SAMAC: A Cross-Layer Communication Protocol for Sensor Networks with Sectored Antennas

Emad Felemban, *Member*, *IEEE*, Serdar Vural, *Member*, *IEEE*, Robert Murawski, *Student Member*, *IEEE*, Eylem Ekici, *Member*, *IEEE*, Kangwoo Lee, *Student Member*, *IEEE*, Youngbag Moon, *Member*, *IEEE*, and Sangjoon Park, *Member*, *IEEE*

Abstract—Wireless sensor networks have been used to gather data and information in many diverse application settings. The capacity of such networks remains a fundamental obstacle toward the adaptation of sensor network systems for advanced applications that require higher data rates and throughput. In this paper, we explore potential benefits of integrating directional antennas into wireless sensor networks. While the usage of directional antennas has been investigated in the past for ad hoc networks, their usage in sensor networks bring both opportunities as well as challenges. In this paper, Sectored-Antenna Medium Access Control (SAMAC), an integrated cross-layer protocol that provides the communication mechanisms for sensor network to fully utilize sectored antennas, is introduced. Simulation studies show that SAMAC delivers high energy efficiency and predictable delay performance with graceful degradation in performance with increased load.

Index Terms—Sensor networks, sectored antennas, scheduling, cross-layer protocols, directional antennas, TDMA, CSMA/CA.

1 Introduction

Wireless Sensor Networks (WSN) [1] have been used to gather information for many diverse applications like habitat monitoring, disaster response, and target tracking. Sensor nodes transmit their sensory data to the sink node(s) as multiple data flows. However, coexistence of multiple flows leads to packet drops, loss of sensory data, and poor network performance due to the interference caused by the traditional omnidirectional antennas used in most WSN systems. The usage of control packets like RTS/CTS may abate the interference problem, however, they inhibit a large neighborhood in all directions from transmitting simultaneously, which again degrades the performance and limits the wireless network capacity.

Directional antennas, as opposed to omnidirectional antennas, concentrate the transmission power toward a certain direction with very limited beam width around this direction [2]. As a result, directional antennas provide many potential advantages to wireless networks. Perhaps, the most prominent advantage is the reduction of interference between neighboring nodes, which increases the spatial reuse of the network and improve the network performance. In Section 2, we will show through theoretical analysis that directional antennas help to reduce the delay and improve the channel throughput of the network. In addition to increasing the spatial reuse, directional antennas reduce energy consumption since less transmission power is needed to cover the same transmission range covered by omnidirectional antennas.

With advanced integrated design, the usage of directional antennas in wireless sensor nodes is feasible [3] and highly desirable due to the potential energy savings and performance gains. To exploit the potential benefits of directional antennas, protocols designed specifically for physical, MAC, and network layers are required to control when and where to point the directional beam for a successful communication. Moreover, essential communication mechanisms like neighbor discovery and medium access need to be redesigned to adapt to directional antenna systems.

The type of directional antennas used in sensor nodes should meet the requirements of WSNs, such as low energy consumption and hardware cost. Two main techniques are used in directional antenna systems that allow the directional transmission/reception. *Steerable* beam technology which provides an electronic control of beam pattern selection. They provide high resolution of transmission azimuth angles at high hardware cost and increased computation complexity due to signal processing which render them unsuitable for sensor networks. *Switched* beam technology, on the other hand, uses fixed antenna patterns for transmission and reception from a specific direction and thus does not require expensive signal processing operations. One realization of switched beam antennas is *Sectored* antennas where different

- *E. Felemban is with the Department of Computer Engineering, Umm al-Qura University (UQU), 1262 Alsemsem Street, AlEskan, Makkah, PO Box 1330, Saudi Arabia. E-mail: eafelemban@uqu.edu.sa.*
- *S. Vural is with the Centre for Communication Systems Research, University of Surrey, Guildford, Surrey GU2 7XH, UK. E-mail: s.vural@surrey.ac.uk.*
- *R. Murawski and E. Ekici are with the Department of Electrical and Computer Engineering, The Ohio State University, 205 Dreese Lab, 2015 Neil Ave., Columbus, OH 43210. E-mail: {murawskr, ekici}@ece.osu.edu.*
- *K. Lee, Y. Moon, and S. Park are with the Electronics and Telecommunications Research Institute (ETRI), 138 Gajeongno, Yuseong-gu, Daejeon 305-700, Korea. E-mail: {stn, moonyb, sangjoon}@etri.re.kr.*

Manuscript received 2 Mar. 2008; revised 9 June 2009; accepted 6 Jan. 2010; published online 17 Mar. 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2008-03-0069. Digital Object Identifier no. 10.1109/TMC.2010.61.

antenna elements can be selected using a simple switch as shown in Fig. 3. This inexpensive and simple architecture can easily be integrated with sensor nodes.

In this paper, the ***Sectored-Antenna-Based Medium Access Control*** protocol, called *SAMAC*, specifically designed for WSNs with sectored-antenna system is introduced. SAMAC provides the basic communication functionality to capture potential advantages of sectored antennas. Specifically, SAMAC is designed to achieve the following three objectives:

1. Enhance throughput and E2E delay characteristics of the sensor network by exploiting the spatial reuse capability of directional antennas.
2. Obtain a high packet delivery ratio by minimizing channel contention and packet collisions in the shared wireless communication medium.
3. Extend sensor battery lifetime by minimizing transmission and reception power and idle listening.

To our knowledge, this is the first complete protocol that integrates a form of directional antennas to WSNs.

The rest of the paper is organized as follows: Section 2 presents a theoretical analysis to motivate the usage of sectored antennas in WSNs. Section 3 discusses the previous related work on directional antenna protocols. In Section 4, important preliminary information to understand the basics of SAMAC is provided. Section 5 provides an overview on SAMAC protocol. Section 6 describes the directional neighbor discovery mechanism proposed with SAMAC to gather the neighborhood information in the sink node. Time schedule computation and distribution are presented in Sections 7 and 8, respectively. The details of data delivery mechanisms are explained in Section 9. Section 10 explains the failure recovery mechanism of SAMAC protocol. Simulation studies and performance analysis are presented in Section 11. Finally, concluding remarks are presented in Section 12.

2 Motivation of Using Sectored Antennas

In most deployed and planned battery-powered sensor networks, applications are envisioned to deliver low data rates with high timeliness tolerance. However, some critical applications require high data rate with stringent delay requirements. Consider an intrusion detection system that consists of sensor nodes equipped with (low-resolution) cameras among other sensors. While sensor nodes are battery operated, they are also equipped with solar cells to replenish energy supplies. Sensor nodes periodically take snapshots of their observation areas. Nodes compare captured images locally and send them to the cluster heads for further processing only when there are significant changes in the scene. Hence, the communication events are triggered mainly by intrusion detection events (in addition to synchronization and other management information exchange).

In most cases, the load on the sensor network is low since the intrusion events are relatively rare. The network conserves energy by adaptive sleeping schedules when the load is low. However, when an intrusion event (or a significant change in the obtained image suggesting a possible intrusion) occurs, then the information needs to be delivered to the cluster head in a timely manner. Furthermore, the load offered to the network increases suddenly due to correlated triggering of nearby sensors by the same event. As such, although the overall load carried in the network is very low on the average, over short periods of time, the load carried in the network is several orders of magnitude larger than the average.

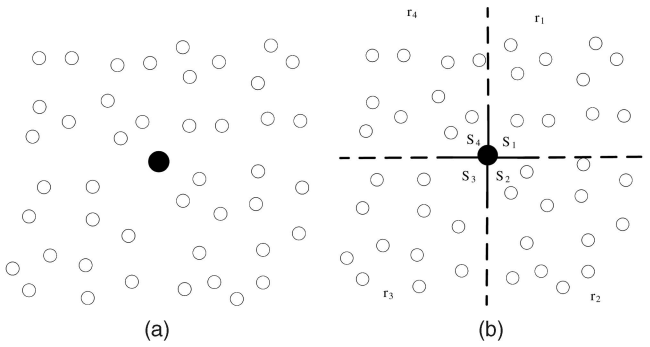


Fig. 1. Single-hop network analysis. (a) Single-hop network using omni antenna. (b) Virtual regions using directional antenna.

This section motivates through simple theoretical analysis the usage of sectored antennas in such critical applications. Let us consider a single-hop network where N nodes send data packets of size L bits to the base station using omnidirectional antennas and CSMA/CA mechanism as shown in Fig. 1a. Assume that we have a function $\mathcal{F}$ based on existing nonsaturation CSMA/CA performance models like [4] that can estimate the average service rate μ_o based on the number of competing nodes and the per node packet arrival rate λ as $\mu_o = \mathcal{F}(N, \lambda)$. Then, the average packet delay D_o including the queuing time can be estimated using the M/M/1 queuing model as $D_o = \frac{1}{\mu_o - \lambda}$ seconds and the channel throughput Th_o as $Th_o = L\mu_o$ bits/seconds.

Now, assume that we have another single-hop network where all the nodes including the base station communicate using K sectored antennas. Then, N nodes will be divided into K regions where the nodes that reside in region k connect to the kth sector of the base station. Nodes in a given sector of the sink contends for the channel using CSMA/CA. Fig. 1b shows an example of such network where $K = 4$. We assume that the base station cannot activate more than one sector simultaneously and, thus, can only serve the nodes that reside on one sector at a time. To fairly serve the nodes in that reside in all regions, the base station need to use a TDMA system with K time slots where it activates the corresponding sectors during the allocated time slot. We assume that all the nodes are fully synchronized and, thus, know exactly when to transmit or stay idle.

Assuming that the nodes are uniformly distributed around the base station, the number of nodes in each region can be estimated as $\frac{N}{K}$. During the active time slot for a sector, $\frac{N}{K}$ nodes compete to access the channel. Using the model presented by the function $\mathcal{F}$, the service rate of single region can be estimated as $\mu_s = \mathcal{F}(\frac{N}{K}, \lambda)$. Then, the average packet delay using sectored antenna D_s can be estimated as

$$D_s = t\left(\frac{K-1}{2} + \left\lfloor \frac{\frac{1}{\mu_s - \lambda}}{t} \right\rfloor K + 1\right), \tag{1}$$

to account for the TDMA delay. Note that t is the slot duration. Similar to the omnidirectional case, the channel throughput Th_s can be estimated as $Th_s = L\mu_s$.

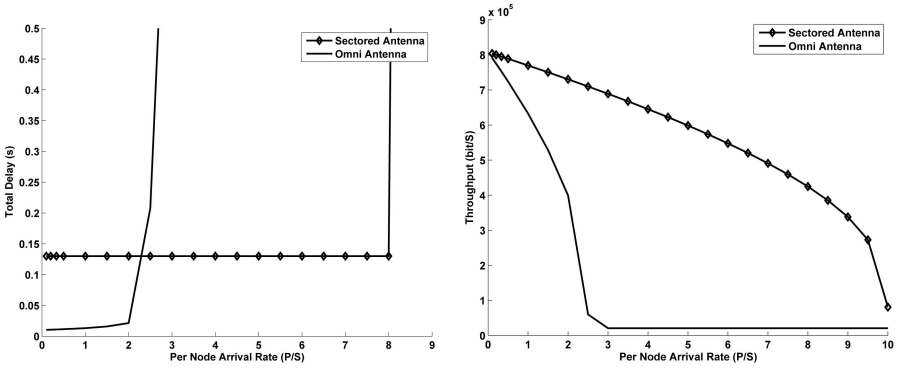


Fig. 2. Performance gain using sectored antenna in a single-hop CSMA/CA network. (a) Service delay. (b) Throughput.

Using the developed models above, we compared the delay and throughput of a 20 node single-hop network using omnidirectional and four-sectored antennas as the per node packet arrival rate increases.

The results shown in Fig. 2 clearly indicate that using sectored antennas can increase the performance of the network significantly, especially for higher arrival rates. Although the performance analysis above is for a single-hop network, it can be used to estimate the performance of a multihop WSN since these rates are not uncommon for event detection applications that generate spurs of reporting activities and around sink nodes where the entire network's traffic concentrated.

3 Related Work

Many energy-efficient sensor network MAC protocols were proposed in the literature. They can be generally classified into three main classes. Contention-based protocols like S-MAC [5], T-MAC [6], and B-MAC [7] save energy by specifying a duty cycle in which nodes become active and contend for the channel. Nodes go to sleep in the remaining part of the cycle to save energy. Despite the simplicity and low overhead of contention-based protocols, they suffer from packet collisions due to hidden terminal problem especially as the density of the nodes increases.

To reduce the contention between neighbor nodes, other MAC protocols use a time schedule to schedule the communications between neighboring nodes to occur at different times like Scheduled Channel Polling-MAC [8], LMAC [9], PEDAMACS [10], and TRAMA [11] where the time schedule is computed locally and exchanged between the neighboring nodes. Other MAC protocols use a hybrid approach between TDMA and CSMA like Z-MAC [12] and FlexiTP [13].

All of the previously mentioned protocols use an omnidirectional physical layer and thus bounded to the limited channel capacity that omnidirectional antennas fundamentally offer. The usage of directional antennas with wireless ad hoc networks has been proposed widely in the literature [14], [15], [16], [17]; however, those approaches are not suitable for WSN due to higher energy consumption and high hardware cost.

An approach for using directional antenna with WSN is proposed in [18], where the sink node equipped with a powerful antenna transmits a beacon to reach all nodes in the network. Upon the reception of the beacon, sensor nodes select the best beam that delivers the best signal to communicate and forward data packets to the sink node. Obviously, such antenna assumption is not practical and would limit the spatial span of the sensor nodes in the network.

Another approach for utilizing directional antennas with WSN is proposed in [19] where a time schedule is computed by each node to schedule the directional communication with other neighbors. After exchanging neighbors information, each node computes and exchanges its version of the time schedule with its neighboring nodes. When the time schedule for a node stabilizes after several exchange rounds, the node starts forwarding data packets. Obviously, the time it takes to stabilize the schedule is very long and cannot be bounded. Moreover, local computation of the time schedule is a challenging operation due sensor nodes' limited processing capabilities and memory sizes. In addition to directional antennas, this protocol also assumes the availability of an omnidirectional antenna for neighbor discovery mechanism.

Our proposed SAMAC protocol uses sectored antenna to increase delivery ratio by reducing contention, increase system throughput by maximizing spatial reuse and decrease energy consumption. SAMAC does not require an omnidirectional antenna in the sensors, unlike the solution proposed in [19]. The inclusion of an omnidirectional antenna in a directional antenna system may increase the hardware cost and degrades the potential benefits of using directional antennas. To synchronize the beam directions, SAMAC uses a TDMA approach where nodes discover where and when to activate certain sectors. Instead of computing the time schedule locally in sensor nodes like [19], SAMAC computes the time schedule in the sink node and then distributes it to the whole network. Central schedule computation of the time schedule 1) relives the resource-limited sensor nodes from such complex and costly operation, 2) allows the construction of an efficient computation of the time schedule, and 3) provides a bounded-time schedule computation operation.

To reduce the length of the superframe, SAMAC assigns the time slots to groups of nodes instead of individual nodes. Nodes with a group can communicate within the specified assigned time slot. To resolve the contention among themselves, they rely on basic CSMA/CA mechanism. To save energy consumption with SAMAC, nodes are only activated during their assigned time slots. In addition, SAMAC adopts an adaptive duty cycle where nodes sleep early when no communication activity is required.

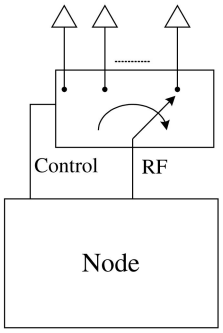


Fig. 3. Node architecture with sectored antenna.

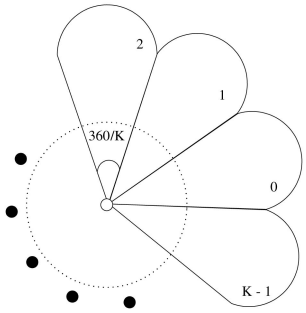


Fig. 4. Idealized sector antenna.

4 Preliminaries

To better understand SAMAC protocol, some preliminary information about antenna, node, and the network models are given in this section.

4.1 Antenna Model

The directional antenna architecture assumed for this protocol is a sector antenna as shown in Fig. 4 where K nonoverlapping sectors cover the entire omnidirectional 360 degree range. One major advantage of sectored antennas compared to other types of directional antennas is its low-cost and implementation simplicity which are very crucial when sensor networks are involved. Selecting different sectors is performed by a simple switch as shown in Fig. 3. Each node can only activate one sector at a time for both transmission and reception.

Note that SAMAC protocol is designed to combat unexpected interference due to third party transmission activities through side lobes for nonidealized realization of sectored antenna by accounting for interference in the time schedule computation as will be shown in Section 7.

Unlike many other wireless protocols proposed for directional antennas such as [19], [20], SAMAC does not require the additional leverage of an omnidirectional antenna for transmission or reception. In addition to the hardware cost and complexity, the usage of omnidirectional antenna causes two major problems. First, the transmission ranges of the omnidirectional antenna and directional antenna are different when the same transmission power is used. This phenomena introduces asymmetry in communication links which affects the performance of routing and MAC later on. Second, the spatial reuse benefits are greatly reduced since omnidirectional communications inhibits more simultaneous transmissions.

4.2 Network Model

In this paper, we assume a sensor network that is partitioned a priori into clusters that are served by predetermined set of high capability sink nodes. Within the cluster, the topology is flat over multiple hop. Throughout this paper, the operation of SAMAC protocol has been described for a single cluster. Cluster formation and the communication between clusters heads (i.e., sink nodes) are out of the scope of this paper.

When deployed in a multicluster network, an independent copy of SAMAC runs in each cluster, where cluster heads act as sinks. We assume that communication among clusters or to a network-wide sink occurs over orthogonal channels, possibly using another dedicated interface. Possible challenges may arise where different clusters can interfere with each other, leading to unexpected interactions among different instances of SAMAC running in these clusters. Intercluster interference is harmful only when it persistently occurs leading to packet collisions. A persistent interference between clusters requires the fulfillment of all the following conditions:

- Both instances of SAMAC must use the same superframe size.
- Interfering nodes in different clusters must be assigned synchronized time slots that lead to activation in overlapping time periods.
- Time synchronization in both clusters must be maintained codependently.

If any of the above conditions is not satisfied, then the intercluster interference is temporary. This observation is further strengthened by the low duty cycle of the sensor nodes that SAMAC takes advantage of. Nevertheless, in case of intercluster interference, SAMAC resolves these cases via CSMA/CA mechanism. In fact, SAMAC does *not* seek to eliminate all contention occurrences even in the formation of groups and handles contention in a graceful manner as the simulation results indicate. Consequently, interactions among clusters are limited in nature, and contention/interference events are overcome with contention resolution algorithms inherent to SAMAC.

The Network traffic is assumed to be either from the sink to the sensors for control messages (Downlink Traffic) or from the sensor nodes to the sink for data messages (Uplink Traffic). In other words, node to node communications are not allowed.

5 SAMAC Overview

Despite the potential benefits of sectored antennas, exploiting these benefits is challenging since a communication pair equipped with sectored antennas must select the appropriate sectors to point to each other during their communication session. Otherwise, the receiving node cannot receive the message due to weak reception power. As a result, trivial functions like neighbor discovery become extremely challenging. To address these challenges, additional functions should be performed before data forwarding can begin. SAMAC protocol can be described as an integrated cross-layer protocol that consists of different communication functionalities that allow high utilization of sectored antenna in WSNs. Fig. 5 shows the basic layout and the interaction of the different functionalities of

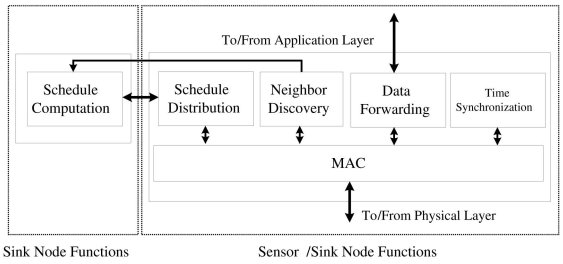


Fig. 5. SAMAC protocol overview.

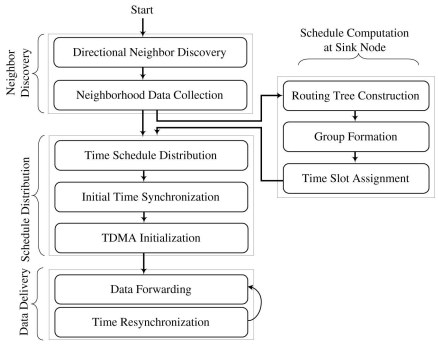


Fig. 6. SAMAC operation flow.

SAMAC protocol. An operational overview of the building blocks of the SAMAC protocol is provided below.

Before sensor nodes can forward data packets, SAMAC needs to initialize the network by performing the following tasks summarized in Fig. 6: 1) Discover and Collect Neighborhood information, 2) Compute and Distribute the time schedule, and 3) Establish initial time synchronization for all nodes.

SAMAC utilizes a directional neighbor discovery mechanism that neither requires an additional omnidirectional antenna nor time synchronization unlike many other proposed directional neighbor discovery mechanisms. To collect an accurate and complete neighborhood information, SAMAC utilizes a Token-Passing approach that serializes the neighbor discovery process among all nodes. When all nodes discover their neighbors, the complete neighborhood information is gathered at the sink node. Note that similar overheads are expected for almost all protocols that require at least local neighbor discovery. Moreover, these overheads are one-time overheads. Subsequent discoveries and updates are again of similar overheads of other protocols. Detailed description of SAMAC's directional neighbor discovery is provided in Section 6.

Transmission activities between nodes are governed by a centrally computed time schedule following TDMA mechanism. Time schedules are computed based on the collected neighborhood information. To minimize the number of time slots used and the computation time, SAMAC's scheduling algorithm assigns transmission time slots to groups of nodes instead of individual nodes. In Section 7, a more detailed description of time schedule computation is provided.

To reliably distribute the computed time schedules to all sensor nodes in the network, SAMAC utilizes the Token-Passing approach again. During this process, initial time synchronization is established among the nodes. More information about distributing the schedule and establishing the time synchronization is provided in Section 8.

Note that these operations are performed once for the whole network at the beginning of the network's lifetime, and thus, their overhead can be compensated by the network's long lifetime. During link or node failures, schedule computation and distribution are performed partially on the affected area of the network, thus, minimizing the overhead of such maintenance procedures.

Sensor node can start data forwarding by following their time schedules in TDMA mode. For successful TDMA operation, SAMAC depends on local time synchronization between the nodes and their parents. The time schedule for each node specifies the time slots at which nodes should be active or not. When active, nodes within a group reserve the channel using RTS-CTS handshake. During inactive time slots, nodes sleep to save energy consumption. The details of the time slots structures, data delivery, and the MAC protocol are described in Section 9.

6 Directional Neighbor Discovery

Neighborhood information plays an important role in multihop wireless networks. The knowledge of neighbor nodes is necessary to build routing trees and forward packets. Neighbor discovery is trivial when omnidirectional antennas are used since a simple broadcast can reach all neighboring nodes. This operation, however, becomes more challenging when directional antennas are used due to the following reasons: 1) Neighboring nodes must know when and where to point their directional beams to be able to discover each other. 2) The limited radial range of the sectored antenna, covering only $\frac{360}{K}$ degree of the azimuth, requires the neighbor discovery scheme to be repeated in K directions to cover the whole azimuth.

Many neighbor discovery protocols have been proposed for wireless networks that use directional antennas. Some of the proposed protocols utilize an omnidirectional antenna to bootstrap the neighbor discovery process like in [16], [19], [21]. Beside the cost factor of adding an omnidirectional antenna (especially to WSNs) omni and directional antennas have different communication ranges when the same transmission power is used. As a result, neighbors discovered using omnidirectional antennas may not reflect all possible neighbors around a node.

Other protocols require time synchronization to perform neighbor discovery like in [16], [22] by either using GPS or other time synchronization schemes. While this approach may be suitable for ad hoc networks, it is, clearly, unsuitable for WSNs due to the large overhead incurred.

In addition to the aforementioned limitations, existing directional neighbor discovery protocols do not gather the neighborhood information in a centralized location which is needed in SAMAC to compute the time schedule.

SAMAC utilizes a fully directional neighbor discovery mechanism without the need of an additional omnidirectional antenna or time synchronization during the neighbor discovery process. In addition, the neighbor discovery in SAMAC gathers the neighborhood information at the sink node for schedule computation. During neighbor discovery process, all nodes use CSMA/CA for communication. To reduce collisions and ensure accurate and complete neighborhood information, SAMAC serializes the neighbor discovery process using a Token-Passing approach. Serialized operation ensures that at any time instant only one

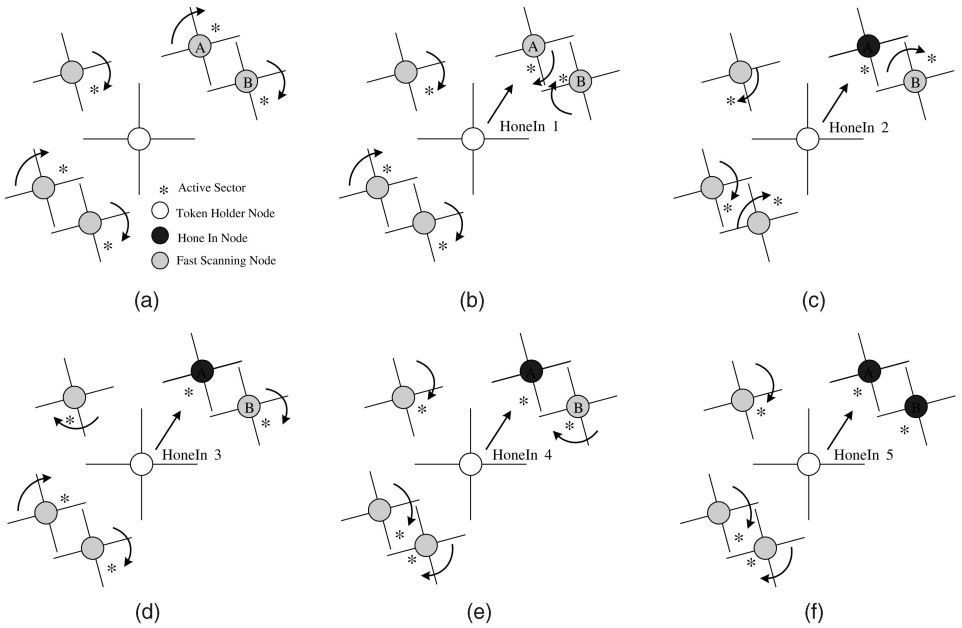


Fig. 7. Hone-In process with Four-Sectored Antenna: (a) All Nodes except the TH node are in Fast Scan mode, (b) TH broadcasts the first Hone-In beacon to its first sector while node A activates the sector pointing toward the TH node, (c) Node A receives the Hone-In beacon and locks its sector while node B continues switching its sectors, (d) TH broadcasts its third Hone-In beacon, (e) Node B switches to the sector pointing toward the TH node and receives the fourth Hone-In beacon, and (f) Node B locks its sector toward the TH node and stops switching its sectors.

node in the network performs neighbor discovery while all other nodes are listening.

Initially, nodes start in *Fast Scan* mode where they switch their sectors sequentially looking for any channel activity in the active sector. During this mode, each node waits for t_{switch}s before switching to another sector. The token is assigned, initially, to the node that will gather all the neighborhood information later on (i.e., sink node). The node that holds the token is called the Token-Holder (TH) node. Once a node holds the token, neighbor discovery process starts which can be summarized in the following three steps: 1) Holding the attention of its neighbors (Hone-In mechanism), 2) Discovering neighbors (Hello-Reply Mechanism), and 3) Passing the token to a neighbor or Releasing the token to the parent (Token-Passing and Releasing Mechanisms). The details of each step is presented below.

6.1 Hone-In Mechanism

Before the TH can discover its neighbors, the TH should hold the "attention" of its neighbors. Recall that, initially, all nodes are in Fast Scan mode switching their sectors looking for any channel activity. For that, the TH starts the Hone-In process where it broadcasts multiple $Hone\text{-}In$ beacons on all of its sectors to catch the attention of all neighboring nodes. Once a neighboring node receives one of the $Hone\text{-}In$ beacons, it stops switching its sectors and Hone In toward the TH by locking onto the sector that receives the beacon.

An example of the Hone-In process in one sector is shown in Fig. 7. Initially, all nodes are in Fast Scan mode switching their sectors except the TH as shown in Fig. 7a. In Fig. 7b, neighbor node A switches to the sector pointing toward the TH node. The TH node starts sending the first Hone-In beacon into its first sector. Neighbor node A receives the beacon and locks its current sector as shown in Fig. 7. Neighbor node B, however, does not receive the first Hone-In beacon because it activates a sector that does not point toward the TH node. TH node keeps sending Hone-In beacons until node B switches to the right sector, receives the beacon, and locks its sector as shown in Figs. 7e and 7f. This exact process is repeated in the reminding $K-1$ sectors of the TH node. To guarantee that all neighbor nodes receive at least one $Hone\text{-}In$ beacon, the TH must send at least $K+1$ $Hone\text{-}In$ beacons.

6.2 Hello-Reply Mechanism

After the Hone-In process ends, the TH starts discovering its neighbors by polling them and waiting for their replies to add them as neighbors. This approach guarantees the discovery of bidirectional links between neighboring nodes.

In each sector, the TH broadcasts a *Hello* packet and waits for the *Reply* packets from its neighbors within that sector. After the reception of the *Hello* packet, neighbor nodes start a slotted time with length of N_{Slots} slots and select a random slot number to send their *Reply* packets. This is done to minimize the collision probability of the *Reply* packets coming from the neighboring nodes. When two nodes select the same slot number, their *Reply* packets collide, and thus, they cannot be discovered by the TH in this Hello-Reply round. Each *Reply* packet includes the ID of and the active sector index of the neighbor node along with a bit that indicates whether this node has performed the neighbor discovery process or not. Once the TH receives *Reply* packet from a neighboring node, it adds the neighbor's information into the neighbor table.

The TH repeats *Hello-Reply* packet exchange for N_{Hello} times to discover new neighbors which their *Reply* packets have been colliding in the previous exchanges. To maximize the probability of discovering all nodes within a bound time, the TH node includes the IDs of all neighboring nodes that have been discovered in the *Hello* packet in successive *Hello-Reply* rounds. Only the nodes that cannot find their IDs in the *Hello* packet should send their *Reply* packets.

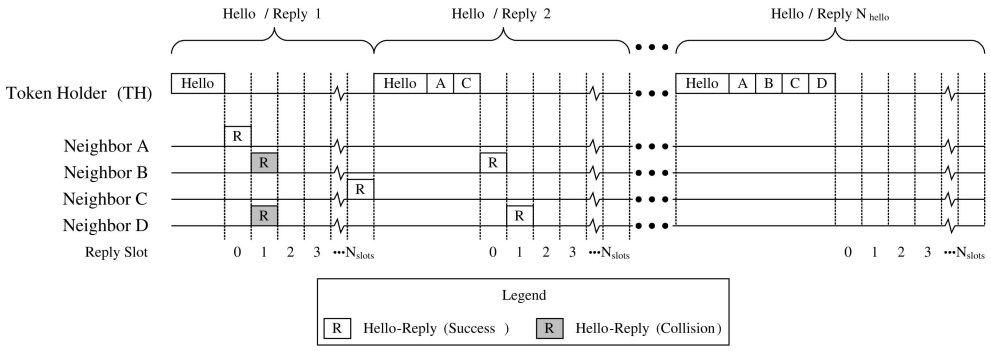


Fig. 8. Hello replay messages exchange.

An example of the *Hello-Reply* packet exchange for one sector is given in Fig. 8. After neighbor nodes receive the first *Hello* packet, they start a slotted time and select a random slot to transmit their *Reply* packets. Both node B and D select the slot number 1 to send their replies. As a result, their reply packets collide. While nodes A and C select the first and last slot, respectively. After the first *Hello-Reply* round, the TH discovers both nodes A and C but not B and D. In the second *Hello-Reply* round, the TH includes the IDs of the already discovered nodes, nodes A and C, in the Hello packet to increase the probability of discovering nodes B and D.

This *Hello-Reply* packet exchange is repeated in the remaining $K-1$ sectors in order to discover all neighbors in all directions.

6.3 Token-Passing Mechanism

Recall that during *Hello-Reply* exchanges, each replying node sets a bit in the reply packet to indicate whether the replying node has performed its neighbor discovery or not. After the TH node finishes its Hello-Reply packet exchanges, it searches its neighbor table for the first node that has not performed the neighbor discovery yet and passes the token to that node. To do that, the TH sends a *GoToFastScan* packet with the ID of the new TH node in the destination field to all K sectors. Upon the reception of this packet, the new TH waits for the token, while other neighbor nodes go to Fast Scan mode and start switching their sectors.

After sending the *GoToFastScan* packet to all K sectors, the TH sends the Token to the new TH and waits for an ACK to ensure a reliable transmission of the token. Once the old TH receives the ACK, it returns back to the Fast Scan mode and starts switching its sectors. The new TH tags the old TH node as a parent. If the TH node could not find any node that needs the token, it releases the token to its parent as will be shown in following section.

6.4 Token Releasing Mechanism

When the TH cannot find any neighbor node that requires the token, it releases the token back to its parent. However, before it releases the token back, it should hold the attention of its parent using a sectored version of the Hone-In process described earlier. Instead of performing the Hone-In process in all sectors, the TH node performs the Hone-In process only in the sector that contains its parent node with the parent node ID included in the *Hone-In* beacons. Only the parent node would hone in toward the TH. Once the Hone-In process finishes, the TH inserts its neighborhood information into the token release message and sends it to the parent node who will become the new TH node.

Once a node receives the token back from one of neighbors, it searches the neighbor table to find another neighbor node that requires the token. If it finds a neighbor node that requires the token, the TH passes the token using the Token Passing Mechanism described above. Otherwise, it releases the token to its parent after augmenting its neighbor table to the token release message. Once all nodes perform their neighbor discovery, eventually, the token is released back to the sink node along with the whole neighborhood information.

7 Schedule Computation

After the complete neighborhood information is gathered at the sink node, the sink node starts the time schedule computation process which can be summarized in the following steps:

1. Routing Tree Construction,
2. Group Formation,
3. Conflict Relations, and
4. Time Slot Assignment.

In SAMAC, time schedule is computed centrally in the resource-rich sink node to relieve the sensor nodes from performing this relatively complex and costly operation and allow the construction of a more efficient time schedule. More detailed description of each step is presented in the following sections.

7.1 Routing Tree and Group Formation

Using the collected neighborhood information, the sink node starts building the shortest hop routing tree rooted at the sink node using the Bellman-Ford algorithm. As a result, parent-child relations are determined during this step.

Instead of assigning time slots to individual nodes, SAMAC's scheduling algorithm assigns time slots to groups of nodes which will reduce the number of time slots needed to schedule the whole network and reduce the computation time of the schedules. Nodes' groups are formed using a recursive process that runs on the constructed shortest hop routing tree. Starting from the sink node, the root of the subtree forms a group in each of its sectors. Each group consists of the root in addition to the immediate children reachable over the same sector. The procedure continues by taking each of the children as a root node and repeating the procedure. By iterating from parents nodes to their child

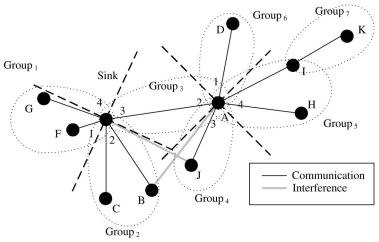


Fig. 9. Group formation.

nodes in the shortest hop tree, this process continues toward the leaf nodes until all nodes are considered. Consequently, each node may be involved in a single group as child and in multiple groups as a parent.

Fig. 9 illustrates this process. The process starts by forming groups around the sink node first. $Group_1$, $Group_2$, and $Group_3$ are formed. Then, starting from node A, $Group_4$, $Group_5$, and $Group_6$ are formed. Finally, $Group_7$ is formed from node I.

7.2 Conflict Relations

Using the complete neighborhood information of the network, the sink node determines conflict relations. Some conflicts are created by potentially contending communications, whereas some others are caused by the specifications of the radio hardware. A conflict relation between two communication sessions requires that these sessions be scheduled in two different time slots.

In Fig. 10a, nodes A-B and C-D are neighbor nodes, respectively. To enable the communication between nodes A and B, they should be scheduled at the same time slot over sectors 2 and 4, respectively. However, transmissions from sector 2 of node A are received at sector 3 of node D. Hence, if the communication between nodes C and D, and the communication between nodes A and B are scheduled to the same time slot, then packet collisions can occur between these two transmissions. Hence, these two communication pairs should be scheduled to different time slots as much as possible. This type of conflict is called *common interface conflict*.

Another type of conflict is shown in Fig. 10b. There are two communicating pairs of nodes: A and B, and A and C. Nodes B and C communicate on different sectors of node A. Since simultaneous transmissions on more than one sector of a node is not possible due to hardware limitations, node A cannot simultaneously communicate with nodes B and C. Hence, these two communication pairs should be scheduled for different time slots. This conflict type is called *common node conflict on multiple sectors*.

The last type of conflicts is the *common node conflict on a single sector* which is shown in Fig. 10c. With this conflict, two nodes B and C are connected to node A through the same sector. While the first two types are resolved using different time slots, this type of conflict is resolved using the basic CSMA/CA mechanism.

Examples of these conflicts can be shown in Fig. 9. Common interface conflict appears between $Group_2$ and $Group_4$ because node B and J can communicate with both the sink node and A. Common node conflict on multiple sectors appears between $Group_1$ and $Group_2$ because nodes in both groups are using two different sectors of the sink node. Finally, common node conflict on a single sector can

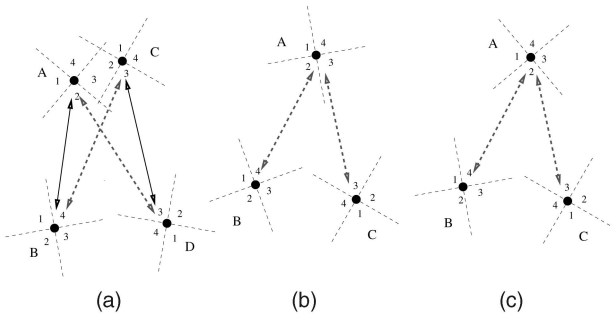


Fig. 10. Conflict types. (a) Common interface conflict. (b) Common node conflict on multiple sectors. (c) Common node conflict on a single sector.

be shown in $Group_5$ between nodes I and H. Conflict relations are used in time slot assignment described in Section 7.3 to avoid assigning the same time slots to conflicting groups such that packet collisions are avoided.

Time slots can be assigned to the groups by accounting for the conflicts between nodes of different groups. A conflict relation between two nodes X and Y belonging to different groups, is inherited by the groups of X and Y as well. The sink node checks each pair of groups for possible common node and common interface conflicts using the neighborhood relations before assigning time slots to the groups.

7.3 Time Slot Assignment

Time slot assignment for wireless network is similar to edge coloring in graph theory which is shown to be an NP-complete problem. Thus, time slot assignment is performed by a heuristic coloring algorithm following a depth-first-search approach. Our heuristic approach minimizes the number of time slots (i.e., colors) and the computation time by assigning communication time slots to group of nodes.

The scheduling algorithm assigns the time slot contiguously over each branch of the routing tree starting from leaf node toward the sink node. The main objective of this scheduling algorithm is to minimize the number of time slots used in superframe. Thus, the maximum number of used slots is incremented whenever a link cannot be scheduled with the existing time slots. The scheduling algorithm terminates when all groups are scheduled with a time slot. Termination of the scheduling algorithm is guaranteed since a link is not reconsidered once a time slot is assigned to it. The steps of this algorithm can be outlined as follows:

1. The groups are sorted in descending order according to their distance from the sink node, then according to the size of the group. The distance of a group is the hop distance of the group's parent to the sink node.
2. From the ordered list, the first unscheduled group is selected. Such group is a leaf group and it is the first group to be assigned a color in the multihop branch according to the shortest hop tree.
3. The currently used maximum color index is kept as ***"maximum color index."*** The selected group is assigned to the first available color value in the set of colors. A color is not available if there is a conflict relation with another group which is also assigned to the same color. If a conflict exists, then the color index is incremented and the new color is checked for a possible assignment. If the maximum color index is reached as a result of the increments, then

the maximum color index is incremented and it is assigned to the group.

4. After a color is assigned to the leaf group, the algorithm iterates group by group over the branch toward the sink node. For each group, the available colors indexes are considered for the assignment in ascending order one by one starting from the most recently assigned color value. If a color with no conflicts with previous assignments is found, then that color is used for the assignment. Otherwise, the maximum color index is exceeded and color index goes back to the first one in the color set and all colors are tried until leaf group's color is reached. In that case, the maximum color index is incremented and the new color is assigned to the group.
5. When all group on this branch of groups are assigned a color and the sink node is reached, then the next group is selected as the first group in the list of group with no color assignments. This new group corresponds to the one with the next largest hop distance to the sink node. Then, steps 2 and 4 are performed for the new branch between this leaf group and the sink node.
6. Color assignment algorithm terminates when all groups are scheduled with a color assignment.

8 Time Schedule Distribution

After computing the time schedule at the sink node, the next task is to distribute the schedule to all nodes. The schedule distribution process should guarantee that each node in the network receives its time schedule correctly. Token passing approach is used again to eliminate collisions and guarantee accurate delivery of the time schedule. In addition, initial time synchronization is performed during the schedule distribution process by determining the time Δt required for processing and transmitting a time synchronization control message.

Note that after the time schedule computation is performed, nodes become bounded with parent-child relations. Once becoming a TH, each parent node is responsible to distribute the time schedule to its children using the following steps:

1. Hone-In mechanism,
2. Token Passing,
3. Initial Time Synchronization,
4. Time Schedule Extraction and Transmission, and
5. Token Releasing.

Recall that, all nodes go to Fast Scan mode after performing their neighbor discovery as discussed in Section 6. Before the TH node can communicate with any of its children, it should perform the sectored Hone-In process to get the attention of that particular child. Once the sectored Hone-In process finishes, the TH node passes the token to the child node. Once the token is received by the child node, it starts the initial time synchronization process.

The initial time synchronization is essential for each node to estimate the processing and transmission time Δt for the time synchronization control message which will be used later to synchronize the nodes during the TDMA data delivery phase described in Section 9.2. Once the child node receives the token from its parent, it initiates a pairwise time synchronization sequence by sending a *TimeSyncRequest* message to its parent node, saving the transmission time stamp t_0 to its local memory as shown in Fig. 11. The TH node saves the time stamp t_1 of when the Request message was received and creates a *TimeSyncReply* message for its child node. The Reply message contains both time stamp t_1 (when the *TimeSyncRequest* message was received) and time stamp t_2 (when the *TimeSyncReply* message was transmitted). The child node saves the time stamp t_3 when it receives the *TimeSyncReply* message. The child node, then, uses all four time stamps to generate the Δt according to the following equation:

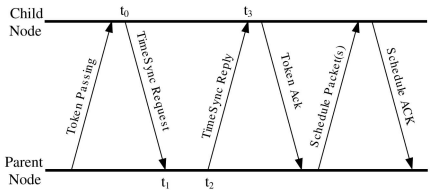


Fig. 11. Schedule distribution and time synchronization packet exchange.

$$\Delta t = \frac{(t_3 - t_0) - (t_2 - t_1)}{2}. \quad (2)$$

After the initial time synchronization is performed, the child node acknowledges the reception of the token and the time schedule transmission is started. To reduce schedule distribution overhead, the parent node only transmits the time schedules for the nodes that belong to the subtree rooted at the new TH. Once the child node (the new TH node) receives the time schedules, it extracts its own time schedule and starts distributing the time schedules to its children using the same procedure. If the TH node does not have any child nodes that need a time schedule (e.g., the TH node is a leaf node). The TH node releases the schedule token back to its parent. The parent node, once receives the token, passes the token to another direct child following the same previous steps.

When the token is released finally to the sink node, the sink node broadcasts a *NetworkReady* message to its children nodes. *NetworkReady* message contains the starting time of the current time slot, the current time slot, and superframe indexes. Upon the reception of this message, children nodes start to follow the TDMA time schedule and become ready for Data Delivery Phase. Following the time schedule, they propagate the *NetworkReady* message to their children during the appropriate time slot as will be shown in Section 9.

9 Data Delivery

When all nodes receive their time schedule, data delivery phase starts. During this phase, sensor nodes follow their time schedule by enabling the corresponding sector at the specific time slot. Nodes go to sleep during inactive time slots to save energy.

9.1 Time Slot Structure

The computed time schedules are based on time slots as the basic building blocks of the protocol operation time. The time slot structure in SAMAC is shown in Fig. 13. Each TDMA time slot begins with a guard time to allow for discrepancies in the time synchronization between child

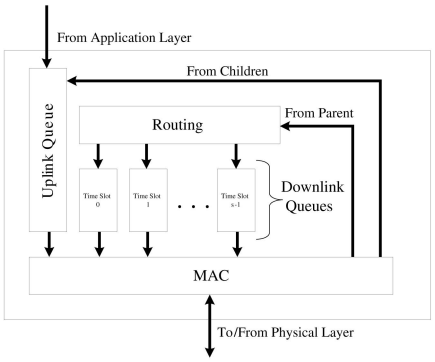


Fig. 12. Queue organization.

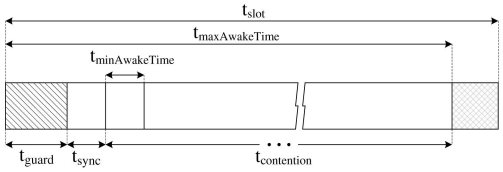


Fig. 13. Time slot structure.

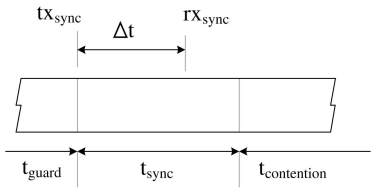


Fig. 14. TDMA synchronization.

and parent nodes. The guard time also is important to account for the sector switching delay. After t_{guard}, the nodes enter the synchronization phase of the TDMA time slot which lasts for t_{sync}. The synchronization process will be explained in Section 9.2. After $t_{guard} + t_{sync}$ time, all child nodes become synchronized with their parent node and ready for data transmission.

The duty cycle of the system is defined by the parameter $t_{maxAwakeTime}$ which defines the maximum time that a node should stay active during each time slot. During this period of the time slot, nodes can engage in any type of transmission activity as will be explained later in Section 9.4. For more power efficiency, SAMAC adopts an adaptive duty cycle where nodes go to sleep earlier based on the traffic level. Specifically, if a node has no data to send and there is no data reception during $t_{minAwakeTime}$ time period from the last time it has been active, the node sleeps in the remaining of the time slot to save energy.

9.2 TDMA Synchronization

During the time schedule distribution described in Section 8, the parent and child nodes initialize their time synchronization by determining the time required for processing and transmission of a time synchronization control message, referred to as Δt. This value is assumed to remain constant throughout the life of the network, as no contention should be present during either the initialization of time synchronization, or during the TDMA synchronization.

During time slots where a node acts as a parent, the node should broadcast a *Sync* message to all children in the sector. When the node acts as a child node during a specific time slot, it waits for the *Sync* message from its parent. The time synchronization process is performed every m superframe where m greatly depends on the clock drift of the sensor nodes hardware.

Fig. 14 illustrates the synchronization process. When the child node receives the *Sync* message at time rx_{sync}, it knows that its parent has sent the *Sync* message at time tx_{sync}. To synchronize with the parent, the child node should wait for $t_{sync} - \Delta t$ time.

After t_{sync}, all child nodes are synchronized with their parent node, and thus, they can start with their normal communication activities as will be explained in Section 9.4.

9.3 Packet Forwarding

As mentioned before, with SAMAC only two types of traffic are allowed. Data traffic coming from sensors toward the sink node is called Uplink traffic. Whereas, control packets coming from the sink toward the sensor nodes are called Downlink traffic.

Uplink traffic is routed toward the sink node by forwarding the packet to the sensor node's parent. Downlink traffic, however, requires routing information at intermediate sensor nodes. As all downlink control packets originate from the sink node, source routing is utilized to reduce the computational overhead in intermediate nodes. When downlink control packets are generated, the sink node uses its complete knowledge of the network to generate an ordered pair of intermediate nodes and their sectors (e.g., {Node 4, Sector 3}), to route the packet to its final destination.

Fig. 12 shows the internal organization of a node during packet delivery phase. During each active time slot, a node must check the packet queue that is associated with the current TDMA time slot. The uplink queue is associated with the TDMA time slot in which the node can communicate with its parent. Downlink queues are associated with time slots where the node acts as a parent node. Note that transmission queues are associated with the active time slots but not sectors. This assignment helps in saving the node's memory since a node may be active for only one time slot (e.g., leaf nodes).

Packets coming from the application layer (i.e., generated data within the node) are stored in the Uplink queue. Similarly, packets received from children are queued in the uplink queue since they represent an uplink traffic and must be forwarded to the sink node. Downlink packets (i.e., coming from a parent node), are stored in the appropriate downlink queue according to the time slot where the routing sector is activated.

Although, SAMAC has a virtual queue for each active time slot. Those queues can be efficiently implemented as a single memory heap with per slot pointers to access the queues for each time slot. In addition, for efficient implementation, downlink queues' sizes can be configured to be very small compared to the single uplink queue which also helps in reducing the memory usage overhead.

9.4 MAC Protocol Details

As mentioned before, during network initialization, all nodes access the channel using basic CSMA/CA mechanism. Upon the reception of the schedule packet and the *NetworkReady* message through the time schedule distribution mechanism as described in Section 8, nodes start to operate in the TDMA mode. During this mode, nodes follow their time schedules

Active	Time Slot	Active Sector	Parent Slot
True	1	2	True
True	2	4	False
True	3	3	False
False	4	X	X

Fig. 15. Schedule information of a node.

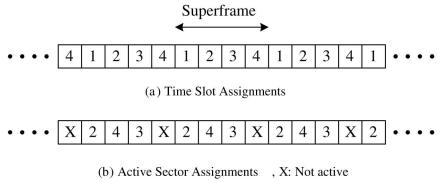


Fig. 16. Superframe realization of the schedule in Fig. 15.

and only access the channel during active time slots through the specified active sector antenna. Fig. 15 shows a time schedule for a given node A. According to the time schedule, node A communicates with its parent node on its second sector in time slot 1. Moreover, node A communicates with its children through sectors 4 and 3 in time slots 2 and 3, respectively. During time slot 4, node A is inactive and, thus, sleeps to save energy. Fig. 16 shows the superframe realization of the time schedule presented in Fig. 15.

During active time slots both parent and children nodes can access the channel. Downlink packets coming from the parent nodes are control packets and, thus, have higher priority than uplink data packets. To achieve this prioritization, parent node accesses the channel immediately at the beginning of the time slot while children nodes back off before sending their uplink data packets. One example of such control packets is the *Sync* message used for time synchronization as described in Section 9.2.

If a child node does not have any packet for transmission, it must wait for $t_{minAwakeTime}$ before it goes to sleep again to be able to receive any downlink control packet from the parent. If the child node receives downlink packets from its parent, it waits for $t_{minAwakeTime}$ after the reception of the last packet and then sleeps if the channel stays idle during this time as shown in Fig. 17a. If, on the other hand, no downlink packets are received before $t_{minAwakeTime}$ time, the child node goes to sleep until its next active time slot as shown in Fig. 17b.

If a child node has a packet for transmission, it should reserve the communication channel with a four-way RTS/CTS/DATA/ACK handshake to reduce contention with other nodes within the same group. Before sending an RTS packet to initiate this process, the node needs to back off from channel access for a certain amount of time. During back off, if the channel becomes busy before the back off period is over, then the node pauses its back off and saves the remaining back off time period, as shown in Fig. 18a. During this time, the node updates its NAV timer and defers from accessing the channel until the ongoing transmission in its neighborhood finishes. After the channel is sensed idle again, the node resumes its back off. If the node has to sleep due to $t_{maxAwakeTime}$ while it is backing off, the back off timers and states are saved to be used in the same time slot of the following superframe as shown in Fig. 18b.

Note that these communication events happen during the active time slot duration $t_{connection}$. If the remaining time in the time slot is not sufficient to transmit a whole packet, the transmission is deferred until the same slot in the next superframe.

10 Failure Recovery

WSNs are prone to network dynamics like nodes dying due to hardware failures or battery depletion or link disconnection due to environment factors or obstacles. Such network dynamics may partition the network and disrupt data delivery.

SAMAC protocol strongly binds the nodes to a strict time schedule to organize the communication using the sectored antenna. To account for network dynamics, SAMAC adopts a reactive approach to discover and report such changes to the sink node, recompute, and redistribute a new time schedule to the affected nodes. In this section, the steps needed to recover from nodes and links failures are presented.

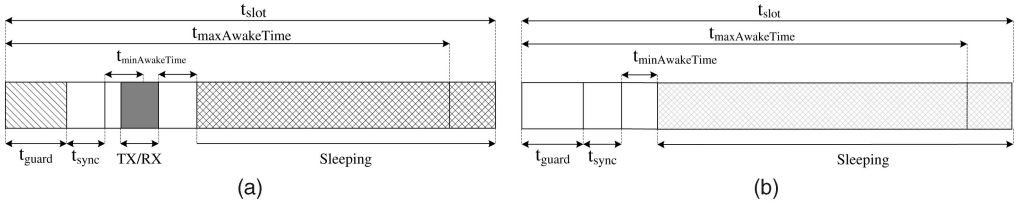


Fig. 17. Node sleeps after $t_{minAwakeTime}$ of inactive channel. (a) Node sleep after one transmission. (b) Node sleeps after no activity during $t_{minAwakeTime}$.

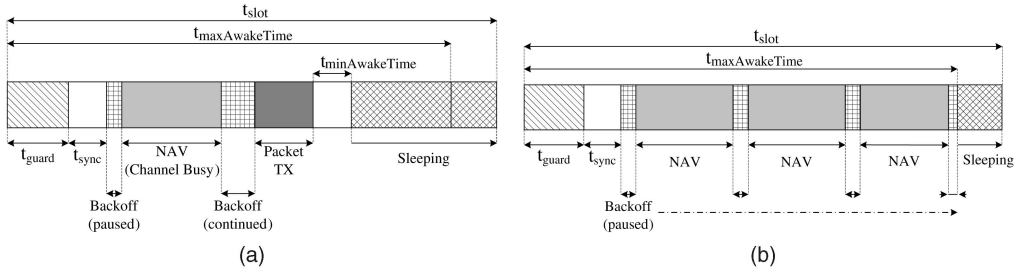


Fig. 18. Back off pausing. (a) Back off pause due to busy channel. (b) Back off pause due $t_{maxAwakeTime}$.

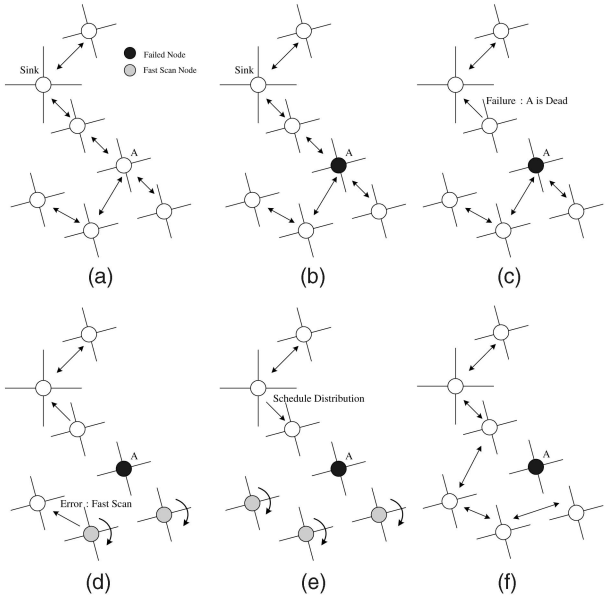


Fig. 19. Node failure recovery mechanism: (a) network topology, (b) node fails, (c) node's A parent inform the sink node about the failure, (d) node's A children fast scan their sectors and inform other children about the failure, (e) sink node recomputes and distributes the new schedule, and (f) packet forwarding resumes using the new time schedule and topology.

With SAMAC, sensor nodes are organized as parent-child groups. When a node or link fails, both the parent and the children of that node would discover the failure and act accordingly. Each node that had children, polls its children when it does not hear from them for $S_{stillAlive}$ superframes. If a child node does not reply to the poll, the parent node assumes that the child node had died or the link between them has failed and reports the failure to the sink node. Meanwhile, when a node does not receive the synchronization message from its parent for two consecutive superframes, it assumes that its parent node has died or the link between them has failed. The node and all its descendent nodes become orphans and disconnected from the whole network. The node that discover the failure sends an *Error* message to all its decedent nodes ordering them to fast scan their sectors for any control packet from the sink node.

Once the sink node received the topology changes, it recomputes the time schedule following the steps presented in Section 7 based on the new topology. After the new time schedule is recomputed, only nodes with a new time schedule are included in the distribution phase. To reach the disconnected nodes who are fast scanning their sectors, parent nodes need to perform the Hone-In process during the time slots that they are supposed to communicate with those nodes.

An example of failure recovery is presented in Fig. 19. When the parent node discovers that node A is dead, it sends the *Error* message to the sink node with the ID of the failed node as shown in Fig. 19b. Meanwhile, child nodes of node A discover that their parent is dead and they switch to Fast Scan mode and inform their children of the error as shown in Fig. 19d. The sink recomputes and distributes the new time schedule to the affected nodes. Once they receive the time schedule, orphan nodes resume their packet forwarding using the new schedule over the new topology.

TABLE 1
SAMAC Configuration Parameters

Parameter	Definition	Value
$t_{SlotDuration}$	Duration of a time slot	200ms
$t_{MinAwakeTime}$	Minimum Awake time	60ms
$t_{MaxAwakeTime}$	Maximum Awake time	0.98 * $t_{SlotDuration}$
$t_{BackoffSlot}$	Length of a single contention slot	20us

TABLE 2
Energy Consumption for Different Activities

Parameter	Value[1]
SLEEP POWER	0.06 mW
RX POWER	59.1 mW
TX POWER OFFSET	59.1 mW
TX POWER	52.2 mW

1. These values are taken from CC2420 data sheet.

11 Performance Evaluation

In this section, the performance analysis of the SAMAC protocol is presented. The analysis includes the comparison of the SAMAC protocol against 802.11 [23], DMAC [15], TRAMA [11], and B-MAC [7]. We have selected 802.11 as the most widely used CSMA-based protocol. DMAC is a CSMA-based directional antenna protocol suited for ad hoc wireless networks. Note that we have included 802.11 and DMAC as a baseline comparison. TRAMA is a schedule-based MAC protocol designed for sensor networks. While, B-MAC is an example of CSMA-Based energy efficient sensor networks MAC protocol. Since it has been shown in [7] that B-MAC outperforms S-MAC and T-MAC, we have not included S-MAC and T-MAC in the comparison. The performance evaluation of SAMAC and other protocols are performed using the Qualnet simulator [24].

11.1 Protocol Parameters

Simulations are performed on sensor nodes randomly distributed on $100\,\mathrm{m} \times 100\,\mathrm{m}$ network. Sensory data are routed to the sink node which is located in the middle of the network (i.e., $\mathrm{X} = 50$, $\mathrm{Y} = 50$). The payload of the sensory data is 40 bytes. Unless otherwise specified, the default number of nodes in the network is 64 nodes. Each simulation runs for 1,000 s, with 80 s initialization phase. An error-free channel is assumed.

Table 1 shows the definitions and values of the configuration parameters of SAMAC. For 802.11 and DMAC, the standardized back off parameters are used as defined in 802.11 standard [23]. We have used the standard 802.11b as a physical layer for both protocols expect for the data rate where it is modified to 100 kbps. For TRAMA, the signal slot duration used is 8.08 ms and the transmission slots duration is seven times longer. The SCHEDULE_INTERVAL parameter is set to 100, and the random access period is set to 504 signal slots (or 72 transmission slots) as suggested in [11]. For BMAC, the LPL check interval is selected to be 100 ms as suggested by Polastre et al. [7] when the number of local neighbors is about 5.

The underlaying physical layer rate for all protocols is modified to transmit 100 kbps, and the power coefficients used to calculate energy consumption are listed in Table 2.

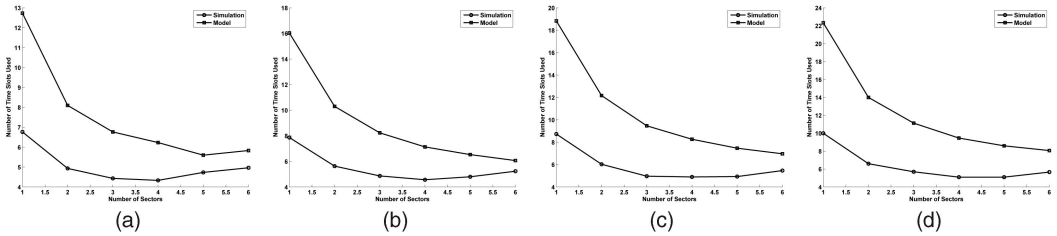


Fig. 20. Average number of time slots used to schedule a network with different number of nodes: (a) 64 nodes, (b) 81 nodes, (c) 100 nodes, and (d) 121 nodes.

Since SAMAC has routing capability, Qualnet's BELLMAN-FORD protocol is used as a routing protocol for all other MAC protocols. To reduce the effect of the routing protocol on the performance of the MAC protocols, the routing protocol collects and updates route information only during the initialization phase. It uses the resulting routing table for the reminder of the simulation without exchanging any further route information. The queue size used for all protocols is 50,000 bytes. The performance metrics observed are E2E Delay, Delivery Ratio, Throughput, Information Propagation Speed, Per node energy consumption and Total network energy consumption. The transmission power for each protocol is set such that each node has a transmission range of 20 m.

11.2 Selecting the Number of Sectors

One of the most important design parameters used in SAMAC is the number of sectors K used in each node's antenna. Since the delay and throughput characteristics of any TDMA system depend greatly on the number of time slots used in the schedule computation. This section will show the relation between the number of time slots used and the number of sectors by developing an analytical model to estimate the maximum number of time slots needed to properly schedule a network topology using the proposed SAMAC protocol. Then, using experimental results, we find the best number of sectors that results in the lowest number of slot needed.

The number of time slots needed to properly schedule the links in a network can be estimated using Vizing's theorem which states that a graph can be edge colored in either Δ or $\Delta + 1$ colors, where Δ is the maximum degree of the graph. Unfortunately, Vizing's theorem cannot be applied directly to our network with SAMAC protocol for two reasons 1) the usage of sectored antenna and 2) a time slot is assigned to group of nodes instead of individual nodes. Recall from Section 7.2 that we have two conflict relations namely Common Interface Conflict and Common Node Conflict on Multiple Sectors, represented by Figs. 10a and 10b, respectively.

Let us consider the first conflict type in Fig. 10a. Nodes B and D are both neighbors of node A through its second sector. However, only node B is a child of node A while node D is a child of another parent node C. As stated earlier in Section 7.2, the two groups that contain B and D should be assigned two different time slots. Otherwise, nodes' B and D transmissions will be collided at their parent nodes. From A's perspective, node D is refereed to as a ***Potential Interferer*** node since it is a neighbor but not a child. Hence, node D needs a different time slot to communicate with its parent (i.e., node C).

Now to represent that in a mathematical form, let us define the Sector-Degree as the degree of the node in a single sector transmission area. For instance, the Sector-Degree for node A and sector 2 is 2. Furthermore, let us define Δ_u^s as the maximum Sector-Degree of node u among all sectors. Now, consider the group that has the maximum Sector-Degree Δ_u^s; the maximum number of potential interferes in this group is $\Delta_u^s - 1$ since at least one of them should be a child of node u. This means that this group, in the worst case, would have conflicts with $\Delta_u^s - 1$ other groups according to the first conflict type.

Now, let us turn to the second conflict type in Fig. 10b. Due to the physical hardware specifications, a node cannot simultaneously communicate through multiple sectors. Thus, a node u would need k different time slots to communicate with its k groups that resides on k sectors where $k \leq K$. Recall that in SAMAC, a node can be member of one group as a child and multiple groups as parent. Then, let us define the Group-Degree of node u Δ_u^g as the number of groups that contains node u either as a parent or as a child. Using both Δ_u^s and Δ_u^g, the modified node degree for node u, $\overline{\Delta_u}$, under SAMAC protocol can be defined as $\overline{\Delta_u} = max(\Delta_u^g, \Delta_u^s - 1)$. Then, the maximum node degree of the graph is given as $\overline{\Delta} = \max_u(\overline{\Delta_u})$ which can be used according to Vizing's theorem to estimate the number of time slots needed to calculate a proper schedule.

The modified node degree $\overline{\Delta}$ greatly depends on the number of sectors. Since the sectored antenna spatially divide the neighbor nodes into K different area (sectors), the Sector-Degree and, thus, the maximum Sector-Degree Δ_u^s is a function of node density and K. Moreover, the Group-Degree Δ_u^g is a function of the number of sectors K.

To validate our upper limit, we run SAMAC's scheduling algorithm with different number of sectors and node densities and measure the average number of time slots used to schedule 30 topologies with each sector-density configuration. The results shown in Fig. 20 show that the modified node degree represented by the equation $\overline{\Delta} = \max_u(\overline{\Delta_u})$ provides an upper limit to the average number of time slots used. Also, the results show that four sectors consistently produce the minimum number of time slots for the different densities. Based on these results, Four-Sectored antennas have been used for performance evaluation.

11.3 Data Gathering

In this set of simulations, data gathering scenarios are simulated where random nodes send periodic information to the sink node. The simulation results reflect the averages of 25 topology-source node placement combinations. These 25 combinations are obtained by randomly generating five topologies and choosing five independent random source

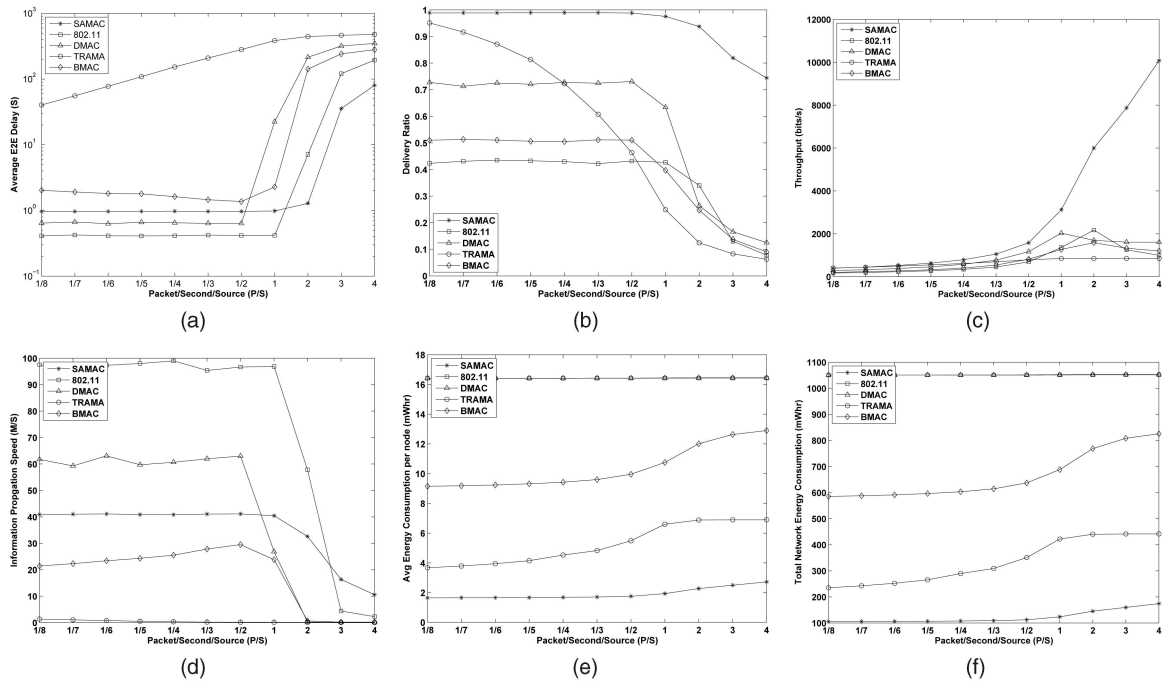


Fig. 21. Data gathering. (a) Average E2E delay. (b) Data delivery ratio. (c) Data throughput. (d) Information propagation speed. (e) Average energy consumption per node. (f) Total energy consumption for the network.

node sets. Each source node set contains 10 randomly selected source nodes which transmit 40 bytes of data at different packet arrival rates. The packet arrival rate for each source node is increased from 1/8 (P/S) to 4 (P/S).

Fig. 21a shows the average E2E delay for all protocols. Average E2E delay is defined as *the time between the generation of the packet in the application layer of the source node until it is successfully received in the application layer of the final destination node*. In general, contention-based protocols (i.e., 802.11 and DMAC) have lower delay compared with schedule-based protocols. The delay of DMAC is higher than 802.11 because of the extra time DMAC needs to transmit the RTS packet in all directions. As the packet generation rate increases, DMAC and 802.11 suffer from high queuing delay due to contention. The E2E delay for BMAC greatly depends on the LPL check interval time which is the sleeping duration of the nodes before it wakes up to transmit their packets. In SAMAC, each node needs to wait for its scheduled slot time for packet transmission; therefore, the delay depends on the slot duration and the superframe size. TRAMA has high delay due to the scheduling mechanism as described in [11].

Although the E2E delay of DMAC and 802.11 is lower than SAMAC for lower packet generation rate, this comes at a price of lower delivery ratio as shown in Fig. 21b. The delivery ratio is defined as *the ratio of the number of packets arriving at the final destination node to the total number of packets generated from the source nodes*. Delivery ratio for contention-based MAC protocols including BMAC is smaller compared with SAMAC. This is due to the interference from other nodes. As the flow of packets is going toward sink node, more packet dropping happens near the sink node.

Since DMAC uses sectored antennas, the effect of interference from other nodes is reduced, resulting in a higher delivery ratio for DMAC compared with 802.11. In the case of SAMAC, however, each node has a scheduled time slot to communicate with its parent. Even when more than one nodes are scheduled to communicate with the same parent in the same time slot, the contention is much smaller. Although TRAMA provides a contention-free channel access, the delivery ratio for TRAMA decreases rapidly as the packet generation rate increases as packets take long time in the queues.

Fig. 21c shows the throughput of all the protocols. The throughput is calculated as *the total number of arrived bits divided by the duration of the application session*. Since the throughput is mainly controlled by the delivery ratio, 802.11 has the lowest throughput. DMAC has better throughput with lower packet generation rate. However, with 2(P/S) packet generation rate, less packets are being delivered with DMAC causing the throughput to decrease. The throughput of 802.11 is topped at 2(P/S) after which it starts to decrease. SAMAC throughput is the highest, which is an advantage of the spatial reuse provided by the sectored antenna.

Fig. 21d shows the Information Propagation Speed which is defined as *the distance between the source and the sink of a packet divided by the E2E delay of the packet* and is given by $\text{IPS} = \frac{d_{i,\mathcal{S}ink}}{D_{i,avg}}$ where $d_{i,\mathcal{S}ink}$ is the distance between node i and the sink node $\mathcal{S}ink$, and $D_{i,avg}$ is the average E2E delay between data source i and sink node. TRAMA has the lowest speed compared with other protocols because of the large average E2E delay. Note that higher speed in 802.11 and DMAC comes with lower delivery ratio. BMAC protocol has a lower speed due to its sleeping mechanism which delays packet transmission.

Figs. 21e and 21f show the average energy consumption per node and the total energy consumption for the whole network, respectively. For fair comparison, we have used the energy consumption rates summarized in Table 2 for all protocols. The energy consumption of DMAC and 802.11 is almost the same. They have higher energy consumption

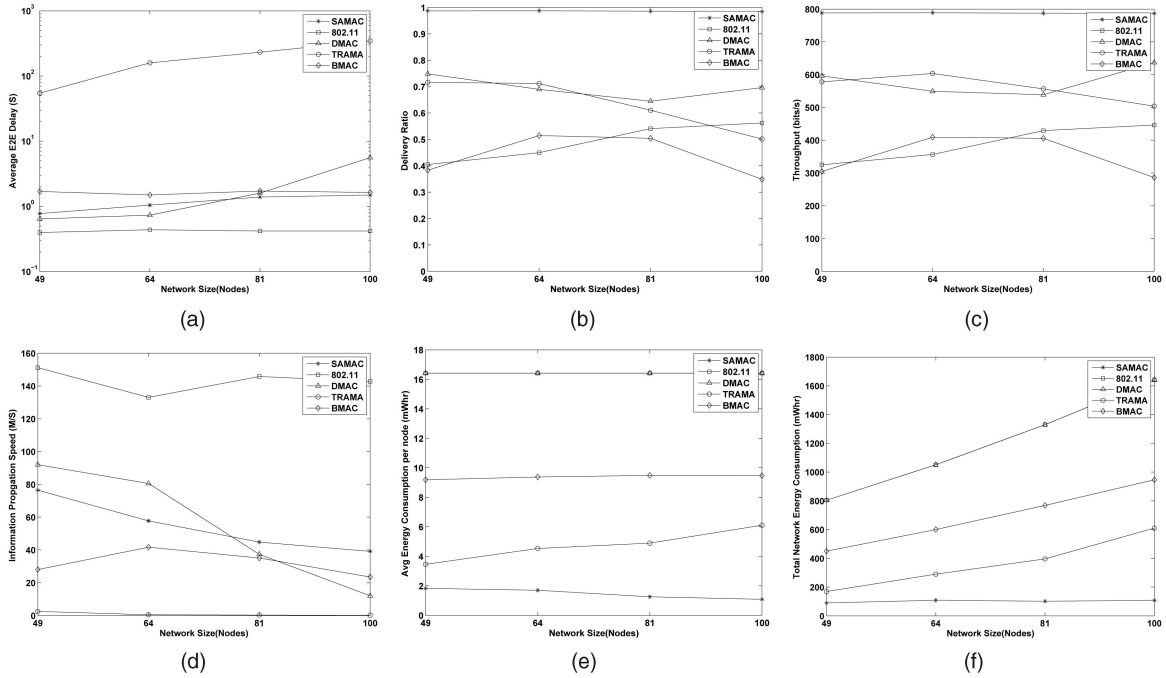


Fig. 22. Network density scenario. (a) Average E2E delay. (b) Data delivery ratio. (c) Data throughput. (d) Information propagation speed. (e) Average energy consumption per node. (f) Total energy consumption for the network.

because they perform idle listening continuously. BMAC performs better than 802.11 and DMAC due to the sleeping mechanism. However, the energy consumption is still larger than TRAMA due to the transmission of the long preamble prior to the transmission of each data packet. On the other hand, scheduling-based protocols consume much less energy because nodes are not active in all the time because of the scheduling. SAMAC consumes the lowest energy among all other protocols due mainly to the usage of sectored antenna which requires much less transmission power to cover the same transmission range provided by omnidirectional antenna. Moreover, SAMAC reduces the contention by scheduling the transmission of different groups of nodes. Finally with low traffic, SAMAC benefits from the adaptive duty cycling where a node goes to sleep when no transmission or reception activity are performed. Consequently, as more packets are generated, nodes become more active in their slots and, thus, will not go to sleep as often which explains the increase of the energy consumption.

11.4 Effects of Network Density

In this set of scenarios, the scalability of the proposed SAMAC protocol is compared to other MAC protocols. Here, the number of sensor nodes increased on the same area from 49 to 100 nodes, thus, increasing the density of the network. The simulation results reflect the averages of 25 topology-source node placement combinations. These 25 combinations are obtained by randomly generating five topologies for each network size and choosing five independent random source node sets. Each source node set contains 10 randomly selected source nodes. The simulation result for density experiment is shown in Fig. 22.

Fig. 22a shows the average E2E delay of the different flows. TRAMA has the highest delay due to its scheduling mechanism. With low network density SAMAC has higher delay compared to contention-based protocols due to the slot duration and the superframe size. As the network size increases, the size of superframe (i.e., the number of time slots used) for SAMAC increases which explains the slight increase in the delay as the density increases. 802.11 delay is the lowest; however, 802.11 has the lowest delivery ratio and throughput as can be seen in Figs. 22b and 22c, respectively. Interestingly, as the network density increases, the average energy consumption per node for TRAMA increases as well as shown in Fig. 22e. This is due to the distributed scheduling mechanism that TRAMA adopts. As the density increases, the number of neighbors and two-hop neighbors increases. Each node needs to distribute its schedule to every neighbor node in a contention-free slot. Thus, increasing the time each node is active. BMAC shows a steady average power consumption with different densities which is a result of the sleeping mechanism that BMAC adopts. Another observation is that as the network density increases, the total network energy consumption increases for all protocols except for SAMAC as shown in Fig. 22f. When more nodes are available in the network, nodes will follow their time schedule to determine sleep time. In their active slots, nodes with no traffic go to sleep early in the slot. Since the traffic load is the same with different network sizes, the network energy consumption stays at the same level for SAMAC.

11.5 Target Tracking

The SAMAC protocol is tested for target detection application and compared to 802.11 and DMAC protocols. Due to its large delay, TRAMA is excluded from this test. In this experiment, three moving objects move randomly in the field using Random Waypoint model. The speed of these moving targets is varied from 0.1 to $5\ \mathrm{m/sec}$. When sensor nodes detect the presence of a target within their sensing ranges, they generate a data packet and send it to the sink node. Nodes sense their ranges every second of the simulation time to monitor the neighborhood for an intruder. The sensing

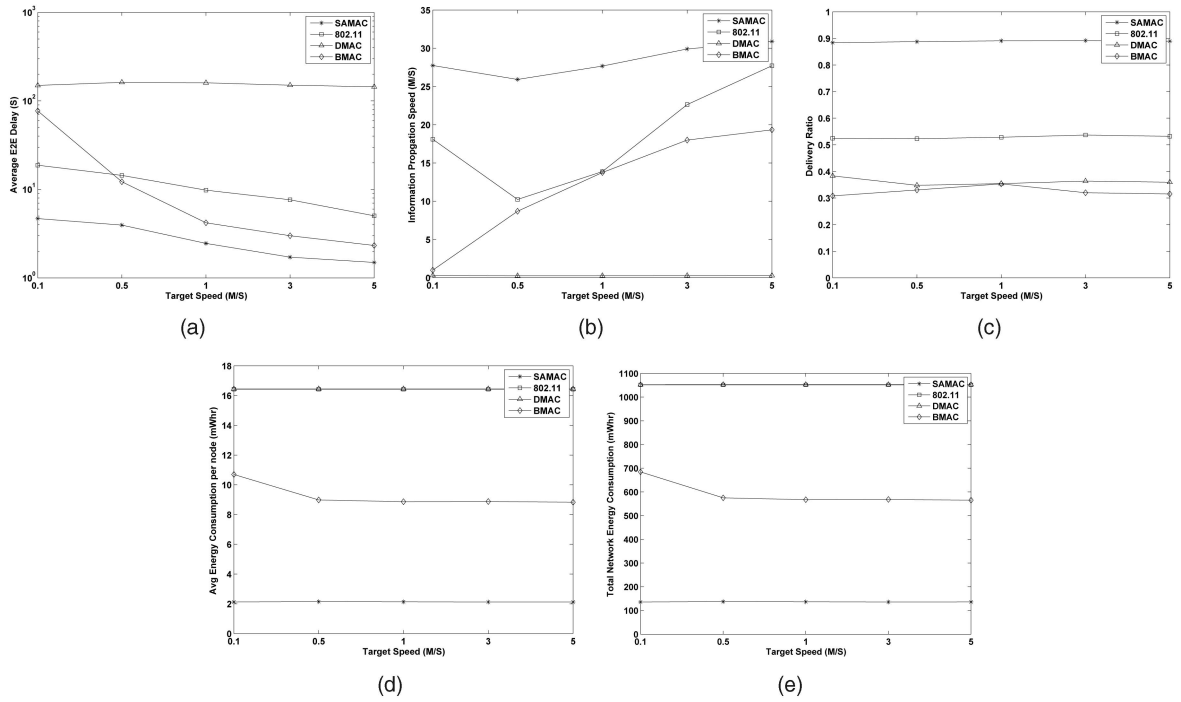


Fig. 23. Target tracking scenario. (a) Average E2E delay. (b) Information propagation speed. (c) Data delivery ratio. (d) Average energy consumption per node. (e) Total energy consumption for the network.

range is set to 20 m. The simulation results shown in Fig. 23 is the average of 25 runs with different topologies, seeds, and initial position of the moving objects.

Fig. 23a shows the average E2E delay for the different protocol under comparison. For all protocols, as the target speed increases the delay decreases. When targets move slowly, they spend long time within an area. Congestion starts to appear in that area and queues start to build up. The large delay is apparent in DMAC since it requires more time to transmit its packets. Recall that DMAC needs to send multiple RTS to different sectors. On the other hand, as the speed of the targets increases, moving objects spend less time in each area. Thus, distributing the load on the network. With slow target speeds, BMAC suffers from high delay due to its transmission mechanism where it waits for its active time to transmit packets. The decrease in the delay for BMAC is apparent as the target speeds increase; however, it is still higher than SAMAC's delay. 802.11 has a higher delay compared to SAMAC because of contention. As a result of low delay for SAMAC, the information propagation speed for SAMAC is higher compared to 802.11 and BMAC as shown in Fig. 23b.

12 Conclusions and Future Work

In this paper, we explored the benefits of using directional antenna with WSN by introducing the SAMAC protocol. Many trivial tasks that can be realized in omnidirectional antenna networks become more challenging when directional antenna are used. SAMAC is an integrated cross-layer protocol that contains full set of communication mechanisms for sensor networks equipped with sectored antennas. The proposed SAMAC protocol delivers high energy efficiency and predictable End-to-End delay. Optimization studies to enhance the performance of SAMAC will be considered in the future work which include time slot assignment, time slot duration, and neighbor discovery parameters selection.

Acknowledgments

This research was supported by the Dual Use Technology Program, South Korea (06-II-LC-01, Surveillance and Reconnaissance Sensor Network Development). This work has been performed while Emad Felemban and Serdar Vural were at The Ohio State University.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine*, vol. 40, no. 8, pp. 102-114, Aug. 2002.

[2] A.E. Zooghby, *Smart Antenna Engineering*. Artech House, 2005.

[3] G. Giorgetti and E.A.A. Cidronali, "Exploiting Low-Cost Directional Antennas in 2.4 GHz IEEE 802.15.4 Wireless Sensor Network," *Proc. 10th European Conf. Wireless Technology*, pp. 217-220, Oct. 2007.

[4] O. Tickoo and B. Sikdar, "Modeling Queueing and Channel Access Delay in Unsaturated IEEE 802.11 Random Access MAC Based Wireless Networks," *IEEE/ACM Trans. Networking*, vol. 16, no. 4, pp. 878-891, Aug. 2008.

[5] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. IEEE INFOCOM*, vol. 3, pp. 214-226, 2002.

[6] T. van Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. ACM SenSys*, pp. 171-180, 2003.

[7] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proc. ACM SenSys*, p. 95107, 2004.

[8] W. Ye, F. Silva, and J. Heidemann, "Ultra-Low Duty Cycle MAC with Scheduled Channel Polling," *Proc. ACM SenSys*, pp. 321-340, 2006.

[9] L. Hoesel and P. Havinga, "A Lightweight Medium Access Protocol (LMAC) for Wireless Sensor Networks," *Proc. Int'l Workshop Networked Sensing Systems (INSS '04)*, June 2004.

[10] S. Ergen and P. Varaiya, "PEDAMACS: Power Efficient and Delay Aware Medium Access Protocol for Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 5, no. 7, pp. 920-930, July 2006.
[11] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Energy-Efficient, Collision Free Medium Access Control for Wireless Sensor Networks," *Proc. ACM SenSys,* pp. 181-192, 2003.
[12] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: A Hybrid MAC for Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 16, no. 3, pp. 511-524, June 2008.
[13] W.L. Lee, A. Datta, and R. Cardell-Oliver, "FlexiTP: A Flexible-Schedule-Based TDMA Protocol for Fault-Tolerant and Energy-Efficient Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 19, no. 6, pp. 851-864, June 2008.
[14] Y. Ko, V. Shankarkumar, and N. Vaidya, "Medium Access Control Protocols Using Directional Antennas in Ad Hoc Networks," *Proc. IEEE INFOCOM,* pp. 13-21, 2000.
[15] T. Korakis, G. Jakllari, and L. Tassiulas, "A MAC Protocol for Full Exploitation of Directional Antennas in Ad-Hoc Wireless Networks," *Proc. ACM MobiHoc,* pp. 98-107, June 2003.
[16] R. Ramanathan, J. Redi, C. Santivanez, D. Wiggins, and S. Polit, "Ad Hoc Networking with Directional Antennas: A Complete System Solution," *IEEE J. Selected Areas in Comm.,* vol. 23, no. 3, pp. 496-506, Mar. 2005.
[17] K. Sundaresan and R. Sivakumar, "A Unified MAC Layer Framework for Ad-Hoc Networks with Smart Antennas," *IEEE/ACM Trans. Networking,* vol. 15, no. 3, pp. 546-559, June 2007.
[18] T. Dimitriou and A. Kalis, "Efficient Delivery of Information in Sensor Networks Using Smart Antennas," *Proc. Algorithmic Aspects of Wireless Sensor Networks,* pp. 109-122, 2004.
[19] A.D.S. Zhang, "A Directional Antenna Based MAC Protocol for Wireless Sensor Networks," *Proc. Int'l Conf. Computational Science and Its Applications (ICCSA '05),* pp. 686-695, 2005.
[20] R. Choudhury, X. Yang, R. Ramanathan, and N. Vaidya, "Using Directional Antennas for Medium Access Control in Ad Hoc Networks," *Proc. ACM MobiCom,* pp. 59-70, Sept. 2002.
[21] R. Santosa, B. Lee, C. Yeo, and T. Lim, "Distributed Neighbor Discovery in Ad Hoc Networks Using Directional Antennas," *Proc. IEEE Int'l Conf. Computer and Information Technology,* 2006.
[22] W.L.G. Jakllari and S. Krishnamurthy, "An Integrated Neighbor Discovery and MAC Protocol for Ad Hoc Networks Using Directional Antennas," *IEEE Trans. Wireless Comm.,* vol. 6, no. 3, pp. 1114-1024, Mar. 2007.
[23] *IEEE Std 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification,* IEEE, 1999.
[24] Qualnet, http://www.scalable-networks.com, 2010.

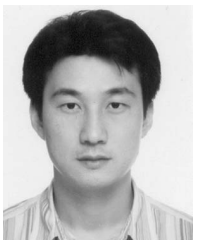
Emad Felemban received the master's and PhD degrees from The Ohio State University in 2003 and 2009, respectively. He is an assistant professor at the Department of Computer Engineering, Umm al-Qura University (UQU), Makkah, Saudi Arabia. Currently, he is the deputy director for research and development of the information technology and technical support center of UQU. His current research interests are in wireless sensor networks, performance analysis of wireless networks, and QoS provisioning. He is a member of the IEEE.

Serdar Vural received the BS degree in electrical and electronics engineering in 2003 from Bogazici University, Istanbul, Turkey, and the MS and PhD degrees in electrical and computer engineering in 2005 and 2007, respectively, from The Ohio State University, Columbus. After completion of his PhD program, he worked as a postdoctoral researcher at the University of California, Riverside, from June 2008 to November 2009. He joined the Centre for Communication Systems Research in the University of Surrey, Guildford, UK, in January 2010. His research interests include medium access control and routing protocols for wireless sensor networks and wireless ad hoc networks and network forensics in wireless mesh networks. He is a member of the IEEE.

Robert Murawski received the BS and MS degrees in electrical engineering in 2003 and 2004, respectively. He then worked in the area of wireless networking, developing wireless communication networks for the United States Military. In 2007, he returned to academia and is currently pursuing the PhD degree in electrical engineering from The Ohio State University. He is a student member of the IEEE.

Eylem Ekici received the BS and MS degrees in computer engineering from Bogaziçi University, Istanbul, Turkey, in 1997 and 1998, respectively, and the PhD degree in electrical and computer engineering from Georgia Institute of Technology, Atlanta, in 2002. Currently, he is an associate professor with the Department of Electrical and Computer Engineering, The Ohio State University. His current research interests include cognitive radio networks, nano-scale networks, vehicular communication systems, and wireless sensor networks, with a focus on routing and medium access control protocols, resource management, and analysis of network architectures and protocols. He is an associate editor of the *IEEE/ACM Transactions on Networking*, *Computer Networks Journal* (Elsevier), and the *ACM Mobile Computing and Communications Review*. He is a member of the IEEE.

Kangwoo Lee received the BEng and MS degrees in computer science and engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 2005 and 2007, respectively. His master's research focused on remote code update for wireless sensor networks. He is currently working for the Electronics and Telecommunications Research Institute (ETRI) in Korea. His current research interests include MAC protocol in wireless sensor networks with a special emphasis on dependability of embedded systems. He is a student member of the IEEE.



Youngbag Moon received the MS degree in electronics engineering from the KyungHee University, Korea, in 2001. Currently, he is working as an engineer in the Electronics and Telecommunications Research Institute (ETRI), Korea, from 2000. His research interests are in wireless ad hoc and sensor networks, and testbed. He is a member of the IEEE.

Sangjoon Park received the BS and MS degrees in electronics engineering from the Kyung-Pook National University in 1988, and 1990, respectively. He received the PhD degree from the Computer Science Department in the North Carolina State University in 2006. He is currently a senior researcher in Electronics and Telecommunications Research Institute (ETRI). He was the project manager of the Surveillance and Reconnaissance Sensor Network Research Project in Electronics and Telecommunications Research Institute (ETRI), Korea. His current research interests are in wireless sensor network, next generation embedded sensor network, multisensor data fusion, and target tracking. He also worked as a senior researcher in the Agency for Defense Development (ADD) from 1990 to 2001. He is a member of the IEEE.