Energy-Efficient Strategies for Cooperative Multichannel MAC Protocols

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

Abstract—Distributed Information SHaring (DISH) is a new cooperative approach to designing multichannel MAC protocols. It aids nodes in their decision making processes by compensating for their missing information via information sharing through neighboring nodes. This approach was recently shown to significantly boost the throughput of multichannel MAC protocols. However, a critical issue for ad hoc communication devices, viz. energy efficiency, has yet to be addressed. In this paper, we address this issue by developing simple solutions that reduce the energy consumption without compromising the throughput performance and meanwhile maximize cost efficiency. We propose two energy-efficient strategies: *in-situ energy conscious DISH*, which uses existing nodes only, and *altruistic DISH*, which requires additional nodes called altruists. We compare five protocols with respect to these strategies and identify altruistic DISH to be the right choice in general: it 1) conserves 40-80 percent of energy, 2) maintains the throughput advantage, and 3) more than doubles the cost efficiency compared to protocols without this strategy. On the other hand, our study also shows that in-situ energy conscious DISH is suitable only in certain limited scenarios.

Index Terms—Control-plane cooperation, altruistic DISH, in-situ energy conscious DISH, wireless ad hoc networks.

◆

1 INTRODUCTION

Using multiple channels in communication is key to improving the quality of service for wireless networks, and multichannel MAC protocol design has thereby attracted substantial attention from the research community. Various design approaches have been proposed in the last decade or so, but most of them require either multiple radios or time synchronization. Recently, Luo et al. [2] proposed a distinct approach called Distributed Information SHaring (DISH), which uses a single radio but operates asynchronously. The authors designed a DISH-based protocol called CAM-MAC [2], in which neighboring nodes share *control information* with each sender-receiver pair to facilitate it to choose collision-free channels or to avoid busy receivers. DISH is essentially a form of node cooperation, but the key difference is that, in traditional cooperation, intermediate nodes help relay *data* for source and destination nodes, but DISH, on the other hand, only requires *control* information to be sent. Therefore, the former can be called *data-plane cooperation* and the latter can be called *control-plane cooperation*.

This approach has been extensively evaluated in [2] using the CAM-MAC protocol. The results demonstrate significant throughput improvement compared to non-DISH-based protocols, including existing representative multichannel MAC protocols.

However, the issue of energy consumption was not considered in the prior work. This is a crucial issue as DISH is designed for ad hoc communication devices which are mostly battery powered. In this paper, for a quantitative understanding, we first conduct simulation to compare CAM-MAC with two protocols, Non-DISH and Non-DISH-psm where:

- Non-DISH is CAM-MAC with the DISH element removed, i.e., neighbors do not share information with senders and receivers who will hence make decisions on their own. Basically, this is a (traditional) noncooperative protocol.
- Non-DISH-psm is Non-DISH using an ideal power saving mode (psm), where each node switches on its radio only when sending/receiving its own packets, i.e., sleep when idle (no overhearing).

More protocol details will be described in Section 3.3. Our simulation results show that, although the throughput of CAM-MAC is 2.65 times Non-DISH and even more than Non-DISH-psm, its energy consumption is 2.94 times Non-DISH-psm and comparable to Non-DISH (detailed results will be given in Section 6). This conveys the message that there is potentially large space for improvement in energy efficiency.

In this paper, we propose two energy-efficient strategies, *in-situ energy conscious DISH* and *altruistic DISH*, to address this issue. In the in-situ strategy, existing nodes rotate the responsibility of information sharing such that nodes without this responsibility can sleep when idle in order to save power. In the altruistic strategy, additional nodes called *altruists* are deployed to take over the responsibility of information sharing so that all the existing nodes can sleep when idle.

- *T. Luo is with the Department of Electrical and Computer Engineering, National University of Singapore, E4-06-12, Singapore 117576. E-mail: eleluo@nus.edu.sg.*
- *M. Motani is with the Department of Electrical and Computer Engineering, National University of Singapore, E4-05-18, 10 Kent Ridge Crescent, Singapore 119260. E-mail: 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, Bengaluru 560095, India. E-mail: vikramsr@alcatel-lucent.com.*

Manuscript received 17 Dec. 2009; revised 6 Nov. 2010; accepted 21 Jan. 2011; published online 17 Mar. 2011.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2009-12-0545.
Digital Object Identifier no. 10.1109/TMC.2011.60.

We conduct qualitative and quantitative investigation on the strategies with the following objectives: 1) reduce energy consumption, 2) maintain or not compromise the high throughput achieved by DISH, and 3) maximize cost efficiency. Yet, the solution must be kept as simple as possible. Via comparing five protocols with respect to these two strategies, our study recommends altruistic DISH in general and in-situ energy conscious DISH only in certain limited scenarios.

We also implemented these protocols on an embedded system based test-bed and carried out experiments. The results further confirmed our findings. Moreover, neither of the two strategies requires multiple radios or time synchronization, which translates to lower cost, smaller hardware size, and low complexity.

The rest of the paper is organized as follows: Section 2 explains DISH in more detail. Section 3 elaborates and gives a qualitative analysis of our proposed strategies, where three important issues are identified: optimal node deployment, cost efficiency, and throughput-energy trade-off. These issues are subsequently investigated in Sections 4, 5, and 6, respectively. We then present our hardware implementation and experiments in Section 7. Next, relevant issues are discussed in Section 8 and related work is reviewed in Section 9. Finally, Section 10 concludes this paper.

2 UNDERSTANDING DISH

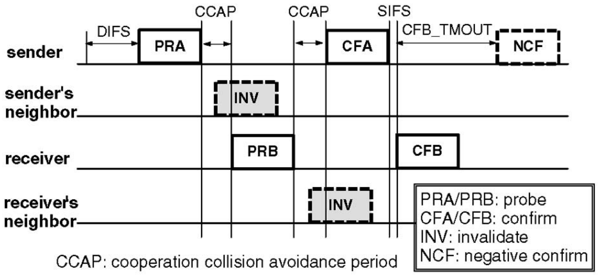
Control information is crucial to communications but can be missing due to various reasons such as shadowing and noise. The dominant reason, however, in a multichannel environment, is that nodes fail to tune radios to certain channels in time, or that a radio can only listen to one channel at a time. This causes the multichannel coordination (MCC) problem which has two variants: 1) channel conflict problem, created when a node selects a busy channel (being used by other nodes), and 2) deaf terminal problem, created when a sender attempts to communicate with a receiver that is on a different channel.

One category of solutions are to dedicate an extra radio to each channel or a common control channel in order not to miss information, as proposed by Wu et al. [3], Nasipuri et al. [4], Nasipuri and Mondhe [5], Jain et al. [6], Adya et al. [7], Maheshwari et al. [8]. However, such solutions will inevitably increase hardware cost and size (and energy consumption as well). Another category of solutions do not require multiple radios but require communication to be set up in specified time slots [9], [10], [11] or require periodic channel switching according to certain sequences [12], [13], [14]. Thus, they rely on time synchronization which adds considerable complexity [15] and degrades scalability [16], especially for multihop networks.

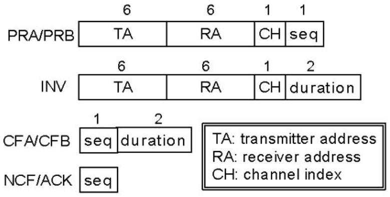
The basic idea of DISH is to compensate for nodes' missing information via cooperation. It exploits neighboring nodes as a resource to "retrieve" missing information from, like from a distributed database, when needed. The need for multiple radios or time synchronization, naturally becomes not necessary.

2.1 DISH-p: A DISH-Based Protocol

For a more tangible understanding, we describe a DISH-based protocol called DISH-p (which was CAM-MAC described in [2]). In DISH-p, a sender and a receiver set up communication using PRA/PRB packets and then confirm using CFA/CFB packets. A neighbor will send INV packet if it identifies an MCC problem via the information carried by PRA/PRB.



(a) Control channel handshake.



(b) Frame format. INV carries the channel usage information of an established and ongoing data exchange on a data channel (which engages the "deaf" receiver in the case of deaf terminal problem).

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

(c) Channel usage table. Each node maintains one to cache its overheard control information.

Fig. 1. Elements of the DISH-p protocol.

One channel is designated as the common control channel and the rest are designated as multiple data channels. On the control channel, a sender and a receiver exchange PRA/PRB (see Fig. 1a) to select a data channel, and then exchange CFA/CFB to confirm the channel selection. The frame format is shown in Fig. 1b. If a neighbor identifies an MCC problem (via PRA or PRB), it will prepare to send an INV packet, during a cooperation collision avoidance period (CCAP), to alarm the sender or the receiver to back off. If there is no MCC problem identified by any neighbor (no INV will be sent), the sender and the receiver will switch to their chosen data channel and start DATA/ACK exchange. During DIFS and CCAP, carrier sensing is turned on to mitigate collisions via CSMA.

CCAP is introduced to mitigate the collision of multiple simultaneously sent INVs. A neighbor who identifies an MCC problem will send INV only if it senses the control channel to be free for a period of *Uniform*[0, CCAP]. Hence a neighbor who sends INV will suppress its neighbors via CSMA.[1] NCF is sent when the sender waits for CFB until time-out (due to the receiver receiving INV), in order to inform the sender's neighbors to disregard CFA.

---

[1] CSMA does not avoid all collisions because not all the neighbors may hear each other. However, a collision of such still conveys an alarm to the sender/receiver because INV represents a *negative* message, and hence the sender/receiver will still back off. What is only compromised is that the sender/receiver will not know precisely how long at least it should back off and hence will have to estimate a backoff period, which has been verified not to be a serious problem.

The applicable scenarios of the protocol are mesh networks and ad hoc networks, not sensor networks. In sensor networks, data packets are usually small and the overhead of the control channel handshake will be significant. Even using a packet train would not suit because sensing traffic is usually periodic and not bursty.

3 ENERGY-EFFICIENT STRATEGIES

The main challenge to achieving energy efficiency for DISH is that a prerequisite of information sharing is *information gathering*, a process that requires nodes to stay awake for overhearing, which presents a challenge for nodes to switch off radio when idle. The strategies we elaborate below meet this challenge and we also provide a qualitative analysis below.

3.1 In-Situ Energy Conscious DISH

In this strategy, all the existing nodes rotate the responsibility of information sharing (i.e., cooperation) such that nodes without the responsibility can sleep when idle.[2] There are two methods to implement this strategy:

- Probabilistic method: Each node decides whether to cooperate or not according to a (static or dynamic) probability. This is similar to probabilistic flooding [17], [18], [19] and probabilistic routing [20], [21] in ad hoc networks, and cluster-head rotating algorithms (e.g., LEACH [22] and HEED [23]) in sensor networks.
- Voting method: nodes periodically vote or elect a subset of nodes to cooperate. This is similar to GAF [24], Span [25], PANEL [26], and VCA [27].

An apparent advantage of the in-situ strategy is that it does not require additional nodes. On the other hand, a runtime probabilistic or voting mechanism must be introduced and must be 1) distributed, 2) fair (in terms of energy consumption), and 3) adaptive (to network dynamics such as traffic and energy drainage). These would introduce considerable complexity and overhead. In addition, it has to consider other factors as listed below.

First, the mechanism would rely on message broadcast as also used by Ni et al. [17], Yassein et al. [18], Zhang and Agrawal [19], Roosta et al. [21], Xu et al. [24], Chen et al. [25], Qin and Zimmermann [27]. However, broadcasting in a multichannel environment is shown by So et al. [15] to be very unreliable and difficult because each broadcast can reach only a subset of neighboring nodes. Alternatively, broadcasts might be reduced or avoided by determining cooperative nodes based on geographic information, like in [20], [24], [26]. However, this requires expensive GPS support or a distributed localization algorithm (e.g., [28], [29]) which introduces additional overhead and complexity to those incurred by rotation itself.

Second, rotating the responsibility of cooperation also involves other resource-consuming factors including two-hop neighbor discovery (shown in [2], [30]) and the assessment of dynamic information (such as energy and traffic, like in [21], [22], [25]).

Third, how to integrate a probabilistic or voting mechanism into a legacy DISH protocol is a nontrivial problem and a viable solution is yet to be found.

In summary, the complexity, overhead, and unreliability of in-situ energy conscious DISH would consume considerable resource and eventually negate its possible performance gain. Nonetheless, for a quantitative understanding, we still evaluate this strategy using a *Genie In-Situ* protocol (detailed in Section 3.3) which establishes an upper bound for all such in-situ protocols.

3.2 Altruistic DISH

In this strategy, additional nodes called *altruists* are deployed to take over the responsibility of information sharing (i.e., cooperation) from the existing nodes, which we call *peers* to distinguish from altruists, so that peers can sleep when idle. Altruists are the same as peers in terms of hardware, but are different in terms of software: they solely cooperate (do not carry data traffic) and always stay awake.

An apparent drawback of this strategy is that it requires additional nodes. However, this is offset by substantive advantages. First, it is very simple to implement the strategy: one only needs to introduce a Boolean flag to disable data related functions on altruists and cooperation related functions on peers. We have done this in both our simulation code and hardware implementation code. Equally importantly, there is no additional runtime mechanism and hence runtime overhead.

Second, unlike the in-situ strategy, this strategy does not have the multichannel broadcasting problem. Altruists always stay on the same channel (control channel) and send/receive packets only on the control channel.

Third, this strategy is robust to network dynamics (such as traffic and residual energy). Every altruist is cooperative and will react to every MCC problem that it identifies; they do not need to adjust any parameter on the fly. In fact, even the deployment of altruists, which is an offline process, can be done with a constant number for any given peer density, as will be shown in Section 4

Fourth, since peers only carry data traffic and need not to cooperate, they are like nodes in traditional (non-DISH) networks and thus can adopt a legacy sleep-wake scheduling algorithm, where a lot of choices are available and will be provided in Section 9.

Finally, unlike the in-situ strategy and the original DISH where cooperation is provided in an *opportunistic* manner—meaning that cooperative nodes are not always available, altruistic DISH provides cooperation in a *guaranteed* manner.

3.3 Protocols to Investigate

In the sequel, we investigate *Genie In-Situ* and *Altruistic*, which are two protocols made by respectively applying the above two strategies to DISH-p (the original DISH protocol). For the purpose of comparison, we also introduce two non-DISH protocols, one with and the other without power saving, viz. *Non-DISH* and *Non-DISH-psm*. The following describes all the five protocols.

1. *DISH-p*: the protocol described in Section 2.1.
2. *Non-DISH*: a (traditional) noncooperative protocol, derived from DISH-p by removing the cooperative

---

[2] We say that a node is idle if it is not engaged in sending/receiving its *own* packets. For example, overhearing (other packets) and waiting for free data channels (though with data packets in queue) are both idle.

element, i.e., neighbors do not share control information with senders or receivers.

3. *Non-DISH-psm*: Non-DISH with a power saving mode, where each node only turns on its radio when sending/receiving packets addressed from/to itself (i.e., they do not overhear). This is an ideal mode because it assumes a receiver can automatically wake up upon a communication request from a sender. We use this rather than adopt an existing sleep-wake scheduling algorithm (which will be reviewed in Section 9) in order to avoid coupling performance to a specific algorithm. Besides, this still keeps our comparison fair because the same PSM will be used by all the other power-saving protocols (*Genie In-Situ* and *Altruistic*).
4. *Genie In-Situ*: this protocol is DISH-p with the in-situ strategy applied. It uses a genie-aided (optimal) rotating mechanism in order to establish upper bound performance for the in-situ strategy. In this protocol, upon each occurrence of an MCC problem, the best neighbor will be chosen (by the genie) to cooperate[3] and all the other neighbors are treated as virtually sleeping (not consuming energy though having gathered information via overhearing) as per the ideal PSM.
5. *Altruistic*: this protocol is DISH-p with the altruistic strategy applied. Altruists stay awake to gather information and, upon identifying an MCC problem, share information (cooperate). All existing nodes do not cooperate and they adopt the ideal PSM to sleep when idle.

3.4 Issues to Investigate

There are three relevant issues that need to be addressed:

1. *Node deployment* (addressed in Section 4): how to deploy altruists for Altruistic DISH.
2. *Cost efficiency* (addressed in Section 5): we propose a metric called bit-meter-price (BMP) ratio which takes into account various factors to measure the overall performance of a protocol.
3. *Throughput-energy trade-off* (addressed in Section 6): zooms in to specifically inspect the throughput and energy performance.

In the rest of the study we assume an ad hoc network with static topology. Each node has a single half-duplex radio that can dynamically switch among all available channels but can only use one at a time. One channel is designated as a control channel and the others as data channels. Data channel selection is random, meaning that a sender/receiver randomly selects one from a list of data channels that it deems free based on its knowledge which it dynamically updates (e.g., channel usage table as in Fig. 1c).[4] Finally, we assume all links are bidirectional, i.e., if node u can hear node v then v can hear u as well.

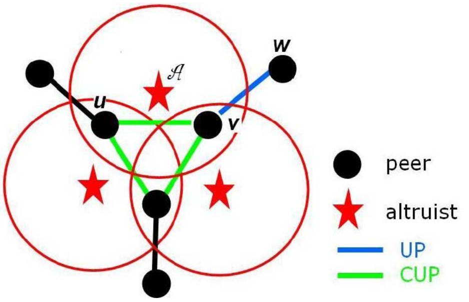


Fig. 2. Illustration of UP and CUP. Node pair (u, v) is a CUP (covered by altruist $\mathcal{A}$) while (v, w) is an UP. Each circle denotes the transmission range of an altruist.

4 OPTIMAL NODE DEPLOYMENT

As a prerequisite, we need to develop a concept called *cooperation coverage*.

4.1 Cooperation Coverage

Definition 1 (UP and CUP). *An unsafe pair (UP) is a pair of peers that can create MCC problems to each other. A covered unsafe pair (CUP) is an UP that both peers are within the transmission range of at least one common altruist.*

An illustration of UP and CUP is given in Fig. 2, and the necessary and sufficient condition for creating MCC problems (i.e., forming a UP) is given in Proposition 1. Briefly speaking, two adjacent peers can create MCC problems if each of them has other communicable neighbor (s), because one peer may switch to a data channel and miss information of the other peer.

Proposition 1. *In an undirected graph where each vertex represents a peer and each edge represents the relationship between two neighboring peers, denote by d_i the degree of an arbitrary vertex i. If PSM is not used, two adjacent vertices i and j form an UP if and only if:*

1. $d_i \geq 2$, $d_j \geq 2$, *and* $d_i = d_j = 2$ *does not hold, or*
2. $d_i = d_j = 2$, *and* i *and* j *are not on the same three-cycle (i.e., triangle).*

If PSM is used (peers sleep when idle), the above condition remains unchanged for the channel conflict problem, but changes to the following for the deaf terminal problem: $d_i \geq 1$, $d_j \geq 1$, *and* $d_i = d_j = 1$ *does not hold.*

See the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2011.60, for the proof.

Definition 2 (Cooperation Coverage—p_{cov}).

$$p_{cov} \triangleq \frac{N_{cup}}{N_{up}},$$

---

[3] The best neighbor is a neighbor with the most helpful information when an MCC problem occurs. For example, in a channel conflict problem where a node u chooses a busy data channel which is used by multiple sender-receiver pairs (consider a multihop environment), the best neighbor is the one who knows which pair has the *longest residual time* in using that channel—this neighbor can inform node u of the minimum duration to back off for.

[4] Another channel selection method is first to try the previously used channel and, if not available, then random selection. Both methods were studied in [2] and shown, in most cases, to result in only marginal difference in the context of DISH.

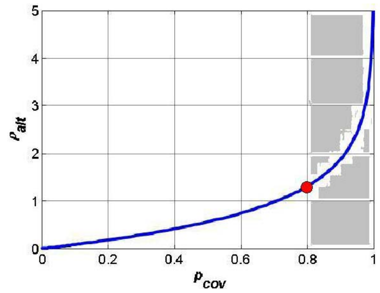


Fig. 3. ρ_{alt} versus p_{cov} (Theorem 1).

where N_{cup} is the number of CUPs and N_{up} is the number of UPs in a network.

We say that a network achieves *full cooperation coverage* if $p_{cov} = 100\%$.

Proposition 2. *Consider a network using altruistic DISH. In order to achieve free of MCC problems, full cooperation coverage is*

1. *necessary for a multihop network, and*
2. *necessary and sufficient for a single-hop network.*

See the Appendix, available in the online supplemental material, for the proof.

4.2 Random Deployment

In random deployment, all nodes are uniformly distributed in a plane region.

Theorem 1. *Consider an infinite network where peers and altruists are randomly distributed (as per a two-dimensional Poisson point process). If the peer density is ρ_{peer}, then in order to achieve a cooperation coverage of p_{cov}, the altruist density, ρ_{alt}, must satisfy*

$$\rho_{alt} > -\frac{\ln(1-p_{cov})}{\left(\frac{2\pi}{3}-\frac{\sqrt{3}}{2}\right)r^2}. \tag{1}$$

Proof. Denote by p_{ij}^{cov} the probability that an arbitrary UP (i,j) is covered (i.e., a CUP). By Definition 1, p_{ij}^{cov} is equivalent to the probability that at least one altruist exists in the common transmission range of i and j, which is given by

$$p_{ij}^{cov} = 1 - e^{-\rho_{alt} A_{ij}}, \tag{2}$$

where A_{ij} is the intersected area of i and j's transmission ranges, and can be proven using simple geometric techniques to be

$$A_{ij} = 2r^2\theta - r^2 \sin 2\theta, \tag{3}$$

where $\theta = \arccos\frac{d}{2r}$, d is the euclidean distance between i and j, and r is the transmission range.

To achieve p_{cov} is equivalent to achieving $p_{ij}^{cov} > p_{cov}$ for all UPs (i,j), meaning

$$\min_{(i,j)} p_{ij}^{cov} > p_{cov}. \tag{4}$$

According to (2), p_{ij}^{cov} is a monotonically increasing function of A_{ij}, and hence is minimized by minimizing A_{ij}. To minimize A_{ij}, consider the minimization domain, namely all UPs. According to (3), A_{ij} is a monotonically decreasing function of d. Since $d \in [0,r]$, A_{ij} is therefore minimized at $d = r$:

TABLE 1
Some Discrete Values of ρ_{alt} versus p_{cov}

p_{cov}	50%	60%	70%	80%	90%	95%	99%
$\rho_{alt} >$	0.56	0.75	0.98	1.31	1.87	2.44	3.75

$$\min_{(i,j)} A_{ij} = A_{ij}|_{d=r} = \left(\frac{2\pi}{3} - \frac{\sqrt{3}}{2}\right) r^2,$$

and thus (4) resolves to

$$\begin{aligned}\min_{(i,j)} p_{ij}^{cov} &= 1 - \exp\left(-\rho_{alt} \cdot \min_{(i,j)} A_{ij} \right)\\ &= 1 - \exp\left[-\rho_{alt} \cdot \left(\frac{2\pi}{3} - \frac{\sqrt{3}}{2}\right) r^2\right]\\ &> p_{cov},\end{aligned}$$

which is then reduced to (1). □

Theorem 1 gives the relationship between altruist density ρ_{alt} and cooperation coverage p_{cov}. Note that ρ_{peer} does not appear in (1). This is important because it implies that altruist deployment is independent of peer density and hence is remarkably simplified. This also makes significant practical sense because, in reality, the number of peers often varies or is uncertain.

Theorem 1 also shows that $\rho_{alt} \to \infty$ if $p_{cov} = 100\%$. This tells network planers not to aim at full cooperation coverage in multihop networks. For single-hop networks, it is easy to see that a single altruist achieves full cooperation coverage.

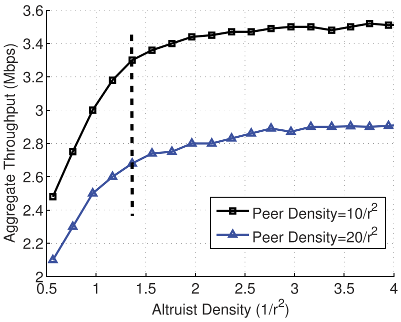
Fig. 3 plots the relationship between ρ_{alt} and p_{cov}, and Table 1 enumerates some discrete values. We can see that beyond the point ($p_{cov} = 80\%$, $\rho_{alt} = 1.31$), ρ_{alt} sharply increases, which indicates a cost spike. This motivates us to investigate the performance trend of altruistic DISH in a range including this point.

4.2.1 Simulation Setup and Results

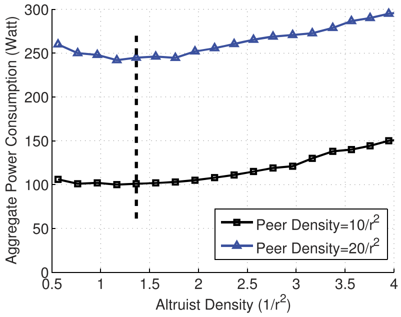
In our simulation, the metrics are aggregate end-to-end throughput and aggregate power consumption (including both peers and altruists if any). In order to compute power consumption, we conducted a survey of power consumption rates of commercial wireless cards. According to [31], a Cisco Aironet 350 series WiFi card consumes 2,250/1,350/75 mW in TX/RX/SLEEP state. According to [32] (with some simple calculation), an IEEE 802.11 WaveLAN PC card consumes 1,327/967/843/66 mW in TX/RX/IDLE/SLEEP state in the 2 Mbps category, and 1,346/901/741/48 mW in the 11 Mbps category. According to other respective sources, Intel Pro 2011, 3Com xJack, Compaq WL1000, and Siemens SS1021 all have the rates with the similar ratio as the above.[5] Therefore, we use the average rates based on our survey, namely 25/18/15/1 × 50 mW in the TX/RX/IDLE/SLEEP states, respectively, to calculate the power consumption in simulation.

---

[5] A (relative) ratio matters more than (absolute) rates as this is a comparative study which focuses on the difference between protocols.



(a) Aggregate throughput.



(b) Aggregate power consumption.

Fig. 4. Finding the optimal altruist density for *Altruistic*.

We set up the simulation as follows: Nodes are randomly placed in a plane area of $100\ \mathrm{m} \times 100$ m for single-hop networks and $1{,}500\ \mathrm{m} \times 1{,}500$ m for multihop networks. The radio transmission range is 250 m and the interference range is 500 m. The capture effect is enabled with a threshold of 6 dB. In single-hop networks, n peers randomly form $n/2$ disjoint flows. In multihop networks, n peers randomly form n nondisjoint flows (each peer is the source of one flow and also the destination of another flow). Shortest path routing is used. There are one control channel and five data channels with bandwidth 1 Mbps each. Packet arrival is Poisson, and data payload is 2 KB. PLCP is 15 bytes (header 6 bytes and preamble (short) 9 bytes). SIFS is 10 μs and CCAP is 35 μs.[6] Channel switching delay is ignored because it is common to all the protocols in this comparative study, and is not long (80 μs according to [14], equivalent to transmitting 10 bytes on an 1 Mbps channel). We use a discrete-event simulator which we developed on Fedora Core 5 with a Linux kernel of version 2.6.9. Each simulation is terminated after a total of 100,000 data packets are sent. All results are averaged over 15 randomly generated networks.

To investigate the performance of altruistic DISH around $\rho_{alt} = 1.31/r^2$, we run *Altruistic* in multihop networks by varying ρ_{alt} from $0.56/r^2$ to slightly more than $3.75/r^2$, which corresponds to varying p_{cov} from 50 percent to more than 99 percent. Data generation (at each peer) is Poisson with rate 25 kbps. The results for peer density (ρ_{peer}) of $10/r^2$ and $\rho_{peer} = 20/r^2$ are shown in Fig. 4 (the results for $5/r^2$ and $30/r^2$ have the similar trend and are omitted). We see that, irrespective of the value of ρ_{peer}, the increasing throughput starts to level off at a knee point at ρ_{alt} of $1.3 - 2/r^2$ (Fig. 4a), and the power consumption achieves the minimum also at ρ_{alt} of $1.3 - 2/r^2$ (Fig. 4b). This observation suggests a judicious choice of ρ_{alt} in this range. Hence, we adopt $\rho_{alt} = 1.31/r^2$ as a near-optimum value which corresponds to $p_{cov} = 80\%$.

The results are explained as follows. Adding altruists converts UPs into CUPs and thereby reduces collisions and retransmissions. This helps increase throughput and save energy as well. On the other hand, as more and more altruists are added, more and more UPs become *redundantly* covered (by more than one altruists), meaning that the growth of p_{cov} will slow down. This leads to the leveling off of throughput. In addition, since adding altruists contributes to a linear increase of energy consumption, the energy consumption starts to rise and deviate from the minimum.

4.3 Arbitrary Deployment

In arbitrary deployment, altruists can be carefully placed on a given topology formed by peers.

Theorem 2. *Consider a network with a given topology formed by peers on a finite plane. The problem of determining the minimum number and the locations of altruists to achieve full cooperation coverage, is NP-hard.*

See the Appendix, available in the online supplemental material, for the proof.

We remark on how to solve this problem in practice. In our proof, we have converted this problem into the classic set cover problem [33] which has approximate solutions using a number of greedy algorithms (see book [34]). In our particular case (node deployment), these algorithms can be executed offline and hence do not introduce any runtime overhead. Regarding the performance of these algorithms, Alon et al. [35] have recently established a lower bound to the approximation ratio, that such a greedy algorithm can achieve in polynomial time, to be $c \cdot \ln n$, where c is a constant coefficient and n is the number of elements to cover (i.e., UPs in our case).

A plausible thought is that we can carefully deploy altruists to cover the *entire region* and thereby achieve full cooperation coverage irrespective of the topology of peers. We can show that the minimum number of altruists to cover a rectangular area of $w \times h$ is $\lceil w/\sqrt{2}r \rceil \cdot \lceil h/\sqrt{2}r \rceil$. However, this argument is not true because covering an entire region is not equal to covering each UP (of two peers by a common altruist).

5 COST EFFICIENCY

We propose a metric called *bit-meter-price ratio* to measure the cost efficiency of a protocol.

5.1 Bit-Meter-Price Ratio

BMP is a network performance metric defined as

$$BMP \triangleq \frac{\overrightarrow{F} \cdot \overrightarrow{D} \cdot b_0}{(N_p + N_a) \cdot \max\left(P_p^{max}, P_a^{max}\right)}, \tag{5}$$

where $\overrightarrow{F}$ is a vector of all the flows' throughput, $\overrightarrow{D}$ is a vector of all the flows' source-to-destination euclidean

---

[6] CCAP needs to be sufficient for a node to detect signal transmission, not need to receive a complete message.

distances, N_p and N_a are the total number of peers and altruists, respectively, P_p^{max} and P_a^{max} are the maximum power consumption rate among all the peers and the altruists, respectively, $b_0 = e_0/c_0$, and e_0 and c_0 are the initial energy and the unit cost of a node (altruists and peers are the same devices), respectively.

BMP can be understood as

$$\frac{Throughput(F) \cdot Distance(D) \cdot Lifetime(L)}{Price(C)},$$

where

$$\begin{aligned} L &\triangleq \frac{e_0}{\max\left(P_p^{max}, P_a^{max}\right)}, \\ C &\triangleq c_0 \cdot (N_p + N_a). \end{aligned} \tag{6}$$

In words, BMP is the total amount of successfully delivered data multiplied by end-to-end distance during the network's operational time and normalized by system resources. So the higher BMP, the better performance. The unit of BMP is $\text{bit} \cdot \text{m}/\$$.

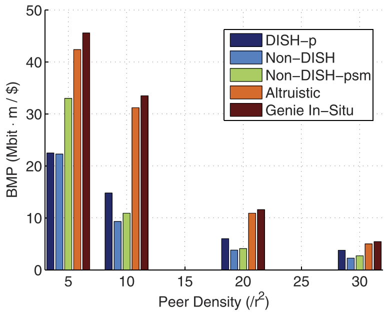
In (6), lifetime is defined as the time until any node (a peer or an altruist) runs out of energy. As peers are the nodes who actually perform the essential task of a network (transferring data), it also makes sense to define lifetime in terms of peers only, viz., $L = e_0/P_p^{max}$. It is easy to see this alternative definition is to the favor of *Altruistic* because it only leads to a *higher* BMP for *Altruistic*. But we still use definition (6) in our study.

The applicable traffic patterns of BMP include many-to-one (tree), many-to-multiple (mesh), and many-to-many (ad hoc). As such, its intended applications broadly cover data collection, Internet access, conferencing, p2p communication, file transfer, etc. BMP can be applied to networks of various topologies and spanning any (regular or irregular) plane areas (which are accounted for by $\overrightarrow{D}$), networks with different node models and numbers of nodes (accounted for by e_0, c_0, and $N_p + N_a$),[7] and networks irrespective of single- or multichannel, single- or multihop. For networks without altruists, simply set $P_a^{max} = 0$ and $N_a = 0$. Ultimately, BMP may be generally used to evaluate cost efficiency for various protocols in various scenarios.

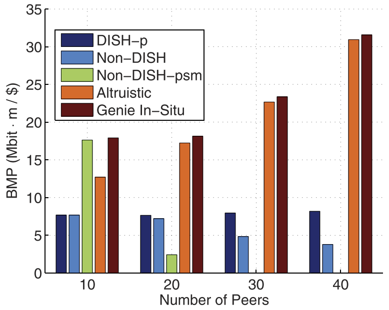
5.2 BMP Evaluation

We conduct simulation and compute BMP for the five protocols. Since all the protocols use the same devices, the value of b_0 does not affect comparison and we set $b_0 = 1$ J/\$. For *Altruistic*, we deploy altruists with density $\rho_{alt} = 1.31/r^2$ in multihop networks according to Section 4.2, and deploy a single altruist in single-hop networks which achieves full cooperation coverage. Each source node generates data at 25 kbps in multihop networks and 160 kbps in single-hop networks.

The results are shown in Fig. 5. We see that, apart from *Genie In-Situ*, *Altruistic* is the clear winner among all the protocols: its BMP is more than twice the BMP of the other protocols in most cases. Compared to *Genie In-Situ*, the BMP of *Altruistic* is only slightly lower. In fact, for a *real* in-situ energy conscious DISH protocol (without a genie), the complexity and overhead for rotating the responsibility of cooperation, as discussed in Section 3, would negate this marginal advantage of *Genie In-Situ* over *Altruistic*.



(a) Multi-hop networks.



(b) Single-hop networks.

Fig. 5. Cost efficiency evaluated via BMP. The higher, the better.

Here, we also provide an intuitive understanding of how *Altruistic* performs well. We inspect each component of BMP for *Altruistic* and *DISH-p* at the peer density of $10/r^2$ in Fig. 5a, as an example.

- *Throughput · Distance*: measured to be 3,826 $\text{Mbit}\cdot\text{m/s}$ for *DISH-p* and 3,822 $\text{Mbit}\cdot\text{m/s}$ for *Altruistic*. These two values are almost equal, which indicates that, since $\overrightarrow{D}$ is statistically the same for the two protocols, a cooperation coverage of 80 percent ($\rho_{alt} = 1.31/r^2$) suffices to achieve a cooperation gain (in terms of throughput) similar to that achieved by the opportunistic cooperation in *DISH-p*.[8]
- *Lifetime*: lifetime of *Altruistic* ($e_0/0.718Watt$) is 2.385 times that of *DISH-p* ($e_0/0.301Watt$)—peers can sleep due to the existence of altruists.
- *Price*: *Altruistic* uses 407 nodes which is 13 percent more than what *DISH-p* uses (360 nodes).

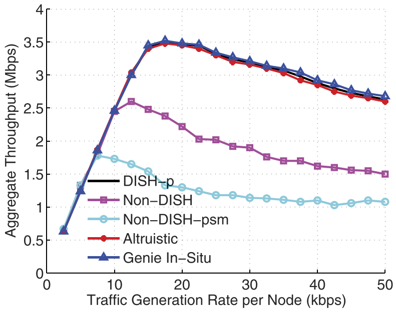
Eventually, BMP of *Altruistic* is 31.2 $\text{Mbit}\cdot\text{m}/\$$ and BMP of *DISH-p* is 14.8 $\text{Mbit}\cdot\text{m}/\$$, which translates to a significant ratio of 2.11.

Finally, we explain the different trends that the five protocols exhibit in different scenarios. In Fig. 5a (multihop networks), the BMP declines as the number of peers

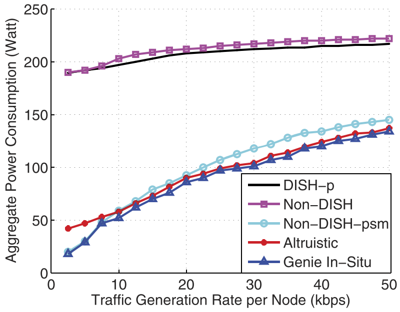
---

[7] It is also easy to extend (5) to accommodate for a heterogeneous network which contains multiple node models, by using aggregative summation.

[8] A theoretical analysis of the probability of obtaining cooperation in DISH-p can be found in [36], [37].



(a) Throughput.



(b) Power consumption.

Fig. 6. Throughput-energy trade-off in multihop networks.

increases. This is because of the drop of throughput (exponentially) as established by Gupta and Kumar [38], the drop of lifetime due to more energy consumed in packet transmission and channel contention, and the increase of cost. In single-hop networks (Fig. 5b),

1. the BMP of *Non-DISH* gradually declines due to the lack of information sharing,
2. the BMP of *Non-DISH-psm* drops remarkably due to the lack of both information sharing and gathering,
3. the BMP of *DISH-p* largely maintains, and
4. the BMP of *Altruistic* and that of *Genie In-Situ* both rise by virtue of energy conservation of the strategies as well as the throughput benefit of DISH.

In summary, the evaluation of cost efficiency demonstrates that the additional cost of altruists pays off; the performance gain from altruistic DISH more than offsets the marginal cost increase.

6 THROUGHPUT-ENERGY TRADE-OFF

This section zooms in to specifically inspect the throughput and energy performance.

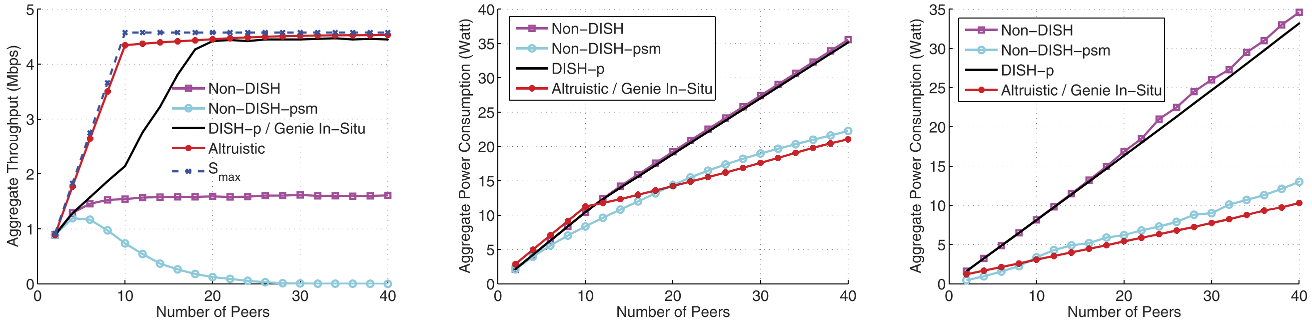
6.1 Multihop Networks

The simulation setup remains the same, and the results are shown in Fig. 6 for $\rho_{alt} = 1.31/r^2$ and $\rho_{peer} = 10/r^2$ (the results for $\rho_{peer} = 20/r^2$ are similar and omitted). Fig. 6a (throughput) clearly indicates three levels as low, medium, and high, corresponding to *Non-DISH-psm*, *Non-DISH*, and the three DISH protocols (*DISH-p*, *Genie In-Situ*, and *Altruistic*), respectively. For example, at the traffic generation rate of 25 kbps, *Non-DISH* achieves 64 percent higher throughput than *Non-DISH-psm*, and the three DISH protocols achieve 65 percent higher than *Non-DISH*. This is readily explained by the use of information gathering and/or sharing. The main message to take away from this set of results, however, is that both of the two energy-efficient strategies can *preserve* the throughput benefit of DISH.

For power consumption as shown in Fig. 6b, we see that both *Altruistic* and *Genie In-Situ* save a remarkable amount (40-80 percent) of energy consumed by *DISH-p* or *Non-DISH*. Noteworthily, *Altruistic* even outperforms *Non-DISH-psm* (though slightly) under higher traffic load, which is somehow counter-intuitive because *Non-DISH-psm* seems to be the most energy-frugal protocol where all nodes sleep whenever possible, and *Altruistic* has additional nodes who are always awake. In fact, the amount of energy saved by the altruists (through avoiding collisions and retransmissions caused by MCC problems) becomes more significant under higher traffic, where MCC problems are created more often, and outweighs the energy consumed by these few altruists.

6.2 Single-Hop Networks

Altruistic uses one altruist in single-hop networks. The simulation was conducted under high-traffic load (source nodes are always backlogged) and low traffic load (traffic generation rate is 160 kbps), respectively, and the results are summarized in Fig. 7. For throughput shown in Fig. 7a, other than observing the similar gaps to Fig. 6a, we notice that *Altruistic* outperforms *DISH-p* and even *Genie In-Situ* when the number of peers is less than 20. This is because, when peers are few and traffic load is high, peers will stay



(a) Throughput (saturated traffic). (b) Power consumption (saturated traffic). (c) Power consumption (light traffic).

Fig. 7. Throughput-energy trade-off in single-hop networks. Some curves overlap almost completely and hence are plotted as one curve for clearer visualization, as can be seen in the legend.

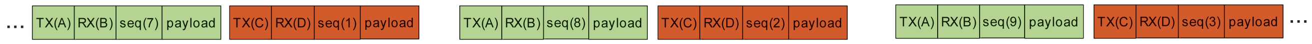


Fig. 8. Virtual collision detection. There are two interleaved fragment sequences, where *TX-RX*'s are *alternate* and *seq*'s are *inconsecutive*.

on data channels most of the time and lead to *DISH-p* and *Genie In-Situ* lacking of cooperative nodes (who must be on the control channel). However, *Altruistic* has a *dedicated* cooperative node and does not face this problem at all. Another observation is that *Altruistic* closely approaches S_{max}, a theoretical throughput upper bound

$$S_{max} = \frac{\min(m, n_f) \cdot T_{payload} \cdot W}{T_{cca}^{min} + T_{ctrl} + T_{data} + T_{sw}}, \tag{7}$$

where m is the number of data channels, n_f is the number of flows, W is the data channel bandwidth, $T_{payload}$ is the transmission time of data payload, T_{cca}^{min} is the minimum CCA duration, T_{ctrl} and T_{data} are the duration of a successful control/data channel handshake, and T_{sw} is channel switching delay. The derivation of S_{max} is given in [2]. Moreover, when there are more than 20 nodes, the throughput of *Non-DISH-psm* is very low, because each data channel has more than four fully-loaded competing nodes on average (recall that there are five data channels) and hence is almost always busy. As nodes in *Non-DISH-psm* do not gather information and always choose a channel from all channels, collision will happen for almost every channel use.

Figs. 7b and 7c present the energy performance. Under both high and low traffic loads, *Altruistic* conserves energy substantially. For example, in the low-load scenario at 40 peers, it consumes only 30 percent power of *DISH-p*. In addition, *Altruistic* again slightly outperforms *Non-DISH-psm*, which has been explained in Section 6.1.

In summary, the simulations demonstrate that altruistic DISH conserves a significant amount of energy and well maintains the throughput benefit of DISH.

7 HARDWARE IMPLEMENTATION

We have also implemented four protocols on COTS hardware (all the five except *Genie In-Situ* which requires a nonimplementable genie). To the best of our knowledge, these are the first full implementation of asynchronous multichannel MAC protocols for ad hoc networks (see review in Section 9.3).

7.1 Implementation

7.1.1 Platform Selection

We chose a microcontroller (MCU)-based platform with an ASIC radio, instead of 1) an FPGA-based platform which was more expensive and required hardware description language (HDL) in programming, or 2) a software radio whose MAC source code was 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 kits (such as HostAP [39] and MadWifi [40] as used by Dhananjay et al. [41]) have more limited MAC layer control than 802.15.4-based software (such as TinyOS [42]).

Eventually, we chose TelosB Mote [43], which is a MCU platform with an ASIC radio (CC2420 [44]) as our hardware platform and TinyOS 2.0 as our software platform. TinyOS has almost full control over the MAC layer, and its component-based architecture and C-like programming language enable rapid development. Note that such a platform choice should not be used to establish benchmarks for WiFi cards, though it suffices for a comparative study like ours.

7.1.2 Overcoming Limitations

There are two major limitations of the hardware we choose. First, the CC2420 radio supports packet size of up to only 127 bytes. We overcome this by substituting each data packet with a sequence of data fragments and treating the interfragment intervals as *payload*. In other words, let n_{frag} be the number of fragments, l_{frag} be the length of each fragment, and τ be the interval, then the transmission time of a data packet is

$$t_{data} = n_{frag}\left(\frac{l_{frag}}{w} + \tau + t_d\right) - \tau, \tag{8}$$

where $w = 250$ kbps is the channel data rate, t_d (100-200μs) is the latency that each fragment takes to be sent into the air after being assembled in memory. The second limitation is that the timing accuracy of TelosB is not reliable at the microsecond level while reliable at the millisecond level. Thus, we proportionally scale all protocol intervals up to milliseconds, e.g., SIFS is scaled to 2 ms. This way, a control channel handshake lasts for $t_{ctrl} \approx 9$ ms. Now getting back to (8), in order to keep the ratio $t_{data} : t_{ctrl}$ close to our simulation, we chose $n_{frag} = 20$, $l_{frag} = 30$ bytes (including preamble), and $\tau = 8$ ms, and consequently $t_{data} \approx 175$ ms.

7.1.3 Virtual Collision Detection

Interestingly, how we overcame the limitations described above enabled us to devise a simple yet accurate technique for *packet collision detection*. Collision detection is useful in many network algorithms (such as collision avoidance, flooding, channel selection, and data aggregation) [45], but is nontrivial because a usual PHY layer cannot distinguish packet collision from *noise corruption*. Prior techniques generally use link quality indicator (LQI) and/or received signal strength indicator (RSSI). However, they are empirical and lack in accuracy, and according to [46], [47], [48], it is still controversial whether RSSI or LQI is a better indicator for link quality.

Our technique is *virtual collision detection* which achieves the goal using *interleaved* fragment sequences. The idea is based on the fact that each data packet is transmitted as a sequence of fragments and the fragment interval (8 ms) is much larger than the fragment transmission time (<1 ms). Therefore, a good indicator of data collision is an interleaved sequence of fragments which contains fragments sent by more than one senders (recall that intervals are counted as actual payload). Fig. 8 illustrates this.

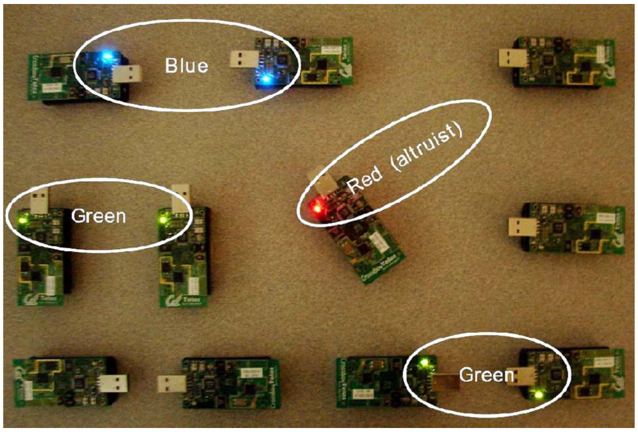


Fig. 9. A snapshot of a trial indoor experiment on *Altruistic* with 11 nodes. The four "green nodes" are two sender-receiver pairs communicating on the two different data channels. A pair of "blue nodes" are performing a control channel handshake, which creates a channel conflict problem because the only two data channels are already occupied. At this moment, the altruist ("red node") identifies this and sends a cooperative message (INV), which informs the blue nodes to back off and thereby avoid colliding with the two ongoing data transmissions.

7.2 Experiments

For visualization purposes, we use the three LEDs on each TelosB mote to indicate specific events of interest (a maximum number 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. 9 gives a snapshot in a trial indoor experiment.

In our experiments, nodes are randomly placed in a $10\ \mathrm{m} \times 10\ \mathrm{m}$ roof area, and the transmission power is set at 0 dBm which is the maximum on CC2420.[9] Nodes are configured as disjoint flows and source nodes are always backlogged. There are three channels (one control channel and two data channels) and each is with data rate 250 kbps. To compute power consumption, we trace the TX/RX/IDLE states on each node to accumulate its sojourn time for each state, and at the end of each experiment, do a weighted sum using the same power consumption rates as in the simulation setup. For protocols using power saving mode, IDLE is treated as SLEEP where peers do not overhear. Alternatively, one can put motes actually into sleep by, e.g., developing a multichannel version of B-MAC [51] or X-MAC [52] so as to measure the actual battery drainage. However, measuring the energy consumption of sensor nodes accurately is not only difficult [53], but also not necessary because 1) sensor nodes have a different energy model from WiFi nodes, 2) it requires using a real sleep-wake scheduling algorithm which will lose generality as explained in Section 3.3, and 3) this is a comparative study and the goal is not to establish absolute-value benchmarks for TelosB. Our approach of computing energy consumption was also used by Kim et al. [53].

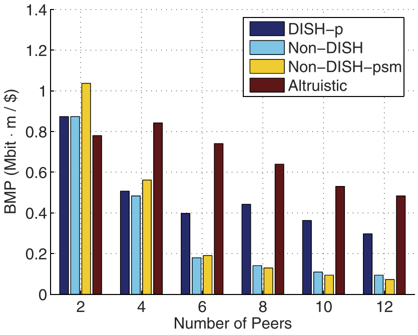
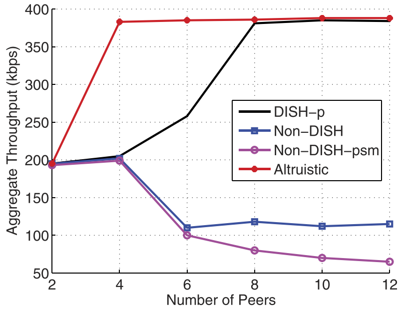
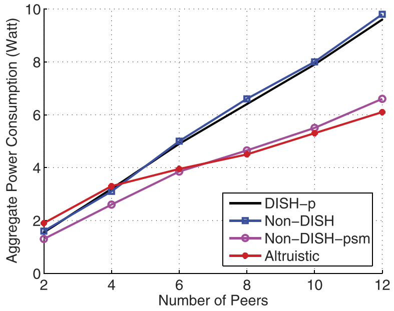


Fig. 10. Experimental results of cost efficiency.



(a) Throughput.



(b) Power consumption.

Fig. 11. Experimental results of throughput-energy trade-off.

In collecting statistics for the four protocols, every data point is by averaging over eight experiments and each experiment runs for 600 real seconds.

Fig. 10 summarizes the experimental results of cost efficiency, which confirms *Altruistic* to be the clear winner among the protocols. The only exception appears when there are only two peers where *Non-DISH-psm* performs the best. The reason is simply that DISH does not help in this contention-less case where there is only one sender-receiver pair, and that adding an altruist only increases cost and energy consumption.

Fig. 11 gives the experimental results for throughput-energy trade-off. We specifically used two data channels in order to see different trends from simulation rather than merely produce a scaled version of simulation. For throughput shown in Fig. 11a, comparing it with Fig. 7a (simulation), we see that *Non-DISH* and *Non-DISH-PSM* in Fig. 11a both have a sharp drop (by about 50 percent) when the number of peers is 6, while in Fig. 7a, the throughput of

---

[9] With this setting, all nodes are within the radio range of each other, which was also used by So et al. [15], [49], Chereddi et al. [50]. To do multihop experiments, a large number of nodes are needed to demonstrate the impact of a small ρ_{alt} on a large ρ_{peer} as shown in Sections 4.2 and 6.1.

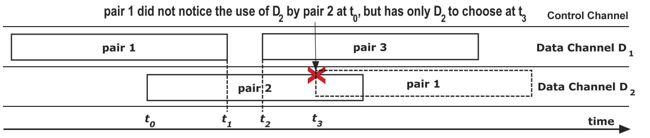


Fig. 12. The peculiarity when there are only two data channels. After pair 1 switching back to control channel at t_1, pair 3 uses D_1 at t_2, leaving pair 1 only one candidate (D_2) to choose. However, D_2 was taken by pair 2 at t_0 which is unknown to pair 1.

Non-DISH keeps increasing until finally saturates and the throughput of *Non-DISH-PSM gradually* decreases. This difference from simulation arises from the peculiarity when there are only two data channels and three fully-loaded node pairs, as illustrated in Fig. 12.

The key message conveyed by Fig. 11a is that *Altruistic* still performs the best and, particularly, better than *DISH-p* when the number of nodes is small, due to the guaranteed provision of cooperation.

Now see Fig. 11b for power consumption. *Altruistic* consumes the lowest power among all the protocols when there are a sufficient number of nodes. Another observation is, although experiments and simulations both use the same power consumption rates, the experiment statistics are consistently lower than the simulation statistics (Fig. 7b). This is because we prolonged the protocol intervals in our hardware implementation to overcome the inaccurate timing of TelosB, as described in Section 7.1, and hence the IDLE state appears more often and the TX/RX state appears less often in experiments than in simulations.

In summary, the testbed experiments confirm that *Altruistic* achieves high throughput and low-energy consumption simultaneously, and is the most cost-efficient among all the protocols under comparison. Our work also shows that multichannel MAC protocols can be indeed implemented on COTS hardware and work with a single radio and asynchronously.

8 DISCUSSION

8.1 Limitations

Altruistic DISH becomes less effective when there are only a few peers (compared to the number of channels) or traffic is light, in which case channel contention is very mild. For instance, *Altruistic* archives lower BMP than *Non-DISH-psm* in Fig. 5b at 10 nodes (five data channels) under low traffic, and similarly in Fig. 10 at two nodes. In such scenarios, in-situ energy conscious DISH could be a better choice as it is able to reduce cooperation by adapting to network dynamics.

Another limitation is that the four-way control channel handshake in the DISH protocols can incur more overhead than usual protocols. Although this can be largely offset by the cooperation gain, it is still desired to reduce the overhead. One effective way is to use *packet train* to amortize the overhead, which was also used by MMAC [10], SSCH [14], and WiFlex [54]. We have adopted this technique in [55] for cognitive radio networks.

8.2 Alternative Methods for Altruistic DISH

An alternative method for altruistic DISH is to add one more radio on a few peers and let these additional radios act as altruists. This may further enhance the cost efficiency as the cost of a radio is much lower than the cost of a node. The trade-off is the need of designing a multiradio MAC protocol which, particularly, must coordinate the use of the control channel shared by the two colocated radios. As the hardware platform (TelosB) does not support multiple radios, this alternative method merits our future study that adopts a different platform.

Another alternative to prolong network lifetime is to add an extra battery to each existing node instead of adding altruists. This is simple but would present a challenge to the size of each node, be it a laptop, a mobile, or a PDA. Also, from the perspective of scalability, the additional cost (due to extra batteries) will increase linearly when the number of peers increases, whereas in the altruistic approach, the additional cost (due to extra nodes, i.e., altruists) remains constant (as shown in Section 4).

8.3 Energy Fairness

A possible concern is that, being always awake, altruists may be overburdened and drain energy very fast. A possible solution is to apply the in-situ strategy on top of altruist DISH such that altruists rotate the role of cooperation. However, this will sacrifice simplicity which is a primary advantage of the altruist strategy. Furthermore, having altruists stay awake is not necessarily energy unfair because our evaluation in terms of BMP, which already takes energy fairness into account (via P_p^{max} and P_a^{max}, see (5)), has shown (in both simulation and testbed) that altruistic DISH performs very well in most cases. Nonetheless, fairness might be a problem under nonuniform traffic patterns and thus merit future study.

9 RELATED WORK

9.1 Energy-Efficient Multichannel MAC Protocols

There are a few proposals on this new topic. In ad hoc networks, PSM-MMAC [56] lets nodes to choose to be awake or doze based on the estimated number of active links, queue length, and channel condition. TMMAC [11] uses the 802.11 ATIM window like MMAC [10], but in addition to negotiating channels, it also negotiates time slots for nodes to sleep in.

In wireless sensor networks (WSNs), MMSN [57] was proposed to use multiple channels. However, energy saving is not one of its design goals, but is a natural and common consequence of using multiple channels (as interference is reduced). Also, when the number of channels is small, it can be seen from the paper that MMSN consumes more energy than single-channel CSMA. Chen et al. [58] propose another protocol for cluster-based WSN. The protocol is shown to be more energy efficient than MMSN by assuming 1) all cluster heads can *directly* communicate with each other and 2) there are many sink nodes and hence no single-sink bottleneck. The practicality of these assumptions can be questioned. CMAC [59], unlike MMSN and [58] which are both synchronous protocols, does not require time synchronization. However, it needs to assign every node a channel that does not overlap with any other node in 2-hop range. This means that for a network with a node density of, say, $10/r^2$, at least 126 channels are needed, which is generally not feasible.

Our work differs from existing work in the following: 1) instead of proposing a *protocol*, we propose *strategies* which can generally apply to a class of protocols (DISH-based protocols), 2) we do not require multiple radios as in PSM-MMAC and CMAC, nor time synchronization as in TMMAC, MMSN and [58], and 3) our proposal can be used in both single-hop and multihop networks, unlike PSM-MMAC which supports WLAN only.

9.2 Energy-Efficient Single-Channel MAC Protocols

In ad hoc networks, Tseng et al. [60] proposed three power-saving protocols for multihop scenarios, with time synchronization not required. These protocols differ in their power saving capability and neighbor discovery time, and can be chosen according to specific application needs.

In WSNs, there are lots of proposals and most of them can be applied to or adapted for static ad hoc networks as sensor devices are more resource-constrained. In S-MAC [61], nodes in each neighborhood negotiate a sleep-wake schedule in order to wake up at the same time. Nodes on the border of two adjacent neighborhoods will maintain two schedules to keep connectivity. In this way, network-wide synchronization is not required. T-MAC [62] improves S-MAC by shortening the awake period when there is no communication request. Each node wakes up at the start of an awake period, listens to the channel for a short time and, if there is no incoming data, returns to sleep immediately without waiting for the end of the awake period. B-MAC [51] introduces low-power listening and long preamble transmission: each data packet has a preamble slightly longer than a node's sleep period, and hence a receiver is always able to detect the transmission from a sender. Time synchronization is not required. X-MAC [52] improves B-MAC by embedding a receiver address into the preamble and strobing the preamble, so that nodes who are not the intended receiver can return to sleep earlier.

9.3 Multichannel MAC Testbed

There are a few hardware implementations of multichannel MAC protocols. Chereddi et al. [50] reported a 4-node single-hop network testbed implemented on Linux with Atheros chipset, for a hybrid multichannel MAC protocol proposed in [63]. The protocol is based on a channel abstraction module and requires two interfaces per node: one is tuned to a fixed channel for packet receiving and the other switches channels for packet transmission. McMAC [49] uses a single radio and was implemented on Telos [43] as a proof of concept. However, the implementation was a simplified version which does not measure performance metrics such as throughput, delay, or energy consumption (the only reported performance was how long it takes to synchronize sender-receiver pairs onto common channels). Y-MAC [53] is another single-radio multichannel MAC but is proposed for WSNs. It is TDMA based and specifically deals with bursty traffic in dense WSNs. It classifies every time slot as a send or a receive slot, and divides each slot into a contention window and a send/receive window. The protocol was implemented in RETOS [64] on TmoteSky motes [65], and demonstrated low-duty cycle and low-delivery latency via experiments. However, throughput was not measured.

All the above protocols require time synchronization ([63] needs loose synchronization). Recently, So et al. [15] showed that it is difficult to achieve synchronization in multichannel networks and it incurs significant overhead. They also implemented a multichannel time-synchronizing protocol, but the protocol only exchanges beacons and does not handle data packets (see Section 7.1 therein).

Most recently, there appeared two implementations of *asynchronous* multichannel MAC protocols, both for WSNs. One is TMCP [66], designed for *data collection* applications (the traffic considered was many-to-one CBR streams) and for networks with only a small number of channels. A network is partitioned into multiple subtrees and each subtree is allocated a different channel. The authors implemented the protocol on MicaZ motes and evaluated packet delivery ratio, which reflects throughput to some extent, but energy was not evaluated. Le et al. [67] built a multichannel MAC testbed also using MicaZ motes and evaluated performance in terms of the number of received messages. The energy issue was not specifically considered. Like TMCP, the protocol was designed for WSN data collection and aggregation applications. Under the random traffic pattern, which is typical in ad hoc networks, it will lead to poor performance (see Section 6 therein).

Our testbed differs from prior work in that 1) it is designed for ad hoc networks using a single radio per node and not using time synchronization, 2) it is able to evaluate typical performance metrics such as throughput and energy consumption, and 3) it is a full implementation of all the protocol functionalities.

10 CONCLUSION

Distributed information sharing can significantly boost the system throughput for multichannel MAC protocols, but it also heighten the energy consumption due to its information sharing component (which subsumes information gathering). In this paper, we propose two energy-efficient strategies and conduct a comparative study on five protocols that differ in the usage of DISH and the strategies. Both of our simulations and testbed experiments show that altruistic DISH 1) is a very simple strategy which does not involve protocol redesign nor incur additional runtime overhead, 2) substantially reduces energy consumption while maintaining (sometimes even enhancing) the throughput benefit from DISH, and also 3) notably improves cost efficiency. The other strategy, in-situ energy conscious DISH, is suitable for applications with few nodes or light traffic, or those that preclude using additional nodes.

The key reason for the success of altruistic DISH is twofold. First, using altruists as dedicated cooperative nodes provides cooperation in a *guaranteed*, as opposed to opportunistic, manner. Second, the use of altruists shifts the resource-consuming tasks (information gathering and sharing) from *all* nodes to only *a few*.

Altruistic DISH clearly separates the data plane and the control plane: peers are solely responsible for forwarding data traffic and altruists are solely responsible for control-plane cooperation, i.e., DISH.

This paper gives the first treatment on energy efficiency for cooperative multichannel MAC protocols. We believe that DISH is an approach worth exploring and that altruistic DISH is a simple yet effective strategy to implement DISH.

ACKNOWLEDGMENTS

A preliminary work was presented in [1].

REFERENCES

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

[2] T. Luo, M. Motani, and V. Srinivasan, "Cooperative Asynchronous Multichannel MAC: Design, Analysis, and Implementation," *IEEE Trans. Mobile Computing*, vol. 8, no. 3, pp. 338-352, Mar. 2009.

[3] 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 (I-SPAN)*, 2000.

[4] 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 '99)*, 1999.

[5] A. Nasipuri and J. Mondhe, "Multichannel CSMA with Signal Power-Based Channel Selection for Multihop Wireless Networks," *Proc. IEEE Vehicular Technology Conf. (VTC '00)*, 2000.

[6] 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 '01)*, 2001.

[7] A. Adya, P. Bahl, J. Padhye, and A. Wolman, "A Multi-Radio Unification Protocol for IEEE 802.11 Wireless Networks," *Proc. First Int'l Conf. Broadband Networks (Broadnets '04)*, 2004.

[8] R. Maheshwari, H. Gupta, and S.R. Das, "Multichannel MAC Protocols for Wireless Networks," *Proc. IEEE Third Ann. Comm. Soc. Sensor and Ad Hoc Comm. and Networks (SECON '06)*, 2006.

[9] J. Chen, S. Sheu, and C. Yang, "A New Multichannel Access Protocol for IEEE 802.11 Ad Hoc Wireless LANs," *Proc. IEEE 14th Personal, Indoor and Mobile Radio Comm. (PIMRC '03)*, 2003.

[10] 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.

[11] 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 '07)*, 2007.

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

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

[14] 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.

[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*, pp. 173-182, 2002.

[17] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *Proc. ACM MobiCom*, 1999.

[18] M.B. Yassein, M.O. Khaoua, L.M. Mackenzie, and S. Papanastasiou, "Improving the Performance of Probabilistic Flooding in Manets," *Proc. Int'l Workshop Wireless Ad-Hoc Networks (IWWAN '05)*, 2005.

[19] Q. Zhang and D.P. Agrawal, "Dynamic Probabilistic Broadcasting in Manets," *J. Parallel and Distributed Computing*, vol. 65, pp. 220-233, 2005.

[20] C.L. Barrett, S.J. Eidenbenz, L. Kroc, M. Marathe, and J.P. Smith, "Parametric Probabilistic Sensor Network Routing," *Proc. Second ACM Int'l Conf. Wireless Sensor Networks and Applications (WSNA '03)*, 2003.

[21] T. Roosta, M. Menzo, and S. Sastry, "Probabilistic Geographic Routing in Ad Hoc and Sensor Networks," *Proc. Int'l Workshop Wireless Ad-Hoc Networks (IWWAN '05)*, pp. 122-131, May 2005.

[22] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proc. 33rd Ann. Hawaii Int'l Conf. System Sciences (HICSS '00)*, 2000.

[23] O. Younis and S. Fahmy, "Distributed Clustering in Ad-Hoc Sensor Networks: A Hybrid, Energy-Efficient Approach," *Proc. IEEE INFOCOM*, Mar. 2004.

[24] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad Hoc Routing," *Proc. ACM MobiCom*, Sept. 2001.

[25] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Proc. ACM MobiCom*, Sept. 2001.

[26] L. Butty¢n and P. Schaffer, "PANEL: Position-Based Aggregator Node Election in Wireless Sensor Networks," *Proc. IEEE Fourth Int'l Conf. Mobile Ad-Hoc and Sensor Systems (MASS '07)*, Oct. 2007.

[27] M. Qin and R. Zimmermann, "VCA: An Energy-Efficient Voting-Based Clustering Algorithm for Sensor Networks," *J. Universal Computer Science*, vol. 12, no. 1, pp. 87-109, 2007.

[28] J. Bruck, J. Gao, and A. Jiang, "Localization and Routing in Sensor Networks by Local Angle Information," *Proc. ACM MobiHoc*, May 2005.

[29] A. Caruso, S. Chessa, S. De, and A. Urpi, "GPS Free Coordinate Assignment and Routing in Wireless Sensor Networks," *Proc. IEEE INFOCOM*, Mar. 2005.

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

[31] Cisco Systems, Inc., "Cisco Aironet 350 Series Client Adapters," http://www.cisco.com/en/US/products/hw/wireless/ps4555, 2011.

[32] L.M. Feeney and M. Nilsson, "Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment," *Proc. IEEE INFOCOM*, 2001.

[33] R.M. Karp, "Reducibility among Combinatorial Problems," *Complexity of Computer Computations.* Plenum, 1972.

[34] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms*, second ed. MIT and McGraw-Hill, 2001.

[35] N. Alon, D. Moshkovitz, and M. Safra, "Algorithmic Construction of Sets for K-Restrictions," *ACM Trans. Algorithms*, vol. 2, no. 2, pp. 153-177, Apr. 2006.

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

[37] T. Luo, V. Srinivasan, and M. Motani, "A Metric for Dish Networks: Analysis, Implications, and Applications," *IEEE Trans. Mobile Computing*, vol. 9, no. 3, pp. 376-389, Mar. 2010.

[38] P. Gupta and P.R. Kumar, "The Capacity of Wireless Networks," *IEEE Trans. Information Theory*, vol. 46, no. 22, pp. 388-404, Mar. 2000.

[39] HostAP, http://hostap.epitest.fi, 2011.

[40] MadWifi, http://madwifi.org, 2011.

[41] A. Dhananjay, H. Zhang, J. Li, and L. Subramanian, "Practical, Distributed Channel Assignment and Routing in Dual-Radio Mesh Networks," *Proc. ACM Special Interest Group on Data Communications (SIGCOMM '09)*, pp. 99-110, 2009.

[42] TinyOS Community Forum, http://www.tinyos.net, 2011.

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

[44] Chipcon Corporation, "CC2420 2.4 GHz Zigbee/802.15.4 RF Transceiver," http://www.chipcon.com, 2011.

[45] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Culler, "Exploiting the Capture Effect for Collision Detection and Recovery," *Proc. IEEE Second Workshop Embedded Networked Sensors (EmNets '05)*, 2005.

[46] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," *Proc. First ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '03)*, 2003.

[47] 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 '04),* 2004.

[48] 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.

[49] H.-S. W. So, J. Walrand, and J. Mo, "McMAC: A Parallel Rendezvous Multi-Channel MAC Protocol," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC '07),* 2007.

[50] C. Chereddi, P. Kyasanur, and N.H. Vaidya, "Design and Implementation of a Multi-Channel Multi-Interface Network," *Proc. ACM Second Int'l Workshop Multi-Hop Ad Hoc Networks: From Theory to Reality (REALMAN '06),* 2006.

[51] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proc. ACM Second Int'l Conf. Embedded Networked Sensor Systems (SenSys '04),* pp. 95-107, 2004.

[52] M. Buettner, G.V. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks," *Proc. ACM Fourth Int'l Conf. Embedded Networked Sensor Systems (SenSys '06),* pp. 307-320, 2006.

[53] Y. Kim, H. Shin, and H. Cha, "Y-MAC: An Energy-Efficient Multi-Channel MAC Protocol for Dense Wireless Sensor Networks," *Proc. ACM/IEEE Int'l Conf. Information Processing in Sensor Networks (IPSN '08),* pp. 53-63, 2008.

[54] J.W. Lee, J. Mo, T.M. Trung, J. Walrand, and H.-S. W. So, "Wiflex: Multi-Channel Cooperative Protocols for Heterogeneous Wireless Devices," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC '08),* 2008.

[55] T. Luo and M. Motani, "Cognitive DISH: Virtual Spectrum Sensing Meets Cooperation," *Proc. IEEE Sixth Ann. Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON '09),* June 2009.

[56] J. Wang, Y. Fang, and D. Wu, "A Power-Saving Multi-Radio Multi-Channel MAC Protocol for Wireless Local Area Networks," *Proc. IEEE INFOCOM,* 2006.

[57] 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.

[58] X. Chen, P. Han, Q. He, S. Tu, and Z. Chen, "A Multi-Channel MAC Protocol For Wireless Sensor Networks," *Proc. IEEE Sixth Int'l Conf. Computer and Information Technology (CIT '06),* 2006.

[59] 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.

[60] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks," *Proc. IEEE INFOCOM,* 2002.

[61] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 12, no. 3, pp. 493-506, June 2004.

[62] T. van Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. ACM First Int'l Conf. Embedded Networked Sensor Systems (SenSys '03),* 2003.

[63] 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.

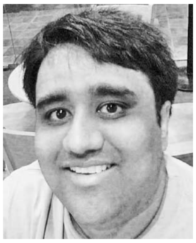
[64] H. Cha, S. Choi, I. Jung, H. Kim, H. Shin, J. Yoo, and C. Yoon, "RETOS: Resilient Expandable, and Threaded Operating System for Wireless Sensor Networks," *Proc. ACM/IEEE Sixth Int'l Symp. Information Processing in Sensor Networks (IPSN '07),* 2007.

[65] Tmote Sky, http://www.sentilla.com/pdf/eol/tmote-sky-datasheet.pdf, 2011.

[66] Y. Wu, J.A. Stankovic, T. He, and S. Lin, "Realistic and Efficient Multi-Channel Communications in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* pp. 1867-1875, 2008.

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

Tie Luo received the BE degree in computer engineering from the Beijing University of Posts and Telecommunications, China, and the PhD degree in electrical and computer engineering from the National University of Singapore. He is currently a research fellow in the Department of Electrical and Computer Engineering, National University of Singapore. His research interests are wireless networks and network economics.



Mehul Motani received the BE degree from Cooper Union, New York, the MS degree from Syracuse University, New York, and the PhD degree from Cornell University, Ithaca, New York, all in electrical and computer engineering. He is currently an associate professor in the Electrical and Computer Engineering Department at the National University of Singapore. Previously, he was a research scientist at the Institute for Infocomm Research in Singapore for three years and a systems engineer at Lockheed Martin in Syracuse, New York, for more than four years. Recently, he has been working on research problems which sit at the boundary of information theory, communications, and networking, including the design and analysis of wireless ad hoc and sensor network systems. He was awarded the Intel Foundation Fellowship for work related to his PhD degree in 2000. He has served on the organizing and technical program committees of ISIT, 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, Bengaluru, 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 are in the area of wireless networks. He is a member of the IEEE.