To be presented in MobiHoc '08

DW-MAC: A Low Latency, Energy Efficient Demand-Wakeup MAC Protocol for Wireless Sensor Networks

Yanjun Sun* yanjun@cs.rice.edu
Shu Du* dushu@cs.rice.edu
Omer Gurewitz† gurewitz@cse.bgu.ac.il
David B. Johnson* dbj@cs.rice.edu

*Department of Computer Science, Rice University, Houston, TX, USA
†Department of Communication Systems Engineering, Ben Gurion University, Israel

ABSTRACT

Duty cycling is a widely used mechanism in wireless sensor networks (WSNs) to reduce energy consumption due to idle listening, but this mechanism also introduces additional latency in packet delivery. Several schemes have been proposed to mitigate this latency, but they are mainly optimized for light traffic loads. A WSN, however, could often experience bursty and high traffic loads, such as due to broadcast or convergecast traffic. In this paper, we present a new MAC protocol, called *Demand Wakeup MAC* (*DW-MAC*), that introduces a new low-overhead scheduling algorithm that allows nodes to wake up on demand during the Sleep period of an operational cycle and ensures that data transmissions do not collide at their intended receivers. This demand wakeup adaptively increases effective channel capacity during an operational cycle as traffic load increases, allowing DW-MAC to achieve low delivery latency under a wide range of traffic loads including both unicast and broadcast traffic. We compare DW-MAC with S-MAC (with and without adaptive listening) and with RMAC using *ns-2* and show that DW-MAC outperforms these protocols, with increasing benefits as traffic load increases. For example, under high unicast traffic load, DW-MAC reduces delivery latency by 70% compared to S-MAC and RMAC, and uses only 50% of the energy consumed with S-MAC with adaptive listening. Under broadcast traffic, DW-MAC reduces latency by more than 50% on average while maintaining higher energy efficiency.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*Access Schemes*

General Terms

Algorithms, Design, Performance

Keywords

Sensor networks, medium access control, duty cycling, unicast traffic, broadcast traffic, latency, energy

MobiHoc'08, May 26–30, 2008, Hong Kong SAR, China.

1. INTRODUCTION

Wireless sensor networks (WSNs) have a significant potential in applications interacting with the physical world, such as surveillance and environmental monitoring. In many of these applications, the use of battery-powered sensor nodes greatly eases the deployment of the network, but the limited capacity of these batteries substantially limits the network lifetime.

One of the largest sources of energy consumption in wireless nodes is the use of *idle listening*, and many solutions to reducing this problem in WSNs have been proposed based on the use of *duty cycling* [20, 27]. In duty cycling, sensor nodes periodically alternate between being active and sleeping. When active, a node is able to transmit or receive data, whereas when sleeping, the node completely turns off its radio to save energy; duty cycles of 1–10% (percentage of time in the active state) are typical in order to maximize energy savings. In order to transmit a packet from one node to another, the radios of both nodes must be on, motivating the use of synchronization between the operational cycles of different nodes. Examples of protocols using synchronized approaches include S-MAC [26, 27], T-MAC [4], and RMAC [5].

For example, in S-MAC [27] time at each sensor is divided into repeated operational cycles, each further divided into three periods: *Sync*, *Data*, and *Sleep*. Nodes in S-MAC wake up at the start of the Sync period to synchronize clocks with each other. During the Data period, all nodes remain active. If a node has a packet to send to a neighbor node, they exchange Request-to-Send (RTS) and Clear-to-Send (CTS) frames during the Data period, followed by the transmission of the data packet and the return of an Acknowledgement (ACK) frame. Nodes not involved in communication initiated during the Data period return to the sleep state at the start of the Sleep period; other nodes return to the sleep state only after completion of the ACK frame.

Although such approaches save energy, they can add significant latency in packet delivery, since transmission of a packet from one node to a neighbor node must wait until the next time the nodes are active, if the nodes are currently sleeping. Furthermore, forwarding a packet over multiple wireless hops, as is common in WSNs, often requires multiple operational cycles to complete.

Several approaches have be proposed to mitigate the additional latency introduced by duty cycling [4, 5, 26], but they are mainly optimized for light traffic loads. A WSN, however, could often experience bursty and high traffic loads. For example, either broadcast [18] or convergecast [29] traffic could suddenly increase channel contention in a local neighborhood. In WSNs, broadcast is widely used for various network-wide queries and updates [24], and convergecast is often observed when multiple sensors that have detected the same event send their reports to the sink node or to a node that does data aggregation [7].

As existing approaches are mainly optimized for light traffic loads, we found that they become less efficient in latency, power efficiency, and packet delivery ratio (fraction of data packets successfully delivered) as traffic load increases. As traffic in a WSN can be quite dynamic, depending on the events being sensed and the sensing application and protocols being used, an ideal WSN MAC protocol should perform well under a wide range of traffic loads, including high loads and bursty traffic.

In this paper, we present a new MAC protocol, called *Demand Wakeup MAC* (*DW-MAC*), that introduces a new low-overhead scheduling algorithm that allows nodes to wake up on demand during the Sleep period of an operational cycle in order to transmit or receive a packet. This demand wakeup adaptively increases effective channel capacity during an operational cycle as traffic load increases, allowing DW-MAC to achieve low delivery latency under a wide range of traffic loads including both unicast and broadcast traffic.

DW-MAC differs from prior work in reducing the additional latency introduced by duty cycling. In DW-MAC, medium access control and scheduling are integrated, in that during a Data period of an operational cycle, the interval of time during which the transmission of an access control frame occupies the medium automatically reserves the proportional interval of time in the following Sleep period for transmitting and receiving a data packet. This integration minimizes scheduling overhead and collisions. Further, by avoiding transmission of data packets in a Data period, DW-MAC maximizes the number of access control frames that can be exchanged in a Data period, thus increasing the number of data packets that can be exchanged in a complete operational cycle. The contributions of this work are as follows:

- We introduced a new low overhead scheduling algorithm that ensures that data transmissions do not collide at their intended receivers.
- We present the design of DW-MAC that wakes up nodes on demand in order to efficiently handle a wide range of traffic load including both unicast and broadcast traffic.
- DW-MAC wakes up a node in a Sleep period only when the node needs to transmit or receive a packet, in order to minimize energy consumption.
- DW-MAC achieves lower latency, higher power efficiency, and higher packet delivery ratio compared to existing schemes.

The rest of this paper is organized as follows. In Section 2, we discuss related work in approaches to reducing latency in synchronized duty cycle MAC protocols for WSNs. Section 3 presents the detailed design of DW-MAC, including support for both unicast and broadcast traffic and for optimized multihop forwarding. Section 4 presents results from our simulation-based evaluation of DW-MAC, including a comparison with S-MAC (with and without adaptive listening) and with RMAC, and finally, in Section 5, we present conclusions.

2. RELATED WORK

A number of previous approaches to reduce latency in synchronized duty cycle MAC protocols for WSNs have been proposed, although none provides the generality or performance of our DW-MAC approach. We discuss these previous approaches here.

S-MAC [27] was one of the original synchronized duty cycle MAC protocols for WSNs, and the developers of S-MAC later introduced a modification known as *adaptive listening* [26] to improve its end-to-end delivery latency over multiple hops. With adaptive listening, if a node overhears another node's communication (e.g., the RTS or CTS) during the Data period, it wakes up

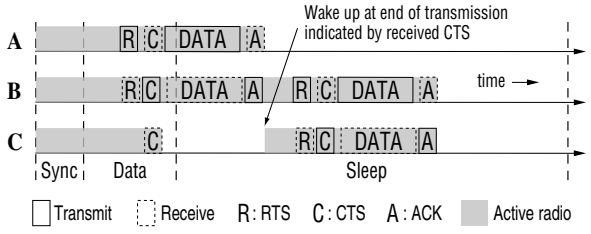


Figure 1: S-MAC with adaptive listening.

for a short time when the overheard communication finishes; if this node is the next-hop node along a multi-hop path, its neighbor can forward the packet immediately to this node rather than waiting for the Data period in the next operational cycle to initiate the forwarding. Figure 1 shows an example of the operation of S-MAC with adaptive listening. Node A here sends a data packet to node B, with a next-hop node of C. When node C overhears the CTS from B, it goes to sleep but wakes up again when the ACK from B should have been completed, based on the information in the overheard CTS. Node B can immediately forward the data packet to C at this time.

S-MAC with adaptive listening can deliver a packet up to 2 hops per operational cycle but generally cannot go beyond that within the cycle since the next hop after C (such as some node D) is unlikely to have been awake to overhear the communication from B to C; node C will transmit an RTS to D but will go back to sleep itself when it fails to receive a CTS in reply from D. The use of adaptive listening can also cause a significant increase in energy consumption, since many neighboring nodes may overhear the RTS or CTS and wake up, whereas only one of them is the next-hop node. Moreover, since a node does not wake up until an overheard communication ends, this node then may not have complete knowledge of the busy state of the wireless medium. For example, the node might have missed hearing an RTS or CTS of another data transmission in the neighborhood; if the node in this case starts transmitting any packet, the packet may cause collisions at other nodes.

Similarly, T-MAC [4] can reduce latency by adaptively changing the ending time of a Data period. Although T-MAC is primarily designed to shorten the Data period when no traffic is around the node, so that nodes can preserve more energy, T-MAC can also extend the Data period to allow multihop forwarding during a single Data period. However, as with S-MAC with adaptive listening, T-MAC can generally deliver a packet over only at most 2 hops within an operational cycle, since nodes further downstream will be unlikely to overhear the upstream communication 2-hops away and thus will not remain awake to receive a forwarded packet; T-MAC may also increase energy consumption, as many nodes other than an intended next-hop node will remain awake.

Several other approaches to reducing latency have been proposed, that make specific assumptions on the communication pattern among nodes or on the other protocols used in the WSN. For example, DMAC [15] reduces latency only for data gathering communication in which multiple nodes try to send data to a sink node through a unidirectional tree of paths. Likewise, the streamlined wakeup optimization proposed by Cao et al. [2] address only the case in which each sensor node sends data to a sink node (although there may be more than one sink node for the network). For a network of tree topology or ring topology, Lu et al. [16] discuss how to minimize end-to-end latency. The work of Keshavarzian et al. [11] analyzes latency for specific communication and wakeup patterns for communication with the sink node and proposed the multi-

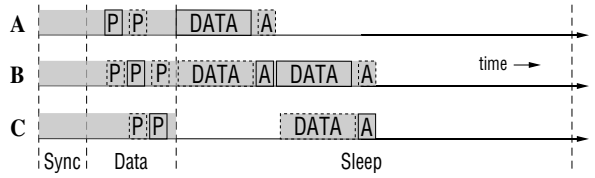


Figure 2: Multihop forwarding of a unicast packet in RMAC. P indicates a PION frame that is used for scheduling.

parent technique to improve performance under the assumption that nodes at higher levels in the communication tree have more than a single neighbor and thus can have more than a single parent. In contrast to each of these protocols, DW-MAC supports arbitrary communication between any nodes, whether to a sink node or to the other peer nodes such as to facilitate in-network processing of sensor data. The fast path algorithm proposed by Li et al. [14] also supports arbitrary communication patterns but assumes that such "fast paths" are long-lived and are set up through the routing protocol; DW-MAC makes no such assumptions and supports arbitrary communication between nodes at any time without relying on other protocols for assistance.

RMAC [5] represents a different approach to reducing latency in multihop forwarding; an example of the operation of RMAC is illustrated in Figure 2. In RMAC, a control frame, called a Pioneer frame (PION), is forwarded over multiple hops (e.g., $A \rightarrow B \rightarrow C$) during a Data period in order to inform nodes B and C when to wake up during the Sleep period to receive or transmit the corresponding data packet. The number of hops over which RMAC can forward a data packet during an operational cycle is limited by the duration of the Data period but may be set to any value depending on the parameters used. However, as a source node always starts transmitting a data packet at the beginning of a Sleep period (e.g., node A in Figure 2), two hidden sources that have succeeded in scheduling through PIONs in a Data period always cause collisions at the beginning of the next Sleep period. In addition, a node waken up due to a previous PION will wake up unnecessarily if the expected data packet cannot arrive due to collisions at previous hops.

The scheduling mechanism in DW-MAC ensures that data transmissions do not collide at their intended receivers, and many other techniques for collision-free transmission in WSNs have been studied by others (e.g., [12, 22]). However, in contrast to these techniques, DW-MAC is a contention-based protocol that integrates medium access control and scheduling seamlessly. Furthermore, DW-MAC supports not only unicast communication but also broadcast communication. Many other techniques for efficient broadcast communication in wireless sensor networks and in wireless ad hoc networks have been studied (e.g., [7,19,19,21,24,25]). However, in contrast to these techniques, a node in DW-MAC wakes up on demand during a Sleep period; scheduling frames during the Data period explicitly coordinate nodes when to wake up during the Sleep period to transmit or receive a packet.

3. DW-MAC DESIGN

3.1. Overview

DW-MAC is a synchronized duty cycle MAC protocol, where each cycle is divided into three periods: Sync, Data, and Sleep (Figure 3). We denote the duration of each period by T_{Sync}, T_{Data}, and T_{Sleep}, respectively. Similar to prior work, DW-MAC assumes that a separate protocol (e.g., [6,9]) is used to synchronize the clocks in

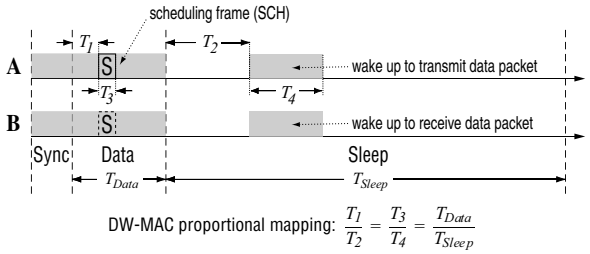


Figure 3: Overview of scheduling in DW-MAC.

sensor nodes during the Sync period with required precision. The basic concept of DW-MAC is to wake up nodes on demand during the Sleep period of a cycle in order to transmit or receive a packet. This demand wakeup adaptively increases effective channel capacity during a cycle as traffic load increases, allowing DW-MAC to achieve low delivery latency under a wide range of traffic loads including both unicast and broadcast traffic.

DW-MAC is unique in the way it schedules nodes to wake up during the Sleep period of a cycle. In DW-MAC, medium access control and scheduling are fully integrated. In a Data period, a node with pending data contends for channel access using a CSMA/CA protocol as in IEEE 802.11. DW-MAC, however, replaces RTS/CTS with a special frame called a *scheduling frame* (SCH). The interval of time during which the transmission of a SCH occupies the wireless medium automatically and uniquely reserves the proportional interval of time in the following Sleep period for transmitting and receiving the pending data packet. Essentially, DW-MAC sets up a one-to-one *mapping* between a Data period and the following Sleep period. An SCH carries no timing information, and the transmission of an SCH simply replaces that of RTS/CTS for medium access control. In this way, DW-MAC minimizes scheduling overhead. As in an RTS, an SCH contains the destination address, so this SCH wakes up only the intended receiver, minimizing energy consumption due to unnecessary wake-ups. Furthermore, this integration ensures that data transmissions do not collide at their intended receivers as discussed below.

Figure 3 shows an overview of scheduling in DW-MAC based on this one-to-one mapping between a Data period and the following Sleep period. In this example, node A wants to transmit a data packet to node B. Node A first contends for channel access and transmits an SCH during the Data period. Suppose transmission of the SCH starts T_1 time units after the beginning of the Data period. Based on T_1 and the duration of the SCH transmission, T_3, nodes A and B will both schedule their wakeup time to T_2 from the beginning of the following Sleep period, and will agree on a maximum wakeup duration of T_4, based on the ratio between T_{Data} and T_{Sleep}, as shown in the figure. If the packet to be transmitted is a unicast packet, node B will return a confirmation SCH frame (not in the figure) SIFS delay after receiving the request SCH from A; if the packet is a broadcast packet, node B takes no further action. When nodes A and B both wake up at the agreed time, node A transmits the actual data packet, which can be either broadcast or unicast. In case of unicast packet, node B acknowledges the successful receipt of the packet with an ACK. Although we show the scheduling for only one pair of nodes in this example, DW-MAC allows multiple contending nodes to exchange SCH frames with their intended receivers during a Data period, so that multiple data transmissions can happen in the following Sleep period.

3.2. Mapping Function for Scheduling

As previously explained, DW-MAC exploits a contention based Data period in order to schedule actual data transmissions during the subsequent Sleep period. To avoid collisions during the Sleep period, a sender must coordinate with its intended receiver to find a period of time in the Sleep period during which the neighboring nodes of both are idle. The challenges in designing such a protocol are twofold:

- minimize message exchanges between a sender, the intended receiver, and their respective neighbors for schedule negotiation; and
- minimize the size of a scheduling frame, e.g., avoid carrying timing information in a scheduling frame.

DW-MAC meets these challenges by employing a one-to-one *proportional mapping* function between time during a Data period and time during the subsequent Sleep period. With this mapping function, DW-MAC schedules data transmissions without exchanging any timing information. In order to further understand the Data to Sleep period mapping, we denote by T_i^D the time difference between a specific time instance t_i in a Data period and the beginning of that Data period. We further denote by T_i^S the time difference between the start of the subsequent Sleep period and the corresponding mapped time instance during the Sleep period. Accordingly DW-MAC defines the following mapping function:

$$T_i^S = T_i^D \cdot \frac{T_{Sleep}}{T_{Data}} \quad (1)$$

By mapping each time instant in a Data period into the subsequent Sleep period, the mapping function scales the time based on the ratio between T_{Sleep} and T_{Data}, and hence a time interval of T_1 time units in the Data period will be mapped into $T_1 \cdot \frac{T_{Sleep}}{T_{Data}}$ time units during the Sleep period. With this mapping function, a sender and its intended receiver(s) can uniquely determine the starting point for data packet transmission in a Sleep period from the starting time of the corresponding SCH transmission during the previous Data period, without including even a single bit of timing information in the SCH. Furthermore, the difference between the mapped beginning and end of the SCH transmission determines the maximum data transmission time. We can state the following proposition for DW-MAC:

PROPOSITION: *Any receiver that wakes up in a Sleep period is never in range of two simultaneous data packet transmissions, i.e., data transmissions by nodes that wake up during the Sleep period do not collide at their intended receivers.*

PROOF. We show this by contradiction. Assume that two data transmissions could collide. In order for data transmissions to collide at a node, they must overlap with each other. Therefore, the respective SCHs should also overlap at that node during the previous Data period. In this case, that node could not have decoded any SCH and thus would not wake up during the Sleep period, which contradicts the assumption. □

Note that the proposition only relates to collisions between data packets. We emphasize that a collision between a data packet and an ACK is still possible. This collision can be easily avoided by delaying the ACK to the mapped start time of the confirmation SCH sent from the node that transmits this ACK. However, such data-ACK collision is a rare event which requires very specific topology and timing setup between the nodes involved in the collision. In our implementation, we require a receiver to immediately acknowledge a data packet, so that both the sender and the receiver could go to sleep immediately and avoid wasting energy waiting for the delayed ACK.

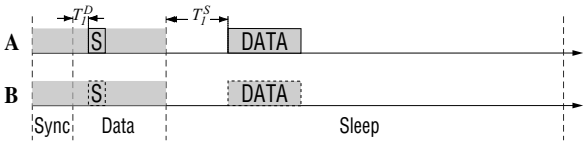


Figure 4: Broadcast in DW-MAC.

3.3. Scheduling Frame (SCH)

In addition to the fields included in an RTS/CTS, such as sender and receiver addresses, and duration of the transmission, an SCH also includes some cross-layer information. For a broadcast packet, an SCH includes the network layer address of its source and its sequence number. This information helps a node to decides whether the incoming broadcast packet has been received before or not, in order to avoid waking up to receive copies of the same packet multiple times. For a unicast packet, an SCH includes the network layer address of its final destination. This cross-layer information *enables a node to set up a schedule to the next hop neighbor before even receiving the actual data packet*, as discussed in Section 3.5.

An SCH serves either as a scheduling request or a scheduling confirmation. For a multihop forwarding, an intermediate node sends a single SCH serving both purposes: first, this SCH confirms the received SCH from the upstream node, and second, it schedules the forwarding of the eventual data packet to the next downstream node. In order to allow a single SCH to serve either or both of these two uses, an SCH includes two bits in the header to indicate which role(s) it is playing. As an access control frame in S-MAC is 10 bytes [26] and the address of a node usually takes two bytes [13], 14 bytes is the size for an SCH to hold the additional cross-layer information in our simulation.

3.4. Broadcast and Unicast in DW-MAC

DW-MAC supports two modes of operation: unicast traffic and broadcast traffic.

Broadcast of a data packet in DW-MAC is illustrated in Figure 4. After successfully transmitting an SCH, a sender (node A) starts broadcasting the packet at the time calculated based on the mapping function (Equation 1), T_1^S in this example. Based on the source address and the sequence number of the packet, which are included in the SCH, each receiver decides whether it has received the packet before. In case the packet has already been received by this node, the SCH is ignored. Otherwise, the receiver sets up a wakeup time for itself for receiving the corresponding data packet. In this example, node B estimates T_1^D based on when the SCH is received and its transmission delay. Using the mapping function, node B sets up a timer to wake up at T_1^S after the beginning of the Sleep period. Note that node B can contend for another SCH transmission and schedule the rebroadcast of the eventual incoming data packet even though it has not yet received the packet.

In the case of unicast traffic, a sender transmits an SCH prior to data transmission as it does for a broadcast packet. However, DW-MAC requires the intended receiver of the data packet to send back another SCH, SIFS after the receipt of the SCH, to confirm the receipt of that SCH. If the confirmation is received in time, the sender sets up a wakeup time for itself for data transmission. Otherwise, the sender attempts to transmit another SCH later, equivalent to a retransmission of an RTS. After receipt of a data packet, the receiver returns an ACK to the sender, SIFS after the completion of the data. As with the confirmation SCH, if the sender does not receive the ACK, it attempts to retransmit the data again (begin-

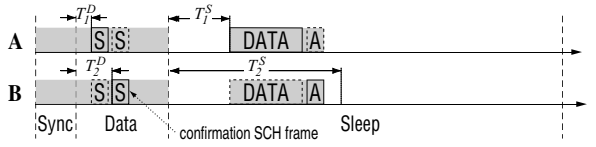


Figure 5: Unicast in DW-MAC.

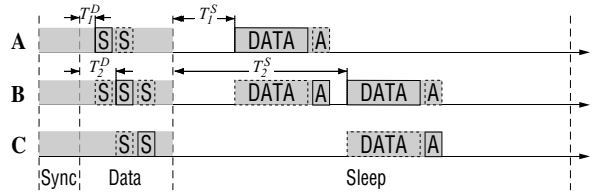


Figure 6: Optimized multihop forwarding of a unicast packet. Node B sends an SCH to wake up node C at the time indicated by T_2^s and confirms the SCH received from node A.

ning with a new SCH) during a later operational cycle. Figure 5 illustrates how node A transmits a unicast packet to node B.

If a sender has more data packets (either broadcast or unicast) than it is able to send during one operational cycle, those packets are simply queued and are attempted during the next cycle.

3.5. Optimized Multihop Forwarding

DW-MAC optimizes the timing of transmitting SCH frames in order to maximize the number of hops either a unicast or a broadcast packet can traverse during a cycle. Figure 6 illustrates the optimized multihop forwarding of a unicast packet. In this example, node A first sends an SCH to node B in order to set up a schedule for a pending data packet with final destination of node C. The SCH contains the network layer address of the final destination C. Upon receiving this SCH, node B calculates the wake up time T_1^S and checks the network layer destination in the SCH. Based on information from the routing layer (e.g., as is done in RMAC), node B will find that C is the next hop for the incoming packet. In this case, node B sends another SCH, SIFS after receiving the SCH from A. This SCH not only confirms the SCH just received from A but also wakes up C at the time indicated by T_2^S (both bits described in Section 3.3 in the header of the SCH are set, indicating that this SCH is serving both roles). In this way, a unicast packet can traverse x hops by only using $x+1$ SCH frames in a cycle, and the gap between two consecutive SCHs is just SIFS, which suggests more SCH exchanges in a Data period and more data transmissions in a cycle. Multihop forwarding in a similar manner is also supported by RMAC. However, DW-MAC dramatically reduce the collisions experienced by RMAC due to schedule conflicts, as DW-MAC ensures that two data frame transmissions will not collide with each other.

DW-MAC can also speed the propagation of a broadcast packet when some neighbor information is available. The main idea is to favor the rebroadcast of a broadcast packet along some path in order to shorten delays between rebroadcasts and to improve spatial reuse. In the SCH a node transmits, the node specifies an *immediate forwarder* that rebroadcasts the SCH SIFS after receiving the SCH. In the example illustrated in Figure 7, node A is specified as the immediate forwarder by node B. Any node other than the immediate forwarder (e.g., node C) backoffs before rebroadcasting the SCH. Nodes A and C will specify an immediate forwarder other than B in the SCH they rebroadcast, respectively. This optimized forwarding makes it possible for an SCH and thus the corresponding data packet to reach further nodes in a single cycle than having all rebroadcasting nodes compete for the medium equally. Although this reduced randomness could increase collision probability, the improved spatial reuse usually offsets this increase or even lowers total collision probability, as we reveal in our evaluation in Section 4. Many criteria can be used for choosing an immediate forwarder, such as location, degree, or the number of children nodes of a neighbor. In our DW-MAC simulation, this optimized forwarding is used when a broadcast tree of a WSN is available and a node knows its children nodes' height (the number of edges on the longest downward path to a leaf). For an SCH to be rebroadcast, if the SCH is received from the parent node, a node chooses the child with greatest height as the immediate forwarder. If this SCH is received from one child node, the parent node of the SCH receiver is chosen as the immediate forwarder.

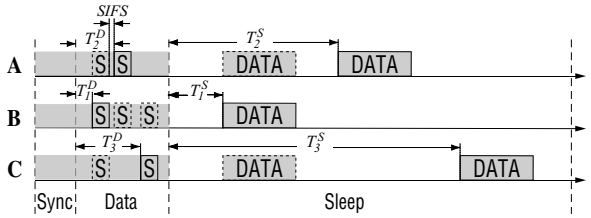


Figure 7: Optimized multihop forwarding of a broadcast packet. Node B specifies node A as the immediate forwarder, which rebroadcasts an SCH SIFS after receiving that SCH from A. Node C rebroadcasts the SCH when its backoff counter expires.

3.6. Implementation Issues

We choose to put a packet size limit in our DW-MAC implementation, although DW-MAC can support larger packet sizes by increasing the size of SCH frames, or by using variable SCH frame sizes for variable packet sizes. This design choice is based on the fact that popular sensor radios usually have a packet size limit. For example, CC1000 in Mica2 [13] and CC2420 in MicaZ [17] have a packet size limit of 256 and 128 bytes, respectively. With a low duty cycle configuration such as is common (and as is used in our simulations), a small SCH can be mapped to a period long enough for these packet limits.

Wake up times calculated at the sender and receiver(s) are not necessarily perfectly aligned due to propagation delay and processing time. However DW-MAC does not require an accurate estimation of the start of a transmission. DW-MAC needs only to ensure that a receiver wakes up early enough during a Sleep period so that an incoming packet is not missed, which can be ensured by wakening a receiver ε time before an estimated arrival time.

In our design of DW-MAC, we currently assume all nodes across the network are able to synchronize their clocks during the Sync period. This assumption, however, could be relaxed by only requiring all nodes within different network regions (e.g., clusters) to synchronize with each other, where the clocks between nodes in different regions need not be synchronized. Nodes on (or near) the boundary between one region and another would then need to be aware of the schedules within each region that they border. Since DW-MAC is compatible with S-MAC, boundary nodes could also fall back to using S-MAC.

At extremely low duty cycles, the proportional time for a data packet based on the size of an SCH frame, as indicated by our

Table 1: Networking Parameters

Bandwidth	20 Kbps	Channel Encoding Ratio	2
Tx Power	31.2 mW	Tx Range	250 m
Rx Power	22.2 mW	Carrier Sensing Range	550 m
Idle Power	22.2 mW	Contention Window (*CW*)	64 ms
Sleep Power	3 μW	Size of RTS/CTS/ACK	10 B
SIFS	5 ms	Size of SCH/PION	14 B
DIFS	10 ms	Size of Data	100 B
Retry Limit	5	State Transition Power	31.2 mW
Duty Cycle	5 %	State Transition Time	2.47 ms

mapping function (Equation 1), can become unnecessarily long due to the very large ratio of T_{Data} to T_{Sleep}. The sender and receiver nodes, however, are not required to remain awake for entire time indicated for a data packet. In particular, once the receiver completes receipt of the data packet and has returned the ACK, it may turn off its radio and go to sleep if desired, saving energy; likewise with the sender, once it receives the ACK. It is also possible to divide the Sleep period into two smaller periods, such that our mapping function maps time within the Data period proportionally only to time within the first portion of the Sleep period; during the second portion of the Sleep period, all nodes would only sleep. By choosing the size of these two portions of the Sleep period, the duration of the proportional time for a data packet based on the size of an SCH can be bounded, regardless of the overall duty cycle. For example, with duty cycles of a fraction of a percent, nodes could remain asleep during the entire (long) second portion of the Sleep period.

4. SIMULATION EVALUATION

We evaluated DW-MAC using version 2.29 of the *ns-2* simulator, under both unicast and broadcast traffic. Under unicast traffic, we compared DW-MAC against S-MAC, S-MAC with adaptive listening, and RMAC. Under broadcast traffic, because broadcast is not supported in S-MAC with adaptive listening or in RMAC, we compared DW-MAC only against S-MAC, in which a broadcast packet is transmitted during a Data period without using RTS/CTS [27].

Table 1 summarizes the key networking parameters used in our simulations. In our simulations, each sensor node has a single omni-directional antenna, and we use the common *ns-2* combined free space and two-ray ground reflection radio propagation model. Except for the parameters on radio power consumption above, which are typical values for Mica2 radios (CC1000) [28], we used the default settings in the standard S-MAC simulation module distributed with the *ns-2.29* package, also used for evaluations of S-MAC and RMAC in previous work [5]. The transition time of the CC1000 radio between sleep and active states is around 2.47 ms [3], but the state transition power is not available in the data sheet. Although the state transition power is normally much lower than Tx or Rx power, in order not to favor DW-MAC, which requires more state transitions than S-MAC in this aspect, we set the state transition power to the same value of Tx power. We observed similar trends in our results even if the state transition power is 0. In evaluating power efficiency, we focus on energy consumed by radios but ignore energy consumed by other components such as CPU and memory [23]. The transmission range and the carrier sensing range are modeled after the 914MHz Lucent WaveLAN DSSS radio interface, which is not typical for a sensor node, but we use these parameters to make our results comparable to those reported in previous work, and since measurements have shown that similar proportions of the carrier sensing range to the transmission range are also observed in some state-of-art sensor nodes [1].

Table 2: Duty Cycle Configuration

	T_{Sync} (ms)	T_{Data} (ms)	T_{Sleep} (ms)	T_{cycle} (ms)
S-MAC	55.2	104.0	3025.8	3185.0
RMAC	55.2	168.0	4241.8	4465.0
DW-MAC	55.2	168.0	4241.8	4465.0

In our simulations, we keep the same duty cycle of 5% for S-MAC, RMAC, and DW-MAC. The durations for the Sync, Data, and Sleep periods we used are shown in Table 2. For DW-MAC, we use the same duty cycle-related parameters that were used in the evaluation of RMAC in [5] for generating comparable results.

The data packet size used in our simulations was 100 bytes, although a maximum packet size of 256 bytes is supported by the CC1000 radios [13] and by the parameters used in our simulations.

To simplify our evaluations, we do not include routing traffic in the simulations and assume that there is a routing protocol deployed to provide the shortest path between any two nodes. We also ensure that every network we used in our simulations is a connected network. In addition, we do not include any synchronization traffic and assume all the nodes in the network have already been synchronized to use a single wake-up and sleep schedule.

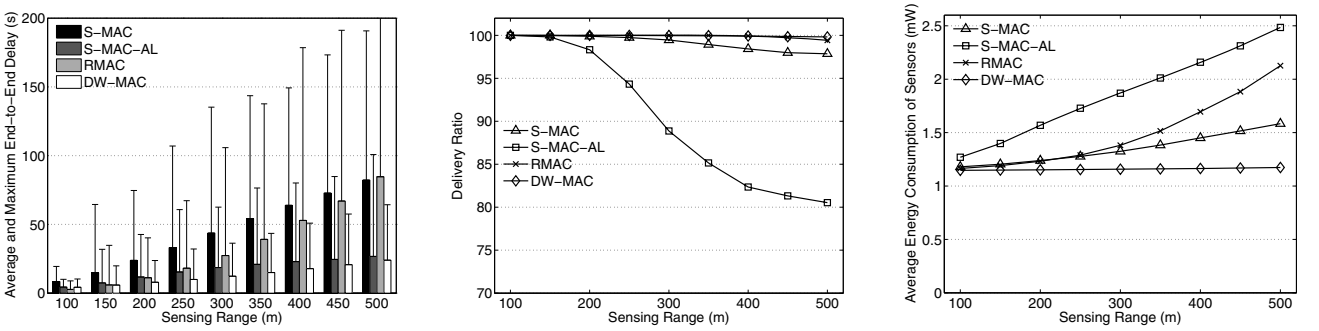
For simulations under unicast traffic, each run contains unicast packets toward a sink node that are triggered by a series of 500 events, and each average value is calculated from the results of 10 random runs. For simulations under broadcast traffic, each run contains 500 broadcast packets generated by a sink node, and each average value is calculated from the results of 30 random runs. Confidence intervals of the average values are not shown because even 99% confidence intervals are so close to average values that they overlap with the data point markers.

4.1. Evaluation under Unicast Traffic

We compare DW-MAC with S-MAC, S-MAC with adaptive listening (shown as S-MAC-AL in all figures), and RMAC both in a 49-node (7×7) grid network and in random networks.

In the grid network, each node is 200 meters from its neighbors, and the sink node is at the center. Based on a correlated-event workload [10], we introduce a Random Correlated-Event (RCE) traffic model to simulate the impulse traffic triggered by spatially-correlated events commonly observed in detection and tracking applications. RCE picks a random (x,y) location for each event. If every node has a sensing range R, only nodes that are within the circle centered at (x,y) with radius R generate packets to report this event. We adjust the sensing rage R to simulate different degrees of workload in a network. In our experiments, a new event is generated once every 200 seconds, and each node having sensed the event sends one packet to the sink node. We vary R from 100 meters to 500 meters; the average number of packets generated per event is listed in Table 3. Note that an event triggers at most one packet when R is 100 meters. The lengths of paths traversed by these packets range from 1 to 6 hops, and the average is 3.05. In this way, we explore how efficiently S-MAC, S-MAC with adaptive listening, RMAC, and DW-MAC handle different degrees of traffic load. The performance of these protocols for unicast traffic in the 49-node grid network scenarios is shown in Figure 8.

Figure 8(a) shows the average and maximum end-to-end latency of packets in the RCE model as the sensing range (and thus traffic load) increases. DW-MAC has a much smaller rate of increase than do S-MAC and RMAC. When there are around 15 packets gener-



(a) Average and maximum end-to-end delay versus sensing range.

(b) Delivery ratio versus sensing range.

(c) Average energy consumption of sensors versus sensing range.

Figure 8: Performance for unicast traffic in 49-node (7×7) grid network scenarios.

Table 3: Average number of packets generated for each event under different sensing ranges in the 49-node grid network

Range (m)	100	150	200	250	300	350	400	450	500
Packets	0.8	1.7	3.1	4.6	6.4	8.4	10.6	12.9	15.2

ated for each event with the 500-meter sensing range, DW-MAC reduces average end-to-end delay by around 70% compared to S-MAC and RMAC. DW-MAC outperforms S-MAC because DW-MAC allows more transmissions in a cycle by using the Sleep period for actual data transmissions. RMAC experiences more delay than DW-MAC as workload increases, because of increased packet collisions caused by scheduling conflicts. It is the retransmission effort to recover these collided packets that results in larger end-to-end delay. When the sensing range is 500 meters, the maximum end-to-end delay with RMAC is 374.95 seconds, which is off the top of the graph. This extreme delay occurs when a packet generated for one event failed to reach the sink before the next event happened. Under the light traffic with the 100-meter sensing range, DW-MAC shows slightly larger delay than RMAC, due to the time that a received data packet is forwarded to the next hop in multihop forwarding. In RMAC, a data packet is forwarded immediately, whereas in DW-MAC, forwarding starts at a later time determined by the corresponding SCH frame. This extra delay experienced by DW-MAC, however, is less than the duration of a Sleep period. S-MAC with adaptive listening shows slightly larger delay compared to DW-MAC. This low delay achieved by adaptive listening, however, comes at the cost of lower packet delivery ratio and increased energy consumption as shown next.

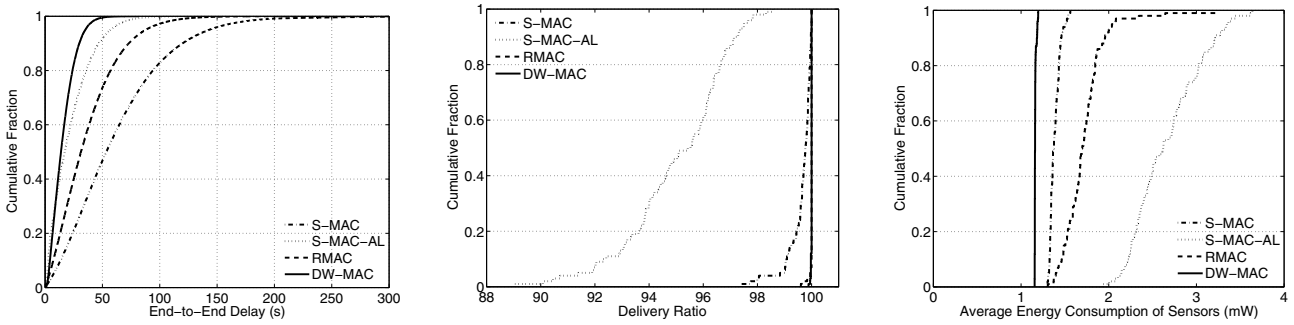
The packet delivery ratios corresponding to Figure 8(a) are shown in Figure 8(b). DW-MAC maintains close to 100% packet delivery ratio and outperforms the other protocols across all sensing ranges. The delivery ratio with S-MAC with adaptive listening drops quickly, since with larger the sensing ranges, more collisions are caused by transmissions from hidden nodes, as we discussed in Section 2; in addition, a node may transmit a packet when its intended receiver is in sleep state, further decreasing packet delivery ratio. DW-MAC and RMAC outperform S-MAC mainly for two reasons. First, they only transmit short scheduling frames during a Data period, avoiding collisions between a control frame and a long data frame. Second, a node does more retransmission attempts for a data packet in DW-MAC and RMAC. Specifically, a scheduling frame sent by an intermediate node in multihop forwarding serves both as RTS and as CTS; even if this frame fails to reach the next-hop neighbor, the intermediate node does not increase its retry count, as the node has not received the corresponding data packet yet, although the node has attempted to reserve the medium to forward the incoming data packet once. Even with such extra retransmission attempts, the delivery ratio of RMAC drops more quickly than that of DW-MAC beyond a 400-meter sensing range, as retransmissions are not enough to recover the increased collisions due to RMAC's scheduling conflicts.

Figure 8(c) shows the average energy consumption of nodes versus sensing ranges in the 49-node grid network scenarios. Under light workload, when the sensing range is 100 meters, all four MAC protocols show almost the same power consumption, but when traffic load increases as the sensing range gets larger, average energy consumption in all protocols except DW-MAC increases quickly (energy consumption for DW-MAC does increase, but increases very slowly). When the sensing range is 500 meters, DW-MAC consumes less than 50% of the energy consumed by S-MAC with adaptive listening to achieve even lower packet delivery latency.

We also compare S-MAC, S-MAC with adaptive listening, RMAC, and DW-MAC in 100 random networks, each with 50 nodes randomly located in a 1000 m $\times$ 1000 m area. For each network, one random node is chosen as the sink, and the RCE model with 250-meter sensing range is used to generate 500 events, once every 200 seconds. We conduct one simulation run for each network, and 3845 packets are generated in each run on average. The results are plotted in Figure 9. For the same reasons discusses above, DW-MAC outperforms the other three protocols in delivery latency, delivery ratio, and energy consumption. Figure 9(a) show the CDF of end-to-end latency for all packets in all 100 runs. Average end-to-end latency with S-MAC, S-MAC with adaptive listening, RMAC, and DW-MAC are 61.8%, 21.6%, 36.7%, and 15.7%, respectively. Although adaptive listening greatly reduces end-to-end latency for S-MAC, this gain is at the cost of lower delivery ratio and more energy consumption. Figure 9(b) shows the CDF of delivery ratios in these 100 runs. The average delivery ratios of S-MAC, S-MAC with adaptive listening, RMAC, and DW-MAC are 99.63%, 95.03%, 99.99%, and 99.99%, respectively. The average energy consumptions of the sensors are plotted in Figure 9(c), where the average values with S-MAC, S-MAC with adaptive listening, RMAC, and DW-MAC are 1.386, 2.666, 1.724, and 1.163 mW, respectively. The trends observed in these random networks are consistent with those observed in the 49-node grid network.

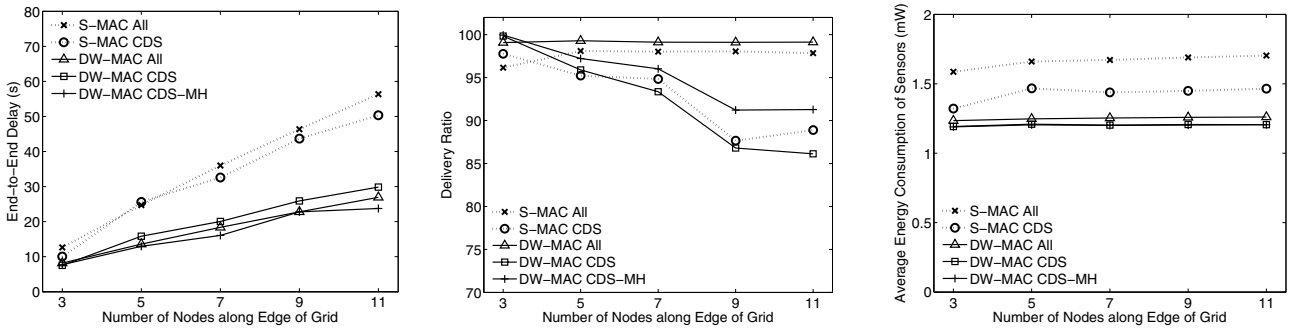
4.2. Evaluation under Broadcast Traffic

We compared DW-MAC with S-MAC, both in regular grid networks and in random networks, under broadcast traffic.



(a) CDF of end-to-end delays *(b)* CDF of delivery ratios *(c)* CDF of average energy consumption

Figure 9: Performance for random correlated-event traffic in 50-node networks with sensing range of 250 m.



(a) End-to-end delay versus network size. *(b)* Delivery ratio versus network size. *(c)* Average energy consumption of sensors versus network size.

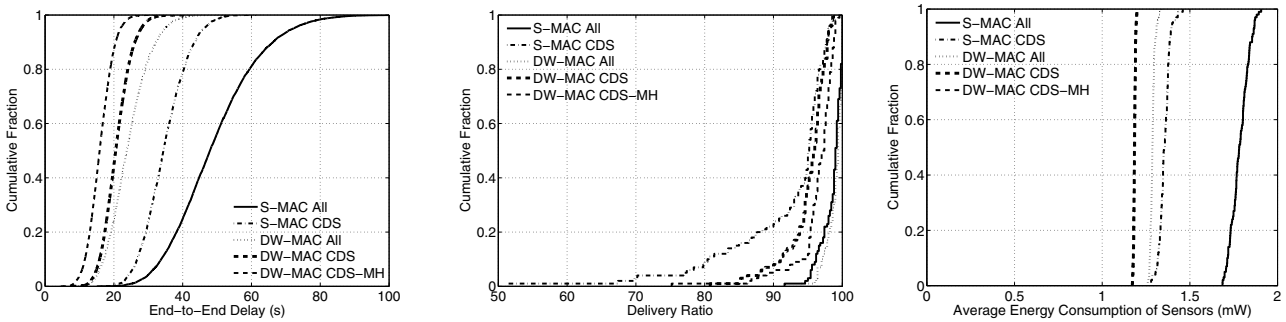
Figure 10: Performance for broadcast traffic in grid networks.

In the grid network, the sink node is at the center, and each node is 200 meters from its neighbors. We vary the grid size from 3×3 (9 nodes) to 11×11 (121 nodes). The sink node generates a broadcast packet once every 100 seconds so that transmissions for one packet complete before the next packet is generated. Due to space limits, we evaluate DW-MAC under only two categories of broadcast protocols: simple flooding (all nodes that have received a broadcast packet rebroadcast it exactly once, indicated by "ALL") and Connected Dominating Set (CDS) based flooding (only nodes in a CDS that have received a broadcast packet rebroadcast it exactly once, indicated by "CDS"). The CDS is formed by the algorithm by Gandhi et al. [8], with a slight modification to always include the sink node in the CDS; the results for our optimized multihop forwarding for broadcast traffic are indicated by "DW-MAC CDS-MH." Note that this CDS algorithm is designed to minimize broadcast latency, and the resulting CDS is not necessarily an minimum CDS.

Simulation results in grid networks are shown in Figure 10. Figure 10(a) shows end-to-end latency (the time it takes for the last node to receive a given broadcast packet) with S-MAC and DW-MAC. DW-MAC reduces the end-to-end latency by around 50% over those with S-MAC, as DW-MAC allows more contending nodes to finish their transmissions in each cycle. When optimized multihop forwarding is enabled, DW-MAC further reduces end-to-end latency, as it increases spatial reuse and reduces delays before a rebroadcast. An interesting trend is that CDS-based flooding shows lower latencies than simple flooding with S-MAC but shows the reverse with DW-MAC. The reason lies in the combination of CDS formation, grid topologies, and duty cycle configuration in our simulation. First, a CDS formed is not necessarily an MCDS. Second, a CDS node may experience more latency before rebroadcasting a packet than does a non-CDS node with DW-MAC, due to defers caused by undecodable frames. When a node fails to decode a received a packet, it defers for some time (such as EIFS in IEEE 802.11) to avoid interrupting ongoing transmission. Since this defer is much shorter than a Sleep period in our simulations, all neighboring nodes still compete for the medium fairly at the beginning of the next cycle with S-MAC. With DW-MAC, however, it is possible that a node is ready to rebroadcast a packet before its defer timer expires, as multiple SCHs can be transmitted during a Data period. A CDS node that defers could be slower in rebroadcasting a packet compared to a non-CDS node that does not defer, resulting in lower latency for DW-MAC All than for DW-MAC CDS. However, DW-MAC still reduces end-to-end latency by around 40% for CDS-based flooding compared to those with S-MAC.

Figure 10(b) shows the delivery ratios (the percent of broadcast packets that are successfully received by *all* nodes in a network) of flooding in the grid networks. Because of the increased redundancy in simple flooding, S-MAC and DW-MAC achieve higher delivery ratio than CDS-based flooding. In simple flooding, DW-MAC outperforms S-MAC, since the use of (short) SCH frames instead of long data packets during contention helps to avoid collisions. However, when CDS-based flooding is used, DW-MAC sometimes shows lower delivery ratios than does S-MAC, mainly due to the special grid topology and selection of CDS as discussed before. Looking at the results in random networks (Figure 11(b)), on average, DW-MAC shows better delivery ratios than S-MAC when CDS-based flooding is used. With improved spatial reuse when optimized multihop forwarding is used, DW-MAC achieves higher delivery ratios than does S-MAC in CDS-based flooding.

Average energy consumption in the grid networks, calculated as we did in evaluations under unicast traffic, is shown in Figure 10(c).



(a) CDF of end-to-end delays. *(b)* CDF of delivery ratios. *(c)* CDF of average energy consumption.

Figure 11: Performance for broadcast traffic in 50-node networks.

The interval between traffic bursts is changed from 200 seconds to 100 seconds to show the differences among protocols more clearly. DW-MAC reduces average energy consumption over S-MAC by about 26% under simple flooding and by about 18% under CDS-based flooding. DW-MAC achieves these savings by not overhearing data transmissions. In DW-MAC, a node only attempts to receive an incoming packet after receiving an SCH that indicates the packet has not been received. Simple flooding consumes more energy because of more rebroadcasts. Whether or not the optimized multihop forwarding is used, a flooding results in the same number of transmissions, so this optimization does not affect energy consumption much.

Finally, we compare these broadcast protocols in 100 random networks, the same networks used for evaluations under unicast traffic. The sink in each network generates 500 broadcast packets in each run, one packet every 100 seconds. Figure 11(a) shows the CDF of end-to-end latency for all packets in the 100 runs. All DW-MAC based broadcast protocols show much smaller end-to-end latency than those based on S-MAC. The average end-to-end latency for S-MAC ALL, S-MAC CDS, DW-MAC ALL, DW-MAC CDS and DW-MAC CDS-MH are 49.1, 34.8, 24.2, 20.8, and 16.0 seconds, respectively. On average, end-to-end latency is reduced by more than 50% both in simple flooding and in CDS-based flooding. Unlike the results in grid networks, DW-MAC shows lower average end-to-end latency in CDS-based flooding than those in simple flooding, because the speedup gained by fast propagation along CDS nodes is often greater than the slowdown caused by defers in these networks. For these 100 runs, the CDF of delivery ratios is shown in Figure 11(b), and the CDF of average energy consumption is shown in Figure 11(c). S-MAC ALL, S-MAC CDS, DW-MAC ALL, DW-MAC CDS, and DW-MAC CDS-MH, respectively, show average delivery ratios of 98.6%, 92.1%, 99.0%, 95.0% and 96.4% and average energy consumption of 1.785, 1.355, 1.288, 1,185, and 1.183 mW. The difference in energy consumption between DW-MAC CDS and DW-MAC CDS-MH is almost invisible because the optimized multihop forwarding does not affect the number of data transmissions much. Overall, DW-MAC achieves lower end-to-end delays, higher delivery ratios, and more energy savings for broadcast traffic in these random networks.

5. CONCLUSION

In this paper, we presented DW-MAC, a new energy efficient duty cycle MAC protocol designed to reduce packet delivery latency for a wide range of traffic loads, including both unicast and broadcast traffic. DW-MAC adaptively increases effective channel capacity during an operational cycle as traffic load increases, allowing DW-MAC to achieve low delivery latency under dynamic traffic loads. The scheduling algorithm in DW-MAC integrates scheduling and access control to maintain a proportional one-to-one mapping function between a Data period and the subsequent Sleep period, which minimizes scheduling overhead while ensuring that data transmissions do not collide at their intended receivers.

We compared DW-MAC with S-MAC (with and without adaptive listening) and with RMAC through extensive simulations. We found that DW-MAC outperforms these protocols, with lower latency, higher power efficiency, and higher packet delivery ratios, and with increasing benefits as traffic load increases. For example, under high unicast traffic load, DW-MAC reduces delivery latency by 70% compared to S-MAC and RMAC, and uses only 50% of the energy consumed with S-MAC with adaptive listening. Under broadcast traffic, DW-MAC reduces latency by more than 50% on average, always reducing energy consumption by more than 15%. In addition, DW-MAC improves packet delivery ratios under all scenarios in our simulations.

ACKNOWLEDGEMENTS

We thank Jingpu Shi for valuable discussions during the design of DW-MAC. This work was supported in part by the U.S. National Science Foundation under grants CNS-0520280, CNS-0435425, CNS-0338856, and CNS-0325971; and by a gift from Schlumberger. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of NSF, Schlumberger, Rice University, Ben Gurion University, or the U.S. Government or any of its agencies.

REFERENCES

[1] G. Anastasi, A. Falchi, A. Passarella, M. Conti, and E. Gregori. Performance Measurements of Motes Sensor Networks. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2004)*, pages 174–181, October 2004.

[2] Qing Cao, Tarek Abdelzaher, Tian He, and John Stankovic. Towards Optimal Sleep Scheduling in Sensor Networks for Rare-Event Detection. In *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, pages 20–27, April 2005.

[3] Chipcon. Single Chip Very Low Power RF Transceiver (CC1000 Datasheet), April 2002.

[4] Tijs van Dam and Koen Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In

Proceedings of the First International Conference On Embedded Networked Sensor Systems (SenSys 2003), pages 171–180, November 2003.

[5] Shu Du, Amit Kumar Saha, and David B. Johnson. RMAC: A Routing-Enhanced Duty-Cycle MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 26th Annual IEEE Conference on Computer Communications (INFOCOM 2007)*, pages 1478–1486, May 2007.

[6] Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-Grained Network Time Synchronization using Reference Broadcasts. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, pages 147–163, December 2002.

[7] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next Century Challenges: Scalable Coordination in Sensor Networks. In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networking (MobiCom 1999)*, pages 263–270, August 1999.

[8] Rajiv Gandhi, Srinivasan Parthasarathy, and Arunesh Mishra. Minimizing Broadcast Latency and Redundancy in Ad Hoc Networks. In *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, pages 222–232, June 2003.

[9] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. Timing-Sync Protocol for Sensor Networks. In *Proceedings of the First International Conference On Embedded Networked Sensor Systems (SenSys 2003)*, pages 138–149, November 2003.

[10] Bret Hull, Kyle Jamieson, and Hari Balakrishnan. Mitigating Congestion in Wireless Sensor Networks. In *Proceedings of the Second International Conference On Embedded Networked Sensor Systems (SenSys 2004)*, pages 134–147, November 2004.

[11] Abtin Keshavarzian, Huang Lee, and Lakshmi Venkatraman. Wakeup Scheduling in Wireless Sensor Networks. In *Proceedings of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2006)*, pages 322–333, May 2006.

[12] Sandeep S. Kulkarni and Mahesh Arumugam. TDMA Service for Sensor Networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW 2004)*, pages 604–609, March 2004.

[13] Philip Levis. TEP 111: message_t. TinyOS 2.0 Documentation, http://www.tinyos.net/tinyos-2.x/doc/.

[14] Yuan Li, Wei Ye, and John Heidemann. Energy and Latency Control in Low Duty Cycle MAC Protocols. In *Proceedings of the 2005 IEEE Wireless Communications and Networking Conference (WCNC 2005)*, March 2005.

[15] Gang Lu, Bhaskar Krishnamachari, and Cauligi S. Raghavendra. An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks. In *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*, April 2004.

[16] Gang Lu, Narayanan Sadagopan, Bhaskar Krishnamachari, and Ashish Goel. Delay Efficient Sleep Scheduling in Wireless Sensor Networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, pages 2470–2481, March 2005.

[17] David Moss, Jonathan Hui, Philip Levis, and Jung Il Choi. TEP 126: CC2420 Radio Stack. http://www.tinyos.net/tinyos-2.x/doc/, 2007.

[18] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networking (MobiCom 1999)*, pages 151–162, August 1999.

[19] Stefan Pleisch, Mahesh Balakrishnan, Ken Birman, and Robbert van Renesse. MISTRAL: Efficient Flooding in Mobile Ad-Hoc Networks. In *Proceedings of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2006)*, pages 1–12, May 2006.

[20] Joseph Polastre, Jason Hill, and David Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *Proceedings of the Second International Conference On Embedded Networked Sensor Systems (SenSys 2004)*, pages 95–107, November 2004.

[21] Nazanin Rahnavard and Faramarz Fekri. CRBcast: A Collaborative Rateless Scheme for Reliable and Energy-Efficient Broadcasting in Wireless Sensor Networks. In *Proceedings of the Fifth International Conference on Information Processing in Sensor Networks (IPSN 2006)*, pages 276–283, April 2006.

[22] Venkatesh Rajendran, Katia Obraczka, and J. J. Garcia-Luna-Aceves. Energy-Efficient Collision-Free Medium Access Control for Wireless Sensor Networks. In *Proceedings of the First International Conference On Embedded Networked Sensor Systems (SenSys 2003)*, pages 181–192, November 2003.

[23] Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the Power Consumption of Large-Scale Sensor Network Applications. In *Proceedings of the Second International Conference On Embedded Networked Sensor Systems (SenSys 2004)*, pages 188–200, November 2003.

[24] Fred Stann, John Heidemann, Rajesh Shroff, and Muhammad Zaki Murtaza. RBP: Robust Broadcast Propagation in Wireless Networks. In *Proceedings of the Fourth International Conference On Embedded Networked Sensor Systems (SenSys 2006)*, pages 85–98, October 2006.

[25] Brad Williams and Tracy Camp. Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks. In *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002)*, pages 194–205, June 2002.

[26] Wei Ye, John Heidemann, and Deborah Estrin. Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, 2004.

[27] Wei Ye, John S. Heidemann, and Deborah Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, pages 1567–1576, June 2002.

[28] Wei Ye, Fabio Silva, and John Heidemann. Ultra-Low Duty Cycle MAC with Scheduled Channel Polling. In *Proceedings of the Fourth International Conference On Embedded Networked Sensor Systems (SenSys 2006)*, pages 321–334, October 2006.

[29] Hongwei Zhang, Anish Arora, Young-ri Choi, and Mohamed G. Gouda. Reliable Bursty Convergecast in Wireless Sensor Networks. In *Proceedings of the Sixth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2005)*, pages 266–276, May 2005.