EMBA: An Efficient Multihop Broadcast Protocol for Asynchronous Duty-Cycled Wireless Sensor Networks

Ingook Jang, Suho Yang, Hyunsoo Yoon, *Member, IEEE,* and Dongwook Kim

***Abstract*—In this paper, we propose an efficient multihop broadcast protocol for asynchronous duty-cycled wireless sensor networks (EMBA) where each node independently wakes up according to its own schedule. EMBA adopts two techniques of *the forwarder's guidance* and *the overhearing of broadcast messages and ACKs*. A node transmits broadcast messages with *guidance* to neighbor nodes. The guidance presents how the node forwards the broadcast message to neighbor nodes by using unicast transmissions. This technique significantly reduces redundant transmissions and collisions. The overhearing of broadcast messages and ACKs helps to reduce the number of transmissions, thus it minimizes the active time of nodes. We implement EMBA and conventional protocols of ADB and RI-MAC broadcast in *ns-2* simulator to compare their performance. The simulation results show that EMBA outperforms ADB and RI-MAC broadcast in both sparse and dense networks. EMBA achieves lower message cost than the conventional protocols and significantly improves the energy efficiency in terms of both duty cycle and energy consumption.**

***Index Terms*—Wireless sensor networks, multihop broadcast, asynchronous duty-cycling, *ns-2*.**

I. INTRODUCTION

THE goal of wireless sensor networks (WSNs) is to reliably report sensing data to the sink. A broad class of WSN applications necessitates energy efficiency [1]–[5]. Nodes in such WSN applications should operate unattended for a long time on limited battery capacity [7].

Most sleep scheduling protocols today in WSNs utilize *duty-cycling* which allows a sensor node to alternate between active and sleeping states to reduce energy consumption. The duty-cycled MAC protocols in the previous literature are generally divided into two categories: *synchronous* and *asynchronous*. Synchronous sleep scheduling approaches synchronize neighboring nodes before data transmissions to reduce energy consumption. In S-MAC [12] and T-MAC [13], nodes in the vicinity synchronize together and form a *virtual cluster* with the same schedules. Each node in the cluster communicates with its neighbors only within common active states. RMAC [6] uses a special message called as a *Pioneer frame* (PION), which is forwarded over multiple hops during an operational cycle in order to inform nodes along the route when to wake up to receive data. Synchronous MAC protocols achieve comparable energy efficiency by reducing idle listening time, but extra complexity and overhead are required for synchronization.

In asynchronous approaches, each sensor node wakes up and operates independently according to its own duty cycle schedule. Such protocols use *low power listening* (LPL) that is one of duty-cycling techniques, where each node periodically samples the channel for a long *preamble* continuously transmitted by a sender. LPL reduces energy consumption by turning off the radio between samples. A sender, in B-MAC [14], starts to transmit data after sending a long preamble which lasts at least as long as a sleep period of a receiver. When the receiver wakes up and detects the preamble, it stays awake to receive data following the preamble. However, a node may unnecessarily stay awake to receive data destined to other nodes. To solve this problem, X-MAC [15] replaces a long preamble with a bunch of sequential short preambles. Each preamble contains the target address and it allows nodes not involved in the communication to go to sleep immediately. B-MAC and X-MAC achieve high energy efficiency in light traffic loads, but busty traffic makes them less efficient due to the high occupancy of the wireless medium. WiseMAC [17] allows a node to adaptively reduce the length of the preamble based on schedules of its neighbor nodes. For this, the protocol requires each node to implicitly synchronize with its neighbors by learning the schedules of them.

Receiver-initiated approaches such as RI-MAC [7] and Koala system [8] are recently introduced. In RI-MAC, a sender that has data waits for a *beacon* (base beacon) from an intended receiver. After the intended receiver wakes up and sends a beacon as an invitation of a transmission, the sender transmits data upon receiving the beacon. If the receiver successfully receives data, it will transmit an ACK beacon that plays roles as both an acknowledgement and a new invitation. RI-MAC significantly reduces the amount of time which a pair of a sender and a receiver occupies the wireless medium for a data transmission. Consequently, this protocol improves the

Manuscript received April 1, 2012; revised September 1 and November 13, 2012; accepted December 27, 2012. The associate editor coordinating the review of this paper and approving it for publication was A. Vosoughi.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MEST) (No. 2012-0005390).

I. Jang and H. Yoon are with the Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), 373-1, Guseong-dong, Yuseong-gu, Daejeon, 305-701, South Korea (e-mail: {ikjang, hyoon}@nslab.kaist.ac.kr).

S. Yang is with the Korea Financial Telecommunications and Clearings Institute (KFTC), 717 YokSam-Dong, KangNam-Gu, Seoul, 135-080, South Korea (e-mail: suho@kftc.or.kr).

D. Kim (corresponding author) is with the Network Business Division, Samsung Electronics, 416, Maetan 3-dong, Yeongtong-gu, Suwon-si, Gyeonggi-do, 443-742, South Korea (e-mail: kimdw@nslab.kaist.ac.kr).

Digital Object Identifier 10.1109/TWC.2013.022013.120477

performance in terms of energy consumption and delivery latency compared to B-MAC and X-MAC. Low Power Probing (LPP) is another receiver-initiated approach in Koala system designed for bulk data transfer applications. To download bulk data from all nodes in a network, a sink node initiates a whole-network wake up service. Each node periodically broadcasts short probes requesting acknowledgments from its neighbor nodes. If an ACK arrives, the node remains active in order to wake up other nodes. The goal of LPP is only to wake up every node, not to transfer actual data frames.

Multihop broadcast (MB) is to propagate a broadcast message to every node in a whole network. It facilitates sensor nodes to serve a wide range of important higher-level operations such as propagating queries across the whole network for data collection and disseminating control messages for network configuration. Broadcast is quite simple in synchronous sleep scheduling protocols because synchronized nodes wake up with the same schedules. They easily transmit and receive broadcast messages on their active schedules. Moreover, one broadcast transmission of a sender reaches multiple nodes. Therefore, propagation of a broadcast message is done with the small number of transmissions. These advantages of synchronization enable multihop broadcast to be easily implemented without incurring a large overhead or excessive cost.

Asynchronous sleep scheduling approaches, however, are inherently weak for supporting multihop broadcast due to independent wake up time of each sensor node. Every node wakes up according to its own duty cycle schedule, and it means that it is hard for one broadcast transmission to reach multiple neighbor nodes. To support multihop broadcast in asynchronous sleep scheduling approaches, a node should use independent unicast transmissions as many as the number of its neighbor nodes. This causes redundant transmissions of the same broadcast messages. In addition, collisions frequently occur when a node simultaneously receives the same broadcast messages from multiple senders. These vulnerabilities lead to significant energy dissipation and delayed propagation of broadcast messages. Therefore, the complexity of multihop broadcast in asynchronous environments is substantial. It is very challenging to design an efficient multihop broadcast protocol for asynchronous duty-cycled WSNs.

In this paper, we propose an efficient multihop broadcast protocol for WSNs (EMBA). EMBA operates on asynchronous MAC protocols and uses two techniques of *the forwarder's guidance* and *the overhearing of broadcast messages and ACKs*. The proposed protocol enables sensor nodes to efficiently support multihop broadcast by using collision and redundant transmission avoidance. EMBA achieves much higher energy efficiency and smaller message cost in both sparse and dense networks.

II. Related Works

The goal of multihop broadcast for WSNs is to efficiently and reliably deliver messages to every node in a network. Trickle [9] disseminates small code updates to all nodes to reprogram the entire network. RBP [10] reliably propagates small messages for routing and resource discovery, based on flooding. These protocols, however, do not consider whether nodes are active or not.

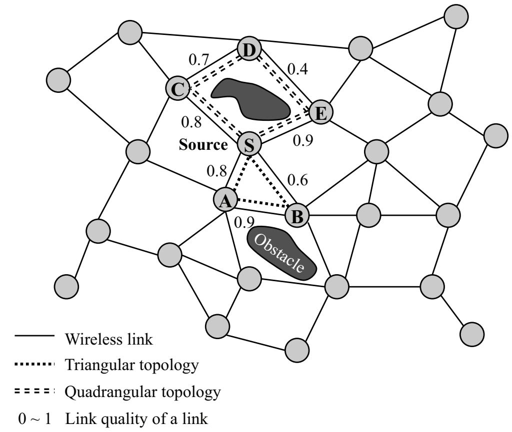


Fig. 1. A WSN which is composed of heterogeneous local topologies such as triangles, quadrangles, and so on.

DW-MAC [11] which is one of synchronous sleep scheduling protocols supports multihop broadcast by using multihop forwarding. An operational cycle in DW-MAC is divided into three parts: Sync, Data, and Sleep. Each node synchronizes its clock with its neighbor nodes during the Sync period. During the Data period, a sender that wants to broadcast transmits a *scheduling frame* (SCH) which indicates the starting point for the broadcast transmission that will be performed within a following Sleep period. The sender starts broadcasting a message at that point. Every node receiving the broadcast message becomes a new sender to forward it to other nodes. DW-MAC simply extends multihop forwarding without substantially increasing overhead in order to support multihop broadcast.

Multihop broadcast in asynchronous approaches is substantially complicated due to independent sleep schedules of nodes. X-MAC-UPMA [16] is the X-MAC implementation for the UPMA (Unified Power Management Architecture for Wireless Sensor Networks) package of TinyOS [23]. This supports broadcast by repeatedly transmitting duplicate copies of a message over a sleep interval. RI-MAC also supports broadcast in two ways, either by the way used in X-MAC-UPMA or by unicast transmissions. However, both schemes are very inefficient because a node receives multiple copies of the same message. These redundant transmissions cause frequent collisions, followed by unnecessary energy consumption. Therefore, multihop broadcast for asynchronous approaches should minimize redundant transmissions and collisions.

ADB [19] is recently proposed to support multihop broadcast for asynchronous duty-cycled sensor networks. ADB is designed based on RI-MAC. ADB adopts unicast and updates the receiver information on the broadcast progress, which enables each node to avoid redundant transmissions by allowing transmission delegating. In Figure 1, node S is a source node and node A wakes up earlier than node B. Upon receiving node A's beacon, node S sends A a broadcast message including an *ADB footer* that indicates the broadcast progress and the link quality information of S. Looking into the footer, node A recognizes that the quality of the link between nodes S and B is poorer than that of link between itself and B. Node A decides to forward the broadcast message to node B and inform node S of this fact by sending ACK with a new footer. Upon receiving this ACK, node S delegates handling of node B to node A.

However, ADB does not efficiently support multihop broadcast in polygonal topologies except triangle shapes. For example, in Figure 1, nodes S, C, E, and D compose a quadrangular topology where node C and node E are located out of each other's communication range. Node S sends a broadcast message including a footer to node C. However, the link quality between nodes S and E in the footer is useless to node C since C and E cannot communicate with each other. Similarly, node E also ignores the link quality between nodes S and C after receiving the broadcast message from S. Both nodes C and E stay active to forward the broadcast message to node D. After node D wakes up, they simultaneously forward the broadcast message to D and a collision occurs at D. Each of nodes C and E retransmits the collided message by using backoff. As a result, node D will receive the two identical broadcast messages after the collision. More seriously, redundant transmissions and collisions occur in all possible wake-up scenarios that can be formed by four nodes (in Figure1, S-C-D-E, S-C-E-D, S-E-C-D, and S-E-D-C). This problem will lead to the unnecessary energy dissipation and the reduced lifetime of the network. The frequency of triangles decreases as the network is getting sparser, which will be presented in Section III-A. In sparse networks, the ADB therefore would show the degraded performance in multihop broadcast.

The proposed EMBA efficiently supports the multihop broadcast in sparse networks by using (i) *the forwarder's guidance*. In addition, EMBA makes good use of (ii) *the overhearing of broadcast messages and ACKs* to minimize redundant transmissions. As a result, EMBA shows good performance even in dense networks. The details of these two techniques are presented in Section IV and, in Section V of performance evaluation, we will show that EMBA significantly outperforms ADB in both sparse and dense networks in terms of message cost and energy efficiency.

Multihop broadcast protocols in asynchronous WSNs are classified as two approaches: *proactive* and *reactive*. ADB is a reactive approach, which determines the link to forward in an on-demand manner. This protocol can cope with rapid and dynamic changes of link conditions. However, it is not suitable for networks with bursty traffic due to the exchange of link quality information for every transmission. On the other hand, EMBA is the first proactive multihop broadcast protocol for asynchronous duty-cycled WSNs. It allows nodes to periodically exchange link quality information with neighbor nodes. With this procedure, EMBA can significantly reduce duty cycle of nodes and also provide a considerable broadcast efficiency for networks with bursty traffic.

III. Motivation and Network Model

A. Motivation

In WSNs, a topology generated by randomly deployed nodes includes various types of polygons (n-gons) such as triangles (3-gons), quadrangles (4-gons). The frequency of each n-gon is mainly affected by the network density which is the average number of 1-hop neighbors per node. Dense networks have higher frequency of triangles than sparse networks. In other words, the occurrence of n-gons where n is equal to or greater than 4 (referred to as $n(\geq 4)$-gons) becomes more frequent as the network is getting sparser. If walls or obstacles are located in a deployment area, the occurrence of $n(\geq 4)$-gons is more frequent because the number of wireless links between nodes is reduced. In addition, controlling transmission power is also one of reasons for an increase in the number of $n(\geq 4)$-gons.

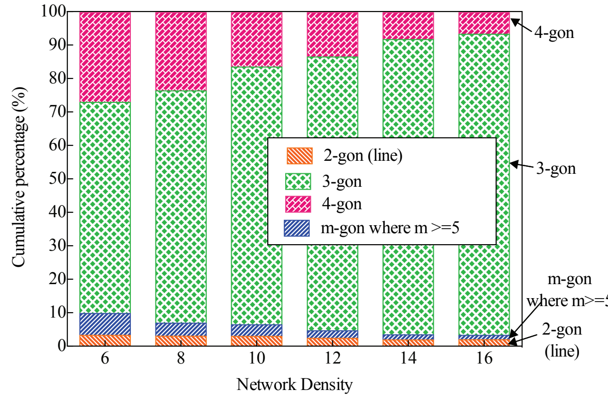


Fig. 2. Cumulative percentage of frequency of each n-gon.

To investigate our motivation, we have performed simulations that examine the frequency of each n-gon in randomly deployed sensor networks. We randomly scatter 50 nodes in 100 networks with each network density of 6, 8, 10, 12, 14, and 16. As shown in Figure 2, $n(\geq 4)$-gons increase as the network gets sparser. Therefore, avoiding collisions and redundant transmissions in $n(\geq 4)$-gons greatly improves performance of multihop broadcast in sparse networks. Figure 2 also shows that the sum of the occurrence ratio of 3-gon and 4-gon reaches more than 90% in both sparse and dense networks. To efficiently support broadcast in both 3-gon and 4-gon has a strong influence on overall performance of the multihop broadcast protocol in not only sparse networks but also dense networks.

B. Network Model

We consider a WSN consisting of fixed nodes that do not have an ability to move. Topology changes (e.g., the addition of a new node, or the elimination of a node from the network due to node fault or exhausted battery power) are infrequent and link quality information are valid for a relatively long time (approximately a few minutes).

In our network model, a transmission of a message is very likely to be unsuccessful when a node sends the message over a poor link. If this happens, the retransmission is required and therefore results in additional energy consumption. To avoid retransmissions, a node should send messages over links with as good quality as possible. EMBA uses knowledge of wireless link quality which can be provided from existing link estimation mechanisms such as SOFA [24], STLE [25], and four-bit link estimation [26]. These link estimators use various signal and decoding statistics such as RSSI (Received Signal

TABLE I
SETS FOR A FORWARDER s'S BROADCAST WITH A BROADCAST MESSAGE i, $BPKT^i$

Set	Element	Initial set
$N^i_{cv}(s)$	*covered* neighbor nodes by $BPKT^i$	ϕ
$N^i_{ucv}(s)$	*uncovered* neighbor nodes	$N(s)$
$N^i_{oblg}(s)$	neighbor nodes which forwarder s is *obligated* to cover	$N^i_{ucv}(s)$
$N^i_{dg}(s)$	neighbor nodes of which covering is *delegated* to another node with better link quality	ϕ

Strength Indicator) [20] [21], SNR (Signal to Noise Ratio) [20], LQI (Link Quality Indicator) [21].

The EMBA operates based on RI-MAC and uses a beacon message generated by a receiver for the start and acknowledgement of a data transmission. RI-MAC achieves higher throughput, higher energy efficiency, and lower end-to-end delay compared to the prior asynchronous approaches. However, EMBA can be implemented on other asynchronous duty-cycled MAC protocols such as B-MAC, X-MAC, and WiseMAC. The details are presented in Section VI-B.

We consider that a *MB procedure* is to propagate one broadcast message to all nodes in a network. A node may receive identical broadcast messages from multiple neighbors. In addition, collisions may occur when multiple senders attempt to transmit broadcast messages to a common node. Both redundant transmissions and collisions result in longer idle listening and more energy consumption because a node which has a broadcast message should stay awake and listen the channel until all of its neighbor nodes are *covered*. A covered node represents that it already received the broadcast message.

Each node maintains a *1-hop neighbor table* that consists of a 1-hop neighbor list and quality information of links between itself and each of its neighbor nodes. We denote the neighbor list of node s as $N(s)$ and we denote the link quality between s and a neighbor node r as $LQ(s,r)$. Node s generates an *advertisement message* including the sequenced pairs of $N(s)$ and $LQ(s,r)$. The node s then exchanges this advertisement message with its neighbor nodes to share the link quality information. We refer to this as the *advertisement procedure*. Advertised information helps for a node to decide either to take responsibility for covering of an uncovered neighbor node or to delegate this transmission to another node that has a better link. The advertisement procedure is periodically conducted and the advertising period is configurable by a network administrator.

Each node maintains another table, *2-hop neighbor table* which is composed of 2-hop neighbor list and link quality information obtained through the advertisement procedure. Every node can obtain 2-hop neighbor information by a total of O(n) messages for a network of n nodes [18]. Supporting $n(\geq 5)$-gon makes the protocol much more complicated. For example, 3-hop neighbor information is required in case of a 5-gon shape. However, supporting $n(\geq 5)$-gon yields very little performance improvement due to the very low frequency of $n(\geq 5)$-gon. Therefore, in EMBA, it would be enough to maintain 2-hop neighbor information.

To reduce the advertisement procedure overhead, an advertisement message is efficiently encoded. Each node ID of the 1-hop neighbor list is represented by 13 bits (at most 8192 nodes). Each $LQ(s,r)$ is divided into eight levels (0 – 7) and thus is represented by only 3 bits. Therefore, the size of an advertisement message is $|N(s)| * (13+3)$ bits. In our simulations in Section V, about 12 – 32 bytes are required as the average number of 1-hop neighbors varies from 6 to 16.

IV. EMBA

In this section, we describe the details of EMBA. We propose two techniques of the forwarder's guidance and the overhearing of broadcast messages and ACKs. We also present additional schemes that improve the completeness of our protocol.

A. Overview of the Forwarder's Guidance

In broadcast, there are two types of senders: a source node of broadcast and a node that forwards a broadcast message generated by a source node. We refer to both of them as *forwarders*. In multihop broadcast, a forwarder s transmits a broadcast message i, $BPKT^i$, to each neighbor node r. Node r prepares to work as a new forwarder after receiving $BPKT^i$. If there is a node v which is a common uncovered neighbor of nodes s and r, v will receive two identical $BPKT^i$s from both s and r. In addition, a collision will occur if nodes s and r simultaneously transmit $BPKT^i$ to node v.

EMBA can significantly reduce redundant transmissions and collisions by using the forwarder's guidance. When a forwarder s sends $BPKT^i$, it constructs sets as shown in Table I. Each set is one of subsets of $N(s)$ and should satisfy following two conditions:

$$N(s) = N^i_{cv}(s) \cup N^i_{ucv}(s), \tag{1}$$

and

$$N^i_{oblg}(s) = N^i_{ucv}(s) - N^i_{dg}(s) \tag{2}$$

The forwarder s is obligated to send $BPKT^i$ to every node in $N^i_{oblg}(s)$ by using unicast. We refer to $N^i_{oblg}(s)$ as the *obligation set* of the forwarder s. Before the forwarder s transmits $BPKT^i$ to a neighbor node $r \in N^i_{oblg}(s)$, it generates a *guidance list (GL)* for r, which is piggybacked in $BPKT^i$. Through the advertisement procedure, node s has obtained the neighbor list of node r ($v_1, \ldots, v_{|N(r)|} \in N(r)$) and the link quality information of r ($LQ(r,v_1), \ldots, LQ(r,v_{|N(r)|})$). Node s as a forwarder provides the guidance about how node r covers each of neighbor nodes of r. We denote the guidance from node s to node r as GL^s_r and it is composed of $GL^s_r[v_1], \ldots, GL^s_r[v_{|N(r)|}]$. The forwarder s assigns one of the following *guidance states* to $GL^s_r[v]$ for each node $v \in N(r)$:

- $GL_r^s[v]$ = *COVERED* if nodes v and s are the same node or v is already covered. In this case, node r does not do anything for node v after receiving this guidance from forwarder s.
- $GL_r^s[v]$ = *DELEGATED* if covering of node v is delegated to another node that can be either the forwarder s or any other 1-hop neighbor of node v except node r. This guidance means that node v is uncovered yet but v will be covered by another node which has a better link to v than node r.
- $GL_r^s[v]$ = *OBLIGATED* if node r is obligated to cover node v. Since node r has the best link quality to node v among the neighbors of node s, r has responsibility for forwarding $BPKT^i$ to v.

GL_r^s is delivered to node r with $BPKT^i$. Before working as a new forwarder, node r eliminates the IDs of nodes assigned by *COVERED* or *DELEGATED* state from its obligation set. The new forwarder r will cover the reduced set of its neighbor nodes by the aid of the forwarder's guidance from node s.

B. The Forwarder's Guidance Details

Algorithm 1 Procedure CONSTRUCT_GL()

1: // s: forwarder, r: receiver, i: broadcast message i
2: **for** each node $v \in N(r)$ **do**
3:   **if** $v \in N_{cv}^i(s)$ or $v = s$ **then**
4:     $GL_r^s[v] \leftarrow$*COVERED*
5:   **else if** $v \in N_{ucv}^i(s)$ **then**
6:     **if** $LQ(s,v) \geq LQ(r,v)$ **then**
7:       $GL_r^s[v] \leftarrow$*DELEGATED*
8:     **else**
9:       $GL_r^s[v] \leftarrow$*OBLIGATED*
10:       $N_{dg}^i(s) \leftarrow N_{dg}^i(s) \cup \{v\}$
11:       $N_{oblg}^i(s) \leftarrow N_{oblg}^i(s) - \{v\}$
12:     **end if**
13:   **else** // if $v \notin N(s)$
14:     find the node t which has the best quality link to v among s's neighbor nodes
15:     **if** $r = t$ **then**
16:       $GL_r^s[v] \leftarrow$*OBLIGATED*
17:       $N_{dg}^i(s) \leftarrow N_{dg}^i(s) \cup \{v\}$
18:       $N_{oblg}^i(s) \leftarrow N_{oblg}^i(s) - \{v\}$
19:     **else**
20:       $GL_r^s[v] \leftarrow$*DELEGATED*
21:     **end if**
22:   **end if**
23: **end for**

Before a forwarder s transmits a broadcast message to a receiver $r \in N_{oblg}^i(s)$, it executes Algorithm 1 to make GL_r^s. The procedure of constructing the *GL* consists of three parts. First, if node v which is a neighbor node of the receiver r is already covered, $GL_r^s[v]$ is simply set to *COVERED* (lines 3 – 4). If node r receives $GL_r^s[v]$ = *COVERED* from the forwarder s, r will never do anything for node v because v is already covered.

The second part (lines 7 – 12) is a case that node v is a common neighbor of nodes s and r ($v \in N(r) \cap N(s)$). To avoid sending the broadcast message over a poor link, EMBA makes the following guidance to choose the forwarder which covers node v:

- If $v \in N_{ucv}^i(s)$ and $LQ(s,v) \geq LQ(r,v)$, $GL_r^s[v]$ is set to *DELEGATED*.
- If $v \in N_{ucv}^i(s)$ and $LQ(s,v) < LQ(r,v)$, node s assigns $GL_r^s[v]$ *OBLIGATED* and updates $N_{dg}^i(s) \leftarrow N_{dg}^i(s) \cup \{v\}$ and $N_{oblg}^i(s) \leftarrow N_{oblg}^i(s) - \{v\}$.

Node r will work as a new forwarder after receiving GL_r^s piggybacked in $BPKT^i$ from node s. If node r receives $GL_r^s[v]$ = *DELEGATED*, r will not do anything for node v because v will be covered by another node which has better link quality to v than r. Node r will cover node v only if $GL_r^s[v]$ is *OBLIGATED*.

The last part (lines 13 – 21) is a case that node v is not a neighbor of a forwarder s ($v \in N(r) - N(s)$). This part is executed when a forwarder is one of nodes forming a quadrangular topology. The forwarder s does not have any knowledge of whether node v is already covered (because nodes s and v cannot communicate with each other). Assume that there are two or more neighbors of node s can communicate with node v. We denote them as $r_1, \ldots, r_n$. If forwarder s assigns *OBLIGATED* to all of $GL_{r_1}^s[v]$, ..., $GL_{r_n}^s[v]$, redundant transmissions and collisions will occur because all nodes $r_1, \ldots, r_n$ will attempt to cover node v. To prevent this problem, a forwarder s gives the guidance (*OBLIGATED* or *DELEGATED*) to each node of $r_1, \ldots, r_n$. Node s gives *OBLIGATED* only to a node r_t which has the best link. Node s, then, gives *DELEGATED* to other nodes $r_1, \ldots, r_{t-1}, r_{t+1}, \ldots, r_n$. The node r_t receiving $GL_{r_t}^s[v]$ = *OBLIGATED* will only attempt to cover node v.

A forwarder s transmits $BPKT^i$ with the *GL* to each neighbor node $r \in N_{oblg}^i(s)$. Upon receiving $BPKT^i$, each node r sends an acknowledgement message to node s. Let ACK^i be an acknowledgement message to signify receipt of $BPKT^i$. After receiving ACK^i from node r, node s updates the sets as follows:

$$N_{cv}^i(s) = N_{cv}^i(s) \cup \{r\} \tag{3}$$

$$N_{ucv}^i(s) = N_{ucv}^i(s) - \{r\} \tag{4}$$

$$N_{oblg}^i(s) = N_{oblg}^i(s) - \{r\} \tag{5}$$

If $N_{oblg}^i(s)$ becomes empty, the forwarder finishes broadcast and goes to sleep.

A receiver node r executes the *ANALYZE_GL* procedure after receiving $BPKT^i$. The purpose of this procedure is to analyze a guidance list delivered from a forwarder s and to follow the guidance. As shown in Algorithm 2, node r decodes the GL_r^s piggybacked in $BPKT^i$ to decide how to handle each neighbor node $v \in N(r)$. First, if $GL_r^s[v]$ is *COVERED*, node r adds the ID of node v into $N_{cv}^i(r)$ and removes it from both $N_{ucv}^i(r)$ and $N_{oblg}^i(r)$ (lines 7 – 10). Second, if $GL_r^s[v]$ is *DELEGATED*, node r inserts the ID of node v into $N_{dg}^i(r)$ and eliminates it from $N_{oblg}^i(r)$ (lines 11 – 13). Node r examines whether $N_{oblg}^i(r)$ is empty after analyzing the GL_r^s. If $N_{oblg}^i(r)$ is empty, node r goes to sleep immediately. Otherwise, node r serves as a new forwarder and performs the same procedures as explained above.

Algorithm 2 Procedure ANALYZE_GL(s,i)

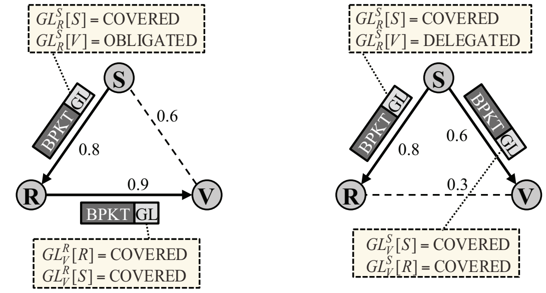
// s: forwarder, r: receiver, i: broadcast message i
$N_{cv}^i(r) \leftarrow \phi$
$N_{ucv}^i(r) \leftarrow N(r)$
$N_{oblg}^i(r) \leftarrow N_{ucv}^i(r)$
$N_{dg}^i(r) \leftarrow \phi$
for each node $v \in N(r)$ **do**
  if $GL_r^s[v]$ =*COVERED* **then**
    $N_{cv}^i(r) \leftarrow N_{cv}^i(r) \cup \{v\}$
    $N_{ucv}^i(r) \leftarrow N_{ucv}^i(r) - \{v\}$
    $N_{oblg}^i(r) \leftarrow N_{oblg}^i(r) - \{v\}$
  else if $GL_r^s[v]$ =*DELEGATED* **then**
    $N_{dg}^i(r) \leftarrow N_{dg}^i(r) \cup \{v\}$
    $N_{oblg}^i(r) \leftarrow N_{oblg}^i(r) - \{v\}$
  end if
end for

We give simple examples to describe the overall process of the forwarder's guidance. Figure 3(a) shows the operation of the forwarder's guidance of EMBA in a triangular topology formed by S, R, and V. The source node S initiates to broadcast with $BPKT^i$. In the triangle, node S transmits a broadcast message with $GL_R^S[V]$ = *OBLIGATED* to node R if the link quality between nodes R and V is better than that between nodes S and V. Node V, therefore, will be covered by node R without any redundant transmissions or collisions. If the link quality from S to V is better than that from R to V, node S informs node R that covering of node V is delegated to the original forwarder S. Therefore, node R does not attempt to cover node V and it goes to sleep immediately upon receiving $GL_R^S[V]$ = *DELEGATED*.

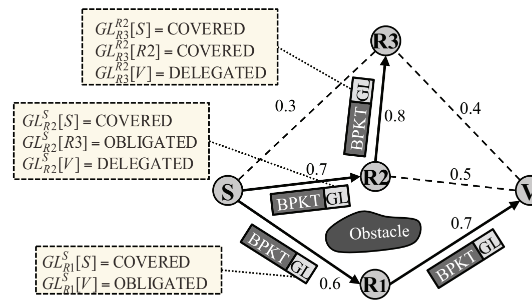
Figure 3(b) gives an example of the operation of EMBA in a complex topology which includes a quadrangle formed by four nodes: S, $R1$, $R2$, and V. Node S transmits $BPKT^i$ to only nodes $R1$ and $R2$, but not R3. If forwarder S sends $BPKT^i$ to node $R3$, the probability of message loss will be increased due to the poor link between S and $R3$. Therefore, covering of node $R3$ is delegated to node $R2$ which has the better link to $R3$. If without the forwarder's guidance, node V will receive the same broadcast messages from nodes $R1$, $R2$, and $R3$. In EMBA, node V will be covered only by node $R1$ since forwarder S gives $GL_{R1}^S[V]$ = *OBLIGATED* only to $R1$ that has the best link quality to V. As a result, nodes $R1$, $R2$, and $R3$ can avoid redundant transmissions and collisions.

C. The Overhearing of Broadcast Messages and ACKs

EMBA adopts another technique of the overhearing of broadcast messages and ACKs ($BPKT^i$ and ACK^i). If a forwarder overhears $BPKT^i$ or ACK^i destined to a certain node during an active state, it can eliminate the ID(s) of the node(s) specified in the message from its obligation set. Therefore, the number of transmissions required for covering of neighbor nodes will be reduced. This technique significantly increases energy efficiency by reducing the active time of each forwarder and the number of transmissions. This proposed technique is more efficient in dense networks since the number of overhearable messages increases.



(a) Operation of the forwarder's guidance in a triangular topology. Forwarder *S* sends a broadcast message with *GL* to a receiver *R*. If the link from *R* to *V* is better than that from *S* to *V*, forwarder *S* gives OBLIGATED guidance to *R* so that *V* will be covered by *R*. Otherwise, forwarder *S* gives DELEGATED guidance to *R* and attempts covering of node *V* for itself.



(b) Operation of the forwarder's guidance in a complex topology including a quadrangle formed by only four nodes: *S*, *R1*, *R2*, and *V*. Forwarder *S* sends the broadcast message only to nodes *R1* and *R2*. Covering of node *R3* is delegated to node *R2* because the link quality from *R2* to *R3* is better than that from *S* to *R3*. Although nodes *R1*, *R2*, and *R3* can communicate with node *V*, *R1* only attempts to cover *V*. It helps to avoid not only redundant transmissions to *V* but also collisions at *V*.

Fig. 3. Operation of the forwarder's guidance of EMBA.

If forwarder x overhears $BPKT^i$ transmitted from node u to node v during an active state, it updates the sets as follows:

$$N_{cv}^i(x) \leftarrow N_{cv}^i(x) \cup \{u\} \tag{6}$$

$$N_{ucv}^i(x) \leftarrow N_{ucv}^i(x) - \{u\} \tag{7}$$

$$N_{oblg}^i(x) \leftarrow N_{oblg}^i(x) - \{u\} \tag{8}$$

$$N_{dg}^i(x) \leftarrow N_{dg}^i(x) - \{u\} \tag{9}$$

A transmission of ACK^i message indicates that both a forwarder and a receiver are already covered. Overhearing ACK^i can reduce the size of the obligation set of a forwarder more quickly than overhearing $BPKT^i$. If forwarder x overhears ACK^i destined from node u to node v, it updates the sets as follows:

$$N_{cv}^i(x) \leftarrow N_{cv}^i(x) \cup \{u, v\} \tag{10}$$

$$N_{ucv}^i(x) \leftarrow N_{ucv}^i(x) - \{u, v\} \tag{11}$$

$$N_{oblg}^i(x) \leftarrow N_{oblg}^i(x) - \{u, v\} \tag{12}$$

$$N_{dg}^i(x) \leftarrow N_{dg}^i(x) - \{u, v\} \tag{13}$$

The number of transmissions required to complete broadcast is very closely related to the size of the obligation set. This simple technique significantly improves the performance of EMBA in terms of both the message cost and energy efficiency.

D. Efficient Encoding of the Guidance List

The guidance list is encoded by using *bitmap* to reduce its size. Forwarder s maintains the sequenced neighbor table of node r ($v_1, \ldots, v_{|N(r)|} \in N(r)$). Forwarder s has only to assign one of guidance states to each element of GL_r^s ($GL_r^s[v_1]$, ..., $GL_r^s[v_{|N(r)|}]$) for node r. Each state can be encoded with only 2 bits because three guidance states are used in the forwarder's guidance in EMBA (***COVERED***, ***DELEGATED***, and ***OBLIGATED***). Therefore, the size of the GL_r^s is only $|N(r)| * 2$ bits. In our simulations, each GL_r^s piggybacked in $BPKT^i$ can be encoded in 2 – 4 bytes.

E. Handling of Network Failure

EMBA makes the best effort to avoid transmissions over poor links. Nevertheless, a message can be lost in the air either due to a collision or a link error. To support reliable multihop broadcast, EMBA follows the network failure resolving mechanism of the MAC protocol cooperated with itself. If $BPKT^i$ or ACK^i is lost, a forwarder will retransmit the $BPKT^i$ after a retransmit timeout. The retransmission can easily compensate for the loss of $BPKT^i$ or ACK^i. However, the loss of $BPKT^i$ or ACK^i in the overhearing is an important problem. If a forwarder x overhears an ACK^i destined from node u to node v and it is lost, x will eliminate the IDs of u and v from $N^i_{oblg}(x)$ but v actually does not receive the ACK^i yet. Since the ACK^i is not delivered to node v within a retransmit timeout, v will retransmit the $BPKT^i$ to node u.

EMBA solves this problem in a best effort manner. Each node maintains the forwarder address, the receiver address, and the broadcast number i of the most recently received $BPKT^i$ or ACK^i for a period of n duty cycles. After this duration, it updates its sets as presented in Section IV-C. In our simulations, we set this duration to 3 duty cycles.

V. Performance Evaluation

We use the *ns-2* simulator to evaluate the performance of EMBA in various network density scenarios with randomly deployed 50 nodes. We compare EMBA with ADB and RI-MAC broadcast. Comparison with ADB and RI-MAC broadcast provides a good baseline on how much improvement EMBA achieves. We simulate two variants of EMBA: (i) EMBA which includes the forwarder's guidance and the overhearing broadcast messages and ACKs (represented as *EMBA* in Figure 4), and (ii) EMBA without overhearing, which uses only the forwarder's guidance (represented as *EMBA w/o OH* in Figure 4). Comparison of EMBA and EMBA without overhearing provides a good insight on each performance of the forwarder's guidance and the overhearing of broadcast messages and ACKs.

RI-MAC can support broadcast in two ways [7]. One (referred to here as *RI-MAC-1*) is by transmitting a broadcast message to each of neighbor nodes by using unicast. The other of RI-MAC broadcast (referred to here as *RI-MAC-2*) is equivalent to the way used in X-MAC-UPMA, which is by repeatedly transmitting the same broadcast messages for a time as long as a receiver's sleep period. In ADB paper [19], *RI-MAC-1* showed better performance than *RI-MAC-2*, thus we adopt *RI-MAC-1* as the protocol compared with EMBA. For RI-MAC broadcast (i.e., *RI-MAC-1*) in our simulations, the interval between two successive beacon generations is uniformly distributed between 0.5 × L and 1.5 × L where L is the sleep interval parameter specified by user. The sleep interval, L, is set to 1 second for all four protocols (EMBA, EMBA w/o OH, ADB, and RI-MAC broadcast).

TABLE II
Radio and MAC Parameters for Simulations

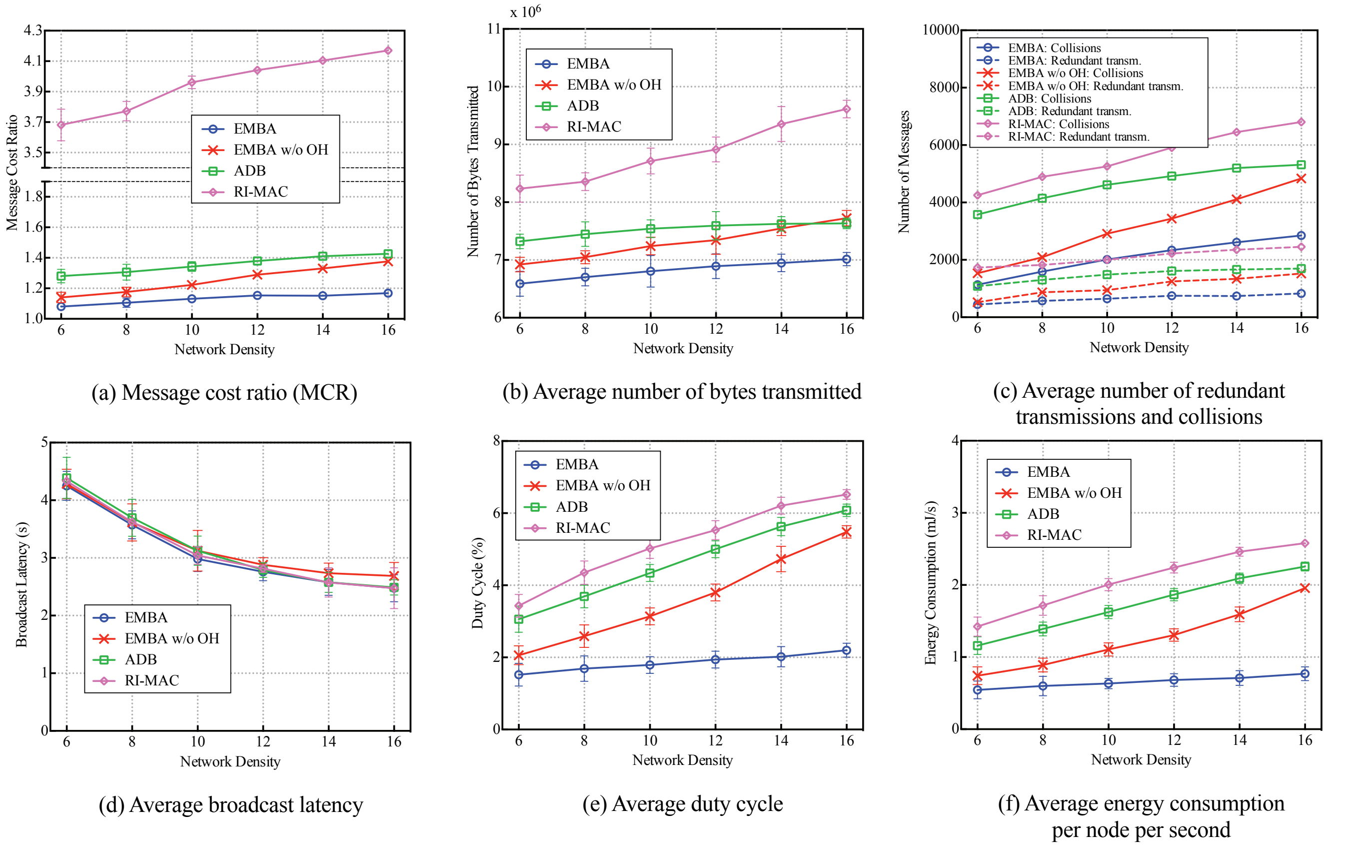
Parameter	Value
Bandwidth	250 Kbps
SIFS	192 μs
Slot time	320 μs
CCA check delay	128 μs
Transmission range	250 m
Carrier sensing range	550 m
Power in transmitting	52.2 mW
Power in receiving	56.4 mW
Power in listening	56.4 mW
Power in sleeping	3 μW
Size of a beacon	6–9 B
Size of ACK^i for $BPKT^i$	10–11 B

To compare EMBA with ADB and RI-MAC broadcast, we adopt radio transceiver and MAC protocol parameters used in [7] and [19], as shown in Table II. The radio protocol parameters are based on the data sheet of CC2420 radio [22], which is popularly used in MICAz and TelosB motes. We use a binary exponential backoff (BEB) that takes values of 0, 31, 63, 127 and 255. The initial size of backoff window (BW) is 32 and the maximum retry count is set to 5. The size of a base beacon is 6 bytes. If a beacon includes the BW field or it is an ACK beacon, a beacon size is approximately 7 – 9 bytes. ACK^i includes the broadcast number of i, so additional 2 bytes for this field are needed.

We perform simulations in 100 random networks with each network density of 6, 8, 10, 12, 14, and 16 (totally 600 runs). The purpose of varying the network density is to show how efficiently EMBA supports multihop broadcast in both sparse and dense networks. As the network density increases, more collisions are easy to occur due to the contentions among multiple forwarders. We use the shadowing channel model in which the packet loss occurs due to scatters such as obstacles. In every simulation, the sink node operates as a source node of broadcast and it periodically originates a broadcast message at a random interval of between 20 and 40 seconds (totally 100 broadcast messages during each run). We set the size of data payload to 28 bytes which is the maximum payload size and is defined as a default value in TinyOS. The advertising period in EMBA is set to 150 seconds. All nodes periodically transmit an advertisement message to each of its neighbor nodes every 150 seconds.

A. Results of Performance Comparison

Figure 4 shows the results of performance comparison obtained in our simulations. The points on the graphs are mean values of each performance metrics obtained from the 100 random networks for each network density (totally 600 runs). Error bars indicate the 95% confidence intervals.



(a) Message cost ratio (MCR)

(b) Average number of bytes transmitted

(c) Average number of redundant transmissions and collisions

(d) Average broadcast latency

(e) Average duty cycle

(f) Average energy consumption per node per second

Fig. 4. The results of performance comparison obtained in our simulations.

We newly define *message cost ratio (MCR)* which is the value of ratio between the actual number of transmissions and the minimum number of transmissions. It is calculated as follows:

$$MCR = \frac{t}{n-1} \tag{14}$$

where t is the average number of actual transmissions and n is the number of nodes. Theoretically, the minimum number of unicast transmissions to propagate a broadcast message to n nodes is $(n-1)$ in an error-free network. If MCR is 1, a network can be covered with the minimum number of transmissions. MCR does not include control messages (beacons and ACKs) and advertisement messages. It also does not count the number of retransmissions of broadcast messages. MCR is a meaningful metric to evaluate how many pairs of a forwarder and a receiver are used for finishing a MB procedure. For simple examples, 2 pairs are minimum in a triangle and 3 pairs are minimum in a quadrangle. Both MCR values of these two examples are equal to 1.

Figure 4(a) shows the MCR of EMBA, EMBA w/o OH, ADB, and RI-MAC broadcast. MCR of all four protocols linearly increases as the network density increases. EMBA achieves about 15.6 – 18.1% improvement compared to ADB and 70.6 – 72% improvement compared to RI-MAC broadcast in terms of MCR. MCR of EMBA w/o OH increases with steeper slope than EMBA and ADB because the overhearing is more useful in dense networks than in sparse networks. The performance gap between EMBA and ADB is widened by the aid of the forwarder's guidance and the overhearing of broadcast messages and ACKs. Since EMBA minimizes redundant transmissions by using these two techniques, EMBA finishes a MB procedure with a smaller number of transmissions (smaller number of pairs of a forwarder and a receiver) than ADB and RI-MAC broadcast in both sparse and dense networks.

Figure 4(b) shows the average number of bytes transmitted by each protocol. The average number of bytes transmitted includes all types of messages such as broadcast messages, control messages (beacons and ACKs), and advertisement messages. The result of EMBA w/o OH shows that the average number of bytes transmitted in EMBA w/o OH exceeds that in ADB at around the network density of 16. EMBA only with the forwarder's guidance reveals a limitation in extremely dense networks. On the other hand, EMBA achieves 8.1 – 10% improvement compared to ADB and 20 – 27% improvement compared to RI-MAC broadcast in both sparse and dense networks. The average number of bytes transmitted in EMBA is smaller than that in ADB and RI-MAC broadcast for every network density. This result indicates that EMBA finishes multihop broadcast with lighter network load than ADB and RI-MAC broadcast, although advertisement procedure is included in EMBA.

Figure 4(c) shows the total number of redundant transmissions of the same broadcast messages and collisions in four protocols of EMBA, EMBA w/o OH, ADB and RI-MAC broadcast. EMBA considerably reduces the redundant transmissions of the same broadcast messages by about 51.4 – 58.9% compared to ADB and by about 65 – 74.5% compared to RI-MAC broadcast. It helps to both support multihop broadcast with low network load and improve energy efficiency. EMBA also reduces the number of collisions by about 46.4 – 68.4% compared to ADB and by about 58.1 – 73.4% compared to RI-MAC broadcast. The large number of redundant transmissions and collisions with ADB is caused by no consideration of frequent occurrence of quadrangular topologies. On the other hand, nodes in EMBA avoid redundant transmissions and collisions by using the forwarder's guidance and the overhearing of broadcast messages and ACKs. Consequently, EMBA significantly reduces the idle listening of each node and thus it leads to much higher energy efficiency.

In multihop broadcast, the broadcast latency is obtained by measuring the time between the beginning of broadcast from a source node and the time that the broadcast message is delivered to all nodes in the network. The broadcast latency performance of EMBA, EMBA w/o OH, ADB, and RI-MAC broadcast is evaluated as varying network density, as shown in Figure 4(d). The broadcast latencies with EMBA, ADB, and RI-MAC broadcast are almost equal while the broadcast latency with EMBA w/o OH is a little larger than that with three protocols in dense networks. The differences between the broadcast latency values of EMBA, ADB, and RI-MAC broadcast are negligible for all network density values. These results are mainly due to the characteristic of the asynchronous duty-cycled MAC protocols. To relay a broadcast message in asynchronous environments, each forwarder should wait until neighbor nodes wake up in every hop because each node has its own duty cycle schedule. In other words, duty cycle (i.e., the sleep interval) is a dominant factor of the broadcast latency. If the sleep intervals of EMBA, ADB, and RI-MAC broadcast are equal, the differences between the broadcast latencies of them are very low. For this reason, the improvement of the broadcast latency is negligible from the point of view of an entire network.

Figure 4(e) shows the average duty cycle with EMBA, ADB, and RI-MAC broadcast. EMBA achieves 50.2 – 63.8% improvement compared to ADB and 55.7 – 66.3 compared to RI-MAC broadcast. Figure 4(f) shows the average energy consumption per node per second in four protocols. EMBA consumes much less energy than ADB and RI-MAC broadcast for all network density values. EMBA reduces the energy consumption by 52.9 – 65.9% compared to ADB and by 61.7 – 70.2% compared to RI-MAC broadcast. Unlike ADB and RI-MAC broadcast, the duty cycle and energy consumption with EMBA are not almost influenced by the network density. Consequently, we conclude that EMBA achieves much higher energy efficiency than ADB and thus it can support multihop broadcast efficiently regardless of network density.

B. Impact of Varying Advertising Periods

Performance of EMBA is influenced by the advertising period due to the characteristic of the proactive approach. To examine the average number of bytes transmitted according to changes in the advertising period, we evaluate three variants of EMBA: *EMBA-30*, *EMBA-90*, and *EMBA-150*. Nodes in each of the variants periodically execute the advertisement procedure every 30, 90, and 150 seconds, respectively.

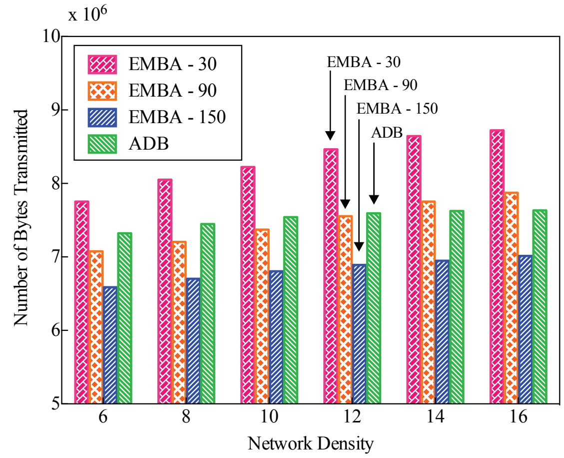


Fig. 5. The results of the average number of bytes transmitted according to changes in the advertising period.

Figure 5 shows the average number of bytes transmitted in EMBA-30, EMBA-90, EMBA-150, and ADB. All results are linearly increased because collisions and redundant transmissions increase as the network gets denser. EMBA-30 shows poor performance due to high traffic load and the increased number of collisions. On the other hand, the result of EMBA-90 is similar to that of ADB. In dynamic networks, nodes need to execute the advertisement procedure more frequently to reflect rapid changes of topology or link quality. If nodes in EMBA periodically advertise with below 90 seconds, EMBA can be less efficient than ADB in terms of the average number of bytes transmitted. From this evaluation, we conclude that EMBA is more efficient in static networks where topology and link quality changes are not frequent than in dynamic networks.

VI. Some Remarks

A. Analysis of Broadcast Latency

We analyze the broadcast latency of EMBA in a simplified model where the network is error- and collision-free. This analysis gives a rough estimation of the upper bounds of broadcast latency.

Lemma 1. *In a network that has N nodes, under the assumption of the error- and collision-free network, the upper bound of the 1-hop broadcast latency is $M_{oblg} * (T_s + T_t)$, where M_{oblg} is the size of the largest obligation set of nodes in the network, T_s is the sleep interval, and T_t is maximum time that a broadcast message takes to be transmitted to a neighbor node.*

Proof. A forwarder s should transmit the broadcast message, $BPKT^i$, to each neighbor node $r \in N^i_{oblg}(s)$. The number of broadcast message transmissions of node s is $|N^i_{oblg}(s)|$. Therefore, the maximum number of transmissions for 1-hop broadcasting is M_{oblg} that is calculated as:

$$M_{oblg} = \max_{0 \leq n < N} |N^i_{oblg}(n)| \tag{15}$$

The forwarder s is able to start a transmission after waiting until the wake-up time of an intended receiver. If the forwarder wakes up as soon as the intended receiver goes to sleep, the waiting time is at most T_s. In this worst case, each transmission of the broadcast message takes at most $T_s + T_t$ time. Thus, the upper bound of the 1-hop broadcast latency is $M_{oblg} * (T_s + T_t)$.

Lemma 2. *Suppose that the depth of the network is the maximum layers by breadth-first-search (BFS) and the depth is D. When a broadcast message generated by a source node s is delivered to all nodes in the network, the upper bound of the broadcast latency is $D * M_{oblg} * (T_s + T_t)$.*

Proof. By the Lemma 1, the 1-hop broadcast latency is at most $M_{oblg} * (T_s + T_t)$ time. Therefore, after $D * M_{oblg} * (T_s + T_t)$, every node in the network will receive the broadcast message.

This analysis goes a long way towards selecting the time interval between consecutive MB procedures. To reliably support successive MB procedures, the next one should start after the previous one completes. The minimum time interval between two MB procedures should be larger than the approximate upper bound of broadcast latency derived from this section. The upper bound analysis of the broadcast latency provides good parameters of the time interval between MB procedures for simulations or real deployments.

B. EMBA over Preamble-based MAC Protocols

EMBA can operate on various asynchronous duty-cycled MAC protocols such as not only receiver-initiated MAC protocol, but also preamble-based MAC protocols like B-MAC, X-MAC, and WiseMAC. EMBA is a proactive multihop broadcast protocol, so each node maintains its neighbor list and link quality information and shares them with its neighbor nodes through advertisement procedure. The advertisement procedure can be easily implemented in preamble-based MAC protocols. To support EMBA in preamble-based approaches, modification of the acknowledgement message of MAC protocols is needed. Unlike an ACK message for a unicast transmission, an ACK for broadcast (i.e., ACK^i) should include the broadcast number to distinguish the broadcast messages.

The sequence of covering of neighbor nodes is determined based on the order of wake-up times of neighbor nodes in the receiver-initiated MAC protocol because the start of a data transmission is initiated by a receiver. In preamble-based MAC protocols that are sender-initiated approaches, however, a forwarder should decide in what order it sends the broadcast message to its neighbor nodes. Several policies are available: the relative order of wake-up times of neighbor nodes, the order of link quality to each of neighbor nodes, the order of IDs of neighbor nodes, and the order of random.

For example, in EMBA over B-MAC, a forwarder s that has a broadcast message sends a preamble to an intended receiver r. Node s starts to transmit the broadcast message with GL^s_r after sending the preamble. Based on GL^s_r, the receiver r will work as a new forwarder or go to sleep immediately.

VII. Conclusions

In this paper, we have proposed an efficient multihop broadcast protocol for asynchronous duty-cycled WSNs (EMBA). EMBA can support multihop broadcast efficiently by using two techniques of the forwarder's guidance and the overhearing of broadcast messages and ACKs. The forwarder's guidance significantly reduces redundant transmissions and collisions in polygonal topologies such as triangle (3-gon) and quadrangle (4-gon). This technique greatly improves the energy efficiency in sparse networks by reducing duty cycle. The overhearing of broadcast messages and ACKs helps to reduce the number of transmissions. This simple technique minimizes the active time of forwarders, which is more efficient in dense networks. EMBA shows much higher energy efficiency in both sparse and dense networks compared to the conventional protocols such as ADB and RI-MAC broadcast. EMBA significantly improves the energy efficiency by 2.2 – 3 times compared to ADB and by 2.5 – 3.3 times compared to RI-MAC broadcast. EMBA also shows better performance of the message cost than the conventional protocols in terms of the message cost ratio (MCR) and the average number of bytes transmitted. Therefore, we have concluded that EMBA achieves much higher energy efficiency and message cost.

References

[1] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *Proc. 2007 ACM/IEEE IPSN*, pp. 254–263.

[2] W. Hu, V. N. Tran, W. Bulusu, C. T. Chou, S. Jha, and A. Taylor, "The design and evaluation of a hybrid sensor network for cane-toad monitoring," in *Proc. 2005 ACM/IEEE IPSN*, pp. 503–508.

[3] G. WernerAllen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proc. 2006 USENIX OSDI*, pp. 381–396.

[4] I. Stoianov, L. Nachman, S. Madden, T. Tokmouline, and M. Csail, "PIPENET: a wireless sensor network for pipeline monitoring," in *Proc. 2007 ACM/IEEE IPSN*, pp. 264–273.

[5] A. Czarlinska, W. Luh, and D. Kundur, "Attacks on sensing in hostile wireless sensor-actuator environments," in *Proc. 2007 IEEE GLOBECOM*, pp. 1001–1005.

[6] S. Du, A. K. Saha, and D. B. Johnson, "RMAC: a routing-enhanced duty-cycle MAC protocol for wireless sensor networks," in *Proc. 2007 IEEE INFOCOM*, pp. 1478–1486.

[7] Y. Sun, O. Gurewitz, and D. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. 2008 ACM SenSys*, pp. 1–14.

[8] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis, "Koala: ultra-low power data retrieval in wireless sensor networks," in *Proc. 2008 IEEE IPSN*, pp. 421–432.

[9] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *Proc. 2004 USENIX/ACM NSDI*, pp. 15–28.

[10] F. Stann, J. Heidemann, R. Shroff, and M. Z. Murtaza, "RBP: robust broadcast propagation in wireless networks," In *Proc. 2006 ACM SenSys*, pp. 85–98.

[11] Y. Sun, S. Du, O. Gurewitz, and D. B. Johnson, "DW-MAC: a low latency, energy efficient demand-wakeup MAC protocol for wireless sensor networks," in *Proc. 2008 ACM MobiHoc*, pp. 53–62.

[12] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated, adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 12, no. 3, pp. 493–506, Jun. 2004.

[13] T. Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. 2003 ACM SenSys*, pp. 171–180.

[14] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2004 ACM SenSys*, pp. 95–107.

[15] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. 2006 ACM SenSys*, pp. 307–320.

[16] K. Klues, G. Hackmann, O. Chipara, and C. Lu, "A component-based architecture for power-efficient media access control in wireless sensor networks," in *Proc. 2007 SenSys*, pp. 59–72.

[17] A. El-Hoiydi and J. D. Decotignie, "WiseMAC: an ultra low power MAC protocol for multi-hop wireless sensor networks," in *Proc. 2004 ALGOSENSORS*, pp. 18–31.

[18] G. Calinescu, "Computing 2-hop neighborhoods in ad hoc wireless networks," in *Proc. 2003 Ad-Hoc, Mobile, and Wireless Networks*, pp. 175–186.

[19] Y. Sun, O. Gurewitz, S. Du, L. Tang, and D. Johnson, "ADB: an efficient multihop broadcast protocol based on asynchronous duty-cycling in wireless sensor networks," in *Proc. 2009 ACM SenSys*, pp. 43–56.

[20] D. Lal, A. Manjeshwar, F. Herrmann, E. Uysal-Biyikoglu, and A. Keshavarzian, "Measurement and characterization of link quality metrics in energy constrained wireless sensor networks," in *Proc. 2003 IEEE Globecom*, pp. 446–452.

[21] IEEE 802.15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs), 1999. IEEE Std. 802.15.4, 2003.

[22] Texas Instrument Inc., CC2420 data sheet. Available: http://www.ti.com/product/cc2420.

[23] TinyOS 2.0 Documentation. Available: http://www.tinyos.net/tinyos-2.x/doc/.

[24] S. Lee, K. Kwak, and A. T. Campbell, "Solicitation-based forwarding for sensor networks," in *Proc. 2006 SECON*, pp. 90–99.

[25] A. Becher, O. Landsiedel, G. Kunz, and K. Wehrle, "Towards short-term link quality estimation," in *Proc. 2008 Emnets*.

[26] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," in *Proc. 2007 HotNets*.

Ingook Jang received his B.S. degree in Computer Science and Engineering from the Chung-Ang University, Seoul, South Korea, in 2008. He is currently working toward the Integrated Master's and Doctoral Degree Program in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. His current research interests include the design and analysis of energy efficient communication protocols, especially broadcast, medium access control, scheduling of packet transmissions, with applications in wireless sensor networks, wireless ad hoc networks, and broadband access networks.

Suho Yang received his B.S. degree in information and computer engineering from Ajou University, Suwon-si, South Korea, in 2008, and his M.S. degree in computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2010. He is currently working for the Korea Financial Telecommunications & Clearings Institute (KFTC). His current research interests include the design and analysis of energy efficient communication protocols for wireless sensor networks and mobile ad hoc networks.

Hyunsoo Yoon (M'89) received his B.S. degree in electronics engineering from Seoul National University, Seoul, South Korea, in 1979, his M.S. degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1981, and his Ph.D. degree in computer and information science from The Ohio State University, Columbus, in 1988. He is currently a Professor of the Department of Computer Science, KAIST. From 1978 to 1980, he was with the Tongyang Broadcasting Company, South Korea, then Samsung Electronics Company, South Korea during 1980 – 1984. From 1988 to 1989, he was a Member of the Technical Staff with AT&T Bell Labs, Indial Hill, IL. Since 1989, he has been a Professor with the Department of Computer Science, KAIST. His research interests include mobile ad hoc networking, wireless sensor networking, next-generation mobile communication networks, and information security.

Dongwook Kim received his B.S. degree in information and computer engineering from Ajou University, Suwon-si, South Korea, in 2002, and his M.S. and Ph. D. degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2004 and 2009, respectively. Since 2009, He is a senior engineer at the Network Business Division, Samsung Electronics. His research interests include the design and optimization of LTE (Long Term Evolution)-advanced systems with SON (Self-Organizing Networks) technologies, optimal deployment of mobile sensor networks, broadcast techniques for sensor networks, and optimal handover schemes for 4G mobile systems.