A MAC Layer Protocol for Priority-based Reliable Broadcast in Wireless Ad Hoc Networks

Murat Demirbas Muzammil Hussain
Department of Computer Science and Engineering
University at Buffalo, SUNY
Buffalo, NY, 14260

Abstract

RTS-CTS handshake based protocols achieve "reliable unicast" by eliminating the hidden node problem effectively, however, these solutions are not directly or efficiently generalizable for solving the single-hop "reliable broadcast" problem, and broadcast remains as a best-effort operation in wireless ad hoc networks. Here we present a simple, light-weight, and self-stabilizing MAC protocol, namely Busy Elimination Multiple Access (BEMA) protocol, for solving the single-hop reliable broadcast problem. BEMA grants on-demand access to the channel —rather than assigning fixed slots as in TDMA based approaches— and also supports prioritization of traffic, thereby providing a useful building block for applications with reliability and quality-of-service requirements.

1 Introduction

Due to recent advances in low power wireless radios and MEMS technology, it has been feasible to deploy large-scale wireless ad hoc networks for real-world application scenarios. Several wireless ad hoc network applications, specifically wireless sensor network applications, have been deployed in the last couple of years, for example, in environmental monitoring for monitoring nesting behavior of endangered birds in a remote island [26], in precision agriculture for monitoring of temperature and humidity in vineyards [4], and in military and surveillance scenarios for sniper localization [23], and for classification and tracking of trespassers [2, 3].

Although support for reliable unicast using RTS/CTS has existed traditionally in 802.11 [1] or in sensor network MAC layer protocols [11], there has not been any support for reliable broadcasting. Broadcasting has always been a best effort operation in 802.11 ad hoc broadcast mode [1] and in wireless sensor network MAC protocols [11, 19]. However, reliable broadcasting is an essential component of future sensor/actuator networks where all nodes need to consistently take a consistent course of action. For example, robotic highway safety/construction markers [13] have to consistently take the correct decisions, otherwise a robot cone that has inconsistent view of the system could enter in to traffic and create a significant hazard. Also, the sensor/actuator devices coordinating regulator valves should take consistent decisions to prevent a malfunction. These sensor/actuator systems would be instrumental in factory automation control systems and military applications, and would require maintenance of consistent states [9].

A major hurdle for reliable broadcast is the hidden node problem, where two transmitters that are outside each other's transmission range fail to detect other's transmission via carrier-sensing and their simultaneous transmission collides at a receiver node that lie within the range of both transmitters. There has been many studies [34, 36] showing the detrimental effects of hidden node problem. In particular, more than %50 message loss has been reported due to hidden node problem under bursty traffic loads in wireless sensor networks [35]. Such bursty traffic loads are possible due to convergecast in environmental monitoring applications and also for remote/wireless network re-

programming tasks. A reliable broadcast service that avoids the hidden node problems successfully would be an important primitive for these applications. Moreover, it is also desirable for the reliable broadcast service to provide support for prioritization of traffic as we would prefer to delay stale and non-critical data to make way for timely delivery of more recent and critical data in tracking and surveillance applications.

The reliable broadcast problem is difficult to solve for ad hoc networks. Simple extensions of RTS/CTS solution for reliable unicast have problems with either reliability or efficiency. Due to collision of CTS packets from multiple receivers in single-hop, in the BSMA protocol [27, 28] the transmitter is not able to determine if all receivers are ready to receive, and as a result collisions of data packets are likely. Moreover, due to collisions of NAKs (negative acknowledgments), it is not possible to ensure guaranteed delivery. In BMW [29] and BMMM [25], due to the high overhead in contacting each receiver individually for an RTS/CTS handshake before data transmission, communication efficiency (goodput) suffers. In TDMA based approaches [5, 15, 21, 32, 33] bandwidth is wasted due to the static or reservation-based scheduling of the transmissions, and hence, this hinders their adoption in low-power, delay-sensitive wireless ad hoc networks, such as wireless sensor networks where traffic is often bursty [35].

BTMA (Busy Tone Multiple Access) [30] provides a solution for the hidden node problem in ad hoc networks. In this protocol, when a node j is receiving data transmission, j uses a separate radio (and a separate frequency) to broadcast a "busy" signal. When a node k intends to transmit, k first listens to the control frequency for any possible busy signal. Only if k does not hear any "busy" signal, then k can start broadcasting in the data frequency. However, since BTMA assumes a separate radio and frequency for control signals, it is not applicable in low-power wireless networks, especially for wireless sensor network platforms [18], as unit cost and energy-requirements constraints these platforms severely.

Contributions of the paper. Our first contribution is an adoption of BTMA, namely BEMA (Busy Elimination Multiple Access), for low power, ad hoc, wireless sensor network platforms. We achieve this by using time synchronized rounds across all nodes (efficiently implemented via [10, 17]) to allocate a control channel in the time domain instead of in the frequency domain. In BEMA each round has a control phase and data phase. Before transmission, a node j listens to the control phase, upon hearing nothing j can transmit, and it can "lock" the receivers for some consecutive rounds, after which the locked receivers broadcast busy in the control phases of the following rounds. The busy signals collide in the control phase, however, using receiver-side carrier-sensing based collision detection techniques [7], it is possible to detect these collisions in the control phase and, hence, conclude that it is unsafe to transmit in the data phase.

Our second contribution is to support prioritization of traffic in our BEMA protocol. In BEMA, the control phase also doubles as a leader election phase. Nodes that have data to send bid for the data-slot in the control phase: depending on the priority of the data to be sent (also with some randomization mechanism), they broadcast a busy-signal for a determined length. This serves as a deferring mechanism for other transmitter candidates. After a candidate transmitter transmits busy signal for a determined time, it switches to listening: If it finds no other busy signal being still transmitted in the channel, then it won the bid, else, upon hearing a busy signal or detecting a collision that implies presence of at least one busy signal, it realizes it lost the bid and defers its transmission. This way the nodes with the highest priority data get to access the channel first. Since a continuing transmission has the most priority—as an interruption would render the data transmitted so far useless—, the "locked" nodes broadcast busy signals for the entire length of the control phase.

Last, but not least, our protocol is self-stabilizing, that is, starting from any arbitrary state our protocol eventually recovers to a state from where its specification is satisfied. Self-stabilization property is especially important in wireless ad hoc networks where initial states of a protocol is hard to configure or enforce. We also present a novel ad hoc round-synchronization algorithm that exploits the structure of the BEMA rounds to provide on-the-fly and on-demand round-synchronization, rather than an always-on global round-synchronization.

Recently, as part of the standardization efforts of wireless sensor network architectures, there has been

work on converging to a standard layer over which other protocols would be built. Analogous to the way that IP [20] is a narrow-waist for Internet protocols, single-hop broadcast communication is identified as a narrow-waist for wireless sensor network protocols [8]. In this context, we believe a priority-based reliable broadcast service, such as BEMA, could serve as a building block for applications with reliability and quality of service requirements.

We implemented our BEMA protocol under Prowler [24], a MATLAB-based, event-driven simulator for wireless sensor networks. Prowler simulates the radio transmission/propagation/reception delays of Mica2 motes [18], including collisions in ad-hoc radio networks realistically. We compare BEMA with CSMA/CA [16], BSMA [28], and BMMM [25]. We show that BEMA has little overhead and provides the highest goodput since it successfully and efficiently eliminates hidden node problems. In future work, we will implement BEMA as a MAC layer protocol under TinyOS [14], and compare its performance with CSMA/CA MAC layers, such as BMAC [19] and CC1000 MAC [14].

Outline. In Section 2 we present related work, and in Section 3 we discuss our program and network model briefly. We present our BEMA protocol and a formal proof of correctness in Sections 4.1 and 4.2 respectively. We discuss self-stabilization of BEMA in Section 4.3 and extensions to the protocol for achieving ad hoc round synchronization and energy efficiency in Section 4.4. In Section 5, we present our simulation results that compare performance of BEMA with several other reliable and unreliable broadcast solutions. We conclude the paper in Section 6.

2 Related work

Here we review related work on both best-effort and reliable single-hop broadcast protocols.

To decrease the probability of loss of broadcast packets over CSMA/CA, in the Robust Broadcast protocol [31] the sender chooses a neighboring node j to get feedback about its broadcast. The sender uses either RTS/CTS messages or acknowledgments from j to ensure that its broadcast reaches at least one node in the neighborhood, namely j, and performs a retransmission only in cases where it fails to receive positive feedback from j. However, receiving a confirmation only from one node j does not guarantee that all the nodes in the neighborhood have received the broadcast, since the hidden node problem may affect other nodes in the neighborhood, when j may be unaffected.

In [27], Tang and Gerla have proposed an extension of the RTS-CTS handshake for broadcast. After the sender broadcasts an RTS packet to single-hop neighborhood, all the receivers not in a YIELD state reply with a CTS and start waiting for data. If the sender receives any CTS within certain time of its RTS, it broadcasts the data packet, else it enters into the contention phase again. In BSMA [28], the authors improve the reliability of their protocol by augmenting it with a negative acknowledgment (NAK) mechanism. In this scheme, the sender waits for any NAKs after it sends its data. If a receiver fails to receive data after receiving an RTS, it transmits a NAK to the sender, which causes the sender to retransmit the data. Both of the above protocols, however, do not guarantee reliable broadcast since receiving a CTS from one of the receivers does not imply that all nodes are ready to receive. The sender's initial RTS may not be received at some of the neighboring nodes due to collisions at those nodes, and as a result these nodes do not expect any data packet nor complain by sending NAKs. Moreover, NAK packets, as well as CTS packets, can collide and get lost when multiple receivers transmit to the sender simultaneously.

In [29] Tang and Gerla propose the Broadcast Medium Window (BMW) protocol that implements a reliable broadcast operation via performing reliable unicast to each neighbor individually. For each node in the neighbor list, the sender transmits data using the RTS-CTS-DATA-ACK handshake. Though BMW provides reliable broadcasting, it does so at a great expense in latency and energy as BMW incurs at least w contention phases for a node with w neighbors.

The Batch Mode Multicast MAC (BMMM) [25] protocol improves on the BMW protocol by combining the w contention phases into one phase and sending the data only once as a broadcast instead of w unicasts. When the sender enters the contention phase, it transmits an RTS to every neighbor one by one and individually seeking their CTSs. After completing the contention phase with at least one CTS, the sender broadcasts the data only once and requests

ACKs from the receivers one by one. For the neighbors that fail to send an ACK, the above procedure is repeated. Thus, data collisions are still possible in BMMM, but through the individual ACK mechanism, BMMM guarantees eventual delivery of data. By way of contrast, BEMA eliminates the data transfer collisions entirely (improving the goodput) and achieves reliable broadcast to single-hop at once, rather than eventually. Also, in contrast to BEMA which is oblivious to the ids of the neighboring nodes, both BMW and BMMM require knowledge of neighbor ids for a broadcast to be performed.

The primary advantage of using TDMA based approaches [5, 15, 21, 32, 33] is that using tight synchronization among nodes, it is possible to predetermine a transmission schedule for avoiding any collision. Also by using TDMA for solving the hidden node problem the need for acknowledgments is eliminated. However, in TDMA based approaches a lot of bandwidth is wasted due to the static or reservation-based scheduling of the transmissions. In contrast to a TDMA based approach, where timeslots are pre-assigned for each node, in BEMA nodes contend in the control phase, analogous to the contention phase in CSMA protocols, and one of the contenders get access to the transmission rights in the data phase on a priority-basis. Therefore, BEMA is suitable for the bursty traffic pattern in wireless sensor networks [35]. Also, in contrast to a TDMA based approach, which is very sensitive to the "global" network topology, BEMA is oblivious to those changes and, hence, is more suitable for wireless ad hoc networks.

BTMA (Busy Tone Multiple Access) [30] provides a solution for the hidden node problem in ad hoc networks as we described in the Introduction. However, since BTMA assumes a separate radio and frequency for control signals, it is not applicable in low-power wireless networks. BEMA implements the BTMA approach in low power wireless ad hoc network platform by using time synchronized rounds across all nodes and allocating the control channel in the time domain instead of in the frequency domain.

HIPERLAN, a WLAN standard developed in Europe as an alternative for the IEEE 802.11, employs the Elimination Yield - Non-Preemptive Multiple Access (EY-NPMA) [12] protocol for prioritization of traffic. Each channel access cycle in EY-NPMA consists of four phases. In the first phase, one-slot priority assertion signals are sent in any of the five slots (as there are five packet priorities). Transmissions in previous timeslots disable scheduled transmissions in next timeslots, rejecting nodes with lower access priorities. In the second phase, a variable length burst is sent in the allocated twelve timeslots. When a node concludes its burst, if it hears other nodes in the following slots, it is eliminated. In the third phase, the node that starts transmission earlier in one of the allocated 14 timeslots dominates the others, and is granted access to transmit data in the fourth phase.

The prioritization scheme in BEMA is similar to the second phase of EY-NPMA. However, BEMA combines the second phase and the first phase of EY-NPMA and achieves both prioritization and elimination in one control phase as we discuss in Action 1 at Section 4.1. Also, in contrast to EY-NPMA, which fails to address the hidden node problem, BEMA solves the hidden node problem by (1) "locking" the receivers for transmissions that span multiple rounds, upon which the "locked" nodes transmit busy signals for the entire duration of the control phase and defer any nodes within single-hop distance from getting access to the channel, and (2) using the 2-hop-transmission rule as described in Section 4.2 when contending for the channel.

3 Preliminaries

A network consists of a (potentially large) number of stationary nodes. Each node has a field of communication, within which it is capable of receiving/transmitting messages. All nodes within this unit field are its immediate neighbors (duplex links). For a node j, we denote j's immediate neighbors as $Nbr(j)$, however, we do not assume that j knows the nodes in its neighborhood, and in this sense our network is an ad hoc network.

Notation. Nodes have unique ids. We use j, k and l to denote the nodes, and $j.var$ to denote a program variable residing at j. We denote a message broadcast by j as $bcast(msg\_j)$.

A *program* consists of a set of variables and actions at each node. Each action has the form:

$$<\textit{guard}> \longrightarrow <\textit{assignment statement}>$$

A $guard$ is a boolean expression over variables. An assignment statement updates one or more variables. A state is defined by a value for every variable in the program, chosen from the predefined domain of that variable. An action whose guard is true at some state is said to be *enabled* at that state and is executed.

Fault model. Nodes may fail-stop and crash, and new nodes may join the network. Moreover, the state of a node's program can be arbitrarily and transiently corrupted. Channels may suffer faults that corrupt, manufacture, duplicate, or lose (e.g., due to collision or fading) messages. These faults can occur in any finite number, at any time and in any order.

A program is *self-stabilizing* iff after faults stop occurring, starting from any arbitrary state, the program eventually recovers to a state from where its specification is satisfied.

Synchronized rounds. We assume globally synchronized rounds. That is, rounds start (and end) at the same time across all the nodes in the network as illustrated in Figure 1. Such synchronized rounds are feasible in wireless sensor networks by using a time synchronization protocol, such as FTSP [17]. Here, each node transmits periodic synchronization messages, for example once every minute, and by compensating for their clock skew (using least-squares estimation on earlier data points) achieve a micro-second level synchronization. It takes about 10 minutes to achieve the initial synchronization for a 10-hops network, but once initial synchronization is achieved the protocol is robust to node failures or network topology changes, and requires very little overhead for maintaining synchronization. FTSP is used in several real-world wireless sensor network deployments, including a sniper localization system [23] that has very tight time-synchronization and real-time delivery guarantees. We assert that BEMA protocol starts after FTSP achieves initial synchronization, and periodic synchronization messages in FTSP are sent over BEMA (rather than independently) to avoid interference with the BEMA layer.

Instead of employing an always-on global round-synchronization, BEMA can also work on top of an ad hoc and on-demand round-synchronization protocol as we describe in Section 4.4.

Receiver-side collision detection via carrier-sensing. Carrier sensing is widely employed in wireless networks with CSMA (Carrier Sense Multiple Access) MAC layers, including IEEE 802.11, IEEE 802.15.4, and wireless sensor network MAC protocols. Traditionally, carrier sensing has been used primarily at the transmitters: Before a transmitter starts it transmission, it senses the medium for any existing transmission, and only begins transmission if the medium is not already busy. We adopt this technique at the receiver side for detecting collisions.

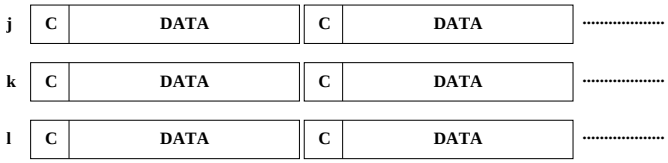

Figure 1. Synchronized rounds in BEMA.

To this end, we employ carrier sensing in the idle state. A node is in the idle state when it is not transmitting, or receiving a message, or synchronizing to receive a message. The node detects a collision when its carrier sensing mechanism detects in the idle state that there is an intense activity on the medium. Due to noise, there is a lot of activity in the transceiver even in the idle state. However, it is easy to differentiate between noise and a genuine activity, such as a message or collision. The random noise has significant variance in channel energy (occasional pits below the noise floor) whereas a genuine activity has fairly constant channel energy (always stays above the noise floor). Our carrier sensing at the idle state searches for these pits: if for a long period no pit is found, this is a good indication of genuine activity in the radio. In our preliminary experiments with the Mica2 mote platform [18], we find that our carrier-sensing based collision detection at the receivers has good performance, detecting more than 95% of the collisions accurately.

4 Busy Elimination Multiple Access Protocol

In this section, we present our BEMA protocol and provide a formal proof of correctness, showing that BEMA eliminates the hidden node problem. We also prove self-stabilization of BEMA in the face of arbitrary state corruptions, and discuss extensions to BEMA for achieving energy-efficiency and ad hoc, on-demand round-synchronization.

4.1 Protocol

Each node j maintains a single variable, $status$. $j.status$ has a domain of {*idle, candidate, waiting, leader, locked*}. As a shorthand, we use $j.x$ to denote $j.status = x$. Hence, $j.candidate$ means j wants to transmit a message, $j.waiting$ means j is trying to get access to the channel for the DATA phase, and $j.leader$ means j had exclusive access to the channel in the DATA phase and it will be transmitting the rest of its packets in the consecutive rounds. $j.locked$ implies that there exists a *leader* k within singlehop of j, and j is reserved to receive more packets from k in the next round. If none of the above holds for j, $j.idle$ is true by default. Initially for all j, $j.status = idle$.

The variable "phase" is an external variable (provided by a round synchronization service), notifying j of which phase of the round, CONTROL or DATA, j is in. All the nodes have consistent view of the phase variable due to our round synchronization requirement.

BEMA protocol consists of six actions as seen in Figure 3. Figure 2 illustrates the effect of actions on the status variable of a node. Note that Actions 1, 2, and 6 are enabled only in the CONTROL phase, and Actions 3, 4, and 5 are enabled only in the DATA phase.

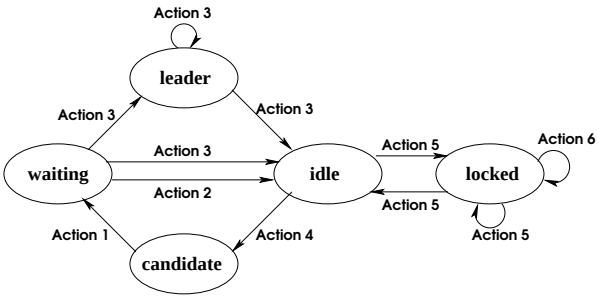

Figure 2. The effect of actions on the $status$ variable.

Action 1 is enabled in the CONTROL phase when a node j is a $candidate$. Upon execution j broadcasts a random length busy signal from a determined range based on the priority of the message, and transits to $waiting$ state[1]. For example, if there are 5 packet priority levels (1 being the lowest priority), and the packet to be transmitted at j has priority 4, then $f_j(\Delta)$ returns a random length from the range $\left[\frac{3\Delta}{5}, \frac{4\Delta}{5}\right)$. This way, we assert that the contention length is always less than Δ, the entire length of the CONTROL phase.

(1) phase=CONTROL $\wedge j.candidate$
$\longrightarrow$ bcast(busy) for $f_j(\Delta)$ time to 2-hop-distance
$j.status := waiting$
[]
(2) phase=CONTROL $\wedge j.waiting \wedge$ receive (busy or $\pm$)
$\longrightarrow$ $j.status := idle$
[]
(3) phase=DATA $\wedge (j.waiting \vee j.leader)$
$\longrightarrow$ bcast(msg_j) for T time
if $msg_j.remaining\_length > 0$
then $j.status := idle$
else $j.status := leader$
[]
(4) phase=DATA $\wedge j.idle \wedge \neg receive(msg_k)$
$\longrightarrow$ if ($data\_to\_send$)
then $j.status := candidate$
[]
(5) phase=DATA $\wedge (j.idle \vee j.locked) \wedge receive(msg_k)$
$\longrightarrow$ if $msg_k.remaining\_length > 0$
then $j.status := locked$
else $j.status := idle$
[]
(6) phase=CONTROL $\wedge (j.locked \vee j.leader)$
$\longrightarrow$ bcast(busy) for Δ time

Figure 3. Program actions for j.

Action 2 is enabled in the CONTROL phase when a $waiting$ node j receives a busy signal or detects a collision. Since both cases imply the existence of another (at least one) candidate with higher priority, j defers its transmission by going back to the $idle$ state. We explain the need for 2-hop-distance transmission in Section 4.2. The 2-hop-distance transmission can be satisfied by transmitting with 4 times the normal transmission power, assuming a quadratic signal fading formula.

Action 3 is enabled in the DATA phase when a node j is in the $waiting$ or $leader$ state. Since j is not deferred by another node via Action 2, this indicates that j has exclusive transmission rights as a high-priority node (provided that the random busy signal selection

[1] Since, switching from transmission to listening is on the order of microseconds, collision detection using the scheme in Section 3 is feasible.

mechanism resolves contentions among nodes with the same priority). Upon execution j transmits its message, and transits to $idle$ state if its message fit in to one DATA phase, or transits to $leader$ state to continue sending the rest of its message in the next rounds.

Action 4 is enabled in the DATA phase for an $idle$ node j. If j has data to transmit, j transits to $candidate$ state to contend for the channel in the upcoming CONTROL phase via Action 1.

Action 5 is enabled in the DATA phase when $idle$ or $locked$ j receives a message. If the $remaining\_length$ field of the message indicates that other packets will be transmitted in the next rounds as part of this message, j transits to $locked$ state and commits to receiving the rest of the packets. Else j transits to $idle$.

Action 6 is enabled in the CONTROL phase for a $locked$ or $leader$ node j. j transmits busy signal for the entire length of the CONTROL phase to defer any candidates from getting transmission rights to the channel, since j has committed to receive the rest of the packets for an ongoing transmission.

4.2 Correctness proof

We assume that the domain Δ is chosen large enough that the random length contention periods that $f_j(\Delta)$ returns are unique for any two contending nodes within transmission range of each other. The transmission range of contending nodes should be set to be at least twice of that of the normal transmission length (i.e., unit length) to avoid hidden node problems. Figure 4 illustrates the problem that may occur when the contention radius is of unit length. In the figure, only j and l are candidates and the remaining nodes are in the idle state. Let k be a node within unit distance (single-hop distance) of both j and l, and j and l are within 2-hop-distance away from each other. When j contends for the channel, it gets access to the channel since there is no node within single-hop contending for the channel. Similarly, l also gets access to the channel for transmitting data, and, as a result in the DATA phase there is a collision at k.

The above scenario is avoided in BEMA by requiring nodes contending via Action 1 to broadcast busy signals for $f(\Delta)$ time to 2-hop-distance. This 2-hop-distance contention rule ensures that contending nodes

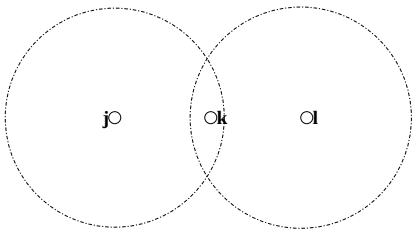

Figure 4. 2-hop-distance is required when contending.

within 2-hops are detectable to each other. Note that "$locked$" nodes broadcast busy signals to only single-hop distance. For example, in the above scenario if k was $locked$ neither j nor l would be able to get access to the channel.

We can now prove Lemma 1 about the contending nodes. Lemma 1 states that if there is no "$leader$" node j within single-hop of a node k in the beginning of a CONTROL phase, then there can be at most one node j within single-hop of k that has access to the channel (either in the $waiting$ or $leader$ state) in the DATA phase of that round.

Lemma 1 (Leader election). If $(\forall j : j \in Nbr(k) : \neg j.leader)$ [2] in the beginning of a round, then $(\forall j, l : j, l \in Nbr(k) : (j.leader \vee j.waiting) \wedge (l.leader \vee l.waiting) \implies j = l)$ in the DATA phase of the round.

Proof. Due to our assumption of unique length contention periods in Action 1, if there are contending nodes within single-hop of a node k, then there exists a unique node j, $j \in Nbr(k)$, with the highest-length contention period. Since contention signals are broadcasted to 2-hop-distance, j dominates all other nodes within single-hop of k, upon which the deferred nodes transit to $idle$ state via Action 2. Thus, only j may be eligible to remain in $waiting$ state in the DATA phase or transit to $leader$ state. □

Note that a chain-of-dominance where nodes with high-priorities dominate over nodes with lower-priorities is possible with a priority-based channel ac-

[2] A formula $(op\ j : R.j : X.j)$ denotes the value obtained by performing the (commutative and associative) op on the $X.j$ values for all j that satisfy $R.j$. As special cases, where op is conjunction, we write $(\forall j : R.j : X.j)$, and where op is disjunction, we write $(\exists j : R.j : X.j)$. Thus, $(\forall j : R.j : X.j)$ may be read as "if $R.j$ is true then so is $X.j$", and $(\exists j : R.j : X.j)$ may be read as "there exists an j such that both $R.j$ and $X.j$ are true". Where $R.j$ is true, we omit $R.j$.

cess policy. Due to the randomization mechanism in Action 1, the maximum length of such a chain is effectively limited to the number of different packet priorities, a typical number is five as in EY-NPMA [12]. In our simulation results, due to randomization in selecting packet-priorities, we do not observe any chain-of-dominance phenomena.

Lemma 2 states that in the absence of faults, starting from initial states, it is always the case that a node k is $locked$ in a CONTROL phase iff there is a $leader$ node j within single-hop of k.

Lemma 2. Let $I1$ denote phase=CONTROL $\wedge(\forall k :: (\exists j : j \in Nbr(k) : j.leader) \iff k.locked))$. $I1$ is an invariant of the BEMA protocol.

Proof. $I1$ holds trivially in the initial states where $\forall j : j.idle$ holds. "$leader$" and "$locked$" states are only modified by Actions 3 and 5. $I1$ is preserved throughout the CONTROL phase since actions 3 and 5 can only be enabled in the DATA phase. $I1$ is also preserved throughout the DATA phases since the only actions that can modify $I1$, namely actions 3 and 5, modify the $leader$ and $locked$ states consistently according to the value of the $remaining\_length$ field. A node j is set to be $leader$ iff the message j broadcasts in the DATA phase of round R has $remaining\_length > 0$, the states for the nodes within single-hop of j in round R are set to $locked$ iff the message j broadcasts has $remaining\_length > 0$. □

Lemma 3 states that if $I1$ holds and there exists a unique $leader$ j within single-hop of k in the beginning of a round, then no other node l, $l \neq j$, within single-hop of k can get access to transmit to the channel in the DATA phase of that round.

Lemma 3 (Leader preservation). If $I1 \wedge (\exists j : j \in Nbr(k) : j.leader \wedge \neg(\exists l : l \in Nbr(k) \wedge l \neq j : l.leader))$ in the beginning of a round, then $j.leader \wedge \neg(\exists l : l \in Nbr(k) \wedge l \neq j : l.leader \vee l.waiting)$ in the DATA phase of the round.

Proof. If $j.leader$ holds in the beginning of a round R for a unique node j within single-hop of k, then due to Lemma 2 $k.locked$ also holds.From Action 6, it follows that k broadcasts a busy signal for the entire duration, Δ, of the CONTROL phase. Note that the only other enabled actions in the CONTROL phase are Actions 1 and 2. Since the contention time of any $candidate$ node l is less than Δ, all such contending nodes would transit to $waiting$ state before the end of CONTROL phase according to Action 1. Therefore, any contending node l for the channel within one hop of node k is deferred from getting access to the channel by k's busy signal according to Action 2 (either busy signal or a collision is delivered at l), and l transits to $idle$ state from $waiting$. □

Lemma 4 states that in the absence of faults, starting from initial states, there can be at most one $leader$ node within single-hop of a node k.

Lemma 4 (At most 1 leader). Let $I2$ denote $(\forall k :: (\forall j, l : j, l \in Nbr(k) : j.leader \wedge l.leader \implies j = l))$. $I2$ is an invariant of the BEMA protocol.

Proof. $I2$ holds trivially in the initial states where $\forall j : j.idle$ holds. Since "$leader$" state is modified only in the DATA phase, $I2$ is preserved throughout the CONTROL phase. $I2$ is preserved throughout the DATA phase due to Lemma 1 and Lemma 3. In a state where $I2$ holds there are two cases (1) $(\forall j : j \in Nbr(k) : \neg j.leader)$ or (2) $(\exists j : j \in Nbr(k) : j.leader \wedge \neg(\exists l : l \in Nbr(k) \wedge l \neq j : l.leader)$. Starting from a state where case 1 holds, Lemma 1 implies that $I2$ is preserved throughout the DATA phase. Starting from a state where case 2 holds, Lemma 3 implies that $I2$ is preserved throughout the DATA phase. □

We now prove Theorem 1, which states that if $I1$ and $I2$ hold (that is, in the absence of faults and starting from the initial states), in the DATA phase of any round there can be at most one node within single-hop of a node k that has access to the channel (in $waiting$ or $leader$ state).

Theorem 1 (No hidden node). $I1 \wedge I2 \implies$ phase=DATA $\wedge(\forall k :: (\forall j, l : j, l \in Nbr(k) : (j.leader \vee j.waiting) \wedge (l.leader \vee l.waiting) \implies j = l))$.

Proof. Proof follows from Lemma 1 and Lemma 3 as follows. $I2$ implies that at any time, there can be at most one node j in the $leader$ state within single-hop of a node k. From Lemma 3, it follows that the existence of such a leader j (j is unique due to $I2$) implies that in the DATA phase $j.leader$ holds, and no other node l can be in $leader$ or $waiting$ state. From Lemma 1, it follows that if no such leader j exists, then in the DATA phase at most one node j can be in the $leader$ or $waiting$ state. □

4.3 Self-stabilization of BEMA

As we prove in Lemma 2 and Lemma 4, in the absence of faults, starting from initial states, $I1$ and $I2$ hold for BEMA and, hence, from Theorem 1 we conclude that BEMA eliminates the node problem. However, due to faults, such as transient memory corruption, message loss, or changes in network topology, $I1$ and $I2$ can be violated. Here, we show that with the addition of a stabilization action, BEMA protocol becomes self-stabilizing, and hence, starting from any arbitrary state, after the faults stop occurring (i.e., no faults occur for a period sufficient enough for stabilization) BEMA starts satisfying its specification, and eliminates collision of DATA packets.

Our stabilization action, Action 7, is enabled in the DATA phase when a node j receives a collision. Since in the absence of faults, starting from initial states, collisions in DATA phase is impossible due to Theorem 1, this action is enabled only from states outside the invariant $I1 \wedge I2$. Upon execution, j transits to $locked$ state to defer any other node within single-hop of j to be able to transit to $leader$ state.

(7) phase=DATA $\wedge j.idle \wedge receive(\pm)$
$\longrightarrow \quad j.status := locked$

Figure 5. Stabilization action for j.

Next we prove Theorem 2 by proving that starting from any arbitrary state BEMA converges to states where $I1$ and $I2$ are satisfied in finite time. More specifically, $I1$ and $I2$ are re-established within at most $max\_message\_length$ rounds, where $max\_message\_length$ denotes the maximum number of packets that a message can span. Note that once the invariant $I1$ and $I2$ is satisfied, Theorem 1 ensures that the hidden-node problem is eliminated in the DATA phase of the subsequent rounds.

Theorem 2 (Self stabilization). BEMA is self-stabilizing.

Proof. Our proof is by demonstrating a variant function g that always decreases outside the invariant states. g is a lexicographical ordering of the tuple $\langle$number of leaders within single-hop of a node k, $remaining\_length\_of\_message\rangle$. We show below that g always decreases until a state where $I2$ is satisfied (i.e., until $g = \langle 1, max\_message\_length\rangle$).

We first show that g cannot increase by considering all possible cases for the status of node k. If $k.leader$ holds, then due to Action 6, k defers any node within single-hop from getting access to the channel to become a $leader$. If $k.locked$ holds, then again due to Action 6, k defers any node within single-hop from getting access to the channel to become a $leader$. If $k.idle$ holds, then due to Action 7 k becomes $locked$, and the problem reduces to the previous case. If $k.candidate$ or $k.waiting$ holds, due to Action 6 k is deferred by one of the leaders within single-hop.

We now show that g decreases due to the $remaining\_length\_of\_message$ that a $leader$ node gets to transmit. Therefore, within at most $max\_message\_length$ rounds, g reduces to $\langle 1, max\_message\_length\rangle$, where $I2$ is satisfied.

Once $I2$ is satisfied, $I1$ is re-established within at most 1 round due to Actions 5 and 7. □

4.4 Extensions

Here, we discuss the extensions to the BEMA protocol for achieving better energy-efficiency and also present a novel ad hoc round-synchronization algorithm for on-the-fly and on-demand round-synchronization.

Energy-efficiency. When a node is listening to the channel, it spends as much energy as transmitting [19]. Therefore, it is important to reduce any idle listening in our MAC protocol. To this end, we assert that when a node j detects that it is not receiving any message transmission in the beginning of a DATA phase, j turns off its radio, sets a timer, and goes to sleep for the rest of the DATA phase. Later, upon expiration of its timer, j wakes up at the beginning of the CONTROL phase if it is a $candidate$ or at the beginning of the DATA phase otherwise.

Similarly, when a contending node j is deferred from access to the channel via Action 2, j turns off its radio and sleeps until the beginning of the DATA phase. In future work, using PowerTOSSIM [22], we will quantify over the energy-savings we achieve by eliminating idle-listening via the above two rules.

Ad hoc round synchronization. By exploiting the structure of the BEMA rounds, we propose an on-the-fly and on-demand round-synchronization protocol here. Since collisions are supposed to happen only

in the control phase, in our ad hoc round synchronization scheme the nodes interpret each collision as the beginning of a control phase. Thus, upon hearing a collision, a node sets its phase to CONTROL, and resets its local round timer. Since the clock skew is negligible for small periods (at most 40 microseconds in a second [17]), synchronizing in the beginning of each round is enough to keep the nodes in tight-synchrony throughout the round.

This scheme converges quickly and achieves round-synchronization for small regions. This ad hoc round-synchronization would be useful for low-power ad hoc networks as it conserves energy by not requiring an always-on global round-synchronization.

In [6], another on-demand round synchronization protocol is presented, which assumes that nodes initiate their round-synchronization around the same times: even though some nodes may start round-synchronization earlier, the remaining nodes are in a "ready" state, anticipating a round synchronization. By way of contrast our ad hoc round synchronization can achieve synchronization from arbitrary states.

5 Simulation results

Here we compare the performance of BEMA with that of BSMA [28], BMMM [25], and CSMA/CA [16]. For our simulations, we use the Prowler wireless sensor network simulation tool [24]. Prowler simulates the radio transmission/propagation/reception delays of Mica2 motes [18], including collisions in ad-hoc radio networks realistically.

In our simulations, we vary the traffic load in a 5-by-5 grid of nodes (a total of 25 nodes) by increasing the number of nodes requesting to transmit data. We measure the cumulative collisions of data packets detected at all nodes and the goodput (bits/sec) as the traffic load varies. We define goodput as the cumulative number of bits received in data packets at all nodes divided by the settling time, where settling time is calculated as the difference between the last time of receive by a node and the first time of send by a node.

We simulated these four protocols using both ideal and realistic Mica2 radio[3]. In the former, the transmissions are free from external influences like noise as well as from multi-path fading. A message in this environment can be lost only when it collides with another message. Thus, in the absence of collisions, all immediate neighbors (up to 8 nodes) of a node j receives j's transmission. Whereas in the latter realistic radio model, the transmissions are subject to Rician fading and multipath interference effects. Moreover, there is a 5% error probability for each message reception.

Table 1 presents the message format of the four protocols we implemented. In BEMA the CONTROL phase is for 100 bit-time, and a data message spans 4 rounds: each of the 4 packets of a message is 960 bits long. Hence the overhead of control packets in BEMA is approximately 100/1060 = 9.4%. The RTS/CTS and other control messages in BSMA and BMMM are of 48 bits length as implemented in SMAC [11]. The data message is send as a 4 back-to-back packets in BSMA and BMMM when a node gets access to the channel. In CSMA there are no control messages, and a message is sent as 4 packets of 960 bits. Since our implementation of BMMM has persistently encountered some deadlocks among transmitters, for our simulations we consider a variation of BMMM, denoted as BMMM', that requires receipt of CTSs from all neighbors—instead of at least one—before initiating a data message transmission. BMMM' avoids deadlocks/livelocks by ordering the resources (i.e., reservation of neighbors) with respect to increasing id. Our simulation codes for all four protocols are available at `http://www.cse.buffalo.edu/~mh69/bema/`.

	Control packets	Data packets	# of packets
CSMA	0 bits	960 bits	4
BSMA	48 bits	4*960 bits	1
BMMM	48 bits	4*960 bits	1
BEMA	100 bits	960 bits	4

Table 1. Message formats of the protocols.

Figures 6 and 7 show the number of cumulative collisions at the receivers for the protocols under ideal and realistic radio, respectively. The results are similar in both graphs with slightly more increased collisions under realistic radio, possibly due to nondeterministic interference among nodes and the error probability. Since CSMA/CA employs no special control messages to prevent collisions, the number of collisions is highest for CSMA due to hidden node problem

[3]Mica2's Chipcon CC1000 radio operates at 433 MHz with a data rate of approximately 40 Kbits/sec.

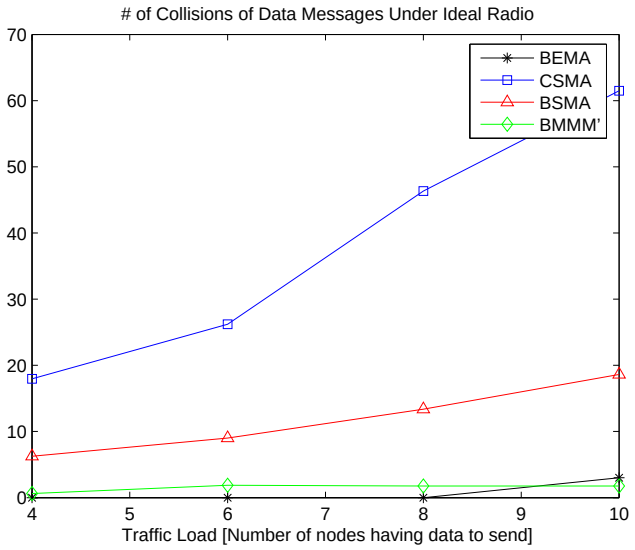

Figure 6. Collision in ideal radio model.

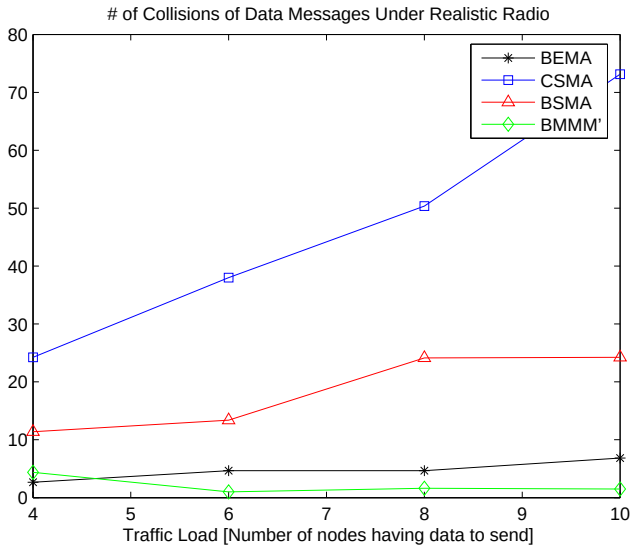

Figure 7. Collision in realistic radio model.

and is linearly increasing with respect to the number of transmitters. BSMA has the next highest number of collisions, again linearly increasing. BMMM' and BEMA have the lowest number of collisions, which are by and large constant with respect to the number of transmitters. The collisions in BEMA may be due to transmitters choosing the same random contention length, or due to *unidirectionality* in some links and non-deterministic interference among nodes.

Figures 8 and 9 show the goodput for the protocols under ideal and realistic radio, respectively. Even though BMMM' guarantees reliable delivery of data to all neighbors (it ensures virtually no collisions), due to the large synchronization overhead and latency it incurs BMMM' has the lowest goodput. The goodput of BSMA is linearly decreasing with respect to the number of transmitters, primarily due to the corresponding linear increase in the number of collisions in BSMA. The goodput of CSMA is constant and high. Even though the number of collisions in CSMA is linearly increasing, the increase in the number of transmitters neutralizes the effects of the former to the goodput since CSMA does not incur any large delays to the transmitters for accessing the channel. Note that the data delivery ratio of CSMA is low due to hidden node induced collisions, however CSMA merely compensates what it lacks in terms of delivery ratio with its bare speed. BEMA has the highest goodput among the protocols. BEMA not only reliably delivers all data by eliminating collisions due to hidden node problem, but at the same time it also scales well with the number of transmitters to provide a constant high goodput rate.

6 Concluding remarks

We presented a simple, light-weight, and self-stabilizing MAC protocol, namely Busy Elimination Multiple Access (BEMA) protocol, for solving the single-hop reliable broadcast problem. BEMA grants on-demand access to the channel —rather than assigning fixed slots as in TDMA based approaches— and also supports prioritization of traffic, thereby providing a useful building block for applications with reliability and quality-of-service requirements. Our simulations show that BEMA has little overhead and provides the highest goodput among BSMA [28], BMMM [25], and CSMA/CA [16], since BEMA successfully and efficiently eliminates the hidden node problem.

We are currently implementing BEMA in TinyOS [14] over the BMAC [19] protocol. In future work, we will investigate the performance improvements BEMA could provide for handling bursty traffic patterns in sensor networks [35] via its hidden node elimination and prioritization scheme. Also, as part of future work, we will work on adopting BEMA in *mobile* ad hoc networks. Its obliviousness to network topology (a node need not know the ids of its neighbors), its simplicity, and its self-stabilization property make BEMA suitable for mobile ad hoc networks.

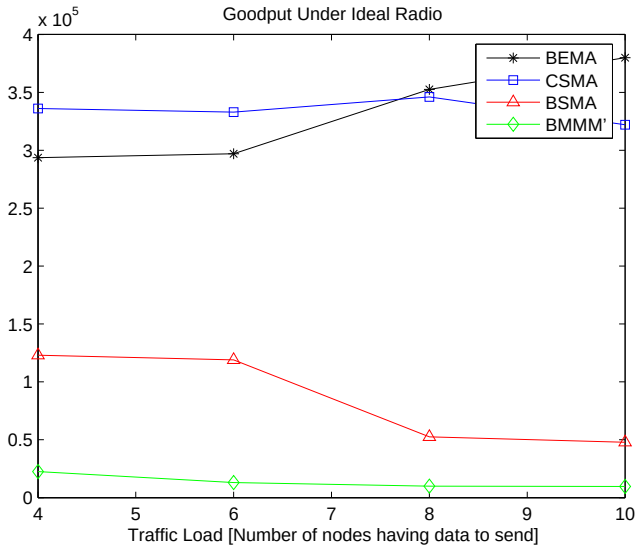

Figure 8. Goodput in ideal radio model.

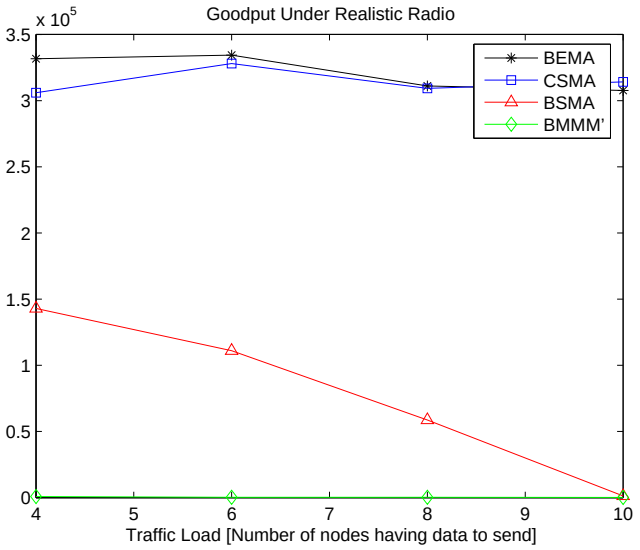

Figure 9. Goodput in realistic radio model.

References

[1] Wireless lan medium access control(mac) and physical layer (phy) specification. IEEE Std 802.11, 1999.

[2] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y-R. Choi, T. Herman, S. S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita. A line in the sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks (Elsevier)*, 46(5):605–634, 2004.

[3] A. Arora and et. al. Exscal: Elements of an extreme scale wireless sensor network. *11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2005.

[4] J. Burrell, T. Brooke, and R. Beckwith. Vineyard computing: Sensor networks in agricultural production. *IEEE Pervasive computing*, 3:38–45, 2003.

[5] C. Busch, M. Magdon-Ismail, F. Sivrikaya, and B. Yener. Contention-free mac protocols for wireless sensor networks. In *DISC*, pages 245–259, 2004.

[6] G. Chockler, M. Demirbas, S. Gilbert, and C. Newport. A middleware framework for robust applications in wireless ad hoc networks. In *Forty-third Annual Allerton Conference on Communication, Control, and Computing*, 2005.

[7] Gregory Chockler, Murat Demirbas, Seth Gilbert, Calvin C. Newport, and Tina Nolte. Consensus and collision detectors in wireless ad hoc networks. In *PODC*, pages 197–206, 2005.

[8] D. Culler, P. Dutta, C. T. Ee, R. Fonseca, J. Hui, P. Levis, J. Polastre, S. Shenker, I. Stoica, G. Tolle, and J. Zhao. Towards a sensor network architecture: Lowering the waistline. *The Tenth Workshop on Hot Topics in Operating Systems (HotOS X)*, 2005.

[9] Shlomi Dolev, Seth Gilbert, Elad Schiller, Alexander A. Shvartsman, and Jennifer L. Welch. Autonomous virtual mobile nodes. In *SPAA*, page 215, 2005.

[10] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Oper. Syst. Rev.*, 36(SI):147–163, 2002.

[11] Deborah Estrin, John Heidemann, and Wei Ye. An energy-efficient MAC protocol for wireless sensor networks. Technical report, October 03 2001.

[12] Etsi. `http://portal.etsi.org/radio/HiperLAN/HiperLAN.asp`.

[13] S. Farritor and S. Goddard. Intelligent highway safety markers. *IEEE Intelligent Systems*, 19(6):8–11, 2004.

[14] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for network sensors. *ASPLOS*, pages 93–104, 2000.

[15] S. S. Kulkarni and M(U). Arumugam. Ss-tdma: A self-stabilizing mac for sensor networks. In *IEEE Press. To appear*, 2005.

[16] W. F. Lo and H.T. Mouftah. Carrier sense multiple access with collision detection for radio channels. In *IEEE 13th Intl. Commun. and Energy Conf.*, 1981.

[17] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. *SenSys*, 2004.

[18] Crossbow technology, mica2 platform. `www.xbow.com/Products/Wireless_Sensor_Networks.htm`.

[19] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 95–107, 2004.

[20] J. Postel. Internet protocol DARPA internet program protocol specification. RFC 791, Internet Engineering Task Force (IETF), September 1981.

[21] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves. Energy-efficient collision-free medium access control for wireless sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 181–192, New York, NY, USA, 2003. ACM Press.

[22] V. Shnayder, M. Hempstead, B. Chen, G. W. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 188–200, New York, NY, USA, 2004. ACM Press.

[23] G. Simon, M. Maroti, A. Ledeczi, B. Kusy, A. Nadas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. *Sensys*, 2004.

[24] G. Simon, P. Volgyesi, M. Maroti, and A. Ledeczi. Simulation-based optimization of communication protocols for large-scale wireless sensor networks. *IEEE Aerospace Conference*, March 2003.

[25] M.-T. Sun, L. Huang, A. Arora, and T.-H. Lai. Reliable MAC layer multicast in IEEE 802.11 wireless networks. page 10, August 2002.

[26] R. Szewczyk, A. Mainwaring, J. Polastre, and D. Culler. An analysis of a large scale habitat monitoring application. *In Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[27] K. Tang and M. Gerla. Mac layer broadcast support in 802.11 wireless networks. pages 544–548, October 2000.

[28] K. Tang and M. Gerla. Random access mac for efficient broadcast support in ad hoc networks. pages 454–459, September 2000.

[29] K. Tang and M. Gerla. Mac reliable broadcast in ad hoc networks. pages 1008–1013, October 2001.

[30] F. A. Tobagi and L. Kleinrock. Packet switching in radio channels: part II the hidden terminal problem in carrier sense multiple-access and the busy-tone solution. *IEEE trans. on commun.*, COM-23:1417–1433, 1975.

[31] Jean Tourrilhes. Robust broadcast : Improving the reliability of broadcast transmissions on CSMA/CA. Technical Report HPL-98-38, Hewlett Packard Laboratories, February 24 1998.

[32] T. van Dam and K. Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 171–180, New York, NY, USA, 2003. ACM Press.

[33] L. F. W. van Hoesel and Paul J. M. Havinga. A TDMA-based MAC protocol for WSNs. In *SenSys*, pages 303–304, 2004.

[34] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 14–27. ACM Press, 2003.

[35] H. Zhang, A. Arora, Y. Choi, and M. G. Gouda. Reliable bursty convergecast in wireless sensor networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 266–276, New York, NY, USA, 2005. ACM Press.

[36] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the first international conference on Embedded networked sensor systems*, pages 1–13, 2003.