RESEARCH ARTICLE OPEN ACCESS

Higher Throughput and Energy Efficiency Multi-Channel MAC Protocol For Ad Hoc Network.

Ms. Suhasini E. R, Darshan A. M
Assistant professor Department of ECE, the Oxford College of Engineering Bommanahalli
M. tech 2nd year Department of ECE, the Oxford College of Engineering Bommanahalli

ABSTRACT
Traditional single-channel MAC protocols for wireless adhoc and sensor networks favor energy efficiency over throughput. More recent multi-channel MAC protocols display higher throughput but less energy efficiency. In this paper we propose EMAC, a negotiator-based multi-channel MAC protocol in which specially designated nodes maintain the sleeping and communication schedules of nodes. Negotiators facilitate the assignation of channels and coordination of communications windows, thus allowing individual nodes to sleep and save energy. Simulation results show that EMAC, at high network loads consumes less energy while providing more throughput than comparable state- of-art multi-channel MAC protocols for ad hoc networks.

I. INTRODUCTION

Traditional MAC protocols for wireless ad hoc and sensor networks restrict themselves to a single frequency using variety of techniques to optimize throughput. Typically designed to work well under low network load, they also attempt to maximize energy efficiency focused attention on multi-channel MAC protocols designed to work efficiently under higher network loads. Although higher throughput has been achieved, this improvement has come at the cost of decreased energy performance, when compared with single channel MAC protocols. An important factor preventing multi-channel MAC protocols from achieving high energy savings is the synchronization required for communication over multiple channels. Dense wireless networks exacerbate the issue by complicating the schedule of channels on which a node can communicate with its neighbors. Existing research has proposed several ways to maintain schedule information. Some protocols assign predictable static schedules and channels and propagate this information to all nodes in the network. However, static assignments underutilize bandwidth and prevent the network from achieving high aggregate throughput. Other protocols use a common contention-based control period" where nodes communicate pairwise on a single channel to coordinate their schedules. This common negotiation period wastes energy when traffic is light, as all nodes must be awake during this period. To meet the dual, and often opposing, goals of improved throughput and reduced energy consumption, EMAC does not adopt direct pairwise negotiation. Instead, EMAC designates a set of negotiators who maintain the schedules of all nodes in the network and assist with channel negotiation. In EMAC, when a sender has packets for a receiver, it requests assistance from a negotiator. Because the negotiator is aware of all communications schedules in its neighborhood, it can assign a time and a channel for the sender to communicate with the receiver. This minimizes the time a receiver stays awake waiting for potential transmissions, thus resulting in higher energy efficiency. It also reduces non-negotiator storage requirements because schedule information is only exchanged between nodes and their negotiators, and not among all neighbors. The main contributions of our work are:

1. A communications negotiator that is responsible for synchronizing senders and receivers on a channel and a time when they can communicate.
2. A multi-channel MAC protocol applicable to both wireless ad hoc and sensor networks with minor modifications.
3. Extensive simulations evaluating the proposed multichannel MAC protocol and showing that it output performs state-of-art multi-channel MAC protocols by achieving significant energy savings and improved throughput.

II. RELATED WORK

A significant number of multi-channel MAC protocols have been proposed for wireless ad hoc networks [5] [8] [10]. Some require special hardware [15], such as the use of multiple radio transceivers to listen to multiple frequencies at the same time. Others, e.g., TMMAC [19] and MMAC [11], are based on the IEEE 802.11 Distributed Coordination Function (DCF) and use control messages for channel negotiations. SSCH [2] uses pseudo random number generators to help with the allocations of frequencies and channel switching. TDMA-based MAC protocols in ad hoc networks have been primarily designed to provide collision-free access to a single channel [3].

In wireless sensor networks, the energy efficiency of MAC protocols has received significant research focus. Single channel protocols [12] [11] [4] [6] use low power listening and sleep schedules to save energy. High traffic loads, due to either application semantics or the sink-oriented topology common in wireless sensor networks, poses additional challenges. To address this issue [12] [1] use a hybrid CSMA/TDMA approach. While single channel MAC protocols have better energy consumption, research has demonstrated that multi-channel protocols can achieve higher throughput. Several multi-channel MAC protocols for wireless sensor networks have been recently proposed. One proposed direction is to have static channel and slot assignments. In [12], a node is assigned a fixed frequency for reception, potentially limiting channel utilization, while [8] proposes that the entire schedule be static. A multi-channel MAC protocol specifically designed for dense sensor networks is proposed in [9]. Al-though implemented on real hardware, it is not evaluated in a highly dense network. EMAC is different from these two protocols because it does not require nodes with special capabilities and be-cause the sleep schedules of non-negotiator nodes allow for aggressive energy savings.

III. EMAC DESIGN

The main idea of our multi-channel MAC protocol is the use of special nodes, called negotiators,that schedule traffic for neighbor nodes. This is fueled by a desire to minimize the principle sources of energy consumption in a wireless network: overhearing, communication, idling and collisions. EMAC trades of increased energy consumption by the negotiator node for energy savings on all non-negotiators. The energy savings are derived from reduced overhearing and collisions, and reduced duty-cycles allowed by longer sleep

NMAC Negotiator Algorithm:

```
1:  Broadcast HELLO messages. Build Neighbor
    Table (Nbr Tbl) based on HELLO messages
    heard.
2:  Set MyTimer according to Equation 1
3:  while (MyTimer not expired) do
4:  if (Received Nbr Tbl from New Negotiator) then
5:  Adjust MyTimer according to Equation 1
6:  Set my neighbors (in received Nbr Tbl) as
    covered
7:  end if
8:  end while
9:  if (I have uncovered neighbors) then
10: Declare myself as negotiator
11: Broadcast my Nbr Tbl
12: end if
```

3.1 Negotiator Election

The negotiator election algorithm, presented in Algorithm , is executed by all nodes during network initializations and has two phases. In the first phase, represented by line 1 in Algorithm 1, each node builds a neighbor table. In the second phase, lines 2-12, each node sets a timer, at the end of which, it will declare itself as a negotiator. The timer value is inverse to the number of neighbors uncovered by negotiators and to its residual energy. It is formally given by.

$$T = ((Nmax - Nunc) * tc + rand(t)) * (1 - E / Emax)) \quad (1)$$

where *Nmax* is a global estimate for the maximum number of neighbors a node can have, *Nunc* is the number of neighbors for which the node does not have a negotiator, *tc* is a global time constant, *rand*(*t*) is a random number between
0 and *tc*, and *E* and *Emax* are the nodes's current and initial energy levels, respectively.

When the timer expires, a node announces itself as a negotiator and broadcasts its neighbor table. The neighbors that receive this announcement, update their negotiator in-formation, recalculate the number of neighbors uncovered (i.e., nodes that are not neighbors of the negotiator) and adjust their timers accordingly.

Because the negotiator needs to be available on the de-fault channel at all times, a design decision we made was that a negotiator does not route traffic (routing traffc would entail switching channels). Consequently, the network connectivity is affected. To better understand the impact of our design decision, in the remaining part of this section we provide the analysis for the effect negotiators have on the degree of network connectivity.

Assuming that *n* nodes are uniformly distributed within one radio range and that *N* negotiators are being elected, the total number of connected links is given by:

$$(n-N)(n-N-1)/2$$

the number of lost links L_t (from a total of $n(n - 1)=2$) due to the negotiator election is:

$$L_t = (2n-1-N)N/2 \quad (2)$$

If $pn = N/n$ is the percentage of negotiators in the net-work, the total number of lost links becomes:

$$L_t=np_n(2n-1-np_n) / 2 \quad (3)$$

Consequently, the percentage of lost links in the network is:

$$PL_t= p_n(2n-1-np_n) / (n-1) \quad (4)$$
$$=2p_n - p_n^2 \text{ since } n>>1 \quad (5)$$

This result indicates that a small decrease in the percentage of negotiators (*pn*) has a significant impact on the number of links that can be used for routing traffic. Consequently, one goal of our negotiator election algorithm is to produce as small a set of negotiators, as possible. It is important to mention that the negotiator election algorithm runs periodically to enable negotiator rotation, and better distribution of energy consumption.

This result indicates that a small decrease in the percentage of negotiators (*pn*) has a significant impact on the number of links that can be used for routing traffic. Consequently, one goal of our negotiator election algorithm is to produce as small a set of negotiators as possible.

3.2 Frame Architecture

EMAC is a TDMA-based multi-channel MAC protocol, thus we assume the presence of a time synchronization scheme. The frame structure, as well as the messages ex-changed, is depicted in Figure 2. Time is divided into Bea-con Intervals which are further divided into time slots. A set of three time slots forms a Group. Intuitively, the grouping of three slots is due to the distinct types of messages that need to be exchanged: the request from a sender to the negotiator, the request from the negotiator to the receiver (done in the second slot of a group) and the acknowledgement from the negotiator to the sender (done in the third slot of a group). All of these messages are sent over the default channel in a contention-based manner. Each node keeps a schedule of its projected activity for each time slot, sleeping or communicating, and the channel to use. A negotiator maintains a copy the schedules for every node it covers.

EMAC supports both broadcast and unicast. Broad-cast can only be sent in the rst slot of a Beacon Interval, a time when all nodes are on the default channel. For unicast, communication is only possible after negotiation. An explanation of the negotiation process in EMAC follows.

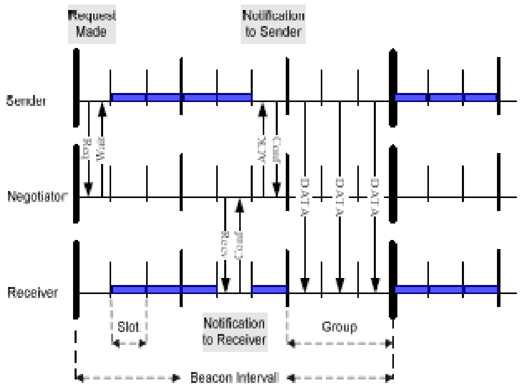

Figure1: Time slot and channel negotiation in EMAC. Protocol signaling and data communication are indicated by vertical arrows. Horizontal blue line indicates when nodes are in sleep

3.3 Negotiation for Unicast

When a sender has unicast packets, a three step process is followed.

First, the sender sends a request (*Request Made* in Figure 2) to the negotiator in charge of the link between it and the receiver. This request can only be made on the default channel during the rst slot of a group. The request from the sender contains the number of packets and the destination. The negotiator examines the schedule of the receiver and replies to the sender with an acknowledgement (*Wait* in Figure 1), containing the time slot when the sender should expect a confirmation/decision (*Notification to Sender*" in Figure 1). After the transmission of the request packet, the sender starts a timer to wait for the acknowledgement. If the timer expires before receiving a reply, the sender reschedules the request packet.

Second, the negotiator examines the schedules of the sender, receiver, and nodes within one hop of either and assigns time slots and channels for the potential communication. The negotiator nodes available slots of the receiver and chooses a random channel from all available channels .This decision (slots labeled as *Recv*" in Figure 2 and channel) is sent to the receiver as a notification packet (*Notification to Receiver*" in Figure 1). The *Recv* request is sent by the negotiator during a time slot when the receiver is awake using the receiver's frequency. Upon receiving this notification, the receiver checks its own schedule to see if there are conflicts between the requested tuple (slots and channel) and its own schedule. If there are no conflicts, the receiver sends a confirmation packet to the negotiator (*Conf* paired with *Recv* in Figure 1). Other negotiators in the neighborhood overhear this confirmation and use it to update their schedules for the receiver.

Third, the negotiator notifies the sender of this decision (in the slot already scheduled with the sender in step 1) as depicted in *Notification to Sender* in Figure 1. The sender updates its schedule according to this decision and sends a confirmation to the negotiator as well. All other negotiators in the neighborhood overhear this confirmation and update their schedules for the sender.

3.4 Sleeping Schedule

One key issue that EMAC addresses is energy consumption. This is accomplished by having non-negotiator nodes operate in a duty-cycle that varies depending on the traffic in the network. Nodes are only awake in four different cases: a) a node is awake during the first time slot of a Beacon Interval. This accommodates broadcast communication; b) a node is awake during the first slot of a group (including the first slot of a Beacon Interval) if it has a request to send to a negotiator, or it has data to send/receive; c) a node is awake during the third slot of a group if it expects an acknowledgement from a negotiator for a previous request or if it sends/receives data; d) a node is awake during the second slot of a group if it has data to send/receive or, or if it expects notification from a negotiator. A receiver infers its potential traffic load based on recent historical data. If the load is heavy, it stays awake during every second slot of a group to wait for notification; if the load is light, it wakes up occasionally (based on the degree of the load) during the second slot of a group and the negotiator has the knowledge of its schedule.

As an example, Figure 1 depicts with horizontal bars the time slots when different nodes are asleep. As shown, negotiators do not duty-cycle. In the example shown, the sender informs the negotiator during slot 1 of its desire to send three packets to the receiver. The negotiator tells the sender to be awake at slot 6 to possibly receive an acknowledgement. In this example, the sender does not expect packets from other nodes, so it can safely sleep for the duration between slots 2-5. Since the negotiation is successful (i.e., it is ACKed in slot 6), the sender is awake during slots 7-9 to send the data packets. As shown, the receiver is awake during slot 1 for a possible broadcast. The receiver does not have packets to send so it can sleep during slots 2-4. Based on the traffic during the previous beacon interval, it expects a *Recv* notification during slot 5 and awakes at that time. If there is no traffic in the network, ordinary nodes are only awake two slots per beacon interval (out of 48 slots) which means the minimum duty cycle of ordinary nodes is ~4%.

IV. PERFORMANCE EVALUATION

In this section, we compare the performance of our pro-posed MAC protocols, EMAC with existing state of art multi-channel protocols in wireless ad hoc and sensor networks: NAMAC [19], LNMAC [14] and MMSN [20], respectively. Our performance evaluation results are obtained through simulations in NS2.The radio range was fixed at 250m and we considered radio transceivers with 4 and 6 channels. We use Geographic Forwarding as the routing protocol and the two-ray radio propagation model. The traffic is CBR with a packet size of 512B. For the single-hop and multi-hop scenarios we generated 15 and 20 pair wise random data flows, respectively. The network load was varied through the packet arrival rate in each flow, which ranged from 1-1000 packets/second.

4.1Aggregate Throughput

Simulation results for the single-hop network are presented in Figure 2. For single-hop communication, the throughput of the three protocols at low network loads is similar as the traffic is still within each protocol's limit. When traffic is high (packet rates greater than 100packets/second), EMAC outperforms NAMAC and LNAMAC protocols because it can assign more packets per negotiation. Also, EMAC does not have a fixed negotiation window that takes a large portion of the total time. Once the negotiation is done, all time slots can be used for communication. We also see that as the number of channels increases, the throughput for the three protocols increases as well.

In the simulation results for the multi-hop network (Figure 5), the throughput of the three protocols at low network loads is similar. In high network traffic, the throughput of EMAC is higher than NAMAC and higher than LNMAC.

4.2Energy Consumption

Remarkably, in the single hop communication scenario, as shown in Figure 3, EMAC is the most energy efficient protocol at low and high network loads. EMAC works especially well at low network loads where it consumes less energy than NAMAC and. At high loads, when nodes are awake most of the time, the energy consumption of EMAC is 10% less than NAMAC and 40% less than LNMAC.

In a multi-hop network scenario, EMAC still consumes the least energy. NAMAC consumes more energy because the nodes in the network that do not participate in routing still need to stay awake during the contention-based interval for potential channel/ slot negotiations.

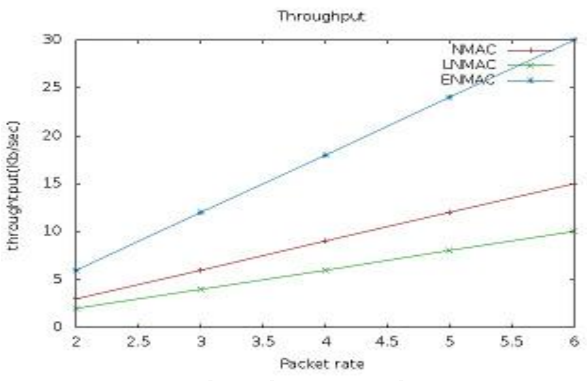

Figure2:Throughput v/s packet rate

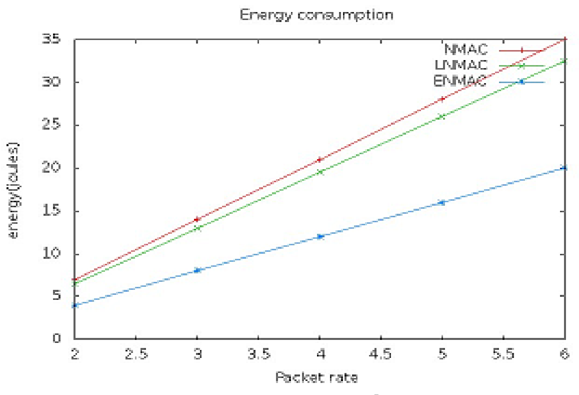

Figure3:Energy v/s packet rate

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented EMAC – an energy efficiency multi-channel MAC protocol, in which specially designated nodes maintain sleeping and communication schedules of nodes. Simulation results show that EMAC, at high network loads, consumes less energy while providing 25% more throughput than state of art multi-channel MAC protocols for ad hoc networks.. We leave for future work the implementation of EMAC on real mote hardware. We plan to add redundancy through additional backup negotiators which can aid in case of negotiator failures. An optimization of our scheme can address the scenario of data streams present in the network. In this scenario, nodes do not need to negotiate frequently.

REFERENCES

[1] G.-S. Ahn, E. Miluzzo, and A. T. Campbell. *A funneling-mac for high performance data collection in sensor networks*. In *Proceedings of ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[2] P. Bahl, R. Chandra, and J. Dunagan. SSCH: *slotted seeded channel hopping for capacity improvement*. In *Proceedings of Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[3] L. Bao and J. J. Garcia-Luna-Aceves. *A new approach to channel access scheduling for ad hoc networks*. In *Proceedings of Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[4] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: *A short preamble mac protocol for duty-cycled wireless sensor networks.* In *Proceedings of ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[5] X. Chen, P. Han, Q. He, S. Tu, and Z. Chen. *A multi-channel mac protocol for wireless sensor networks*. In *Proceedings of CIT*, 2006.

[6] A. El-Hoiydi and J.-D. Decotignie. WiseMAC: *An ultra low power mac protocol for the downlink of infrastructure wireless sensor networks*. In *Proceedings of ISCC*, 2004.

[7] M.-S. C. Ergen and F.-P. Varaiya. PEDAMACS: *Power efficient and delay aware medium access protocol for sensor networks. IEEE Transactions on Mobile Computing*, 5(7), 2006.

[8] M. Jovanovic and G. Djordjevic. TFMAC: *Multi-channel mac protocol for wireless sensor networks*. In *Proceedings of TELSIKS*, 2007.

[9] Y. Kim, H. Shin, and H. Cha. Y-MAC: *An energy-efficient multi-channel mac protocol for dense wireless sensor networks.* In *Proceedings of IPSN*, 2008.

[10] P. Kyasanur and N. H. Vaidya. Capacity of multi-channel wireless networks*: impact of number of channels and interfaces*. In *Proceedings of Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2005.

[11] *Towards Higher Throughput and Energy Efficiency in Dense Wireless Ad Hoc and Sensor Wei Zhou*, Radu Stoleru. Department of Computer Science and Engineering Texas A&M University. *SAC'10* March 22-26, 2010.

[12] J. So and N. H. Vaidya. Multi-channel mac for ad hoc networks: *handling multi-channel hidden terminals using a single transceiver*. In *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2004