



Channel allocation and medium access control for wireless sensor networks

Kaushik R. Chowdhury^{a,*,1}, Nagesh Nandiraju^{b,1}, Pritam Chanda^{c,2}, Dharma P. Agrawal^d, Qing-An Zeng^e

^a Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

^b Motorola Inc., Horsham, PA, USA

^c Department of Computer Science and Engineering, SUNY, Buffalo, New York 14260, USA

^d CDMC, University of Cincinnati, Cincinnati, OH 45221, USA

^e WMN Laboratory, University of Cincinnati, Cincinnati, OH 45221, USA

ARTICLE INFO

Article history:

Received 1 August 2007

Received in revised form 6 March 2008

Accepted 20 March 2008

Available online 11 April 2008

Keywords:

Channel allocation

Coloring

Multi-channel MAC

Sensor network

Wake-up radio

ABSTRACT

Recent developments in sensor technology, as seen in Berkeley's Mica2 Mote, Rockwell's WINS nodes and the IEEE 802.15.4 Zigbee, have enabled support for single-transceiver, multi-channel communication. The task of channel assignment with minimum interference, also named as the 2-hop coloring problem, allows repetition of colors occurs only if the nodes are separated by more than 2 hops. Being NP complete, development of efficient heuristics for this coloring problem is an open research area and this paper proposes the Dynamic Channel Allocation (DCA) algorithm as a novel solution. Once channels are assigned, a Medium Access Control protocol must be devised so that channel selection, arbitration and scheduling occur with maximum energy savings and reduced message overhead, both critical considerations for sensor networks. The contribution of this paper is twofold: (1) development and analysis of the DCA algorithm that assigns optimally minimum channels in a distributed manner in order to make subsequent communication free from both primary and secondary interference and (2) proposing CMAC, a fully desynchronized multi-channel MAC protocol with minimum hardware requirements. CMAC takes into account the fundamental energy constraint in sensor nodes by placing them in a default sleep mode as far as possible, enables spatial channel re-use and ensures nearly collision free communication. Simulation results reveal that the DCA consumes significantly less energy while giving a legal distributed coloring. CMAC, our MAC protocol that leverages this coloring, has been thoroughly evaluated with various modes in SMAC, a recent protocol that achieves energy savings through coordinated sleeping. Results show that CMAC obtains nearly 200% reduction in energy consumption, significantly improved throughput, and end-to-end delay values that are 50–150% better than SMAC for our simulated topologies.

© 2008 Elsevier B.V. All rights reserved.

* Corresponding author. Tel.: +1 404 894 6616.

E-mail addresses: kaushik@ece.gatech.edu (K.R. Chowdhury), nmandiraju@motorola.com (N. Nandiraju), pchanda@cse.buffalo.edu (P. Chanda), dpa@cs.uc.edu (D.P. Agrawal), [qzeng@cs.uc.edu](mailto;qzeng@cs.uc.edu) (Q.-A. Zeng).

¹ This work was completed when the authors were research assistants at the Center for Distributed and Mobile Computing (CDMC), University of Cincinnati.

² This work was completed when the author was with the Wireless and Mobile Networking (WMN) Laboratory, University of Cincinnati.

1. Introduction

Wireless sensor nodes are low power, battery operated devices with limited computation and transmission ability [1]. With improvement in sensor hardware, support for multi-channel communication is already in an advanced stage of implementation. Berkeley's third generation Mica2 Mote has an 868/916 MHz multi-channel transceiver [2]. In Rockwell's WINS nodes, the radio operates on one of

40 channels in the ISM frequency band, selectable by the controller [3]. The new IEEE 802.15.4 standard, popularly called as Zigbee, defines 11 channels at 868/915 MHz and 16 channels at 2.4 GHz [4]. Typically, physical parameters like temperature [5], pressure [6], humidity [7], location [8], amongst others, are sensed and communicated over a shared wireless channel. A medium access control (MAC) protocol is required to coordinate the access to the channel, while ensuring good throughput, fairness, low latency at a reasonable energy cost. With improvements in hardware, communication need not be limited to a single common channel thus necessitating multi-channel MAC protocols.

In the first part of our work, we propose a distributed channel allocation scheme to exploit this multi-channel capacity in sensor networks while taking into interference avoidance. If allocated channels do not repeat within 2 hops of a node, both primary (sender and receiver using same channel) and secondary (communication between sender and receiver interfering with another pair-wise data transfer) interference [9] can be avoided. In addition, our channel assignment algorithm has to ensure minimum energy consumption for its operation and use less number of channels, separating them as much as possible in the frequency band, thus achieving high frequency reuse.

Continuing this research further, we address the following question in this paper: “How do nodes, having been assigned non-repeating and distinct channels in their 2-hop range, establish communication with a single transceiver and a basic wake-up scheme?” This is addressed in CMAC, our proposed energy efficient MAC protocol that exploits the multi-channel capability of the sensor nodes. The main concern in wireless sensor networks is the energy loss due to collisions and re-transmissions, overhearing, control packet overhead, and idle listening. CMAC overcomes these issues by (i) supporting a default sleep mode in which sensors switch off their radio during idle times and (ii) enabling multi-channel communications with minimal hardware requirements of a low power wakeup radio (LR), in the lines of the Berkeley pico-radio [19,20], and a main half duplex transceiver (MR). The LR radio used for wakeup can only emit a short train of pulses and thus cannot be used as a second interface to send data. In the absence of complex signal processing hardware [21], we allow the LR to be capable of merely discerning the presence or absence of 6–8 pulses during the control message exchange phase.

In our scheme, the LR is always used to monitor a node's default channel while the MR is placed in the sleep mode thus conserving energy. The LR plays two roles: (1) when a node wishes to transmit, the receiver is woken up through a series of pulses and (2) channel negotiation is undertaken before the MR is switched on. Data communication is carried out by the MR and the LR can only sense the presence of a packet transmission if it is monitoring the same channel. The adoption of a multi-channel scenario enables communication to occur simultaneously and in a collision free manner. Further, we leverage the distinct channel assignment to enable waking-up of a node only when a packet is to be transmitted, thus providing improvement over probabilistic and periodic wakeup

cycles. Through our suggested handshaking mechanism, a node can issue a time duration for which it will be busy even when a reception is in progress. The control pulses are devoid of node addresses and senders are identified by channels alone, thereby further reducing the control overhead. Thus, the main contribution of this work is suggesting an approach for the problem of energy efficient MAC design in sensor networks in a multi-channel scenario without allowing performance degradation.

The rest of this paper is organized as follows. Section 2 discusses related work in the area of channel allocation and arbitration. Our DCA algorithm and an analysis of its messaging overhead are presented in Section 3. In Section 4, we describe our multi-channel MAC protocol, CMAC, with its assumptions and design considerations. In Section 5, we provide detailed simulation results that measure the performance of the DCA and CMAC respectively. Concluding remarks and directions for future research can be found in Section 5.1.

2. Related work

The problem of channel allocation is similar to the code assignment problem in Code Division Multiple Access (CDMA) networks that eliminates collisions through spread spectrum techniques and orthogonal codes. Earlier works on CDMA code allocation have approached this as a graph coloring problem, where colors can be repeated only at 3 hops or more, unlike traditional graph coloring as surveyed in [10]. We shall henceforth refer to coloring under this constraint as the NP complete 2-hop coloring problem. Centralized code assignment schemes are unsuitable when directly applied to sensor networks as it may be infeasible to establish direct communication between the base station (BS) and the individual sensor nodes. Past work has also involved presenting optimal 2-hop coloring algorithms for special topologies, such as the ring, bus, chain, tree and hexagonal grids. In the analysis of our proposed Distributed Coloring Algorithm (DCA), we consider the more general case of random deployment of the sensor nodes. Recent work, including the algorithm proposed in [11] reduces, under assumptions of ordering, to the Hidden Primary Collision Avoidance (HP-CA) suggested in [12] and modified in [13,14]. In [12], when a node chooses a color, it is propagated to its 2-hop neighbors having an ID lower than itself. The simple modification of making a node wait till it hears from all nodes having a higher ID and within its 2-hop range before announcing its own non-conflicting color provides an elegant solution. We call this HP-CAM with ‘M’ for modified. In [15], a node chooses a unique ID and broadcasts it amongst its 2-hop neighbors. In case of a conflict, the information is conveyed back to the transmitting node and new ID is chosen thus repeating the process. Though simple, this process involves large message passing as $n-1$ re-tries may be required for each node in the worst case. In HP-CAM, each node is aware of its 2-hop information and generates an ordering based on node weight. This results in reduced number of messages as there is no color conflict once an assignment has been done. We thus compare the DCA with the HP-CAM while measuring the performance parameters in this paper.

from any node in i and $\neq i$. As an example, in Fig. 1, considering cluster 2, $NS_2 = \{x, a, b, s, t, c, CH_3, CH_4\}$. The problem of color allocation is essentially coloring of a node such that there is no conflict with the already assigned colors of the nodes in the neighbor set. In particular, consider two nodes, x and y linked through a and each belonging to a different cluster. This configuration poses a difficult task in assignment of non-conflicting color and we define it as the hidden cluster problem. This necessitates the longest chain of color information propagation across cluster boundaries.

The DCA begins at a point where each CH is aware of its own weight, $w(CH)$, as well that of the other CHs that serve its neighbor set, i.e., CH_2 knows $w(CH_1)$, $w(CH_3)$ and $w(CH_4)$. A CH proceeds to color its own cluster only when color assignment by the larger weight CHs of the neighbor nodes is complete. If $w(CH_2) > w(CH_3) > w(CH_4) > w(CH_1)$, CH_2 starts coloring first, followed by CH_3 . On their completion, the color assignment is propagated and thus CH_1 , CH_4 begin coloring simultaneously ($x, CH_1 \notin NS_4$). It should be noted that there is no possible color conflict in clusters 1 and 4, as the neighbor set of one has no common node from the other cluster.

Each node in cluster i maintains a color table (CT) containing a list of its neighbor nodes belonging to a CH having an ID higher than its own CH, e.g., $CT_y = \emptyset$ while $CT_a = \{-v, y, CH_2\}$. As a node receives color information about these nodes, this list is updated and all these colors are forbidden for the node itself. In addition, the CH has a table, which we call as the cluster color table (CCT). The CCT has an entry for each node in its cluster and the colors forbidden for it are continuously updated, as in the case of CT. A CH starts coloring only when this CCT is full, i.e., all the forbidden colors for each node in its cluster are available. Each CH performs this check at the receipt of a message, in order to check whether the information contained in the CCT is sufficient to start the coloring process.

The DCA is entirely message driven, i.e. an action taken by a node is a function of the type and information contained in the incoming packet. We define three types of messages: The channel assignment message that is sent by the CH letting a node know of its assigned color, the update message aimed at 1-hop neighbors to make them aware of a forbidden color, and the information message which alleviates the hidden cluster problem while notifying the CH of a forbidden color for a node in its cluster. We now describe, through an example (Fig. 2), the actions taken by nodes for different control messages.

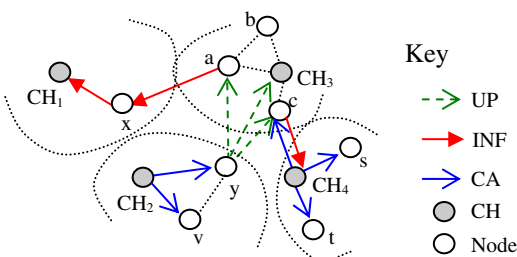


Fig. 2. Types of messages and their flow between clusters.

3.1. Message types

3.1.1. Channel assignment (CA) message

The CA message consists of an ordered pair of the type (node ID, assigned color) for each cluster member. A node first identifies whether the message was sent by its CH. It then extracts its own color and that of nodes 1 hop away from it and member of the same cluster. Thus node y identifies the color assigned to it and that of CH_2 , v . y then broadcasts an UP packet containing this information.

Similarly, if a node c is in 1-hop distance of a neighboring CH, i.e. CH_4 , it will receive a CA message not intended for its own cluster. It then extracts the colors of the 1-hop nodes appearing in its CT_c i.e. (CH_4) and 2-hop nodes (s and t). In such a case, an INF message is generated by c and the change made in its CT is included in the message (Fig. 2). The INF message has a 1-hop list and a 2-hop list as payload containing ordered pairs of the type (node, color) which are updated with (CH_4) and (s and t), respectively.

3.1.2. Update (UP) message

The UP message is targeted at nodes within transmission range of a recently colored node, notifying the former of the colors assigned to their 1- and 2-hop neighbors. When such a node, a , receives the UP message from y , CT_a is updated with the colors assigned to y , v and CH_2 . An INF message is broadcast that contains the colors and IDs of the 1-hop (y) and 2-hop (v and CH_2) neighbors of a , that were updated due to the received UP message (Fig. 3). As 1-hop clusters are formed, the INF message broadcast by any node lets its CH know that its CT is updated. CH_3 is thus made aware through a 's INF broadcast that the colors assigned to v , y , and CH_2 are now forbidden for a . Nodes belonging to the same cluster as the originator of the UP message, e.g. v , ignore it. As before, if a CH, say CH_3 , receives an UP message from non-cluster node, say y , it implies that the node y is in 2-hop range of every member of its cluster. It adds the color of node y as a forbidden color for every node in the cluster, including a , if this information is not already present. Also, CH_3 adds the colors of all 1-hop cluster members of node y , i.e. v and CH_2 , as forbidden colors for itself.

3.1.3. Information (INF) message

It is generated on two separate occasions. (1) The INF broadcast (IB) message is generated by a node as a result of an UP message is useful for its own CH and all other neighbor nodes. (2) An INF unicast message (IU), directed at the CH, is generated when a node receives an IB from a sender not belonging to the same cluster. For example, The IB message sent by a is used by CH_3 to update its CCT. An adjacent cluster node x , overhears the IB, extracts

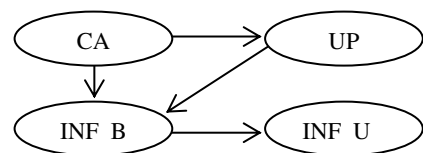


Fig. 3. Message dependencies: the receipt of one type of message generates another.

the 1-hop list (merely the color of y in this example) and updates its color table, CT_x . An IU message is then sent by x to CH_1 having a single node-color pair ($y, color$). We reason this as follows: The IB message sent by a node a contains, in the 1-hop color table, the color assigned to node y , otherwise hidden from x . For node x , y is a 2-hop neighbor and hence must appear in the 1-hop list of the incoming IB message. Consequently, only this information is included in the IU message sent by x . This two stage INF propagation allows CH_1 to assign a color to x such that it does not conflict with the color given to y , 2 hops away from x . If a node, b , receives an INF message from its own cluster member, a , it is ignored unless the receiving node is also the CH for that cluster. We note that the INF is the most commonly generated message (Fig. 1): (1) the result of hearing a CA directly from another cluster, (2) on receiving an UP message and (3) another INF broadcast message. Successive INF/UP messages received at the node from different source nodes may include color information that the node has already propagated earlier. In our implementation, we adopt the following enhancement geared to reduce the number of messages in the network: the color information forwarded by a node in an INF or UP message is not sent again through another INF or UP , respectively.

3.2. Analysis of number of messages

In this section we derive expressions for the expected number of channel assignment, update and inform broadcast messages for the DCA. While the analysis presented in the following propositions is approximate owing to the restricting assumptions of multi-hop modeling, all equations derived are verified in Section 5 and found to be in good agreement.

Consider N nodes to be distributed independently and uniformly in a large area $R \subseteq R^2$ where R^2 signifies the 2-dimensional Euclidian space. If R is large, the placement of the nodes essentially represents a Poisson process [16]. We assume:

1. All nodes have the same transmission radius r .
2. The average degree of a node is given by δ_{avg} .

Proposition 1. The total number of channel assignment messages transmitted is approximately given by

$$n(CA) = \frac{N - CH_0}{\delta_{avg}} + CH_0 \quad \text{where}$$

$$CH_0 = \frac{N}{\delta_{avg} + 1} \left(\frac{\delta_{avg}}{\delta_{avg} + 1} \right)^{\delta_{avg}}$$

Proof. For a single node A , the coverage area is πr^2 . The probability that there are m nodes within its transmission range is given by: $p_m(A) = ((\rho\pi r^2)^m / m!) \exp(-\rho\pi r^2)$.

The expected number of nodes within its 1-hop communication range, also the average degree δ_{avg} , can be calculated as

$$E[x] = \sum_{m=1}^N m p_m(A) = \sum_{m=1}^N \left(\frac{(\rho\pi r^2)^m}{m!} \exp(-\rho\pi r^2) \right)$$

The weight based clustering algorithm forms 1-hop clusters by associating nodes with their 1-hop neighbors that have a higher weight, with ties being broken by node ID. If all the neighbors of a node A link with other higher weight CH s, A may be forced to announce itself CH of the single node cluster. We now calculate the expected number of such single node clusters. We list the nodes in the 1-hop neighborhood of A , along with A itself, in the following rank table. The rank of a node in this table is decided by the highest weight of its 1-hop neighbor set. From Fig. 4, each neighbor of A is connected to another node having a weight higher than that of A , i.e. $W_1 > W(A)$, $W_2 > W(A)$ and $W_3 > W(A)$ and hence A forms a 1-hop cluster. A is at position 1 on the table with the increasing index indicating increasing rank.

Out of $\delta_{avg} + 1$ nodes, the probability of only A occurring in slot 1 of the rank table is given by

$$P[\text{position1} = A] = P[A = \text{slot1}] \cdot P[1, 2, \dots, \delta_{avg} \neq \text{slot1}]$$

$$P[\text{position1} = A] = \frac{1}{\delta_{avg} + 1} \left(1 - \frac{1}{\delta_{avg} + 1} \right)^{\delta_{avg}} = \frac{1}{\delta_{avg} + 1} \left(\frac{\delta_{avg}}{\delta_{avg} + 1} \right)^{\delta_{avg}}$$

This is also the probability of a node forming a single node cluster as all its neighbors have been drawn away by other higher weight 2-hop neighbors. The expected number of such single node clusters is

$$CH_0 = \frac{N}{\delta_{avg} + 1} \left(\frac{\delta_{avg}}{\delta_{avg} + 1} \right)^{\delta_{avg}}$$

The approximation to the average number of clusters having δ_{avg} nodes is improved by subtracting single node clusters, i.e., $(N - CH_0)/\delta_{avg}$ and it follows that the total clusters formed is approximately, $E[\text{Clusters}] = (N - CH_0)/\delta_{avg} + CH_0$. The CA message is broadcast by the CH only and each cluster has a single CH . Hence $n(CA) = E[\text{Clusters}]$ giving the required result. \square

Proposition 2. The total number of update messages transmitted is approximately given by, $n(UP) = N - n(CA)$.

Proof. Every node (other than the CH itself) is assigned a color by the CH of its cluster. This color assignment occurs through the CA message sent by the CH (or an announcement done by the node itself if it is a single node cluster). Each node of the cluster generates one UP message, on reception of the CA message. As colors are assigned only once, every non- CH node will transmit a single UP message during the coloring process. We note that the number of CH nodes = $n(CA)$. It follows that the total number of UP messages equals the total number of non- CH nodes and hence, $n(UP) = N - n(CA)$. \square

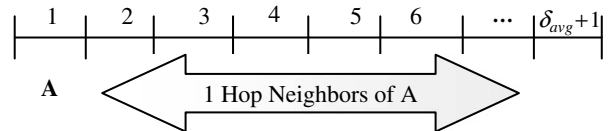


Fig. 4. Ranked table of adjacent nodes on the basis of 1 hop neighbor weights.

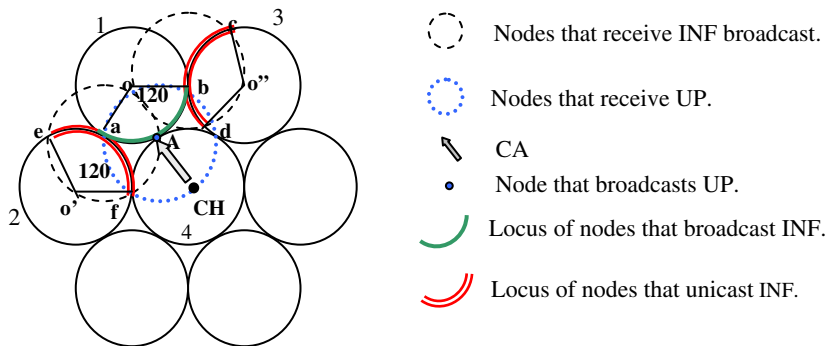


Fig. 5. Calculation of $n(INF) = n(IB) + n(IU)$.

Proposition 3. The total number of inform messages are given by: $n(INF) = \{(2\rho/3)\pi r^2 - 1\}n(UP)$ approximately.

Proof. We prove the proposition in two parts. First we show that the average number of *IB* messages is given by $(\delta_{avg}/3)n(UP)$. We begin with the simplifying assumption that clusters formed are approximately of the same size (each having a radius equal to the transmission range of a node). From geometric considerations, such a cluster can have at most six neighboring clusters. Consider one such cluster, 4, as shown in Fig. 5. The CA broadcast by the CH is received by node A, which then sends an UP message. All nodes that receive the UP message and send an IB in response, have their locus shown by sector $o-a-b$ that subtends an angle of 120° to the center o of cluster 1. Thus the average number of nodes in this sector gives the average number of IB messages generated in response to a single UP message. For $n(UP)$ messages, $n(IB) = \rho(120/360)\pi r^2 n(UP) = (\rho/3)\pi r^2 n(UP)$. The IB messages sent by the nodes in sector $o-a-b$ are received by nodes lying in sectors $o''-d-c$ and $o'-e-f$ in the adjacent clusters 2 and 3. These are the nodes that are eligible to generate IU for their CHs. The number of such nodes is given by: $2 \times \rho (120/360)\pi r^2 n(UP) = (2\rho/3)\pi r^2 n(UP)$. But the above set of nodes generating IU also includes nodes that have already heard an UP by the member nodes of cluster 4 that originated the CA message. Similarly, a single node can hear more than one IB, and we suppress the generation of IU for multiple IB messages containing the same information. Thus, we subtract the number of nodes that have already received UP and IB messages to get the actual number of nodes that transmit the IU message, given by $n(IU) = (2\rho/3)\pi r^2 n(UP) - [n(UP) + n(IB)]$. Hence, the average number of INF messages (both IU and IB) is: $n(INF) = n(IU) + n(IB) = (2\rho/3)\pi r^2 n(UP) - n(UP) = \{(2\rho/3)\pi r^2 - 1\}n(UP)$. \square

Propositions 1–3 are verified through simulations in Section 5. Having established the channel allocation algorithm, we now describe our proposed MAC protocol, CMAC, which leverages this channel assignment.

4. CMAC – MAC design considerations

CMAC has been designed to ensure maximum possible energy conservation without any compromise in fairness

or latency. Recognizing that idle listening consists of 90% of power consumption [33], CMAC puts nodes in the sleep mode whenever the MR is idle. We outline the assumptions of our protocol below.

4.1. Hardware assumptions

1. A single half-duplex transceiver (MR) capable of being dynamically tuned to any of the pre-decided set of channels is present. The MR transmits at a constant power level in the selected band but can be switched off. All data transmission/reception is handled by the MR.
2. A low-power radio (LR) is present which can only emit and receive a short train of wakeup pulses. We do not assume time synchronization amongst the nodes of the network.
3. Nodes have been allocated channels that overlap at 3 hops or more through the execution of a suitable channel assignment algorithm.

CMAC relies on three types of control messages for its operation: Request (REQ), Confirm (CON) and Wait (WAIT). All these three messages are essentially short pulse trains and are sent and received through the LR. The MR is in the default sleep state to conserve energy, delegating the task of channel monitoring and negotiation to the LR. Once this negotiation is completed, and the receiver is ready, the MR is used to transmit the actual data packet. Our protocol adopts a novel, dual mode communication architecture. During the negotiation phase, the sender tunes its LR to the channel of the receiver. Once the receiver is ready to accept the packet, CMAC follows a transmitter-oriented communication model in which the receiver tunes its MR to the channel of the sender for data packet exchange. The details of the protocol are given below and with reference made to Fig. 6 for clarity of representation.

4.2. Control message types and channel negotiation

The control messages are a sequence of $k + 1$ pulses, where 2^k is the total number of assigned channels. The LR is always tuned to a node's default channel, unless the MAC receives a data packet for transmission from the higher layer.

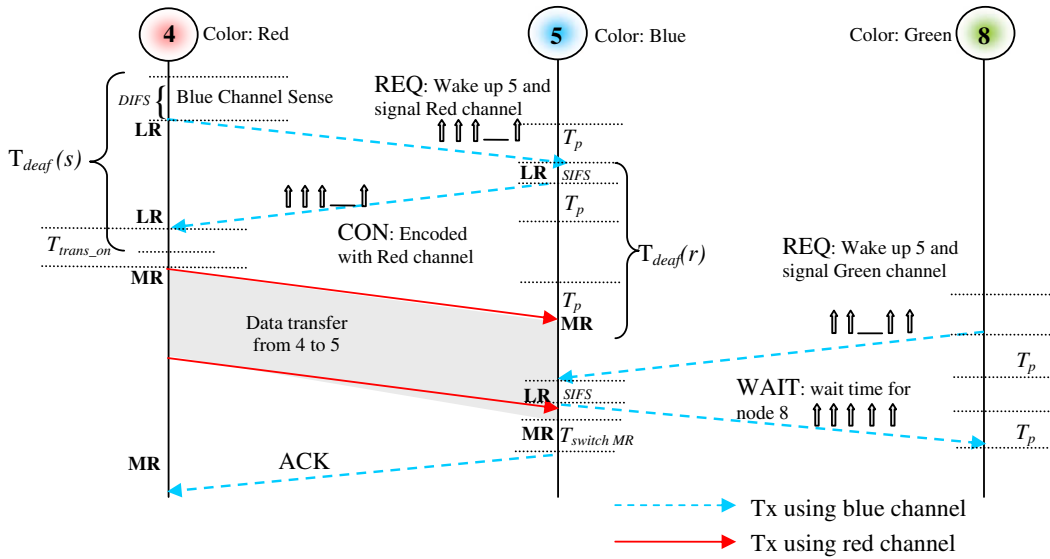


Fig. 6. CMAC message exchanges.

4.2.1. REQ

When a packet has to be transmitted, the LR of the sender is tuned to the receiver's channel, (R). It monitors R for DIFS time period to avoid collision with ongoing transmissions. In case the channel of the intended receiver is sensed busy, the sender waits till the channel is available in a manner similar to the operation of 802.11. If R is found unused at DIFS timer expiry, the sender transmits an REQ in R , after a random delay decided by the backoff interval. This REQ comprises of a set of pulses that identifies the sender's default channel (S). The identification is achieved through a unique mapping of pulse patterns and the associated channel. As an example, the color S may be represented by a pulse pattern 11001. As allocated channels are unique within 2-hop range, the sender is identified through the choice of S alone. The response by the receiver is dependent on whether it is idle or busy receiving. Consider Fig. 6, in which, nodes 4 and 5 are assigned channels *red* and *blue*, respectively.³ Node 4 first tunes its LR to *blue* and sends the REQ coded for identifying its default channel, i.e., *red*.

4.2.2. CON

If the receiver is idle, it responds to the REQ with a CON. This is sent by the receiver's LR in R after SIFS and consists of a sequence of pulses with the encoded representation of S . We reason as follows: a receiver can get multiple REQs from senders, each of which is tuned to R and awaiting a reply. The encoding of S helps to distinguish the node that has won the contention. From Fig. 6, node 5 accepts the request by node 4 and sends a CON, coded with *red*, node 4's default channel.

4.2.3. WAIT

The presence of a single pulse in the $k + 1$ st position differentiates this message from a CON. In case a node currently receiving a data packet on its MR hears a REQ on its LR, it informs the interested sender the time at which it will complete its current transaction. We accomplish this through WAIT. This message contains k pulses and when multiplied with a constant ψ , gives the approximate time for the next possible reception. In the case of multiple REQs from different senders, each of which resulted in a WAIT, it is necessary to identify the node which received the first WAIT. This node has transmission priority when the current data packet reception is completed. The provision of an additional pulse indicates whether the node receiving WAIT was the first and accordingly adopts its WAIT policy. We discuss this message further in the following section and introduce variations that enable quality of service (QoS) considerations.

Thus, we see that the LR of the sender is tuned to the channel of the receiver at the start of the DIFS interval and remains so till it gets a CON/WAIT or in their absence, suffers a timeout. After this, the LR is reset to the default channel, free to monitor incoming requests. From Fig. 6, when node 8 transmits a REQ to node 5 which is engaged in data reception, it gets a WAIT in reply. This gives the time for which node 5 will be busy.

4.3. Data transmission/reception

If the channel negotiation described earlier results in receipt of a CON by the sender, its MR is switched on and data transmission can begin. After the transmission of the CON, the receiver immediately switches its MR in the active state and tunes it to the channel described in the REQ (S). We make no assumption on the format of the data frame, save, its size be placed in the first few positions of the MAC header. Sensor data frames are assumed to be

³ (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

large [29] and an early knowledge of the size helps the receiver to calculate the anticipated time for complete reception. All outgoing WAIT messages are encoded with the remaining time for reception. This scheme also allows the use of variable size data packets and a node need not pad a frame with additional bits. After reception of the data packet, the receiver replies with an ACK in its own channel. Thus, after the transmission of the data packet, the sender switches its MR to the channel of the receiver. We do not continue with the ACK in the same channel as that used for sending the data packet owing to a possibility of collision outlined in the following section. The MR reverts back to sleep mode after the ACK is sent (by the intended receiver) and received (by the sender). In Fig. 6, after sending the CON, 5 switches on its MR, and tunes it to red, 4's default channel. It replies with an ACK in blue, 5's default color after the packet is received.

4.4. CMAC-performance parameters

In this section we analyze the working of CMAC from the perspective of energy consumption. We identify various sources of energy loss in sensor networks and explain how CMAC is designed to minimize, if not solve them completely. Table 1 gives the power dissipation parameters for the two radios which are in accordance with Berkeley MICA2 mote specifications [2] and the pico-radio project [20], respectively. In Figs. 7 and 8, numbers (1, 2, 3, 4) identify the nodes and the letters (a, b, c) stand for their allotted channels.

4.4.1. Idle time reduction

CMAC attains energy conservation chiefly by placing nodes in the sleep mode and waking them up only if communication is necessary. Thus, energy spent in idle listen-

ing is significantly reduced. This idle energy cannot be neglected and is comparable to that used by the radio during reception (about 30 mW). Recognizing that idle listening consists of 90% of power consumption [33], our emphasis on the default sleep mode results in large savings when the network is in operation for prolonged time intervals. Most periodic self-wakeup based MAC schemes [28–30] have a *listening* time in every cycle and all data exchange is undertaken only in this period. We allow an on-demand wakeup making our protocol sensitive to reporting sudden, unusual activity with lower latency. This makes CMAC preferable in time-critical applications like radiation monitoring in power plants, intruder tracking and sniper location, amongst others.

4.4.2. Deafness and overhearing

Deafness is caused when node 3 (Fig. 7a) attempts to contact node 2, which is currently engaged in communication with a third node (node 1). In our scheme, if node 2 is transmitting data, it will do so in its default channel, *b*. Recall that node 3 monitors *b* through its LR before transmitting the REQ and finding the channel occupied, it continues monitoring till the channel becomes free. Similarly if node 2 is receiving, its MR is tuned to the sender's default channel but its LR is still monitoring *b*. In this case, 3 finds *b* unused and transmits a REQ in channel *b*. This REQ will be received by 2 and replied with a WAIT (Fig. 7b). We conclude that there are only two possible deaf periods: The first is the time between sending the REQ and switching back to the default channel after the reception of the CON/WAIT (or timeout), which we define as $T_{\text{deaf}(s)}$. In this case, a sender's LR is tuned to the intended receiver's default channel and it cannot hear the control messages sent by another node for $T_{\text{deaf}(s)}$ interval (Fig. 6). We assume an inter-node separation of 40 m, a train of 8 pulses for REQ and CON. The values of DIFS and SIFS are assumed similar to SMAC [29], where DIFS = $10 \times$ slot time and SIFS = $5 \times$ slot time. Channel switching time T_{switch} is assumed to be 100 μ s. Here. Considering the channel capacity as 20 kbps and neglecting propagation time, the deaf period is then given by

$$T_{\text{deaf}(s)} = \text{DIFS} + T_{\text{REQ}} + \text{SIFS} + T_{\text{CON}} + T_{\text{switch}}$$

$$= (10 + 0.04 + 5 + 0.04 + 0.1) \text{ ms} = 15.18 \text{ ms}$$

At the receiver end, the worst case deaf period, $T_{\text{deaf}(r)}$, begins at the time of hearing the first REQ and extends till the header information in the data frame is read (Fig. 6). Recall that the header contains the size of the packet and WAIT packets can be sent only after the duration of the data reception is known thus causing this delay. We take into account the time required ($T_{\text{trans_on}}$) for the MR to be switched on after reception of the CON on the LR. For the ATmega128 microcontroller on the MICA2 mote [2], $T_{\text{trans_on}} = 180 \mu$ s and 6μ s for the TI MSP430 used in Telos [34]. In the following calculation for $T_{\text{deaf}(r)}$ we assume that 20 bytes of header suffice for conveying the wait time.

$$T_{\text{deaf}(r)} = \text{SIFS} + T_{\text{CON}} + \text{MRT}_{\text{trans_on}} + T_{\text{H}} + T_{\text{switch}}$$

$$= (5 + 0.04 + 0.180 + 20 \times 0.04 + 0.1) \text{ ms}$$

$$= 6.12 \text{ ms}$$

Table 1
Specifications of the MR and LR

	MR (mW)	LR (mW)
$P_{\text{transmission}}$	36	1
$P_{\text{reception}}$	14.4	0.450
P_{idle}	14.4	0.05
P_{sleep}	0.015	X^a

^a LR does not sleep.

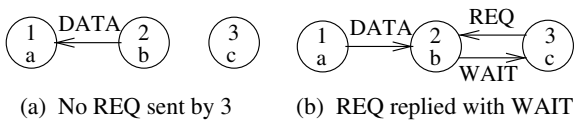


Fig. 7. Overcoming the deafness problem in CMAC.

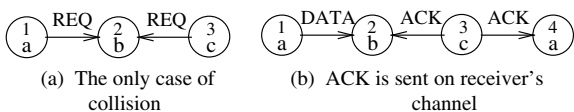


Fig. 8. Collision prevention.

Similar to 802.11, after a successful transmission, there is a *post-backoff* which prevents channel capture by a node and allows this deaf period to be spaced in time, even if there are packets queued for transmission. Through WAIT, we eliminate re-transmissions of the REQ in a node's effort to contact a particular neighbor. Overhearing occurs when a node receives a packet not destined for it. Through non-repeating channel assignment and careful channel transitions, CMAC ensures that neither control messages nor data is heard by any node other than the intended destination.

4.4.3. Collision prevention

CMAC is collision free once data transmission begins. We identify only two cases of collisions during the phase of REQ/CON/WAIT control message exchanges. The REQ sent by a node to an intended receiver can collide with a CON or WAIT sent by the latter. We assume that the pulses, being of a very short duration are not identified by the LR during the channel sensing phase. Again, the REQ is sent on the default channel of the receiver. Hence, two or more REQ pulse trains may collide when different nodes attempt to wake up the same receiver (Fig. 8a). In this case, analogous to 802.11, both nodes *backoff* and retry for the channel with an increase in their contention window. As all control message exchanges occur in the receiver's default channel, CMAC splits the collision domain to include only those nodes who are contending for the same destination node in their neighborhood. Recall that the control pulses are of a very short duration, ranging in the order of half a millisecond for a 20 kbps channel and the resulting probability of collision is significantly reduced.

4.4.4. ACK policy

In the earlier section, we have described the operation of the MR during transfer of data packets. Data transmission has been carried out in the channel of the sender and the receiver tuned its own MR to this channel. However, on completion of the transmission, the sender switches its MR to the channel of the receiver for the ACK. We now provide a simple explanation for the ACK being sent on the receiver's default channel, and not the one used for data reception. In Fig. 8b, there are two sets of simultaneous data transmissions: from nodes 4 → 3 and 1 → 2. On the reception of the data packet by node 3, an ACK in channel *a* would interfere with the ongoing communication between 1 and 2, also in the same channel. In our scheme, the ACK is sent in a node's default channel, here *c*, thus eliminating possibility of a collision.

4.4.5. WAIT policy

When a node currently engaged in data reception with a transmitter receives a REQ from yet another node, it generates a WAIT packet. In this packet, it places the binary representation of the index *k* which is a measure of the time duration left to complete the ongoing data reception. The actual time T_{left} (in millisecond units) is calculated as $T_{\text{left}} = 2^k + c$, where *c* is a constant.

In Fig. 7b, node 3 receives the WAIT from node 2 which, in turn, is receiving a data packet from node 1. As the period T_{left} indicates the minimum duration before which a

subsequent RTS should be sent to node 2, 3 should try and schedule packets to other nodes, if any, in that time. Note that the non-overlapping channel assignment makes it possible for node 3 to transmit to a node other than 2 in its neighborhood without interfering with ongoing communication at node 2. We now outline a simple scheduling policy, Q-WAIT, in which we leverage the knowledge of a node's busy period. This policy allows reduction in number of REQ/CON control message exchanges and accommodates the fact that sensor network packets may be of variable length. Here, the MAC maintains a dual queue that prevents head of the line (HOL) blocking and hence we aptly name our scheduling policy as Q-WAIT. To limit the overhead of maintaining an additional queue at the MAC layer of the sensor node, we describe the scheme for a maximum queue length of 2 for the additional queue.

4.5. Q-WAIT

Considering the example in Fig. 7b, when node 3 receives the REQ from node 2, it inserts the T_{left} period in the WAIT reply. On receiving this packet w_1 , node 2 first checks whether it was the first node that sent the REQ to node 3. As mentioned earlier, this information is included in w_1 by the provision of an additional pulse. If node 2 is indeed the first node, it then suspends all transmissions up to time T_{left} . After this time, the data packet reception by node 3 is completed and 2 now re-transmits without any further channel sensing. Though DIFS period is no longer utilized in this case, we still allow for the REQ/CON handshaking procedure to alert the receiver (here node 3) of node 2's next incoming transmission. However, we still allow for data reception by 2 in this T_{left} interval and on the arrival of an incoming REQ, node 2 may suspend its wait and begin reception with the new sender. Once a new data packet reception is initiated, the earlier packet that incurred the WAIT is now treated as any other in its queue and must undergo the process of channel sensing and the arbitration procedure all over again. Thus, we see that a node receiving WAIT is guaranteed transmission if the following conditions are satisfied:

1. The node is the first one that received the WAIT (or sent the REQ).
2. The node does not receive an incoming REQ in the T_{left} interval.

In case a node was not the first to receive the WAIT, it places the data packet in a temporary queue which we name Q_t . It then draws the next packet from the MAC queue and begins the arbitration procedure for its transmission. Now if this packet too incurs a WAIT, it is added to Q_t . Recall that we have set the maximum queue length for Q_t as 2 and the second WAIT packet fills Q_t completely. The node now cannot draw another packet for transmission from the higher layer till space is available in Q_t . There are now two conditions for the second wait packet w_2 received by the node:

1. The node received the first WAIT for the packet w_2 in which case it follows a procedure similar to the one

described earlier. Even if the packet receiving w_1 is present in Q_t for longer time duration, the second packet is guaranteed transmission and is sent out first after the duration T_{left} for w_2 .

2. If the node did not receive the first wait for both w_1 and w_2 , a node waits for the minimum of T_{left} for w_2 and the time remaining for w_1 before sensing the channel. Both packets are considered as ordinary packets and are scheduled as such after appropriate channel sensing and handshaking procedure.

As soon as one of the two packets queued Q_t is transmitted, the MAC can draw the next packet sent by the higher layer and resume its usual operation. Apart from simplicity and ease of implementation, the advantage of this scheme is that the control overhead for probing a busy node is avoided. Similarly, as is often seen in sensor aggregation schemes, a single node may be multiple routes through it, one of which may have a bottleneck node as the next hop. By utilizing the time for which this bottleneck is busy, channel utilization efficiency can be improved accompanied by lowering of end-to-end delay values.

5. Simulation results

In this section, we perform a comparative study of the DCA and the HP-CAM as well as validate the theoretical results in Section 3 through computer simulations. We created a simulation environment using the JAVA based discrete event simulation platform, SimJava [18], in which we simulated the DCA and the HP-CAM with 500–1500 nodes in steps of 100, distributed randomly in a square of side 700 units. Each experimental value noted represents an average taken over 10 samples, the topology generated being similar to both the DCA and the HP-CAM. We assume a transmission radius of 40 ensuring 99% connectivity for the above densities. As metrics for comparison for the channel assignment algorithm, we have used: *energy consumed*, *messages transmitted* and *colors required*. We neglect idle time power consumption, and associate a unit energy cost in sending and receiving a byte of information. For simulating CMAC, we use *ns-2* simulator (version 2.26) [35] have undertaken a thorough analysis of the throughput, latency, and energy consumption. Two modes of SMAC, with No-sleep (SMAC(N)) and SMAC with 10% duty cycle without adaptive listen (SMAC(10%)), for a similar chain topology as in [29] are considered for ease of comparison (Fig. 15). Our choice of SMAC for the purpose of comparison is motivated by the fact that, it is a contention based MAC protocol similar to CMAC, and incurs significant energy savings by a judicious use of the sleep-wakeup cycle. SMAC has a single radio for its data transmission similar to CMAC. We, however, account for the energy consumption of the LR in CMAC as the control overhead in the energy computations. This comparison hence serves to distinguish between the advantages of our scheme along with the overhead of operating the second, simple radio. The additional scenarios shown in Figs. 16(a) and 16(b) are described as follows: In Fig. 16(a), nodes 1, 2 and 3 are not within mutual transmission range

while they can hear each other in Fig. 16(b). Nodes are separated by 40 m such that only adjacent nodes are in communication range. Distinct channels are assigned to nodes, satisfying the constraints of 2-hop coloring described earlier. For the chain topology, the source node (Node 1) generates 20 data messages each 100 bytes long and are sent to the destination node (Node 10). In Figs. 16(a) and 16(b), nodes 1 and 2 generate a similar traffic towards node 0. Node 3 has two flows originating from it and ending at nodes 0 and 4, respectively. Table 1 summarizes the energy consumption of the two radios.

5.1. Results for DCA

Fig. 9 compares the total number of messages transmitted for a given number of nodes. We find that the DCA performs significantly better than the HP-CAM with the difference increasing as density increases. For high densities, (1500 nodes), the DCA transmits about 40% lesser messages.

Fig. 10 compares the average energy spent during the coloring process. We observe that there is a constant improvement in energy savings for all densities considered with allowed variation decided by 95% confidence intervals. We observe that despite the significant reduction in number of messages with increasing nodes, the difference between HP-CAM and DCA is approximately constant. This is primarily because the data field in each message transmitted in the HP-CAM is constant while in the DCA, the amount of information contained is a function of node degree and hence though fewer messages are sent, size of the data field is also larger. The IEEE 802.15.4 standard for sensor networks [4] defines 6 bytes in its physical layer (PHY) and 13 bytes in its MAC layer header. The energy saving incurred is significant in the DCA if the additional overhead of 19 bytes is considered for each data packet sent as seen in Fig. 11.

Fig. 12 compares the analytical and simulation results for the number of CA, $n(CA)$, and UP messages, $n(UP)$ gener-

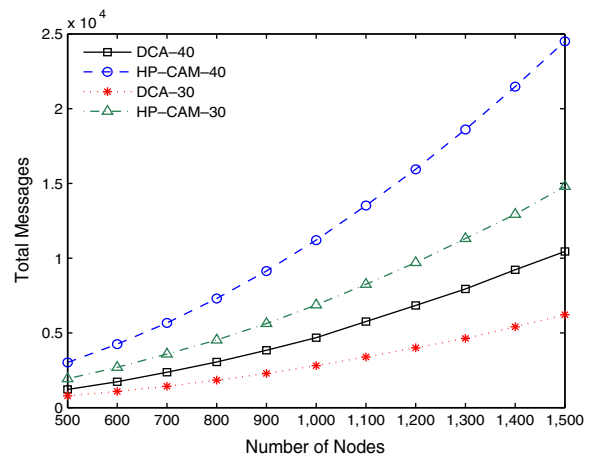


Fig. 9. Total number of messages generated by HP-CAM and DCA for $R = 30$ and $R = 40$.

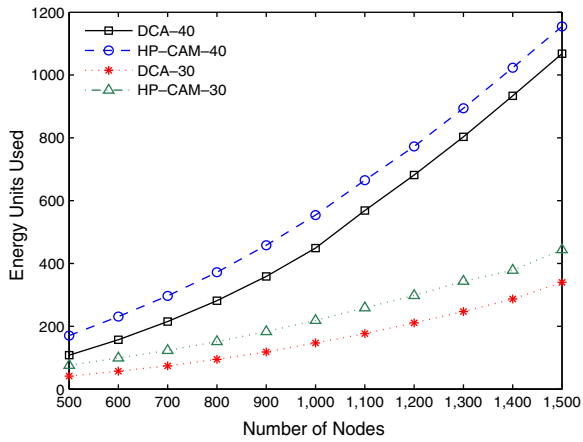


Fig. 10. Energy spent in HP-CAM and DCA for $R = 30$ and $R = 40$ for data communication only.

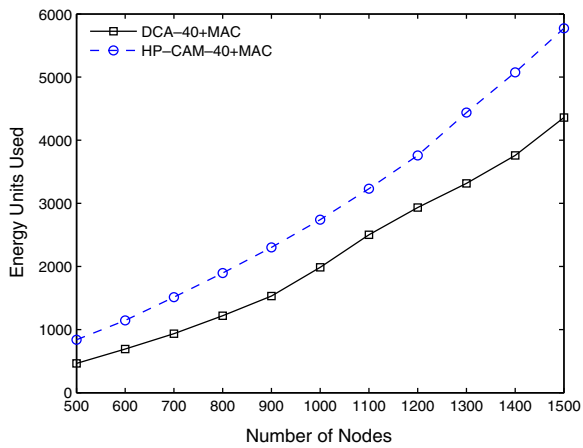


Fig. 11. Energy spent in HP-CAM and DCA for $R = 40$ with 19 bytes of PHY + MAC header.

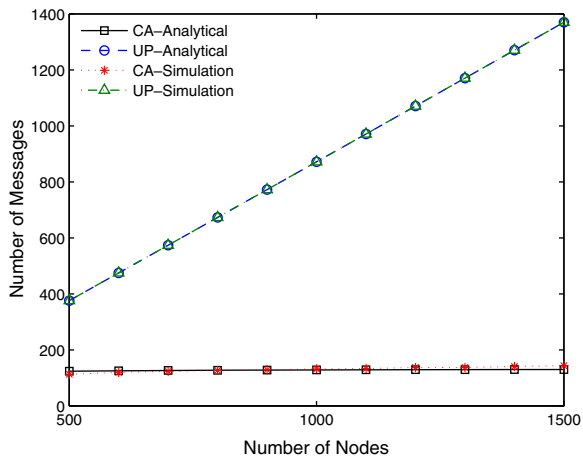


Fig. 12. Comparison of analytical and simulation results for CA and UP messages.

ated by the algorithm for a given node density. It is seen that $n(CA)$ increases at a much lower rate than $n(UP)$. This occurs primarily because with increase in node density, the average degree of a node increases. This causes each cluster to comprise of progressively larger number of nodes, causing more number of update messages to be generated for each channel assignment. At this point we note the dependence of the DCA on the clustering algorithm as $n(CA)$ is dependent on the number of clusters formed and hence does not significantly change, with additional nodes preferring to join existing clusters than form separate ones by themselves. Fig. 13 gives the plots for the INF broadcast, $n(IB)$ and INF unicast, $n(IU)$ messages, which show the expected increase with node density.

Fig. 14 indicates the number of colors used by HP-CAM and the DCA to legally 2-hop color the network graph. We find the DCA uses approximately the same number or marginally fewer colors as the HP-CAM. This graph is important while choosing the sensor for the application. In our simulation, the IEEE 802.15.4 Zigbee (16 channels) is pre-

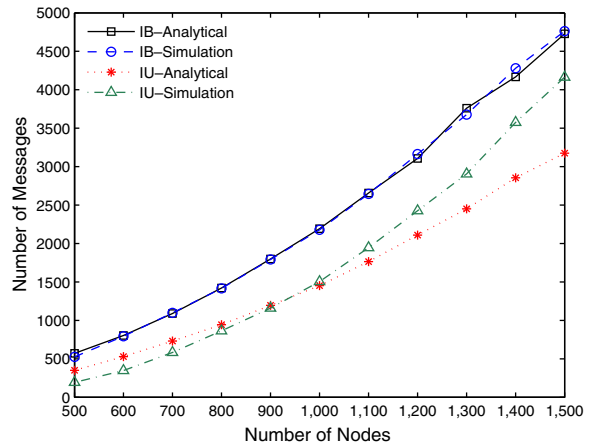


Fig. 13. Comparison of analytical and simulation results for IB and IU messages.

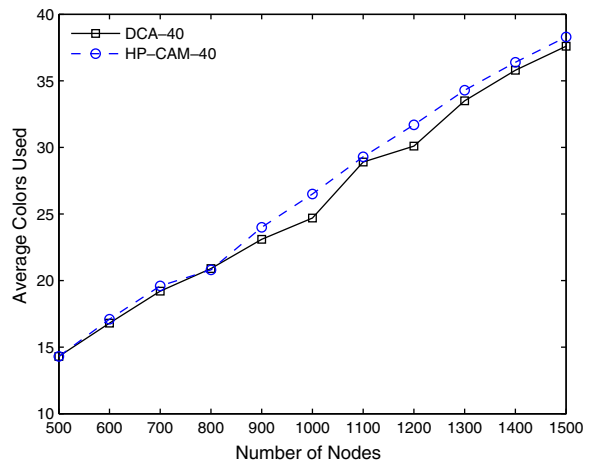


Fig. 14. Comparison of average colors used by DCA and HP-CAM.

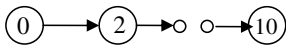


Fig. 15. Chain topology used for simulation study.

ferred over the MICA2 mote while deploying 500 nodes. Similarly, the average colors predicted at a deployment of 1000 nodes ≈ 27 , necessitating the choice of the WINS mote having 40 channels.

5.2. Results for CMAC (chain topology)

A. Energy consumption. Fig. 17 shows the comparative aggregate energy consumption of nodes. Energy consumption of a node is obtained by calculating the total transmit, receive, idle and sleep time for both the radios in CMAC and the single radio in S-MAC. Results show that our protocol out-performs both the modes of SMAC with nearly 200% reduction in the total energy consumed. As can be observed from the figure, CMAC has very low and near-constant energy consumption. Recall from Section 4, that, in CMAC, the MR wakes up only when it has to transmit or receive data and spends rest of the time in sleep mode. To avoid synchronization/deafness problems, the LR is switched on all the time. As the LR has negligible power requirements, it barely contributes to the total energy consumption and hence the energy consumed by the control

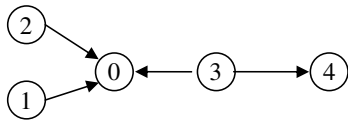


Fig. 16(a). Only the usage of WAIT is seen.

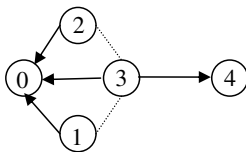


Fig. 16(b). WAIT usage as well as multi-channel advantage.

messages is negligible. Only the MR's transmit/receive periods contribute significantly to the total energy consumed as the idle period is very minimal. Moreover, as nodes in the neighborhood are assigned different channels, overhearing is also avoided. As the total number of DATA messages to be transmitted is constant, the energy consumption is nearly constant. However, in S-MAC(N), the radio has to be active all the times even though it has no data to transmit, thus expending a lot of energy in idle listening and overhearing.

As the inter-arrival period increases, the nodes spend more time in idle listening, adding to the energy cost. This is reflected in the linear increase in the total energy consumed. This idle listening is avoided by the SMAC(10%) by keeping the radio active only during listen times and sync periods. Although the SMAC(10%) has fair savings when compared to the SMAC(N), it has an adverse effect on throughput, as we observe next.

B. Aggregate throughput. We measure the end-to-end throughput by varying the inter-arrival times of the messages as shown in Fig. 18. Both CMAC and SMAC(N), transmit data whenever the next hop is ready to receive, resulting in high and almost comparable throughput. SMAC(10%) is allowed to transmit/receive for only one-tenth of the frame time and must wait for the next cycle to forward a received packet. This results in a severe reduction in throughput. However, as the inter-arrival period increases, messages arrive with greater separation in time and thus decreasing the throughput in both CMAC and SMAC(N). SMAC(10%) exhibits a decline in performance as the packet arrival delay adds to that induced by periodic sleep.

C. Per hop latency. In this section we discuss and analyze the per-hop latency at low and high traffic loads (Figs. 19(a) and 19(b), respectively). An inter-arrival time of 10 s represents low-load conditions while high load means all messages are generated and ready for transmission at the source node at the same time. All the schemes display the expected increase in per-hop latency with SMAC(N) and CMAC having comparable values. SMAC(10%) performs poorly, ranging from 150% to 50% higher at the last hop for low and high loads, respectively.

Due to periodic sleeping, a node can only transmit a received packet to next-hop node in the listen period in the

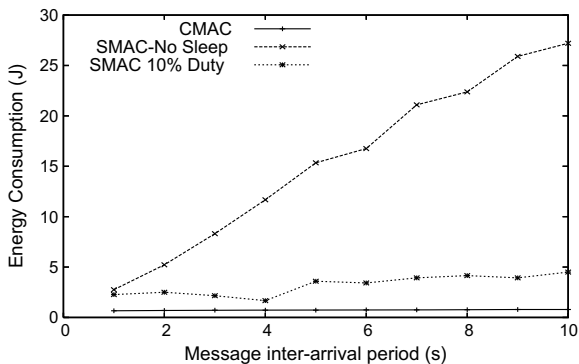


Fig. 17. Aggregate energy consumption.

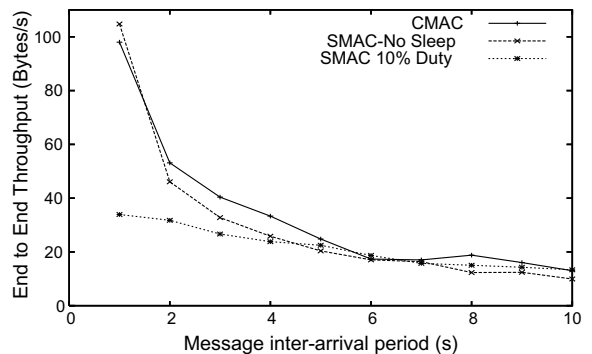


Fig. 18. Throughput.

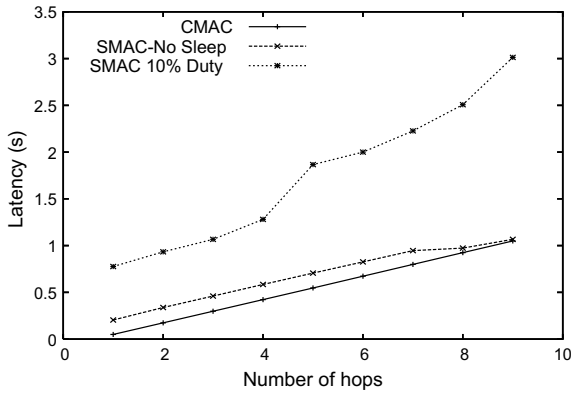


Fig. 19(a). Low load.

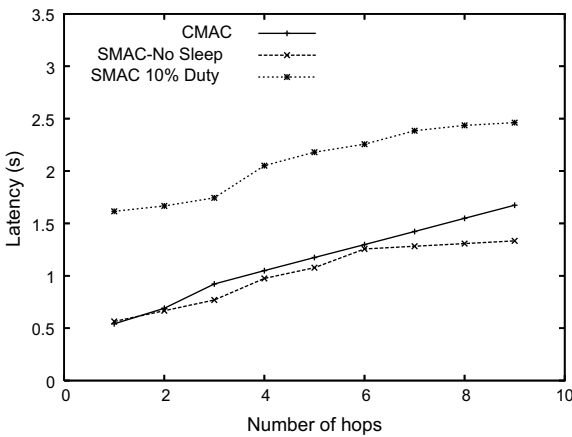


Fig. 19(b). High load.

successive frame, thus adding additional delay. However in our protocol, nodes forward the packet immediately when the next-hop node is ready for reception. At high load, the latency increases for all the protocols due to higher queuing delay. We can also observe slightly higher latency of our protocol at high load when compared with no-sleep node of SMAC. This delay is attributed to the channel negotiation period and a small deafness period when a node communicates with the next-hop node. While the chain topology evaluated earlier gives us an insight to the energy savings and improvement in latency, the advantage of multi-channel communication is not immediately seen. The WAIT policy incorporated in CMAC does not come into play in a chain topology as a single receiver can receive a data packet from no more than one sender.

5.3. Results for CMAC (topologies in Fig. 16)

We next consider the topologies presented in Figs. 16(a) and 16(b) to analyze the MAC layer delay for above cases measured at the node originating the flow. We define the flows 1, 2, 3, as those originating from nodes 1, 2 and 3, respectively, towards 0. Flow 4 is directed from 3 to 4 in both the considered topologies. From Figs. 20(a)

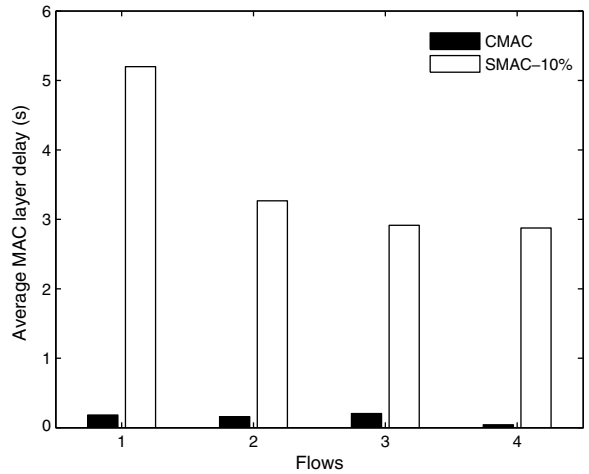


Fig. 20(a). Average MAC layer packet delay for topology shown in Fig. 16(a).

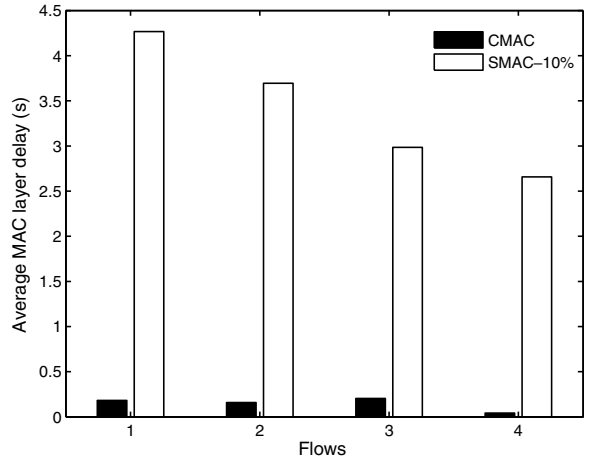


Fig. 20(b). Average MAC layer packet delay for topology shown in Fig. 16(b).

and 20(b), we find that the average delay seen at the MAC layer is much lower in CMAC than in SMAC with 10% listen cycle. In addition, this low value remains constant irrespective of the considered topology. As the channels allocated do not overlap in 2-hop range, CMAC is oblivious to the effects of proximity of hop nodes to each other, if they are separated by a distance of 2 hops or less. In both the topologies, node 3 experiences a WAIT whenever node 0 is busy receiving from nodes 1 or 2. It can then wait for the end of the specified duration or immediately schedule a packet to node 4. Note that for CMAC, flow 3 experiences higher delay than any of the remaining nodes. This is because there are multiple cases of the packet from node 3 being assigned a second WAIT by node 0. This results in node 3 scheduling the packet to node 4 and saving the bottleneck node 0 from further contention. In SMAC, interestingly,

the delay performance shows improvement when nodes come closer to each other. In the topology shown in Fig. 16(b), when nodes 1 and 2 are within range of node 3, the SYNC packet from node 3 are received by both and thus the listening cycles of nodes 0, 1, 2, 3 are synchronized. Thus after each reception, there is no longer a delay of an entire frame, unlike the topology shown in Fig. 16(a), and this effect is more pronounced when compared with the chain topology. However, the average values of delay are significantly higher than CMAC, thus stressing on the benefits of a multi-channel scheme.

6. Conclusion

In this paper, we presented a distributed channel assignment algorithm, DCA, which efficiently allocates channels to randomly deployed sensor nodes. Our proposed solution can be used as a base algorithm for several different schemes like allocating channels, assigning non-overlapping TDMA slots and nodes addresses to sensor networks. Our de-synchronized multi-channel MAC protocol for sensor networks, CMAC, addresses the critical issue of energy conservation without trade-offs in latency and throughput. Through a wakeup radio in addition to the main transceiver, multi-channel communication is accomplished resulting in a virtually collision-free protocol. CMAC enables maximum possible sleep-time, prevents overhearing and has minimal control overhead. In addition to the above, our protocol supports high data rates making it application independent. CMAC however needs an adequate number of available channels to satisfy the 2-hop coloring constraint. Extensive simulations as well as analysis reveals that DCA achieves a significant reduction in latency, reduced message complexity and considerable energy savings in the coloring process when compared with the existing schemes. We did a comprehensive study of CMAC and simulation results reveal that our protocol achieves a nearly 200% improvement in energy savings and 50–150% decrease in latency when compared to SMAC. The current implementation of the DCA uses single hop clusters and as part of future work, we plan to extend it for larger, k -hop clusters with node mobility and failure models. For CMAC, the addition of new nodes and discovering their assigned channel in the current MAC framework is another direction of research. By choosing the WAIT periods as a function of traffic class, service differentiation and a Quality of Service metrics can be incorporated. An analytical model of a multi-channel MAC scheme is another new research direction and work is underway to define such a framework for CMAC.

References

- [1] D. Agrawal, Q.-A. Zeng, Introduction to Wireless and Mobile Systems, Brooks/Cole Publishing, 2003, ISBN 0534-40851-6. 436pp.
- [2] Website: XBOW MICA2 Mote Specifications, <<http://www.xbow.com/>>.
- [3] J.R. Agre, L.P. Clare, G.J. Pottie, N.P. Romanov, Development platform for self-organizing wireless sensor networks, In: Proceedings of SPIE-Aerosense, Unattended Ground Sensor Technologies and Applications, vol. 3713, March 1999, pp. 257–268.
- [4] Ed Callaway, P. Gorday, L. Hester, J.A. Gutierrez, M. Neave, B. Heile, V. Bahl, Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks, IEEE Communication Magazine 40 (8) (2002) 70–77.
- [5] A. Dunkels, L.M. Feeney, B. Grönvall, T. Voigt, An integrated approach to developing sensor network solutions, in: Proceedings of the Second International Workshop on Sensor and Actor Network Protocols and Applications, Boston, Massachusetts, USA, August 2004 (Invited paper).
- [6] S. Cheekiralla, Wireless sensor network based tunnel monitoring, in: Proceedings of the Workshop on Real-World Wireless Sensor Networks, Stockholm, Sweden, June 2005.
- [7] R. Szewczyk, J. Polastre, A. Mainwaring, D. Culler, Lessons from a sensor network expedition, in: Proceedings of the First European Workshop on Wireless Sensor Networks (EWSN), Berlin, Germany, January 2004.
- [8] T. Liu, C.M. Sadler, P. Zhang, M. Martonosi, Implementing software on resource-constrained mobile sensors: experiences with Impala and ZebraNet, in: Proceedings of the Second International Conference on Mobile Systems, Applications, and Services (MobiSys), Boston, MA, USA, June 2004.
- [9] S. Ramanathan, Errol L. Lloyd, Scheduling algorithms for multihop radio networks, IEEE/ACM Transactions on networking 1 (2) (1993) 166–172.
- [10] I. Gupta, Minimal CDMA recoding strategies in power controlled ad-hoc wireless networks, Technical Report, Department of Computer Science, Cornell University, 2001.
- [11] C. Schurgers, G. Kulkarni, M.B. Srivastava, Distributed on-demand address assignment in wireless sensor networks, IEEE Transactions on Parallel and Distributed Systems 13 (10) (2002) 1056–1065.
- [12] A.A. Bertossi, M.A. Bonucceli, Code assignment for hidden terminal interference avoidance in multihop packet radio networks, IEEE/ACM Transactions on Networking 3 (4) (1995) 441–449.
- [13] R. Battiti, A.A. Bertossi, M.A. Bonucceli, Assigning code in wireless networks: bounds and scaling properties, Journal of Wireless Networks 5 (3) (1999) 195–209.
- [14] L. Hu, Distributed code assignments for CDMA packet radio networks, IEEE/ACM Transactions on Networking 1 (6) (1993) 668–677.
- [15] T. Herman, S. Tixeuil, A distributed TDMA slot assignment algorithm for wireless sensor networks, in: Proceedings of the First Workshop on Algorithmic Aspects of Wireless Sensor Networks (AlgoSensors) Finland, Lecture Notes in Computer Science, Springer-Verlag, July 2004, pp. 45–58.
- [16] Peter Hall, The Theory of Coverage Processes, John Wiley and Sons, 1988.
- [17] S. Basagni, Distributed clustering for ad hoc networks, in: Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks, June 1999, pp. 310–315.
- [18] Fred Howell, Ross McNab, Simjava: a discrete event simulation package for Java with applications in computer systems modeling, in: Proceedings of the First International Conference on Web-based Modelling and Simulation, Society for Computer Simulation, January 1998.
- [19] J. Rabaey, J. Ammer, J. da Silva, D. Patel, S. Roundy, Picoradio supports ad-hoc ultra-low power wireless networking, IEEE Computer Magazine 33 (7) (2000) 42–48.
- [20] Y.H. Chee, A.M. Niknejad, J. Rabaey, An ultra-low power injection locked transmitter for wireless sensor networks, in: Proceedings of IEEE Custom Integrated Circuits Conference, 2005, pp. 797–800.
- [21] C. Guo, L.C. Zhong, J.M. Rabaey, Low power distributed MAC for ad hoc sensor radio networks, in: Proceedings of Global Telecommunications Conference (GLOBECOM), IEEE, vol. 5, November 2001.
- [22] M.J. Miller, N.H. Vaidya, A MAC protocol to reduce sensor network energy consumption using a wakeup radio, IEEE Transactions on Mobile Computing 4 (3) (2005) 228–242.
- [23] C. Schurgers, V. Tsiatsis, S. Ganeriwal, M. Srivastava, Optimizing sensor networks in the energy-latency-density design space, IEEE Transactions on Mobile Computing 1 (1) (2002) 7080.
- [24] C. Schurgers, V. Tsiatsis, S. Ganeriwal, M. Srivastava, Topology management for sensor networks: exploiting latency and density, in: Proceedings of ACM MobiHoc 2002, June 2002.
- [25] Z. Chen, A. Khokar, Self organization and energy efficient TDMA MAC protocol by wake up for wireless sensor networks, in Proceedings of the IEEE International Conference on Sensors and Adhoc Communication and Networks (Secan), October 2004.

- [26] K.A. Arisha, M.A. Youssef, M.F. Younis, Energy-aware TDMA based MAC for sensor networks, in: Proceedings of IEEE IMPACCT 2002, New York City, May 2002.
- [27] C. Busch, M. Ismail, F. Sinrikaya, B. Yener, Contention-free MAC protocols for wireless sensor networks, in: Proceedings of the 18th Annual Conference on Distributed Computing (DISC 2004), LNCS 3704, October 2004, pp. 245–259.
- [28] Kyle Jamieson, Hari Balakrishnan, Y.C. Tay, Sift: A MAC protocol for event-driven wireless sensor networks, MIT Laboratory for Computer Science, Technical Report MIT-LCS-TR-894, 2003.
- [29] W. Ye, J. Heidemann, D. Estrin, Medium access control with coordinated, adaptive sleeping for wireless sensor networks, *IEEE/ACM Transactions on Networking* 12 (3) (2004) 493–506.
- [30] T.V. Dam, K. Langendoen, An adaptive energy-efficient mac protocol for wireless sensor networks, in: Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys), November 2003.
- [31] J. Ai, J. Kong, D. Turgut, An adaptive coordinated medium access control for wireless sensor networks, in: Proceedings of the Ninth IEEE Symposium on Computers and communications (ISCC 2004), Alexandria, Egypt, June, 2004.
- [32] J.S. Pathmasutharam, A. Das, A.K. Gupta, Primary channel assignment based MAC (PCAM) – a multi-channel MAC protocol for multi-hop wireless networks, in: Proceedings of the IEEE WCNC, 2004/03.
- [33] J. Reason, J. Rabaey, A study of energy consumption and reliability in a multi-hop sensor network, in: ACM SIGMOBILE Mobile Computing and Communications Review, vol. 8, January 2004, pp. 84–97.
- [34] Datasheet available at the website: <<http://www.moteiv.com/products/docs/telos-revb-datasheet.pdf>>.
- [35] UCB/LBNL/VINT Network Simulator (NS-2), <<http://www.isi.edu/nsnam/ns/index.html>>.



Kaushik R. Chowdhury received his B.E. degree in Electronics Engineering with distinction from VJTI, Mumbai University, India, in 2003. He received his M.S. degree in computer science from the University of Cincinnati, OH, in 2006, graduating with the best thesis award. He is currently a Research Assistant in the Broadband Wireless Networking Laboratory and pursuing his Ph.D. degree at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. His current research interests include cognitive radio architectures,

and resource allocation in wireless multimedia sensor networks. He is a student member of the IEEE.



Nagesh Nandiraju is currently working as a Senior Systems Engineer at Motorola Inc., Horsham, PA. He graduated with a Ph.D. in Computer Science and Engineering from the University of Cincinnati, OH, USA. He has received his B.E. in Computer Science and Engineering with First Class and Distinction from *University of Pune*, India in 2001. Before joining Motorola, he was a research assistant in the *Center for Distributed and Mobile Computing Lab* at the *University of Cincinnati*. His research interests are in the broad area of wireless networking, specifically focusing

on performance evaluation and design of efficient MAC and Network layer protocols for multi-hop wireless mesh, ad hoc and sensor networks, provisioning QoS in integrated networks. He was with the Corporate R&D group of *Qualcomm Inc.* where he researched and developed novel algorithms for Next Generation Technologies such as High Altitude Platforms. He was also with *Kiyon Inc.* where he was involved in the design of proprietary routing protocol for wireless mesh routers. Earlier he was a faculty member in *National Institute of Technology, Silchar, India* (2002–2003), where he taught Computer Networks and Compiler Design.



Pritam Chanda obtained his bachelor's degree in Computer Science from Jadavpur University, Kolkata India in 2000 and master's degree in Computer Engineering from the University of Cincinnati, Ohio in 2005. He is currently pursuing PhD in the department of Computer Science and Engineering, SUNY Buffalo



Dharma P. Agrawal (M'74, F'87) is the Ohio Board of Regents Distinguished Professor of Computer Science and the founding director for the Center for Distributed and Mobile Computing in the Department of ECECS, University of Cincinnati, OH. He was a Visiting Professor of ECE at the Carnegie Mellon University, on sabbatical leave during the Autumn 2006 and Winter 2007 Quarters. He has been a faculty member at the N.C. State University, Raleigh, NC (1982–1998) and the Wayne State University, Detroit (1977–1982). His recent research interests include His

recent research interests include resource allocation and security in mesh networks, efficient query processing and security in sensor networks, and heterogeneous wireless networks. His co-authored introductory text book on *Wireless and Mobile Computing* has been widely accepted throughout the world and a second edition was published in 2006. The book has been reprinted both in China and India and translated in to Korean and Chinese languages. His second co-authored book on *Ad hoc and Sensor Networks* published in spring of 2006 has been named as the best seller by the publisher. He has given tutorials and extensive training courses in various conferences in USA, and numerous institutions in Taiwan, Korea, Jordan, Malaysia, and India in the areas of Ad hoc and Sensor Networks and Mesh Networks. He is an editor for the *Journal of Parallel and Distributed Systems*, *International Journal on Distributed Sensor Networks*, *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, and *International Journal of Ad Hoc & Sensor Wireless Networks*. He has served as an editor of the *IEEE Computer magazine*, the *IEEE Transactions on Computers*, and the *International Journal of High Speed Computing*. He has been the Program Chair and General Chair for many international conferences and meetings. He has received numerous certificates and awards from the IEEE Computer Society. He was awarded a *Third Millennium Medal*, by the IEEE for his outstanding contributions. He has also delivered keynote speech for five international conferences. He also has five patents in wireless networking area. He has also been named as an *ISI Highly Cited Researcher* in Computer Science. He is a Fellow of the IEEE, the ACM, the AAAS, and the WIF.



Qing-An Zeng is an Assistant Professor in the Department of Computer Science at University of Cincinnati, USA and the Director of Wireless and Mobile Networking Laboratory (WMN Lab) at University of Cincinnati. Dr. Zeng holds both M.S. and Ph.D. degrees in EE from Shizuoka University in Japan. In 1997, he joined NEC Corporation, Japan, where he was engaged in the R&D of 3G systems. In 1999, he joined the Department of ECECS at the University of Cincinnati. He is a co-author of the book titled *Introduction to Wireless and Mobile Systems* (2nd edition) published by Thomson

in 2005. His research interests include Wireless and Mobile Networks, Handoff and Resource assignment, Mobility Management, Heterogeneous Networks, Mesh Networks, Wireless Ad Hoc Networks, Wireless Sensor Networks, Wireless Internet, QoS Issues in Communication Networks, Security in Wireless Networks, UWB (Ultra-wideband), NoC (Network on Chip), Modeling and Performance Analysis, Queuing Theory. He is a senior member of IEEE.