Cooperative Asynchronous Multichannel MAC: Design, Analysis, and Implementation

Tie Luo, *Student Member*, *IEEE*, Mehul Motani, *Member*, *IEEE*, and Vikram Srinivasan, *Member*, *IEEE*

Abstract—Medium access control (MAC) protocols have been studied under different contexts for decades. In decentralized contexts, transmitter-receiver pairs make independent decisions, which are often suboptimal due to insufficient knowledge about the communication environment. In this paper, we introduce *distributed information sharing* (DISH), which is a distributed flavor of *control-plane cooperation*, as a new approach to wireless protocol design. The basic idea is to allow nodes to share control information with each other such that nodes can make more informed decisions in communication. This notion of control-plane cooperation augments the conventional understanding of cooperation, which sits at the *data plane* as a *data relaying* mechanism. In a multichannel network, DISH allows neighboring nodes to notify transmitter-receiver pairs of *channel conflicts* and *deaf terminals* to prevent collisions and retransmissions. Based on this, we design a single-radio cooperative asynchronous multichannel MAC protocol called CAM-MAC. For illustration and evaluation purposes, we choose a specific set of parameters for CAM-MAC. First, our analysis shows that its throughput upper bound is 91 percent of the system bandwidth and our simulations show that it actually achieves a throughput of 96 percent of the upper bound. Second, our analysis shows that CAM-MAC can saturate 15 channels at maximum and our simulations show that it saturates 14.2 channels on average, which indicates that, although CAM-MAC uses a control channel, it does not realistically suffer from control channel bottleneck. Third, we compare CAM-MAC with its noncooperative version called UNCOOP, and observe a throughput ratio of 2.81 and 1.70 in single-hop and multihop networks, respectively. This demonstrates the *value of cooperation*. Fourth, we compare CAM-MAC with three recent multichannel MAC protocols, MMAC, SSCH, and AMCP, and find that CAM-MAC significantly outperforms all of them. Finally, we implement CAM-MAC and UNCOOP on commercial off-the-shelf hardware and share lessons learned in the implementation. The experimental results confirm the viability of CAM-MAC and the idea of DISH.

Index Terms—Distributed information sharing (DISH), control-plane cooperation, CAM-MAC, multichannel coordination problem, MAC protocol, ad hoc networks.

1 Introduction

The vagaries of the wireless channel and its location-dependent nature often prevent wireless networking devices from acquiring sufficient knowledge about the communication environment. This often leads to nodes in a distributed environment making suboptimal decisions and thereby clamps down on system performance. In a multichannel network, this issue of knowledge insufficiency becomes aggravated due to a fundamental hardware limitation: commercial off-the-shelf (COTS) multichannel wireless cards do not support operating on multiple frequencies simultaneously, and thus, a node will miss information disseminated over the channels that it is not tuned to. We categorize this as a *multichannel coordination* (MCC) problem which has two variants. One is called a *channel conflict* problem, which is created by a node (unintentionally) selecting a busy channel for data transmission and would result in data collision. The other is called a *deaf terminal* problem, which is created by a node initiating communication with another node on a different channel and would result in unnecessarily retransmissions. Although a rich body of prior work exists, there still lacks in a cheap yet effective solution.

In prior work, one mainstream approach is using multiple radios[1] and dedicating one of them to monitoring channel usage when the other is engaged in communication [5], [6], [7], [8], [9], [10]. The other mainstream approach is regulating the irregular node behaviors using well-known time slots [2], [11], [12] or channel hopping sequences [3], [13], [14]. The first approach clearly increases hardware cost, size, and possibly energy consumption. The second approach requires time synchronization which is a difficult task involving considerable overhead and complexity [15], and also compromises network scalability [16].

In this paper, we introduce *distributed information sharing* (DISH), which is a distributed flavor of *control-plane cooperation*, as a new approach to wireless protocol design, and then apply it to multichannel medium access control (MAC) to solve the MCC problem. The basic idea is to allow nodes to share control information with each other such that nodes can make more informed decisions in communication. This notion of control-plane cooperation augments the conventional understanding of cooperation, which sits at the *data plane* as a mechanism for intermediate nodes to help *relay data* for source-destination pairs.

Applying DISH to multichannel ad hoc networks, we allow neighboring nodes who identify an MCC problem to

- *T. Luo and M. Motani are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576. E-mail: {tie, motani}@nus.edu.sg.*
- *V. Srinivasan is with Bell Labs Research, Salarpuria Ascent, 3rd floor, No. 77, Jyothi Nivas College Road, Koramangala Industrial Layout, Ward No. 68, Bangalore 560095, India. E-mail: vikramsr@alcatel-lucent.com.*

Manuscript received 16 July 2007; revised 26 Jan. 2008; accepted 7 July 2008; published online 17 July 2008.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2007-07-0209. Digital Object Identifier no. 10.1109/TMC.2008.109.

1. In this paper, we use "radio" and "transceiver" interchangeably.

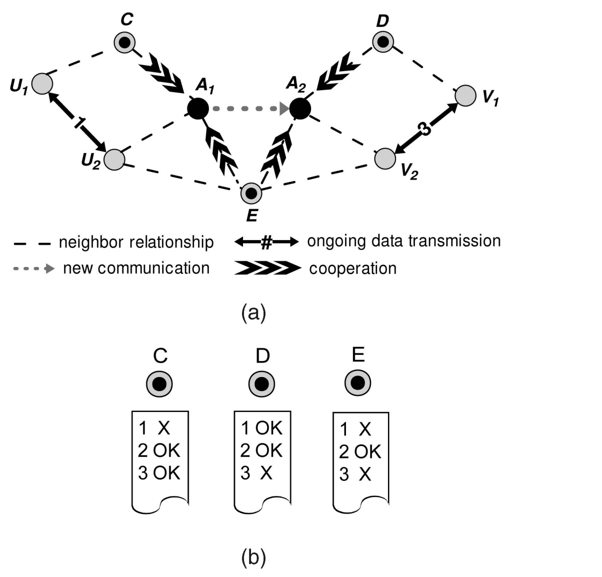


Fig. 1. An illustration of the DISH idea. (a) A multichannel scenario. (b) Knowledge at individual nodes. By consolidating the knowledge at nodes C and D, or acquiring knowledge from node E, it shows that the conflict-free channel is channel 2.

notify the transmitter-receiver pair of the problem to avoid collisions and retransmissions. Fig. 1a gives an illustration. Two node pairs, (U_1, U_2) and (V_1, V_2), are performing data exchanges on channels 1 and 3, respectively, and node A_1 is to initiate a communication with A_2 at this moment. If A_2 is on a channel different from A_1, a deaf terminal problem is created. If (A_1, A_2) selects channel 1 or 3 for data exchange, a channel conflict problem is created. In either case, the neighboring nodes C, D, or E may have relevant channel usage information (see Fig. 1b) and could share with (A_1, A_2) to solve the MCC problem.

Based on the idea of DISH, we design a single-radio cooperative asynchronous multichannel MAC protocol called CAM-MAC for ad hoc networks. We evaluate CAM-MAC from both theoretical and practical perspectives, where we choose a specific set of protocol parameters for illustration and evaluation purposes:

1. we show that its throughput upper bound is 91 percent of the system bandwidth and its average throughput approaches this upper bound with a mere gap of 4 percent,
2. we show that it can saturate 15 channels at maximum and 14.2 channels on average, which indicates that, although CAM-MAC uses a control channel, it does not realistically suffer from control channel bottleneck,
3. to investigate the *value of cooperation*, we compare CAM-MAC with its noncooperative version called UNCOOP, and observe a throughput ratio of 2.81 and 1.70 between them in single-hop and multihop networks, respectively, and
4. we compare CAM-MAC with three recent and representative multichannel MAC protocols, MMAC [2], SSCH [3], and AMCP [4], and the results show that CAM-MAC substantially outperforms all of them.

For a further and more realistic validation, we implemented CAM-MAC and UNCOOP on COTS hardware and conducted experiments. To the best of our knowledge, these prototypes are the first full implementation of single-radio asynchronous multichannel MAC protocols.

We review literature in Section 2, and identify new challenges to designing a cooperative protocol in Section 3. Then, we present the protocol details in Section 4 together with mathematical analysis. Following that, Section 5 provides simulation results in various scenarios, and Section 6 describes our hardware implementation and experiments. We discuss relevant issues in Section 7 and, finally, give concluding remarks in Section 8.

2 Related Work

Multichannel MAC protocols for ad hoc networks can be categorized into two general classes as below.

2.1 Single-Radio Solutions

2.1.1 Control-Data Window Schemes

MMAC [2] assumes the IEEE 802.11 power saving mode and divides time into beacon intervals. Each beacon interval is 100 ms and consists of a 20-ms ATIM window and an 80-ms data window. Nodes negotiate and reserve channels in the ATIM window on a common channel, and transmit data in the data window on multiple channels concurrently. The data window size is fixed, and hence, it has to be set according to the *maximum* data packet size, leading to inefficiency. By contrast, MAP [11] varies the data window size and avoids this problem. However, like MMAC, its reservation interval (i.e., control window) is still fixed, and thus, both protocols suffer from the inflexibility to different node densities: at low density, the control window has long idle time; at high density, the control window cannot accommodate all negotiations and some nodes have to wait for the next control window. Furthermore, MMAC and MAP requires time synchronization over the entire network, which is a notoriously hard task involving considerable overhead and complexity [15], and compromises scalability [16]. LCM MAC [17], on the other hand, allows each neighborhood to negotiate the boundaries of control-data windows independently, in order to avoid time synchronization. However, the negotiated window size can hardly fit for all nodes in the neighborhood, and this window negotiation, plus a fine-tuning mechanism, considerably complicates the protocol. Besides, it has a starvation problem lacking in an appropriate solution. Finally, all these control-data window schemes have a common problem: during each control window, all channels other than the common channel cannot be utilized, resulting in channel underutilization.

CAM-MAC is a simple protocol that does not need time synchronization at all, and there is no requirement for all nodes (MMAC and MAP) or a set of nodes (LCM MAC) to follow a control-data window which sacrifices efficiency.

2.1.2 Channel Hopping Schemes

In SSCH [3], each node hops among all channels according to a pseudorandom sequence such that neighboring nodes will have channels overlap in time periodically. Since a transmitter can only communicate to a receiver when they hop onto the same channel, large delay can be incurred; in the worse case, a transmitter has to wait for $m_0T_{sw} + (m_0 - 1)T_0$ before communicating to its intended receiver, where m_0 is the number of channels, T_{sw} is channel switching delay, and T_0 is a node's sojourn time on each channel (including possible data exchanges). In addition, frequent hopping makes the protocol performance very sensitive to channel switching delay which varies from tens to thousands of microseconds. In CHMA [13] and CHAT [14], the entire network adopts a *common* hopping sequence. This does not really solve the large delay problem because each node sojourns on each channel for different periods of time. Moreover, all the channel hopping schemes require clock synchronization.

In CAM-MAC, each node stays on a common channel and only switches channel when a data exchange is established successfully or finished. This avoids switching channel too often and, due to the common channel, does not incur large delay. Besides, again, no clock synchronization is required.

2.1.3 Routing and Channel Assignment Schemes

CBCA [18] combines channel assignment with routing. It proposes to assign each set of intersected flows, called a component, with a single channel, in order to avoid channel switching delay, node synchronization,[2] and scheduling overhead at flow-intersecting nodes. CAM-MAC uses a control channel, which automatically avoids the problem of node synchronization and scheduling overhead. Regarding channel switching delay, its effect on network performance is much less than MCC problems: the channel switching delay is 40-150 μs [19], but a channel conflict can collide *at least* one data packet whose delivery several and even tens of milliseconds.[3]

In fact, CBCA shifts complexity from the MAC layer to the routing layer. Also, compared to packet, link, and flow-based channel assignments, it has the *least* flexibility in exploiting multichannel diversity: Each component, which spans all intersecting flows, can only use one channel. As a consequence, any two nodes in a common component cannot transmit simultaneously unless they are at least three or four hops apart (depending on the interference range). In a single-hop network, since all flows are intersected, a multichannel network degrades to a single-channel network.

2.1.4 Other Schemes

AMCP [4] uses a single transceiver and is also asynchronous, like CAM-MAC. A node attempts to always use its previously used channel unless the channel is occupied by other nodes, in which case it waits until an *avail timer* expires and then randomly selects a free channel. To avoid collision, the avail timer is set as the duration of a complete handshake that assumes the maximum data packet size. This conservative approach results in channel underutilization. On the contrary, CAM-MAC takes an aggressive approach; a transmitter always attempts to initiate communication unless it is sure that all channels are not available or the receiver is busy. Cooperation would come into play if the attempt creates an MCC problem.

Overall for this section, not all schemes address deaf terminal problem, whereas CAM-MAC solves both deaf terminal and channel conflict problems.

2.2 Multiradio Solutions

Using multiple radios can easily solve MCC problems by dedicating one radio for monitoring channel usage information. DCA [5] uses two transceivers, one for exchanging control packets and the other for exchanging data packets. The control packets are used to allocate the channels on the data transceiver on demand. A multichannel CSMA protocol with soft channel reservation was proposed in [6]. It assumes the number of channels to be equal to the number of transceivers per node, so that all channels can be used simultaneously. This is very expensive. Jain et al. [8] is a protocol similar to DCA in that it also dedicates a transceiver for control purposes, but the difference is that channel selection is done at the receiver end based on signal-to-noise ratio. MUP [9] also uses two transceivers but it allows both transceivers to exchange control messages and data packets. xRDT [17] extends RDT [20], which uses a (possibly different) quiescent channel for each node to receive packets, by adding a busy-tone radio to each node in order to inform the neighborhood of ongoing data reception, in order to avoid collision and deafness. Kyasanur and Vaidya [21] proposed link-layer protocols for routing in multiradio multichannel ad hoc networks. Each node is assigned a fixed interface for receiving packets and multiple switchable interfaces for transmitting packets. This is similar to the idea of quiescent channel but uses more radios to simplify overcoming MCC problems.

Obviously, the key drawback of multiradio protocols is the increase of device size, cost, and potentially energy consumption.

Summary. The most salient feature that distinguishes CAM-MAC from prior work is that CAM-MAC introduces and explores control-plane cooperation as another degree of freedom in protocol design. We believe that this new notion of cooperation has great potential in many other networking scenarios as well.

3 Caveats to Cooperative Protocol Design

We identify three major issues in designing a cooperative MAC protocol, which will adversely affect protocol performance unless properly addressed.

3.1 Control Channel Bottleneck

Using a dedicated control channel can facilitate the design of a cooperative protocol, because a control channel provides a unique rendezvous for nodes to disseminate, gather, and share information. However, this design scheme may come with a drawback: When a large number of channels and nodes are present, the single control channel which is used to set up communications can be highly congested and become a performance bottleneck. In this section, we define a metric to analyze this bottleneck

2. The "synchronization" stated in [18] does not mean time synchronization but node synchronization.

3. For example, transmitting a 1.5-Kbyte packet over an 802.11b 2-Mbps channel takes 6-ms transmission time plus handshaking and backoff periods.

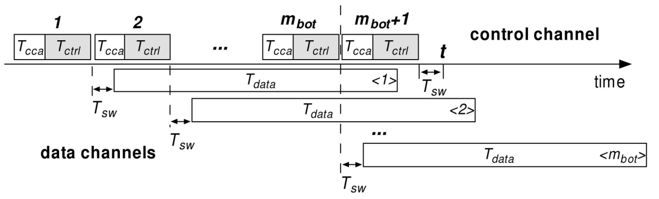


Fig. 2. Illustration of control channel bottleneck: No more than m_{bot} data channels can be simultaneously in use.

problem, and derive a condition by which this problem can be avoided.

Without loss of generality, suppose a complete communication process comprises a *control channel handshake* preceded by a random clear channel assessment (CCA) period, and immediately followed by a *data channel handshake*. We use the following notation:

- T_{ctrl}: duration of a successful control channel handshake.
- T_{data}: duration of a successful data channel handshake.
- T_{cca}: duration of a CCA period. Let $T_{cca}^{min} = \min(T_{cca})$.
- T_{sw}: channel switching delay.

Consider a network where all nodes are within communication range of each other. We define a metric, m_{bot}, to be the maximum number of data channels that can be simultaneously used for a given protocol (with the above parameters). When a bottleneck problem happens (Fig. 2), m_{bot} data channels are simultaneously in use, and there are still free data channels and backlogged nodes on the control channel. However, no more than m_{bot} data channels can be used, because at the time t when a subsequent ($(m_{bot}+1)$th) data channel usage happens, at least one data channel will become free (indicated by <1> in Fig. 2). Therefore, m_{bot} reflects the "capacity" for a multichannel system.

By noticing in Fig. 2 that T_{data} is bounded by the duration of m_{bot} successive control channel handshakes, each of which lasts a period of at least $T_{cca}^{min} + T_{ctrl}$, m_{bot} is given by

$$m_{bot} = \left\lceil \frac{T_{data}}{T_{cca}^{min} + T_{ctrl}} \right\rceil. \tag{1}$$

Note that T_{sw} does not affect m_{bot} (a data channel is actually free during T_{sw} and can be used by other nodes).

Suppose the objective is to design a protocol capable of saturating $m_{bot}^{\star}$ data channels, then $m_{bot} \geq m_{bot}^{\star}$ must be satisfied, which is equivalent to

$$\frac{T_{data}}{T_{cca}^{min} + T_{ctrl}} > m_{bot}^{\star} - 1.$$

Note that the equality sign can be removed because the r.h.s is an integer. The above resolves to

$$T_{cca}^{min} + T_{ctrl} < \frac{T_{data}}{m_{bot}^{\star} - 1}. \tag{2}$$

This is the condition for the design of a multichannel protocol to avoid control channel bottleneck. Note that T_{cca}^{min} and T_{ctrl} are variables subject to design, while T_{data} and $m_{bot}^{\star}$ are given parameters (although T_{data} involves variables such as the size of ACK, it is dominated by payload size which is typically a given parameter).

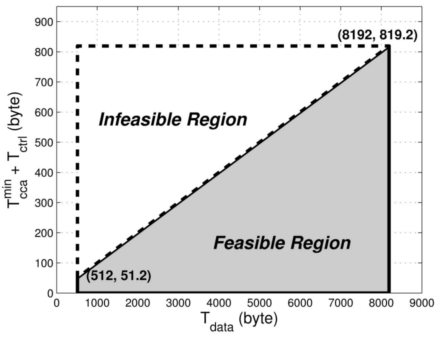


Fig. 3. The feasible region for choosing design variables for a multichannel MAC protocol based on IEEE 802.11a. We use byte as the unit of duration (a duration τ is converted into bytes via $\tau C/8$, where $C = 54$ Mbps is channel capacity), and suppose $T_{data} \in [512, 8, 192]$ bytes. The shaded area gives the feasible values of $T_{cca}^{min} + T_{ctrl}$ to saturate all the 12 channels.

As an example, suppose we are designing a multichannel protocol based on IEEE 802.11a, and we wish to saturate all the 12 nonoverlapping channels that 802.11a supports. This leads to $m_{bot}^{\star} = 11$ as there is a control channel. Therefore, we need to satisfy $T_{cca}^{min} + T_{ctrl} < T_{data}/10$ according to (2), which determines a feasible region for choosing protocol design variables, as plotted in Fig. 3.

The condition given by (2) is necessary and not sufficient, but a protocol satisfying this condition can practically avoid the bottleneck problem with high probability. This is based on our observations in simulations, whose details will be provided in Section 4.2 where we revisit this issue. On the other hand, we point out that the bottleneck problem is not necessarily catastrophic even if a protocol insignificantly violates the condition. This is because the control channel bottleneck problem requires at least $m_{bot} + 1$ transmit-receive pairs in a single-broadcast region *and* that each pair carries heavy traffic, which is not often the case.

3.2 Cooperation Coordination

An MCC problem can be identified by multiple neighboring nodes, and hence, their simultaneous response of sending cooperative messages will result in collision. This creates an issue of *cooperation coordination*. One solution is to make neighbors sequentially respond via a priority-based or slot-based mechanism, thereby ensuring all cooperative messages to be transmitted without collision. However, this is very inefficient because 1) there can be many wasted (idle) intervals because not all neighbors may identify the problem and 2) cooperative messages pertaining to the same MCC problem carry redundant information, and hence, receiving all of them is not necessary. Another solution is to let each neighbor send such messages probabilistically, in order to reduce the chance of collision. However, an optimal probability (optimal in the sense of minimizing the chance of collision) is hard to determine, and such a scheme can result in no response which essentially removes cooperation. Therefore, a simple yet effective coordination mechanism is needed.

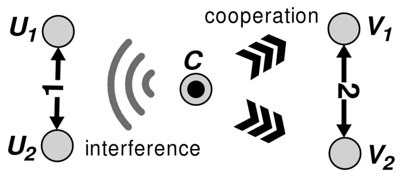


Fig. 4. Cooperation interference. (U_1, U_2) and (V_1, V_2) are not within interference range of each other, but if (V_1, V_2) creates an MCC problem and node C sends a cooperative message, it may interfere with (U_1, U_2) setting up communication.

3.3 Cooperation Interference

This issue means that the cooperative messages sent by neighbors for a transmit-receive pair can unconsciously cause interference to another (nearby) transmit-receive pair, as illustrated in Fig. 4, This is a new type of interference created by the introduction of cooperation, and our simulations found that it frequently happens and considerably intensifies channel contention. As such, a mechanism needs to be devised to address this deleterious side effect.

4 Protocol Design and Analysis

Our assumption is that each node is equipped with a single half-duplex transceiver that can dynamically switch between a set of orthogonal frequency channels but can only use one at a time.

We do not assume specific channel selection strategies; CAM-MAC runs on top of any such strategy. For quantitative performance evaluation, we will consider two strategies in simulations and experiments: 1) RAND selection, where a node randomly selects one from a list of channels that it deems free based on its knowledge, and 2) *most recently used* (MRU) selection, where a node always selects its MRU data channel unless it finds the channel to be occupied by other nodes, in which case RAND selection strategy is used.

We do not assume equal channel bandwidth or channel conditions such as noise levels; these can be taken into account by channel selection strategies (e.g., choose the channel with the highest SNR) which are not in our assumptions.

We also do not assume any (regular) radio propagation patterns, nor assume any relationship between communication ranges and interference ranges. Intuitively, none of the nodes is *responsible* for providing cooperation; a node cooperates if it *can* (it is idle and overhears a handshake that creates an MCC problem), and simply does not cooperate otherwise. Actually, there often exists at least one neighboring node that can cooperate, and even in the worse where no one can cooperate, the protocol still proceeds (as a traditional noncooperative protocol).

4.1 Protocol Design

One channel is designated as the control channel and the other channels are designated as data channels. A transmitter and a receiver perform a handshake on the control channel to set up communication and then switch to their chosen data channel to perform a DATA/ACK handshake, after which they switch back to the control channel. The control channel handshake is depicted in Fig. 5. A transmitter sends a PRA and its receiver responds with a PRB, like IEEE 802.11 RTS/CTS for channel reservation. Meanwhile, this PRA/PRB also probes the neighborhood inquiring whether an MCC problem is created (in the case of a deaf terminal problem, it is probed by PRA only). Upon the reception of the PRA or PRB, each neighbor performs a check and, if identifying an MCC problem, sends an INV message to invalidate the handshake (the receiver can also send INV after receiving PRA, since it is also one of the transmitter's neighbors). If no INV is sent and the transmitter correctly receives PRB, it sends a CFA to confirm the validity of PRA to all its neighbors (including the receiver), and the receiver will send a CFB to confirm the validity of the PRB if it correctly receives CFA. This marks the end of a control channel handshake. If any INV is sent, the handshake will not proceed and the transmitter will back off. The NCF is merely used by the transmitter to inform its neighbors that the PRA and CFA are invalid when it fails to receive CFB (the receiver gets INV after sending PRB).

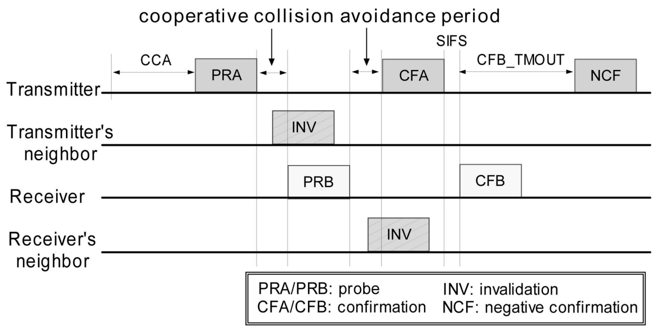


Fig. 5. The CAM-MAC control channel handshake.

The cooperative collision avoidance period is for mitigating INV collision caused by multiple neighbors sending INVs simultaneously. It is a simple CSMA-based mechanism where each neighbor schedules to send INV at a random point in this period and continues sensing the channel. Once the node that schedules at the earliest time starts to send, others in its vicinity cancel sending their INVs (a receiver can also cancel its PRB).

A possible set of frame formats is shown in Fig. 6. Both $\mathrm{PRA} + \mathrm{CFA}$ and $\mathrm{PRB} + \mathrm{CFB}$ carry the channel usage information of a communication *being established*, and an INV carries the channel usage information of an *established* communication that is to be collided (in the case of a channel conflict problem) or engages the receiver (in the case of a deaf terminal problem). A node may overhear this channel usage information and will cache it in the node's

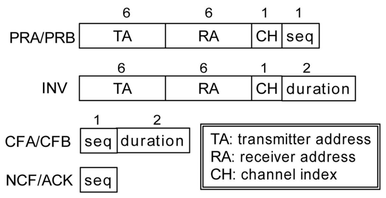


Fig. 6. A possible set of frame formats.

TA	RA	CH	until
A_1	A_2	1	11:30:52
B_1	B_2	3	11:30:56

Fig. 7. Channel usage table.

channel usage table, shown in Fig. 7. Note that the `until` column does not imply clock synchronization: It is calculated by adding the `duration` in a received CFA/CFB/INV message to the node's *own* clock. Similarly, when sending INV, a node does a reverse conversion from `until` to `duration` using a subtraction. Also note that this table is by caching overheard information while not by sensing data channels. This is because sensing data channels often obtains different channel status at the transmitter and the receiver, and resolving this discrepancy adds protocol complexity. In addition, this may lead to more channel switchings and radio mode (TX/RX/IDLE) changes and thus incurs longer delay.

4.2 Caveats Revisited

Now, we explain how we address the caveats stated in Section 3 in the design of CAM-MAC.

4.2.1 Control Channel Bottleneck

Recall the metric, m_{bot}, which is the maximum number of data channels that can be simultaneously used. Now we can calculate that $m_{bot} = 14$ ($\lceil 13.92 \rceil$) according to (1) based on the example parameters (Fig. 6) for CAM-MAC, where $T_{ctrl} = 113.75$ bytes, $T_{cca}^{min} = 37.25$ bytes, $T_{payload} = 2{,}048$ bytes, $T_{data} = 2{,}101.5$ bytes. This means that the protocol can theoretically saturate up to 15 channels (including the control channel), which sufficiently exceeds the number supported by current standards (3 and 12 channels by IEEE 802.11b/g and 802.11a, respectively).

Since the analysis only provides the maximum value, we also evaluate average performance via simulations. We configure 30 source-destination pairs where source nodes are always backlogged in order to simulate heavy traffic scenarios. We vary the number of data channels from 1 to 30. The results are summarized in Fig. 8, where CAMMAC-RAND and CAMMAC-MRU are CAM-MAC using RAND and MRU channel selection strategies, respectively. We see that 12.5 and 13.2 data channels (hence, 13.5 and 14.2 channels) can be saturated by these two versions of CAM-MAC, respectively. They are close to the theoretical maximum (15 channels) and also exceed what current standards support. Therefore, we can conclude that CAM-MAC does not realistically suffer from the control channel bottleneck.

4.2.2 Cooperation Coordination

Recall that this issue is to coordinate multiple neighbors to send cooperative messages as efficiently as possible. We address this using the cooperative collision avoidance period described in Section 4.1. It ensures that only one node will send out a cooperative message (INV) in each single-broadcast region, assuming that propagation delay is negligible. We found via simulations that this can reduce 70 percent-85 percent collisions between cooperative messages.

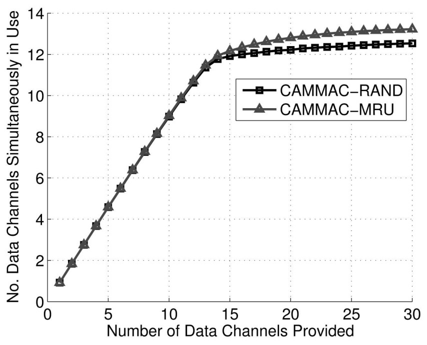


Fig. 8. The number of data channels that CAM-MAC can saturate.

In case that collisions still happen (due to propagation delay or because not all cooperative nodes can hear each other), it is not a serious problem because CAM-MAC makes such collisions *meaningful* by using *negative feedback* only. That is, since INV always means invalidation, a collision resulting from INV still conveys that the handshake should not proceed. Actually, using negative feedback is a logical design. First, a node expects a *binary feedback* since it selects one channel, instead of selecting a list of channels which needs multibit feedback indicating busy/free channels. Second, sending a positive feedback can be misleading because ensuring no MCC problem requires full information while a cooperative node cannot guarantee to have.

4.2.3 Cooperation Interference

Recall that this issue is that cooperative messages may cause interference to nearby transmitter-receiver pairs. We address this using *loyal periods*, which borrows the idea of IEEE 802.11 NAV. A node enters a loyal period when it hears a PRA (from a transmitter) or PRB (from a receiver) and does not identify an MCC problem with this handshake H_0. During this loyal period, the node *always keeps silent* (becomes "loyal" to H_0) even if it 1) identifies an MCC problem with *another* control channel handshake H_1 or 2) receives another PRA addressed to it (itself being an intended receiver). It exits this loyal period after H_0 ends successfully or is invalidated by cooperation. Note that rules 1 and 2 are reasonable because, although rule 1 disallows the "loyal" node to cooperate, there most probably exist other nodes that can cooperate (with H_1), and for rule 2, the loyal node should be able to respond to a *subsequent* PRA (retry) from the same transmitter since the loyal period will expire shortly. This mechanism of loyal period effectively mitigates cooperation interference. We observed via simulations a throughput improvement of 5 percent-30 percent in various scenarios. The details are omitted due to space constraints.

4.3 Protocol Analysis

We analyze the throughput upper bound for CAM-MAC in single-hop networks. This serves two purposes: 1) it tells whether the upper bound can approach total channel capacity and 2) the upper bound can be used to compare against the actual throughput obtained via simulations to see how close this upper bound can be approached.

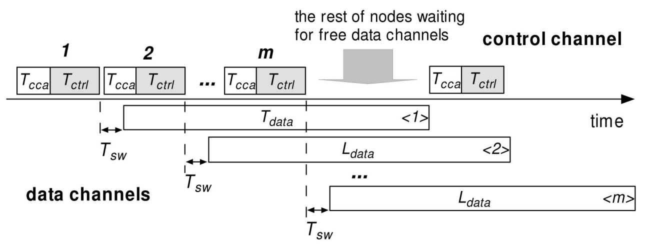


Fig. 9. Case 1. $m \le m_{bot}$. The bottleneck is at data channels, and thus, some nodes have to wait for free data channels. A node starts a control channel handshake only if there is at least one free data channel.

In this analysis, for achieving the maximum throughput, MCC problems are eliminated (nodes can always choose conflict-free channels and receivers are always available) and protocol overhead is kept at the minimum level. Part of notation is from Section 3.1.

Unsaturated network. A network is unsaturated (stable) if all input traffic gets through within finite delay. The throughput upper bound is simply given by

$$S_{max} = \sum_i \lambda_i,$$

where λ_i is the offered load of node i. Then, by assuming a homogeneous traffic pattern in which $\lambda_i = \lambda, \forall i$, the above reduces to

$$S_{max} = n_f \lambda, \tag{3}$$

where n_f is the number of source-destination pairs (i.e., flows).

Saturated network. The arrival traffic exceeds the network capacity and the queues of nodes build up infinitely. Denote the number of data channels by m:

1. $m \le m_{bot}$ *(bottleneck at data channels)*. If $n_f > m$, all m data channels can be simultaneously in use by m node pairs, and the rest of nodes have to wait on the control channel until some data channel becomes free. Fig. 9 depicts this scenario, where we can see a period of $T_{cca} + T_{ctrl} + T_{sw} + T_{data}$ will appear periodically. Hence, the *maximum utilization* of a data channel is

$$\eta_{max} = \frac{T_{payload}}{T_{cca}^{min} + T_{ctrl} + T_{sw} + T_{data}}, \tag{4}$$

where $T_{payload}$ is the transmission time of the payload in a data packet. So,

$$S_{max} = \eta_{max} m C, \tag{5}$$

where C is the capacity of a data channel.

If $n_f \le m$, S_{max} is achieved by assigning each source-destination pair with a dedicated data channel. In this case, η_{max} remains the same as (4), and

$$S_{max} = \eta_{max} n_f C. \tag{6}$$

2. $m > m_{bot}$ *(bottleneck at the control channel)*. If $n_f > m_{bot}$, the control channel becomes bottleneck (the reader can refer back to Fig. 2). Since data channels are more than what can be saturated $(m > m_{bot})$, the best case is that each control channel handshake leads to a successful transmission of $T_{payload}$, which translates to a *maximum system gain* of

$$G_{max} = \frac{T_{payload}}{T_{cca}^{min} + T_{ctrl}} \tag{7}$$

and

$$S_{max} = G_{max} C. \tag{8}$$

If $n_f \le m_{bot}$, then S_{max} is achieved again by the dedicated channel assignment as in (6), i.e.,

$$S_{max} = \eta_{max} n_f C.$$

Finally, the necessary and sufficient conditions for a network to be unsaturated, are derived from the above:

$$\begin{array}{ll} \lambda < \eta_{max} C, & \text{if } n_f \le \min(m, m_{bot}), \\ \lambda < \eta_{max} m C / n_f, & \text{if } n_f > m, \text{ and } m \le m_{bot}, \\ \lambda < G C / n_f, & \text{if } n_f > m_{bot}, \text{ and } m > m_{bot}. \end{array}$$

Remark. First, we compute two key quantities, the maximum channel utilization η_{max} and the maximum system gain G_{max}, by substituting the example protocol parameters (in Section 4.2, and $T_{sw} = 0$ to compute the maximum) into (4) and (7). We get

$$\eta_{max} = 91 \text{ percent} \quad \text{and} \quad G_{max} = 13.56.$$

Then, we revisit the two purposes mentioned in the beginning of this section. 1) Compared against total channel capacity, CAM-MAC can theoretically achieve a utilization of 91 percent. 2) In the comparison between the upper bound and simulation results, there are two cases: in the case of control channel bottleneck, the upper bound is $13.56C$ and the throughput of CAM-MAC is $13.2C$ (Fig. 8), indicating a ratio of 97 percent; in the case of no control channel bottleneck, the upper bound is $0.91mC$ or $0.91n_fC$ ((5) and (6)), and the throughput of CAM-MAC achieves 96 percent of the upper bound, as will be shown in Section 5.1.

5 Performance Evaluation

We evaluate and compare five protocols, namely IEEE 802.11, CAMMAC-RAND, CAMMAC-MRU, UNCOOP-RAND, and UNCOOP-MRU, using a discrete-event simulator which we developed on Fedora Core 5 with a Linux kernel of version 2.6.9. In these five protocols, IEEE 802.11 is used as a baseline in comparison, X-RAND and X-MRU are two versions of protocol X using RAND and MRU channel selection strategies, respectively. The protocol UNCOOP is identical to CAM-MAC except that the cooperation element is removed, i.e., neighboring nodes do not participate in communication by sending INV messages. This comparison will enable us to investigate the value of cooperation.

We use three performance metrics: 1) aggregate (end-to-end) throughput, 2) data channel conflict rate, defined as the packet collisions on data channels per second over all nodes, and 3) packet delivery ratio, defined as the number of data packets successfully received by destinations

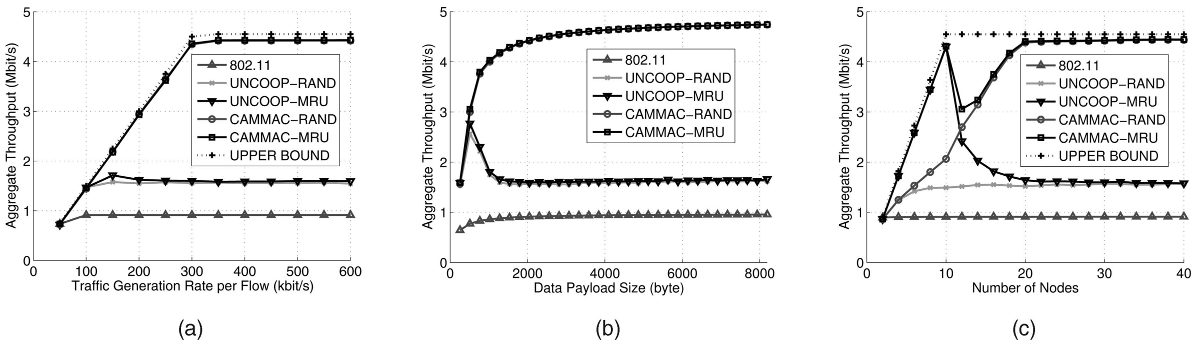


Fig. 10. Single-hop simulation results. (a) Impact of traffic load. (b) Impact of data payload size. (c) Impact of the number of nodes.

normalized by the number of data packets sent by sources. The packet delivery ratio takes into account deaf terminal problems which can lead to packet drops.

Nodes are uniformly distributed in a plane area. The transmission range is 250 m and the interference range is 500 m. Capture threshold is 6 dB. Each node has a single data packet queue (instead of *per-neighbor queues*, such as used by [3] and [22], which bypass head-of-line (HOL) blocking and will yield higher throughput and lower delay). In single-hop scenarios, the terrain is 100×100 m and nodes form disjoint source-destination pairs (i.e., flows). In multi-hop scenarios, the terrain is $1{,}500 \times 1{,}500$ m and n nodes form n nondisjoint flows randomly (each node is the source of one flow and the destination of another flow). Shortest path routing is used.

There is one control channel and five data channels with bandwidth 1 Mbps each. PHY and other MAC layer parameters, i.e., PLCP, SIFS, and retry limit, are the same as in IEEE 802.11 [23]. Each source generates data packets with 2-Kbyte payload according to a Poisson point process. The cooperative collision avoidance period is 35 μs. In the comparison of CAM-MAC and UNCOOP, we ignore channel switching delay as both protocols use the same handshake. However, in comparison to the other protocols, namely MMAC, SSCH, and AMCP, we use the parameters that they, respectively, use, including channel switching delay.

We terminate each simulation when a total of 100,000 data packets are sent over the network, and all results are averaged over 15 randomly generated networks.

5.1 Single-Hop Networks

5.1.1 Impact of Traffic Load (Fig. 10a)

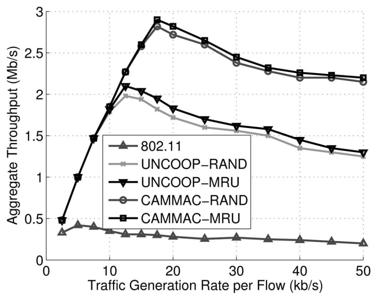
There are 30 nodes (i.e., 15 flows), and each source node generates traffic from 50 to 600 Kbps. We see that the throughput of UNCOOP quickly saturates at 1.6 Mbps while that of CAM-MAC keeps increasing until saturation at 4.5 Mbps. This indicates a remarkable ratio of 2.81. CAM-MAC also approaches the throughput upper bound with a gap of merely 4 percent. Another important observation is that there is almost no difference between the MRU and the RAND version of either CAM-MAC or UNCOOP. We explain this in discussing the impact of the number of nodes (Fig. 10c).

5.1.2 Impact of Data Payload Size (Fig. 10b)

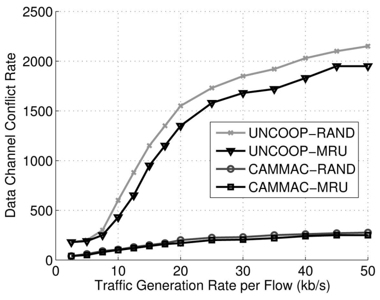
There are 30 nodes while source nodes are always backlogged, and data payload size is varied from 256 to 8,192 bytes. Interestingly, the throughput of UNCOOP is not monotonic; it first ascends, then descends, and finally levels off. This results from three counteracting factors: 1) a larger payload size offsets protocol overhead more effectively, and thus, lead toward higher throughput, 2) a longer data packet is more susceptible to channel conflicts, i.e., it is more likely to be collided, and 3) longer data packets keep nodes on data channels longer, and hence, fewer nodes will be possible to initiate new communication on the control channel, which reduces the possibility of channel conflicts. On the other hand, in CAM-MAC, cooperation resolves many channel conflicts and, hence, weakens factors 1 and 3. Therefore, factor 1 stands out and the throughput of CAM-MAC continuously increases.

5.1.3 Impact of the Number of Nodes (Fig. 10c)

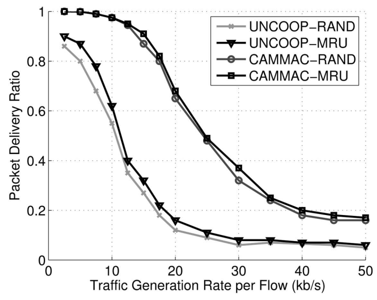
Unlike the previous two sets of simulations (varying traffic and payload size), this set of simulations shows an evident difference between the RAND and the MRU version: when the number of nodes is not more than 10 (i.e., ≤ 5 flows), the throughput of X-MRU is much higher than that of X-RAND and approaches the upper bound. This is because MRU strategy in effect assigns each flow with a *dedicated* data channel (recall that there are five data channels). When there are 12 nodes, MRU channels are frequently occupied by nonowner nodes since the number of transmitter-receiver pairs is one more than the number of data channels. This degrades MRU strategy close to RAND strategy. After that, as the number of nodes increases, MRU channels are more frequently occupied but additional nodes also boost the availability of cooperation. This explains why the throughput of UNCOOP-MRU continues to decrease while that of CAMMAC-MRU gradually recovers. Finally, at saturation, MRU and RAND versions converge, like in the previous two sets of simulations. This is because, at a large number of nodes, MRU channels are deprived very frequently, and thus, MRU strategy degrades to RAND strategy in effect. On the other hand, we see that only cooperation makes big difference: there is a large gap between CAM-MAC and UNCOOP, and CAM-MAC again achieves a throughput of 2.81 times that of UNCOOP, meanwhile approaching the upper bound with a factor of 96 percent.



(a)

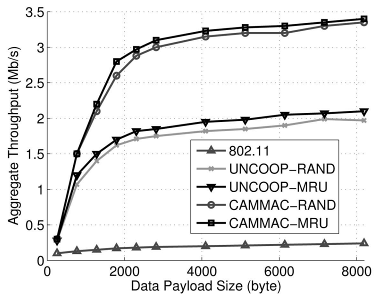


(b)

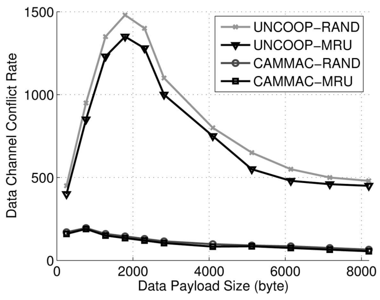


(c)

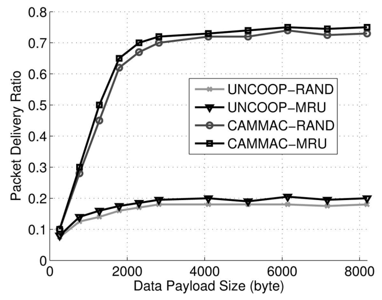
Fig. 11. Impact of traffic load in multihop networks. Node density is $10/r^2$. (a) Aggregate throughput. (b) Data channel conflicts. (c) Packet delivery ratio.



(a)



(b)



(c)

Fig. 12. Impact of data payload size in multihop networks. Node density is $10/r^2$ and traffic generation rate is 20 Kbps. (a) Aggregate throughput. (b) Data channel conflicts. (c) Packet delivery ratio.

5.2 Multihop Networks

5.2.1 Impact of Traffic Load (Fig. 11)

We randomly place 360 nodes in the network, which translates to a node density of $10/r^2$, where r is the transmission range. Traffic generation rate varies from 2.5 to 50 Kbps per flow. The throughput results are shown in Fig. 11a, where we see that the four multichannel MAC protocols achieve much higher throughput than 802.11 due to the higher spatial utilization. The other large difference is between CAM-MAC and UNCOOP; for example, at the traffic generation rate of 50 Kbps, the throughput ratio between CAM-MAC and UNCOOP is 1.70. The results of channel conflict rate are summarized in Fig. 11b, where we see remarkable gap between CAM-MAC and UNCOOP. This similarly happens to the results of packet delivery ratio shown in Fig. 11c.

A noticeable phenomenon is that the difference between CAM-MAC and UNCOOP in channel conflict rates is often much larger than that in throughput. This is because throughput does not relate to channel conflict rates *linearly*: A cooperative protocol has less data channel usages than an uncooperative protocol, because many conflicting channel usages are prevented by cooperation.

5.2.2 Impact of Data Payload Size (Fig. 12)

There are 360 nodes and the traffic load is 20 Kbps. Data payload size varies from 256 to 8,192 bytes. Interestingly, for all the protocols, although throughput (Fig. 12a) and packet delivery ratio (Fig. 12c) monotonically increase, the channel conflict rate (Fig. 12b) exhibits a bell shape. Actually, this is accounted for by the two contradicting factors 1 and 3 described in the discussion of Fig. 10b (Section 5.1).

5.2.3 Impact of Node Density (Fig. 13)

We vary node density from 2 to $20/r^2$ and fix traffic load at 20 Kbps. The curves are similar to Fig. 11 (impact of traffic load). This is simply because increasing node density and increasing traffic load have the same consequence of (linearly) increasing the *aggregate* traffic load for the network.

Summary. Our single-hop and multihop simulations manifestly demonstrate the value of cooperation: Cooperation effectively mitigates MCC problems and substantially enhances system performance.

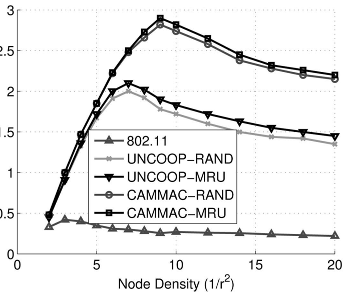
5.3 Comparison with MMAC, SSCH, and AMCP

We compare CAM-MAC with three recent multichannel MAC protocols, MMAC [2], SSCH [3], and AMCP [4].[4] They all use a single transceiver, but MMAC and SSCH require clock synchronization while AMCP does not. In the comparison, CAM-MAC adopts MRU strategy, and uses the same setup as MMAC, SSCH, and AMCP use, respectively, for the purpose of comparing with their reported results under the same settings.

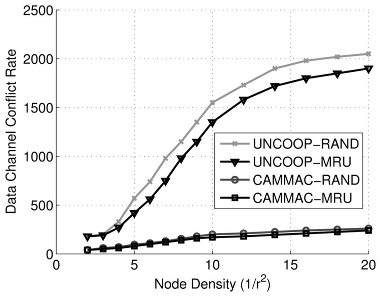
5.3.1 Comparison with MMAC

The parameters are shown in Table 1 (CAM-MAC uses only two and three data channels in single- and multihop networks, respectively). The first set of simulations are conducted in a wireless LAN, where nodes are configured as disjoint and fully loaded flows, the same as in MMAC.

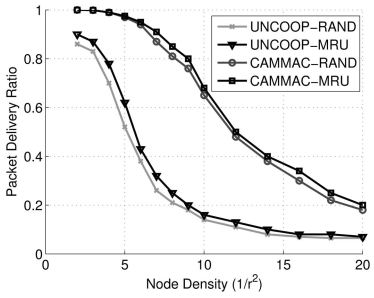
4. DCA [5] is an early representative protocol and uses multiple transceivers. It has been shown in [2] to be outperformed by MMAC.



(a)



(b)



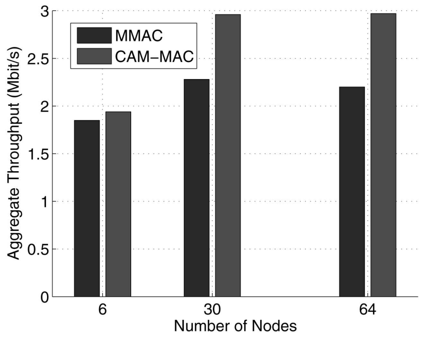
(c)

Fig. 13. Impact of node density in multihop networks. Traffic generation rate is 20 Kbps. (a) Aggregate throughput. (b) Data channel conflicts. (c) Packet delivery ratio.

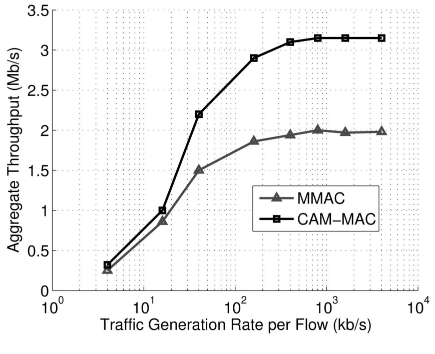
The results are presented in Fig. 14a, where we see that CAM-MAC achieves a throughput of 1.05, 1.30, and 1.35 times that of MMAC at 6, 30, and 64 nodes, respectively. The second set of simulations is conducted in a multihop network. Also the same as in MMAC, 100 nodes are randomly placed in a 500×500 m area, and 40 sources and 40 destinations are randomly chosen. The results are shown in Fig. 14b. We see that, at saturation, CAM-MAC achieves 1.57 times the throughput of MMAC.

TABLE 1
Parameters for Comparison with MMAC

channel	WLAN		multi-hop	
capacity	no. channels	packet size	no. channels	packet size
2Mb/s	3	512byte	4	1024byte



(a)



(b)

Fig. 14. Comparison with MMAC. (a) Wireless LAN. (b) Multihop networks.

5.3.2 Comparison with SSCH

The parameters are shown in Table 2 (CAM-MAC uses 12 data channels). As in SSCH, a disjoint-flow configuration and a nondisjoint-flow configuration are used, where the latter configuration means randomly selecting source-destination pairs (flows) on a packet-by-packet basis. In both configurations, all flows are always backlogged.

Note that the simulation parameters (Table 2) are favorable to SSCH but unfavorable to CAM-MAC. In SSCH, since nodes hop among channels following their *respective* sequences, a transmitter often has to wait for its intended receiver to hop onto the same channel before starting communication. Therefore, SSCH prefers short data packets and channel switching delay to reduce this waiting time (the reader may refer back to Section 2). On the other hand, CAM-MAC favors long data packets to offset control packet overhead, and its performance does not depend on channel switching delay as significantly as channel hopping schemes such as SSCH. Finally, SSCH uses *per-neighbor queues* which bypass the HOL blocking while CAM-MAC uses only a single queue.

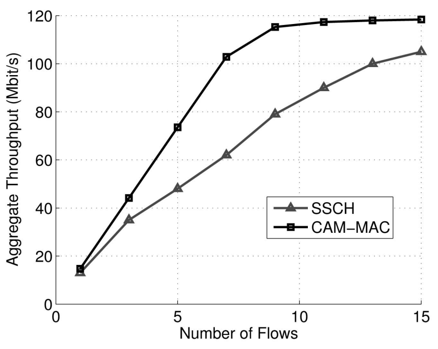
Nevertheless, from the results shown in Fig. 15, we see that CAM-MAC still outperforms SSCH with a factor of up to 1.50, in both disjoint and nondisjoint flow cases.

5.3.3 Comparison with AMCP

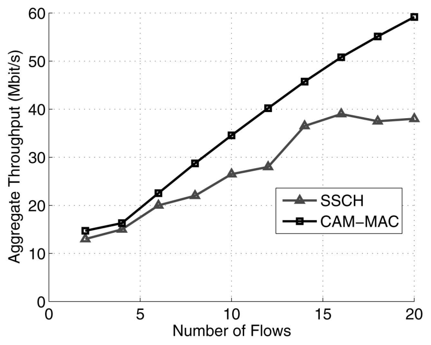
The parameters are shown in Table 3. There are 30 nodes forming 15 disjoint flows in a single-hop network. In the first set of simulations, the flows are always backlogged and the number of channels varies from 2 to 12. The results are shown in Fig. 16a. We see that CAM-MAC achieves a throughput of 11.86 Mbps while AMCP achieves 8.5 Mbps when there are 12 channels, which indicates a ratio of 1.40. Furthermore, AMCP saturates at 10 channels whereas CAM-MAC still exhibits a growing trend beyond 12 channels. Note that this is consistent with our analysis in Section 4.3. To see how, substitute the parameters (in Table 3 and Section 4.2) into (1) and obtain $m_{bot} = 7\ (\lceil 6.98 \rceil)$. This means $m > m_{bot}$ and $n_f > m_{bot}\ (n_f = 15)$, which directs us to use (7) and (8)

TABLE 2
Parameters for Comparison with SSCH

no. channels	channel capacity	packet size	channel switching delay
13	54 Mb/s	512byte	$80\mu s$

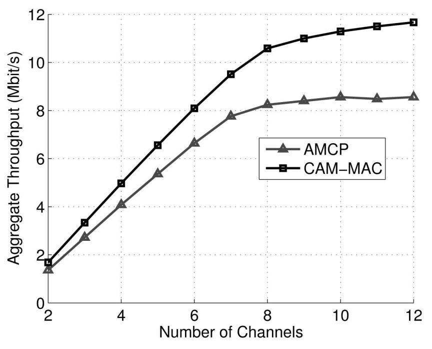


(a)

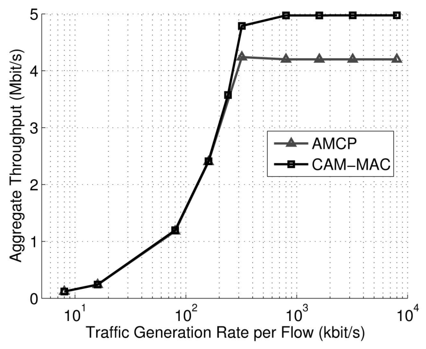


(b)

Fig. 15. Comparison with SSCH. (a) Disjoint flows. (b) Nondisjoint flows.



(a)



(b)

Fig. 16. Comparison with AMCP. (a) Throughput versus number of channels. (b) Throughput versus traffic load. Four channels.

and accordingly obtain $S_{max} = 13.24$ Mbps ($G_{max} = 6.62$). Comparing this upper bound S_{max} with the throughput that CAM-MAC achieves at 12 channels (11.86 Mbps) shows that CAM-MAC still has space for throughput growth (recall that, in our simulation results in Section 5.1, CAM-MAC approaches the upper bound very closely).

In the second set of simulations, there are four channels and the traffic generation rate varies from 8 Kbps to 8 Mbps. The results are shown in Fig. 16b. Both CAM-MAC and AMCP have equal throughput at light traffic load, but apparent difference appears at medium to high load, and finally, CAM-MAC saturates at 5 Mbps while AMCP saturates at 4.2 Mbps.

Furthermore, recalling the channel underutilization of AMCP as mentioned in Section 2, we can expect larger difference if *variable* data packet sizes are used.

Summary. Our extensive comparison with representative multichannel MAC protocols demonstrates the high productivity of CAM-MAC.

6 Testbed Demonstration

We implemented CAMMAC-RAND, CAMMAC-MRU, UNCOOP-RAND, and UNCOOP-MRU on COTS hardware and conducted testbed experiments. To the best of our knowledge, these are the first full implementation of single-radio asynchronous multichannel MAC protocols for ad hoc networks. Recently, So et al. [15] implemented a multichannel time synchronization protocol. It periodically exchanges beacon packets but data handshaking was not implemented. Vedantham et al. [18] provide a testbed for routing and channel assignment via statical manual configuration instead of hardware implementation. Most recently, Wu et al. [24] and Le et al. [25] implement a multichannel MAC protocol, which is designed for sensor network data collection applications assuming the many-to-one traffic pattern.

TABLE 3
Parameters for Comparison with AMCP

channel capacity	packet size	channel switching delay
2 Mb/s	1000byte	$224 \mu s$

6.1 Hardware Implementation

6.1.1 Platform Selection

We chose a microcontroller (MCU)-based platform with an ASIC radio, instead of 1) an FPGA-based platform, which is more expensive and requires hardware description language (HDL) in programming, or 2) a software radio, whose MAC source node is not fully available. Among the ASIC radios, we chose 802.15.4 radios instead of 802.11 radios because 802.11-radio-based devices (such as laptops and PDAs) have higher cost and bigger size than 802.15.4 devices, and 802.11-based development software (such as HostAP [26] and MadWifi [27]) has more limited MAC layer control than the software available to 802.15.4, such as TinyOS [28]. Eventually, we chose TelosB Mote [29], which is an MCU platform with an ASIC radio (CC2420 [30]) as our hardware platform and TinyOS 2.0 as our software platform. TinyOS has almost full control over the MAC

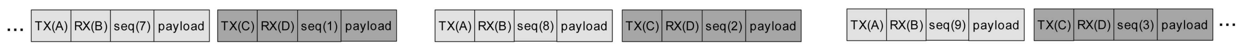

...
TX(A) RX(B) seq(7) payload
TX(C) RX(D) seq(1) payload
TX(A) RX(B) seq(8) payload
TX(C) RX(D) seq(2) payload
TX(A) RX(B) seq(9) payload
TX(C) RX(D) seq(3) payload
...

Fig. 17. Packet collision detection via an interleaved fragment sequence, where TX/RX IDs are *alternate* and seq's are *inconsecutive*.

layer, and its component-based architecture and C-like programming enables rapid development.

This platform choice suffices for our *comparative* study but, as a caveat, is not suitable for establishing *benchmarks* for multichannel WiFi cards.

6.1.2 Overcoming Limitations

There are two limitations of the hardware. First, the maximum packet size that CC2420 supports is only 127 bytes. To overcome this, we transmit a *sequence of fragments* as the substitution of a long data packet. The intervals τ between the fragments are counted as actual payload via τC, where $C = 250$ Kbps is the channel bandwidth, and the intermediate fragments are counted as pure payload without frame headers and footers. The second limitation is that the accuracy of timing on TelosB motes is not reliable at the microsecond level while reliable at the millisecond level. We circumvent this by *proportionally* prolonging all intervals, such as SIFS, CCA, and fragment intervals, up to milliseconds. Consequently, to transmit a 2-Kbyte data packet, a node transmits a sequence of 20 fragments with the length of 30 bytes each (including preamble) and the 19 intervals of 8 ms each. This results in a total of $\sim$175 ms to transmit a data packet (each fragment needs 100-200 μs to be sent in the air after assembled in memory). Under the same setting, a control channel handshake lasts $\sim$9 ms. The ratio between these two durations is close to that in our simulations.

6.1.3 Collision Detection

Interestingly, the methods that we used to overcome the limitations enabled us to devise a very simple yet accurate technique to detect packet collision, which can be generally used in other scenarios. Collision detection is useful in that it benefits collision avoidance, flooding, channel selection, and data aggregation algorithms [31] in differentiating between the two causes of *packet corruption*: packet collision and channel imperfection (such as noise and multipath effects). A typical prior technique is CRC checking, which unfortunately does not solve the detection problem because it only indicates packet *corruption*. Other solutions such as using link quality indicator (LQI) and/or received signal strength indicator (RSSI) are empirical in essence and lack in accuracy. In fact, according to [32], [33], and [34], it is still controversial whether RSSI or LQI is a better indicator for link quality.

Our technique is called *interleaved fragment sequence* detection. The key idea is based on 1) the fragmented data transmission and 2) the large difference between the fragment interval (8 ms) and the fragment transmission time ($<$ 1 ms), as described in Section 6.1.2. As such, if a node receives a sequence of fragments from more than one transmitter, as illustrated by Fig. 17, it indicates a data packet collision (since intervals are actual payload). Therefore, packet collision can be easily detected by simply checking fragment headers.

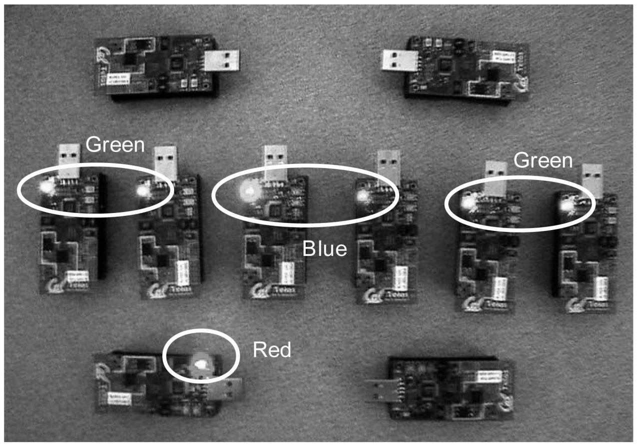


Fig. 18. A snapshot in an experiment on CAM-MAC with 10 nodes. The four "green nodes" are two transmitter-receiver pairs communicating on two different data channels. The two "blue nodes" are performing a control channel handshake (specifically, a PRA was just sent from one to the other). This creates a channel conflict problem since there are only two data channels which are already being in use. At this moment, a neighboring node, indicated by the red LED, identifies this (via the PRA) and sends a cooperative message (INV). Then, the two blue nodes will back off to discontinue the control channel handshake, and thus, data collision is prevented.

6.2 Experimental Results

For visualization purposes, we use the three LEDs on each TelosB mote to indicate specific events of interest (a maximum of $2^3 = 8$ events can be represented). For example, a blue LED indicates an ongoing control channel handshake, a green LED indicates an ongoing data channel handshake, and a red LED indicates transmitting a cooperative message. Other events are indicated by LED combinations. Fig. 18 is a snapshot in an experiment.

In our experiments, the transmission power is 0 dBm which is the maximum on CC2420. Nodes are configured as disjoint flows in an indoor area, and source nodes are always backlogged. Three channels are used as one control channel and two data channels, each with bandwidth 250 Kbps.[5] In collecting statistics for each of the four protocols, each single data point is by averaging over six experiments and each experiment runs for 360 actual seconds.

The experimental results are presented in Fig. 19. When the number of nodes is four, the two MRU protocols have about twice throughput of the two RAND protocols. This is because MRU strategy in effect assigns each pair a *dedicated* data channel, while RAND strategy encounters channel conflicts with probability 0.5 at each selection (there are two data channels). The reason why CAMMAC-RAND and UNCOOP-RAND perform the same is that, any time when a transmitter-receiver pair selects a channel conflicting with the other pair, there is no additional node on the control channel to cooperate.

5. By this setting, all nodes are within the radio range of each other, which was also used by So et al. [15]. We leave multihop experiments as future work due to practical constraints such as cost and complexity. In addition, we do not use a large number of channels which require many nodes to achieve throughput saturation but do not change the trends that we will show.

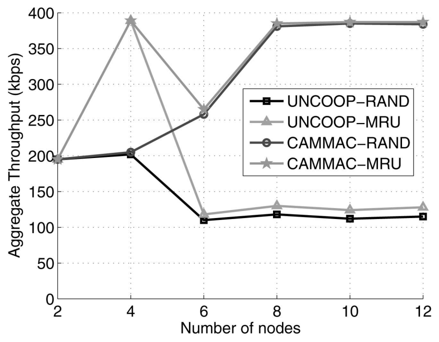


Fig. 19. Experimental results. The maximum utilizable bandwidth is 500 Kbps.

When the number of nodes is six, throughput of all protocols sharply declines except for CAMMAC-RAND. This is similar to the simulation results in Fig. 10c at 12 nodes, and the explanation is that the deprivation of MRU channels, due to one more node pair, degrades MRU strategy to RAND strategy. CAM-MAC achieves a moderate throughput of $\sim$260 Kbps and is more than UNCOOP because the two more nodes can occasionally provide cooperation.

Note that this performance degradation at six nodes is consistent with the comparison with MMAC shown in Fig. 14a, where CAM-MAC has only marginal improvement over MMAC in the case of six nodes and two data channels.

Beyond six nodes, the throughput of CAM-MAC recovers since cooperative nodes become often available. RAND and MRU strategies do not make much difference due to the reason described in the simulation results. Finally, at saturation, CAM-MAC more than *triples* the throughput of UNCOOP. Our experimental results further justify the value of cooperation.

7 Discussion

7.1 Availability of Cooperation

An important issue related to CAM-MAC is how likely a node can obtain cooperation. This is addressed in [35] where we proposed a metric p_{co}, which is defined as the probability of obtaining cooperation when an MCC problem is created, to characterize the availability of cooperation. We analyzed this metric in multihop multichannel networks, and the results show that it is high (> 0.7) in most cases. While cooperation is not always available, it does not mean that, on the average, a percentage of $1 - p_{co}$ handshakes will suffer from MCC problems; the percentage is in fact much lower. This is because p_{co}, by its definition, only counts those handshakes that create MCC problems (and not those that will succeed in data transmission without cooperation). Therefore, the probability that an arbitrary handshake will suffer from an MCC problem is lower than $1 - p_{co}$. Combined with the high level of p_{co}, this helps explain why CAM-MAC can significantly outperform UNCOOP even if cooperation is not always available.

7.2 Two-Hop Neighbor Discovery

CAM-MAC needs two-hop neighbor information for cooperation. To visualize this, see Fig. 1a again, where node C can only cooperate if it knows that node U_2 is adjacent to node A_1, though U_2 is not adjacent to C itself. One simple way to acquire this information is to make use of the `Hello` messages traditionally used in (one-hop) neighbor discovery or other broadcast messages used in routing, etc. Specifically, each node simply piggybacks its neighbors' IDs as well as its own ID when sending `Hello` messages, and nodes that receive this message can easily learn the two-hop neighbor information. This process does not noticeably incur overhead and complexity because it occurs in a very low frequency or only in the initialization phase, and can reuse the existing one-hop neighbor discovery process.

7.3 Impact of Mobility

Mobility is a major factor attributing to network dynamics and affecting the reliability of (one-hop and two-hop) neighbor information. One simple way of adapting CAM-MAC to a mobile environment is to accordingly increase the frequency of updating neighbor information. We conducted multihop simulations using random waypoint model [36], with the same setup as in Section 5.2. We do the same three sets of simulations (varying traffic, payload size, and node density), except that each node moves at a speed uniformly distributed in (0, 10] m/s and toward a randomly chosen target point for each movement. Each node independently updates neighbor information every 8 seconds. Our results showed only a marginal (3 percent-8 percent) performance degradation in comparison to the static scenarios in Figs. 11, 12, and 13. The details are omitted for space constraints. Actually, these results are not surprising because 1) in essence, cooperation is not a *compulsory* coordination mechanism but an *auxiliary* helping mechanism, which means that communication can proceed without cooperation, and 2) *one cooperative node is enough* to prevent MCC problems, and thus, mobility can rarely make all neighboring nodes fail to cooperate. Consequently, cooperation is robust to low mobility levels which is the case in most application scenarios.

7.4 Energy Consumption

Energy consumption is a critical issue for battery-powered devices commonly used in ad hoc networks. To save energy and prolong network lifetime, nodes should be kept in sleep mode as long as possible. However, they also need to stay active to gather channel usage information for both self-use (select free channels and receivers) and for cooperation purposes. This dilemma presents a challenge to multichannel MAC protocol design especially in a cooperative environment. In this paper, as an initial attempt of designing cooperative protocols, we focus on throughput improvement while do not specifically consider energy. This is also for a fair comparison with other state-of-the-art protocols, as those protocols do not consider energy either. Nevertheless, we recognize that energy conservation is an important issue and have addressed it in a separate work [37].

7.5 Multichannel Sensor Networks

Wireless sensor networks were initially motivated by low data rate applications, but new applications demanding higher throughput and/or lower delay quickly emerged after a few years, such as those in wireless multimedia sensor networks [38]. Current sensor platforms, however, only provide very limited bandwidth, e.g., 19.2 Kbps on MICA2 [39], and 250 Kbps on MICAz [39] and Telos [29]. On the other hand, some RF transceivers such as CC2420 used by MICAz and Telos provide multiple frequency channels. Therefore, we believe that multichannel sensor MAC protocols are both needed and feasible. Two such protocols, MMSN [40] and CMAC [41], recently appeared. However, MMSN is highly complicated and requires tight clock synchronization, and CMAC requires a large number of channels to be operable. In our future work, we would like to investigate these issues and particularly consider cooperative sensor protocols.

8 Conclusion

In this paper, we have introduced *DISH*, which is a distributed flavor of *control-plane cooperation*, as a new approach to wireless protocol design. It enables transmitter-receiver pairs to exploit the knowledge at individual idle neighbors to make more informed decisions in communication. Applying DISH to multichannel ad hoc networks, we propose a cooperative multichannel MAC protocol called CAM-MAC, where idle neighbors share control information with transmitter-receiver pairs to overcome MCC problems. This protocol uses a single transceiver and, unlike many other protocols, is fully asynchronous.

The simple idea of DISH turns out to be very effective. In the comparison of CAM-MAC with and without DISH, we observe remarkable performance difference. In the comparison with three recent and representative multichannel MAC protocols, MMAC, SSCH, and AMCP, CAM-MAC significantly outperforms all of them. Our implementation on COTS hardware and experiments further validates the advantages of CAM-MAC and the DISH idea.

In a sense, DISH enables nodes to *store* channel usage information at their neighbors, and *retrieve* this information when it is needed. We also highlight that this is not a compulsory *coordination* mechanism; a network does not *rely on* cooperation and still operates when cooperation is not available. Ultimately, we believe that control-plane cooperation merits due consideration in the future design of wireless network protocols.

Acknowledgments

Part of this work was presented in [1]. This work was supported in part by the Temasek Defense Science Institute at the National University of Singapore under Grants R-263-000-395-592, R-263-000-396-422, and R-263-000-396-592.

References

[1] T. Luo, M. Motani, and V. Srinivasan, "CAM-MAC: A Cooperative Asynchronous Multi-Channel MAC Protocol for Ad Hoc Networks," *Proc. IEEE Third Int'l Conf. Broadband Comm., Networks and Systems (BROADNETS '06)*, Oct. 2006.

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

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

[4] J. Shi, T. Salonidis, and E.W. Knightly, "Starvation Mitigation through Multi-Channel Coordination in CSMA Multi-Hop Wireless Networks," *Proc. ACM MobiHoc*, 2006.

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

[6] A. Nasipuri, J. Zhuang, and S.R. Das, "A Multichannel CSMA MAC Protocol for Multihop Wireless Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, 1999.

[7] A. Nasipuri and J. Mondhe, "Multichannel CSMA with Signal Power-Based Channel Selection for Multihop Wireless Networks," *Proc. IEEE Vehicular Technology Conf. (VTC)*, 2000.

[8] N. Jain, S.R. Das, and A. Nasipuri, "A Multichannel CSMA MAC Protocol with Receiver-Based Channel Selection for Multihop Wireless Networks," *Proc. 10th Int'l Conf. Computer Comm. and Networks (ICCCN)*, 2001.

[9] A. Adya, P. Bahl, J. Padhye, and A. Wolman, "A Multi-Radio Unification Protocol for IEEE 802.11 Wireless Networks," *Proc. IEEE First Int'l Conf. Broadband Comm., Networks and Systems (BROADNETS)*, 2004.

[10] R. Maheshwari, H. Gupta, and S.R. Das, "Multichannel MAC Protocols for Wireless Networks," *Proc. Third Ann. IEEE Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. and Networks (SECON)*, 2006.

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

[12] J. Zhang, G. Zhou, C. Huang, S.H. Son, and J.A. Stankovic, "TMMAC: An Energy Efficient Multi-Channel MAC Protocol for Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm. (ICC)*, 2007.

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

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

[15] H.-S.W. So, G. Nguyen, and J. Walrand, "Practical Synchronization Techniques for Multi-Channel MAC," *Proc. ACM MobiCom*, 2006.

[16] L. Huang and T.-H. Lai, "On the Scalability of IEEE 802.11 Ad Hoc Networks," *Proc. ACM MobiHoc '02*, pp. 173-182, 2002.

[17] R. Maheshwari, H. Gupta, and S.R. Das, "Multichannel MAC Protocols for Wireless Networks," *Proc. Third Ann. IEEE Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. and Networks (SECON)*, 2006.

[18] R. Vedantham, S. Kakumanu, S. Lakshmanan, and R. Sivakumar, "Component Based Channel Assignment in Single Radio, Multi-Channel Ad Hoc Networks," *Proc. ACM MobiCom*, 2006.

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

[20] N. Shacham and P. King, "Architectures and Performance of Multichannel Multihop Packet Radio Networks," *IEEE J. Selected Areas in Comm.*, vol. 5, no. 6, p. 1013C1025, 1987.

[21] P. Kyasanur and N.H. Vaidya, "Routing and Link-Layer Protocols for Multi-Channel Multi-Interface Ad Hoc Wireless Networks," *Mobile Computing and Comm. Rev.*, vol. 10, no. 1, pp. 31-43, Jan. 2006.

[22] J. Mo, H.W. So, and J. Walrand, "Comparison of Multichannel MAC Protocols," *Proc. Eighth ACM/IEEE Int'l Symp. Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2005.

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

[24] Y. Wu, J.A. Stankovic, T. He, and S. Lin, "Realistic and Efficient Multi-Channel Communications in Wireless Sensor Networks," *Proc. IEEE INFOCOM '08*, pp. 1867-1875, 2008.

[25] H.K. Le, D. Henriksson, and T. Abdelzaher, "A Practical Multi-Channel Media Access Control Protocol for Wireless Sensor Networks," *Proc. IEEE/ACM Int'l Conf. Information Processing in Sensor Networks (IPSN '08)*, pp. 70-81, 2008.

[26] *HostAP*, http://hostap.epitest.fi/, 2008.

[27] *MadWifi,* http://madwifi.org/, 2008.

[28] *TinyOS Community Forum,* http://www.tinyos.net, 2008.

[29] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling Ultra-Low Power Wireless Research," *Proc. IEEE/ACM Fourth Int'l Conf. Information Processing in Sensor Networks (IPSN/SPOTS '05),* Apr. 2005.

[30] Chipcon Corporation, *CC2420 2.4 GHz Zigbee/802.15.4 RF Transceiver,* http://www.chipcon.com, 2008.

[31] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Culler, "Exploiting the Capture Effect for Collision Detection and Recovery," *Proc. Second IEEE Workshop Embedded Networked Sensors (EMNETS),* 2005.

[32] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys),* 2003.

[33] N. Reijers, G. Halkes, and K. Langendoen, "Link Layer Measurements in Sensor Networks," *Proc. IEEE Int'l Conf. Mobile Ad Hoc and Sensor Systems (MASS),* 2004.

[34] D. Lal, A. Manjeshwar, F. Herrmann, E. Uysal-Biyikoglu, and A. Keshavarzian, "Measurement and Characterization of Link Quality Metrics in Energy Constrained Wireless Sensor Networks," *Proc. IEEE Global Telecomm. Conf. (GLOBECOM '03),* 2003.

[35] T. Luo, M. Motani, and V. Srinivasan, "Analyzing DISH for Multi-Channel MAC Protocols in Wireless Networks," *Proc. ACM MobiHoc,* 2008.

[36] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proc. ACM MobiCom '98,* pp. 85-97, 1998.

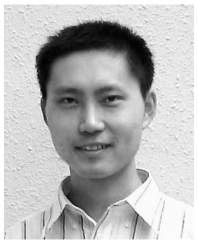
[37] T. Luo, M. Motani, and V. Srinivasan, "Altruistic Cooperation for Energy-Efficient Multi-Channel MAC Protocols," *Proc. ACM MobiCom,* 2007.

[38] I.F. Akyildiz, T. Melodia, and K.R. Chowdhury, "A Survey on Wireless Multimedia Sensor Networks," *Computer Networks,* no. 51, pp. 921-960, 2007.

[39] Crossbow Technology Inc., http://www.xbow.com, 2008.

[40] G. Zhou, C. Huang, T. Yan, T. He, J. Stankovic, and T. Abdelzaher, "MMSN: Multi-Frequency Media Access Control for Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2006.

[41] K.R. Chowdhury, N. Nandiraju, D. Cavalcanti, and D.P. Agrawal, "CMAC—A Multi-Channel Energy Efficient MAC for Wireless Sensor Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC '06),* pp. 1172-1177, 2006.



Tie Luo received the BEng degree in computer communications from the Beijing University of Posts and Telecommunications, China. He is to receive the PhD degree in electrical and computer engineering from the National University of Singapore, and is currently a research fellow at the National University of Singapore. His research interests are in the area of wireless networks, including ad hoc, sensor, mesh, WiMAX, and cognitive radio networks. He is a student member of the IEEE.

Mehul Motani received the BS degree from Cooper Union, the MS degree from Syracuse University, and the PhD degree from Cornell University, all in electrical and computer engineering. He is an assistant professor in the Department of Electrical and Computer Engineering at the National University of Singapore. He has also been a member of the technical staff at the Institute for Infocomm Research in Singapore for three years and a member of the technical staff at Lockheed Martin in Syracuse, New York, for over four years. His research interests are in the area of wireless networks. He has recently been working on research problems which sit at the boundary of information theory, networking, and communications, including the design of wireless ad hoc and sensor network systems. He was awarded the Intel Foundation Fellowship for work related to his PhD degree in 2000 and won the Telecom Italia Mobile prize at SIMAGINE in 2003. He has served on the organizing committees of ISIT and ICCS and the technical program committees of MobiCom, InfoCom, SECON, and various other conferences. He participates actively in the IEEE and ACM and has served as the secretary of the IEEE Information Theory Society Board of Governors. He is a member of the IEEE.

Vikram Srinivasan received the BS degree in physics from the University of Chennai in 1994, the ME degree in electrical communications engineering from the Indian Institute of Science, Bangalore, in 1998, and the PhD degree in electrical and computer engineering from the University of California at San Diego in 2003. He was an assistant professor in the Department of Electrical and Computer Engineering, National University of Singapore, from 2003 to 2007. He is currently with Bell Labs Research, India. His research interests include the area of wireless networks. He is a member of the IEEE.