UC Berkeley
Working Papers

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

Permalink
https://escholarship.org/uc/item/1870b4g7

Authors
Coleri, Sinem
Varaiya, Pravin

Publication Date
2004-07-01

eScholarship.org Powered by the California Digital Library
University of California

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

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

Sinem Coleri, Pravin Varaiya
University of California, Berkeley

California PATH Working Paper
UCB-ITS-PWP-2004-6

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department Transportation, Federal Highway Administration.

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California. This report does not constitute a standard, specification, or regulation.

Report for Task Order 5301

July 2004

ISSN 1055-1417

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

Sinem Coleri and Pravin Varaiya
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720
Email: {csinem, varaiya}@eecs.berkeley.edu

Abstract

We consider a class of sensor networks with two special characteristics. First, the nodes periodically generate data for transfer to a distinguished node called the access point. Second, the nodes are (transmit) power and energy limited, but the access point, which communicates with the 'outside world', is not so limited. Such networks might be used for instance when a geographically distributed physical process, such as traffic on a freeway or at an urban street intersection, is periodically sensed for purposes of process control. We propose a medium access control scheme, called PEDAMACS, for this special class of networks.

PEDAMACS uses the high-powered access point to synchronize the nodes and to schedule their transmissions and receptions in a TDMA manner. The protocol first enables the access point to gather topology (connectivity) information. A scheduling algorithm then determines when each node should transmit its data, and the access point announces the transmission schedule to the other nodes. The scheduling algorithm ideally should minimize the delay—the time needed for data from all nodes to reach the access point. However, this optimization problem is NP-complete. PEDAMACS instead uses a polynomial-time scheduling algorithm which guarantees a delay proportional to the number of nodes in the sensor network.

Because PEDAMACS schedules node transmissions, its performance is much better than that of protocols designed for more general contention (or random access) networks in terms of power consumption, delay, fairness, and congestion control. The comparison is 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.

1 Introduction

A wireless sensor network consists of a group of nodes, each comprising one or more sensors, a processor, a radio, and a battery. Because of their low cost, small size, and wireless data transfer, these networks might be widely used

in the future. Unlike general wireless networks, however, wireless sensor networks are designed and deployed for specific applications.

In transportation, for example, measurements of traffic at several locations on a freeway or at an urban street intersection are transmitted every 30 sec. to the access point on the side of the freeway or intersection. The access point, in turn, transfers the data to the traffic management center (TMC) over a phone line. For real time traffic control the data must be received at the access point within a prescribed time delay. In the traffic application, the sensor nodes would have limited (transmit) power and energy, but the access point, which communicates with the TMC, would not be so limited.

Thus we study the class of networks (or applications) with two characteristics: The nodes periodically generate data and have limited energy and power, but the access point (AP), which 'consumes' these data, is not energy or power limited. Consequently, packets from nodes must travel over several hops to reach the access point, but packets from the access point can reach all nodes in a single hop.

In a sensor network most of the battery energy is consumed by the radio. So the network's medium access control (MAC) protocol, which determines how the radios are operated, has a decisive influence on battery lifetime. This lifetime determines the cost of network maintenance, and hence whether it is economical to deploy such a network.

The paper presents a MAC protocol, called PEDAMACS, for these networks. The protocol first enables the access point to gather information about the network topology. It then calculates and broadcasts a periodic *schedule*, which determines when each node should listen for incoming packets and when it should transmit its own packets or those received from 'upstream' nodes; the rest of the time, the node 'sleeps'. When a change in network topology is detected, the access point repeats the process of topology discovery and schedule determination.

The optimum schedule is the one that minimizes delay—the time needed for the data generated in one period from all nodes to reach the access point. This optimization problem is NP-complete. Therefore, instead of finding the optimum schedule, PEDAMACS incorporates a polynomial-time scheduling algorithm that achieves a delay proportional to the number of nodes of the network.

It is not surprising that for our traffic application PEDAMACS performs better than a contention-based or random access protocol. The latter is more flexible because it may be used even when the data are generated at random time instances rather than periodically.

What *is* surprising, perhaps, is the dramatic quantitative reduction in power consumption in a PEDAMACS vs. a random access network. For this comparison, we take as values of the power consumed in various operations (transmit, listen, sleep, sampling, and so on) to be those for the Berkeley mica node, [3]. (See table 1 in section 7.4 below.) We implement the protocols using the TinyOS operating system, and conduct a Monte Carlo simulation of the two protocols in TOSSIM, the TinyOS simulation framework. We assume that a node has a pair of AA batteries, which can supply 2200 mAh at 3 V.

Figure 1 suggests that, for an application with a 2-minute packet generation period, a PEDAMACS network will last 1200 days, whereas even a contention network based on 10% sleep schedules will only last 60 days while increasing the delay considerably without any delay guarantee.

The packet generation period in a traffic application is 30sec. A contention network, with even shorter lifetime, is

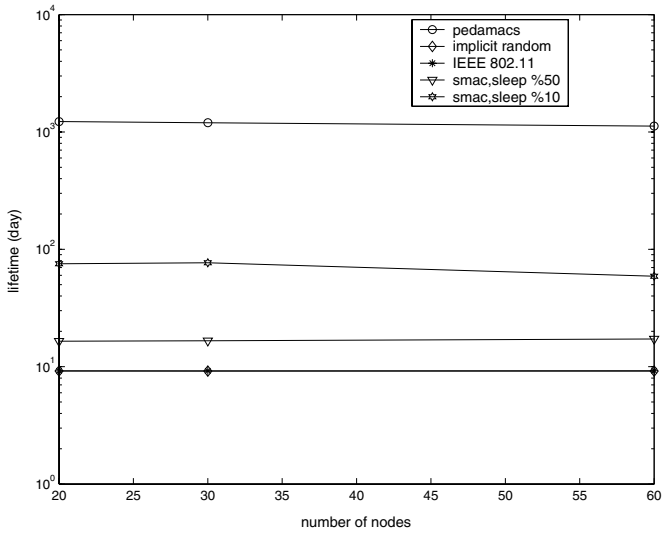

Figure 1: Comparison of the lifetime of PEDAMACS with competing schemes for different number of nodes.

therefore not suitable for the traffic application. The lifetime of a PEDAMACS network for 30-sec packet generation period and 60 nodes, however, is calculated to be around 770 days. If this can be increased by a factor of four, which appears possible, PEDAMACS system would be competitive with current traffic measurement equipment, which lasts almost 10 years without maintenance.

Alternative approaches The big disadvantage of a random access network is that its nodes consume a lot of power because they constantly listen to the channel, overhear the packets that are not destined for them, and may have their transmissions collide. Several proposals have been advanced to reduce these effects by putting the radio in sleep mode. The proposals differ in their radio-wakeup algorithms, the extra hardware they need, and the transmission of control packets that do not carry data.

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 [4]. If a neighbor wants to transmit a packet, it first sends a wake-up beacon over a separate wake-up channel to trigger the power-up of the normal (data) radio and then sends the packet over the data channel. The scheme avoids overhearing and idle listening, but it does not overcome collisions. Moreover, the difference in the transmission range of the two radios may pose significant problems.

The PAMAS protocol [5] also avoids overhearing among neighboring nodes by using a separate signaling channel to power off nodes that are not listening or transmitting as in [4]. Simulations reported for networks of 10-20 nodes indicate power savings of $10-70\%$ as compared with a similar protocol without the separate channel.

The S-MAC protocol [6] is a MAC protocol designed for sensor networks that reduces overhearing by an 'in-band' signaling scheme, based on the RTS/CTS packets as in IEEE 802.11. When an interfering node hears a RTS or CTS packet, it goes into sleep mode. Idle listening is avoided through periodic listen and sleep modes, whose schedules are known by neighbors. Simulations in Section 7.4 give power savings of a factor of $8-10$ while causing an increase of a factor of $6-10$ in the delay for 10% sleep schedules. STEM [7] saves energy at the expense of latency,

through listen/sleep modes as in [6], but using a separate radio and a paging channel.

PEDAMACS is a Time Division Multiple Access (TDMA) scheme. In contrast to random access networks, TDMA systems are more power efficient since they allow the nodes in the network to enter inactive states until their allocated time slots. The TDMA schemes that have been proposed in [9, 10, 11] are all based on forming clusters, within which all nodes are one hop away from the cluster-head and use TDMA schedules. However, the overhead of forming clusters and inter-cluster communication may significantly reduce the advantages of a 'pure' TDMA scheme that PEDAMACS adopts. The LEACH protocol in [10] achieves as much as a factor of 8 reduction in energy dissipation compared to conventional routing protocols by combining lossy compression and direct data transmission to AP at cluster-heads. The direct transmission from sensor nodes may not be feasible since the sensor nodes are generally transmission power limited. Also, when the lossy compression is not tolerable, the power saving can decrease to a factor of 2.

TDMA systems for the nodes that are single hop away from the base station such as HIPERLAN/2 [12] are based on the allocation of time slots in the reverse channel depending on allocation request received from the nodes. This allows devices to enter inactive states when not scheduled for the time slot to conserve power. However, direct transmission from all sensor nodes to the base station may not be feasible since they are all power and energy limited.

TDMA schedules for multi-hop radio networks are based on assigning time slots to the nodes based on graph coloring heuristics to construct minimum length schedules [13, 14, 15, 16]. These TDMA schedules activate each link or node only once during each TDMA frame after a topology discovery phase. Thus the schedules do not take into account the increased data flows over wireless links closer to the access point, thereby increasing the delay in transferring data from nodes further away from the access point. Moreover, these studies are based on limited transmission range assumption where no interference is caused beyond that range. A more general TDMA scheme that takes into account the nodes whose signal level is too weak to be decoded but strong enough to interfere with another signal is needed.

Random access networks all have the advantage of accommodating random access. At the cost of considerable increase in hardware or control complexity, they achieve power savings up to a factor of ten. On the other hand, the TDMA schemes (except for some cluster based schemes) do not take advantage of the fact that all sensor data are destined for a single access point and introduce distributed synchronization overhead. For the limited application contexts for which it is suitable, these schemes cannot compete with the power savings of PEDAMACS.

The rest of the paper is organized as follows. Section 2 summarizes standard performance metrics for sensor networks. Section 3 presents the assumptions necessary for formulating the problem. Section 4 describes the PEDAMACS protocol. The scheduling algorithm is in section 5. Simulation results comparing the performance of PEDAMACS with different random access networks are in section 7. Section 8 collects some conclusions.

2 Performance metrics

The following performance metrics are often used to compare sensor network MAC protocols. PEDAMACS protocol aims at performing well in terms of all of these metrics without sacrificing one for the other.

Power efficiency The radio is the biggest consumer of energy. Energy is wasted in collisions (requiring packet retransmission), idle listening (even when no packets are received), overhearing packets not intended for the listening node, and transmission of control packets that do not carry data.

Real-time guarantees When data received by the access point are used to control a physical process (as in the traffic application), a guaranteed bound on the delay is necessary for effective control action. Without such a bound, the protocol may not be used.

Congestion control Contention networks must control the packet generation rate at a node based on feedback from the network. The control ensures that nodes near the access point receive more bandwidth because they carry more traffic than other nodes. Congestion control affects both the delay and the power consumed per successful transmission. If traffic at a node exceeds the bandwidth allocated to it, packets will be dropped, wasting the energy consumed in bringing the packet to that node, and triggering delay-increasing retransmission.

Fairness In each application, the packets are generated at a certain rate at each node. The MAC protocol should schedule the transmissions such that the transmission of none of the packets of a node is sacrificed for the sake of transmitting another node's packets. Random access is known to favor the transmission of the packets closer to the AP as shown in [8].

3 Assumptions

In this section, we provide the assumptions underlying this study.

1. Consider a wireless ad hoc network that consists of one access point (AP) and several sensor nodes that generate data for transfer to the AP. There is no wireline infrastructure to connect these nodes.
2. Each node is supported by an omnidirectional antenna.
3. Sensor nodes are static. Although the system can still perform better than previously proposed schemes in terms of energy consumption for the case of low users' mobility, it may fail performing well in terms of other metrics such as delay (See Section 7.4).
4. Sensor nodes are capable of adjusting their transmission power (e.g. Berkeley mica nodes [3]).
5. The transmission power of all the sensor nodes are the same across the network although it may change over time. This provides proper functioning of our protocol by achieving the bi-directionality of links. Since all physical paths from node i to node j can be reversed, be they multi-path or reflection, and the attenuation is the same in either direction, it follows if two nodes i and j transmit at the same power, then if node i can hear node j, node j can also hear node i. Bidirectional links are also needed for the proper functioning of some network protocols such as distributed Bellman-Ford algorithms [17].
6. The AP can reach all the sensor nodes (in one hop) since it is not energy or power limited. The path from a sensor node to the AP comprises several hops, because these nodes have limited power and energy. The case where it is not possible to reach all the nodes with one AP is considered as an extension in Section 6.3.

7. Nodes periodically generate data packets of fixed length at the same rate. The straightforward extension of the algorithm for different packet generation rates is given in Section 6.1. Since the periodicity assumption brings delay guarantee and energy efficiency together, we will show how to use this algorithm for the event-driven sensor detection, where the nodes send data only when there is an event, as an extension in Section 6.2.
8. We assume that the data generated at each node is not correlated with other nodes so that they should all be transmitted to the AP without data aggregation. This assumption helps us compare PEDAMACS with other MAC protocols as a first step of the study and gives us worst case lifetime one can get from a network operating on PEDAMACS. Including data aggregation within the proposed framework is out of the scope of this paper and is a subject of future research.
9. Routing is not considered in this study. Any routing protocol to optimize any power-aware metric can be used in conjunction with PEDAMACS protocol.
10. Any interference model can be adopted in PEDAMACS implementation to determine the nodes that can communicate with a node and the nodes that interfere with it.

4 The PEDAMACS protocol

4.1 Transmission ranges

A PEDAMACS network employs three transmission ranges. The longest range reaches all sensor nodes in one hop. The AP uses it to broadcast its coordination packets. The transmission power used by the AP to achieve longest range is denoted by P_l. The AP can also decrease its transmit power to help the sensor nodes determine their next hop.

Sensor nodes use the shortest range to transmit data to the AP. The range should be as short as possible to minimize power consumption, while maintaining network connectivity. A sensor node also uses a medium transmission range to identify its interferers in addition to its neighbors, where *neighbor* is defined to be a node that can be reached at shortest range with an acceptable BER (bit error rate) and *interferer* is defined to be a node whose signal level is too weak to be decoded but strong enough to interfere with another signal when it transmits at shortest range. The transmission power used by the nodes to reach shortest range and medium range are called P_s and P_m respectively.

4.2 Protocol phases

The protocol operates in four phases: the topology learning phase, the topology collection phase, the scheduling phase and the adjustment phase. In the topology learning phase, each node identifies its (local) topology information, i.e. its neighbors and its 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 this 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: In particular, the node sleeps when it is not scheduled either to transmit a packet or to listen for one. The adjustment phase is included if necessary to learn the local

topology information that has not been discovered in topology learning phase or that has changed depending on the application and the number of successfully scheduled nodes in scheduling phase.

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

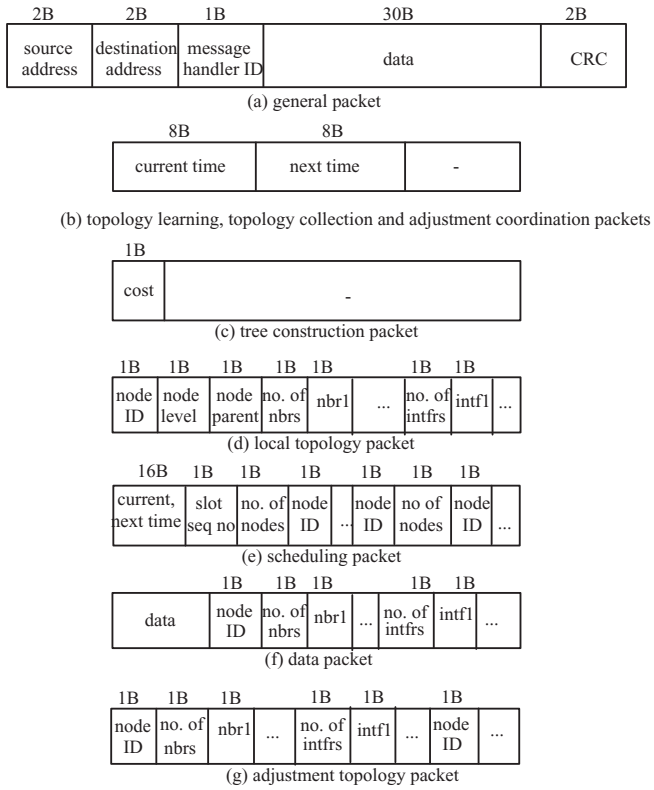

Figure 2: (a) TinyOS packet structure, (b)-(g) packet structures used in the four protocol phases, excluding the packet header and CRC.

Topology learning phase The phase begins when the AP broadcasts (using the longest transmission range) to all the sensor nodes a *topology learning* coordination packet (figure 2(b)). The packet includes *current time* and *next time*. All nodes synchronize with *current time*. They stop transmitting and listen for the next AP coordination time in the future at *next time*.

Following this coordination packet, the AP floods the network with the *tree construction* packet (figure 2 (c)), using the medium range transmission. 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 some 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 neighbor that is next hop on the least cost path to the AP.

Any interference model can be adopted in PEDAMACS implementation. One of these is now described for illustration purposes. The condition for successful reception of packets is that signal-to-interference-and-noise ratio (SINR) is greater than SINR threshold β, where β is a pre-specified threshold that depends on the acceptable BER, detector structure, modulation/demodulation scheme, and channel coding/decoding algorithm. On the other hand, SINR depends on the channel, multiuser interference, antenna gain and transmission power. The SINR from node i to node j at shortest range is given as follows:

$$SINR_{ij} = \frac{P_{r,s}^{ij}}{I_j^i + \sigma^2} \tag{1}$$

where $P_{r,s}^{ij}$ is power received at node j from the transmission of node i at transmission power P_s, σ is the receiver thermal noise power and I_j^i is the interference power at node j from transmitters other than node i, which is given by $\Sigma_{k \neq i,j} P_{r,s}^{k,j}$.

We assume that no interference is caused beyond the medium transmission range in contrast to the assumption of the previously proposed TDMA protocols that no interference is caused beyond the shortest transmission range. Thus, the ratio $\frac{P_m}{P_s}$ can be chosen arbitrarily large compared to 1 in previous studies. The larger the ratio, the higher the probability of correct reception of packets but the larger the delay the system experiences due to the increasing number of interferers in the system. The effect of this ratio on the number of data packets successfully reaching the AP and on the delay is examined in Section 7.

Since all the system interferers are assumed to be known and 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 by the following equation upon reception of the tree construction packet:

$$SINR_{ij} = \frac{P_{r,m}^{ij} \frac{P_s}{P_m}}{\sigma^2} \tag{2}$$

If $SINR_{ij} > \beta$ then the nodes i and j are neighbors at the shortest range otherwise they are interferers. More involved interference model where some subsets of interferers together are allowed in the system based on the β value that can be tolerated is out of scope of this paper and is subject to future work.

A random access scheme is used in this 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. The simulations reported in section 7.1 adopt a CSMA scheme. 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.

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 (figure 2 (b)) 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 (figure 2 (d)), listing the node's parent, neighbors, and interferers, using the shortest range transmission to its parent.

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 [2]. Section 7.2 compares these two acknowledgement schemes in terms of delay and power consumption.

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 is larger than the packet duration by a guard interval to compensate the clock drifts.

At the beginning of the phase, the AP broadcasts the *scheduling* packet (figure 2 (e)). As with the 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 one *data* packet (figure 2 (f)), which is sent to the AP according to the schedule using the shortest transmission range. The data packet includes the data that is to be sent to the AP and the new topology information that consists of the nodes and their neighbors and interferers discovered during the adjustment phases 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 the next scheduled time slot. The scheduling algorithm ensures that all packets reach the AP by the end of this phase.

Guard interval is assumed to be a small percentage of the total slot duration. Typical clock drift of a sensor node in $1sec$ is $10\mu sec$. If the packet generation period at each node is $30sec$, the maximum clock drift will be $0.3msec$ compared to $14msec$, the duration of the packet transmission of 37 byte TinyOS packet at $50kbps$ [3]. 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.

Adjustment phase This phase is included at the end of the scheduling phase to learn complete topology of the network, to detect the movement of the 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. 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. Adjustment phase helps the protocol to update the schedule for the 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 subject to future research.

The phase begins when the AP broadcasts to all the sensor nodes an *adjustment* coordination packet (figure 2(b)). 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 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 that contains new topology information (figure 2 (g)) if the channel is idle. Meanwhile, they can receive other nodes' packets.

Adjustment phase is used to detect interferers and neighbors, therefore the transmission range is medium. 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 this information is guaranteed to reach the nodes that can successfully send their data packets to the AP during the scheduling phase. The AP then receives this information and checks and corrects the schedule if necessary.

System performance improves with an increase in the proportion of the number of scheduling phases to topology learning and collection phases, because the latter packets do not carry data. This proportion will increase with the increase in the duration for which the information gathered in the topology collection phase is valid. If the proportion of the scheduled nodes whose packets are correctly received at the AP during the latest scheduling phase is greater than or equal to some pre-determined threshold (which may be set to 100 percent), AP sends a special scheduling coordination packet to resynchronize the sensor nodes and to tell them to implement the same schedule used in the previous scheduling phase. However, if the proportion is below this threshold, the AP first assumes that some nodes may not have heard the scheduling coordination packet so it retransmits this coordination packet. If the proportion is still below the pre-determined threshold, it assumes that the AP's topology information have become invalid either because, in the topology learning phase, some nodes failed to hear from their neighbors or interferers, or because of a change in network topology as a result of a change in the radio channel. At that time, depending on the number of successfully scheduled packets, AP can restart the topology learning phase by transmitting another topology learning coordination packet if the percentage of successful delivery is not high enough or can start adjustment phase otherwise.

PEDAMACS performs well in terms of the metrics introduced in section 2. The length of the schedule provides an upper bound on the delay between acquisition of data by the nodes and their reception by the AP. Energy is conserved because (in the scheduling phase) nodes sleep unless they are transmitting or receiving. Congestion control is achieved because data generation occurs at the beginning of the scheduling phase, and all packets are received at the AP by the end of the scheduling frame. Fairness is realized, because in one scheduling frame all the packets are transmitted to the AP without sacrificing one for another.

5 The schedule

After it gathers the full topology information, the AP executes the scheduling algorithm. The algorithm assigns a group of non-conflicting nodes to transmit in each time slot, in such a way that one data packet from each node reaches the AP by the end of the scheduling frame. This section explains the algorithm. The transmission and network model is described in section 5.1, the complexity analysis of the scheduling problem is in section 5.2, and the PEDAMACS algorithm is in section 5.3. Some special cases are analyzed in section 5.4.

5.1 Network and transmission model

At the beginning of the scheduling phase, the network may be represented by a graph $G = (V, E)$, in which V is the set of nodes, including the access point AP. The (undirected) edges $E \subset V \times V$ are the (transmission) links to be scheduled. The graph forms a *tree*, rooted at AP. All traffic is destined for AP, so every data packet at a node is forwarded to the node's parent.

A node may interfere with another node, so these nodes should not transmit simultaneously. The *interference* graph $C = (V, I)$ is assumed known (at the beginning of the scheduling phase). Here $I \subset V \times V$ is the set of edges such that $(u, v) \in I$ if either u or v can hear each other or one of them can interfere with a signal intended for the other (even if they cannot hear each other). So, if u is transmitting, v should not be scheduled to receive from another node at the same time. I therefore consists of the links between each node and its neighbors and interferers except the node's parent.

The *conflict* graph corresponding to $G = (V, E)$ and $C = (V, I)$ is the graph $GC = (V, EC)$ in which EC comprises the edges between node pairs that should not transmit at the same time. EC contains two kinds of edges. First, if $(i, j) \in E$, $(i, j) \in EC$, because a parent node and a child node cannot transmit at the same time. Second, if $(i, j) \in I$ or $(i, j) \in E$ and c_j is a child of j in G, $(i, c_j) \in EC$: Because i and j interfere, if i is transmitting, the child c_j of j cannot transmit at the same time because j would hear from both i and c_j. Figure 3 illustrates the relation between G, C, GC.

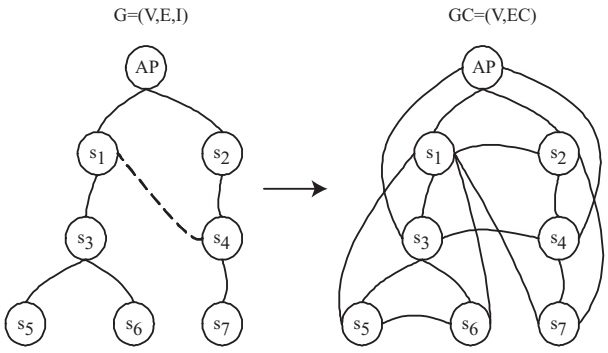

Figure 3: Transformation from $G = (V, E)$ and interference graph $C = (V, I)$, which also includes the dashed edge, to the conflict graph GC.

A *scheduling frame* is the duration from the end of the scheduling coordination packet until the end of the scheduling phase. Thus, the scheduling frame starts when each node has generated exactly one packet and it ends when all these packets have reached AP.

The frame is divided into time slots. A slot is long enough to transmit one data packet plus a guard interval to compensate for synchronization errors. A *schedule* assigns one or more time slots to each edge in E. A node u may receive a packet from its child v during a time slot assigned to $(u, v) \in E$.

We use the following notation: The *distance* $d(u, v)$ between nodes u and v is the number of edges in the path between them in G; and a node u is at *level* k if it is at distance k from AP.

5.2 The scheduling problem

Each node of G (except AP) generates one packet at the beginning of the scheduling frame. Given the interference graph C, the scheduling problem is to find a minimum length frame during which all nodes can send their packets to AP.

Theorem 1 *The scheduling problem is NP-complete.*

Proof We reduce the NP-complete problem of finding the chromatic number of a graph to the scheduling problem.[1] Let $GP = (VP, EP)$ with $VP = \{v_1, \cdots, v_N\}$ being an instance of a graph whose chromatic number we want to find. We first construct a conflict graph $GC = (V, EC)$. First, GC includes all the nodes and edges of GP. Next, for each node v_i, add another node w_i. Then add edges $(w_i, w_j), (v_i, w_j) \in EC$ for all i, j. Lastly add another node AP and edges (AP, w_i) for all i. See figure 4.

The conflict graph GC is such that if w_i is active, none of the nodes in $V \setminus \{w_i\}$ can be active at the same time. Also, if v_i is active, none of the nodes w_j or the conflicting nodes from V, determined by the edges EP, can be active.

We now construct a tree $G = (V, E)$ and an interference graph $C = (V, I)$ whose conflict graph is $GC = (V, EC)$. The edges of the tree are $E = \{(AP, w_i), (w_i, v_i) \mid 1 \leq i \leq N\}$.

Because AP is a parent of w_i, $(w_i, AP) \in EC$ for all i; moreover $(w_i, w_j) \in EC$ for all i, j, because they have the same parent, AP. And $(v_i, w_i) \in EC$ because w_i is the parent of v_i.

Let I consist of edges (v_i, AP) for all i, and $(v_i, w_j), (v_j, w_i)$, whenever $(v_i, v_j) \in EC$.

Since $(v_i, AP) \in I$ and $(w_j, AP) \in E$, $(v_i, w_j) \in EC$ for all i, j. Lastly, if $(v_i, w_j) \in I$ and $(v_j, w_i) \in I$, $i \neq j$, $(v_i, v_j) \in EC$ because the parent of one of them is interfered by a transmission of the other. Thus GC is indeed the conflict graph corresponding to the tree graph G and interference graph C.

Consider the minimum schedule length for GC such that each node $v_i, w_i, 1 \leq i \leq N$, has one packet destined for AP. A packet in w_i takes the path (w_i, AP) and a packet in v_i takes the path (v_i, w_i, AP). Because each w_i conflicts with the nodes $w_j, j \neq i$ and all nodes v_i, it takes N slots to transmit the packets generated at level one to AP, independently of the rest of the network. Also, when the N packets from level two arrive at level one, it takes

[1] The chromatic number of a graph G is the smallest number k such that G is k-colorable. G is k-colorable if its vertices can be colored using k different colors in such a way that adjacent vertices have different colors.

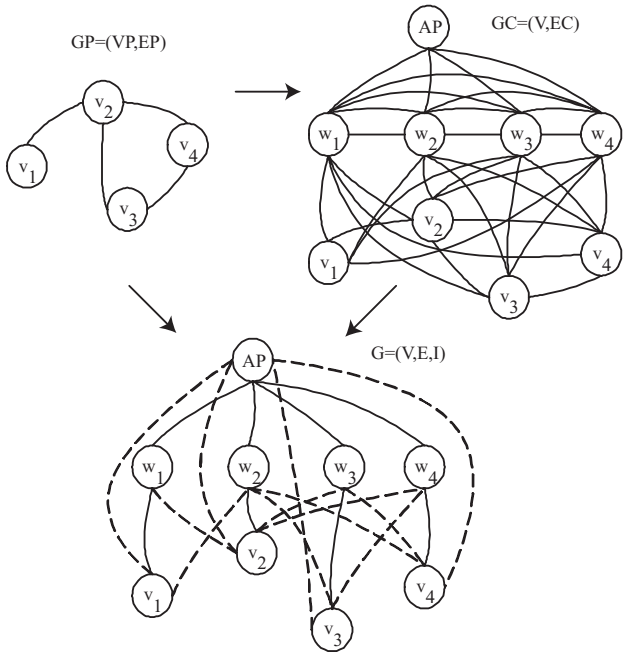

Figure 4: Transformation from $GP = (VP, EP)$ to $GC = (V, EC)$ and then to a tree network $G = (E, V)$ with interference graph $C = (V, I)$.

another N slots to forward them to AP.

Thus to minimize the time to transmit all packets to AP, we must minimize the time to transmit the packets from level two to level one. But the conflict graph at level two is determined by the original graph GP, so the minimum scheduling time is exactly $2N + c$, where c is the chromatic number of the original graph GP. □

Theorem 2 *The minimum schedule length is at least* $|V| - 1$ *and at most* $\frac{1}{2}(|V| - 1)|V|$*, where* $|V|$ *is the number of nodes in* V*.*

Proof AP can receive at most one packet in each slot, so at least $|V| - 1$ slots are needed for all packets to reach AP. This gives the lower bound. The worst interference graph is complete: Transmission from any node interferes with all other nodes. In this case at most one packet may be transmitted (within the network) in one slot. In the worst case of a 'linear' network (in which each parent has at most one child), the total number of hops through which all the packets must travel is $1 + 2 + \cdots + |V| - 1 = \frac{1}{2}(|V| - 1)|V|$. □

5.3 The PEDAMACS scheduling algorithm

The scheduling problem is complex because many subsets of non-conflicting nodes are candidates for each time slot, and the subset selected for transmission in one slot affects the number of transmissions in the next time slot, as some schedulable nodes may not have any packets to transmit because of the subset selected in the previous slot.

Input: (V, E, I, EC).
Output: (VL, EL, IL, ECL).
begin
add node v_1 to VL
$l = 2$
while $l \leq levelOfTree$
add node v_l to VL
add edge (v_{l-1}, v_l) to EL
If $\exists(u, v) \in I(EC)$ with u at level l
and v at level j satisfying $j < l$
add edge (v_j, v_l) to $IL(ECL)$
$l++$
end

Figure 5: Algorithm to find linear network corresponding to original network.

We describe a polynomial-time algorithm that guarantees a bound on the length of the scheduling frame. The advantage of PEDAMACS scheduling over a traditional TDMA schedule is demonstrated in Section 7.4.

The PEDAMACS scheduling algorithm has three parts. In the first part, we obtain a linear network $GL = (VL, EL)$ with interference graph $CL = (VL, IL)$ resulting in conflict graph $GCL = (VL, ECL)$ corresponding to the original network. In the second part, we color this linear network so that the nodes with the same color form a maximal independent set of GCL. In the third part, we schedule the links in the original network, $(u, v) \in E$, based on the coloring of the linear network.

5.3.1 The linear network

If the original tree network has depth N, the linear network $GL = (VL, EL)$ has nodes $VL = \{v_1, \cdots, v_N\}$ with node v_l corresponding to all nodes at level l in the original network and edges $(v_i, v_{i+1}) \in EL$ for $1 \leq i < N$. The interference graph $CL = (VL, IL)$ includes edge (v_j, v_l) if there is an interference edge between a node at level j and a node at level l in the original network for $j, l \geq 1$. The resulting conflict graph $GCL = (VL, ECL)$ then includes edge (v_j, v_l) if the transmissions of a node at level j and a node at level l conflict in the original network for $j, l \geq 1$. Figure 5 gives the algorithm to find the corresponding EL,CL and ECL. The running time of the algorithm is $O(|V|^2)$.

5.3.2 Coloring the linear network

We now color the linear network so that two nodes with the same color can transmit at the same time [13]. The algorithm has two phases. In the first phase, figure 6, one color is assigned to each node. The second phase, figure 7,

Input: $VL = \{v_1, v_2, ..., v_N\}$, graph $GL = (VL, EL)$ with conflict graph $GCL = (VL, ECL)$.
Output: One color assigned to each node $\{(v_1, c_{v_1}), (v_2, c_{v_2}), ..., (v_N, c_{v_N})\}$ in which $c_{v_i} \in \{1, 2, , ..., M\}$ and M is the number of colors.

```
begin
  for l = 1 to N
      s = 1
      while (there is a node conflicting with
v_l with color s)
          s = s + 1
      assign color s to v_l
end
```

Figure 6: Assigning one color to each node in linear network.

guarantees that nodes containing the same color form a maximal nonconflicting set by checking all the nodes for each color to see whether the nodes can also be assigned this color.

Phase I assigns a slot to node i in $O(i)$ steps, so the running time of this algorithm is $O(|V|^2)$. The running time of phase II is $O(|V|^2 M)$, if M colors are used.

5.3.3 Scheduling the original network

If nodes v_i, v_j in the linear network are assigned the same color, they do not interfere, and can transmit at the same time. By construction of the linear network, there is also no interference between nodes at levels i and j in the original network and so any two nodes one chosen from level i and the other from level j can also transmit at the same time.

A *superslot* is a collection of consecutive time slots such that each level of the tree with at least one packet at the beginning of the superslot forwards at least one packet to the lower level during the superslot. Because two nodes at different levels assigned the same color can transmit at the same time, the number of slots in a superslot is at most equal to the total number of colors used for coloring the linear network. After determining the levels corresponding to the current time slot from the coloring of the linear network, a nonconflicting set of nodes at these levels, which have packets to transmit, are selected for transmission. This non-conflicting set may just contain one node or any group of nodes as long as they do not contain any edge belonging to the conflict graph GC between them.

The algorithm is given in Figure 8. If one node is chosen from each level corresponding to the slot of each color, the running time of the algorithm is $O(l)$, where l is the number of slots in scheduling frame given in Section 5.4.

Input: $VL = \{v_1, v_2, ..., v_N\}$, graph $GL = (VL, EL)$ with conflict graph $GCL = (VL, ECL)$, one color assigned to each node $\{(v_1, c_{v_1}), (v_2, c_{v_2}), ..., (v_N, c_{v_N})\}$ in which $c_{v_i} \in \{1, 2, , ..., M\}$.
Output: Color assignment to each node such that each color corresponds to a maximal nonconflicting set.

```
begin
  for s = 1 to M
    for i = 1 to N
      if (no node with color s conflicts with v_i)
        add color s to the color set of v_i
end
```

Figure 7: Assigning more than one color to each node in linear network.

Figure 9 shows PEDAMACS scheduling algorithm for an example network.

5.4 Analysis of special cases

We consider four special graphs.

Case 1: The tree graph $G = (V, E)$ is linear, that is each node $u \in V$ has at most one child. The interference graph $C = (V, I)$ is such that $I = \emptyset$.

Case 2: The tree graph $G = (V, E)$ is general. The interference graph $C = (V, I)$ satisfies the ancestor property, that is, there do not exist u, v, b such that $(u, v) \in I$ and $|d(u, b) - d(v, b)| > 1$. This represents the case where shortest path routing is used with the cost of each path being equal to the number of nodes on that path and only nodes that can hear each other can interfere, which is the assumption of previously proposed TDMA scheduling algorithms.

Case 3: The tree graph $G = (V, E)$ is general and the interference graph $C = (V, I)$ is such that the maximum difference between the levels of two interfering nodes is K.

Case 4: The tree graph $G = (V, E)$ and the interference graph $C = (V, I)$ are both general.

Theorem 4 *In cases 1 and 2 the maximum length of the frame is* $3|V| - 3$ *time slots; in case 3 it is* $(K + 2)(|V| - 1)$*; and in case 4 it is* $\alpha(|V| - 1)$*, in which* α *is the number of colors used in the linear network corresponding to G.*

Proof *Case 1.* If the tree graph G is linear and the interference graph C satisfies $I = \emptyset$, the corresponding linear tree interference graph CL also satisfies $IL = \emptyset$. It is easy to see that this linear tree can be colored optimally with

Input: Graph $G = (V, E)$ with conflict graph $GC = (V, EC)$, color assignment of the corresponding linear network using M colors.
Output: Transmission schedule for nodes of G.
begin
while (at least one packet has not reached AP)
for $s = 1$ to M
set_s = set of levels corresponding to color s
$T = \emptyset$
for $j = 1$ to $|set_s|$
$T = T \cup \{$a nonconflicting set of nodes from level $j \in set_s$ with at least one packet$\}$
assign current slot to set T, pass to next slot
update the place of the packets
end

Figure 8: Scheduling Algorithm.

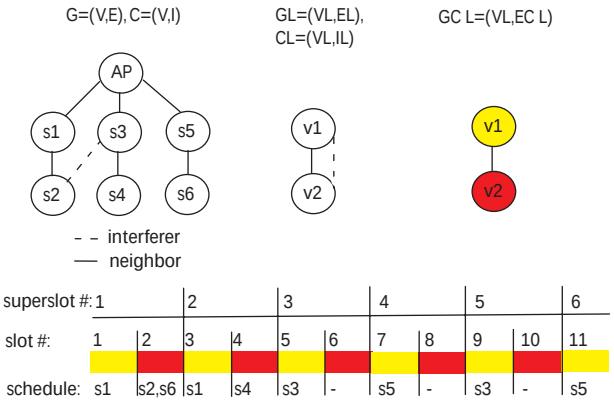

Figure 9: PEDAMACS schedule generation for an example network.

three colors when the number of levels is more than two. The colors are assigned in a round robin fashion starting with the node one hop away from the root, AP.

At the beginning of the frame, each node has exactly one packet. In the first superslot, one packet is transmitted from any level to the next lower level. Because each node is a parent of one node except for the node at the highest level $|V|-1$, it also receives one packet during the superslot. Thus, at the end of the first superslot, each node at level less than $|V|-1$ has exactly one packet to transmit, the node at level $|V|-1$ has no packet, and each node has transmitted exactly one packet during the superslot. This means that at the end of the first superslot, each packet has moved by one hop and one packet has reached to the final destination AP.

In the same way, at the beginning of the second superslot, each node at level less than $|V|-1$ has one packet to transmit, and at the end of the second superslot, each packet has moved by one more hop, there are no more packets at levels greater than or equal to $|V|-2$ and the node AP has received exactly one packet. Continuing in this manner, at the end of $(|V|-1)$ superslots, all packets will have reached the final destination AP.

The maximum number of time slots in each frame is at most the product of the maximum number of slots in each superslot and the maximum number of superslots necessary for all packets to reach the destination AP, namely $3(|V|-1)$.

Case 2. Because the interference graph of the tree network satisfies the ancestor property, the corresponding linear tree interference graph CL satisfies $IL = \emptyset$. It can therefore be colored optimally with 3 colors.

First assume that we select exactly one node to transmit from each level (of the original tree graph $G = (V, E)$) corresponding to the color of the slot. At the beginning of the frame, each node has one packet. In the first superslot, one packet is transmitted from each level to the next lower level. Except at the highest level, each level receives one packet. Therefore, one packet has moved one hop closer to the AP at each level, one packet from level one has reached AP, and nodes at the level of the $depth$ of the tree may have no more packets.

At the end of the second superslot, the number of packets transmitted from one level to one lower level is again one except, possibly, for level $depth$. Each level less than $depth-1$ has one packet to transmit, while nodes at levels $depth$ or $depth-1$ may have exhausted all packets. Continuing in this manner, by the end of i-th superslot, there are no more packets above some *threshold* level, and there is at least one packet at levels lower than this threshold. Since each level below the threshold is guaranteed to have a packet, and all levels with at least one packet can transmit once in each superslot, one packet reaches AP in each superslot. Therefore, the number of superslots required needed for all packets to reach AP is $|V|-1$. Since there are three slots in each superslot, the maximum frame length is again $3(|V|-1)$.

The PEDAMACS scheduling algorithm allows a subset of non-conflicting nodes (instead of a single node) at each level to transmit, the resulting frame length will also be at most $3(|V|-1)$.

Case 3. The worst case is when there is an interfering edge between a node at level j and every node at level i with $|i-j| \leq K$. The corresponding linear graph can be colored by $K+2$ colors in that case.

Assign color 1 to v_1. The color of the nodes $\{v_2, \cdots, v_{K+2}\}$ cannot be 1. Assign the smallest color, 2, to node v_2. The color of $\{v_3, \cdots, v_{K+3}\}$ cannot be 2. Assign the smallest color, 3, to v_3. Continuing in this way, v_{K+2} is assigned color $K+2$. Node v_{K+3} is assigned color 1, since its color is restricted not to be $2, \cdots, K+2$. Thus,

the algorithm colors this network with $K+2$ colors in a round robin fashion with color 1 assigned to v_1. The interference of any other network is a subset of this worst case.

The same reasoning as in Case 2, now indicates that at least one packet reaches AP in each superslot. So the number of superslots needed is at most $|V|-1$. Hence the frame length is at most $(K+2)(|V|-1)$ time slots.

Case 4. The number of superslots required for all packets to reach AP is the number of packets in the network, which is $|V|-1$. The maximum number of slots in each superslot is the number of colors, α. The upper bound on the frame length is then $\alpha(|V|-1)$. □

Remark If interfering nodes are at most K levels apart, the delay of the PEDAMACS schedule is at most $(K+2)(|V|-1)$, linear in the network size. In the worst case of complete interference the delay is $|V|(|V|-1)$.

6 Extension of PEDAMACS

PEDAMACS system requires that there is only one AP in the network and every sensor node periodically generates data at the same rate for transfer to the AP. The system framework, however, is quite flexible and can be generalized in many ways. Sections 6.1, 6.2 and 6.3 give the generalization of PEDAMACS to handle periodic data generation at different rates, non-periodic data generation and existence of more one AP respectively.

6.1 PEDAMACS with Different Packet Generation Rates

Consider the sensor network where all the nodes generate data packets periodically at possibly different generation rates to be transferred to an AP. For this generalization, the protocol phases of PEDAMACS do not require any changes except for the scheduling algorithm described in Section 5.

Let us represent the network by the graph $G_d = (V_d, E_d)$ and conflict graph $GC_d = (V_d, EC_d)$. The scheduling frame starts when each node $i \in V_d$ has generated g_i packets and ends when all packets have reached AP.

Given G_d and GC_d, the scheduling problem is to find a minimum length frame during which all nodes send their packets to the AP. The problem is NP-complete since the scheduling problem proved to be NP-complete in Theorem 1 is a special case where $g_i = 1$ for all $i \in V_d$.

We find a schedule for this problem by providing an algorithm that takes the tree graph $G_d = (V_d, E_d)$ with conflict graph $GC_d = (V_d, EC_d)$ and generation rate $g_i, i \in V_d$ and obtains a graph $G = (V, E)$ with conflict graph $GC = (V, EC)$ and generation rate 1 at each node $i \in V$. Then PEDAMACS schedule for G maps directly to a schedule for G_d.

For each $i \in V_d$, there are g_i corresponding nodes in V. Denote them by $n_i^k, k \in [1, g_i]$. If i is the parent of j in G_d, then $(n_i^1, n_j^l) \in E$ for $l \in [1, g_j]$. The resulting graph $G = (V, E)$ is a tree where n_i^1 is used to forward all the incoming packets to node i.

EC contains two kinds of edges. First, if $(i, j) \in EC_d$, then $(n_i^k, n_j^l) \in EC$ for all $k \in [1, g_i]$ and $l \in [1, g_j]$, because nodes i and j cannot transmit at the same time. Second, $(n_i^k, n_i^l) \in EC$ for all $k, l \in [1, g_i], k \neq l$ for all

$i \in V_d$, because n_i^k and n_i^l, $k \neq l$, represent the same node i and the slot duration is only long enough to carry one packet of node i.

PEDAMACS schedule for G maps directly to a schedule for G_d. If n_i^k for any $k \in [1, g_i]$ is assigned to a slot in the schedule for G then assign node i to that slot in the schedule for G_d. The resulting schedule is valid since none of the conflicting nodes or the nodes corresponding to the same node are assigned to the same slot and the routing of packets is as given in G_d.

The maximum length for the schedule depends on the number of colors used to color the linear network corresponding to $G = (V, E)$ with $GC = (V, EC)$ and the number of nodes in V, which is equal to the number of packets in G_d, $\Sigma_{i \in V_d} g_i$.

6.2 PEDAMACS for Event-driven Sensing

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

The main idea of extending the PEDAMACS system to event-driven sensing applications is that the slots assigned to the nodes do not have to be used. After learning the topology of the network during topology learning and collection phases, the AP broadcasts the schedule of nodes' transmissions in the *scheduling* packet. During the first scheduling phase, all nodes transmit one packet to the AP. If the proportion of the scheduled nodes whose packets are correctly received is below some predetermined threshold, the AP continues by first retransmitting the scheduling packet. If this does not work, it either restarts the topology learning phase or adjustment phase to learn the topology. This continues until the proportion of the successfully scheduled nodes is above some threshold.

Once the AP decides that the topology information is correct, the sensor nodes do not need to transmit data packets unless there is an event. When there is no event, they 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.

Between the event detections, they can also send packets with a period much larger than the duration of the scheduling phase to detect the sensor failures or any change in the radio link.

The advantage of this scheme over random scheduling scheme with listen-sleep schedules is that the nodes put their radio in sleep mode except for the scheduling coordination packet and the beginning of the slots they are assigned to receive packets whereas they have to wake up regularly to see if other nodes are transmitting any packets in random access schemes. This will also eliminate the delays in the transmission of the packets resulting from listen-sleep schedules.

6.3 PEDAMACS Including More than One AP

Consider a sensor network where there are more than one AP and every sensor node periodically generates data at the same rate for transfer to any of the APs. Including more than one AP in the system increases the coverage of the network and brings scalability to the system.

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 transmission power level of each AP at longest range is adjusted such that all the sensor nodes in the network can be reached by at least one of them. The APs then avoid the 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. Each slot is assigned to a set of nonconflicting APs, which are defined to be the APs sufficiently separated such that their coordination packets and the packets of the sensor nodes inside their longest range do not interfere. Assigning slots to the APs is then the same problem as assigning frequencies in cellular radio systems [18].

At the beginning of each phase, each AP runs the schedule generated above to transmit its coordination packet such that the interference from multiple transmissions at the sensor nodes is low enough to decode the packets. The coordination packets include additional fields for the ID of the AP and the time for the end of the transmission of coordination packets.

The protocol then operates as follows. In topology learning phase, each node only retransmits the packet if it is coming on the minimum cost path from an AP that contains it inside the longest range. It determines its neighbors, interferers and next hop on the minimum cost paths to each of these APs. In topology collection phase, each node sends its topology information to each of the APs it is connected to on the corresponding minimum cost path.

The scheduling algorithm at each AP should take into account the interference from the neighboring cells. Recall that the APs assigned the same color to transmit their coordination packet do not have any interfering transmission at the sensor nodes. The APs assigned to the first color then color the linear network corresponding to their network, finds PEDAMACS schedule and broadcast this information to sensor nodes and neighboring APs. Neighboring APs with the next color assign the same color to the levels of their network containing a common node, which is defined to be a node inside the longest range of both APs, and assign a different color to the levels containing an interferer of a specific color. They also preassign the common nodes to the slots they are already assigned by their neighboring AP. Then they color the remaining linear network and assign each slot a non-conflicting set of nodes according to the PEDAMACS scheduling algorithm. The APs corresponding to each color consider the schedules of the APs of the previous colors in this way.

Let the maximum number of colors used in coloring the linear network corresponding to each AP be α. If the linear network coloring of an AP uses less number of colors then all nodes except possibly common nodes inside its network sleep during the remaining slots. Since the network connected to an AP may have to wait for the scheduling of the common nodes if the rest of the nodes at the same level interfere with these nodes, the maximum number of the extra slots included for the neighboring APs is the total number of packets that must be forwarded by these common nodes times the number of colors. Let us call the number of these packets x. Then the maximum duration

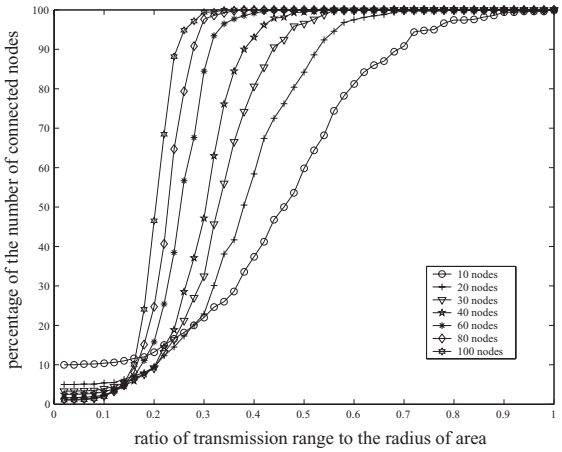

Figure 10: Percentage of the average number of nodes connected to AP as a function of the transmission range of the sensor nodes.

for the total schedule length for an AP is $\alpha(|V| + x - 1)$. Since the value of x only depends on the neighboring APs not all APs, the schedule length is scalable.

7 Simulation

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

The simulation environment is TOSSIM [19], a discrete event simulator for TinyOS [3], the operating system developed for the Berkeley sensor nodes. TinyOS and TOSSIM are not described here. We note that TOSSIM is a bit level simulator, which is better suited to the analysis of MAC protocols than a packet level simulator. Another advantage is that the TOSSIM simulation compiles directly from the TinyOS code used to implement the protocol.

In the simulations the nodes are randomly distributed in a circular area of radius 100 units. The results discussed below are averages of the performance of ten different random configurations unless otherwise stated. Variations around the averages are presented in [27].

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.

Figure 10 shows changes in network connectivity as a function of transmission range and number of nodes. The transmission range is measured as the fraction of the radius of the circle in which the nodes are randomly distributed. As expected, when the number of nodes is large there is a sharp threshold: If the transmission range exceeds the threshold, the network is connected [20, 21]. In the simulation results the transmission range is chosen to be slightly

larger than this threshold.

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

Section 7.1 aims to understand the effect of different random access schemes in reaching the nodes in the network in topology learning phase and the total time it takes to discover the nodes. Section 7.2 describes the simulation of different CSMA access schemes based on implicit and explicit acknowledgement for topology collection phase. Section 7.3 gives the performance of topology adjustment phase. Section 7.4 compares the delay and energy performance of PEDAMACS with different random access and TDMA schemes. Section 7.5 proves the advantage of taking interferers into consideration in PEDAMACS in terms of the number of packets successfully received at AP.

7.1 Topology Learning Phase

A random access scheme is used in this phase, because the nodes do not (as yet) know their topology information. Upon reception of a *tree construction* packet, a node decides to rebroadcast it if it is coming from the shortest path.

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 *listening window size* and decreases it by 1 at each radio clock tick, generated at the transmission rate (e.g. 50kbps), as 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 the collisions. The node starts to transmit when the listening time decreases to 0. Meanwhile, the node can receive another packet and return to the same state it has left.

Figure 11 shows the percent of nodes that the *tree construction* packet reaches in flooding for different backoff and listening window sizes. All the nodes are reached if the backoff window size is large enough. Even with small window sizes, the tree construction packet reaches more than 99% of the nodes. This means the average number of neighbors, 5, is large enough such that if a node does not receive a flooding packet from one of its neighbors, it can still connect to AP via other neighbors.

Figure 12 may be used to explore the trade-off between increase in delay and increase in connectivity as the back-off and listening window sizes increase. Delay increases significantly with increasing backoff window size whereas it is almost constant regardless of the listening window size. The best flooding strategy therefore is to choose a small backoff window size and a large listening window size.

7.2 Topology collection phase

The random access scheme used in topology learning phase is associated with an acknowledgement for the topology collection phase to guarantee the successful arrival of all the packets in the network. [27] shows that although the

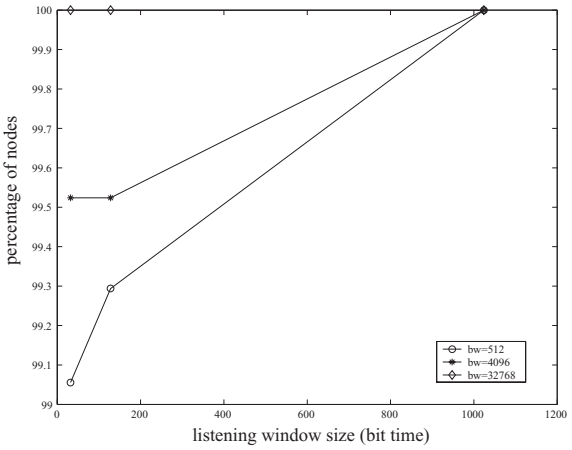

Figure 11: The number of nodes reached by flooding as a percentage of the number of nodes that are theoretically reachable for 30 nodes.

percent of successful arrival increases from 10% to 70% as the backoff window size increases, hidden nodes and bad channel conditions may prevent the successful delivery of all packets.

The acknowledgement schemes fall into one of two categories: implicit and explicit acknowledgement. For both schemes, we choose the backoff window size large enough to 'break' transmission synchronization and to enable the reception of 100% of nodes' packets at the AP.

Implicit acknowledgement, referred to *implicit random* in figures, algorithm works as follows. When a node transmits a packet, it does not delete the packet from the transmit queue. When it receives a packet that is not broadcast or is not destined for itself, it checks whether it is one of the non-acked packets in the queue. If it is non-acked, the packets that are not acked and earlier in the FIFO queue are put to the end of the queue for retransmission. There is also a timeout after which all non-acked packets are placed at the end of the queue for retransmission. The size of this timeout, called *acknowledgement window size*, affects the delay experienced by the network as shown in Figure 13. If the window size is chosen to be very small, the nodes will increase the load in the network by re-sending 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 point and use this value for comparison to other protocols.

Explicit acknowledgement, referred to *IEEE 802.11* in figures, algorithm adopts RTS/CTS/DATA/ACK mechanism used in IEEE 802.11 [2]. RTS/CTS control packets of shorter length 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 a packet destined to another node, it puts its radio in sleep mode and does not transmit during this time. This is called virtual carrier sense. 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 7.1. The backoff and listening window sizes are chosen to be the ratio of RTS control packet length to data packet length times those used in implicit acknowledgement scheme for a fair comparison.

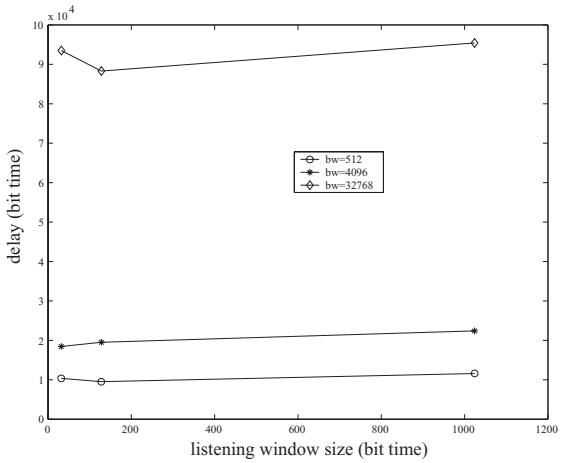

Figure 12: Maximum delay experienced by a tree construction packet for 30 nodes.

Figure 16 shows that the delay experienced by explicit acknowledgement is slightly smaller than that for the implicit acknowledgement scheme since the first knows immediately whether the transmission is successful whereas the latter has to wait for the acknowledgement window size. Also, Figure 18 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 Figure 1 is almost the same for both schemes. Therefore, the delay advantage of explicit acknowledgement scheme without requiring any adaptive scheme for acknowledgement window size adjustment makes it suitable for topology collection phase.

7.3 Adjustment Phase

Random access schemes use flooding and collection to send packets to AP; PEDAMACS uses them to learn the network topology. Therefore, in PEDAMACS it is not enough for the nodes to be able to send packets to AP. The nodes must also hear from all of their neighbors in the topology learning phase to prevent scheduling of conflicting nodes in the same slot during the scheduling phase.

Incorrect topology information can be detected by AP by checking whether it receives packets from scheduled nodes. Since topology discovery is performed in the topology learning phase, the parameters affecting the average number of successfully scheduled nodes are backoff and listening window sizes.

Figure 14 shows that the number of successfully scheduled nodes increases from 90% to 95% as backoff window size increases. This can be increased further by increasing 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.

Figure 15 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

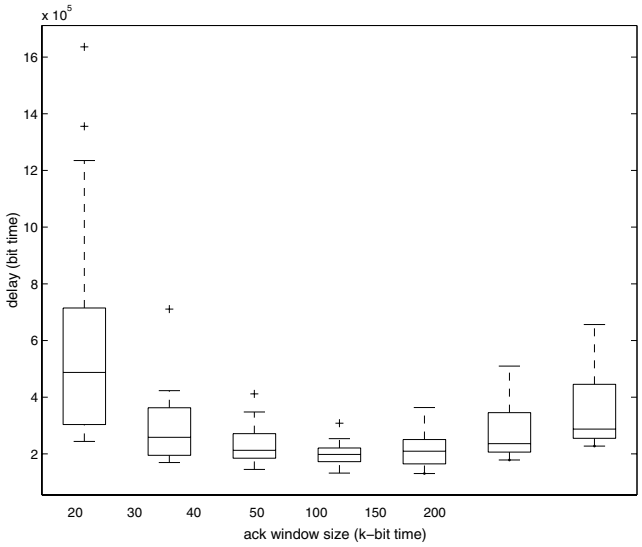

Figure 13: Delay of random access scheme with implicit acknowledgement for 30 nodes and different acknowledgement window sizes.

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 40k-bit time that is less than 1sec at 50kbps.

7.4 Comparison of PEDAMACS with Existing Protocols

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

We have compared PEDAMACS with 5 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 7.2. The only difference here is that they are used to send data packets instead of topology packets in random access networks.

SMAC [6] is a MAC protocol specifically 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% duty cycles, denoting them by *smac* 50% and *smac* 10% in figures.

TDMA scheme, denoted by *tdma*, is included to compare the delay performance of PEDAMACS with TDMA protocols developed for general networks. TDMA scheme collects the topology information with topology learning, collection and adjustment phases of PEDAMACS system. However, upon learning the graph $G = (V, E)$ and the conflict graph $GC = (V, EC)$, it colors the original network GC instead of coloring the linear network GCL and activates each node once during each superslot instead of activating each level of the tree once during each superslot.

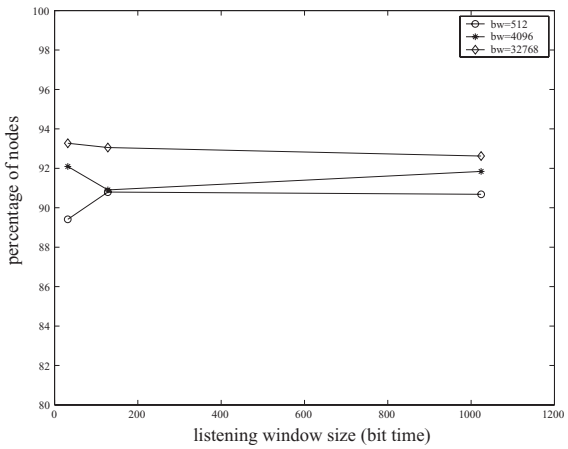

Figure 14: The number of successfully scheduled nodes as a percentage of the number of nodes reaching AP for 30 nodes.

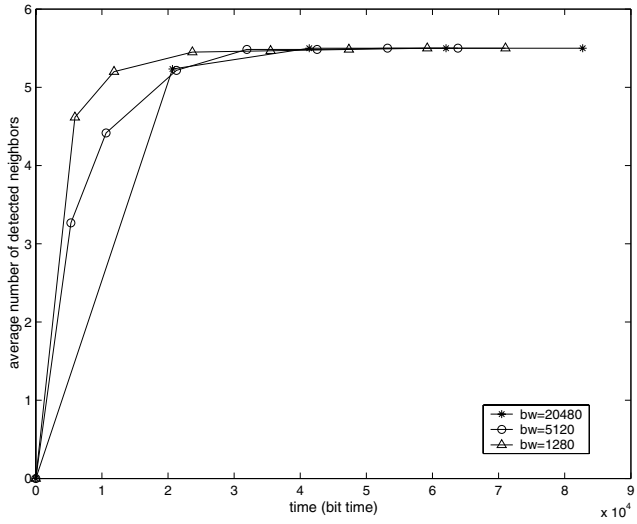

Figure 15: The average number of neighbors that are discovered as time evolves for different backoff window sizes corresponding to different adjustment phase lengths for a 60-node random network.

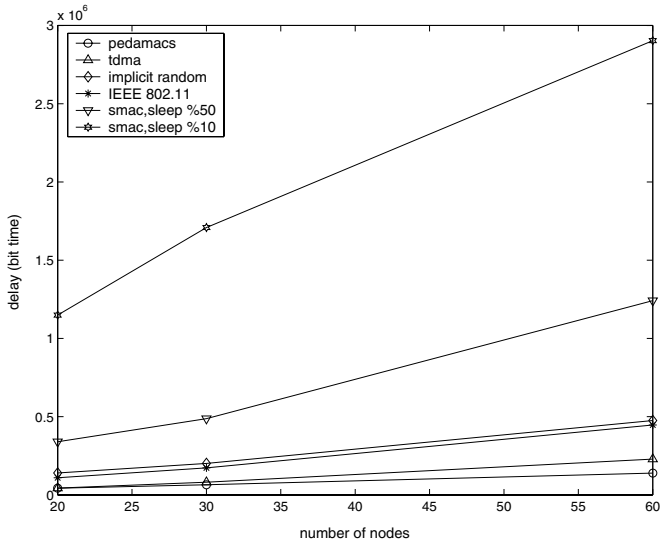

Figure 16: Comparison of the delay of PEDAMACS with the existing schemes for different number of nodes.

7.4.1 Comparison of delay

Figure 16 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% duty cycles respectively over the delay of IEEE 802.11. This factor decreases as the number of the nodes increases.

For a 60-node network the average delay of IEEE 802.11 scheme is nearly 5×10^5 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 scheduling algorithm is slightly smaller than the TDMA scheduling algorithm. The difference between them increases as the number of nodes increases.

To further understand the advantage of PEDAMACS scheduling over current TDMA schedules, we examine the relation of delay to the interference range of the nodes. Figure 17 shows the effect of the $\frac{P_m}{P_s}$ ratio on the delay for a 60-node random network. As the interference range increases, the delay of TDMA schedule increases much more significantly than that of PEDAMACS schedule. This is because the length of PEDAMACS superslot is proportional to the number of colors used for linear network compared to the whole network in TDMA schemes.

7.4.2 Comparison of power consumption

The power-consuming operations in a sensor node 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

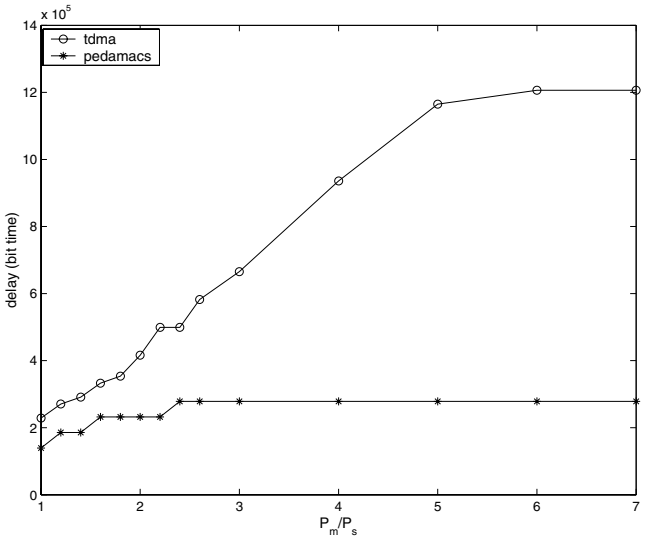

Figure 17: Comparison of the delay of PEDAMACS and TDMA schemes as a function of the ratio of medium transmission range to shortest range for a 60-node random network.

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

Table 1: Power consumption of basic operations in Berkeley mica nodes.

nodes [3].

The network lifetime is estimated from the average energy consumed between two consecutive packet generations of the nodes and the packet generation period at each node, assuming a 50 kbps transmission rate and the use of a pair of AA batteries, which can supply 2200 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. [26] takes into account the fact that nodes closer to AP forward more packets).

Figure 1 gives the lifetime estimates of PEDAMACS and existing protocols for 128Hz 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 (Maximum delay in Figure 16 is $3 * 10^6$ bit time, which is equivalent to $60sec$ at $50kbps$, for 60-node *smac* 10%).

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% duty cycle. The lifetime of PEDAMACS system, on the other

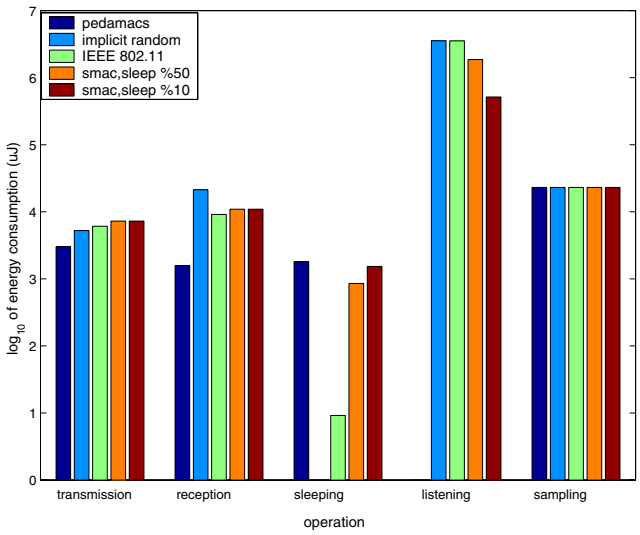

Figure 18: Comparison of power consumption in a PEDAMACS network vs. contention networks for different node operations. Note that the vertical axis is $\log_{10}$.

hand, is about 1200 days. The reason for the dramatic difference becomes clear from figure 18, which compares the power consumed by the 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 $1,000$.

The difference in lifetimes also arises from differences in the amount of energy spent in transmission and reception. The reason of extra transmission energy spent in *implicit random* over PEDAMACS is 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% adds 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 eliminates it by transmitting shorter *RTS/CTS* packets.

The lifetime estimate of PEDAMACS in the simulation is based on the assumption that the time spent for topology discovery is negligible compared to the time spent in scheduling phase. The ratio of rescheduling –induced by sensor movement, link fluctuations – PEDAMACS can tolerate before it loses its advantage in energy savings over SMAC 50% is calculated to be 53% percent of the total time, that is equivalent to taking data and topology information for one period by using *IEEE 802.11* and using PEDAMACS schedule in the next period.

Figure 1 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 energy consumption per bit may decrease or increase depending on the hardware. For RFM TR1000 [22], the radio used in the Berkeley nodes [3], the energy

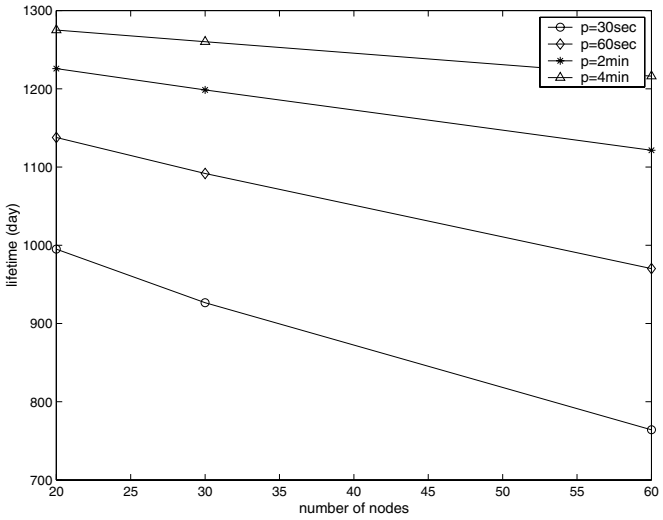

Figure 19: Lifetime of PEDAMACS network for different packet generation periods and different number of nodes.

per bit increases as the rate decreases because the same current is drawn at the same voltage for a longer time. But with most coding schemes, the energy consumed decreases as the rate decreases [23]. Therefore, it is possible to decrease the energy consumption per packet by decreasing the rate. Furthermore, energy consumed in listening to the channel and sampling is constant for different rates.

In some applications the sensor sampling rate would be higher, 10-100 Hz, the samples would be locally processed, and a packet generated every 30 sec. Energy consumed in sampling may then increase significantly, necessitating sensor and sampling schemes that consume less power [24, 25].

As the packet generation period decreases, some of these schemes may not suitable because of their delay characteristics. The lifetime of PEDAMACS is given in Figure 19 for different packet generation rates at 128Hz sampling rate. As the packet generation period decreases, the slope of the decrease in the lifetime as a function of the number of nodes is much sharper because of the decreasing dominating effect of listening energy over the transmit and reception energy.

7.5 Advantage of Considering Interference

Traditional TDMA schemes are based on the assumption that the nodes interfering with a receiver are within its transmission range, which is called *shortest range* in this paper. However, the power needed for interrupting 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 named *medium range* here.

Consider a network protocol that does not consider interferers beyond the shortest range r. In our example network, node i is transmitting to node j whereas node k is transmitting to another node at the same time. Node k is at a distance larger than r so it can be scheduled in the same slot as node i. Figure 20 shows that as the distance between

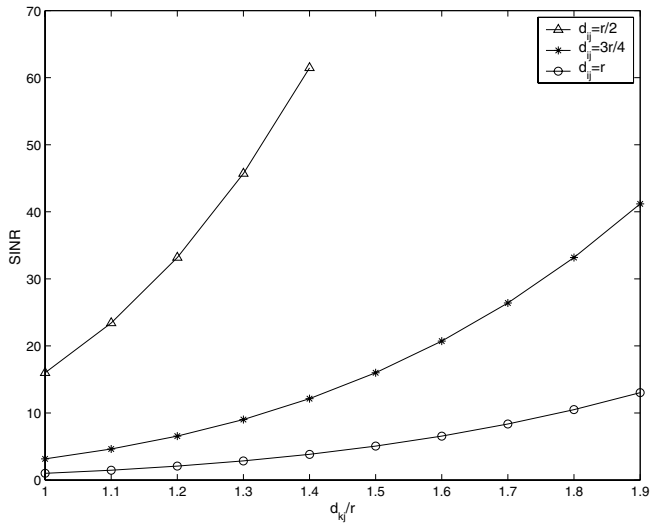

Figure 20: Interference to the transmission from node i to node j by the transmission of node k.

nodes i and j is closer to r, the interference from one node decreases SINR below 10, a common SINR threshold value, for a large range of the distance between nodes j and k. Figure 21 shows the SINR value due to two interferers k and l for the case in which the distance between nodes i and j is $3r/4$. It proves the dramatic effect of increasing number of interferers on SINR value.

Figure 22 examines the number of packets successfully received at the access point as a function of the $\frac{P_m}{P_s}$ ratio for a 60-node random network. The transmissions with a SINR value smaller than SINR threshold are considered not to be successful. As $\frac{P_m}{P_s}$ 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.

8 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, total delay, congestion, and fairness.

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* transmission in a way that eliminates collisions and puts a node in sleep mode during a time slot when it is not scheduled to receive or transmit a packet.

For the application considered here, the PEDAMACS network provides a lifetime of several years compared to

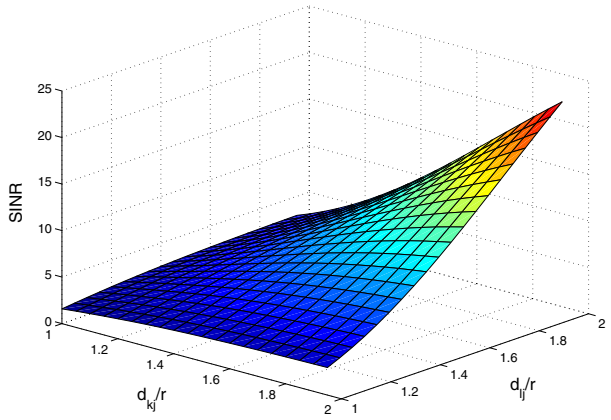

Figure 21: Interference to the transmission from node i to node j at distance $3r/4$ by the transmission of the nodes k and l at different distances.

several months and days based on random access schemes with and without sleep cycles respectively. Moreover the PEDAMACS protocol guarantees a bounded delay, fairness, 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 periodic data generation at different rates, event-driven data generation and existence of more one AP.

In the future, we plan to focus on three areas: interference, stability of the wireless links and redundancy. This paper considered all the nodes inside an interference range, called *medium range*, as interferers. A more involved interference model should be developed to exclude a subset of these nodes from the set of interferers such that their total effect does not decrease SINR below the threshold therefore can be tolerated. This will help the network to decrease the delay increase introduced by increasing number of conflicting nodes.

The packets may not successfully reach the destination because of the time varying characteristics of the wireless channel such as fading. To combat the unstable links, we plan to focus on sending packets over multiple paths or using redundancy in the data.

Redundancy can also be used to increase the lifetime of the system. Let us define the redundant group to be a group consisting of nodes that are located together so that they collect the same measurements. The network schedules only one node from each of these groups while the ones that are not scheduled remain in sleep mode throughout the scheduling phase. The question is how to determine these redundant groups based on the data they have provided in the past.

Acknowledgement This work was performed as part of the California PATH Program of the University of California, in cooperation with the California Department of Transportation. The contents of this paper reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California.

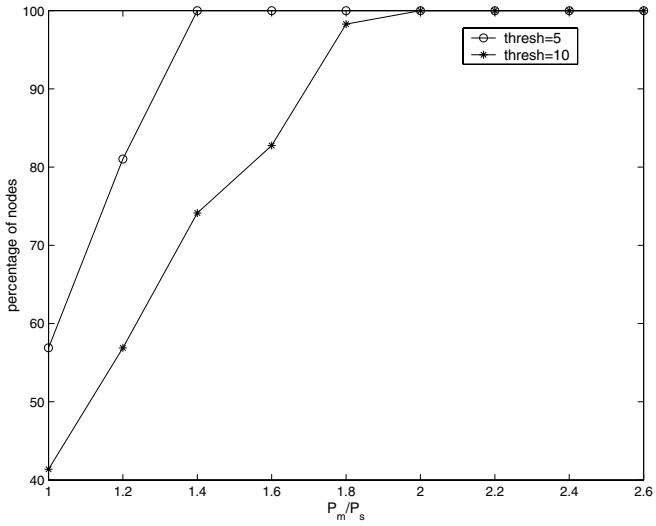

Figure 22: The number of nodes whose packets successfully arrive AP as a percentage of the number of nodes that are scheduled for a 60-node random network.

References

[1] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, *Wireless Sensor Networks for Habitat Monitoring*, ACM WSNA, Atlanta, Georgia, USA, September 2002.

[2] LAN-MAN Standards Committee of the IEEE Computer Society, *Wireless LAN medium access control(MAC) and physical layer(PHY) specification*, IEEE, New York, NY, USA, IEEE Std 802.11-1997 edition, 1997

[3] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, *System Architecture directions for networked sensors*, ASPLOS 2000.

[4] 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), November 2001

[5] C. S. Raghavendra and S. Singh, *PAMAS - Power Aware Multi-Access Protocol with Signalling for Ad Hoc Networks*, Computer Communications Review, July 1998

[6] W. Ye, J. Heidemann, and D. Estrin, *An Energy-Efficient MAC Protocol for Wireless Sensor Networks*,IEEE INFOCOM 2002, June 2002

[7] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, *Optimizing Sensor Networks in the Energy-Latency-Density Design Space*, IEEE Transactions on Mobile Computing, vol.1, no.1, January-March 2002

[8] A. Woo, and D. Culler, *A Transmission Control Scheme for Media Access in Sensor Networks*, Proc. ACM Mobicom 2001, Rome, Italy, July 2001

[9] J. C. Haartsen, *The Bluetooth radio system*, IEEE Personal Communications Magazine, pp.28-36, February 2000.

[10] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, *Energy-efficient communication protocols for wireless microsensor networks*, Hawai International Conference on System Sciences, January 2000.

[11] M. Gerla and J. T. Tsai, *Multicluster, Mobile, Multimedia Radio Network*, Wireless Networks, 1995, pp. 255-65

[12] N. R. Prasad and H. Teunissen, *A state-of-the-art of HIPERLAN/2*, Fall IEEE VTC, 1999, volume:5 September 1999, pp.2661-2666

[13] R. Ramaswami and K. K. Parhi, *Disributed Scheduling of Broadcasts in a Radio Network*, Proceedings of the Eighth Annual Joint Conference of the IEEE Computer and Communications Societies. Technology: Emerging or Converging, IEEE , 1989 pp. 497 -504 vol.2, INFOCOM 1989.

[14] K. W. Hung and T. S. Yum, *Fair and Efficient Transmission Scheduling in Multihop Packet Radio Network*, IEEE GLOBECOM 1992, December 1992, pp. 6-10 vol.1

[15] C. D. Young, *USAP: a unifying dynamic distributed multichannel TDMA slot assignment protocol*, Military Communications Conference, 1996. MILCOM 1996, Conference Proceedings, IEEE, Volume:1, 549-553 Oct.1996.

[16] A. Bhatnagar and T. G. Robertezi, *Layer Net: A New Self-Organizing Network Protocol*, IEEE MILCOM 1990, October 1990, pp. 845-849 vol.2

[17] Swetha Narayanaswamy, Vikas Kawadia, R. S. Sreenivas, and P. R. Kumar, *Power Control in Ad-Hoc Networks: Theory, Architecture, Algorithm and Implementation of the COMPOW protocol*, Proceedings of European Wireless 2002, February 2002, Italy.

[18] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall PTR, 1996

[19] Using TOSSIM Simulator to Develop TinyOS Components, *http://webs.cs.berkeley.edu/to/tinyos-1.x/doc/tutorial/lesson5.html*.

[20] B. Krishnamachari, S. B. Wicker, and B. Bejar, *Phase Transition Phenomena in Wireless Ad-Hoc Networks*, Symposium on Ad-Hoc Wireless Networks, GlobeCom2001, San Antonio, Texas, November 2001.

[21] P. Gupta and P. R. Kumar, *Critical Power for Asymptotic Connectivity in Wireless Networks*, Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming, pp. 547-566, Boston, 1998.

[22] RFM TR1000 *http://today.cs.berkeley.edu/tos/hardware /design/data_sheets/RFM.pdf*

[23] B. Prabhakar, E. Uysal-Biyikoglu, and A. E. Gamal, *Energy Efficient Transmission over a Wireless Link via Lazy Packet Scheduling*, in Proc. INFOCOM 2001

[24] Ultra Low Power Wireless Sensor Project Homepage, *http://www-mtl.mit.edu/jimg/project_top.html*

[25] Ultra Low Power Circuits for Distributed Sensor Networks, *http://buffy.eecs.berkeley.edu/IRO/Summary/00abstracts /warneke.1.html*

[26] S. Coleri, M. Ergen and T. J. John Koo, *Lifetime Analysis of a Sensor Network with Hybrid Automata Modelling*, ACM WSNA, Atlanta, September 2002.

[27] S. Coleri, *PEDAMACS: Power efficient and delay aware medium access protocol for sensor networks*, MS Thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, December 2002.