www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 9 Sept., 2013 Page No. 2688-2697

Energy-Efficient Strategies for Cooperative Multichannel MAC Protocols

P.V.Naga Lakshmi B.Sasmitha, C.Rama Krishna,
M.tech (CSE) Student,
PRRCET,Medak(dist), Andhra Pradesh India 502 300 ,

nagmtechloyola@gmail.com

Assistant Proffessor,
PRRCET Medak(dist) Andhra Pradesh India 502 300

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.

Ç

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.

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. Next, relevant issues are discussed in Section 7 and related work is reviewed in Section 8. Finally, Section 9 concludes this paper.

2 UNDERSTANDING DISH

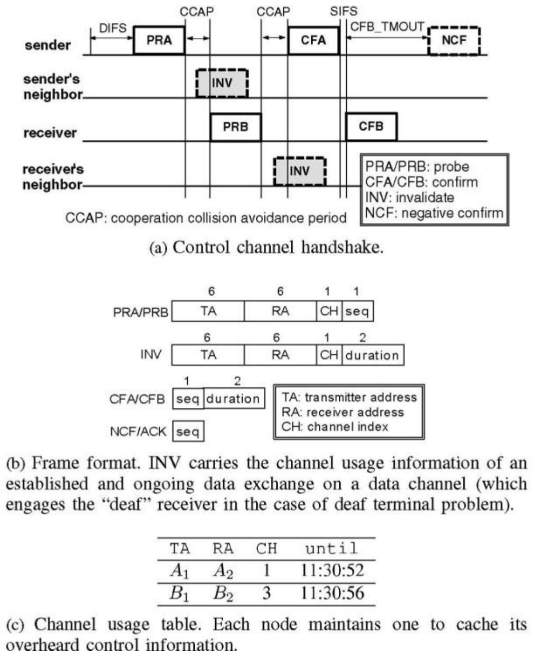
Control information is crucial to communications but can be missing due to various reasons such as shadowing and noise. The dominant reason, however, in a multichannel environment, is that nodes fail to tune radios to certain channels in time, or that a radio can only listen to one channel at a time. 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.

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.

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 sig-nificant. Even using a packet train would not suit because sensing traffic is usually periodic and not bursty.

[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.

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.

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]).

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. 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 consider-able 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.

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.

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.

DISH-p: the protocol described in Section 2.1.

Non-DISH: a (traditional) noncooperative protocol, derived from DISH-p by removing the cooperative element, i.e., neighbors do not share control information with senders or receivers.

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).

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.

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.

Issues to Investigate

There are three relevant issues that need to be addressed:

Node deployment (addressed in Section 4): how to deploy altruists for Altruistic DISH.

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.

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

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.

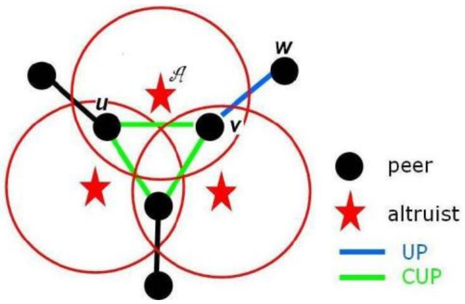


Fig. 2. Illustration of UP and CUP. Node pair ðu; vÞ is a CUP (covered by altruist A) while ðv; wÞ is an UP. Each circle denotes the transmission range of an altruist.

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.

4 OPTIMAL NODE DEPLOYMENT

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

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.

4.2 Deployment Types

4.2.1 Random Deployment

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

4.2.2 Arbitrary Deployment

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

5 COST EFFICIENCY

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

Bit-Meter-Price Ratio

BMP is a network performance metric defined as

F D b
0
BMP 4 ! _ ! _ ; 5

¼ ðNp þ NaÞ _ max Ppmax; Pamax ð Þ

where ! is a vector of all the flows' throughput, ! is a vector of all the flows' source-to-destination euclidean 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, b_0 ¼ 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

Throughtput(F).Distance(D).Lifetime(L)

Price©

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

6 THROUGHPUT-ENERGY TRADE-OFF

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

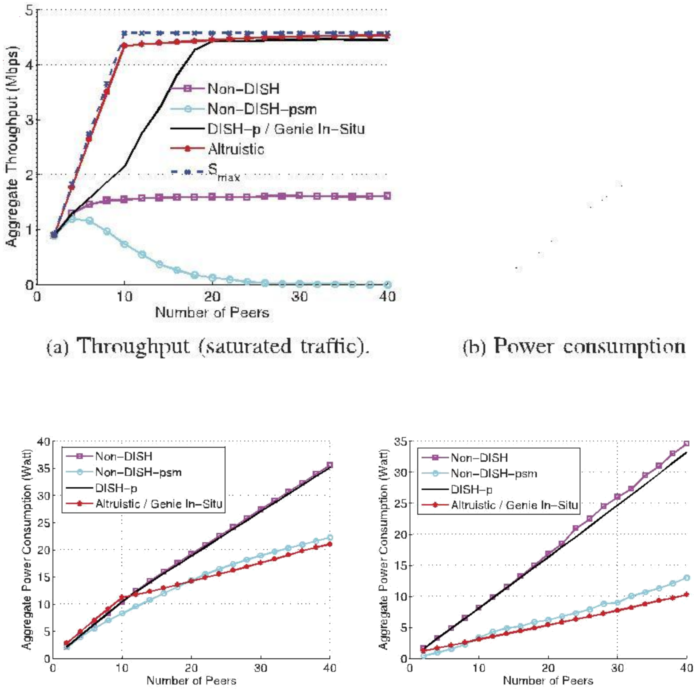
Multihop Networks

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 genera-tion 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 addi-tional 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 pro-blems are created more often, and outweighs the energy consumed by these few altruists.

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 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



(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.

minðm; n_f Þ _ $T_{payload}$ _ W

Smax ¼ T_{cca}^{min} þ T_{ctrl} þ T_{data} þ T_{sw} ;

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 DISCUSSION

7.1 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.

7.2 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).

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. None-theless, fairness might be a problem under nonuniform traffic patterns and thus merit future study.

8 RELATED WORK

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 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

T. Luo, M. Motani, and V. Srinivasan, “Altruistic Cooperation for Energy-Efficient Multi-Channel MAC Protocols,” Proc. ACM MobiCom, 2007.

T. Luo, M. Motani, and V. Srinivasan, “Cooperative Asynchro-nous Multichannel MAC: Design, Analysis, and Implementa-tion,” IEEE Trans. Mobile Computing, vol. 8, no. 3, pp. 338-352, Mar. 2009.

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.

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.

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.

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.

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.

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.

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.

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.

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.

A. Tzamaloukas and J. Garcia-Luna-Aceves, “Channel-Hopping Multiple Access,” Proc. IEEE Int’l Conf. Comm. (ICC ’00), 2000.

A. Tzamaloukas and J. Garcia-Luna-Aceves, “Channel-Hopping Multiple Access with Packet Trains for Ad Hoc Networks,” Proc. IEEE Device Multimedia Comm., 2000.

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.

H.-S. W. So, G. Nguyen, and J. Walrand, “Practical Synchroniza-tion Techniques for Multi-Channel MAC,” Proc. ACM MobiCom, 2006