WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2011; **11**:1628–1645
Published online 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.1229

RESEARCH ARTICLE

A wireless sensor network for precision agriculture and its performance

Herman Sahota*, Ratnesh Kumar and Ahmed Kamal

Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, U.S.A

ABSTRACT

The use of wireless sensor networks is essential for implementation of information and control technologies in precision agriculture. We present our design of network stack for such an application where sensor nodes periodically collect data from fixed locations in a field. Our design of the physical layer consists of multiple power modes in both the receive and transmit operations for the purpose of achieving energy savings. In addition, we design our MAC layer to use these multiple power modes to improve the energy efficiency of wake-up synchronization phase. Our MAC protocol also organizes all the sender nodes to be synchronized with the receiver simultaneously and transmit their data in a time scheduled manner. Next, we design our energy aware routing strategy that balances the energy consumption over the nodes in the entire field and minimizes the number of wake-up synchronization overheads by scheduling the nodes for transmission in accordance with the structure of the routing tree. We develop analytical models and simulation studies to compare the energy consumption of our MAC protocol with that of the popular S-MAC protocol for a typical network topology used in our application under our routing strategy. Our MAC protocol is shown to have better energy efficiency as well as latency in a periodic data collection application. We also show the improvements resulting from the use of our routing strategy, in simulations, compared with the case when the next hop is chosen randomly from the set of neighbors that are closer to the sink node.

KEYWORDS
wireless sensor networks; performance analysis; energy efficiency; wireless application protocol

***Correspondence**
Herman Sahota, Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, U.S.A.
E-mail: hsahota@iastate.edu

1. INTRODUCTION

Precision agriculture refers to the use of information and control technologies in agriculture. Agricultural inputs such as irrigation and fertilizers can be applied in precise quantities as determined by modeling of crop growth patterns to maximize the crop yield and to minimize the impact on the environment. Fertilizer uptake and irrigation needs within a field depend on factors that vary in space and time. A wireless sensor network with sensor nodes spread throughout the field can periodically collect and relay soil data to the information processing center. This data can be used as inputs to the modeling software(s) to determine the optimal quantities of the agricultural inputs (fertilizers, irrigation, pesticides etc.) required in different locations and at different times in the field.

In [1] and [2], we presented the design of our sensor network for automated collection of soil data from a farm-field and our analytical model used to compute the energy efficiency of our MAC protocol with that of S-MAC under the settings of our application and network layers. This paper is an extension of the same work where we provide more details of our design and our performance modeling approach. Our application requires the collection of soil data over the entire duration of crop season(s) at a sampling frequency of about once per hour and a spatial resolution of about $50 \times 50\ m^2$ (Figure 1). Sensor nodes are placed in a rectangular grid at the required spatial resolution and are integrated into a network that collects data periodically. The data is relayed to a sink node located at one of the corners of the rectangular field at regular time intervals, consistent with the measurement period. We design our network stack (physical (PHY), MAC, and network layers) to meet the requirements of the application. Models used in precision agriculture such as those to predict crop growth patterns, hydrologic flow, and carbon cycle require data collection from the entire field at an interval of an hour or more. Thus, although latency is not

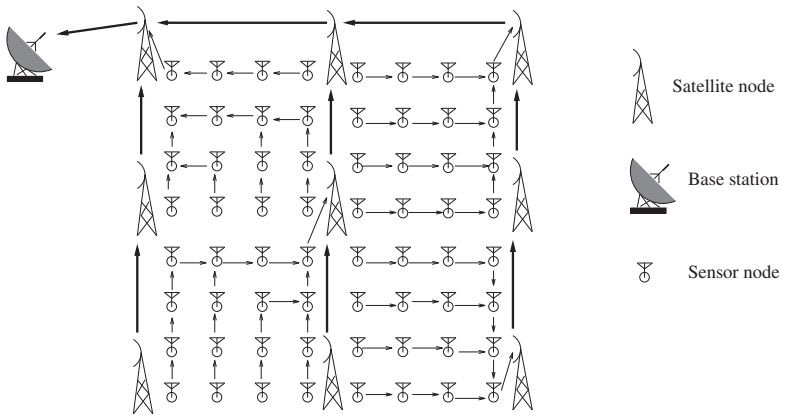


Figure 1. Wireless sensor network showing sensor and satellite station deployments and our hierarchical routing strategy.

a concern; energy efficiency and reliability (data loss) are of interest.

Sensor nodes are deployed in a farm-field that is divided into rectangular areas (Figure 1). Within each field, nodes are deployed in a rectilinear grid form. Nodes use a transmission power that enables them to communicate with their adjacent nodes only. They are buried in the soil at a depth of about 30 cm. Such deployment of sensor nodes was chosen in order to fit the application's requirement that samples should be taken from spatial points in the field where the spatial resolution enables the model to predict the moisture and nitrogen distribution. Also, farming equipment usually traverses the field in regular patterns that can be used to plant the sensor nodes in the field in the rectilinear grid.

At the corners of each field, special nodes called *satellite nodes* are deployed. Communication between sensor nodes in a field and the satellite nodes at the corners of the field is asymmetric in the sense that a transmission from a satellite node can reach all sensor nodes in the field, but a transmission from a sensor node reaches adjacent sensor nodes only and can reach a satellite node if it is close enough. The four corner satellite nodes assume the role of sink in a round-robin manner and collect information from the sensor nodes in the adjoining field to relay the collected information to the base station where the information is processed. We use hierarchical routing in our system. At level one, the sensor nodes in a field relay the data to the designated sink node (one of the satellite nodes), and at level two, the satellite nodes relay the data collected to the base station. We focus our current work on designing the network used to relay the data collected from one such rectangular field to its designated sink node. Satellite nodes also dispatch routing and scheduling information to the nodes in the field and participate in the sensor localization process.

Wireless communication is the most energy intensive process at sensor nodes. Hence, our goal is to design PHY layer, MAC, and network protocols to minimize the energy consumption at the transceiver. The communication pattern in our application is best described as periodic and bursty. Nodes are active for a relatively short period when they relay the collected data to the sink node. This is followed by a long period of inactivity, the duration of which depends on the frequency of data collection (approximately once per hour). To conserve energy, the transceivers are shut down during this period of no active communication. At the beginning of a data collection round, nodes are synchronized by one of the satellite nodes and provided with route and schedule information. Nodes are scheduled to relay their data over multiple hops to the designated sink node. After synchronization by the satellite node, all sensor nodes turn off their transceivers and set a timer to wake up at their respective scheduled time of communication. However, because of frequency drifts in the crystal oscillators used in hardware timers, a node may wake up at a different instant than its neighboring node requiring wake-up synchronization. Note that any energy consumed during this phase is an overhead. In current designs of energy efficient MAC protocols for wireless sensor networks ([3], [4], [5]), wake-up synchronization between a sender and a receiver is established using regular transmission-power and receiver-sensitivity levels of PHY layer. Our strategy is aimed at reducing the energy consumption during this phase. We consider a modified wake-up synchronization scheme in which a node lowers its energy consumption by reducing its own receiver sensitivity level. In contrast, we increase the transmission-power level of the wake-up synchronization signal to allow for a successful detection by a receiver operating at a lower sensitivity level. It turns out that this new scheme saves the overall energy consumption during the synchronization phase because the protocol uses a relatively short pulse for wake-up signal [1]. Additionally, we take advantage of the convergecast nature of sensor network traffic to minimize the number of high energy pings transmitted. This is achieved by making the downstream node wake up all its upstream neighbors with a common ping signal. Existing sensor network MAC protocols such as S-MAC [3], WiseMAC [4], and T-MAC [5] establish individual link communications independently, regardless of whether two or more links share a common

downstream node. To bound the clock drift from growing indefinitely, nodes are synchronized globally by the satellite nodes periodically. In our design, we make the period of global synchronization equal to the data collection interval. After synchronization by the satellite node, each node enters a sleep mode to be woken up by a timer event at the scheduled time of communication.

Besides our MAC layer that aims to minimize energy consumption, we design our network layer to balance the rate of energy depletion among all the nodes in the network. We use an energy aware routing strategy to compute the multihop routes from each node in the field to the designated sink node located at one of the four corners of the rectangular field. At each node, the next hop is chosen as the node that has the maximum remaining energy from the set of neighboring nodes that are closer to the sink node. Any of the satellite nodes located at the four corners of the rectangular field are capable of functioning as the sink node. To reduce the load on the relay nodes located near the corners, the role of the sink is rotated among the four corner nodes. In addition, to minimize the number of link establishments and hence the overhead of wake-up synchronizations between any given pair of nodes, a transmission schedule is computed to ensure that a node relays its collected data to the next hop only after it has received (or attempted to receive) the data from all the nodes in the routing subtree rooted at itself.

Majority of the MAC protocols proposed for wireless sensor networks achieve energy efficiency by switching off the transceivers when no communication is taking place. S-MAC is such a protocol. We present our analytical method to model the performance (throughput, latency, and energy consumption) of our MAC protocol and that of S-MAC [3] under the settings of our application and network layers. We validate our performance models against the simulation results, written in nesC and Python using the event driven simulation framework provided by TOSSIM. We also use simulations to quantify the improvement in load balance attained by our network layer.

The following are the contributions of this paper:

- We consider multiple power modes at physical layer that reduce the energy consumed during wake-up synchronizations and present the design of our energy efficient MAC layer.
- We develop our network layer to balance out the load of communicating sensor data to a sink node and schedule the communications according to the computed routes to keep the number wake-up synchronizations to a minimum.
- We present an analytical approach to model the performance (throughput, latency, and energy consumption) of the MAC protocol (ours and the standard S-MAC) under a given routing topology. The results of our model show a 25% improvement in latency and a 65% improvement in energy consumption for the same level of throughput, confirmed by simulations.
- We develop TOSSIM-based simulations of a wireless sensor network, modeling the designed PHY, MAC, and routing layers to validate our analytical MAC performance models and to study the load balancing gains achieved through our network layer. Our simulation studies show a more balanced energy consumption profile over the sensor field reducing the probability of network partitioning.
- We present an initial implementation of our MAC protocol on CC1110 system-on-chip with low-power radio frequency (RF) transceiver and 8051 Microcontroller (MCU) from Texas Instruments [6].

The organization of the paper is as follows: Section 2 reviews our wake-up synchronization strategy, MAC layer features, and routing strategy from [1]; Section 3 presents our performance evaluation approach using simulation studies and analytical models to analyze throughput, latency, and energy consumption for S-MAC and our MAC protocol under the realm of our routing strategy. We also present simulation studies to analyze the load balance achieved using our network layer. Section 4 highlights related work, and we conclude with Section 5.

2. ENERGY EFFICIENT WAKE UP AND DATA COLLECTION

Our sensor network collects data periodically. We refer to the period of time during which sensor nodes actively forward their collected data to the sink node as a data collection round. All nodes receive routing and scheduling data at the beginning of each round from the sink node of that round and use this event to synchronize their clocks. However, clocks drift during the course of the round owing to the fact that sensor nodes are implemented using inexpensive crystals that may experience frequency drifts of the order of 100 parts per million [7]. When two nodes wake up to synchronize, their clocks could drift by an amount $\pm\Delta$ relative to a true clock, where Δ represents the absolute clock drift. Therefore, the overhead energy spent can be significant if Δ is large, which is the case for long inter-synchronization intervals. The unnecessary energy consumption can manifest itself at the sender in the form of long preambles transmitted with every frame [4], [8] or at the receiver in the form of longer idle listening periods.

To establish the wake-up synchronization in an energy efficient manner, we take the approach based on a trade-off between the energy consumed for wake-up synchronization at the transmitter versus at the receiver: The node transmitting the wake-up signal transmits at higher power level but only for a fairly short duration, whereas the receiver uses less power in the detection of the wake-up signal by degrading its sensitivity. This can be achieved by bypassing some amplifier stages, thereby saving the energy they consume for their operation. The wake-up signal is a signal of very short duration and carries no data and its sole purpose is to generate an interrupt in the receiver circuit to indicate that it needs to wake up. A similar philosophy for

wake up has been proposed by the use of radio-frequency identification (RFID) technology in [9].

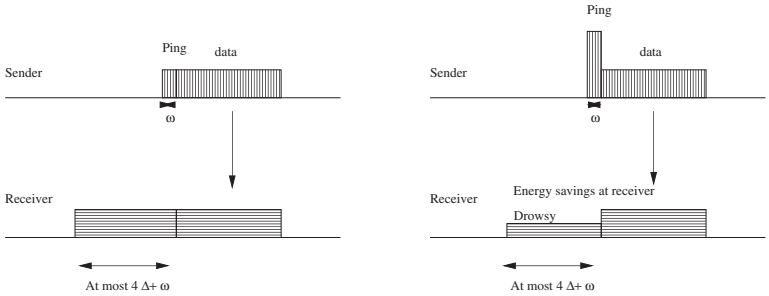
We refer to the mode of short duration and high-power wake-up signal transmission as the *ping* mode. Likewise, the mode of operation of the receiver circuit in the degraded sensitivity is termed as *drowsy*. These two modes of the radio are used in the wake-up strategy employed by our MAC protocol, discussed in the next subsection.

Figure 2 illustrates the energy saving achieved using the drowsy and ping modes as opposed to the regular transmit and receive levels, when two nodes wake up to establish a wireless link after a sleep period. Let ω denote the duration of a ping, P_t and P_r denote the power of regular transmit and receive modes, respectively, and P_p and P_d denote the power of ping and drowsy modes, respectively. The energy saved during a wake-up synchronization can be up to $(4\Delta+\omega)(P_r-P_d)-\omega(P_p-P_t)$ (the first term denotes the energy saved at the receiver, and the second term subtracts the extra energy needed for ping). The appearance of the term '4Δ' can be understood as follows: The sender may wake up Δ time units earlier than scheduled, and in order not to miss the ping, the receiver must target waking up 2Δ units earlier than scheduled (so that it will wake up latest by the sender's scheduled wake-up time). But then, the receiver may wake up as early as 3Δ units prior to the sender's scheduled wake-up time, whereas the sender may wake up as late as Δ unit after its own scheduled wake-up time; and as a result, the receiver may remain drowsy for at most 4Δ units before it witnesses the ping signal. In order that Δ does not grow unbounded, the clocks need to be synchronized periodically, and in our application, this is carried out at the beginning of a round.

At network layer, we further aim at energy conservation by seeking load balancing in the computation of routes from the sensor nodes in a field to the designated sink node and keeping the number of wake-up synchronizations to a minimum by assigning a transmission schedule based on the routing tree. The routes are computed by taking into consideration the remaining battery level of the nodes: A downstream node is chosen to be a neighbor closer to the sink node that has the maximum remaining energy level. Also, the role of the sink is rotated among the four corner nodes after each round of data collection to distribute the energy consumption more evenly among the relay nodes closer to the corner nodes. The schedule for the transmission/reception of messages is computed in accordance with the routing tree to ensure that the number of sender–receiver wake-up synchronizations and internodal communications is kept to a minimum. We discuss our MAC and network layers in the following subsections.

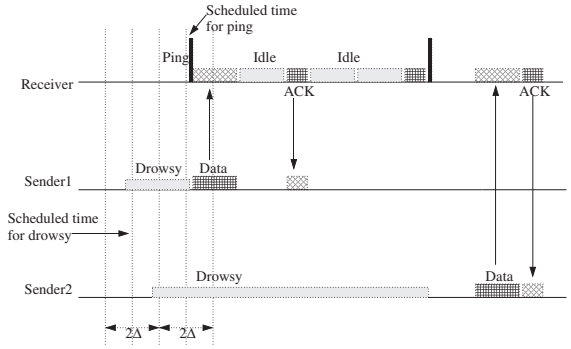
2.1. MAC layer

There are two communication patterns in our sensor network: broadcast (for distributing the route and schedule information, and a clock reset signal to the nodes) and convergecast (for data collection in each round). The route, schedule, and global clock reset are distributed from the sink to sensor nodes over a single hop. This is possible because the transmission range of the satellite nodes is considerably larger than that of the data collection sensor nodes. Accordingly, the main functionality of our MAC protocol is aimed at data collection (convergecast communication), where data from sensor nodes is sent to a sink node adhering to routing and scheduling information made available to all nodes. The data flow in our network can be characterized as periodic and bursty. Because of the periodic nature of the traffic, a time scheduled access to the channel by each node is desirable to avoid unnecessary packet loss and delay inherent in a carrier sense multiple access (CSMA)-based protocol such as 802.11 or 802.15.4 (for contention-based access mode). Also, the sparse node density, because of carefully chosen transmit power level and receiver sensitivity to conserve energy, does not require a contention-based access used in traditional MAC protocols to avoid collisions. However, because of the drift in the nodes' clocks when the transceivers are switched on for communication, a synchronization process is needed. We use the high energy ping transmitted at one node coupled with lower receiver sensitivity drowsy mode at another node as described in Section 2 to conserve energy during the synchronization phase. The receiver node is assigned the task of transmitting the synchronizing ping to wake up all its upstream neighbors. Because of the convergecast nature of the communication, this strategy

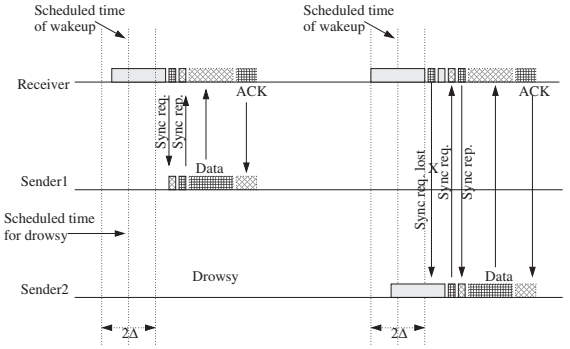


(a) Traditional radio (b) Radio with multiple power modes

Figure 2. Illustration of energy saving at receiver by our scheme.



(a) An execution scenario of multiple senders to one receiver communication links with PD-MAC.



(b) An execution scenario of multiple senders to one receiver communication links with S-MAC.

Figure 3. Example execution scenarios: (a) PD-MAC and (b) S-MAC.

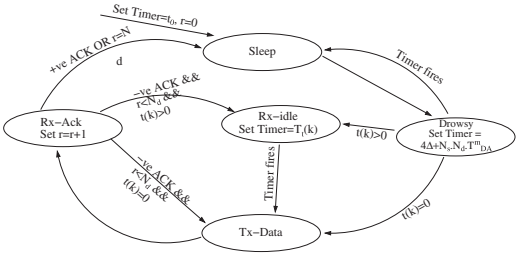
minimizes the number of high energy ping transmissions. The sender nodes must transition to drowsy modes before the receiver node transmits ping. We call our MAC protocol PD-MAC (letters 'P' and 'D' stand for ping and drowsy, respectively).

We denote the receiver node by m and the set of all upstream neighbors (senders) of node m by U_m. All nodes $k \in U_m$ are scheduled to transition to the drowsy mode from sleep at the same instant (denoted by t_0). To account for the $\pm\Delta$ clock drift, node m is scheduled to transmit the high energy ping at $t_0 + 2\Delta$ which ensures that all sender nodes have transitioned into drowsy mode when the ping is transmitted. A ping transmission is followed by $|U_m|$ number of time slots, one for each sender and an additional slot for the collective ACK message transmitted by node m to indicate the successfully received data. We denote the duration of $|U_m|$ data slots and the accompanying ACK slot by T^m_{DA}. Each sender node k is assigned an index $t(k) \in \{0, 1, 2, \ldots, |U_m| - 1\}$ which indicates the time slot in which the node must transmit its data. The ACK message is followed by another set of $|U_m|$ data transmission time slots and an ACK transmission time slot and so on to allow up to N_d data transmission attempts for each synchronized sender node before it receives a positive acknowledgement message. A sender node that receives an ACK or has attempted N_d data attempts transitions to sleep mode and does not transmit any more in the current data collection round. If, after N_d data attempts, the receiver node m does not receive data from all its sender nodes, it is assumed that the *failed* sender nodes did not receive the ping signal. In that case another ping is transmitted, followed by up to another N_d data attempts by each sender node synchronized by the most recent ping. Up to N_s ping transmissions are allowed per receiver node. So, if at least one of the synchronized sender nodes misses all N_d attempts, the receiver node transmits all the N_s pings and waits for data in regular receive mode for $N_d.T^m_{DA}$ time units corresponding to each ping transmitted. Figure 3(a) shows the time line of events for our MAC protocol for one receiver versus two upstream senders for $N_d = 2$ and $N_s = 2$. Sender 2 misses the first ping but is synchronized by the second ping and transmits its data successfully in one attempt. Sender 1 is synchronized by the first ping attempt and successfully transmits its data in the first attempt.

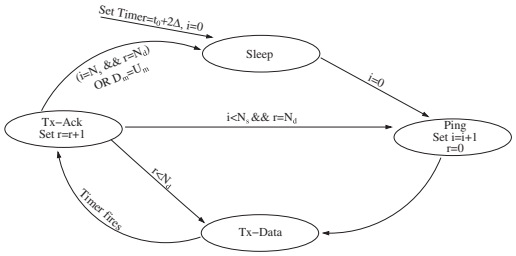
We can formally understand the operation of PD-MAC using the finite state machines shown in Figure 4(a) and (b) for the sender and receiver nodes, respectively. Receiver node m and each sender node $k \in U_m$ receive the route and schedule information from a satellite node, synchronize their local clock and transition to sleep mode. Node $k \in U_m$ sets the timer to schedule wake-up to drowsy mode at time t_0, the scheduled time of communication with the downstream node m. On the other hand, node m sets the timer to schedule ping transmission at time $t_0 + 2\Delta$ to ensure that all nodes $k \in U_m$ have already transitioned to drowsy mode taking into account $\pm\Delta$ time units clock drift. Each sender node k that successfully detects the wake-up ping signal[†] transmits data in its allotted time slot. Receiver node m transmits a collective ACK message, containing a bit vector of length $|U_m|$ to indicate the successfully received data transmissions, after the last sender's ($|U_m|^{\text{th}}$) data slot as shown in Figure 5.

To allow for N_s synchronization attempts interspersed with N_d data attempts, the sender node sets a timer for duration $4\Delta + N_s.N_d.T^m_{\text{DA}} + N_s.T_s$. The 4Δ term was explained in Section 2. $N_d.T^m_{\text{DA}}$ accounts for N_d data attempts following every ping transmission, where T^m_{DA} denotes the time duration allotted for one data attempt by every sender node $k \in U_m$ and the transmission of the collective acknowledgement by node m. T_s denotes the duration of the ping signal. If the sender node does not detect the ping and the timer fires, a channel access failure is assumed, and the node goes back to sleep mode. If the sender node detects the ping signal, two possible

[†] we denote the probability that the ping signal is not detected by q



(a) Send protocol for a node $k \in U_m$. $t(k)$ denotes the transmission order of node k among all nodes $j \in U_m$. $T_t(k) = \sum_{j \in U_m | t(j) < t(k)} T_D(|R_j|)$ denotes the starting time of node k's data transmission after receiving the ping or collective acknowledgement packet.



(b) Receive protocol for node m. $T_D = \sum_{k \in U_m} T_D(|R_k|)$.

Figure 4. Finite state machines for PD-MAC. Counters *i* and *r* keep track of the number of ping transmissions and data attempts made, respectively.

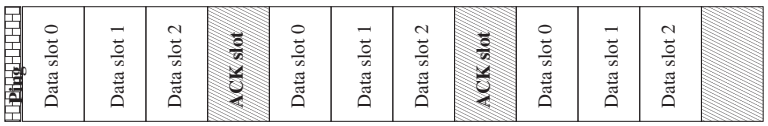


Figure 5. Data and acknowledgement slots once synchronization is established.

actions take place depending on the value of the transmission order $t(k)$. If $t(k) = 0$, that is, node k is assigned the first slot to transmit its data, it transitions the transceiver to transmit mode immediately to transmit the data. Otherwise, it waits in sleep mode for all other sender nodes that have a lower transmission order than itself to transmit their data. The duration of sleep is computed as $T_t(k) = \sum_{j \in U_m | t(j) < t(k)} T_D(|R_j|)$, where $T_D(|R_j|)$ denotes the duration of time slot needed for node j to transmit its data, which consists of data from all the nodes in the routing subtree R_j rooted at node j. Each sender node transmits up to N_d times until it receives a positive acknowledgement at which point it transitions to sleep mode and waits for the next data round's route and schedule information from the satellite node.

On the other hand, the receiver node m, upon receiving the route and schedule information from the satellite node, transitions to sleep mode and sets the timer for $t_0 + 2\Delta$. When the timer fires, it transmits a ping to synchronize all nodes $k \in U_m$ and immediately switches the transceiver to regular receive mode. It stays in receive mode for a duration $T_D = \sum_{k \in U_m} T_D(|R_k|)$ to allow one data attempt for every sender node, keeping track of all successful transmissions. When the timer fires, it transitions to regular transmit mode to transmit the acknowledgement bit vector of length $|U_m|$. The process is repeated for up to N_d times for every ping transmitted. If it detects successful transmission by all nodes $k \in U_m$ before or at the end of N_d attempts, it transitions to sleep mode and waits for the next route and schedule update from the satellite node. Otherwise, if it does not receive data from all nodes $k \in U_m$ at the end of N_d attempts, it transmits another synchronizing ping signal and repeats the whole process for up to N_s times.

2.2. Routing layer

Given a sink node for a round, a routing tree is constructed for achieving load balancing. Each node will forwards its data to that neighbor which has witnessed the least depletion of energy so far. The schedule to transmit data packets is computed complying with the order imposed by the routing.

Multihop routes from all the sensor nodes to a sink are computed for each round. The objective is to minimize the energy consumption by way of load balancing and avoiding more than one successful transmission by each node in a round because each transmission incurs the additional energy overhead for wake-up synchronization. That is, a node must forward to a downstream node only when it has received the data from all of its upstream nodes to avoid multiple transmissions. In our application, the sensor field is laid out in the form of a rectangle with the nodes placed at the various grid points. In each round, the nodes forward their data to a sink located at one of the four corners of the field. Within a round, a node forwards its data to a neighboring node that is closer to a designated sink node. In case of multiple choices for such a neighbor, the data is forwarded to the one with maximum remaining energy level. Such routing strategy helps balance out the energy consumption across the entire sensor field. The role of the sink is rotated among the four corners of the field to prevent asymmetrical depletion of energy among the corner relay nodes. A sequence of 4 rounds, in which each corner node has once acted as a sink, is called a *macro-round*.

Routes determine the order in which nodes must forward their data. The scheduling of the nodes must be performed to determine the exact timings of events such as when a

node shall initiate wake up by generating ping, when shall a node go into drowsy to capture a ping and transmit its data together with the data it has collected from its upstream nodes. All such scheduling is also performed within the network layer. A goal of this scheduling is to ensure that the data collection takes place with as few pings as possible. Note that this scheduling is different from packet level scheduling which is performed as a part of MAC layer. The number of ping transmissions is also minimized by grouping together all the senders to a receiver and awakening them by a single ping from the receiver. This is an instance of the cross-layered approach in our network architecture where MAC layer uses information about the network topology. The routing and the scheduling information is computed by the one of the corner satellite stations and is then distributed to the nodes in the field.

3. MATHEMATICAL ANALYSIS AND SIMULATION STUDIES

In this section, we present our performance modeling approach to evaluate the performance of PD-MAC versus that of S-MAC and validate our models using event driven simulations. Also, we use simulations to quantify the performance improvements gained as a result of our energy aware routing strategy and sink assignment strategy for load balancing.

We develop analytical models to compute the throughput, delay, and energy consumption of PD-MAC and S-MAC under the settings of our application and routing layers. For comparison purposes, both protocols operate under the same routing and scheduling information made available by the network layer. We consider a 25-node network placed in a 5×5 rectangular grid. For the purpose of modeling the MAC protocols, we use a fixed routing tree as depicted in Figure 6. A corner node, node 0 acts as the sink at which all the data from the network is collected. The scheduling strategy ensures that no two links in the network are active at the same time to avoid interference

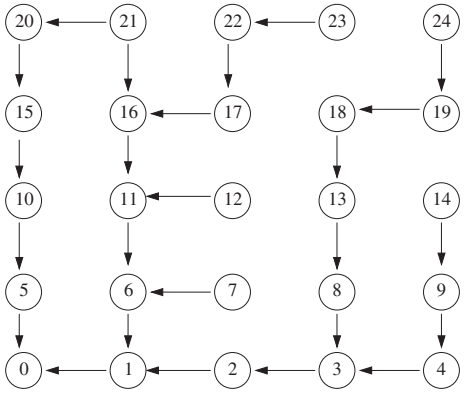


Figure 6. Routing tree R used for performance evaluation.

and also ensures that a node begins transmitting data to its downstream node only after it has already attempted to collect data from all of its upstream nodes. For example, in Figure 6, link $17 \rightarrow 13$ is scheduled after the links $23 \rightarrow 21$ and $21 \rightarrow 17$, respectively.

The differences between S-MAC and PD-MAC lie in the implementation of the routes and schedules and also the wake-up synchronization protocol. A wake-up synchronization protocol is used to synchronize two nodes that awaken to communicate with each other after a sleep period. In S-MAC, the synchronization handshake as well as the data/ack exchange occurs in a pairwise manner between one sender node and one receiver node. If a node has more than one upstream neighbors, then the respective sender→receiver links are scheduled in a sequence. For each such pair of nodes, a maximum of N_s synchronization attempts are allowed, and up to N_d data attempts are allowed once synchronization is established. When a node wakes up to establish a communication link with its neighbor, it waits for a predefined duration before transmitting a synchronization request packet so as to guarantee that the other node on the link is awake to receive the synchronization request. The duration of this period is accordingly chosen to be the worst case relative drift (2Δ) between the clocks of the two nodes. Accordingly, letting $2T_S$ denote the time to send synchronization request and receive reply packet, the *discovery duration* (T_{DD}), defined as the period of time a node must wait before transmitting a synchronization request when it wakes up to discover any existing neighboring sleep–listen schedules, lasts $T_{DD} = 2\Delta + 2T_S$ units of time. The node that receives the synchronization request packet replies with a synchronization reply packet. Keeping the primary goal of comparing latency and energy savings, we make a simplifying assumption that the reply packet is delivered error free. In addition, in regular S-MAC, the synchronization reply (a.k.a rebroadcast of the synchronization packet) by the node that receives a synchronization request happens after a random back-off to prevent multiple nodes from transmitting at the same time. However, in our application setting, only a sender–receiver pair can be active at a time. Hence, a back-off is not required. The two nodes keep awake (i.e., maintain a 100% duty cycle) until they expend all synchronization and data transfer attempts or until data transmission succeeds. This way, S-MAC is not penalized with a longer round duration by being forced to sleep part of the time due the choice of a less that 100% duty cycle. A sender node is guaranteed to have data to send to its downstream neighbor when it wakes up, eliminating the need for a sleep–listen schedule. Figure 3 shows a possible execution scenario of a schedule consisting of one receiver and two sender nodes by S-MAC, under the choice of parameters $N_s = 2$ and $N_d = 2$. The two links are scheduled sequentially. Sender 1 synchronizes in the first synchronization attempt and successfully transmits data in one attempt. Sender 2 misses the first synchronization attempt but synchronizes in the second attempt and then transmits its data successfully in one attempt.

On the other hand, in PD-MAC, a node synchronizes all of its upstream nodes with a common ping. A maximum number (N_s) of ping attempts are allowed in case of unsuccessful synchronization. Between successive pings, the upstream neighbors are allowed a maximum of N_d data transmission attempts in a time division manner. This process of synchronization and data collection constitutes the communication time of a node in PD-MAC. Additionally, as explained in Section 2, PD-MAC uses multiple radio power modes to conserve energy during the wake-up synchronization phase.

In case of PD-MAC, a possible execution of which under the same scenario as S-MAC is also shown in Figure 3, a common ping signal is transmitted by the receiver to synchronize the two senders. Sender 2 misses the first ping. After allowing $N_d = 2$ transmission attempt to both senders, when the receiver does not get all data, it transmits another ping signal ($N_s = 2$). Sender 2 captures this ping and successfully transmits its data this time, at which point the communication ends.

In the following subsections, we present our simulation framework, analytical models, and results to compare the performance (throughput, latency, and energy consumption) of PD-MAC and S-MAC. Note that throughput is determined by the average number of data packets collected per unit time and can be obtained by taking the ratio of the average data count at the sink node per round and the average latency of the round. So, it suffices to determine data count and latency per round. We also present performance evaluation of our routing strategy using simulations. We start with a description of the simulation framework used in our performance evaluation studies in Section 3.1. Sections 3.2–3.4 detail our analytical models used for performance analysis of PD-MAC and S-MAC under the setting of our application. Section 3.5 presents the performance evaluation of our routing strategy.

3.1. Simulation framework

We use the event driven simulation framework provided by TOSSIM [10] to model the behavior of both PD-MAC and S-MAC under a fixed routing strategy as well as to evaluate the performance of the network layer. We use the results obtained from the simulation studies of PD-MAC and S-MAC to validate the analytical models presented in subsequent subsections. TOSSIM implements an event queue containing events scheduled for specific instants as per a global clock. The following code fragment shows relevant fields of the event structure

```
typedef struct sim_event
{
  sim_time_t time;          //scheduled execution time
  unsigned long  mote;        //node-id for node affected by this event
  void* data;                   //pointer to memory location that stores associated data for event, e.g.
      MAC frame contents for a send/receive event
    void (*handle)(sim_event_t* e);     //function pointer to the event handler
}sim_event_t;
```

Events are executed in the order of increasing value of the *time* field of the entries stored in the event queue, and the global clock reading is advanced appropriately. For example, the following code fragment demonstrates an event called *start-ping* being created for currently executing node (TOS_NODE_ID) that causes it to send a ping signal to all the nodes that are listening to the channel in its neighborhood in drowsy mode. The event is scheduled 10 s after the current time.

```
sim_event_t* evt = (sim_event_t*)malloc(sizeof(sim_event_t));
evt->mote = TOS_NODE_ID;
evt->time = sim_time() + sim_ticks_per_sec()*10;
evt->handle = start_ping_handle;
sim_queue_insert(evt);
```

When the entry representing this event is at the head of the event queue, the queue handler of TOSSIM advances the global clock reading to the value in the *time* field of the event and invokes the appropriate callback function (*void start_ping_handle(sim_event_t*)*). We use this simulation framework to simulate the behavior of S-MAC and PD-MAC for our 5×5 nodes network. The timing behavior of the various events of interest is stored as a trace where each line of text denotes a particular event at a particular node. For example, the following trace fragment indicates that node 0 started transmitting ping at instant 24345666 (units of simulation ticks). In this case, node 1 is able to capture the ping, whereas node 2 misses it.

```
.
.
.
1  Begin  Drowsy 24340853
2  Begin  Drowsy 24340987
0  Begin  Tx-ping 24345666
1  Begin  Rx-ping 24345666
.
.
.
```

To model the frequency drift of the nodes' crystal oscillators, we use a uniform random number generator centered at the scheduled wake-up time and a range of $\pm\Delta$. Δ is computed as the product of the sleep duration and the frequency drift in parts per million (δ_{ppm}) of the crystal oscillator (manufacturer specified value). We used a drift of 30 ppm for an oscillator of frequency 1 MHz in our simulations.

$$\Delta = \delta_{\text{ppm}} \times T_{\text{sleep duration}} \tag{1}$$

Knowing the scheduled wake-up time of a node, an event is inserted in the event queue, scheduled for an instant picked uniformly randomly in the interval $[T_{\text{scheduled}} - \Delta, T_{\text{scheduled}} + \Delta]$. At the end of the simulation run, a python script computes the data count, latency, and energy consumption based on the trace.

We implemented MAC and PHY layers in nesC and route and schedule computations (network) in Python script. Python scripts were also used to control the simulation runs such as accessing the various nodes in the sensor field to inspect internal variables, switching nodes on or off, and injecting schedule messages into the network.

To compute the energy consumption, we used the values for the current drawn from the battery as per the datasheet of CC1110 for various modes of the radio. The values are summarized in Table I. The values are for the radio operating in the frequency band of 433 MHz at a datarate of 1.2 KBauds and at a system clock frequency of 26 MHz.

3.2. Modeling data count at sink

The goal of the network is to deliver all the data units generated in the sensor field to the sink node. However, because of channel noise and other sources of signal impairment, some data units are lost. We model the probability distribution of the number of data readings collected at each node at the end of a round. $P_D^m(i)$ denotes the probability that node m gathers i data readings (including locally generated reading) after it has attempted to collect data from its upstream neighbors, U_m. We evaluate $P_D^m(i)$ recursively as shown in Equation (2). The leaf nodes of the routing tree do not have any upstream neighbors and so they collect exactly one data unit. Therefore, the base case of the recursion is: For all leaf nodes m, $P_D^m(i) = 1$ for $i = 1$, and $P_D^m(i) = 0$ for $i > 1$. For the non-leaf nodes, $P_D^m(i)$ is given by the following formula:

$$P_D^m(i) = \sum_{\substack{L_m \subseteq U_m: \\ \sum_{k\in L_m} i_k = i-1}} \left\{ \left(\prod_{k\in L_m} P_D^k(i_k) P_{\text{suc}}(i_k) \right) \cdot \left(\prod_{k\in U_m \setminus L_m} P_D^k(i_k) P_{\text{fail}}(i_k) \right) \right\}. \tag{2}$$

The aforementioned expression computes the probability that a subset of upstream neighbors, $L_m \subseteq U_m$, collectively transmits $i-1$ data units to node m, where the upstream neighbor $k \in L_m$ transmits i_k data units such that $\sum_{k\in L_m} i_k = i - 1$. $P_{\text{suc}}(i_k)$ denotes the probability that synchronization is established in at most N_s attempts and a data packet containing i_k data units is transmitted

Table I. Values of model parameters.

Parameter	Value	Description
q	0.1	Probability that transmitted ping is not detected
p_e	0.01	Bit error rate for data packets
H	8 bits	MAC frame header bits
D	8 bits	MAC frame payload bits per data unit
S	8 bits	MAC frame payload bits for synchronization packets in S-MAC
Δ	2.592 s	Absolute maximum clock drift when nodes wake up for communication
N_d	3	Total number of data attempts allowed
BPS	1200 bps	Bit rate of channel
T_{SYN}	0.1 s	Duration of ping
P_t	15 mA	Current drawn during regular transmit mode
P_p	33.5	Current drawn during ping mode
P_r	19.8	Current drawn during receive mode at regular sensitivity
P_d	10.0 (assumed)	Current drawn during receive mode at reduced sensitivity

successfully in at most N_d attempts, whereas $P_{\text{fail}}(i_k) = 1 - P_{\text{suc}}(i_k)$.

$$P_{\text{suc}}(i_k) = \sum_{s=1}^{N_s} (1-q)q^{s-1} \times \sum_{r=1}^{N_d} \left((1-p_e)^{D(i_k)} (1-(1-p_e)^{D(i_k)})^{r-1} \right), \quad (3)$$

where q is the probability of synchronization failure, p_e is the bit error rate, $D(i_k)$ denotes the bit length of a data packet containing i_k data units. The expression for the data-count probability for PD-MAC and S-MAC remains the same as given in Equations (2) and (3) except for the value of the parameter q. For S-MAC, this is computed in accordance with the bit error probability and the number of bits in the synchronization packet, whereas for PD-MAC, this denotes the probability of detection failure of ping, which is a much lower value owing to its shorter length and no data content.

For our performance evaluation studies, we used bit error rate $p_e = 0.01$, ping detection error probability $q = 0.1$, bits per data unit $D = 8$, header bits per packet $H = 8$, payload bits in an S-MAC synchronization request/reply packet $S = 8$, ping pulse duration $T_{\text{SYN}} = 0.1$s, and baud-rate $BPS = 1200$ bps. The parameters are summarize in Table I. The expected data count at the sink-node 0 is given by $\sum_{i=1}^{N} i P_D^0(i)$, where N is the total number of nodes in the field. Figure 7 shows the expected data count at node 0 versus the number of synchronization attempts allowed for the two protocols. Both protocols have comparable performance as far as the data count at the sink node is concerned. This behavior is expected because we allow the same number of synchronization attempts and data transmission attempts for the two MAC protocols. The difference is in the value of the probability of synchronization. But this difference only matters when the number of synchronization attempts is small (1 or 2). Next, we compare the two protocols with respect to the delay.

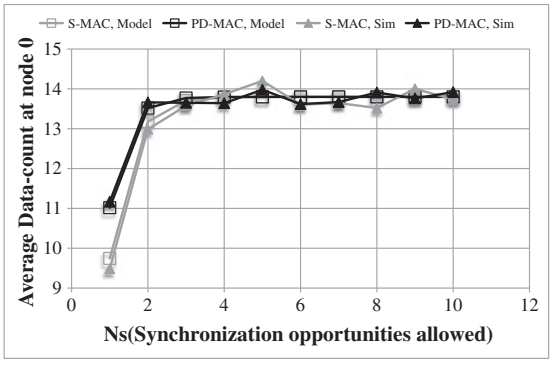


Figure 7. Expected data count stored at sink.

3.3. Modeling latency or round duration

For the analysis, we divide the round duration into delays encountered over multiple hops in the field. Define the *communication time* of a node as the time taken by a node to synchronize with and collect data from all its upstream senders. Round duration is the sum of the communication times for all nodes in the routing tree. In the following two subsections, we model the communication time for a node for the two MAC protocols.

3.3.1. Modeling communication time for S-MAC.

Owing to the pairwise communications between sender–receiver pairs, in case of S-MAC, the total communication time for a node is given by the sum of the communication times corresponding to each of its upstream neighbors. Let T_{km} denote the expected communication time for a receiver node m, corresponding to an upstream neighbor k as modeled in Equation (4). Here, $P_{\text{sync}}(i)$ denotes the probability that synchronization succeeds in ith attempt ($i \leq N_s$), and $P_{\text{sync-fail}}$ denotes the probability that synchronization fails all N_s attempts. If synchronization fails N_s attempts, data phase does not take place. However, if synchronization succeeds in at most N_s attempts, the two nodes attempt to exchange the data packet for up to a maximum of N_d attempts.

$$T_{km} = \sum_{i=1}^{N_s} P_{\text{sync}}(i) \times \left[T_{\text{sync}}(i) + T_{\text{data}}^k \right] + P_{\text{sync-fail}} \times T_{\text{no-sync}}, \quad (4)$$

where $T_{\text{sync}}(i)$ denotes the expected duration of the synchronization phase given that synchronization succeeds in i^{th} attempt, T_{data}^k denotes the expected data phase duration consisting of a maximum of N_d attempts (its value depends on k as different upstream neighbors have different number of data units to send), and $T_{\text{no-sync}}$ denotes the expected duration of the synchronization phase when synchronization fails all N_s attempts. In the latter case, no data phase takes place. Equation (5) computes $T_{\text{sync}}(i)$ which denotes the time period from the awakening of the first of the nodes m or k to synchronization in the i^{th} attempt, whereas the formulae for $T_{\text{no-sync}}$ and T_{data}^k are given in Equations (6) and (7), respectively.

$$T_{\text{sync}}(i) = \left\lfloor \frac{i+1}{2} \right\rfloor (T_{DD} + \{(i+1) \bmod 2\}) \mathbb{E}(Y). \quad (5)$$

Equation (5) can be understood as follows: two nodes that are attempting to synchronize with each other take turns in transmitting synchronization request frames. Therefore, if synchronization occurs in i^{th} attempt, the node that starts first takes a total of $\lfloor \frac{i+1}{2} \rfloor$ turns, each of which lasts for the discovery duration of T_{DD}. In addition, if synchronization succeeds in an even-numbered attempt, an additional delay corresponding to the difference in the wake-up times of the two nodes, denoted by Y

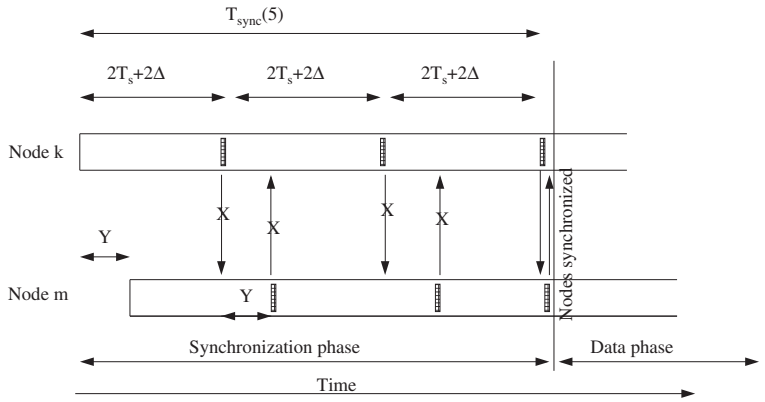


Figure 8. Synchronization phase in S-MAC. Y represents the absolute difference between the wake up times of the sender and receiver nodes.

in Equation (5), is also incurred denoting synchronization request frame transmission by node that wakes up second. For example, Figure 8 shows $T_{\text{sync}}(5)$ and Y. Similarly,

$$T_{\text{no-sync}} = \left\lfloor \frac{N_s + 1}{2} \right\rfloor (T_{DD}) + \{(N_s + 1) \bmod 2\}\, \mathbb{E}(Y). \tag{6}$$

Note, $Y = |X_k - X_m|$, where the random variables for wake-up time of k and m, $X_k, X_m \sim U(0, 2\Delta)$ have density functions $f_{X_m}(x)$, $f_{X_k}(x)$. We can derive the distribution of Y as follows:

$$\begin{aligned} f_Y(y) &= \int_{0 \le x \le 2\Delta} [f_{X_m}(x-y) + f_{X_m}(x+y)] \\ &\quad \times f_{X_k}(x)\mathrm{d}x \\ &= \int_{x=y}^{2\Delta} f_{X_m}(x-y) f_{X_k}(x)\mathrm{d}x \\ &\quad + \int_{x=0}^{2\Delta-y} f_{X_m}(x+y) f_{X_k}(x)\mathrm{d}x \\ &= \int_{x=y}^{2\Delta} \frac{1}{2\Delta}\frac{1}{2\Delta}\mathrm{d}x + \int_{x=0}^{2\Delta-y} \frac{1}{2\Delta}\frac{1}{2\Delta}\mathrm{d}x \\ &= \frac{2\Delta - y}{2\Delta^2}. \end{aligned}$$

Then, the expected value of Y is given by

$$\mathbb{E}(Y) = \int_0^{2\Delta} y \frac{2\Delta - y}{2\Delta^2}\mathrm{d}y = \frac{2}{3}\Delta.$$

We now compute T^k_{data}, the expected time to transmit data by an upstream neighbor k that is synchronized in up to N_s attempts. Let R_k denote the routing subtree rooted at node k. Then, because this subtree has size $|R_k|$ and because each node senses one data unit, node k can have at most $|R_k|$ data units to transmit to its downstream node m (some data packets may be lost). The probability distribution of the number of data packets contained at every node is expressed by Equation (2). Let $T_D(l)$ denote the time slot for a MAC frame containing l data unit, and $T_A(s)$ denote the time slot for a MAC frame containing an acknowledgement vector for s number of senders (for S-MAC $s = 1$ always). Note that $T_D(l) = D(l)/BPS$ and $T_A(s) = A(s)/BPS$, where BPS denotes the baud-rate of the radio transceiver, and $A(s)$ denotes the bit length of an acknowledgement packet containing a bit vector of length s. Then, the size of time slot needed to accommodate the maximum-sized data from node k (of size $|R_k|$) and one acknowledgement frame for node k is given by $T^k_{DA} = T_D(|R_k|) + T_A(1)$. We can model the expected data phase duration as follows:

$$\begin{aligned} T^k_{\text{data}} &= \sum_{l=1}^{|R_k|} P^k_D(l) \\ &\quad \times \left(\sum_{r=1}^{N_d} P^r_{\text{suc}}(l).r.T^k_{DA} + P^{N_d}_{\text{fail}}(l).N_d.T^k_{DA} \right), \end{aligned} \tag{7}$$

where $P^k_D(l)$ is defined in Equation (2), $P^r_{\text{suc}}(l)$ is the probability of successful transmission of a data packet containing l data unit in the r^{th} attempt. The packet error rate for a data packet containing l data unit is given by $p_l = 1 - (1 - p_e)^{D(l)}$. Hence, the formulae for $P^r_{\text{suc}}(l)$ and $P^{N_d}_{\text{fail}}(l)$ are as follows:

$$P^r_{\text{suc}}(l) = (1 - p_l)p_l^{r-1}; \quad P^{N_d}_{\text{fail}}(l) = p_l^{N_d}. \tag{8}$$

3.3.2. Modeling communication time for PD-MAC.

In PD-MAC, node m attempts to synchronize all of its upstream neighbors U_m using a common ping. Each upstream neighbor is allocated a time slot that is long enough to accommodate a data packet that contains data from all the nodes in the routing subtree rooted at that neighbor. The time slots are scheduled relative to the end of the ping transmission. As in the case of S-MAC, there

are N_s synchronization attempts. After every synchronization attempt, a total of N_d attempts are allocated for every upstream neighbor. Only the nodes, synchronized by the most recent ping, attempt data transmissions. Every data attempt terminates with a collective acknowledgement message transmitted by node m. Here, we obtain a recursive model for the communication time of a node m.

Let $T^m(i, r, S_m, D_m, Q_m)$ denote the remaining communication time, given that $i-1$ synchronization attempts have taken place and $r-1$ data attempts associated with the i^{th} synchronization attempt have taken place. $S_m \subseteq U_m$ denotes the subset of upstream neighbors that are synchronized so far, $D_m \subseteq S_m$ denotes the subset of upstream nodes that have successfully delivered their data so far, and $Q_m \subseteq U_m \setminus S_m$ denotes the subset of upstream nodes that have synchronized in the most recent (i^{th}) synchronization attempt. Q_m gets updated only following a synchronization attempt (and remains unaltered following a data transmission attempt). A recursive model for $T(i, r, S_m, D_m, Q_m)$ is given by Equation (9), where T_{SYN} denotes the duration of ping, and $\widehat{T}^m_{DA} = T_A(|U_m|) + \sum_{k \in U_m} T_D(|R_k|)$ represents the duration of entire time slot accommodating data packets by all senders and acknowledgement packet by the downstream node m. Then, $T(1, 1, \emptyset, \emptyset, \emptyset)$ gives the total communication time for node m starting from the configuration when $i = 1$ (no synchronization attempt has been made), $r = 1$ (no data attempt has been made), $S_m = \emptyset$ (no upstream neighbors have synchronized), $D_m = \emptyset$ (no upstream neighbors have send their data), and $Q_m = \emptyset$.

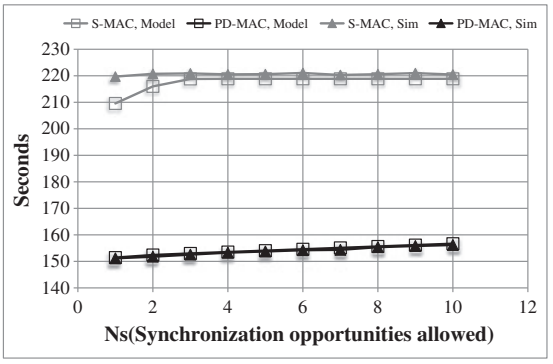


Figure 9. Expected round duration versus the number of synchronization opportunities.

$$T(i, r, S_m, D_m, Q_m) = \begin{cases} 0, & \text{if } C := [(D_m = U_m) \vee (i = N_s + 1)] \\ T_{\text{SYN}} + \sum_{R_m \subseteq U_m \setminus S_m} \sum_{C_m \subseteq R_m} \widehat{T}^m_{DA} + q^{|U_m \setminus S_m \setminus R_m|}(1-q)^{|R_m|} P_{\text{suc}}(C_m) P_{\text{fail}}(R_m \setminus C_m) \\ T(i + \lfloor (r/N_d) \rfloor, (r \bmod N_d) + 1, S_m \cup R_m, D_m \cup C_m, R_m), & \text{if } \neg C, r = 1 \\ \sum_{C_m \subseteq Q_m \setminus D_m} \widehat{T}^m_{DA} + P_{\text{suc}}(C_m) P_{\text{fail}}(Q_m \setminus D_m \setminus C_m) \\ T(i + \lfloor (r/N_d) \rfloor, (r \bmod N_d) + 1, S_m, D_m \cup C_m, Q_m), & \text{if } \neg C, r > 1. \end{cases} \tag{9}$$

Case 1 in Equation (9) denotes the base condition: The remaining communication time $T(S_m, D_m, Q_m, i, r)$ is zero if node m has already received data from all its upstream neighbors (captured by the condition: $D_m = U_m$), or if it has already expended all N_s synchronization attempts and N_d data attempts associated with each synchronization attempt (captured by the condition: $i = N_s + 1$). Case 2 models the i^{th} ping transmission which takes a duration of T_{SYN} and the ensuing data attempt which takes a duration of $\widehat{T}^m_{DA}$. In a synchronization attempt (captured by the condition $r = 1$), a subset $R_m \subseteq U_m \setminus S_m$ may get synchronized (captured in the first summation) with a probability of $q^{|U_m \setminus S_m \setminus R_m|}(1-q)^{|R_m|}$ and participate in at most N_d data attempts. During the first data attempt, a subset $C_m \subseteq R_m$ of nodes may be successful in data transmission (captured in the second summation) with a probability of $P_{\text{suc}}(C_m) P_{\text{fail}}(R_m \setminus C_m)$. Case 3 represents the r^{th} data attempt for $r > 1$, which takes the duration $\widehat{T}^m_{DA}$. During a data attempt, a subset $C_m \subseteq Q_m \setminus D_m$ of newly synchronized nodes may be successful in transmitting data with a probability of $P_{\text{suc}}(C_m) P_{\text{fail}}(Q_m \setminus D_m \setminus C_m)$. (Note that only the newly synchronized nodes attempt to send data, because the nodes previously synchronized already expended their allowed data attempts.)

The results for the round duration (= sum of communication times of all nodes on the routing tree) of the analytical models as well as the simulation studies for both S-MAC and PD-MAC are plotted against the number of synchronization attempts in Figure 9. It should be noted that PD-MAC achieves a 25% shorter round duration than S-MAC. This is because PD-MAC combines the wake-up synchronization of all the upstream neighbors of a node and schedules the data communications in a time division manner once synchronization is established. Such combination is not allowed in S-MAC.

3.4. Computation of energy consumption

To model the energy consumption of a node, we split the communication time given by Equations (4) and (9) into durations of constant power consumption for each node. Multiplying the power consumed in various radio modes such as idle, transmit, receive, drowsy, and ping (a constant voltage times the current drawn) with the respective durations of the modes, we arrive at the model for the energy consumed.

3.4.1. Energy consumption model for S-MAC.

The communication time T_{km} in Equation (4) is expressed in terms of $T_{\text{sync}}(i)$, T^k_{data}, and $T_{\text{no-sync}}$. $T_{\text{sync}}(i)$, consists of three parts given by Equations (10)–(12), each with its own level of transmitter and/or receiver power consumption

$$T^{\text{sync}}_{\text{idle}}(i) = \frac{1}{2}\left(\left\lfloor \frac{i+1}{2} \right\rfloor T_{\text{DD}} + ((i+1) \bmod 2)\mathbb{E}(Y) - (i+1)T_S\right) + \frac{1}{2}\left(\left\lfloor \frac{i}{2} \right\rfloor T_{\text{DD}} + (i \bmod 2)(T_{\text{DD}} - \mathbb{E}(Y)) - (i+1)T_S\right), \quad (10)$$

$$T^{\text{sync}}_{\text{tx-sync}}(i) = \frac{1}{2}\left\lfloor \frac{i+2}{2} \right\rfloor T_S + \frac{1}{2}\left\lfloor \frac{i+1}{2} \right\rfloor T_S, \quad (11)$$

$$T^{\text{sync}}_{\text{rx-sync}}(i) = T^{\text{sync}}_{\text{tx-sync}}(i) + \frac{1}{2}T_S. \quad (12)$$

such that

$$T_{\text{sync}}(i) = T^{\text{sync}}_{\text{idle}}(i) + T^{\text{sync}}_{\text{tx-sync}}(i) + T^{\text{sync}}_{\text{rx-sync}}(i), \quad (13)$$

$T_{\text{no-sync}}$ can be split in a fashion similar to Equations (10)–(12).

$$T_{\text{no-sync}} = T^{\text{no-sync}}_{\text{idle}} + T^{\text{no-sync}}_{\text{tx-sync}} + T^{\text{no-sync}}_{\text{rx-sync}}, \quad (14)$$

where

$$T^{\text{no-sync}}_{\text{idle}} = T^{\text{sync}}_{\text{idle}}(N_s) + T_S, \quad (15)$$

$$T^{\text{no-sync}}_{\text{tx-sync}} = \frac{1}{2}\left\lfloor \frac{N_s+1}{2} \right\rfloor T_S + \frac{1}{2}\left\lfloor \frac{N_s}{2} \right\rfloor T_S, \quad (16)$$

and

$$T^{\text{no-sync}}_{\text{rx-sync}} = T^{\text{no-sync}}_{\text{tx-sync}}. \quad (17)$$

Similarly, T^k_{data} consists of two parts given by Equations (19)–(20), each with its own level of transmitter and receiver power consumption

$$T^k_{\text{tx-data}} = \sum_{l=1}^{|R_k|} P^k_D(l) \times \left\{\sum_{r=1}^{N_d} P^r_{\text{suc}}(l).r.T_D(l) + P^{N_d}_{\text{fail}}(l).N_d.T_D(l)\right\} \quad (18)$$

$$= T^m_{\text{rx-data}}, \quad (19)$$

$$T^m_{\text{tx-ack}} = \sum_{l=1}^{|R_k|} P^k_D(l) \times \left(\sum_{r=1}^{N_d} P^r_{\text{suc}}(l).r.T^1_A + P_{\text{fail}}(l).N_d.T^1_A\right) = T^k_{\text{rx-ack}}. \quad (20)$$

such that

$$T^k_{\text{data}} = T^k_{\text{tx-data}} + T^m_{\text{tx-ack}} = T^m_{\text{rx-data}} + T^k_{\text{rx-ack}}. \quad (21)$$

Accordingly, following Equations (10)–(21), the total expected energy consumed by node m and its upstream neighbor k can be obtained by splitting the time duration terms on the right-hand side of Equation (4) into durations where power level remains constant and multiplying those time duration terms by the corresponding power values

$$E_{km} = \sum_{i=1}^{N_s} 2.P_{\text{sync}}(i) \times \Big[\{T^{\text{sync}}_{\text{idle}}(i)\}.P_{\text{idle}} + \{T^{\text{sync}}_{\text{tx-sync}}(i)\}.P_{\text{tx}} + \{T^{\text{sync}}_{\text{rx-sync}}(i)\}.P_{\text{rx}} + \{T^k_{\text{tx-data}} + T^m_{\text{tx-ack}}\}.P_{\text{tx}} + \{T^m_{\text{rx-data}} + T^k_{\text{rx-ack}}\}.P_{\text{rx}}\Big] + 2.P_{\text{sync-fail}} \times \left\{T^{\text{no-sync}}_{\text{idle}}.P_{\text{idle}} + T^{\text{no-sync}}_{\text{tx-sync}}.P_{\text{tx}} + T^{\text{no-sync}}_{\text{rx-sync}}.P_{\text{rx}}\right\}. \quad (22)$$

The factor of 2 in case of the time duration terms corresponding to the synchronization phase is used to cover both nodes k and m.

3.4.2. Energy consumption model for PD-MAC.

Similar to the energy computation for S-MAC, we compute the total energy consumption for PD-MAC by splitting the communication time in Equation (9) into parts of constant power consumption for each participating node. In the recursive model, communication time is expressed in terms of T_{SYN} and $\widehat{T}^m_{DA}$. T_{SYN} denotes the ping duration during which downstream node m is in ping mode, whereas an upstream node $k \in U_m$ is either in drowsy mode (if it has not been synchronized yet) or sleep mode (if it has already been synchronized and has already attempted data transmission). We split $\widehat{T}^m_{DA}$ into durations where the power level remains constant for nodes $k \in U_m$ and m. Let M denote the set of upstream nodes that are synchronized in the most recent ping attempt and that transmit data during a particular data phase captured by $\widehat{T}^m_{DA}$ of Equation (9). Note, in Case 2: $M = R_m$, whereas in Case 3: $M = Q_m \setminus D_m$. For $k \in M$, $\widehat{T}^m_{DA}$ can be split into three parts given by Equations (23)–(25)

$$\hat{T}^k_{\text{tx-data}} = \sum_{i_k=1}^{|R_k|} P^k_D(i_k).T_D(i_k), \quad (23)$$

$$\hat{T}^k_{\text{rx-ack}} = T_A(|U_m|), \text{ and} \tag{24}$$

$$\hat{T}^k_{\text{idle}} = \hat{T}^m_{DA} - \hat{T}^k_{\text{tx-data}} - \hat{T}^k_{\text{rx-ack}}, \tag{25}$$

Accordingly, following Equations (23)–(28), the time duration $\widehat{T}^m_{DA}$ in Equation (9) can be split into constant power time slots to compute the total energy consumed by nodes $\{m\} \cup U_m$ by multiplying those constant power time slots by their corresponding power values as shown in Equation (29).

$$E(i, r, S_m, D_m, Q_m) = \begin{cases} 0, \quad \text{if } C := [(D_m = U_m) \vee (i = N_s + 1)] \\ T_{\text{SYN}}.\{P_{\text{ping}} + \sum_{k \in U_m \setminus S_m} P_{\text{drowsy}}\} + \\ \sum_{R_m \subseteq U_m \setminus S_m} \sum_{C_m \subseteq R_m} \left\{ \sum_{k \in R_m} \left[T^k_{\text{tx-data}}.P_{\text{tx}} + \hat{T}^k_{\text{rx-ack}}.P_{\text{rx}} + T^k_{\text{idle}}.P_{\text{idle}} \right] + \right. \\ \hat{T}^m_{\text{rx-data}}.P_{\text{rx}} + \hat{T}^m_{\text{tx-ack}}.P_{\text{tx}} + \hat{T}^m_{\text{idle}}.P_{\text{idle}} + q^{|U_m \setminus S_m \setminus R_m|}(1-q)^{|R_m|} P_{\text{suc}}(C_m) P_{\text{fail}}(R_m \setminus C_m) \\ \left. E(i + \lfloor (r/N_d) \rfloor, (r \bmod N_d) + 1, S_m \cup R_m, D_m \cup C_m, R_m) \right\}, \quad \text{if } \neg C, r = 1 \\ \sum_{C_m \subseteq Q_m \setminus D_m} \left\{ \sum_{k \in Q_m \setminus D_m} \left[T^k_{\text{tx-data}}.P_{\text{tx}} + \hat{T}^k_{\text{rx-ack}}.P_{\text{rx}} + T^k_{\text{idle}}.P_{\text{idle}} \right] + \right. \\ \hat{T}^m_{\text{rx-data}}.P_{\text{rx}} + \hat{T}^m_{\text{tx-ack}}.P_{\text{tx}} + \hat{T}^m_{\text{idle}}.P_{\text{idle}} + P_{\text{suc}}(C_m) P_{\text{fail}}(Q_m \setminus D_m \setminus C_m) \\ \left. E(i + \lfloor (r/N_d) \rfloor, (r \bmod N_d) + 1, S_m, D_m \cup C_m, Q_m) \right\}, \quad \text{if } \neg C, r > 1. \end{cases} \tag{29}$$

where $\hat{T}^k_{\text{tx-data}}$, $\hat{T}^k_{\text{rx-ack}}$, and $\hat{T}^k_{\text{idle}}$ represent the expected time durations spent by node k in regular transmit mode to transmit data frames, regular receive mode to receive acknowledgement frames and idle mode, respectively. We make the assumption that node k does not overhear any data packets transmitted by other upstream neighbors of node m during the period $\widehat{T}^m_{DA}$.

An upstream node $k \in U_m$ that has not been synchronized yet spends the entire duration $\widehat{T}^m_{DA}$ in drowsy mode. This happens in Case 2 when: $k \in U_m \setminus S_m \setminus R_m$, whereas in Case 3 when $k \in U_m \setminus S_m$.

All other nodes $k \in U_m$ that have been synchronized and attempted the allotted data transmissions spend the entire duration $\widehat{T}^m_{DA}$ in sleep mode.

On the other hand, the downstream node m spends energy, during the period $\widehat{T}^m_{DA}$, in idle/transmit/receive modes, until it receives data from all nodes $k \in U_m$, that is, until the condition $D_m = U_m$ is met. When $D_m = U_m$, node m goes to sleep.

If $D_m \neq U_m$, $\widehat{T}^m_{DA}$ can be split into three parts given by Equations (26)–(28)

$$\hat{T}^m_{\text{rx-data}} = \sum_{k \in L} \hat{T}^k_{\text{tx-data}}, \tag{26}$$

$$\hat{T}^m_{\text{tx-ack}} = T_A(|U_m|), \text{ and} \tag{27}$$

$$\hat{T}^m_{\text{idle}} = \widehat{T}^m_{DA} - \sum_{k \in L} \hat{T}^k_{\text{tx-data}} - T_A(|U_m|), \tag{28}$$

where $\hat{T}^m_{\text{rx-data}}$, $\hat{T}^m_{\text{tx-ack}}$, and $\hat{T}^m_{\text{idle}}$ represent the expected time durations spent by node m in regular receive mode to receive data frames, regular transmit mode to transmit acknowledgement frames and idle receive mode, respectively.

We estimate the power consumed in a particular mode as the product of the voltage and the current drawn. Because the voltage is approximately a constant, we use the value of current in place of power throughout our analytical model to get a normalized value of the energy consumption. We used the values for the current drawn from the battery as per the datasheet of CC1110 for various modes of the radio [6], see Table I. For the proof of concept, we have implemented PD-MAC for the case of two nodes; that is, one uplink node and one downlink node [1].

The results for the total energy consumption (= sum of energy consumption of all nodes in the field) from the analytical model as well as simulations for both S-MAC and PD-MAC are plotted against the number of synchronization attempts in Figure 10. It should be noted that PD-MAC achieves 65% lower energy consumption compared with S-MAC. This is a result of the proposed PD-MAC that uses alternative power modes for radios during wake-up synchronization and also its clever exploitation of the convergecast nature of communication to minimize the number of high energy pings.

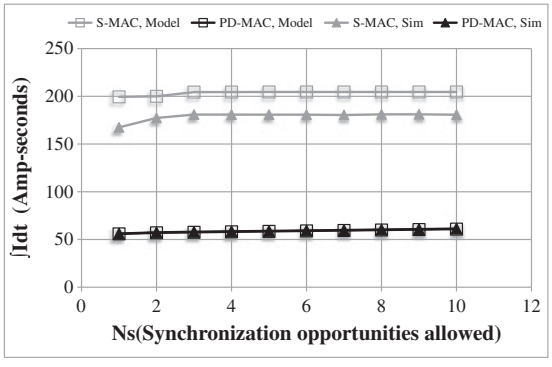


Figure 10. Expected energy consumption per node.

3.5. Load balancing under proposed routing

In our network, data flows over multiple hops from the sensor nodes to the designated sink. Because of differing lengths of the data frames transmitted and received by nodes depending on their location in the routing tree, energy consumption is not uniform throughout the field. Nodes that are closer to the sink node consume more energy than nodes that are further away. This makes the nodes close to the sink drain their energy faster than the rest of the field making the sink node partitioned from the rest of the network even when most of the nodes in the rest of the network may have sufficient battery energy. A strategy for load balancing that we use is to rotate the role of the sink node among the four corner nodes.

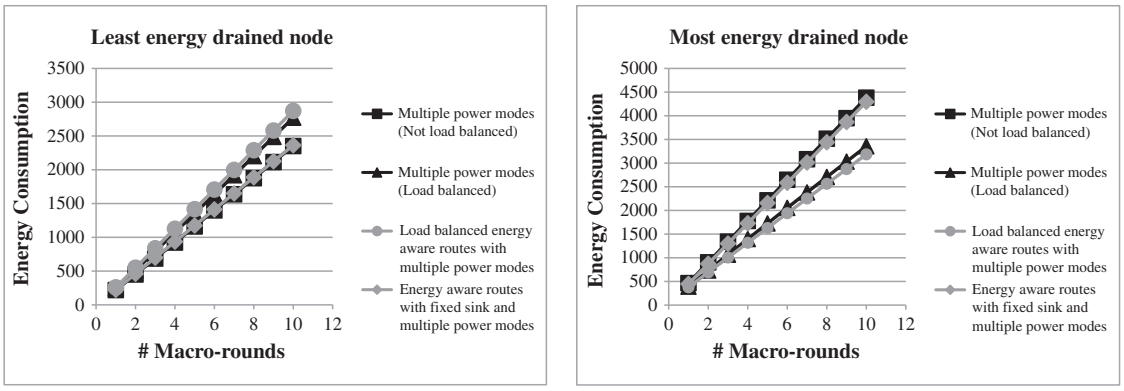
Further, we use energy aware routing strategy that chooses the next hop for a given node as the neighboring node in the forward path to the sink that has the maximum level of remaining energy. We use simulation studies to compare the energy consumption for our routing strategy versus the one in which a next hop is chosen uniformly randomly among the neighboring nodes in the forward path to the sink. We also compare with the case where sink node remains fixed (is not rotated).

Figure 11 shows the expected energy consumed for the least energy drained node and the most energy drained node as a function of the number of macro-rounds for a 5×5 sensor node field for different combinations of the routing strategy and sink assignment. Note that energy aware routing coupled with a load balanced strategy for sink assignment leads to the most balanced energy consumption in the field. The difference grows linearly in the number of macro-rounds at the rate of about 330 units per macro-round. This shows that an energy aware and load balanced network layer coupled with PD-MAC helps increase the network lifetime by preventing the node failures because of unbalanced energy consumption.

4. RELATED WORK

There has been much work on designing energy efficient network protocols for sensor networks. Energy may be unnecessarily consumed during collisions, overhearing, control packet overheads, idle listening, over-remitting, and so on. Time division multiple access (TDMA)-based protocols avoid collisions and overhearing but at the cost of higher overhead in control packets and synchronization. A number of MAC protocols for wireless sensor networks designed with the goal of energy efficiency are based on a duty cycle-based approach. S-MAC [3], T-MAC [5], and DS-MAC [11] reduce idle listening by transitioning the transceiver through sleep and active cycles. Such MAC protocols trade-off increased latency for reduced energy consumption. In S-MAC, the nodes in the same virtual cluster determined by the communication range of the nodes share the same sleep–listen schedule. The nodes on the boundary of two clusters follow two different sleep–listen schedules. In T-MAC, during the listen period, if the node does not detect any activation event for a certain timeout period, it goes back to sleep. This reduces the idle listening period at the cost of increased probability of early sleeping caused by loss of synchronization between the nodes. DS-MAC adds the dynamic duty cycle feature to S-MAC to decrease the latency.

B-MAC [12], WiseMAC [4], and X-MAC [8] use extended preamble accompanying every transmitted frame with periodic preamble sampling at the receiver. Extended preamble with preamble sampling-based approaches allow a very short duty cycle for nodes at the cost of longer preambles while transmitting packets, causing unnecessary energy consumption in the form of over-remitting. Each data packet is preceded by a long preamble to alert the receiver of the oncoming transmission. The nodes periodically wake up to listen to any activity on the channel. So, there is unnecessary energy consumption at nodes which are not the intended receivers of the frame but happen to



(a) (b)

Figure 11. Expected energy consumption of (a) least energy drained node and (b) most energy drained node versus the number of macro-rounds.

be in the transmission range of the sender node. S-MAC, T-MAC, and DS-MAC are synchronized MAC protocols, in that nodes exchange synchronization messages to negotiate the beginning and end of the active and sleep periods. B-MAC, WiseMAC, and X-MAC are asynchronous MAC protocols in that they do not involve the overhead of periodic synchronization message exchange. Synchronization takes place as a part of the communication itself. A duty cycle-based MAC protocol can lead the sensor nodes to idle listen for long periods while transceivers are switched between active and sleep modes during the time when no data is being relayed in the application. Synchronized MAC protocols additionally involve the overhead involved in the maintenance of synchronization between the neighboring nodes' sleep–active cycles. For a periodic data monitoring application with long sleep periods such as ours, it is useful to have a MAC protocol that can allow the transceivers to be turned off for long periods and wake up to establish communication with the minimal overhead cost.

On the other hand, there has been a lot of work on performance evaluation of sensor network protocols to compute the throughput, latency, and energy consumption. Zamalloa *et al.* [13] presents an analytical model to compare different forwarding strategies used in geographic routing over lossy links based on a realistic channel model. The authors consider a low data rate wireless sensor network assuming no packet collisions and time division multiple access-based MAC layer. A comparison of distance-based, reception-based, and a combination of the two forwarding strategies is performed using analytical models and simulation studies. It is shown that the suitability of the metric depends on whether Automatic Repeat reQuest (ARQ) is used or not. They also derive an analytical model for the optimal forwarding distance to maximize the expected value of the packet reception rate. However, in that work, it is assumed that the node radios are always powered on. Although computing the energy efficiency, the energy consumed by the nodes idle listening the channel is not considered. Lee *et al.* [14] presents a discrete time Markov chain model for dynamic low power listening scheme used in duty cycle-based MAC protocols used in conjunction with long preambles (such as that in B-MAC and WiseMAC) preceding every frame transmitted. The polling interval is adjusted based on the channel conditions. The authors study the effect of the busy and idle thresholds on the energy consumption. The model considers one sender and one receiver node. Bianchi [15] introduces a Markov chain model for the distributed coordinated function (that uses CSMA/CA) of 802.11 MAC protocol standard for wireless local area networks to compute the saturation throughput under ideal channel conditions and a fixed number of terminals. Only single-hop communication is considered. Also, it is assumed that packets are generated at the individual terminals using a Poisson random process and are always available for transmission; that is, the transmissions queues at individual terminals are always non-empty. In our precision agriculture application, packets are generated only intermittently with a certain period. There are bursts of channel activity followed by long periods of inactivity. We have a deterministic generation of data; therefore, using CSMA/CA would mean extra overhead in energy and delay costs. Rusli *et al.* [16] presents a Markov chain-based model of opportunistic routing protocol that exploits the spatial and temporal characteristics of a wireless network to compute the end-to-end delay and reliability metrics.

5. CONCLUSION

We have presented the design of our Physical, MAC, and network layers for a precision agriculture application with periodic data collection. We have shown that the use of multiple power modes at the physical layer increase the energy efficiency of the wake-up synchronization phase of communication. We support our claims using our performance modeling approach based on analytical modeling to compare metrics such as energy consumption, communication delay, and throughput of a wireless sensor network employed for a periodic monitoring application. We have successfully tested our MAC protocol in hardware employing the new wake-up synchronization strategy for one sender and one receiver case. We also present our analytical model for S-MAC under the settings of our application. Although the two protocols can achieve the same packet success rates, the proposed new protocol is able to accomplish this in 25% less time and 65% less energy (as analyzed and simulated for a simple sensor field of 5×5 nodes). The results of the analytical model match closely with our event driven simulations.

We have also shown the gains in load balancing achieved using our energy aware routing strategy using simulations. Our future work plan includes enhancing the protocol to incorporate fault management (both detection and mitigation).

ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation under the grants NSF-ECS-0424048, NSF-ECS-0601570, NSF-ECCS-0801763, NSF-CCF-081141, and NSF-ECCS-0926029.

REFERENCES

1. Sahota H, Kumar R, Kamal A, Huang J. An energy-efficient wireless sensor network for precision agriculture. In *Proceedings IEEE Symposium on Computers and Communications*. IEEE Computer Society:

Riccione, Italy, June 2010; 347–350. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/ISCC.2010.5546508.
2. Sahota H, Kumar R, Kamal A. Performance modeling and simulation studies of MAC protocols in sensor network performance. In *Proceedings International Conference on Wireless Communications and Mobile Computing*. ACM: Istanbul, Turkey, July 2011. [Online]. Available: ....
3. Ye W, Heidemann J, Estrin D. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking* 2004; **12**(3): 493–506. [Online]. Available: http://dx.doi.org/10.1109/TNET.2004.828953.
4. El-Hoiydi A, Decotignie J-D. Wisemac: an ultra low power mac protocol for the downlink of infrastructure wireless sensor networks, In *Proceedings IEEE Symposium on Computers and Communications*, Alexandria, Egypt, 2004; 244–251.
5. van Dam T, Langendoen K. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings SenSys '03*. ACM: Los Angeles, CA, USA, 2003; 171–180. [Online]. Available: http://doi.acm.org/10.1145/958491.958512.
6. True system-on-chip with low power rf transceiver and 8051 mcu. Texas Instruments, 9–11, cc1110 datasheet. [Online]. Available: http://www.ti.com/lit/gpn/cc1110f32.
7. Schmid T. Time in wireless embedded systems, *Ph.D. dissertation*, University of California, Los Angeles, 2009. [Online]. Available: projects.nesl.ucla.edu/~thomas/pdfs/dissertation_schmid.pdf.
8. Buettner M, Yee GV, Anderson E, Han R. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, SenSys '06. ACM: New York, NY, USA, 2006; 307–320. [Online]. Available: http://doi.acm.org/10.1145/1182807.1182838.
9. Ruzzelli AG, Jurdak R, O'Hare GM. On the rfid wake-up impulse for multi-hop sensor networks. In *Proceedings SenSys '07*. ACM: Sydney, Australia, November 2007.
10. Levis P, Lee N, Welsh M, Culler D. Tossim: accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03. ACM: New York, NY, USA, 2003; 126–137. [Online]. Available: http://doi.acm.org/10.1145/958491.958506.
11. Lin P, Qiao C, Wang X. Medium access control with a dynamic duty cycle for sensor networks. In *Proceedings IEEE Wireless Communications and Networking Conference*, Vol. 3. IEEE: Atlanta, GA, USA, March 2004; 1534–1539.
12. Polastre J, Hill J, Culler D. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys '04. ACM: New York, NY, USA, 2004; 95–107. [Online]. Available: http://doi.acm.org/10.1145/1031495.1031508.
13. Zamalloa MZn, Seada K, Krishnamachari B, Helmy A. Efficient geographic routing over lossy links in wireless sensor networks. *ACM Transactions on Sensor Networks* June 2008; **4**: 12:1–12:33. [Online]. Available: http://doi.acm.org/10.1145/1362542.1362543.
14. Lee S, Choi J, Na J, Kim C-k. Analysis of dynamic low power listening schemes in wireless sensor networks. *Communications Letters* January 2009; **13**: 43–45. [Online]. Available: http://portal.acm.org/citation.cfm?id=1650422.1650437.
15. Bianchi G. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications* 2000; **18**: 535–547.
16. Rusli M, Harris R, Punchihewa A. Markov chain-based analytical model of opportunistic routing protocol for wireless sensor networks, In *TENCON 2010 - 2010 IEEE Region 10 Conference*, November 2010; 257–262.

AUTHORS' BIOGRAPHIES

Herman Sahota received his B.Tech. in Electrical Engineering and M.Tech. in Information and Communication Technology from the Indian Institute of Technology, Delhi, India in 2006. He is, currently, a PhD candidate in Electrical and Computer Engineering Department at Iowa State University, Ames, Iowa. Sahota's research interests include protocol design for wireless networks, performance evaluation, and applications of probability and statistics.

Ratnesh Kumar is a professor of ECE at the Iowa State University since 2002. Prior to this, he held faculty position at the University of Kentucky (1991–2002) in ECE and has held visiting positions at the University of Maryland, Applied Research Laboratory (at Penn State University), NASA Ames, Idaho National Laboratory, and United Technologies Research Center. He received his B.Tech. in EE from IIT Kanpur in 1987 and MS and PhD in ECE from the University of

Texas, Austin in 1989 and 1991, respectively. His research interests are in event-driven, real-time, and hybrid systems, and their applications to embedded systems and software, cyberphysical systems, web-services, sensor networks, power systems, and precision farming. He serves on the program committee for the IEEE Control Systems Society, the International Workshop on Discrete Event Systems, the International Workshop on Dependable Control of Discrete Systems, and the IEEE Workshop on Software Cybernetics. He is or has been an associate editor of SIAM Journal on Control and Optimization, IEEE Transactions on Robotics and Automation, Journal of Discrete Event Dynamical Systems, International Journal on Discrete Event Control Systems, IEEE Control Systems Society. He has been a general co-chair for the International Workshop on Discrete Event Systems, and a program co-chair for the IEEE Workshop on Software Cybernetics. He is a fellow of the IEEE.

Ahmed E. Kamal received a BSc (distinction with honors) and an MSc both from Cairo University, Egypt and an MASc and a PhD both from the University of Toronto, Canada, all in Electrical Engineering in 1978, 1980, 1982, and 1986, respectively. He is currently a professor of Electrical and Computer Engineering at Iowa State University. Earlier, he held faculty positions in the Department of Computing Science at the University of Alberta, Canada and the Department of Computer Engineering at Kuwait University, Kuwait. He was also an adjunct professor at the Telecommunications Research Labs, Edmonton, Alberta.

Kamal's research interests include optical networks, wireless sensor networks, cognitive radio networks, and performance evaluation. He is a Fellow of the IEEE, a senior member of the Association of Computing Machinery, and a registered professional engineer. With his students, Kamal received the 1993 IEE Hartree Premium for papers published in Computers and Control in IEE Proceedings for his paper entitled Study of the Behaviour of Hubnet and the best paper award of the IEEE Globecom 2008 Symposium on Ad Hoc and Sensors Networks Symposium.

Kamal served on the technical program committees of numerous conferences and workshops. He was the chair or co-chair of the Technical Program Committees of a number of conferences and symposia including the Optical Networks and Systems Symposia of the IEEE Globecom 2007 and the IEEE Globecom 2010, the IEEE Globecom Cognitive Radios and Networks Symposium in 2012. He is on the editorial boards of the Computer Networks journal and the Optical Switching and Networking journal, both published by Elsevier, and the IEEE Communications Surveys and Tutorials.