protocols.[1]
- Design of a fully asynchronous protocol for ad hoc networks.
- Demonstration of the value of cooperation by comparing with a non-cooperative counterpart via extensive simulations.
- Demonstration of the performance of the proposed protocol by comparing with other representative multi-channel MAC protocols.

The rest of the paper is organized as follows. Sec. II reviews the related work. Sec. III presents system model and design considerations. Sec. IV describes our protocol. Sec. V presents our simulation model and results. Finally, Sec. VI concludes this paper.

II. Related Work

A. Multi-transceiver solutions

Wu et al. [8] proposed Dynamic Channel Assignment (DCA) to allocate channels on demand. Each node is equipped with two transceivers, one dedicated to exchanging control messages and the other dedicated to data messages. Its power controlled extension is found in [7]. Nasipuri et al. [3] propose a multi-channel CSMA protocol with "soft" channel reservation. It is assumed that the number of channels is equal to the number of transceivers on each node, which is very expensive. Later they extended their work to base channel selection on signal strength [6]. Jain et al. [5] propose a protocol similar to DCA [8], using one dedicated transceiver for control purposes, but they use a receiver-based channel selection strategy (comparing signal-to-noise ratios at receivers). A multi-radio unification protocol (MUP) is proposed by Adya et al. [1], using two transceivers but both for control messages and data. The two transceivers work independently, each operating an 802.11 protocol.

The key drawback of this category of protocols is the requirement of at least one more transceiver on each node, increasing both device cost and size. In addition, multiple transceivers typically incur multiple folds of energy consumption, which is especially hostile to battery-powered devices.

B. Single-transceiver solutions

So and Vaidya [2] propose a protocol called MMAC (multi-channel MAC) for ad hoc networks, in which each node has only one transceiver. 802.11 PSM (power saving mode) is assumed, time is slotted and a subslot called ATIM window is used to exchange negotiation messages for subsequent data transmissions in the same time slot. The fixed time slot necessitates a slot length accommodating the longest possible packet, which could be a waste if packets are variable-sized. A similar protocol is proposed by Chen et al. [10], but the duration of the time slot is variable. However, a common disadvantage of [2], [10] is that the duration of negotiation phase (the ATIM window in MMAC) has to be long enough to accommodate all nodes in the neighborhood, making a protocol inflexible to varying node densities.

Channel hopping is another type of scheme in this category, as adopted by CHMA [11], CHAT [12] and SSCH [9]. In [11] and [12], all nodes follow a *common* hopping sequence. A sender and a receiver stop hopping when they communicate with each other, and rejoin the common hopping sequence afterward. This scheme requires a very tight clock synchronization. Bahl et al. [9] propose SSCH (slotted seeded channel hopping) where, unlike common hopping, the number of hopping sequences is equal to the number of channels. Each node randomly adopts a sequence, and thus a conversation is successful only if the sender and the receiver hop onto the same channel. This results in extra delay. Moreover, all these channel hopping schemes have a common flaw: the performance depends heavily on *channel switching latency*. According to current commercial off-the-shelf products [14], channel switching takes 150-200 μs, about twice the transmission time of a RTS packet [15] (80μs) under the 802.11b 2Mbps rate. Consequently, networks employing channel hopping protocols are sensitive to hardware characteristics.

The common disadvantage of this category of protocols is the requirement of clock synchronization. Some even demand a tight synchronization [2], [10]–[12], which is especially difficult in a distributed environment without a central coordinator. Also, according to Tseng et al. [16], even if clock synchronization can be achieved, two partitioned network may not be able to discover each other if their schedules are out of synchronization.

The gap in existing solutions motivated us to propose CAM-MAC, which solves multi-channel problems using a single transceiver and completely eliminates synchronization, through the exploitation of cooperation.

III. System Model

We consider both multi-hop and single-hop wireless ad hoc networks, where no central entity exists to coordinate medium access and channel allocation. The bandwidth is divided into multiple orthogonal channels. Each node is equipped with a single transceiver. A transceiver can be tuned to different channels, but can only use one channel at a time.

All nodes have a common and fixed communication range that is the same as the interference range. A node receiving a signal can decode it correctly if and only if there is only one node transmitting within its communication range and both nodes have tuned to a common channel.

[1]Existing cooperative MAC protocols are all for single-channel networks and more importantly, the cooperation in that context is defined as intermediate nodes helping *relay* data for source-destination pairs.

There are one control channel and m data channels ($m \geq 1$) (a quantitative argument for this design scheme is given in Sec. III-A). The control channel is used for negotiations and, consequently, serves as a *rezendevous* for all nodes to disseminate and acquire information. Therefore it is beneficial for a node to listen to the control channel as long as it is idle.

For the convenience of description, we use the term *session* to refer to the entire process of interaction between a sender and a receiver to deliver a data packet in the MAC layer. A session is comprised of a *control session* and a *data session*, the former referring to message exchanges on the control channel, and the latter refering to message exchanges on a selected data channel. Currently, we allow a data session to transmit only one data packet.

To ensure reliability of packet delivery, an acknowledgment is sent on the selected data channel[2] to confirm receipt of a data packet. Communication on a data channel is therefore *two-way*, and thus two adjacent senders *or* receivers on the same data channel can also interfere with each other. This shows that the so-called "exposed terminal problem" does not exist in such a (actually typical) two-way communication model.[3]

In the current model nodes do not perform carrier sensing on data channels for channel selection. This is because in the multi-hop environment a sender and a receiver may have difference carrier sensing outcomes, and hence extra coordination is involved. Also, more channel switching incurs more delay. Therefore it is not conclusive whether carrier sensing is worthwhile. We would like to investigate this issue in future work.

A. Control Channel Bottleneck: An Analytical Study

A plausible counter-argument to the use of a dedicated control channel is the so-called "control channel bottleneck" problem pointed out in [2] [17], that when the number of channels is large, the control channel becomes a bottleneck as it is the only rendezvous for all nodes to negotiate their sessions. However, we will show via simple analysis this problem is not really a concern.

Consider a single-hop network where all nodes are within communication range of each other. This is the case most prone to control channel bottleneck (if it exists) because no spatial reuse is possible on the only control channel. Accordingly, by assuming collision avoidance on the control channel (backoff before initiating a control session), the average number of concurrent transmissions is upper bounded by (see Fig. 2)

$$Tx_{max} = \left\lfloor \frac{L_{data} + L_{ack}}{L_{bk} + L_{ctrl}} \right\rfloor, \quad (1)$$

[2]ACK is not sent on the control channel because, after receiving DATA on a data channel, the receiver could not realize other ongoing control sessions and thus an ACK is prone to causing collisions on the control channel.

[3]One way to avoid collision between DATA and ACK from two neighboring sessions is to let one session add an extra interval before sending ACK. This however complicates the protocol and we would like to consider it as a future improvement.

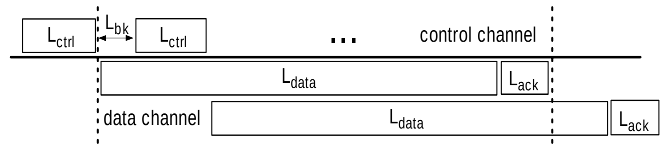


Fig. 2. Examining the control channel bottleneck.

where $L_{data}, L_{ack}, L_{bk}, L_{ctrl}$ are the average length of a DATA packet, an ACK packet, a backoff interval, and a control session (consisting of, e.g., RTS and CTS) respectively (for simplicity, units of L_{ctrl} and L_{bk} are converted to bytes equivalently). This upper bound is achieved when channels are always available, nodes are always backlogged, and no collision happens on control or data channels.

The average throughput is thus upper bounded by

$$\begin{aligned} Th_{max} &= \frac{L_{data} + L_{ack}}{L_{bk} + L_{ctrl}} \times \frac{L_{data}}{L_{data} + L_{ack}} \times C \\ &= \frac{L_{data} C}{L_{bk} + L_{ctrl}}, \end{aligned} \quad (2)$$

where C is the capacity of a data channel (assume equal capacity for all channels).

We take typical parameters for numerical illustration. Supposing $L_{data} = 2000$ bytes, $L_{ack} = 10$ bytes, $L_{ctrl} = 70$ bytes, $L_{bk} = 30$ bytes, $C^{802.11b} = 2$ Mbps and $C^{802.11a} = 54$ Mbps, we have:

$$Tx_{max} = 20$$

$$Th_{max}^{802.11b} = 40Mbps, \;\; Th_{max}^{802.11a} = 1.08Gbps.$$

These maxima are achieved when the control channel and all Tx_{max} data channels are saturated.[4] The above typical parameters indicate a maximum of 21 channels can be fully utilized, which significantly exceeds the current standard specification (IEEE 802.11b supports 3 channels, and 802.11a supports 12 channels). This number also indicates that, at the saturation point, 40 nodes (Tx_{max} pairs) transmit or receive data *concurrently* in a single-hop network, which implies a very high node density ($>> 40$ nodes) and heavy traffic load are accommodated. In addition, as we can see, the maximum throughput is remarkably high at the saturation point.

Our analysis shows that the problem of control channel bottleneck can be avoided or mitigated in practice, provided that the protocol is properly designed. We will show by simulation in Sec. V that CAM-MAC does perform well in overcoming this bottleneck problem.

B. Difficulties

In spite of the simplicity of the idea, building cooperation into a real protocol is not trivial; the following difficulties must be carefully considered in design:

- *Multiple responses*: In a cooperative protocol, in order to make a correct decision, nodes request for channel

[4]Note that, in practice, it does not necessarily mean when the number of data channels is greater than Tx_{max}, the throughput can not increase. This is because in a real network the channels may not get fully utilized due to random backoff and trade-offs on channel selection strategy. This will be observed later through our simulation.

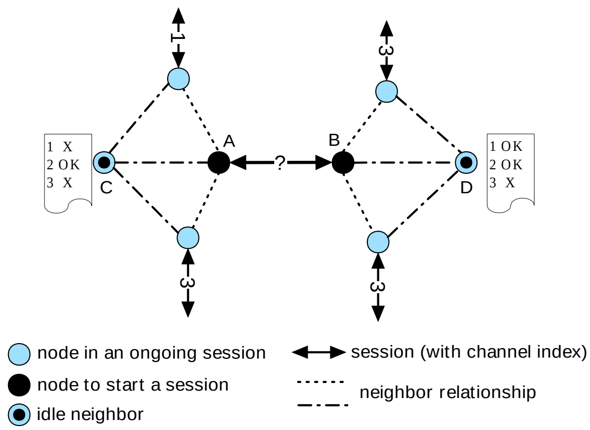


Fig. 3. A typical scenario where cooperation can help.

usage information from their neighborhood. But if every idle neighbor responds, collisions are most likely to happen. One way is to make the neighbors respond sequentially, but this is impractical. First, there is no arbitrator in an ad hoc network. Second, pre-assigning every node a response slot is very inefficient because only a subset of neighbors may respond. Another way is to use a probabilistic scheme in which each neighbor uses a probability to determine whether to respond or not. But this cannot guarantee at least one response is issued.

- *No response*: This depends on what solution is used for handling the multi-response problem. Accordingly we need to decide what decision should be made if no feedback comes.
- *Cooperative interference*: Cooperation incurs more message exchanges and correspondingly creates two new types of interferences. We call messages for establishing a session *primary* messages (e.g., RTS/CTS in 802.11), and messages for cooperation purposes *auxiliary* messages. Then one of the two interferences is the interference between primary and auxiliary messages from different sessions, and the other is the interference between auxiliary messages from difference sessions. Without a proper design, these interferences can impose deleterious effects on system performance.
- *Control channel bottleneck*: Although, as indicated in Sec. III-A, a control channel is theoretically not a bottleneck, an efficient design is necessary to overcome this problem in practice.

IV. Protocol Design

For the purpose of making the idea of cooperation more concrete, a typical scenario where cooperation may provide useful information to sender-receiver pairs is illustrated in Fig. 3. The key question is, how to make all these nodes interact for setting up a session collaboratively.

A. Overview

CAM-MAC employs a handshake shown in Fig. 4, which consists of three phases: a probing phase, a feedback phase,

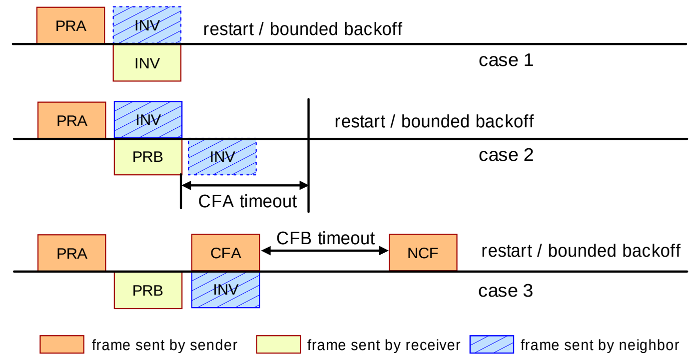


Fig. 5. Typical cooperation cases. Frames with dotted boundary are optional.

and a confirm phase (for the sake of efficiency, these phases are interleaved with each other). After sensing the control channel to have been idle for DIFS (distributed interframe space [15]) or a bounded backoff interval (explained later), a sender selects a data channel based on its own knowledge and then broadcasts a PRA (Probe-A) which carries the channel index. Upon receiving the PRA, all neighbors including the receiver verify whether the selected channel will cause a collision. If so, they will issue INVs (invalid) after SIFS (short interframe space [15]). Otherwise, the intended receiver sends a PRB (Probe-B) which replicates the channel index from PRA, and all other nodes keep silent. Subsequently, the receiver's neighbors perform verification on the PRB and may issue INV messages in the same manner, while the sender will send a CFA (Confirm-A). Finally, the receiver sends a CFB (Confirm-B) if it receives the CFA correctly, and then both nodes tune to the selected data channel and start their data session. After successful receipt of a data packet, the receiver sends an ACK and the session ends. Then the sender and the receiver tune back to the control channel. In summary, PRA and PRB comprise the probing phase, INVs form the feedback phase, and CFA and CFB form the confirm phase.

Typical scenarios are illustrated in Fig. 5 and the explanation is as follows.

- Case 1: The intended receiver finds the selected channel unsafe and sends an INV (if some other neighbors also send INVs, the resultant collision still conveys to the sender the unsafeness of the selected channel), and the sender will restart a session with another selected channel.
- Case 2: A PRB-INV collision occurs at the sender (INVs are from the sender's neighbors). The sender will not send CFA and the receiver will then get a CFA-timeout.
- Case 3: A CFA-INV collision occurs at the receiver (INVs are from the receiver's neighbors). The sender will get A CFB-timeout and thus broadcast a NCF (non-confirm), which is used to invalidate the preceding CFA.

The *bounded backoff* indicated in Fig. 4 is a technique that can be viewed as a multi-channel extension to the virtual carrier sensing (NAV) in IEEE 802.11 [15]. A sender will backoff when it finds all the data channels unavailable. This backoff duration is randomly chosen from interval $[res, res + cw]$, where res is the residual time of the session that will

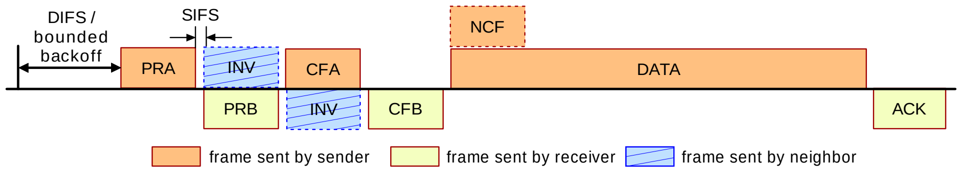


Fig. 4. CAM handshake. Frames with dotted boundary are optional.

release the channel first, and cw is a constant for perturbation. Therefore the backoff interval is *lower* bounded by res, rather than 0 as in IEEE 802.11 ([0, CW]).

The frame formats are listed in Fig.6. During every control session, a learning process occurs at every node. Senders and receivers learn session information from INV messages (an INV specifies an ongoing session (cf. Fig.6) that prevents the current session from being established due to channel conflict), and their neighbors learn session information from PRA+CFA (senders' neighbors) or PRB+CFB (receivers' neighbors), which also describe a complete session, as well as from INVs.

To consolidate the learning results, each node maintains a knowledge base called the *channel usage table* (Tab. I). Each entry specifies a session with the fields being self-explanatory. Note that the "until" field does not imply clock synchronization; it is converted to a *duration* relative to the node's *own* clock when being sent, and is converted back by calculating using the *duration* field in a received message (cf. Fig. 6).

sender	receiver	channel	until
A	B	2	11:30:52
C	D	3	11:30:56

TABLE I
CHANNEL USAGE TABLE

B. Handling Difficulties

- *Multiple responses*: Instead of requesting neighbors for a safe- or unsafe-channel list, which makes each neighbor responsible to give a feedback, a sender selects a channel first and then expects a binary feedback on the selection. Moreover, only negative answers are issued. The idea is that a positive answer is not sufficient to guarantee the correctness of a channel selection due to insufficient knowledge (the reader may refer to Fig.3), while a negative answer (INV) is indeed effective. By doing these we reduce the number of responses, and make collisions meaningful—they also invalidate the channel selection.
- *No response*: Since the protocol only allows for negative feedback, no response represents the *consent* of all neighbors on channel selection (except those busy on data channels).
- *Cooperative interference*: This is mitigated by introducing a concept of *loyal period*, during which a (cooperative) node remains silent to all messages. A node enters a loyal period when it agrees to the channel selection of a sender-receiver pair, and exits the period when the control session ends. The concept of loyal period is to protect sessions started earlier, obeying the FCFS philosophy. In addition, the bounded backoff mechanism reduces retries and thus also helps mitigate cooperative interference.
- *Control channel bottleneck*: This bottleneck is suppressed from two aspects: (1) the duration of a control session should be made as short as possible. First, four messages are *necessary* for a cooperative handshake: PRA and PRB are for probing in two different regions—communication floors of a sender and a receiver, and CFA and CFB serve as confirmation to these two regions. Second, the CAM-MAC handshake does not allocate any *dedicated* time slot for feedback messages; INVs are "embedded" into the empty intervals and thus does not prolong a control session. (2) control channel collisions should be reduced as much as possible. This is done by mitigating the cooperative interference via the loyal period and the bounded backoff mechanism.

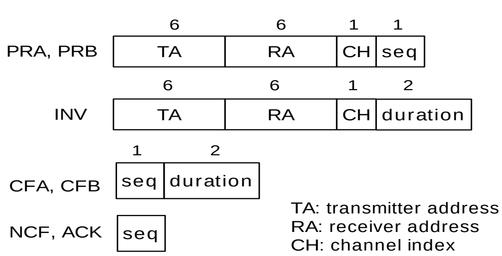


Fig. 6. Frame formats. The numbers indicate field lengths (bytes). The common parts of all messages, viz. a frame control field (2 bytes) and a FCS (frame check sequence) field (4 bytes) are omitted for simplicity.

It is worth noting that a channel selection strategy is also very important because it reduces nodes' *dependence* on cooperation, thereby mitigating cooperative interference and control channel bottleneck. Due to space limitation, we only describe a core strategy called *MRU-channel reuse*. A node's MRU channel is a node's most recently used data channel without collision. As long as this channel is not busy based on the knowledge of its previous user, it is indeed free. This is because a node always goes back to the control channel immediately after using a data channel, and hence it knows the status of its MRU channel unless it chooses another data channel afterward. Therefore, the MRU channel is given the highest priority in CAM-MAC when a node makes its initial decision.

V. EXPERIMENTAL EVALUATION

For the purpose of comparison, we also developed a non-cooperative multi-channel MAC protocol, NON-COOP, which is derived from CAM-MAC. In order to make an unbiased and meaningful comparison and to isolate the effect of cooperation, we retain all the properties of CAM-MAC in NON-COOP except removing cooperation-related interactions. In particular, nodes do not participate in other nodes' sessions, and sender-receiver pairs make decisions on their own. Nevertheless, other features such as the learning process, channel usage table, channel selection strategy, bounded backoff, etc., are all retained in NON-COOP as in CAM-MAC.

We developed our own discrete event-driven wireless network simulator for experimentation. As a baseline, IEEE 802.11 was also implemented.

A. Simulation Model

Our simulations are conducted in two types of networks.

1) **Single-hop networks**: We use a *pair-wise* configuration, that is, half of nodes are designated to be sources and the other half are designated to be destinations. All these pairs are configured during initialization and do not change throughout the simulation.
2) **Multi-hop networks**: Conventional multi-hop simulation models, in which nodes are placed in a region according to a certain distribution, often lead to askew results. This is because of *boundary effects*: nodes near the boundary are likely to have much higher throughput than nodes near the center due to much less contention. Consequently the final averaged results often leads to *higher throughput* and *lower delay* than the real situation.

   We adopt a simulation model used by Wang et al. [18] and illustrated in Fig. 7. Three concentric circles have radius of $R,\ 2R$, and $3R$ respectively, where R is the common communication range of all nodes. N nodes are placed in the innermost circular region, $3N$ nodes in the middle ring, and $5N$ nodes in the outermost ring, all subject to a uniform distribution. Consequently, this results in a multi-hop network with node density $\frac{N}{\pi R^2}$.

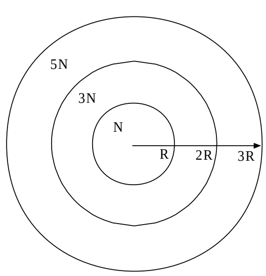


Fig. 7. Multi-hop network simulation model.

   All statistics are collected only from the N nodes in the innermost circle, while simulations are run over the entire network with $9N$ nodes. It is shown by [18] that nodes outside the outermost ring almost have no influence on the throughput of the innermost N nodes.

   Upon each packet arrival, the source node randomly chooses one of its neighbors as the destination of that packet. Accordingly, we measure MAC layer throughput rather than end-to-end throughput. The same configuration is also used by [19]–[21].

In all the simulations, nodes generate constant-bit-rate (CBR) traffic, and are always backlogged. For practicality, we only allow a single queue at each node[5], therefore a *head-of-line blocking* problem will happen in multi-hop scenarios, meaning that a head-of-line packet will block all pending packets from being sent if the receiver of the packet is busy. Nevertheless, we will see in the next subsection that CAM-MAC still achieves very good performance.

B. Simulation Results

We choose two metrics to evaluate the protocol performance:

- **Data channel collisions**: A data channel collision is defined as a collision caused by DATA/ACK packets (since they are transmitted on data channels), or a data channel is found busy just before a node starts receiving DATA/ACK on that data channel. We compute the sum of the number of data channel collisions over all nodes averaged by simulation time.
- **Aggregate throughput**: The total amount of data delivered by all nodes averaged by simulation time (measured in Mbps).

For the sake of description, we denote by Th_{cam} and $Th_{noncoop}$ the throughput of the network applying CAM-MAC and NON-COOP respectively, and by Cl_{cam} and $Cl_{noncoop}$ the corresponding data channel collisions.

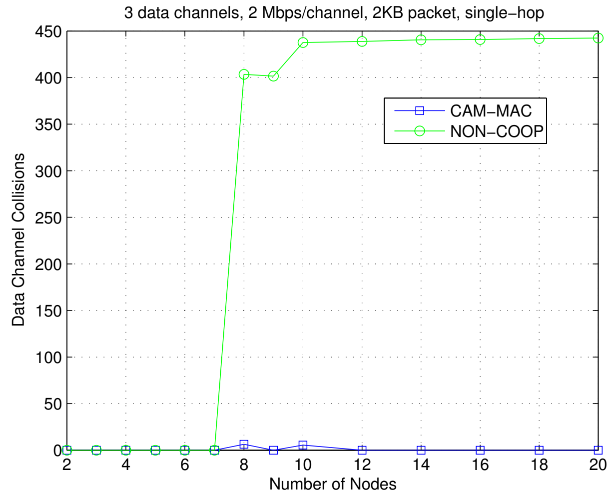
For both single-hop and multi-hop networks, the simulations are carried out for varying number of nodes, varying number of channels, and varying packet size. Each simulation is conducted for 30 seconds, and the statistics are averaged over 10 sample networks. All the other configuration parameters can be readily read from the figures given.

1) Single-hop Network:

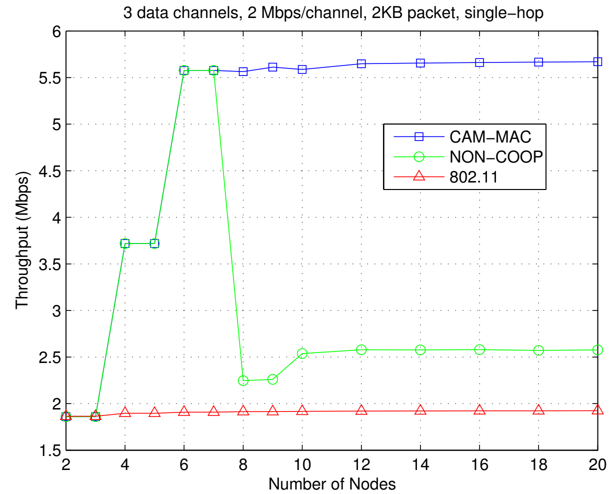
- **Varying number of nodes**: When the number of nodes $n \leq 7$, both CAM-MAC and NON-COOP have zero data channel collisions (Fig.8(a)). This is because there are not more than three pairs of nodes and there are three data channels, therefore a node is always able to select free channels, provided a proper protocol design (in particular, via learning and MRU channel reuse). However, sharp divergence of the two curves appears at $n = 8$. The reason is that more than three node pairs exist and the multi-channel hidden terminal problem arises (a node's MRU channel may be occupied by other node pairs, and that node has to select from other channels, whose current usage can be unknown to it). On the contrary, CAM-MAC continues to maintain zero collisions (except negligible collisions at $n = 8, 10$)[6] by virtue of cooperation.

[5] Some literature such as [17] assume *per-neighbor* queues at each node, which avoids *head-of-line blocking* problem and yields higher throughput and lower delay. However, this is normally impractical and not adopted by us.

[6] CAM-MAC solves multi-channel hidden terminal problem via cooperation, but when $n = 8$ or 10 cooperative neighbors may all enter data channels, making cooperation unavailable. We accordingly introduce *proactive cooperation*, allowing a node, if appropriate, to send an *alarm* message before entering data channel. But the alarm is not always sent due to efficiency consideration; that is why collisions at $n = 8, 10$ is non-zero.

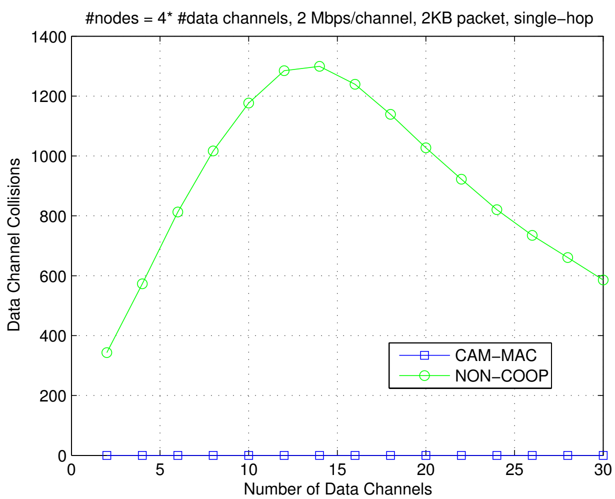


(a) Data channel collisions.

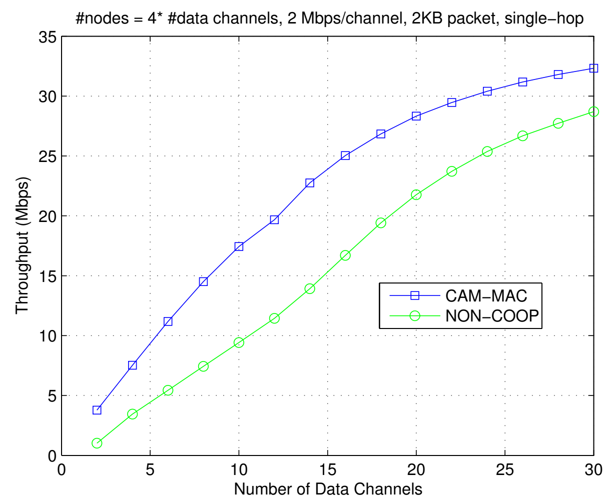


(b) Aggregate throughput.

Fig. 8. Simulation results: varying number of nodes, single-hop network.



(a) Data channel collisions.



(b) Aggregate throughput.

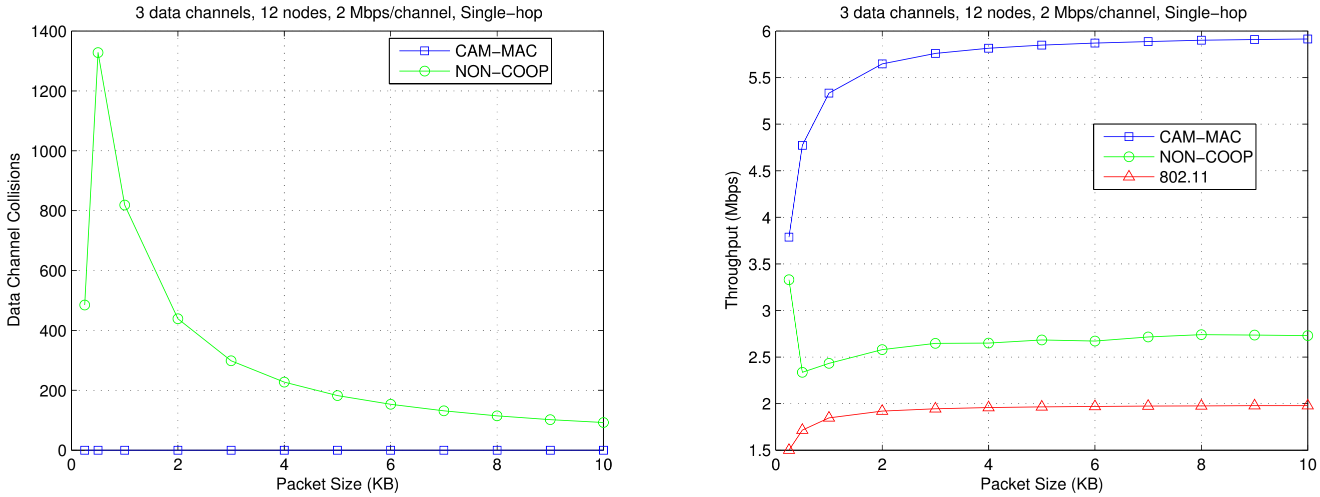
Fig. 9. Simulation results: varying number of channels, single-hop network.

The results of data channel collisions are consistent with the throughput results shown in Fig.8(b). When $n \leq 7$, both protocols have equal throughput. Significant difference appears when $n \geq 8$ — CAM-MAC maintains $Th_{cam} = 5.67$Mbps, a notable 120% improvement over NON-COOP, whose throughput stays at $Th_{noncoop} =$ 2.58Mbps.

- **Varying number of channels**: We scale the number of nodes as $n = 4m$, where m is the number of data channels, in order to simulate a fairly intense contention (each data channel has two pairs of competing nodes on average).

  CAM-MAC maintains zero data channel collision for all m, while NON-COOP reaches a maximum of 1300/sec. Its curve exhibits a bell shape (Fig.9(a)), which can be attributed to two counteractive factors: more channels accommodate more concurrent data sessions, thereby increasing the probability of being collided; on the other hand, more channels implies more nodes ($n = 4m$), and hence any node starting a control session will suppress more other nodes to initiate new sessions, thereby reducing the number of potential colliding sessions.

  As to throughput shown in Fig.9(b), both CAM-MAC and NON-COOP keep increasing and become only slightly slower as the number of channels becomes large. In particular, when $m = 10$, $Th_{cam} = 17.5$Mbps, which amounts to a channel utilization of $\frac{17.5}{2\times 10} = 87.5\%$. When $m = 20$, $Th_{cam} = 28.3$Mbps which amounts to a channel utilization of $\frac{28.3}{2\times 20} = 71\%$. This again shows that the so-called control channel bottleneck does not affect performance severely.
- **Varying packet size**: CAM-MAC still maintains zero



(a) Data channel collisions.

(b) Aggregate throughput.

Fig. 10. Simulation results: varying packet size, single-hop network.

data channel collisions for all packet sizes, while NON-COOP reaches a maximum of 1330/sec when the packet size is 500 bytes (Fig.10(a)). After that, it decreases as the packet size increases since fewer packets can be transmitted during a given period.

The throughput of both protocols (Fig.10(b)) keeps increasing (because a control session becomes shorter and shorter relative to a data session) until it saturates at $Th_{cam} = 5.91$Mbps (very close to the capacity, 6Mbps) and $Th_{noncoop} = 2.70$Mbps respectively. We see again a significant improvement of 119%.

2) Multi-hop Network: Recall that in a multi-hop network, nodes are not configured pair-wise and each node can be a source or destination on a packet-by-packet basis, therefore the head-of-line blocking problem will further reduce the throughput.

From Figs. 11, 12 and 13,[7] we see that CAM-MAC significantly outperforms NON-COOP in the multi-hop scenario. When varying the number of nodes, CAM-MAC has only 20% as many collisions as NON-COOP, and makes a 200% improvement of throughput over NON-COOP. When varying number of channels and varying packet size, results similar to corresponding single-hop cases have also been observed.

3) Comparison with MMAC and SSCH: We also compared CAM-MAC with MMAC [2] and SSCH [9], which are among the most representative and recent multi-channel MAC protocols using a single transceiver. All configurations and parameters are chosen the same as MMAC and SSCH in a single-hop network respectively, which are reproduced in Tab.II and Tab.IV for convenience. The disjoint flow setting used by SSCH is the same as the pair-wise configuration used by us, and the non-disjoint flow setting means sources and destinations are randomly chosen on a packet-by-packet basis, like what we did in multi-hop network scenarios.

[7]In all these figures, the number of nodes refers to the number of nodes in the innermost circle, i.e. N.

TABLE II
PARAMETERS FOR COMPARISON WITH MMAC

no. of channels	channel capacity	packet size
3	2Mbps	512 byte

TABLE III
COMPARISON WITH MMAC

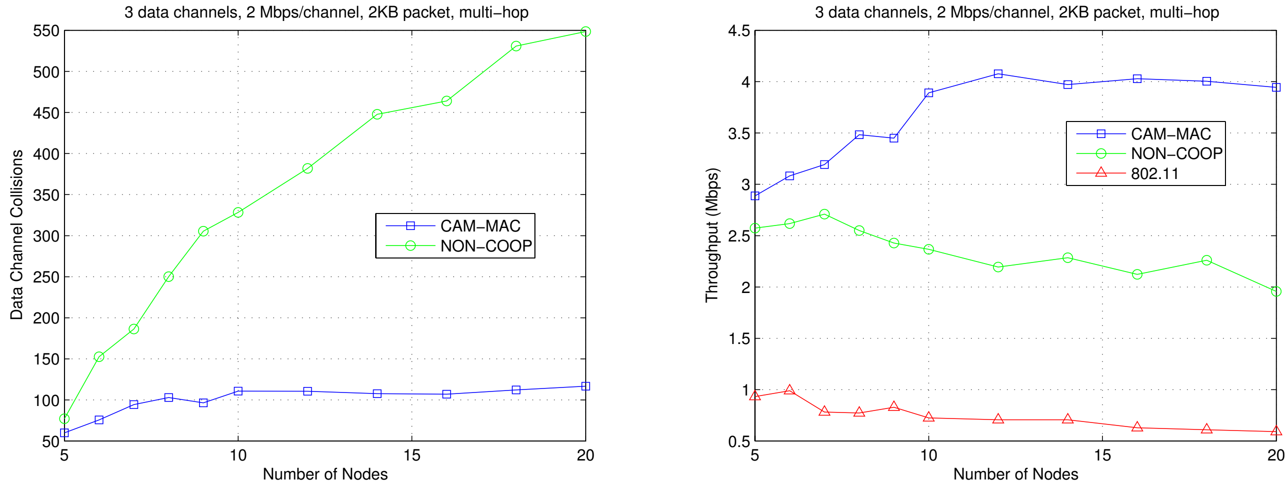
no. of nodes	throughput of MMAC	throughput of CAM-MAC	ratio of throughput
6	1.85 Mbps	3.20 Mbps	1.73
30	2.28 Mbps	3.30 Mbps	1.45
64	2.2 Mbps	3.31 Mbps	1.50

For comparison with MMAC, simulations over 6, 30 and 64 nodes are carried out respectively (note that in Tab.II CAM-MAC uses two data channels since the total number of channels is three) and the results are shown in Tab.III. As we can see, CAM-MAC achieves throughput improvement of 73%, 45% and 50% respectively.

To compare with SSCH[8] (CAM-MAC uses 12 data channels), when the number of nodes is 10, we get a throughput improvement of 119% in the disjoint flow case (Tab.V), and a throughput improvement of 254% in the non-disjoint flow case (Tab.VI).

The statistics of MMAC and SSCH are collected from their published data, one via NS-2 [22] and the other via Qualnet [23]. We admit different platforms may cause discrepancy, but the remarkable performance difference implies CAM-MAC would still outperform them on the same platform. Also, recall that those two protocols require clock synchronization.

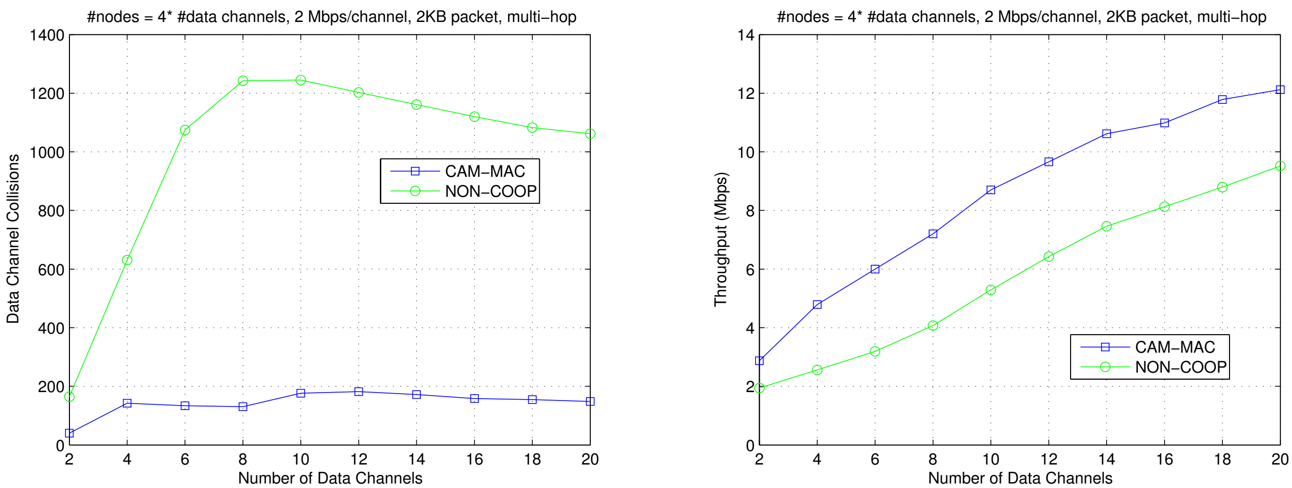
[8]13 channels are adopted by SSCH, while 802.11a specifies 12.



(a) Data channel collisions.

(b) Aggregate throughput.

Fig. 11. Simulation results: varying number of nodes, multi-hop network.



(a) Data channel collisions.

(b) Aggregate throughput.

Fig. 12. Simulation results: varying number of channels, multi-hop network.

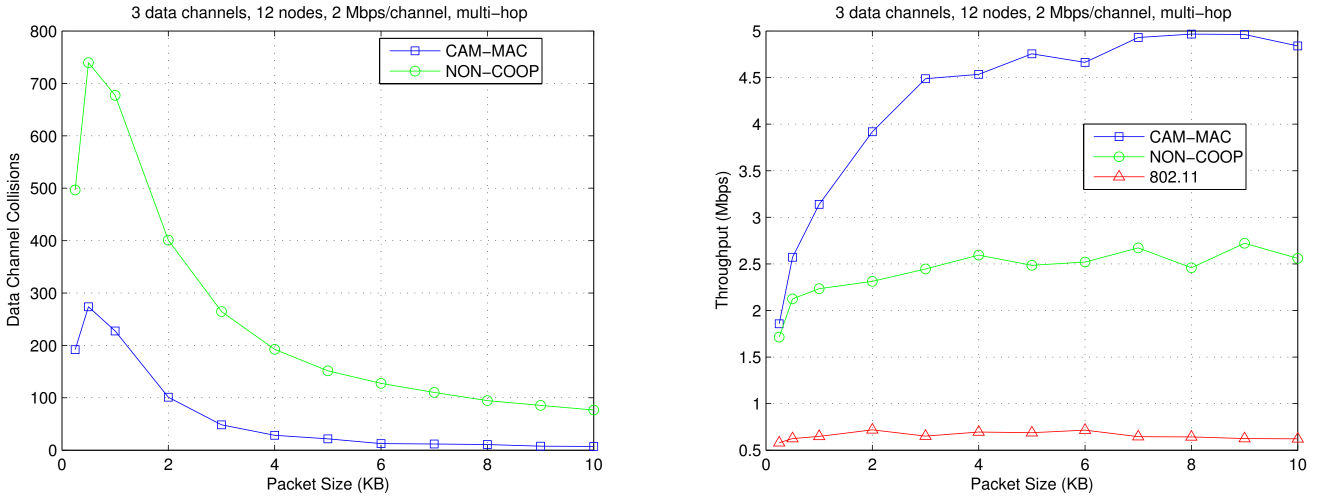
VI. Reflections

In this paper we introduce cooperation into multi-channel ad hoc networks, and by virtue of that, we are able to propose a multi-channel MAC protocol, called CAM-MAC, using a single transceiver and without clock synchronization. Comprehensive simulations are conducted to compare our protocol with a corresponding non-cooperative protocol and two existing representative multi-channel MAC protocols, MMAC [2] and SSCH [9]. Despite its simplicity, cheapness and low overhead, CAM-MAC achieves impressively higher performance. We reflect below on other aspects of the protocol we have proposed.

Two-hop Neighbor Discovery: For each node to provide more useful feedback information, it can be seen from Fig.3 that each node needs to know which nodes are around its direct neighbors. The acquisition of such two-hop neighbor information actually does not incur noticeably more traffic than the usual one-hop neighbor discovery, because it can be simply done by each node piggybacking its own *one-hop* neighbors in the usual *Hello* messages, thereby each node can figure out the two-hop neighbor information from the received Hello messages.

Node Mobility: Mobility will affect the accuracy of the (one- and two-hop) neighbor information, but CAM-MAC can still be applied provided that the dynamics of neighbor information due to mobility is not dramatic. Alternatively, the frequency of neighbor discovery message exchange can be increased accordingly.

Multi-Channel Power Control: Due to the channel assignment scheme of control channel and data channels, CAM-MAC can be easily integrated with power control techniques. This is because the power for control packets and data packets (and thus radio range) can be completely independent.



(a) Data channel collisions.

(b) Aggregate throughput.

Fig. 13. Simulation results: varying packet size, multi-hop network.

TABLE IV

PARAMETERS FOR COMPARISON WITH SSCH

no. of channels	channel capacity	packet size
13	54 Mbps	512 byte

TABLE V

COMPARISON WITH SSCH: DISJOINT FLOWS

no. of nodes	throughput of SSCH	throughput of CAM-MAC	ratio of throughput
6	33 Mbps	90 Mbps	2.73
10	48 Mbps	105 Mbps	2.19
14	62 Mbps	110 Mbps	1.69
20	81 Mbps	118 Mbps	1.46

TABLE VI

COMPARISON WITH SSCH: NON-DISJOINT FLOWS

no. of nodes	throughput of SSCH	throughput of CAM-MAC	ratio of throughput
6	20 Mbps	75 Mbps	3.75
10	26 Mbps	92 Mbps	3.54
14	37 Mbps	99 Mbps	2.67
20	38 Mbps	109 Mbps	2.89

Sensor Networks: Our single-transceiver solution without synchronization is especially suitable for sensor networks due to the reduced overhead, hardware cost and size. However the trade-off between throughput and energy consumption needs to be investigated.

REFERENCES

[1] A. Adya, P. Bahl, J. Padhye, and A. Wolman, "A multi-radio unification protocol for IEEE 802.11 wireless networks," in *IEEE Broadnets*, 2004.

[2] J. So and N. Vaidya, "Multi-channel mac for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," in *MobiHoc*, 2004.

[3] A. Nasipuri, J. Zhuang, and S. R. Das, "A multichannel CSMA MAC protocol for multihop wireless networks," in *WCNC*, 1999.

[4] M. A. Marsan and D. Roffinella, "Multichannel local area networks protocols," *IEEE Journal on Selected Area in Communications*, vol. 1, p. 885C897, October 1983.

[5] N. Jain and S. Das, "A multichannel CSMA MAC protocol with receiver-based channel selection for multihop wireless networks," in *IC3N*, 2001.

[6] A. Nasipuri and J. Mondhe, "Multichannel CSMA with signal power-based channel selection for multihop wireless networks," in *IEEE VTC*, 2000.

[7] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu, "A multi-channel MAC protocol with power control for multi-hop mobile ad hoc networks," *The Computer*, vol. 45, no. 1, pp. 101–110, 2002.

[8] ——, "A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks," in *I-SPAN*, 2000.

[9] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *Mobicom*, 2004.

[10] J. Chen, S. Sheu, and C. Yang, "A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs," in *PIMRC*, 2003.

[11] A. Tzamaloukas and J. Garcia-Luna-Aceves, "Channel-hopping multiple access," in *ICC*, 2000.

[12] ——, "Channel-hopping multiple access with packet trains for ad hoc networks," in *IEEE Device Multimedia Communications (MoMuC)*, 2000.

[13] L. Huang and T.-H. Lai., "On the scalability of IEEE 802.11 ad hoc networks," in *MobiHoc*, 2002.

[14] Maxim Integrated Products Inc., *MAX2820, MAX2820A,MAX2821, MAX2821A 2.4GHz 802.11b Zero-IF Transceivers Data Sheet rev. 04/2004*, Sunnyvale, California, USA, 2004.

[15] IEEE 802.11 Working Group, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, 1999.

[16] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," in *INFOCOM*, 2002.

[17] J. Mo, H. W. So, and J. Walrand, "Comparison of multichannel mac protocols," in *MSWiM*, 2005.

[18] Y. Wang and J. Garcia-Luna-Aceves, "Modeling of collision avoidance protocols in single-channel multihop wireless networks," *Wireless Network*, vol. 10, pp. 495–506, 2004.

[19] J. P. Monks, V. Bharghavan, and W. W. Hwu, "A power controlled multiple access protocol for wireless packet networks," in *INFOCOM*, 2001.

[20] A. Baiocchi, A. Todini, and A. Valletta, "Why a multichannel protocol can boost IEEE 802.11 performance," in *MSWiM*, 2004, pp. 143–148.

[21] A. Nasipuri and S. R. Das, "Multi-channel mac with dynamic channel selection for ad hoc networks," *Tech. Report*, 2004.

[22] VINT Group, "Ucb/lbnl/vint network simulator NS (version 2)."

[23] Qualnet, *http://www.qualnet.com/*.