Exploiting Routing Redundancy using MAC layer Anycast to Improve Delay in WSN

Farhana Ashraf, Robin H. Kravets and Nitin H. Vaidya
University of Illinois at Urbana-Champaign
Email: {fashraf2, rhk, nhv}@illinois.edu

***Abstract*— Source-destination pairs in wireless sensor networks often have multiple shortest hop paths because the nodes are densely deployed. These multiple paths provide a great opportunity to reduce delay as well as energy in an asynchronous duty-cycled network. In this paper, we exploit the redundancy available at the route layer using MAC-layer anycasting and reduce the delay incurred at each hop as the sender waits for its next hop node to be awake. By applying anycast to existing X-MAC and NPM protocols, we show that anycast can be incorporated into duty-cycled MAC protocols by using small modifications. Our evaluations in *ns-2* show that the modified X-MAC and NPM protocols can achieve delay improvements of 30% and 12% respectively by exploiting the route level redundancy using anycast.**

I. Introduction

Energy-efficiency and end-to-end delay are two major challenges for applications in wireless sensor networks. Due to the limited battery power available, sensor nodes need to save as much energy as possible to extend network lifetime. However, the solutions adopted by energy-efficient protocols frequently lead to increased delay. Essentially, energy-efficient MAC protocols for sensor nodes adopt periodic wakeup-sleep schedules to save the energy wasted during idle listening. Since nodes are no longer awake all of the time, a sender must wait for its receiver (i.e., the next hop node) to be awake before the sender can send its data packet. As the nodes sleep more and so save more energy, the sender must wait longer for the receiver to be awake, resulting in increased per hop delay. As the traffic generation rate increases and the network gets saturated, the negative effect on the delay deteriorates even more.

Several approaches have been proposed to provide support for delay-sensitive applications in duty-cycled wireless sensor networks [1]–[4]. The basic idea of these approaches is to fix the wakeup times of the next hop nodes along a path such that a sender does not need to wait long for its next hop node to become awake. However, in a realistic network with the unavoidable clock drift expected with cheap sensor motes [5], it is difficult to maintain such wakeup schedules. In the end, the protocols either fail to reduce delay or expend excessive energy synchronizing the wakeup schedules of the next hop nodes. Moreover, these protocols essentially require the nodes to have pre-knowledge about the paths between all possible sender-receiver pairs. When one of the paths change, the protocols require all nodes along that path to reset their wakeup schedules. Thus, these approaches are not suitable for large networks with diverse traffic patterns.

In our research, we explore a completely different approach to reducing delay in a duty-cycled network. Instead of setting up and synchronizing the wakeup times of the duty-cycled nodes, we investigate whether we can exploit route level redundancy to improve delay in a randomly synchronized duty-cycled network. The relatively dense deployment of cheap sensors in a wireless sensor network (WSN) or an active RFID network offers multiple shortest hop paths between the same source-destination pair. Essentially, the sender can choose any of these shortest hop paths to send its data packets. From a MAC layer perspective, this redundancy at the routing layer translates into multiple next hop options available for the sender, while still utilizing shortest hop paths. In a duty-cycled network where the wakeup times of the nodes are randomly synchronized, different next hop nodes wake up at different times. Thus, with multiple next hop options, the sender now has the opportunity to find the next hop node that wakes up the soonest. By exploiting the available redundancy, the delay incurred at each hop waiting for the next hop node (i.e., the receiver) to be awake can be reduced.

To exploit the redundancy available at the route layer and the diversity in the wakeup times of the different nodes in the network, we augment existing duty-cycled MAC protocols with MAC layer anycast. This novel combination enables significant improvement in delay for most asynchronous duty-cycled sensor networks. When multiple next hop nodes are available, we enable the sender to select the best next hop in terms of delay. This enhancement allows the sender to send its data to the next hop node that wakes up the soonest, incurring less delay at each hop, while still conserving energy. The key benefit of our approach, beyond decreasing delay, comes from the lack of dependence on the mechanisms that different MAC protocols use to determine whether the next hop nodes are awake or not. Essentially, our solution can be applied to many duty-cycling MAC protocols with asynchronous wakeup schedules. In this paper, we demonstrate how these techniques can be applied to two existing duty-cycling MAC protocols, X-MAC [6] and NPM [7], using simple modifications to the protocols. Our evaluations show that the addition of MAC layer anycast enables a 30% improvement in delay by exploiting the redundancy at the routing layer. In addition to improved delay, the protocols augmented with the MAC layer anycast

[0]This work is supported in part by NSF CNS 05-19817

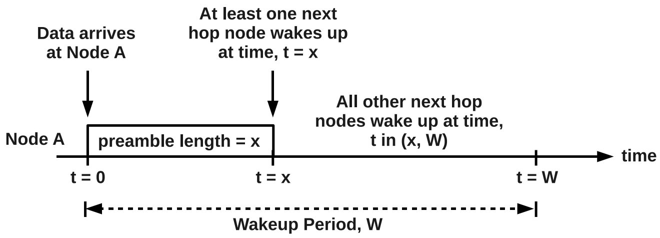

Fig. 1. X-MAC with anycast, when preamble length = x

also showed reduced energy consumption, with up to a 30% savings for X-MAC and up to a 12% savings for NPM.

The remainder of the paper is organized as follows. Section II presents the opportunities available in duty-cycled MAC layer protocols to improve delay by exploiting redundancy. Section III describes how MAC-layer anycast can be incorporated into some existing duty-cycling MAC protocols to exploit the available redundancy. In Section IV, we evaluate different protocols with anycast and compare the performance with the original version of the protocols. Finally, Section V concludes the paper and presents future research directions.

II. Leveraging Redundancy to Reduce Delay in Duty-Cycled Networks

Delay in a duty-cycled network is mostly incurred while the sender waits for its next hop node (i.e., the receiver) to be awake. In a network where the wakeup times of the different nodes are randomly synchronized, providing multiple options for a next hop offers the sender great opportunities to find a next hop node that wakes up the soonest and thus reduce the delay incurred at each hop.

Figure 1 illustrates the delay incurred in a duty-cycled network when the sender has n next hop options. If each node wakes up periodically maintaining a wakeup period W and the wakeup times of the different nodes are uniformly distributed, the sender finds the next hop node that wakes up the soonest after a time period of $\frac{W}{n+1}$ on an average (from order statistics). As a result, the sender can achieve improved delay if the sender has more next hop options available.

Multiple options for next hop nodes are available due to the redundancy inherent in dense sensor networks, which results in the existence of multiple shortest hop paths to the sink. To demonstrate the extent of redundancy inherent in a sensor network, consider an $N \times N$ grid network where the sink is located at one corner of the grid (Figure 2 shows an example 5×5 grid network). In such a network, the N nodes along the diagonal towards the sink and the immediate neighbors of the sink (in this case, 2 nodes excluding the one neighbor along the diagonal) only have one shortest hop path to the sink. The remaining $(N^2 - N - 2)$ nodes have two or more next hop options to route a packet towards the sink. For example, in a 10×10 grid network, 88 of 100 nodes have more than one next hop option to send their data packets to the sink.

To benefit from the redundancy available at the routing layer, it is essential that the different next hop nodes of the sender wake up at different times. However, when the nodes in the neighborhood wake up depends on the underlying duty-cycling MAC protocol. While some MAC protocols require their wakeup schedules to be synchronized, others operate in an asynchronous network.

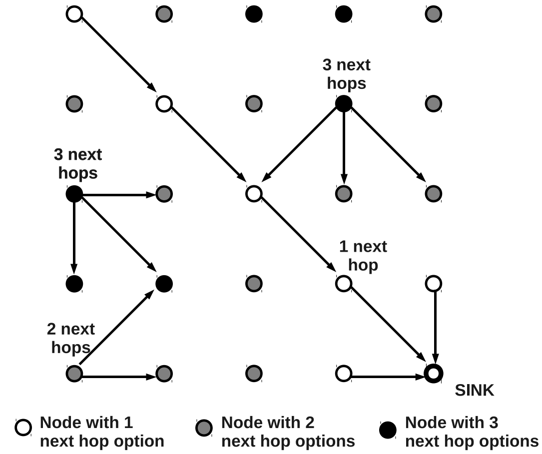

Fig. 2. Redundancy in a 5×5 grid network

Synchronous MAC protocols (such as S-MAC [8], T-MAC [9], IEEE 802.11 PSM [10]) have all nodes in the neighborhood wake up at the same time. Nodes achieve and maintain this synchronization by periodically exchanging schedule information among the neighbors. Since in these protocols, all nodes in the neighborhood wake up synchronously, providing multiple next hop options does not help the sender to find a next hop node that wakes up any sooner. Thus, these synchronous duty-cycling MAC protocols are not appropriate candidates for exploiting redundancy to improve delay. For the same reason, MAC protocols such as SCP-MAC [11] that loosely synchronize the wakeup schedules of the neighbors can not gain significant benefit from the redundancy at the route layer. Interestingly, these protocols put all of their effort into synchronization, hurting their chances to leverage other beneficial properties of these networks.

Asynchronous MAC protocols (such as B-MAC [12], X-MAC [6], SpeckMAC [13], WiseMAC [14], NPM [7]) have nodes wake up at different times, providing the sender an opportunity to find a next hop node that wakes up the soonest. However, whether the sender in an asynchronous MAC protocol can take advantage of these opportunities, and so benefit from the multiple next hop options depends on the mechanism that the protocol uses to determine whether the next hop node is awake or not. While some protocols (B-MAC, X-MAC, SpeckMAC) use signaling to wake up their next hop nodes, some protocols (WiseMAC) maintain schedule information about the neighbors to know exactly when the next hop node is going to wake up. There are also protocols (NPM) that apply a combination of both the mechanisms.

Nodes in signal-based asynchronous MAC protocols are completely unaware of the wakeup schedules of their neighbors. The most basic signaling protocol, B-MAC, always sends

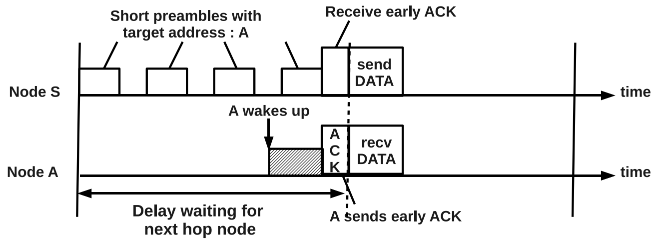

Fig. 3. Original X-MAC protocol

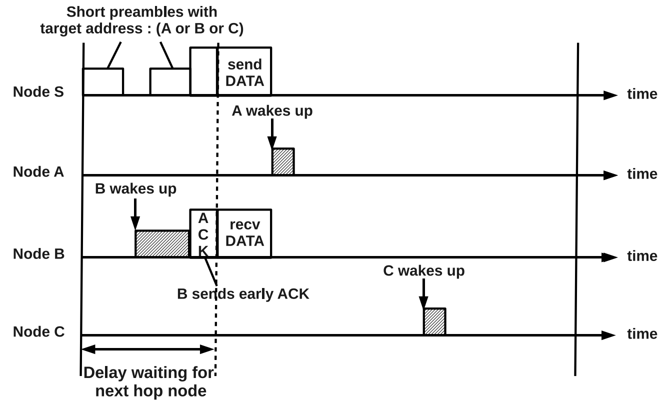

Fig. 4. Modified X-MAC protocol exploiting redundancy

a signal, which is the entire length of the wakeup period, to wake up the next hop node. Since the delay incurred at each hop in B-MAC is fixed and does not depend on when the next hop actually wakes up, B-MAC cannot benefit from multiple next hop options.

While B-MAC was one of the first signal-based protocols, many newer protocols provide improved performance by optimizing the length of the wake-up signal based on when the next hop wakes up. These optimizations enable these protocols to take advantage of the redundancy in the network. For example, X-MAC and SpeckMAC stop signaling as soon as they identify that the next hop node is awake. In order to notify the next hop node of the upcoming data transmission, a X-MAC sender sends short strobed signals with embedded target information. Upon receiving the signal, the receiver (i.e., the next hop node) sends back an acknowledgement to stop the signal and initiate the actual data transmission. The early acknowledgement from the awake next hop node allows X-MAC to take advantage of the multiple next hop options. By embedding a list of the multiple targets in the strobed signals, the X-MAC sender can now find a next hop node that wakes up the soonest. This next hop node that wakes up first can send the acknowledgement earlier, thus reducing the delay incurred. SpeckMAC uses a mechanism similar to X-MAC to identify whether the next hop node is awake. However, instead of sending strobed signals like X-MAC, a SpeckMAC sender repeatedly sends the data messages. As the next hop node wakes up and receives the data, it sends back an acknowledgement to the sender to indicate a successful data transmission. In this case, it is possible to incorporate multiple next hop node addresses inside the original SpeckMAC data packet to take advantage of the redundancy. Like X-MAC with multiple next hop options, the SpeckMAC next hop node that wakes up first can send back the acknowledgement earlier and can reduce the wait time of the sender.

In comparison, some asynchronous MAC protocols maintain synchronization information about the wakeup schedules of their neighbors. This knowledge is achieved through the periodic exchange of schedule information, similar to synchronous MAC protocols. However, the synchronization mechanisms are used for a different purpose. Instead of synchronizing the wakeup times of the neighbors, nodes use the synchronization information to maintain the neighborhood wake up schedule information. For example, WiseMAC utilizes the information available in a neighbor schedule table to enable the sender to know how long to wait until the next hop node wakes up and only then sends its data packets. This synchronization mechanism allows WiseMAC to take advantage of the route level redundancy. With multiple next hop options, the WiseMAC sender can identify the next hop node that wakes up the soonest by consulting its neighborhood schedule table. The WiseMAC sender can then wait for that particular next hop node to wake up before sending its data packets. This reduces the wait time of the WiseMAC sender.

Finally, Neighborhood-based Power Management (NPM) uses a combination of both signaling and synchronization to wake up the next hop node in an asynchronous network. NPM enables all nodes awakened by the wakeup signal to send their data messages (referred to as opportunistic sending) to amortize the cost of signaling over multiple data transmissions. Since the wakeup signal may not wake up the receivers for all senders in the neighborhood (due to reasons such as the receiver being out of transmission range of the wakeup signaling node), NPM exchanges control messages during a control window just after the wakeup signal, to identify whether the receiver is awake. After identifying the awake receivers, the senders exchange their data messages during a data window which follows the control window. The nodes which do not have awake receivers or are not the receivers of any data transmission, go back to sleep after the control window. Nodes in NPM opportunistically gain wakeup schedule information from piggybacked information onto the control messages. This schedule information helps NPM to shorten its wakeup signal. Since, the synchronization in NPM is loose, the NPM sender shortens its wakeup signal only when the schedule information is up-to-date. When recent information is not available, NPM sends a long wakeup signal.

The signaling mechanism in NPM provides the senders opportunity to take advantage of the redundancy available in wireless sensor networks: First, multiple next hop options allow the wakeup signaling node of NPM opportunity to reduce the time it waits for the next hop node to wake up by choosing the next hop node that wakes up the soonest as the target of the wakeup signal. Second, with multiple next hop options, NPM sender now has higher probability of having recent wakeup schedule information about at least one of the next hop nodes, thus increasing its opportunity to shorten the wakeup signal. Third, multiple next hop options increases the probability that a sender woken up by wakeup signal will find an awake nect hop node, thus increasing the possibility of opportunistic sending.

To summarize, asynchronous MAC protocols where the delay depends on the exact wakeup time of the next hop node can benefit from route level redundancy and diverse wakeup schedules of the nodes. Depending on the MAC protocol, the redundancy can provide benefits in three different ways:

- Signal-based asynchronous protocols [6], [13] can benefit from multiple next hop options since the sender can stop sending wakeup signals as soon as it detects any of the next hop nodes to be awake.
- Synchronization-based asynchronous protocols [14] can benefit from multiple next hop options since the sender has the opportunity to choose the closest awake next hop node using its knowledge about the neighbors' wakeup schedules.
- Multiple next hop options increases the probability of the sender finding an awake next hop node even when the sender sends its data opportunistically without sending a wakeup signal.

III. Applying MAC layer Anycast in duty-cycled networks

MAC-layer anycast provides support for successful data transmission when the sender has multiple next hop options, thus allowing the nodes to exploit route level redundancy. MAC-layer anycast has been proposed by Choudhury et. al. [15] for wireless ad hoc networks where the sender utilizes its knowledge of channel conditions (available at the MAC layer) while sending data to its next hop node to ensure high data delivery ratios. In our research, we take a novel approach and integrate MAC-layer anycast into duty-cycled wireless sensor networks. This application MAC-layer anycast enables the exploitation of route level redundancy, improving delay and, in some cases, energy in duty-cycled wireless sensor networks.

In this section, we describe the basic MAC-layer anycast framework for wireless ad hoc network that we then apply to wireless sensor networks. We also provide some examples of how anycasting can be incorporated into two duty-cycling MAC protocols, X-MAC and NPM, using very small modifications to the original protocols.

A. Basic MAC-layer anycast framework

In the base MAC-layer anycast framework [15], the routing protocol supplies multiple next hop options to a sender. From the sender's perspective, sending its data packet to any of the next hop nodes progresses the data towards the destination. Although all next hop nodes (except the ones that are asleep due to duty cycling) receive the data because of the broadcast nature of wireless medium, only one node should forward the data.

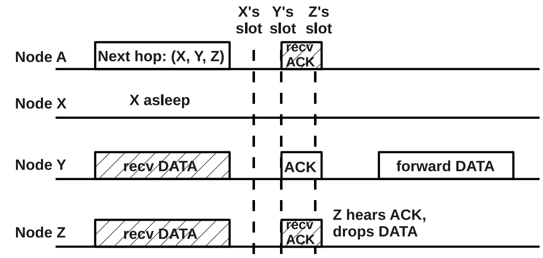

Fig. 5. Slotted ACK mechanism for MAC-layer anycast

Receivers of the anycast packet adopt a slotted ACK mechanism (see Figure 5) to avoid ACK collisions from multiple receivers. This mechanism also allows the receivers to determine the next hop forwarder node in a distributed fashion.

The anycast sender can forward its data packet using any of the next hop nodes since each of them provide an equal cost path to the destination. However, the sender can also set its preference for a particular next hop node to be the forwarder by assigning different priorities to the different next hop nodes and including the priority information inside the anycast packet. The receivers can use the priorities embedded inside the data packet to choose their slots (each slot long enough for a receiver to start transmitting its acknowledgement) for sending ACKs. If no priority is set by the anycast sender, the next hop nodes can use the order in which they are listed inside the anycast packet as their priority to be the forwarder. In this scheme, any receiver that overhears an ACK refrains from sending its ACK and drops the data packet. For this to work, we assume that the next hop nodes are within range of each other, and thus one next hop node can overhear the acknowledgement sent by another next hop node. Thus, the first next hop node that sends back the ACK will be the node that forwards the data packet.

With the slotted ACK mechanism, anycast incurs no extra delay when the primary next hop node (i.e., the next hop node with the highest priority to be the forwarder) is awake. After receiving the anycast packet, the receiver sends back the ACK just after one slot, acting exactly like unicast. Moreover, the multiple next hop options in MAC-layer anycast offer opportunities for successful data transmissions even when the primary next hop node is not awake.

B. Applying MAC-layer Anycast to X-MAC and NPM

It is relatively easy to augment appropriate MAC layer protocols. In order to benefit from redundancy, MAC protocols do not need to enable anycast for all of its messages (i.e., data and control). Once the sender identifies the next hop node that is awake with the help of anycast, the sender can directly send its data messages to that particular next hop node without performing anycast. Since MAC protocols generally identifies awake next hop nodes by using special control messages, we need to enable anycast for only these specific control messages of the MAC protocols.

In this section, we discuss how MAC-layer anycast can be incorporated into X-MAC [6] and Neighborhood-based Power Management (NPM) [7], [16] to enable these protocols exploit redundancy in the network:

- The augmented X-MAC protocol sends `anycast wakeup signals` to enable the sender identify the next hop node that wakes up the soonest. This next hop node sends back an acknowledgement according the slotted ACK mechanism. Once the awake next hop node is identified, the sender sends unicast data packets to that node.
- The augmented NPM protocol sends `anycast control messages` during its control window to identify whether any of its next hop options are awake. If a next hop node is awake during the control window, it sends back an acknowledgement using the slotted ACK mechanism. NPM sender then directly sends its data messages to that particular next hop node during the following data window.
- Before sending the wakeup signal, the augmented NPM protocol consults the neighbor schedule table and identifies the next hop node that wakes up the soonest. The `anycast preambling node` (i.e., the node that sends the wakeup signal) then selects that particular next hop node as the target of its wakeup signal, thus reaching the next hop node and then initiating the control window sooner.

The end result of this modification is improved wakeup signaling for X-MAC, and increased use of opportunistic sending as well as reduced wait time for NPM.

IV. Evaluation

The goal of our evaluation is to analyze how MAC protocols for WSNs can benefit from the inherent redundancy at the routing layer. As a proof-of-concept, we added MAC-layer anycast support to two duty cycling MAC protocols: X-MAC and NPM using *ns-2*, and analyzed their delay and energy performance while exploiting different levels of redundancy. Results for the modified X-MAC with our proposed anycast portrays how delay and energy can be reduced exploiting redundancy since anycast allows the sender to choose the first awakened next hop node. On the other hand, results for the modified NPM with incorporated anycast shows how anycast improves its signaling and opportunistic sending (explained in section II), resulting in improved delay performance.

Simulation Setup

We evaluated our prototypes using *ns-2* in two different network setups: a simple 5 node network with ideal wakeup schedules for the nodes, and a 100 node grid network with nodes having random wakeup schedules.

Simple Network Setup: The simple 5 node network consists of one source node and one sink node, with the source node having 3 possible next hop options to forward its data packets towards the sink (Figure 6 shows the simple network used for our simulations). The three intermediate nodes are spaced 140 meter apart, so that one next hop node can detect channel activity due to transmissions from other the intermediate (i.e., next hop) nodes.

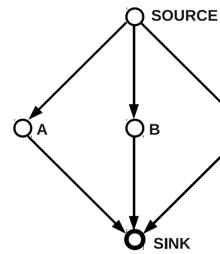

Fig. 6. Simple Network

Each node in the simple network wakes up periodically every 100 `msec` and remains awake for a 2 `msec`. We selected the wakeup times of the nodes in this network such that the wakeup schedules of the 3 next hop nodes of the source are equally spaced within a 100 `msec` time period. Thus, the simulation results for this simple network with an ideal wakeup scenario shows the maximum benefit achievable by exploiting the redundancy at the routing layer and the diversity of wakeup schedules. In this network, we only evaluate X-MAC.

Grid Network Setup: We also evaluated our prototypes in a 100 node grid network. The nodes in this network are spaced at 140 meter distance from each other. With a transmission range of 250 meters, a node in this network setup can have a maximum of eight neighbors. The network has a single sink, located at one corner of the grid (similar to the network in Figure 2). Each node in the network (except the sink) generates data packets towards the sink.

Each node in the grid network wakes up periodically maintaining a 100 msec wakeup period. However, to simulate a realistic network scenario, the first wakeup times of the different nodes in the network are chosen from a uniform random distribution, $U = (0, 100)$ `msec`. We evaluated both X-MAC and NPM in the grid network. During each 100 `msec` wakeup period, nodes in X-MAC remain awake for 2 `msec`, whereas the nodes in NPM remain awake for 1 `msec`. The reason for the longer awake time for X-MAC is due to the signaling strategy of X-MAC. Since X-MAC senders wait for an acknowledgement from the receiver before sending the next signal, X-MAC receivers need to remain awake for a longer time to detect a signal directed towards themselves. The other protocol specific parameters for NPM are listed in Table IV.

Control window	30 msec
Data window	600 msec
Guard time around wakeup signal	2 msec
Refresh timer for neighbor schedule table	60 msec
Immediate timer	10 msec

TABLE I
NPM PROTOCOL PARAMETERS

While results in the simple network show the maximum benefit achievable in an ideal setup, the results for the grid network captures the average benefit achievable in a realistic network, because:

- Different nodes in the grid network have different numbers of next hop options, depending on the location of the nodes in the grid (explained in detail in section II).
- The wakeup schedules of the different nodes in the network are random (as expected in an asynchronous duty-cycled network).

For both the simple ideal network and the realistic grid network, data is generated from a CBR traffic generator. We varied the load in the network by varying the inter-arrival time between successive CBR packets from 600 sec to 5 sec. Thus, in our traffic setup, an inter-arrival time of 600 sec represents low load, whereas a 5 sec inter-arrival represents high load.

To evaluate how the modified MAC protocols benefit from the different levels of redundancy, we varied the maximum number of next hop options the route layer provides to the MAC layer from 1 to 2 to 3. With 1 next hop option, the protocols act as their original version of the protocols that do not exploit redundancy. We collected data over a 1 hour simulation period.

Evaluation Metric

We compared the delay and energy performances of each baseline protocol (X-MAC and NPM) with its own modified version, incorporated with MAC-layer anycast to exploit the route level redundancy. The comparison shows how much route level redundancy can benefit an asynchronous duty-cycling protocol.

We use *delay per hop* as a comparison metric for delay. In the grid network, data packets are generated by all nodes in the network (except the sink). Since, different nodes are located at different hops from the sink, each packet traverses different length paths before it reaches its final destination (i.e. the sink). Thus, average delay per hop provides us the appropriate metric to compare delay across varying traffic generation rates when different packets traverse different length paths. Moreover, delay per hop captures the time a sender spends waiting for its next hop node to be awake, which anycast targets to reduce by exploiting redundancy.

Voltage	3.0 V
Transmission power (P_{tx})	$17.4mW$
Reception power (P_{rx})	$18.8mW$
Idle power (P_i)	$1mW$
Sleep power (P_s)	$0.1mW$

TABLE II

POWER CONSUMPTION AT DIFFERENT RADIO STATES

To compare the energy performance across different traffic generation rates, we choose *energy per bit* as a metric. Since, energy consumption varies as the radio switches between different states (transmit, receive and idle), in our simulations, we track the time each node spends in the different states. By using the power characteristics of the radio transceiver, CC2420 of sensor mote, MicaZ (Table II lists the energy profile of CC2420), we calculated the total energy consumption for all nodes in the network.

A. Simple Network

We simulated both the original and the modified version of the X-MAC protocol, augmented with anycast to exploit the redundancy available in the simple network. Since the source in this network can have a maximum of 3 next hop options and the wakeup schedules of the available next hop options are equally spaced, we expect to observe the maximum benefit of redundancy in this setup.

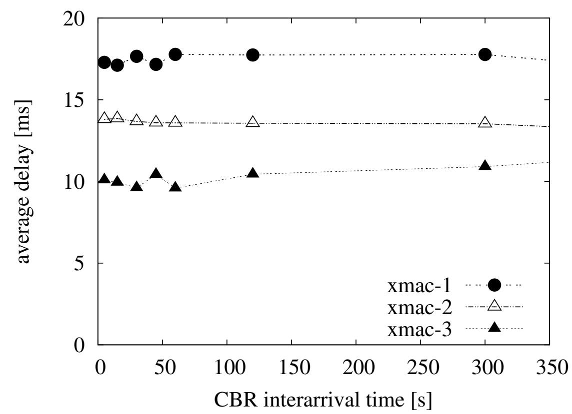

Fig. 7. Delay in Simple Network

We varied the number of next hop options from 1 to 3 to determine the benefit (both in terms of delay and energy) gained at the different levels of redundancy. With 2 next hop options (represented by `xmac-2` in the graphs), we observe a 34% reduction (see Figure 7) in the average delay per hop for the modified X-MAC, compared to the delay for the original X-MAC protocol (represented by `xmac-1` in the graphs). However, as we increased the number of next hop options to 3 (represented by `xmac-3` in the graphs), we observe a 20% improvement from having 2 next hop options. Thus, the delay improvement achieved by X-MAC with anycast closely tracks the delay expected ($E[\delta_n] = \frac{W}{n+1}$) when nodes have n next hop options. The slight difference occurs due to the single next hop option available at the last hop in this network. Since X-MAC delay is caused by wakeup signals, reduction in the per hop delay in this case infers that X-MAC now saves energy that will otherwise be spent in sending wakeup signals. Thus, we observed a similar improvement in energy as achieved for delay in X-MAC (the energy graph is omitted due to space constraints).

In an ideal setup, more next hop options provide the sender with more opportunities to choose a next hop node that wakes up earlier, and hence to achieve more delay and energy improvements. However, the percentage of improvement diminishes as the number of next hop options increases (as seen in Figure 7).

B. Grid Network

We simulated both the original and the modified version of X-MAC and NPM protocols to evaluate the benefit achieved in a more realistic network. Since, different senders in this network have different numbers of next hop options due to their location in the grid, and the wakeup schedules of the next hop options are distributed randomly, we expect to achieve less benefit in this network compared to the ideal setup. However, the results here represent the average case improvement achievable in a more generic realistic network.

Like the ideal setup, we varied the maximum number of next hop options allowed by the routing layer to the underlying MAC-layer (referred to as level of redundancy) from 1 to 3 to determine the benefit both in terms of delay and energy gained by X-MAC and NPM at different redundancy level.

Delay and Energy for X-MAC with anycast

The average delay per hop reduces as the level of redundancy increases (see Figure 9). We observed around 33% reduction (see Figure 8) in the average length of wakeup signals for the modified X-MAC with anycast capability, when the routing layer provided 2 next hop options to the MAC layer (represented by `xmac-2`) compared to the original X-MAC protocol (represented by `xmac-1`), when each sender has just 1 next hop option. This extent of improvement achieved in this case is quite similar to the improvement achieved in an ideal setup. However, in a grid setup, the modified X-MAC protocol does not achieve much benefit when the routing layer provides 3 next hop options to the MAC-layer. We observed only an 8% improvement in this case compared to when the routing layer provides 2 next hop options. The lower improvement with 3 next hop options (represented by `xmac-3`) in the grid network compared to the ideal setup stems from the very few opportunities available in a grid setup to actually benefit from the higher level of redundancy:

- Not all nodes in the grid setup have 3 next hop options.
- As packets traverse further towards the sink node, the number of next hop options at each hop also reduces.

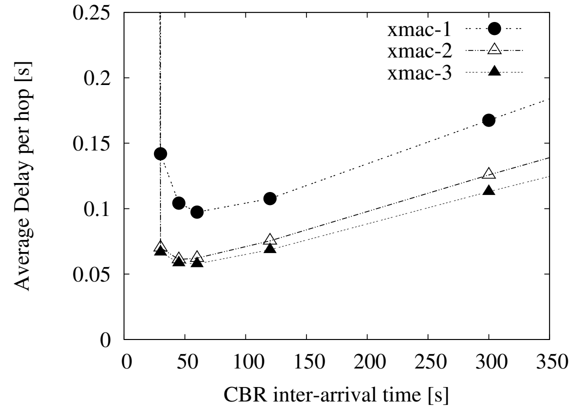

Fig. 9. Delay for XMAC with anycast

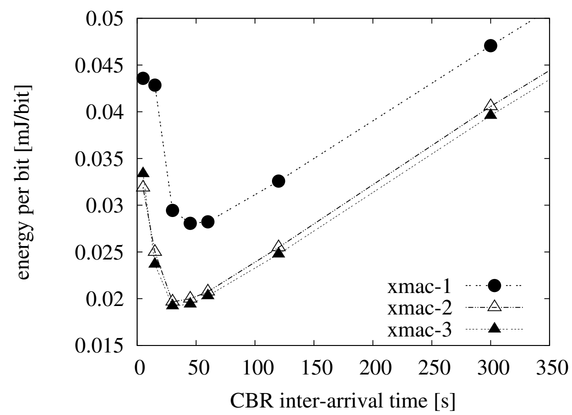

Fig. 10. XMAC with MAC-layer anycast

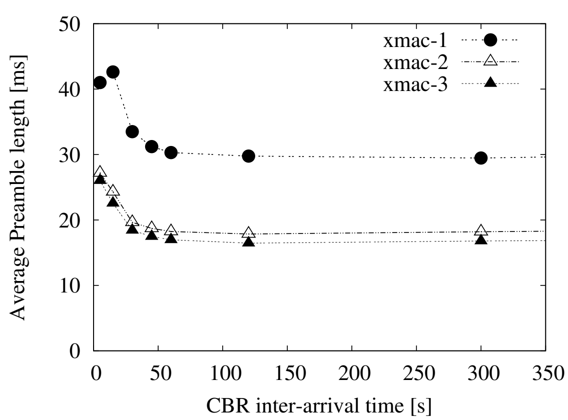

Fig. 8. Signal length for XMAC with anycast

However, in the grid setup, almost all nodes (except the ones located along the diagonal to the sink, and the direct neighbors of the sink), have at least 2 next hop options at the next hop. Thus, the modified X-MAC can achieve benefit close to the ideal setup when the routing layer provides 2 next hop options to the nodes. The reduced wakeup signal length of modified X-MAC with more anycast options results in similar improvement in delay and energy (see Figure 9 and 14).

As the traffic generation rate increases, nodes in X-MAC need to wait longer to find an idle channel so that the sender can start transmitting its own wakeup signals. At very high traffic, the network gets saturated, and thus the delay for the original X-MAC protocol shoots up (see Figure 9). Anycast enables X-MAC nodes to handle network saturation better by incurring less delay at each hop and by distributing the load across multiple paths. We measured the delivery ratio of X-MAC to capture the protocol's ability to handle high traffic loads. The delivery ratio for X-MAC increases as the number of next hop options increases (see Figure 11). The lower delay with more next hop options allows the sender to send its data faster so that queues only start building up at a relatively higher traffic rate. Moreover, since the sender now has multiple next hop options to forward its data, the load on a particular next

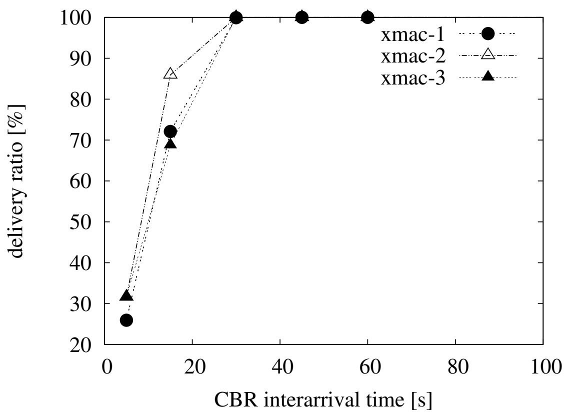

Fig. 11. Delivery ratio for X-MAC with anycast

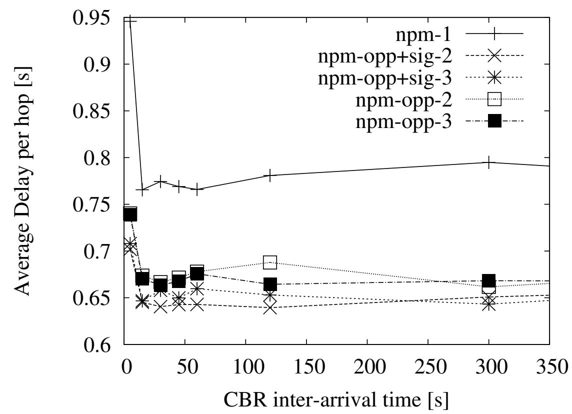

Fig. 13. Delay for NPM-targeted with anycast

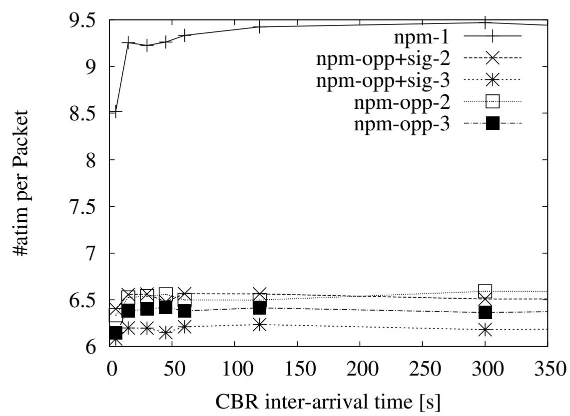

Fig. 12. Number of ATIMs per Pkt for NPM-targeted with anycast

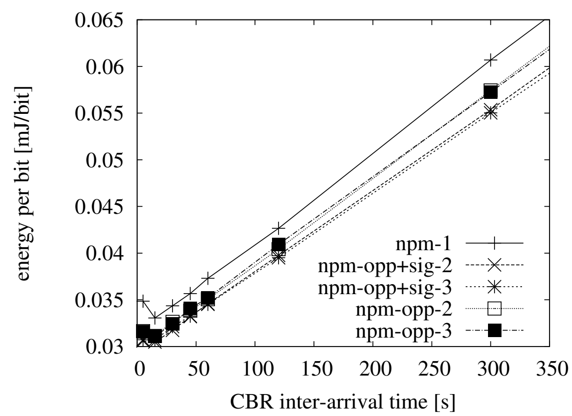

Fig. 14. Energy for NPM-targeted with anycast

hop node reduces due to the diversity. This also results in a reduced rate for the queue buildup of the nodes in the network, improving their delivery ratio.

Delay and Energy for NPM with anycast

We implemented two prototypes for the modified NPM to exploit the route level redundancy. The first augments the control messages of NPM with anycast capability to gain benefits from increased possibility of opportunistic sending (referred to as `npm-opp` in the graphs). The second enables anycasting for the wakeup signals of NPM in addition to having anycast control messages to gain both reduced wait times and increased opportunistic sending (referred to as `npm-opp+sig` in the graphs).

For the first prototype `npm-opp`, we observed more successful opportunistic sending as the number of next hop options provided from the routing layer increased. This is evident as we analyze the total number of control messages exchanged as the number of next hop options increases. With more next hop options, NPM has more probability of finding an awake next hop node, resulting in more control message exchanges ending successfully. As a result, we observed fewer control messages exchanged with more next hop options (see Figure 12). Since the data sent opportunistically no longer needs to wait for a wakeup signal, delay reduces (see Figure 13). However, since packets in NPM are delayed by the control window introduced after the wakeup signal to enable opportunistic sending, the delay improves by around 12%. Additionally, as more data packets are sent using one wakeup signal, the number of wakeup signals required also reduces (see Figure 16). However, since the wakeup signals and the control messages are small, NPM with anycast shows only little improvement in terms of energy.

With anycast improving both signaling and opportunistic sending of NPM (i.e. for `npm-opp+sig`), the delay improvement is higher because NPM now takes advantage of the multiple next hop options to reduce its wait time for sending a wakeup signal. The better improvement of delay (see Figure 13) stems from two reasons:

- As the number of next hop options increase, NPM has more probability to choose a next hop node that wakes up sooner.
- With more next hop options, NPM has a higher probability of having recent information about at least one of its next hop options. Since, NPM sends a long preamble only when it cannot utilize the wakeup schedule information,

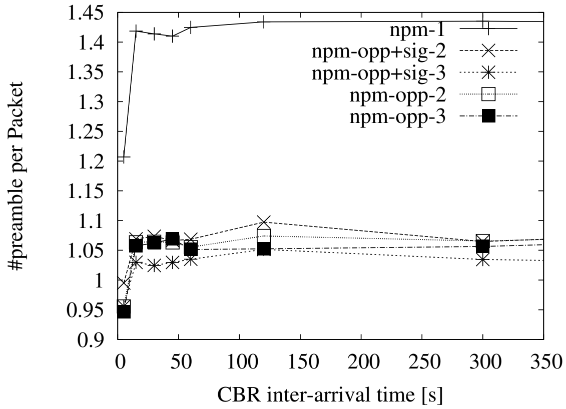

Fig. 15. Number of Preambles per Pkt for NPM-targeted with anycast

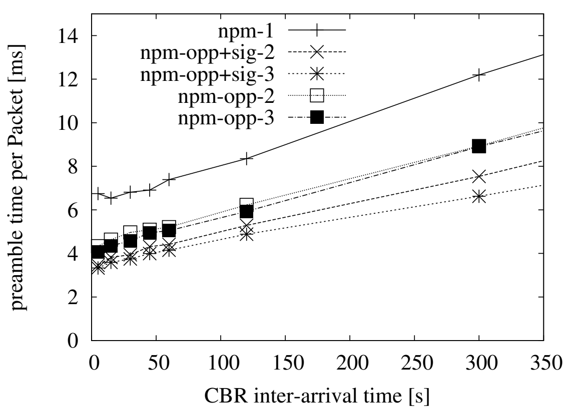

Fig. 16. Preamble time per Pkt for NPM-targeted with anycast

more next hop options definitely reduces the frequency of sending a long wakeup signal.

We can verify this claim by analyzing the length of wakeup signals (see Figure 16) for `npm-opp+sig` as NPM has more next hop options. Because of the reduced wakeup signal length, we observe relatively better energy improvements for `npm-opp` than `npm-opp+sig`.

Thus, to summarize, by exploiting the inherent redundancy in WSNs, MAC protocols can definitely improve delay and to some extent energy by applying anycast. Moreover, the simple modifications required to incorporate anycast into a duty-cycling MAC protocol and the promising improvements achieved in a realistic network encourages the adoption of exploiting redundancy.

V. Conclusion and Future Work

Wireless sensor networks often have multiple shortest hop paths between the same source-destination pair due to the dense deployment of sensors. This redundancy inherent in the network, offers great opportunity to reduce the delay incurred at each hop in an asynchronous duty-cycled network. We propose to exploit this redundancy by incorporating anycast into MAC protocols for reducing the time the senders wait for their next hop nodes to be awake. We have shown that using minimal modification to the existing MAC protocols, we can enable MAC-layer anycasting and achieve both delay and energy improvements. Our prototypes for X-MAC and NPM modified to incorporate anycast capability achieve 30% and 12% reduction in delay respectively as the routing layer provides 1 additional next hop option to the sender.

In this paper, we have explored the benefits that the duty-cycled MAC protocols can achieve from redundancy by applying minimal modification to the original protocols. However, further benefits can be achieved from redundancy using MAC-layer anycasting, if we do not restrict ourselves to applying minimal modifications. For example, MAC-layer anycasting can allow a duty-cycled MAC protocol to choose its next hop nodes selectively to enable data exchange using fewer nodes. In future, we want to explore whether such an approach can extend network lifetime.

References

[1] R. Zheng, J. C. Hou, and L. Sha, “Asynchronous wakeup for ad hoc networks,” in *MobiHoc*, 2003.
[2] Y. Wong, L. Ngoh, and W. Wong, “An adaptive wakeup scheme to support fast routing in sensor networks,” in *PE-WASUN*, 2005.
[3] A. Keshavarzian, H. Lee, and L. Venkatraman, “Wakeup scheduling in wireless sensor networks,” in *MobiHoc*, 2006.
[4] J. Jiang, Y. Tseng, C. Hsu, and T.-H. Lai, “Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks,” in *ICPP*, 2003.
[5] K. Romer, P. Blum, and L. Meier, “Time synchronization and calibration in wireless sensor networks,” *Handbook of Sensor Networks: Algorithms and Architectures*, 2005.
[6] M. Buettner, G. Yee, E. Anderson, and R. Han, “X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks,” in *SenSys*, 2006.
[7] F. Ashraf, R. Crepaldi, and R. Kravets, “Know Your Neighborhood: A Strategy for Energy-efficient Communication,” in *MASS*, 2010.
[8] W. Ye, J. Heidemann, and D. Estrin, “An energy-efficient MAC protocol for wireless sensor networks,” in *INFOCOM*, 2002.
[9] T. van Dam and K. Langendoen, “An adaptive energy-efficient MAC protocol for wireless sensor networks,” in *SenSys03*, 2003.
[10] J. Choi, Y. Ko, and J. Kim, “Enhanced power saving scheme for IEEE 802.11 DCF based wireless networks,” *Lecture notes in computer science*, 2003.
[11] W. Ye, F. Silva, and J. Heidemann, “Ultra-low duty cycle MAC with scheduled channel polling,” in *SenSys*, 2006.
[12] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” in *SenSys*, 2004.
[13] K.-J. Wong and D. K. Arvind, “SpeckMAC: Low-power decentralised MAC protocols for low data rate transmissions in specknets,” in *REALMAN*, 2006.
[14] A. El-Hoiydi and J. Decotignie, “WiseMAC: an ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks,” in *iscc*, 2004.
[15] R. Choudhury and N. Vaidya, “Mac-layer anycasting in ad hoc networks,” *ACM SIGCOMM Computer Communication Review*, 2004.
[16] F. Ashraf and R. Kravets, “Poster abstract: neighborhood-based power management,” *ACM SIGMOBILE Mobile Computing and Communications Review*, 2010.