# PEDAMACS: Power Efficient and Delay Aware Medium Access Protocol for Sensor Networks

Sinem Coleri Ergen, *Member*, *IEEE*, and Pravin Varaiya, *Fellow*, *IEEE*

**Abstract**—PEDAMACS is a Time Division Multiple Access (TDMA) scheme that extends the common single hop TDMA to a multihop sensor network, using a high-powered access point to synchronize the nodes and to schedule their transmissions and receptions. The protocol first enables the access point to gather topology (connectivity) information. A scheduling algorithm then determines when each node should transmit and receive data, and the access point announces the transmission schedule to the other nodes. The performance of PEDAMACS is compared to existing protocols based on simulations in TOSSIM, a simulation environment for TinyOS, the operating system for the Berkeley sensor nodes. For the traffic application we consider, the PEDAMACS network provides a lifetime of several years compared to several months and days based on random access schemes with and without sleep cycles, respectively, making sensor network technology economically viable.

**Index Terms**—Sensor networks, energy efficiency, delay guarantee.

## 1 Introduction

A wireless sensor network consists of a group of nodes, each node comprising one or more sensors, a processor, a radio, and a battery. These networks may be widely used because of their low cost, small size, and wireless data transfer. Since the radios consume much power, the network's medium access control (MAC) protocol, which determines how the radios are operated, has a decisive influence on battery lifetime. MAC protocols are of two types: random access and time division multiple access (TDMA).

Modifications to the standard random access scheme seek to reduce a node's useless radio activities such as idle channel listening, overhearing packets not intended for itself, and packet transmission collisions. One proposal uses a separate ultra-low power "wake up" radio that constantly listens to the channel and wakes up the main receiver when a packet is about to arrive [1]. Another proposal, the S-MAC protocol [2], uses an "in-band" signaling scheme, based on the RTS/CTS packets as in IEEE 802.11, to reduce overhearing; and periodic listen and sleep modes to decrease idle listening. These proposals achieve power savings up to a factor of 10 over the standard scheme, at a cost of increased hardware or control complexity. The proposals do not improve delay guarantees.

TDMA systems, by contrast, are more power efficient since they allow nodes to enter inactive states until their allocated time slots. TDMA schemes based on scheduling of nodes that are one hop away from the base station [3], [4] are not suitable for multihop sensor networks. On the other hand, schemes that schedule nodes in a multihop network based on graph coloring heuristics to construct minimum length schedules [5], [6], do not propose an energy efficient method to synchronize nodes, discover routes, or to determine interferers beyond the transmission range. Even recent TDMA schemes that emphasize energy efficiency such as TRAMA [7] ignore synchronization inaccuracy and interferers, and trade delay for energy efficiency as in S-MAC.

This paper proposes a very energy efficient scheme that discovers the network topology, and keeps nodes synchronized to properly execute a TDMA schedule. The scheme, PEDAMACS, can be used for multihop networks with one characteristic: All data packets are destined for the same node, called the access point, which has sufficient transmit power to reach all other nodes in one hop. For these networks, PEDAMACS is much better than existing random access protocols in terms of network lifetime and guaranteed delay.

The rest of the paper is organized as follows: Section 2 presents the assumptions necessary for the execution of the protocol. Section 3 describes PEDAMACS. The protocol is extended for more general communication patterns and networks in Section 4. Simulation results comparing the performance of PEDAMACS with different random access networks are in Section 5. Section 6 collects some conclusions.

## 2 Assumptions

Proper execution of the PEDAMACS protocol places the following conditions on transmission power, data traffic, link conditions, and routing:

1. The wireless network consists of one access point (AP) and several sensor nodes. As needed, the AP can reach all the sensor nodes in one hop. The path from a sensor node to the AP comprises several hops. The case when not all nodes are a single hop from one AP is considered in Sections 4.2 and 4.3. Sensor nodes can adjust their transmission power (e.g., Berkeley mica nodes [8]).

• *The authors are with the Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720. E-mail: {csinem, varaiya}@eecs.berkeley.edu.*

*Manuscript received 17 Sept. 2004; revised 26 Jan. 2005; accepted 30 Mar. 2005; published online 16 May 2006.*
*For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0267-0904.*

2. Sensor nodes periodically generate data, possibly at different rates, for transfer to the AP. The scheme easily extends to cases when only some nodes forward data to the AP. The generalization of the protocol for event-driven data generation is described in Section 4.1.
3. Links are bidirectional. This is required for proper functioning of network protocols such as distributed Bellman-Ford algorithms [9]. Bidirectionality is achieved if all sensor nodes transmit at the same power. Differences in actual transmission power due to the hardware differences are compensated by setting up links based on received signal strength as explained in Section 3.2.1.
4. Sensor nodes have low mobility. For high mobility, the scheme may fail to perform well in terms of delay (See Section 5.4).
5. Any routing protocol based on minimizing an additive link cost metric, e.g., [10], can be used in conjunction with PEDAMACS protocol.

# 3 The PEDAMACS Protocol

## 3.1 Transmission Ranges

PEDAMACS nodes have three power levels, $P_l > P_m > P_s$, corresponding to three transmission ranges $r_l > r_m > r_s$. The largest transmission range level, $P_l$, is used only by the AP to broadcast its coordination packets to all sensor nodes in one hop. The smallest transmission range level, $P_s$, is used by all sensor nodes to forward their data packets to the AP over multihop paths. $P_s$ is small to reduce power consumption, but large enough to maintain network connectivity.

When a node transmits its data packet at power level $P_s$, some nodes can decode that packet with an acceptable bit error rate (BER). These nodes are its *neighbors*. Some other nodes receive the packet at a signal level that is too weak to decode, but strong enough to interfere with another signal. These nodes are its *interferers*. The local topology of a node comprises its neighbors and interferers. Nodes transmit at the medium transmission range level, $P_m$, to discover their local topology, as explained later.

## 3.2 Protocol Phases

The protocol operates in four phases: topology learning, topology collection, scheduling, and adjustment. In the topology learning phase, each node identifies its (local) topology, i.e., its neighbors, interferers, and its parent node in the routing tree rooted at the AP obtained according to some routing metric. In the topology collection phase, each node sends its local topology information to the AP so, at the end of this phase, the AP knows the full network topology. At the beginning of the scheduling phase, the AP broadcasts a schedule. Each node then follows the schedule and sleeps during time slots when it is not scheduled to transmit a packet or to listen for one. The adjustment phase is triggered as necessary to learn the local topology information that was not discovered during the topology learning phase or to discover changes.

We now describe each phase in more detail. The appropriate packet structures are displayed in Fig. 1. The basic TinyOS packet has a 5-byte header, a 30-byte data payload, and a 2-byte CRC.

### 3.2.1 Topology Learning Phase

The phase begins when the AP broadcasts (at level $P_l$) a *topology learning* coordination packet to all the sensor nodes. The packet includes *current time* and *next time*. All nodes synchronize with *current time*. They stop transmitting and listen for the next AP coordination packet at *next time*.

Following the topology learning coordination packet, the AP floods the network (at level $P_m$) with the *tree construction* packet. This packet contains the *cost* of the transmitting node in the routing tree, e.g., minimum number of hops to reach AP. Upon reception of a tree construction packet, a node first decides whether it comes from a neighbor or interferer based on the received signal strength according to its interference model. If the transmitting node is a neighbor of the receiving node and is the next hop on a path of smaller cost than previously learned paths, the receiving node updates this *cost* by including its own cost and rebroadcasts the *tree construction* packet. It also keeps this neighbor or interferer node information associated with its cost and received signal strength in an array. At the end of the flooding, it chooses its parent node to be the next hop neighbor on the least cost path to the AP.

Any interference model can be adopted in PEDAMACS. We illustrate one model. The condition for successful reception of packets is that signal-to-interference-plus-noise ratio (SINR) is greater than a threshold $\beta$, which depends on the acceptable BER, detector structure, modulation/demodulation scheme, and channel coding/decoding algorithm. On the other hand, SINR depends on the channel, interference, antenna gain and transmission power. The SINR from node $i$ to node $j$ at shortest range is:

$$SINR_{ij} = \frac{P^{ij}_{r,s}}{I^i_{m,j} + I^i_{l,j} + \sigma^2}, \quad (1)$$

in which $P^{ij}_{r,s}$ is power received at node $j$ from the transmission of node $i$ at transmission power $P_s$, $\sigma^2$ is the receiver thermal noise power, $I^i_{m,j} = \Sigma_{k\neq i,j,|d(k,j)|<=r_m} P^{kj}_{r,s}$ is the interference power at node $j$ from transmitters other than node $i$ inside its medium transmission range and $I^i_{l,j} = \Sigma_{k\neq i,j,|d(k,j)|>r_m} P^{kj}_{r,s}$ is the interference power at node $j$ from transmitters other than node $i$ outside its medium transmission range, where $|d(k,j)|$ is the distance between nodes $k$ and $j$.

Use of the medium range level, $P_m$, allows nodes to determine the interferers inside the medium range and eliminate the term $I^i_{m,j}$. The total interference, $I^i_{tot,j} = I^i_{m,j} + I^i_{l,j}$, is constant for the same configuration of the nodes. Therefore, the larger the ratio $\frac{P_m}{P_s}$ the system uses, the higher is the probability of correct reception of packets due to the increase in $\frac{I^i_{m,j}}{I^i_{l,j}}$ but the larger is the delaythe system experiences due to the increasing number of interferers in the system. The effect of this ratio on the delay and the number of data packets successfully reaching the AP is examined in Section 5. (Previously proposed TDMA protocols assume that $P_m = P_s$ so $I^i_{m,j} = 0$).

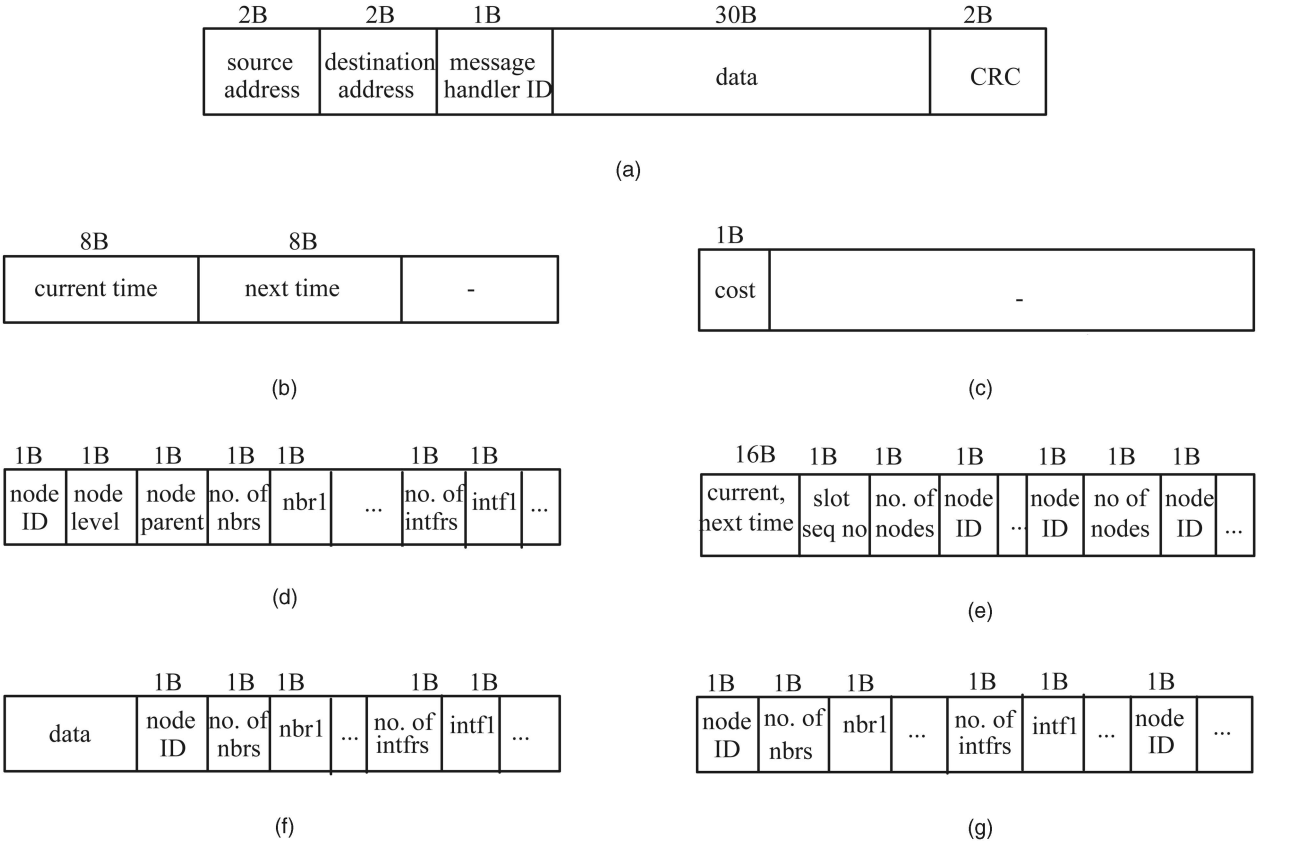


Fig. 1. (a) TinyOS packet structure. (b)-(g) Packet structures used in the four protocol phases, excluding the packet header and CRC. (b) Topology learning, topology collection, and adjustment coordinated packets. (c) Tree construction packet. (d) Local topology packet. (e) Scheduling packet. (f) Data packet. (g) Adjustment topology packet.

Since the reception power at medium range transmission $P_{r,m}^{ij}$ is related to that at shortest range transmission $P_{r,s}^{ij}$ by $P_{r,s}^{ij} = P_{r,m}^{ij} \frac{P_s}{P_m}$, the SINR from node $i$ to node $j$ at shortest range can be calculated from

$$SINR_{ij} = \frac{P_{r,m}^{ij} \frac{P_s}{P_m}}{I_{l,j}^{i} + \sigma^2}. \quad (2)$$

If the medium range is large enough to cover almost all interferers, one can ignore the term $I_{l,j}^{i}$ and calculate $SINR_{ij}$ based on received signal strength $P_{r,m}^{ij}$ upon reception of the tree construction packet. If $SINR_{ij} > \beta$, then nodes $i$ and $j$ are neighbors at the shortest range; otherwise, they are interferers. A more elaborate interference model in which subsets of interferers can transmit together based on the $\beta$ value that can be tolerated is beyond scope of this paper.

Choosing $\beta$ large enough is important in providing reliable routing paths in the network since the nodes are to be scheduled without retransmission during the scheduling phase. To get an estimate of $\beta$, we measured successful reception probability and asymmetry of the links between Berkeley mica2dot motes as a function of the received signal strength. Fig. 2 shows that the reception probability is above 0.95 if the receive signal strength is above a certain value, which is $-85dBm$ in this case. Fig. 3 shows the asymmetry of the links at different received signal strengths. The asymmetry of a link $(i, j)$ is measured by the absolute difference between the reception probabilities at node $i$ and $j$. We observe from the figure that if the signal strength is larger than a certain value, which is $-80dBm$ in this case, the links can be considered bidirectional. Since there are no interfering nodes in this experiment, comparing $SINR$ to the threshold $\beta$ is equivalent to comparing the received signal strength at medium range to another threshold:

$$P_{r,m}^{ij} \geq \alpha \frac{P_m}{P_s}, \quad (3)$$

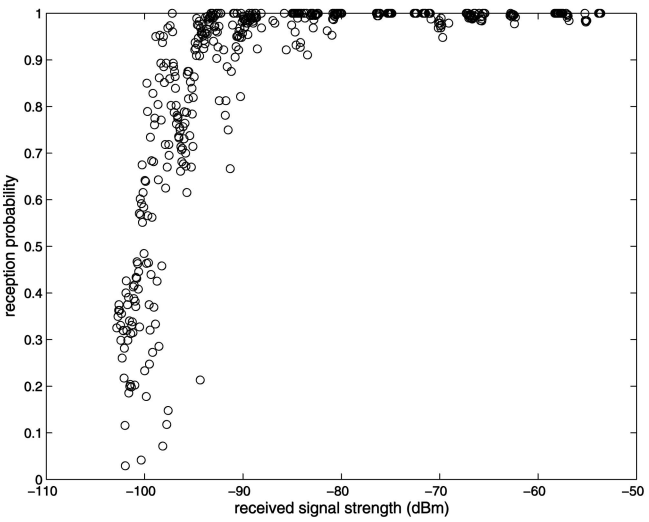


Fig. 2. Reception probability versus received signal strength for Berkeley mica2dot motes.

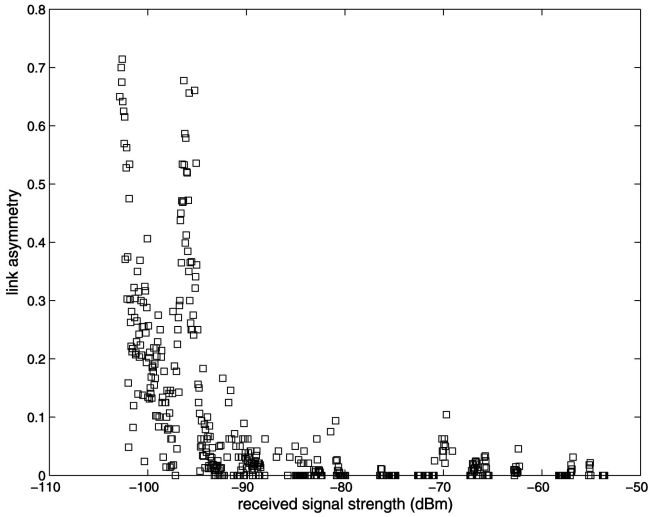


Fig. 3. Link asymmetry versus received signal strength for Berkeley mica2dot motes.

where $\alpha = \beta\sigma^2$ is equal to the threshold for good link quality at shortest transmission range, which is $10^{-8}mW$ in this case. Simulations in Section 5.6 gives an estimate for $\frac{P_m}{P_s}$ that eliminates the interference. A more elaborate scheme can be used for the cases where the medium transmission range is not large enough: $\alpha$ should be replaced by $\alpha + I\beta$, where $I$ is the upper bound on $I^i_{l,j}$. This is beyond the scope of this paper.

A random access scheme is used in the topology learning phase, because the nodes do not (as yet) have a transmission schedule. The scheme is designed so that, at the end of this phase, almost all nodes can correctly determine their neighbors and interferers with high probability. We adopt a carrier sense multiple access (CSMA) scheme similar to IEEE 802.11 [11]. The nodes listen for a random time before transmitting, and then transmit if the channel is idle. A random delay is added before carrier sensing to further reduce collisions.

#### 3.2.2 Topology Collection Phase

At the end of this phase, the AP has complete topology information. The phase starts with a broadcast by the AP of the *topology collection* packet at the *next time* announced in the *topology learning* packet. This packet, too, contains both the *current time*(for synchronization) and the *next time* at which the AP will broadcast the next coordination packet.

Upon receiving the *topology collection* packet broadcast by the AP, each node transmits its *local topology* packet, listing the node's parent, neighbors, and interferers, to its parent using the shortest range transmission.

The topology collection phase also uses a CSMA scheme. However, because a collision will lead to the destruction of the local topology information of at least two nodes, the CSMA scheme by itself cannot guarantee that the AP will receive the full topology information. So, the scheme is modified to include an implicit acknowledgement, which occurs when a node detects the transmission by its parent node to the latter's parent (packets from nodes at level 1 are explicitly acknowledged by AP, which retransmits all the packets it has received), or an explicit acknowledgement as in IEEE 802.11 [11].

#### 3.2.3 Scheduling Phase

The AP explicitly schedules all the nodes, based on its knowledge of the complete network topology. The scheduling frame is divided into time slots. A slot extends the packet duration by a guard interval to compensate for synchronization errors.

At the beginning of the phase, the AP broadcasts the *scheduling* packet. As with other coordination packets, this packet contains the *current time* for synchronization, and the *next time* when the next coordination packet will be broadcast, in addition to the schedule.

At the beginning of the scheduling frame, each node generates *data* packets, which are sent to the AP according to the schedule using the shortest transmission range level, $P_s$. The data packets include the data that are to be sent to the AP, together with any new topology information, consisting of the nodes' neighbors and interferers, discovered during the adjustment phase since the last topology learning and collection phases. If the packet length is not enough to carry all new topology information, this information is included in a round robin fashion in each data packet. The length of the data field in the packet depends on the application.

The schedule determines the time slots when the nodes are allowed to transmit. When a node receives a packet, it does not attempt to transmit it immediately. Instead, it enqueues the packet and sleeps until its next scheduled time slot. The scheduling algorithm ensures that all packets reach the AP by the end of the scheduling phase. The algorithm used here can be based on the coloring of the original conflict graph [5] for a general communication traffic or a linearized version of the conflict graph [18] for all-to-one traffic with the AP as the data sink. Details of the algorithms are not included in this paper.

The guard interval is a small fraction of the total slot duration. Since the AP reaches all the nodes with the same packet, the error in synchronization from the delay between time-stamping and sending the packet at the transmitter is eliminated. Since the range of the AP is on the order of hundreds of meters, the propagation delay is also negligible (a few $\mu$sec). Based on the assumption that all the nodes run the same software, all the nodes timestamp the packet at the same time. Therefore, the only error of synchronization in this application comes from clock skew, the difference in the clock tick rates of the nodes. Typical clock drift of a sensor node in 1 sec is 30-50 $\mu$sec. If the packet generation period at each node is 30 sec, the maximum clock drift will be 0.9-1.5 msec compared with 14 msec, the duration of the packet transmission of one 37-byte TinyOS packet at 50 kbps [8]. Additional coordination packets can be transmitted by the AP between the beginning and end of the scheduling phase to decrease this guard interval even further.

#### 3.2.4 Adjustment Phase

This phase is included at the end of the scheduling phase to identify the complete network topology and to detect the movement of nodes or the addition of new nodes as needed depending on the application. If the number of successfully

scheduled nodes is not 100 percent, this means that some conflicting nodes may have been scheduled for the same slot during the scheduling phase due to incorrect topology information at the AP. However, restarting the topology learning phase may cause a delay for the packets of successfully scheduled nodes if the percentage of these nodes is not too low. The adjustment phase helps the protocol to update the schedule for small changes in the topology without restarting the topology learning phase. Another reason for the decrease in the percentage of successfully scheduled nodes may be the unstable links between the nodes and their parents. This can be handled by generating redundant data in the network or sending the data over multiple paths and is the subject to future research.

The adjustment phase begins when the AP broadcasts to all the sensor nodes an *adjustment* coordination packet. The packet includes *current time* for synchronization and *next time* for the broadcast of the next coordination packet.

The *backoff window size* is calculated by subtracting the packet transmission time and guard interval from the length of the adjustment phase, which is equal to the difference between *next time* and *current time*. The nodes wait for a random time chosen from the backoff window size and transmit their *adjustment topology* packet with the new topology information if the channel is idle. Meanwhile, they can receive other nodes' packets.

The adjustment phase uses medium transmission range level, $P_m$, to detect interferers and neighbors. If the nodes detect any new neighbors or interferers in this phase, they include this information in their data packets transmitted during the scheduling phase. They also include this information in their adjustment topology packets in the next adjustment phase so that the information is guaranteed to reach the nodes that can successfully send their data packets to the AP during the scheduling phase. Based on this information, the AP may update the routing paths and/or correct the schedule if necessary.

## 4 Extension of PEDAMACS

Up to now, we have assumed that there is only one AP in the network and every sensor node periodically generates data for transfer to the AP. The system framework, however, is quite flexible and can be generalized in many ways. Sections 4.1, 4.2, and 4.3, respectively, indicate extension of the algorithm to handle nonperiodic data generation, existence of more than one AP, and nodes located beyond the longest range of the AP.

### 4.1 Event-Driven Sensing

Suppose sensor nodes generate data packets to be transferred to an AP only upon the occurrence of an event such as fire, security breach. This network can still use TDMA scheduling with slight changes on the use of time slots assigned to each node.

The main idea of extending the scheduling algorithm for periodic data generation to event-driven sensing applications is that the slots assigned to the nodes do not have to be used. When there is no event, the nodes only wake up to receive the scheduling packet to keep synchronized and to check whether there is any transmission in the slots they are assigned to receive a packet. During these slots, they only listen to the channel for the duration of the guard interval plus the length of preamble. If there is a packet, they continue to listen, otherwise they put their radio back in sleep mode. When there is an event, they use their assigned slot to transmit their own packet and the other nodes forward their packets even if they do not detect any event.

### 4.2 More than One AP

Consider a network with more than one AP and sensor nodes that periodically generate data for transfer to one of the APs. Including more than one AP in the system increases the coverage of the network, brings scalability to the system and makes the system distributed by eliminating single point of failure, which is the AP.

The PEDAMACS system requires two modifications to include more than one AP. First, the coordination packets of neighboring APs should not be transmitted at the same time to avoid the interference at the sensor nodes. Second, the APs should take into account the interferers that are outside their longest range while generating the schedule.

After distributing the APs, the largest transmission power level, $P_l$, of each AP should be adjusted so that all the sensor nodes in the network can be reached by at least one of them. If not all nodes can be reached, the extended version of PEDAMACS described in Section 4.3 should be used. The APs avoid interference at the sensor nodes from the transmission of more than one coordination packet by generating schedules among themselves. The schedules consists of time slots that are long enough to carry one coordination packet.

The protocol that assigns time slots and power levels to the APs and time slots to the sensor nodes then operates as follows: At the beginning, the time slots for the APs are adjusted so that only one AP transmits a coordination packet at a specific time slot. In the topology learning phase, the APs transmit their topology learning coordination packet in their prespecified time slots. The coordination packets include additional fields for the ID of the AP and the time for the end of the transmission of coordination packets. A sensor node will hear zero, one or more than one topology learning coordination packet. Each node then retransmits the tree construction packet if it arrives on the minimum cost path from an AP that contains it inside the longest range if the node has heard from one or more APs, or if it is coming on the minimum cost path from any AP if the node did not hear from any AP. The node then determines its neighbors, interferers and next hop on the minimum cost paths to each of these APs. In the topology collection phase, each node sends its topology information to each of the APs it is connected to on the corresponding minimum cost path while choosing a subset of them for sending data packets during the scheduling phase.

Based on the topology information about the number of the nodes inside the transmission range of more than one AP or no AP at all, the APs can decide to decrease or increase their transmission power and restart the topology learning and collection phases. When the APs are satisfied with their coverage, the conflicting APs are determined to

be the ones that have common sensor nodes inside their longest range.

The APs are assigned colors so that no two conflicting APs are assigned the same color (e.g., a simple algorithm that constructs a $\Delta + 1$-coloring in at most $n$ steps is given in [12] for $\Delta$ and $n$ being the maximum degree of a vertex and the number of vertices respectively in the graph with vertices as APs and edges between the conflicting APs). The APs assigned to the first color then find the schedule and broadcast this information to sensor nodes and neighboring APs. Neighboring APs with the next color assign the time slots to the nodes inside their range, taking into account the already assigned common nodes, which are defined to be the nodes inside their range and the range of at least one of the APs of previous colors, and the interferers inside the range of another AP. The APs corresponding to each color consider the schedules of the APs of the previous colors in this way.

Since the length of the schedules are only affected by the common nodes with the conflicting APs and the interference coming from neighboring APs and the schedules of sufficiently separated APs are not affected from each other, the resulting system is scalable.

### 4.3 Handling Nodes Outside the Range of AP

Consider a network in which some sensor nodes cannot hear a coordination packet from the AP. Extending PEDAMACS to include these nodes in TDMA schedule increases the coverage of the network at the cost of increased synchronization overhead and delay.

PEDAMACS requires modifications to handle the discovery of nodes outside the range of the AP and relay of scheduling coordination packets for their synchronization and TDMA schedules. During the topology learning phase, even if the nodes do not hear the topology learning coordination packet, they can still be discovered. Upon reception of a tree construction packet, a node retransmits it if it arrives on the minimum cost path from an AP while determining its neighbors and interferers. When it starts to hear local topology packets, the node understands that the topology collection phase has started and transmits its local topology packet to the AP on the minimum cost path. Inside the local topology packet, the node also indicates that it did not hear the coordination packets to inform the AP and their ancestors in the tree that it is outside the range of the AP. In the scheduling phase, the scheduling packets need to be replicated for the nodes that cannot hear the coordination packets at the parent of each of these nodes. This new network structure is in fact a combination of a pure PEDAMACS network and a pure multihop network.

Apart from an increase in the synchronization overhead, the system will experience greater delay and energy consumption compared to the pure PEDAMACS. Denote by $\delta$ the clock offset resulting from the difference between the time-stamping of the packet at the transmitter and receiver. Call the extra depth of the routing tree at the nodes that cannot hear the coordination packets $l$. At the end of synchronization, the maximum clock offset among the sensor nodes is $(l+1)\delta$. Therefore, the guard interval at the beginning of each slot should be increased by $l\delta$ (The clock drift is already included in the guard interval and does not depend on the nodes outside the range of the AP). If the slot duration that does not consider the nodes outside the longest range of the AP is $t_s$, the new schedule length will be $t_s + l\delta$, which causes an increase of a factor of $\frac{t_s+l\delta}{t_s}$ in the delay and listening energy consumption.

## 5 Simulation

The purpose of our simulations is to evaluate the effectiveness of PEDAMACS protocol by examining each phase separately, comparing it with protocols having different energy conserving features in terms of delay and energy consumption, and analyzing the effect of interference on the performance of the protocol.

The simulation environment is TOSSIM [13], a discrete event simulator for TinyOS [8], the operating system developed for the Berkeley sensor nodes. In the simulations, the nodes are randomly distributed in a circular area of radius 100 units. The transmission range is chosen to be slightly larger than the threshold necessary for network connectivity [14]. The results discussed below are averages of the performance of ten different random configurations, unless otherwise stated. Variations around the averages are presented in [18].

The window sizes and the delays are given in units of *bit time*—the radio tick period, so the absolute time delay for any data rate is the product of the number of bit times and the radio tick period. The sensor network lifetimes are estimated for a 50 kbps transmission rate.

Shortest path routing is used in the simulations. The average depth of the resulting routing trees is 4.4, 5.2, and 7 for 20, 30, and 60 nodes, respectively; correspondingly the average number of neighbors is 4.6, 5.0, and 5.5. The data packet length is 37 bytes, the control packet length is 10 bytes. The schemes assume that $\frac{r_m}{r_s} = 1$ unless otherwise stated to make a fair comparison between the existing protocols that do not take interference into account.

Section 5.1 aims to understand the effect of different random access schemes in reaching the nodes in the network in the topology learning phase. Section 5.2 describes the simulation of different CSMA schemes based on implicit and explicit acknowledgement for the topology collection phase. Section 5.3 gives the performance of the topology adjustment phase. Section 5.4 compares the delay and energy performance of PEDAMACS with different random access schemes. Section 5.5 focuses on self-configuration speed of PEDAMACS. Section 5.6 proves the advantage of taking interferers into consideration in PEDAMACS in terms of the number of packets successfully received at AP.

### 5.1 Topology Learning Phase

A random access scheme similar to IEEE 802.11 protocol [11] is used in this phase because the nodes do not (as yet) know their topology information. Before transmitting a packet, the node waits for the channel to be idle for a certain time, randomly chosen from the *backoff window size*. The backoff window size is chosen large enough to create a phase difference between the packet transmissions. The node then chooses another random listening time from

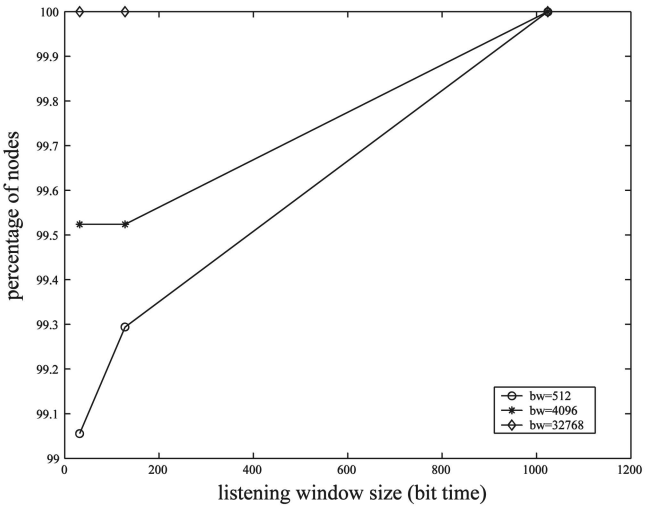


Fig. 4. The number of nodes reached by flooding as a percentage of the number of nodes that are theoretically reachable for 30 nodes.

*listening window size* and decreases it by one at each radio clock tick, generated at the transmission rate (e.g., 50 kbps), so long as the channel is idle for the last Inter Packet Interval (IPI) time (corresponding to the Inter Frame Space (IFS) in 802.11). The listening window size is chosen large enough to avoid collisions. The node starts to transmit when the listening time decreases to zero. Meanwhile, the node can receive another packet and return to the same state it has left.

Fig. 4 shows the percent of nodes that the *tree construction* packet reaches in flooding for different backoff and listening window sizes. All nodes are reached if the backoff window size is large enough. Even with small window sizes, the tree construction packet reaches more than 99 percent of the nodes. This means that the average number five of neighbors is large enough so that if a node does not receive a flooding packet from one of its neighbors, it can still connect to AP via other neighbors. The delay increases from 10 to 100 kbit times as the backoff window size increases, equivalent to 0.2-2 sec at 50 kbps [18].

### 5.2 Topology Collection Phase

The random access scheme used in the topology learning phase is augmented with an acknowledgement for the topology collection phase to guarantee the successful arrival of all packets in the network.

The schemes use either implicit or explicit acknowledgement. For both types of schemes, we choose the backoff window size large enough to "break" transmission synchronization and to enable the reception of 100 percent of nodes' packets at the AP.

The implicit acknowledgement algorithm, denoted *implicit random* in the figures below, is based on listening to the packets transmitted by the parent. The nonacked packets are retransmitted after a certain timeout time, called *acknowledgement window size*. If the window size is too small, the nodes will increase the load in the network by resending the packets even though these packets are still in the queue awaiting transmission. If the window size is too large, the nodes will wait for an unnecessarily long time. We assume that the system can adaptively adjust the acknowledgement window size to get the minimum delay and we use this value for comparison to other protocols.

The explicit acknowledgement algorithm, denoted *IEEE 802.11* in the figures, adopts RTS/CTS/DATA/ACK mechanism used in IEEE 802.11 [11]. RTS/CTS control packets are used to acquire the channel before data packet transmission and include a duration field that indicates how long the remaining transmission will be. So, if a node receives an RTS or CTS packet destined for another node, it puts its radio in sleep mode and does not transmit during this time. This is called virtual carrier sense and overhearing avoidance. Physical carrier sense is performed at the physical layer by listening to the channel for a randomized carrier sense time, similar to the CSMA scheme described in Section 5.1. The backoff and listening window sizes are chosen to be smaller than those used in implicit acknowledgement scheme by a factor of the ratio of data packet length to RTS control packet length for a fair comparison.

Fig. 6 shows that the delay experienced by explicit acknowledgement is slightly smaller than that for the implicit acknowledgement scheme since the former knows immediately whether the transmission is successful whereas the latter has to wait for the acknowledgement window size. Also, Fig. 9 shows that the explicit acknowledgement scheme slightly reduces the energy consumption in *listening* and *reception* by putting the radio in sleep mode during neighboring nodes' transmissions while causing an increase in *transmission* energy through RTS/CTS/ACK control packets. The resulting lifetime shown in Fig. 8 is almost the same for the implicit random and IEEE 802.11 schemes. Therefore, the explicit acknowledgement scheme, which does not require any adaptive scheme for acknowledgement window size adjustment, is better for topology collection phase.

### 5.3 Adjustment Phase

Simulations show that the number of successfully scheduled nodes increases from 90 to 95 percent as backoff window size increases. This can be increased further by enlarging the backoff window size enabling the nodes to hear from a larger number of their neighbors by increasing the number of successful transmissions. Another alternative is to start the adjustment phase to help the nodes learn about their remaining neighbors and interferers.

Fig. 5 shows the rate of detecting these neighbors from the start as a function of the time for consecutive adjustment phases. The time at which the nodes learn about all of their neighbors is almost independent of the backoff window size whereas the rate of increase is larger for smaller window size. The total time to discover the neighbors, which is 5.5 on average, is 40 kbit times, which is less than 1 sec at 50 kbps.

### 5.4 Comparison of PEDAMACS with Existing Protocols

This section provides a quantitative measure of improvement of the PEDAMACS scheme over existing schemes in terms of delay and energy consumption.

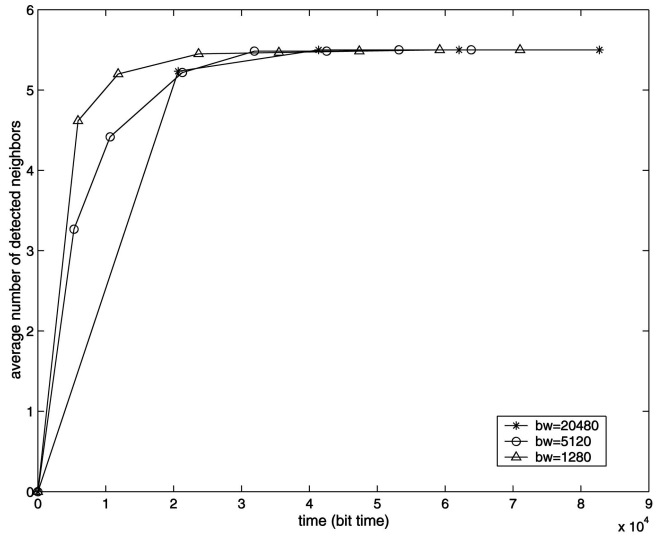


Fig. 5. The average number of neighbors that are discovered as time evolves for different backoff window sizes corresponding to different adjustment phase lengths for a 60-node random network.

We compare PEDAMACS with five existing schemes: *implicit random*, *IEEE 802.11*, *smac 50%*, *smac 10%*, and *TDMA*. *Implicit random* and *IEEE 802.11* refer to the random access schemes with implicit and explicit acknowledgements respectively described in Section 5.2. The only difference here is that they are used to send data packets instead of topology packets in random access networks.

SMAC [2] is a MAC protocol designed for energy efficiency of sensor networks. It provides low-duty cycle operation of each node by periodic sleeping. Although periodic sleeping trades latency for energy conservation, the adaptive listening reduces this cost by enabling each node to switch mode according to the traffic in the network. IEEE 802.11 is equivalent to SMAC without sleeping. We simulated SMAC for 50 and 10 percent duty cycles, denoted by *smac 50%* and *smac 10%* in the figures.

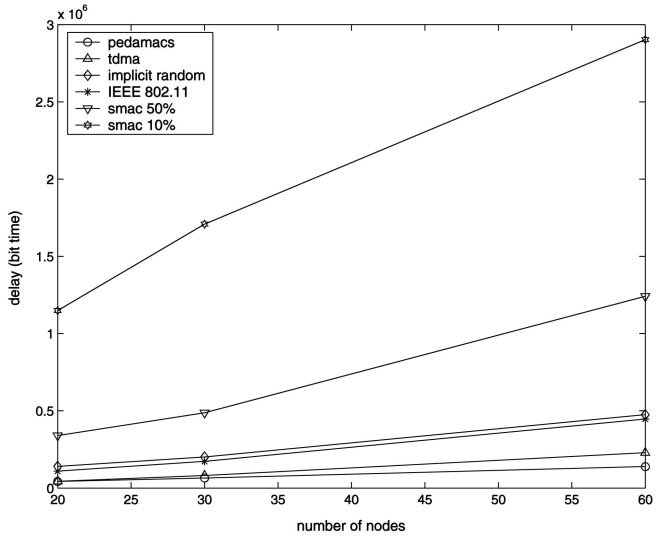


Fig. 6. Comparison of the delay of PEDAMACS with competing schemes for different number of nodes.

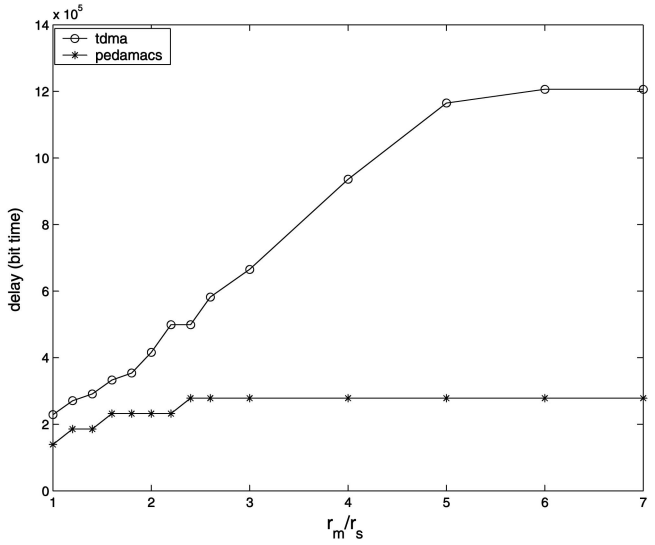


Fig. 7. Comparison of the delay of PEDAMACS with a TDMA scheme as a function of the ratio of medium transmission range to shortest range for a 60-node random network.

A TDMA scheme, denoted by *tdma*, is the PEDAMACS protocol based on the coloring of the original graph [5] whereas *pedamacs* is the PEDAMACS protocol based on PEDAMACS scheduling described in [18].

### 5.4.1 Comparison of Delay

Fig. 6 shows the delay comparison of PEDAMACS with existing protocols for different number of nodes. IEEE 802.11 provides slightly smaller delay compared to implicit random access schemes. SMAC increases the delay by a factor of 2-3 and 7-10 for 50 and 10 percent duty cycles, respectively, over the delay of IEEE 802.11. The reason for this increase is the early sleeping problem, which is defined as nodes going to sleep when a neighbor still has messages for it. Timeout-MAC (TMAC) [15] tries to solve this problem by dynamically ending the listening part of each period. The latency of TMAC is lower bounded by that of IEEE 802.11. However, the variance of this latency increases as the duty cycle increases: The length of the period is lower bounded by $l_p = \frac{l_l}{d_c}$, where $l_l$ is the minimum length of the listening part, i.e., it should be long enough to receive at least the start of the CTS packet, and $d_c$ is the duty cycle. In the simulations, $l_l = 10^4$ *bit times* so if at least one node that will forward a packet goes to sleep, the delay increases by $10^5$ and $10^6$ *bit times* over the IEEE 802.11 delay for 10 and 1 percent duty cycles, respectively.

For a 60-node network the average delay of IEEE 802.11 scheme is nearly $5 \times 10^5$ bit times, which is about 10 sec for a 50 kbps transmission rate. Taking the random variation in the actual delay into account may make a random access scheme unsuitable for the traffic application, which generates data every 30 sec.

The delay experienced by PEDAMACS is slightly smaller than the TDMA scheduling algorithm. The difference between them increases as the number of nodes increases. Fig. 7 shows the increase in the delay of PEDAMACS and TDMA as a function of the $\frac{r_m}{r_s}$ ratio for a 60-node random network.

TABLE 1
Power Consumption of Basic Operations in Berkeley Mica Nodes

| operation | power consumption |
|---|---|
| transmitting one packet | $0.92mJ$ |
| receiving one packet | $0.69mJ$ |
| listening to channel | $29.71mJ/sec$ |
| operating radio in sleep mode | $15\mu J/sec$ |
| sampling sensor | $1.5\mu J$/sample |

### 5.4.2 Comparison of Power Consumption

The sensor node power-consuming operations considered are transmission and reception of a packet, listening to the channel, sampling, and running the microprocessor. Table 1 gives power consumption figures for the Berkeley mica nodes [8].

The network lifetime is estimated from the average energy consumed at sensor nodes, assuming a 50 kbps transmission rate and the use of a pair of AA batteries, which can supply 2,200 mAh at 3V, at each node. A better estimate of the lifetime for a specific application can be performed by considering the routing protocol and the communication between the nodes upon the death of a node (e.g., [17] takes into account the fact that nodes closer to AP forward more packets).

Fig. 8 gives the lifetime estimates of PEDAMACS and existing protocols for 128 Hz sampling rate at each node. We have chosen 2-minute packet generation period in order to guarantee the successful arrival of packets within the period since maximum delay in Fig. 6 is $3 \times 10^6$ bit times, equivalent to 60 sec at 50 kbps, for 60-node *smac 10%*. We have also included *smac 1%* since an adaptive S-MAC may be successful to decrease this delay.

The lifetime of random access schemes, *implicit random* and *IEEE 802.11*, is about ten days, whereas the lifetime of SMAC protocol increases up to 60 days for 10 percent duty cycle. The lifetime of PEDAMACS system, on the other hand, is about 1,200 days. The reason for the dramatic difference becomes clear from Fig. 9, which compares the power consumed by these schemes in different operations for a 60-node random network. The primary cause is in the total energy consumed by the radio in "listening" and "sleeping" modes. *smac 10%* can decrease this energy by a factor of 10 whereas PEDAMACS decreases it by a factor of more than 1,000.

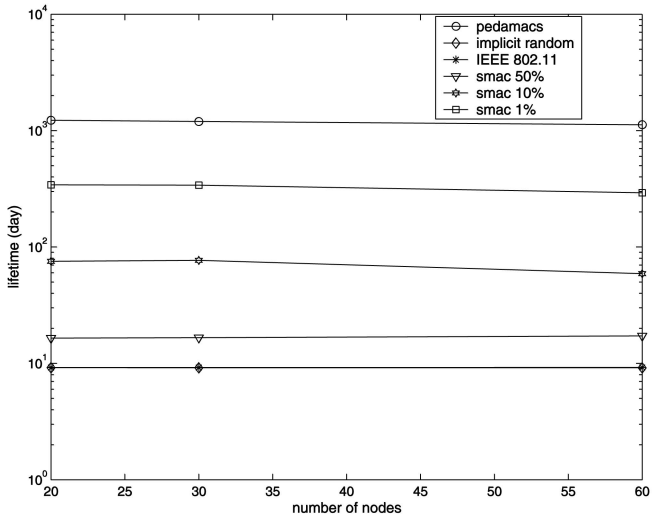


Fig. 8. Comparison of the lifetime of PEDAMACS with competing schemes for different number of nodes.

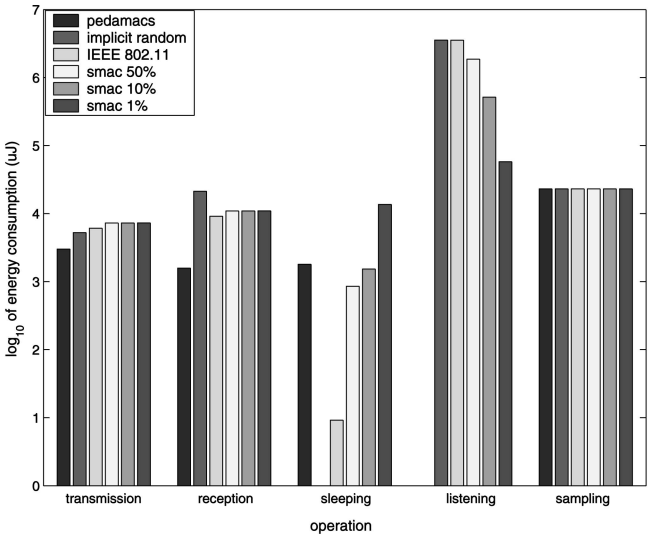


Fig. 9. Comparison of power consumption in a PEDAMACS network versus contention networks for different node operations. Note that the vertical axis is $\log_{10}$.

The difference in lifetimes also arises from differences in the amount of energy spent in transmission and reception. The extra transmission energy spent in *implicit random* over PEDAMACS is due to retransmission as a result of collision. *IEEE 802.11* spends even more energy in transmission for RTS, CTS, and ACK control packets. *smac 50% and 10%* add the energy spent in the transmission of synchronization messages to that of *IEEE 802.11*.

The average receive energy differs because of the "overhearing effect:" In random access schemes, when one node transmits a packet, all neighboring nodes receive it whereas only the parent of that node receives the packet in PEDAMACS (the other neighbors are in sleep mode). The difference is largest for *implicit random* since the neighboring nodes listen to whole packets whereas *IEEE 802.11* and *SMAC* slightly reduces this by transmitting shorter *RTS/CTS* packets.

The lifetime estimate of PEDAMACS in the simulation is based on the assumption that topology discovery is performed if fewer than 90 percent of the nodes are successful at transmitting to the AP and there are multiple adjustment phases of a couple of seconds just after the topology discovery phase and one adjustment phase every 10 minutes afterwards. The ratio of rescheduling, induced by sensor movement and link fluctuations, that PEDAMACS can tolerate before it loses its advantage in energy savings over *smac 50%* is calculated to be 53 percent of the total time, which is equivalent to taking data and topology information for one period by using *IEEE 802.11* and using PEDAMACS schedule in the next period.

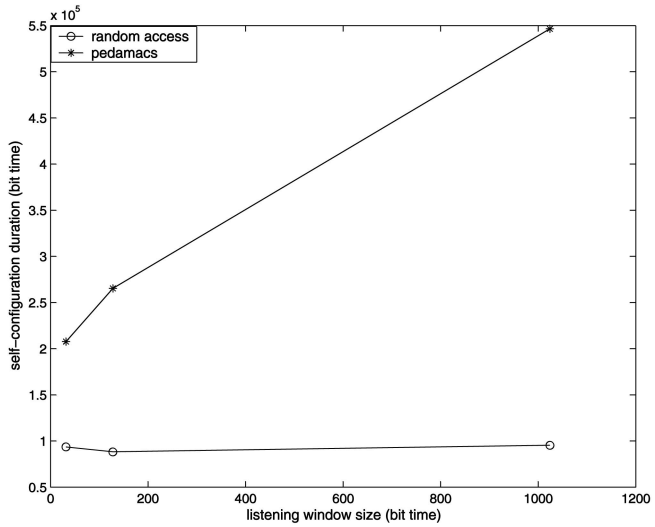


Fig. 10. Comparison of the self-configuration duration of PEDAMACS with random access scheme for different listening window sizes and backoff window size of 32,768 bit-times.

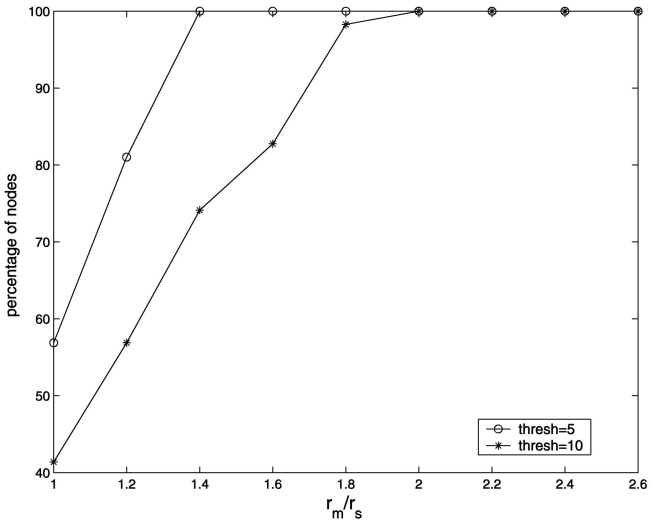


Fig. 11. The number of nodes whose packets successfully arrive AP as a percentage of the number of nodes that are scheduled for a 60-node random network.

Fig. 8 is for a network with a 50 kbps rate and a 2-minute packet generation period. If the rate decreases, the delay experienced will increase in inverse proportion. However, the transmission energy consumption per bit may decrease or increase depending on the hardware, modulation, coding and transmission distance [16]. Furthermore, energy consumed in listening to the channel and sampling is constant for different rates.

### 5.5 Self-Configuration Speed of PEDAMACS

The topology update mechanism for random access schemes and PEDAMACS are the same although the response time to the topology changes are different. Tree construction packets are transmitted either periodically or upon the failure of receiving from more than a specific percentage of nodes in both schemes. However, in the case of random access schemes, the time duration from the topology failure until nodes learn about their local topology and the routes to the AP is the time until the tree construction packet reaches all the nodes. On the other hand, in PEDAMACS, the nodes additionally have to send their topology information back to the AP and get their schedules. Fig. 10 shows that PEDAMACS increases self-configuration duration by a factor of 2 to 5 over a random access scheme.

### 5.6 Advantage of Considering Interference

Traditional TDMA schemes are based on the assumption that the nodes interfering with a receiver are within its transmission range, called *shortest range* in this paper. However, the power needed for disrupting a packet reception is much lower than that of delivering a packet successfully. This section shows the necessity of considering the interferers within larger range, which is called *medium range* here. In simulations, we assume that the receiving signal power is inverse proportional to $d^4$, where $d$ is the distance between transmitter and receiver, and ignore the thermal noise since it is much smaller than the interference signal.

Fig. 11 examines the number of packets successfully received at the access point as a function of the $\frac{r_m}{r_s}$ ratio for a 60-node random network. The reception of packets with a SINR value smaller than threshold, called "thresh" in the graph, is considered not to be successful. As $\frac{r_m}{r_s}$ increases, the number of successfully received packets increases. The value of the ratio for which all transmissions are successful is 2 for this case, where the nodes are randomly distributed and the average number of neighbors at shortest range is 5.5. As the density increases, the graph is expected to shift to the right.

## 6 Conclusion

In sensor networks, measurements made at the nodes must be transferred to a distinguished node, which we call an access point (AP). The MAC protocol for a sensor network is decisive in determining network performance in terms of power consumption and total delay.

We consider a special class of sensor networks with two distinguishing characteristics. First, AP has unlimited power so that packets broadcast by AP can reach all other nodes in one hop, whereas packets from the latter must travel over several hops to reach AP. Second, the nodes periodically generate packets. These two characteristics are exploited by the PEDAMACS protocol to *schedule* transmissions while providing synchronization of the nodes, discovery of the routing paths, and determination of the interferers beyond the transmission range in an energy efficient manner.

For the application considered here, the PEDAMACS network provides a lifetime of several years compared to several months and days based on random access schemes with and without sleep cycles respectively. Moreover, the PEDAMACS protocol guarantees bounded delay and eliminates congestion. Of course, only some applications can use a PEDAMACS network. However, the system is quite flexible and can be generalized in many ways. We have shown the generalization of PEDAMACS to handle

event-driven data generation, existence of more than one AP and the nodes located outside the range of the AP.

## ACKNOWLEDGMENTS

This work was supported by the US National Science Foundation under Grant CMS-0408627 and California Department of Transportation.

## REFERENCES

[1] C. Guo, L.C. Zhong, and J.M. Rabaey, "Low-Power Distributed MAC for Ad Hoc Sensor Radio Networks," *Proc. Internet Performance Symp. (Globecom 2001),* Nov. 2001.
[2] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Sleeping for Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 12, no. 3, June 2004.
[3] P.J.M. Havinga and G.J.M. Smit, "Energy Efficient TDMA Medium Access Control Protocol Scheduling," *Proc. Asian Int'l Mobile Computing Conf. (AMOC 2000),* Nov. 2000.
[4] E. Uysal-Biyikoglu, B. Prabhakar, and A. El Gamal, "Energy-efficient Packet Transmission over a Wireless Link," *IEEE/ACM Trans. Networking,* vol. 10, no. 12, pp. 487-499, Aug. 2002.
[5] R. Ramaswami and K.K. Parhi, "Disributed Scheduling of Broadcasts in a Radio Network," *IEEE INFOCOM Conf.,* vol. 2, pp. 497-504, 1999.
[6] A. Bhatnagar and T.G. Robertezi, "Layer Net: A New Self-Organizing Network Protocol," *Proc. IEEE Military Comm. Conf. (MILCOM),* vol. 2, pp. 845-849, Oct. 1990.
[7] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems,* Nov. 2003.
[8] J. Polastre, R. Szewczyk, C. Sharp, and D. Culler, "The Mote Revolution: Low Power Wireless Sensor Network Devices," *Proc. Hot Chips 16: A Symp. High Performance Chips,* Aug. 2004.
[9] S. Narayanaswamy, V. Kawadia, R.S. Sreenivas, and P.R. Kumar, "Power Control in Ad-Hoc Networks: Theory, Architecture, Algorithm and Implementation of the COMPOW Protocol," *Proc. European Wireless Conf.,* Feb. 2002.
[10] J.H. Chang and L. Tassiulas, "Maximum Lifetime Routing in Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 12, no. 4, Aug. 2004.
[11] LAN-MAN Standards Committee of the IEEE Computer Society, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification," IEEE Std 802.11-1997, 1997.
[12] S.T. Hedetniemi, D.P. Jacobs, and P.K. Srimani, "Fault Tolerant Distributed Coloring Algorithms that Stabilize in Linear Time," *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS),* Apr. 2002.
[13] "Using TOSSIM Simulator to Develop TinyOS Components," www.tinyos.net/tinyos-1.x/doc/tutorial/lesson5.html, May 2006.
[14] B. Krishnamachari, S.B. Wicker, and B. Bejar, "Phase Transition Phenomena in Wireless Ad-Hoc Networks," *Proc. Symp. Ad-Hoc Wireless Networks (GlobeCom 2001),* Nov. 2001.
[15] T. van Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems,* Nov. 2003.
[16] S. Cui, A.J. Goldsmith, and A. Bahai, "Energy-Constrained Modulation Optimization," *IEEE Trans. Wireless Comm.,* 2005.
[17] S. Coleri, M. Ergen, and T.J. John Koo, "Lifetime Analysis of a Sensor Network with Hybrid Automata Modelling," *Proc. ACM Int'l Workshop Wireless Sensor Networks and Applications,* Sept. 2002.
[18] S. Coleri, "PEDAMACS: Power Efficient and Delay Aware Medium Access Protocol for Sensor Networks," technical report, Dept. of Electrical Eng. and Computer Science, Univ. of California, Berkeley, Dec. 2002.

**Sinem Coleri Ergen** received the BS degree in electrical and electronics engineering from Bilkent University, Ankara, Turkey, in 2000, and the MS and PhD degrees in electrical engineering and computer sciences from the University of California Berkeley (UCB) in 2002 and 2005. Since January 2006, she has been a postdoctoral researcher in electrical engineering and computer sciences at UCB. Her research interests are in wireless communications and networking, collaborative signal processing, and intelligent transportation systems. She is a member of the Sensor Networks for Traffic Monitoring project at UCB. She received a Regents Fellowship from the University of California Berkeley in 2000 and Bilkent University Full Scholarship from Bilkent University in 1995. She is a member of the IEEE.

**Pravin Varaiya** is Nortel Networks Distinguished Professor in the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley. From 1975 to 1992, he was also a professor of economics at Berkeley. From 1994 to 1997, he was the director of the California PATH program, a multiuniversity research program dedicated to the solution of California's transportation problems. His current research is concerned with communication networks, transportation, and hybrid systems. He has taught at MIT and the Federal University of Rio de Janeiro. Dr. Varaiya has held a Guggenheim Fellowship and a Miller Research Professorship. He received an Honorary Doctorate from L'Institut National Polytechnique de Toulouse, and the Field Medal of the IEEE Control Systems Society. He is a fellow of IEEE and a member of the National Academy of Engineering. He is on the editorial board of several journals, including *Discrete Event Dynamical Systems* and *Transportation Research—C*. He has coauthored three books and more than 250 technical papers. The second edition of *High-Performance Communication Networks* (with Jean Walrand) was published by Morgan-Kaufmann in 2000. *Structure and Interpretation of Signals and Systems* (with Edward Lee) was published by Addison-Wesley in 2003. He is a member of the board of directors of Sensys Networks.