EVAM-MAC: An Event Based Medium Access Control for Wireless sensor Networks with Multihop Support

Zaher Merhi[1], Mohamed Elgamel[2], Samih Abdul-Nabi[1], Amin Haj-Ali[1], and Magdy Bayoumi[3]

[1]Department of Computer and Communication Engineering
Lebanese International University
Beirut, Lebanon
{zaher.merhi, samih.abdulnabi, amin.hajali}@liu.edu.lb

[2]Arab Academy for Science
Technology and Maritime Transport Alexandria, Egypt
cshaheen@hotmail.com

[3]Center for Advanced Computer Studies
University of Louisiana at Lafayette
LA, USA
mab@cacs.louisnana.edu

Abstract

As wireless sensor network applications are becoming more complex, the need for a versatile medium access control that is able to deliver high data rates is essential. In event-based systems, sensor nodes spend most of their time at sleep state waiting for an event to occur. When an event is detected, sensor nodes experience a short abrupt period of high data contention where the data packets are large in size. As the number of sensor nodes per-hop increases, the contention generated will lower the throughput, increase latency and deteriorate the application's performance. EVAM-MAC is a medium access control that is tailored specifically for event-based systems. EVAM-MAC shifts the contention generated by multiple sensor nodes trying to deliver the collected measurements to the control phase. Furthermore, it arranges data transfer in a contention free environment by dynamically creating a TDMA-like schedule without global synchronization or global slot assignments. EVAM-MAC offers a platform of configurable operations that can be programmed prior to deployment. The protocol is simulated with NS2 and compared with an implementation of S-MAC and 802.11 for multiple scenarios where EVAM-MAC presented its superiority in throughput, Latency and Energy consumption.

Keywords : *Clustering Algorithms, Event Detection, Medium Access Control, Wireless Sensor Networks*

1 Introduction

Medium access control protocols (MAC) are considered an integral part of any wireless sensor network (WSN) application. These applications are usually constrained by both power and time. Typically, WSN are battery operated where replacing and/or charging the battery is not always feasible. Thus, it is crucial for the application longevity to preserve power consumption. Furthermore, real time applications such as target tracking, cannot tolerate latency in data delivery. Dropped packets due to high data contention wastes both power and time. Initially, the trend with MAC protocols for wireless sensor networks was to trade for latency with energy consumption by duty cycling the radio transceiver [2] [3]. This suited monitoring application since wireless sensor devices transfer small amount of information every periodic timeframe. On the other hand, event based applications spend most of their time at sleep state waiting

for an event to occur. Once an event is detected, a short abrupt period of high data contention is exhibited due to multiple nodes contending for the channel. Thus, trading energy for latency is no longer a valid choice [1].

High throughput applications usually employ TDMA schemes. However, they are not efficient for wireless sensor networks [5], [6]. In TDMA schemes, data transmission is regulated by assigning a unique slot for each sensor node in the two-hop neighborhood. In event-based systems, only sensors detecting the event should participate in data transmission. Thus, time slots are wasted where sensors not wishing to transmit are forced to sleep. Slot reuse is employed by hybrid TDMA/CSMA schemes as in [7]. However, the overhead in creating and maintaining a schedule across multiple sensors at multiple hops consumes significant amount of energy. Although this overhead only occurs at deployment the cost of updating the slot distribution across the network for nodes joining and leaving the network (which is frequent in wireless sensor networks) is significant and cannot be neglected. Moreover, these schemes also employ global time synchronization that incur even more overhead in packet exchange and in maintaining clock drift corrections over time.

On the other hand, wireless sensor network is an application centric device. MAC protocols should offer a set of policies and procedures that can be configured to optimize event-based systems according to the applications' designers need. Moreover, event-based systems exhibits certain characteristics that the MAC protocol should take into consideration to achieve an optimized design. For an example, dynamic clustering is very desirable in event-based systems as only sensors detecting that event should wake up and deliver measurements to a local cluster head. Furthermore, it is more efficient to aggregate data at local cluster heads rather than having each node transmitting its measurements throughout the network. Static clustering is not efficient for wireless sensor applications since nodes are consistently joining and leaving the network. However, as sensor node count increases per cluster, the contention is heightened as multiple sensors try to deliver their recorded measurements to the cluster head. Since measurements are highly corrupted by noise in wireless sensor networks, prioritizing data transfer according to the received signal strength (RSS) of the event restrains the system from overshooting and insures quality of service [9]. On the other hand, measurements recorded from the captured event are highly correlated which make it sometimes useless to send more than a certain number of measurements according to a certain policy. For an example, in a 10-node network scenario, an application could be satisfied with only 5 high quality readings (i.e. high RSS). Transmitting all 10 readings will increase contention and waste both bandwidth and energy. A mechanism should be devised in order to instruct a subset of sensors belonging to a cluster to transmit according to the quality of their readings.

Furthermore, optimizations can be achieved by tailoring the MAC protocol to a pattern of communications that the application requires. For example, distributed signal processing application usually share the readings recorded by the sensor nodes that are in the one-hop neighborhood for processing via broadcast. Afterwards the result is unicasted to the cluster head for aggregation. Some applications require several rounds of such. The control packet overhead (RTS-CTS-ACK) can be minimized if the sensor nodes participating in the event register their interest. Thus, the MAC protocol can be configured to follow the patterns required by the application by dynamically constructing an efficient schedule. Lowering the control overhead will increase throughput and lower energy consumption by restraining the sensor nodes from contending for the channel each time a communication sequence is started for the same event when detected.

EVAM-MAC is an event based medium access control that addresses all the problems listed above. When an event is detected EVAM-MAC will dynamically create a virtual cluster containing only nodes that detected the event. The cluster is constructed by introducing a control gap that registers sensor nodes that have recorded viable data. A Schedule will be created for this cluster that is unique across the two-hop neighborhood. Thus, each node will have a contention free slot to send their data. EVAM-MAC is the extension of our work presented in [8] . This article is organized as follows: Section II discusses the related work, Section III presents EVAM-MAC protocol design, Section IV discusses the simulation experiments and results, and Section V is the conclusion and future work.

2 Related Work

Most medium access control for wireless sensor networks lowers energy consumption by decreasing the duty cycle of the radio. Having a low duty indicates that the radio is spending most of the time at sleep state. However, this has the effect of lowering the throughput. In Event-based systems the sensor nodes spend most of their time at sleep state waiting for an event to occur. Once the event is detected, a short

abrupt period of high data contention follows.

S-MAC [2] and T-MAC [3] are scheduled based protocols that preserve energy by duty cycling the radio. These MAC protocols trade energy with latency. However, in event-based systems it is desirable to have a real time response. S-MAC synchronizes nodes using a SYNC messages that assigns a schedule to each node. The overhead produced by the SYNC messages introduces huge latencies as the one hope neighborhood grows in size. T-MAC [3] improves on S-MAC by reducing the idle time spent using a time-out scheme. All communications are moved to the beginning of the frame which set the nodes into a fierce mode for acquiring the channel. However, it still suffers from the shortcomings of S-MAC.

Hybrid protocols such as the Z-MAC [7] and F-MAC [12], solves the problem of high data contention by incorporating both CSMA (Carrier Sense Multiple Access) and TDMA(Time Division Multiple Access) techniques. In Z-MAC, CSMA is used in low contention and TDMA is used in high contention. it uses DRAND [13] which assigns unique slots for the TDMA scheme in the two-hop neighborhood which is performed at deployment. This operation is a resource intensive task that includes neighborhood discovery, the operations of DRAND, and global synchronization. However, the overhead exhibited in nodes joining and leaving the network is large and cannot be neglected. Global synchronization also induces an overhead in maintaining coherent timing information that deteriorates the system performance.

Leach [14], [15] protocol groups sensor nodes into cluster where each node reports to the local cluster head. Data aggregation takes place at the local cluster head where aggregated data is sent directly to the base station. Leach operations are divided into rounds where each round consists of a setup and steady state phase. The disadvantage of Leach is that all nodes are required to be in communication reach of a base station which is impractical for wireless sensor network applications. Moreover, the number of sensors nodes and clusters formed are required to be known a priori which limits the flexibility that characterizes wireless sensor networks.

In BMA [4], the cluster head forms a schedule by arranging data transfer at each round. The cluster head accepts requests for data transfer and assigns a slot for each node wishing to transmit. Each round of data transfer is divided into contention, data transmission and idle period. Cluster formation is done by using Leach protocol [14]. The drawback of this protocol is that it permits centralized control where the cluster head is responsible for maintaining a schedule across nodes belonging to the same cluster. However, in wireless sensor networks it is more efficient to have a decentralized control which eliminates single point of failure. Moreover, nodes that have no data to send waste time slots in the contention period where idle listening and overhearing occurs. Furthermore, each session has a fixed time for its completeness. Thus, if few nodes wish to send data, idle time is wasted there which increase latency. Moreover, in dense deployment, if more nodes wish to send data, sessions may overlap and channel contention may occur in the data transmission period especially if they join before the cluster head set up its candidate nodes.

B-MAC [16] and X-MAC [17] utilizes CCA (clear channel assessment) to assess channel clarity and LPL (low power listening) for duty cycling the radio. Channel activity is checked by measuring the signal strength samples when the channel is assumed to be free. Nodes communicate with each other by sending a long preamble that is twice the check interval of the receiving nodes. Thus, nodes at sleep state have enough to time to wake up and detect the transmission. SCP-MAC [18] focuses on lowering the duty cycle by combining scheduling and channel polling. However, similar to previous protocols, they suffer from the problem of poor operation under abrupt high contention traffic especially in multihop scenarios.

AS-MAC [21] utilizes hello packets to build a neighborhood table. The receiver of the data packet follows a periodic schedule that the sender is aware of. Based on this schedule Nodes wishing to send data packets will wake up accordingly. AS-MAC uses static clustering with is inefficient in event-based systems. Likewise, RI-MAC [19] also relies on the receiver to announce its readiness to receive Data transmission by sending a beacon frame. All nodes follow a periodic schedule to check for pending data transmission. When a beacon is received, nodes with pending data start transmission. PW-MAC [20] enhances RI-MAC by introducing a predictive wake up schedule on the sender side. The sender node will predict the approximate time of the receivers wake-up schedule and transmit data based on that time. Although these MAC protocols present an enhancement on synchronous scheduled protocols, they still are not optimized for event-based systems. EVAM-MAC is specifically designed to address the needs of event-based systems where sensor node count per-hop is high. EVAM-MAC dynamically creates a contention free schedule each time an event is detected. EVAM-MAC achieves high throughput, maintains acceptable energy consumption levels, and control packet overhead due to the reduced number of dropped data packets.

3 EVAM-MAC Protocol Design

The main principle behind EVAM-MAC is to shift contention generated from multiple sensor nodes (in the two-hop neighborhood) transmitting the collected measurement from the data phase to the control phase. In event-based applications for WSN such as target tracking, sensor nodes transmit simultaneously large amounts of data. Thus, dropped packets due to the high contention generated, have a significant effect on performance. Having an optimized contention free schedule is crucial for real time operation and energy savings. EVAM-MAC creates a TDMA-like schedule dynamically, and in a decentralized fashion. Furthermore, it eliminates the use of global synchronization that has a high overhead to maintain and substitutes for it with local synchronization. EVAM-MAC does not assume any infrastructure constraints and does not require a setup phase as in hybrid protocols which facilitates nodes joining and leaving the network. Moreover, it prioritized the schedule based on received signal strength (RSS) reading of the collected data by giving high priority for sensor nodes with high RSS. This way sensor nodes with reliable data can be given a higher privilege than those with lower ones which otherwise can deteriorate the convergence of the application.

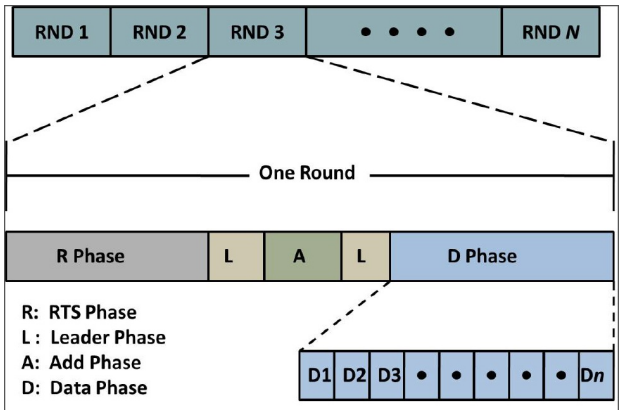


Figure 1: Round operations

EVAM-MAC's operation is divided into rounds as shown in Fig. 1. When an event is detected, only sensor nodes detecting that event initiate a round. Each round is dedicated for a single event where subsequent events are buffered for future rounds. The purpose of a round is to fulfill the application layer requirements for the successful delivery of the information for processing. This includes the communication pattern dictated by the application, such as, a broadcast of the collected data and then a unicast of the processed data to a dedicated node. EVAM-MAC adopts a modified version of B-MACs CCA and LPL techniques to assess channel activity and to duty cycle the radio. Instead of using a long preamble for LPL phase, EVAM-MAC replaces it with a train of packets to accommodate the CC2420 radio that is available on most wireless sensor motes. Once an activity is detected during LPL phase the sensor nodes stays awake for the entire round. As shown in Fig. 1, each round is divided into sub-rounds where each one has a specific role.

3.1 Neighborhood Discovery

The main purpose of the RTS Sub-round operation is to allow nodes detecting the same event to be included in the contention-free schedule. Once an event is detected by a group of sensor nodes that are in the communication reach of each other the RTS sub-round timer is initiated as shown in Fig. 2. EVAM-MAC assumes that sensor nodes detecting the same event are in the 1-hop neighborhood of each other. This assumption is considered reasonable since the communication range is usually much larger than the sensing range for most applications. Only nodes detecting the event will start broadcasting RTS packets to notify all nodes that a round is started. Each node checks for channel activity (using CCA) before transmitting its RTS packet. If the channel is busy, it backs off and tries again. Nodes periodically send their RTS packets to notify other nodes about the event. No acknowledgement is used for the RTS packets since each node detecting the event will continuously broadcast the RTS packet regardless of any successful reception as shown in Fig. 2. The reason behind this implementation is that we don't know a priori the number of nodes in the communication reach of the transmitting node or the number of nodes detecting the event. The RTS sub-round timer value is an application specific parameter that is provided according to the approximate sensor count and application needs. Higher values of the RTS sub-round timer will give more chances for nodes to be included in the schedule. The reason behind this is that as the sensor count increases, collisions also increases and some nodes will not be able to successfully broadcast their packets before the round timer expires. However, care must be taken when choosing the RTS sub-round timer value since large values will incur unnecessary overhead and with low values nodes may not be able to participate in the schedule.

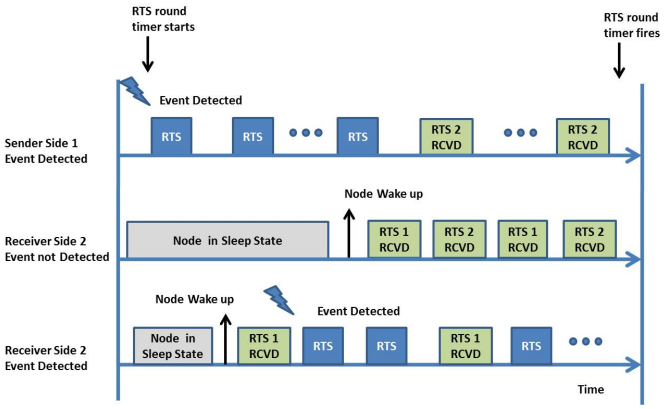


Figure 2: RTS Round operations

The RTS round phase is responsible for choosing the leader node, synchronizing neighboring nodes, dynamically creating the cluster and constructing a preliminary schedule. All of these operations are performed simultaneously in the RTS sub-round. The RTS packet is piggybacked with the following attributes:

- RSS: The received signal strength of the detected event (8-bits).
- Leader_Node: The node with the lowest time to detect the event (8-bits).
- Recv_Addr: The receiver address of the data packet (8-bits).
- Rts_Leader: Indicates if the sender node is a leader (1-bit), used for multi-hop resolution.
- TTE (Time to expire): The time required to finish the RTS sub-round timer (16-bits).
- ID: Node ID (8-bits)

3.1.1 Leader Selection and Time Synchronization

Since each node detecting the event starts its RTS sub-round timer independently, time synchronization is required for the next sub-rounds. When the schedule is created, each node detecting the event must have a slot that is synchronized and unique across the 2-hop neighborhood in the DATA sub-round. Otherwise, overlapping of data transmissions may occur which defies the whole purpose of EVAM-MAC (i.e. having a contention free channel for DATA transmissions).

EVAM-MAC implements only local synchronization each time an event is detected or a round is initiated. For the RTS sub-round, each node is faced with two options, if it has detected the event or not. A node detecting the event will have its RTS sub-round timer initiated and will be sending out RTS packets. Moreover, it will assume leadership of this round. Leadership status has no advantages in terms of processing or capabilities. It is just a privilege granted to a node so that it can maintain synchronization and scheduling. Nodes detecting the event dynamically elect the Leader node. The criteria used to choose the Leader node is the node with the earliest time to detect the event. This value is computed by determining the remaining time to expire (TTE) of the RTS sub-round timer which is piggybacked with the RTS packet. Sensor nodes not detecting the event in the two-hop neighborhood will be only receiving the RTS packets. Moreover, these nodes will update their RTS sub-round timer to be synchronized with other nodes. This step is crucial for nodes detecting a subsequent event not to interfere with an ongoing round. Furthermore, these nodes are forced to sleep for the entire DATA sub-round if they are not the intended receiver of the data to be sent. When a node receives an RTS packet it checks first if its own RTS sub-round timer has already started. If not, it will update the leader node to the senders ID and start its RTS sub-round timer. The value of the timer is set in accordance to the remaining time of the RTS sub-round timer of the sender. If the RTS sub-round timer has already started, it will check if it should remain the leader. If so, it will add the senders ID to the Schedule and discard timing information. If the node is not the earliest to detect the event, it will update the leader node and its RTS sub-round timer. The details of the RTS sub-round is presented in Algorithm 1

To illustrate the synchronization phase of the RTS sub-round operation, consider a 3 node topology that is shown in Fig. 2. The details of the operations is presented in Fig. 3. In this topology nodes 1 and 3 have detected the event and node 2 is only monitoring the round. Node 1 is the first node to detect the event and consequently it will be the leader of this round. As shown in Fig. 3, the RTS sub-round timer of node 2 is started with the remaining portion of the Leaders RTS sub-round timer minus the time to receive an RTS packet (TTR). TTR also include any processing delays exhibited at the receiver and the sender. These delays can be approximated based on the sensor nodes platform. Node 3 will have to relinquish its leadership status and it will update its timer based on the value received from node 1 minus TTR. Nodes 1 and 3 will keep on broadcasting RTS packets until the RTS sub-round timer expires where both carry the same leader and TTE informa-

tion. All nodes receiving RTS packets will have their timer value expiring at the same time and these nodes will be ready for the subsequent round.

Algorithm 1 EVAM-MAC RTS sub-round operations

```
Duty_Cycle(Node(i))
if EventDetected then
  if RTS_STARTED == FALSE then
    RTS_Timer.start(RTS_ROUND)
    Leader_Node = i
    createRTSPacket(RSS, LEADER_Node,
    Rts_Leader, Recv_Addr, TTE, ID)
    if RTS_Timer.Expired() == FALSE then
      sendRTSTrain()
    end if
  else
    createRTSPacket(RSS, LEADER_Node,
    Rts_Leader, Recv_Addr, TTE, ID)
    if RTS_Timer.Expired() == FALSE then
      sendRTSTrain()
    end if
  end if
end if
if Received(RTS_PACKET) then
  if RTS_STARTED == FALSE then
    {didnot detect the event}
    RTS_Timer.start(RTS_ROUND
    −RTS_Packet.TTE − TTR)
    {TTR: time to receive RTS packet}
    Leader_Node = RTS_Packet.Rts_Leader
    AddNodeToSchedule(RTS_Packet.ID,
    RTS_Packet.Recv_Addr, RTS_Packet.RSS)
    {arrange schedule in decreasing order of RSS, save
    destination address of the data packet}
  else
    TTE_Recv = RTS_ROUND −
    RTS_Packet.TTE − TTR
    if TTE_Recv < TTE then
      RTS_Timer.update(TTE_Recv)
      Leader_Node = RTS_Packet.Rts_Leader
      AddNodeToSchedule(RTS_Packet.ID,
      RTS_Packet.Recv_Addr, RTS_Packet.RSS)
    else
      AddNodeToSchedule(RTS_Packet.ID,
      RTS_Packet.Recv_Addr, RTS_Packet.RSS)
    end if
  end if
end if
```

3.1.2 Dynamic Clustering and Schedule Creation

Most event-based WSN applications are based on cluster topology. This is mainly due to that fact that redundant measurements are collected by the nodes detecting the same event. It is inefficient to broadcast all collected data across the network as it wasted unnecessary bandwidth and exhibits an overhead in energy consumption. Data is usually aggregated on the local cluster head before it is broadcasted to the base station. Clustering is usually static or dynamic. With static clustering, nodes arrange themselves into groups based on the communication range. On the hand, dynamic clustering is achieved by creating a group each time an event is detected. Although static clustering has a lower overhand in creation, dynamic clustering is more efficient. This is mainly because the sensing range is much smaller than the communication range in most applications. Moreover, as the event is traversing the sensor field and arriving at the boundaries of two or more clusters, nodes detecting the event will deliver their data to different cluster heads. This has the effect of increasing congestion and deteriorating the application performance. Moreover, dynamic clustering eliminates single points of failure and minimizes the overhead of nodes joining or leaving the network. In our implementation, the elected leader node acts as a cluster head every time an event is detected.

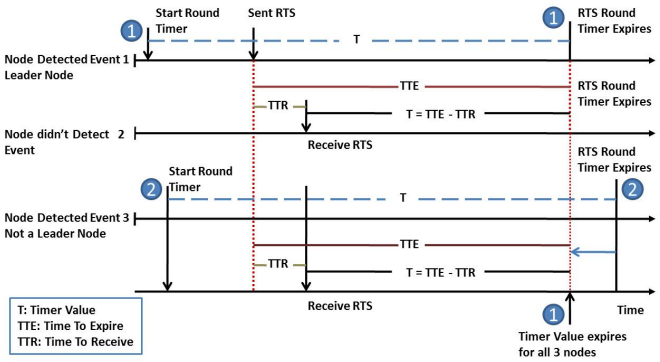


Figure 3: Locally synchronizing nodes and leader selection: All nodes are synchronized with the timer value of the leader at 1

EVAM-MAC achieves dynamic clustering in the RTS sub-round. The cluster is created by the nodes that detected the event. As shown in Algorithm 1, when a node receives an RTS packet, it add the source address of the packet to the virtual cluster. Moreover, it saves the RSS of the event detected and the receiver address. At the end of the RTS sub-round, each node detecting the event will generate the list of nodes that are participating in the event in a decentralized fashion. The schedule for the data sub-round is created by assigning a slot for each node in decreasing order of the RSS. If two nodes have the same RSS values, lower ID nodes will win the slot. Although the leader node

is guaranteed to be unique within all nodes, the neighborhood list may have some variations from node to node depending on the level of congestion. This is mainly because not all sensor nodes will receive all RTS packets. The disambiguation in the schedule between the nodes is resolved in the Leader and Add sub-rounds.

3.2 Leader, Add and Data Sub-rounds

As shown in Fig. 1, four sub-rounds follow the RTS sub-round: Leader sub-round-1, Add sub-round, Leader sub-round-2, and the Data sub-round. The details of these four sub-rounds is presented in Algorithm 2. The purpose of the Leader sub-rounds is to resolve the discrepancies between schedules that were created independently by sensor nodes detecting the same event. This sub-round has a predefined value that is known to all sensor nodes during which all nodes are awake and only the elected leader node is allowed to transmit. The leader node will send out a Leader packet that contains its own created schedule. Each node receiving this packet will discard its created schedule and adopts the leaders schedule. The reason why all nodes create a schedule dynamically is that we dont know a priori the ID of the elected leader. The dynamic nature of EVAM-MAC offers the ability for all nodes to be a leader node provided that they have the lowest time to detect the event. This information is not resolved until the RTS sub-round timer expires.

When nodes receive the Leader packet, they will check if their ID is present in the schedule provided that they have detected the event. If their ID were not found, they have the opportunity to add it in the Add sub-round. Nodes that have their ID present in the schedule are forced to sleep for the entire value of the Add sub-round thus reducing overhearing. The Add packet contains the senders ID and the intended receiver of the data packet. Once the leader receives this packet, it will add the ID of the sender at the end of the schedule. An Acknowledgment packet is sent out to the sender to restrain the node from sending the Add packet again.

As with all sub-round timers, the add sub-round timer is used to synchronize EVAM-MAC sub-rounds. The value of the Add sub-round timer is based on statistics done on the topology of the sensor nodes. EVAM-MAC assign the value of the add sub-round timer to be 10% of the RTS sub-round timer. Although the add sub-round exhibits an overhead on EVAM-MACs round operations, its existence enhances the overall performance. Depending on the level of congestion, some sensor nodes will not have a chance to be included in the schedule especially in dense topologies. These nodes will have to start another round for the same event which exhibits a much larger overhead than the add sub-round. Increasing the value of the RTS sub-round timer will not have the same effect since in that sub-round all sensor nodes will continuously be sending out RTS packets in a high contention environment. In the Add sub-round, only sensor nodes that are not included in the schedule send out the Add packet. Thus, nodes are given the opportunity to send out their missing packets in an environment with less contention.

At the end of the Add sub-round, all nodes will wake in preparation for the Leader sub-round-2. This phase has the same effect and operation of Leader sub-round-1. The appended schedule is sent out to all sensor nodes in the communication reach and each node updates its schedule information. After the second leader phase the data sub-round is initiated. The data sub-round is divided into slots. The slot is granted for nodes that are included in the final schedule. Each node has a unique slot that is guaranteed to be free across the two-hop neighborhood (as shown in the next section). Thus, the total value of the Data sub-round timer varies from round to round depending on the number of nodes detecting the event. Nodes that did not detect the event and are not the intended receiver of any Data packet are forced to sleep for the entire Data sub-round. During each slot only the sending node and the receiving node are awake, thus reducing idle listening. In case of a broadcast all nodes in the cluster will be awake to receive the data. As opposed to TDMA like protocols, there are no wasted slots here. Only nodes that have data to send are included in the schedule which reduces latency. The slot size is computed according to the size of the data packet. Although the medium is considered to be free, an optional ACK packet can be sent during the slot time to ensure delivery. Nodes that did not receive an ACK packet will participate in the next round to send their captured data again.

3.3 Multi-hop resolution

Multi-hop networks poses a threat on the correctness of EVAM-MAC since intersecting cluster as shown in Fig. 4 can introduce ambiguity in schedule information of each cluster. To resolve this issue, only one cluster in the two-hop neighborhood of the leader node is allowed to transmit at a time. Each time a cluster terminates its round, the remaining intersecting clusters contend for the current round at the RTS sub-round as shown later. Usually in event-based systems for wireless sensor networks, the sensing range

is much smaller than the communication range. Thus in real deployments, sensors detecting the event will yield small sparse disconnected clusters with no interference between them. Events occurring in sparse clusters have no effect on the operations of EVAM-MAC. Events occurring interchangeably (i.e. not at the same time) in intersecting clusters, as shown in Fig. 4, also have no effect on the correctness of EVAM-MAC provided one round finishes before the other. Moreover, nodes detecting an event after the RTS sub-round timer of the neighboring cluster expires, will be aware of that round and will not initiate it until the current round is finished.

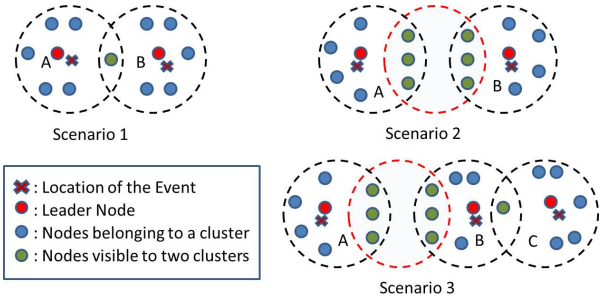


Figure 4: Two hop scenarios

As shown earlier, all nodes detecting the event will broadcast the RTS packet to all neighboring nodes. Nodes belonging to two clusters (as shown in Fig. 4, scenario one) and/or are in the communication range of some nodes in each cluster (as shown in Fig. 4, scenario two) will notify each other by the RTS packets. Consider the case shown in Fig. 4, scenario one. Assume that Cluster A detects the event before Cluster B and that the latter detects the event any time after the RTS sub-round timer of the former expires. The intersecting node will broadcast its RTS packet to all the nodes in Cluster B that is in its communication reach within the duration of the RTS sub-round timer. These nodes will start their RTS sub-round timer upon the reception of the RTS packet from the intersecting node. When the RTS sub-round timer expires, these nodes will wait for the leader node to transmit its Leader packet. However, the leader node is not in communication reach of the nodes in Cluster B. The reason behind this is that nodes that can overhear the leader node will belong to the cluster of the leader which is Cluster A. After the Leader sub-round expires, these nodes will not receive a Leader packet. They will then raise a multi hop flag and will be forced to sleep for the entire duration of the next sub-rounds. However, these nodes have no information about the duration of the Data sub-round since the number of sensors in the schedule of cluster A is unknown to them. In order to relieve any possible interference, the amount of time assigned to the Data sub-round for nodes in Cluster B are given the maximum value which is equal to the time to transmit a DATA packet times the maximum number a cluster can attain. Although this technique introduces delays, it is more efficient than actually forcing the node to contend for the medium as it will introduce interference on the transmitting nodes in Cluster A. Moreover, it will reduce overhearing thus achieving higher energy efficiency. The maximum number any cluster can attain can be obtained before deployment by approximating the number of sensors that are in communication reach of each other. Any event occurring at cluster B after the RTS sub-round timer expires will be buffered until nodes in Cluster B are woken up. After the sleep timer expires, all nodes will repeat the whole operation again.

On the other hand, multiple events can occur simultaneously between neighboring clusters before the RTS sub-round timer expires. In this case all clusters in the communication range of each other contend to win the round. The cluster with the lowest TTE (time to expire) will win this round. If both clusters have equal TTE, the lowest ID leader node will win the round. All other clusters are forced to sleep for the entire round and defer their data transfer to the subsequent round. Consider the case shown in Fig. 4, scenario two. Following the same analogy as before, each node detecting the event will start their RTS sub-round timer and broadcast periodically the RTS packet. Let's assume that Clusters B detects the event before Cluster A. The nodes that are in communication reach of Cluster B will hear the RTS packet and update their leader and TTE information. These nodes will then broadcast their RTS packets piggy-backed with the new information. However, care must be taken to choose an RTS sub-round timer value that is large enough to make this propagation possible. Upon a successful broadcast of an RTS packet of the leader node at Cluster B, any of the boundary nodes will notify Cluster A with the existence of the other round by also broadcasting a RTS packet. As discussed earlier, the cluster will shut down for the entire round for the duration of the following sub-rounds where the Data sub-round has the maximum value. Nodes belonging to Cluster A will commence their round operation after the sleep timer expires.

Algorithm 2 EVAM-MAC Leader, Add, Data sub-rounds operations

```
if RTS_Timer.Expired() == TRUE then
  Leader1_Timer.Start()
  if IsLeader() then
    createLeaderPacket(Schedule[0, 1, ..., n])
    sendLeaderPacket()
  end if
end if
if Leader1_Timer.Expired() == TRUE then
  AddTimer.start()
  if CheckNodeID(Schedule) == TRUE then
    sleep()
  else
    createAddPacket(ID, Recv_Addr)
    while ACKNotReceived() do
      sendADDPacket(LeaderNode)
    end while
  end if
end if
if Add_Timer.Expired() == TRUE then
  wakeup()
  Leader2_Timer.Start()
  if IsLeader() then
    createLeaderPacket(Schedule[0, 1, ..., n
    +appenedNodes])
    sendLeaderPacket()
  end if
end if
if Leader2_Timer.Expired() == TRUE then
  CheckSchedule()
  if CheckNodeID(Schedule) == FALSE then
    sleep()
  else
    wakeup(SlotID)
    if IsSender() then
      sendData()
    end if
  end if
end if
if Received.LeaderPkt() then
  Schedule = LeaderPkt.Schedule
else if Received.AddPkt() then
  Schedule.append(AddPkt.ID)
end if
```

In the case where we have more than two adjacent cluster as shown in Fig. 4, scenario three, both Cluster A and C are forced to sleep provided that Cluster C detected the event after Cluster B. Consider the case where Cluster A detect the event first then Cluster C and lastly Cluster B. Provided enough time is given for the RTS sub-round timer, Cluster C will also be forced to sleep. However, Cluster C can commence a round in parallel with Cluster A with no interference since Cluster B is forced to sleep. A mechanism must be devised to notify Cluster C that it can commence its round operations. In this context, a bit (RTS_Leader) is piggybacked with the RTS packet that is set if the RTS packet was received from the actual Leader node and unset if the packet was forwarded from another node that is not the Leader node. However, if a node received a RTS packet from the Leader node once, it will always set the RTS_Leader bit disregarding any forwarded packets with the same Leader address and TTE information. Cluster B will forward the RTS packets to Cluster C with the RTS_Leader bit unset. Upon the reception of the forwarded RTS packet, Cluster C will detect that the RTS_Leader bit is unset and will disregard the Leader and TTE information and commence with its round. Thus the two Clusters (A and C) can operate in parallel. The disadvantage of this technique is that if Cluster B detects the event before Cluster C and the RTS packets from Cluster A didn't have enough time to reach Cluster C, Cluster C has no way of knowing the existence of Cluster A and assumes that its Cluster B's turn. Thus, it has to wait for the next round to operate.

3.4 EVAM-MAC Programmable features

The dynamic nature of EVAM-MAC makes it customizable for the applications needs and requirements. Programmable feature is an option that is implemented in EVAM-MAC that allows customizable Data sub-rounds. These options can be set by the application layer at deployment. The default operation of EVAM-MAC is to unicast the data packet to the elected cluster head which will forward them to the sink. However, some applications require several rounds of communications provided by the same sensor nodes detecting the event. These rounds can be a mixture of broadcast and unicast. In target localization applications, sensor nodes usually broadcast the collected measurement to the all sensor nodes detecting the target for processing. Once a location is computed, each sensor node will unicast the results back to the cluster head for aggregation. These operations can be programmed in EVAM-MAC where each round consists of 2 data sub-rounds, one for broadcast and the other for unicast. Multiple data rounds can be added depending on the applications needs. In this case the same schedule is used over again with the desired communication pattern. The overhead for multiple rounds is much lower than those of a single round with respect to the amount of Data transfer since the amount of control packet exchanged stayed the same. Thus, by lowering the control overhead

that is required to construct the data sub-rounds, and giving the application designers the freedom to choose their communication patterns and operations, optimizations can be achieved in throughput, energy consumption and control overhead.

Restraining sensor nodes transmissions is another programmable feature that is resourceful in WSN application. In dense deployments, sensor nodes collect several measurements for the same event that are highly correlated. Some applications require only a subset of sensor nodes detecting a certain type of event to transmit its collected data. For an example, consider a 15-node cluster monitoring a certain event. The application layer can be satisfied with only 5-high quality measurements. The number of measurements required and the criteria used to indicate the quality of the measurements is based on the application. EVAM-MAC arranges data schedule based on RSS of the event detected. If this feature is enabled, the Leader node will broadcast a schedule containing only the required sensor nodes disregarding the rest. Upon the reception of the Leader packet, the sensor nodes will check if their ID is present in the schedule. When they realize that they are off the schedule, they force themselves to sleep for the entire Data sub-rounds. This way we achieve higher energy conservation by prohibiting all remaining sensors from transmitting their redundant Data packets.

4 Experimentation and Simulation Results

EVAM-MAC was implemented and simulated using Network simulator (NS-2 [10]). Furthermore, EVAM-MAC is compared with an NS-2 implementation of S-MAC [2] and 802.11 [11], for one-hop, two-hop and multi-hop benchmarks. As shown in Table I, the communication bandwidth used for all benchmarks was 19.2kbs and the DATA packet payload was 60 Bytes. The RTS sub-round timer was assigned a value of $200ms$ for one-hop networks and $300ms$ for multi-hop networks since in multi-hop networks more time is required at the RTS sub-round phase to propagate the RTS packet to neighboring clusters. The Leader and Add phases are fixed to $11ms$ and $22ms$. All sub-round values depend on the communication bandwidth. Higher bandwidth values will yield smaller sub-round values. Table II presents a comparison of the control packet sizes for all three protocols. The Leader packet of EVAM-MAC has a large value of 27 bytes since it contains the piggybacked schedule. However, the Leader packet is only broadcasted twice per round.

Three metrics were used to compare EVAM-MAC's performance with other protocols: net throughput, average energy consumption, and percentage of control packet overhead for all three benchmarks. The net throughput is computed by counting the number of DATA packets successfully received by the sink where only the payload is considered. The average energy consumption is computed by averaging the energy consumed by all nodes in the network and dividing it by the total sensor count for the duration of the simulation run. The percentage of control packet overhead is computed by averaging the number of control packet bytes sent for all nodes in the network with respect to the number of data bytes successfully received by the sink. For all three metrics, the setup phase of S-MAC was not included in the computation since it is an overhead that occurs only at deployment. However, the sync messages used after the nodes are synchronized are considered in the computations.

Table 1: Default Parmeters for EVAM-MAC protocol

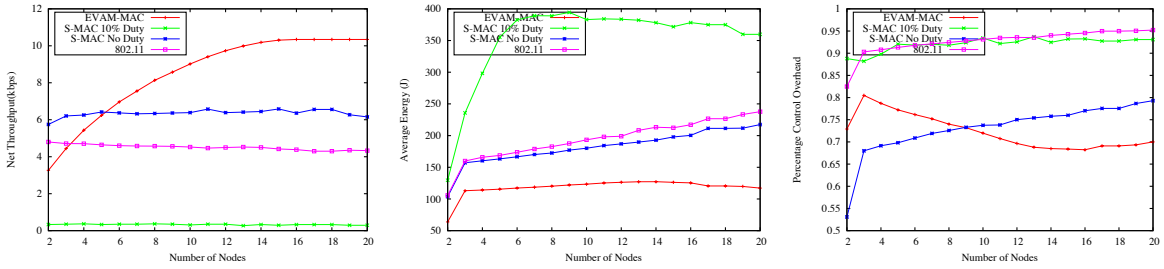
RTS-subround one-hop	200 ms
RTS-subround multi-hop	300 ms
Leader sub-round	11 ms
ADD sub-round	22 ms
DATA sub-round	Variable
Data Payload	60 Bytes
Communication Bandwidth	19.2 kbs

Table 2: Packet Size Information for Differnet Protocols

Type	**EVAM-MAC**	**S-MAC**	**802.11**
RTS	16 Bytes	10 Bytes	44 Bytes
CTS	NA	10 Bytes	38 Bytes
Leader	27 Bytes	NA	NA
ADD	5 Bytes	NA	NA
Sync	NA	10 Bytes	NA
ACK	5 Bytes	10 Bytes	38 Bytes

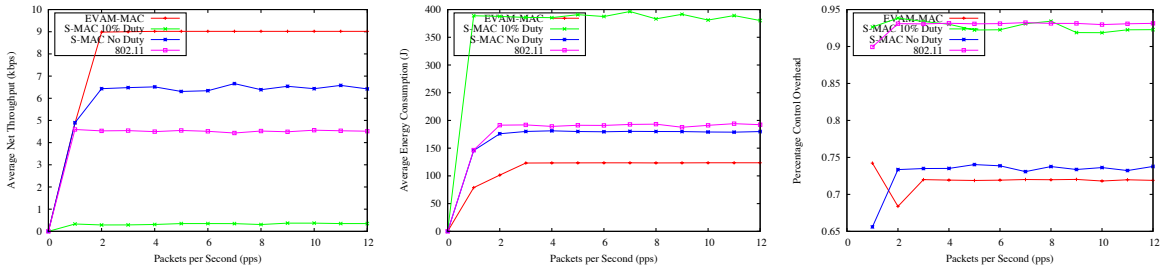
4.1 One-hop Benchmarks

The one-hop benchmarks consist of 20 nodes which are all in the communication range of each other and of a sink. Two applications were employed for this evaluation. The first application is a UDP (User Datagram Protocol) agent with a CBR (constant bit



(a) Throughput wrt sensor count (b) Avg. eng. wrt sensor count (c) % of all control pkts sent

Figure 5: One-hop simulations for nodes with CBR and UDP agents Traffic rate = 8 pps



(a) Throughput wrt sensor count (b) Avg. eng. wrt sensor count (c) % of all control pkts sent

Figure 6: One-hop simulations for nodes with CBR and UDP agents Number of nodes = 10

rate) traffic generator. All nodes generate traffic at approximately similar time and unicast their data directly to the sink. Two simulation experiments were performed. In the first experiment, the sensor count is increased from 2 to 20 while the data traffic is kept constant to a value of 8 packets per second. As shown in Fig. 5a, the net throughput is compared for different MAC protocols as a function of sensor count. EVAM-MAC produces an exponential growth from sensor count 2 to 15 while it saturates around 10kbs. S-MAC was operated with a 10% duty cycle where its performance is the worst since the overhead of synchronization deteriorates the throughput severely. S-MAC with no duty cycle and 802.11 achieve similar performance. EVAM-MAC achieves higher throughput than all other protocols when the sensor count increase above 5. This is because the RTS sub-round timer is fixed to a value taking into account that 20 nodes may have data to send at the same time. When 5 nodes or less are transmitting, the RTS sub-round timer is large enough to lowers the throughput. However, EVAM-MAC is tailored for high sensor count and traffic application. Moreover, as shown in Fig. 5b, EVAM-MAC has the lowest energy consumed with respect to all MAC protocols. Fig. 5c, shows the average percentage of control packets sent by each MAC protocol of all nodes with respect to all packets sent. EVAM-MAC has the lowest control overhead as sensor count increases above 9 nodes. This is due to the fact that the percentage of control bytes sent stays constant since the RTS sub-round timer is fixed while the number of DATA bytes increases. In the second experiment we fixed the number of nodes to 10 and we varied the number of packets sent per seconds from 1pps to 12pps. This experiment is performed to evaluate EVAM-MAC from low to high traffic conditions. As shown in Fig. 6a, the net throughput of EVAM-MAC is superior to all other protocol when the traffic generated by each node is equal or greater to 2pps. EVAM-MAC is also superior to all protocols compared in terms of energy consumed as shown in Fig. 6b, and percentage of control overhead sent shown in Fig. 6c. This proves the effectiveness of EVAM-MAC in both high sensor count and high traffic application in single hop networks.

The second application is a simulated event based system where nodes detecting an event are required to broadcast their Data packets to all nodes. After each node receives a data packet from every other node detecting the event it will unicast a Data packet to the sink. However, the number of nodes detecting the event is required to be known so that nodes receiving the broadcast information know when to unicast its Data packet. For EVAM-MAC, the number of nodes detecting the event can be known from the collected

RTS packets. However, for S-MAC protocol, nodes will have no a priori information about the number of nodes detecting the event. In this scenario, we assume a fixed number of nodes that is supplied to them at deployment. In real deployment, S-MAC will have to exchange some packets to know the number of nodes detecting the event. This will incur an overhead in energy consumption and will increase latency. EVAM-MAC uses the two round programmable feature that was discussed in Section 3.4. As shown in Fig. 6a, the net throughput is compared as the number of nodes detecting the event increases. EVAM-MAC throughput growth is exponential which reaches a value of 12kbs. EVAM-MAC's performance is superior to S-MAC when the sensor count increase above 5. Furthermore, as shown in Fig. 6b and Fig. 6c, EVAM-MAC has similar energy consumption values and control packet overhead to that of SMAC with 10% duty cycle. Thus, EVAM-MAC can achieve nearly double the throughput of S-MAC with no duty cycle with the same value of energy spent and control packet sent of S-MAC with 10% duty cycle.

4.2 Two-hop Benchmarks

To prove the correctness of EVAM-MAC for two hop scenarios, we simulated the proposed algorithm for scenarios 1 and 2 shown in Fig. 4. In both scenarios, detected event forms two clusters simultaneously where the maximum number of nodes in each cluster is equal to 12. In scenario 1, one node is in communication range of both clusters. In scenario 2, three nodes are in the interference range of both clusters. Each cluster has a sink that is in communication reach of all nodes belonging to that cluster. When an event is detecting all nodes will unicast their Data packets to the sink of the corresponding cluster. As shown in Fig. 7a, d, the net throughput is compared as the number of sensors detecting the event increases at both clusters. All protocols show a linear increase as the number of sensors detecting the event increases. However, EVAM-MAC presents superiority when the sensor count is around 16, which translates to 8 sensor nodes per cluster. The reason behind this is that EVAM-MAC shuts down the whole cluster while the other cluster is delivering data. As the number of sensor nodes in each cluster reaches 8, it becomes worthwhile to shut down the cluster as the contention produced from two neighboring clusters introduces latency. In Fig. 7b, e, the average energy consumed of EVAM-MAC has close performance with respect to S-MAC with no duty cycle. 802.11 has the worst energy consumed with respect to all compared protocols. Moreover, EVAM-MAC has greater control packet overhead with respect to S-MAC (no duty cycle) as shown in Fig. 7c, where 802.11 has also the worst performance. However, this overhead is compensated for by the reduced number of dropped data packet which is evident in Fig. 7b,e.

4.3 Multi-hop Benchmarks

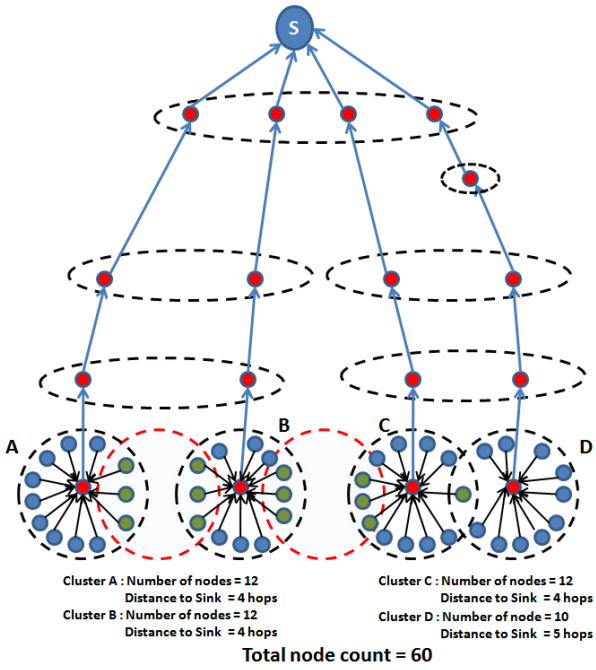
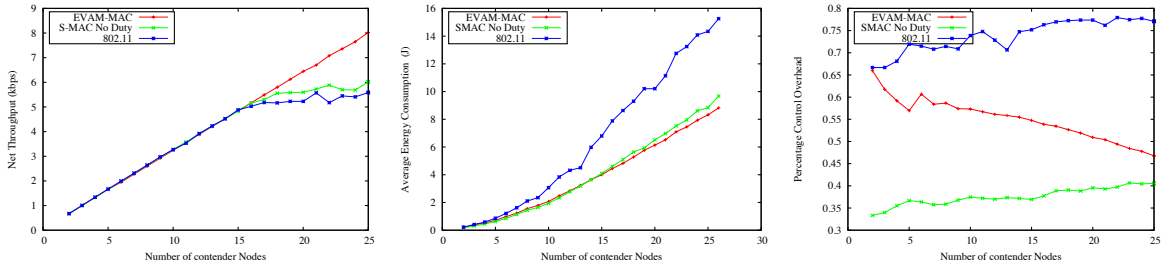
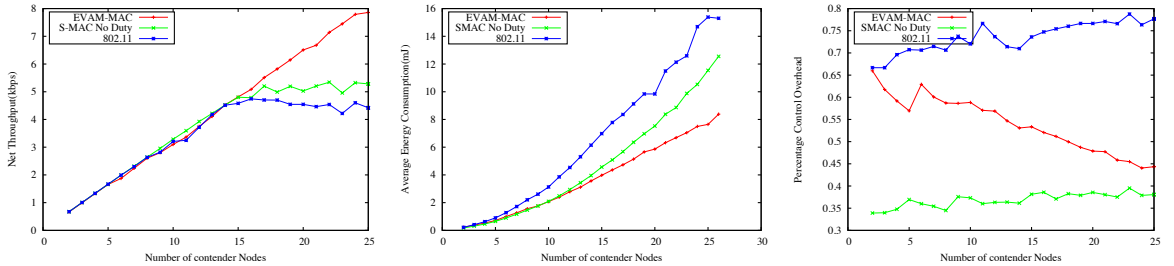


Figure 8: Multihop Layout

To evaluate EVAM-MAC in a multi-hop scenario, the proposed algorithm was simulated in a 60 node network as shown in Fig. 8. The maximum number of nodes detecting the event simultaneously is equal to 50. Four clusters exist as shown in Fig. 8 where cluster A, B and C are 4 hops away from the sink and each contain 12 nodes. Cluster D is 5 hops away from the sink and contains 10 nodes. To eliminate the overhead of routing, fixed routing tables were supplied to the nodes. Each node detecting the event will unicast its data to the local sink. Upon the reception of all data packet for all nodes detecting the event, the local sink will forward the aggregated Data packet through several hops until it reaches the final sink as shown in Fig. 8. Four metrics were used for this simulation: net throughput, average energy consumed, average control packet overhead and average latency. The average latency is the average time that all nodes take to transmit their packet to the final sink. As shown in Fig. 9a, we evaluated EVAM-MAC's throughput as a function of the number of

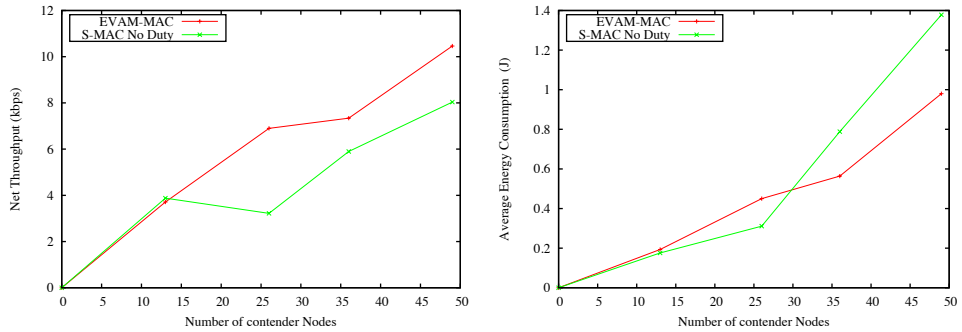


(a) Throughput wrt sensor count (scenario 1) (b) Avg energy consumed wrt sensor count (scenario 1) (c) Percentage of all control pkts sent (scenario 1)

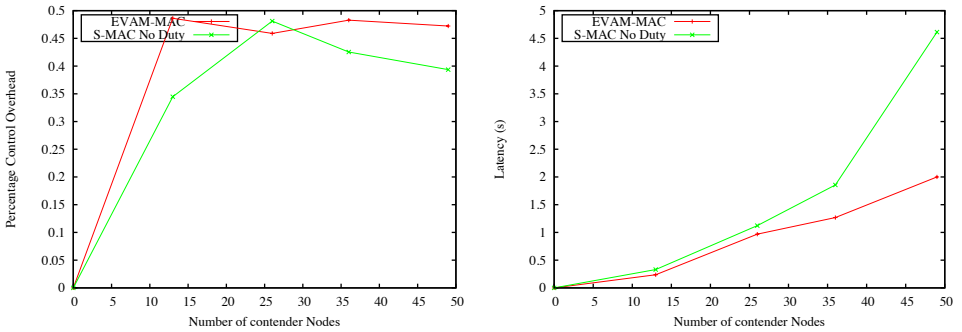


(d) Throughput wrt sensor count (scenario 2) (e) Avg energy consumed wrt sensor count (scenario 2) (f) Percentage of all control pkts sent (scenario 2)

Figure 7: Two hop simulations for an event based system



(a) Throughput in multi-hop Scenario (b) Average Energy Consumption



(c) Percentage of control overhead (d) Total Latency in (s)

Figure 9: Multihop Simulation

sensors detecting the event. EVAM-MAC achieves higher throughput than S-MAC with no duty cycle as the number of sensors detecting the event grows above 15. We didn't compare our protocols with S-MAC with duty cycle since it is not scalable to this amount of nodes. Fig. 9b presents the average energy consumed by all nodes as sensor count increases. Initially, EVAM-MAC has similar performance as that of S-MAC with no duty cycle. However, when the sensor count increases above 25 EVAM-MAC has lower energy consumption. On the other hand, the control packet overhead of EVAM-MAC is higher than that of S-MAC as shown in Fig. 9c. However, this overhead didn't reflect negatively on energy consumption since the percentage of dropped packets is greatly reduced with EVAM-MAC. Moreover, EVAM-MAC has lower latency than S-MAC as shown in Fig. 9d. When the number of sensors detecting the event reaches 50, EVAM-MAC has a 3 to 1 advantage over S-MAC in latency. Thus, EVAM-MAC achieves lower latency and higher throughput without introducing a big overhead in energy consumption and control packets sent. EVAM-MAC is very desirable in event based systems as it reliefs contention from multiple nodes contending for the Data packet delivery.

5 Conclusion and Future Work

In this work a novel MAC protocol tailored for event based systems for wireless sensor networks is presented. EVAM-MAC creates a contention free schedule for the collected measurements for transmission in a dynamic decentralized fashion. EVAM-MAC does not require any global synchronization or setup phase at deployment. This has the advantage of eliminating the overhead of nodes joining and leaving the network. EVAM-MAC shifts the contention generated by multiple sensor nodes detecting an event from the Data phase to the control phase where packet sizes are much smaller in size. Furthermore, EVAM-MAC provides a rich platform of programmable features that the application designers can utilize to achieve high performance operations. EVAM-MAC presented its superiority in throughput and energy conservation for single and multi hop scenarios when compared with S-MAC and 802.11.

Our future work includes performing a real wireless sensor implementation on wireless sensor motes. EVAM-MAC will be tested on target localization application for wireless sensor networks. Furthermore, more simulations and experimentation with different topologies and higher sensor count is being performed.

References

[1] P. Suriyachai, U. Roedig, A. Scott,"A Survey of MAC Protocols for Mission-Critical Applications in Wireless Sensor Networks", In IEEE Communications Surveys & Tutorials , vol.14, no.2, pp.240-264, Second Quarter 2012, doi: 10.1109/SURV.2011.020211.00036

[2] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient Mac Protocol for Wireless Sensor Networks", Proc. IEEE Conf. Computer and Communications Societies (INFOCOM 2002), vol. 3, pp. 1567-1567, Jun. 2002, doi: 10.1109/INFCOM.2002.1019408.

[3] T.V. Dam and K. Langendoen, "An Adaptive Energy-Efficient Mac Protocol for Wireless Sensor Networks", Proc. ACM Conf. Embedded Networked Sensor Systems (SENSYS '03), pp. 171-180, Nov. 2003, doi: 10.1145/958491.958512.

[4] J. Polastre, J. Hill, D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks", Proc. ACM Conf. Embedded Networked Sensor Systems (SENSYS '04), pp. 95-107, Nov. 2004, doi: 10.1145/1031495.1031508.

[5] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Trans. Netw., 12(3):493-506, 2004. doi: 10.1109/TNET.2004.828953

[6] Shams ur Rahman, Mushtaq Ahmad, Shafaat A. Bazaz, "A new Energy-Efficient TDMA-based MAC Protocol for Periodic Sensing Applications in Wireless Sensor Networks", In IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012.

[7] I. Rhee et al., " Z-MAC: a hybrid MAC for wireless sensor networks", Proc. ACM Conf. Embedded Networked Sensor Systems (SENSYS '05), pp. 90-101, Nov. 2005, doi: 1098918.1098929.

[8] Z. Merhi, M. Elgamel, and M. Bayoumi, "EVA-MAC:An EVent based Medium Access Control for Wireless Sensor Networks", Proc. of the IEEE Workshop on Signal Processing Systems (SiPs2011), pp. 61-66, Oct. 2011, doi:10.1109/SiPS.2011.6088950

[9] Z. Merhi, M. Elgamel, and M. Bayoumi, "A Lightweight Collaborative Fault Tolerant Target Localization System for Wireless Sensor Net-

works," IEEE Transactions on Mobile Computing, 29 Apr. 2009. IEEE computer Society Digital Library. IEEE Computer Society, doi: 10.1109/TMC.2009.81

[10] The Network Simulator NS-2, http://www.isi.edu/nsnam/ns/, 2009.

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

[12] G. Ahn et al., " Funneling-MAC: a localized, sink-oriented MAC for boosting fidelity in sensor networks", Proc. ACM Conf. Embedded Networked Sensor Systems (SENSYS '06), pp. 293-306, Nov. 2006, doi: 10.1145/1182807.1182837.

[13] I. Rhee, A. Warrier and L. Xu, " Randomized dining philosophers to TDMA scheduling in wireless sensor networks" Technical report, Computer Science Department, North Carolina State University, Raleigh, NC, 2004.

[14] W. B. Heinzelman , A. P. Chandrakasan, and H. Balakrishman, An application-specific protocol architecture for wireless microsensor networks, IEEE Trans. Wireless Communication, vol. 1, pp. 660670, Oct. 2002, doi:10.1109/TWC.2002.804190.

[15] Chunyao FU et. al, "An Energy Balanced Algorithm of LEACH Protocol in WSN" in IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No 1, January 2013.

[16] J. Polastre, J. Hill, D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks", Proc. ACM Conf. Embedded Networked Sensor Systems (SENSYS '04), pp. 95-107, Nov. 2004, doi: 10.1145/1031495.1031508.

[17] M. Buettner et al.," X-MAC: A Short Preamble MAC Protocol For Duty-Cycled Wireless Sensor Networks", In Proc. of 5th ACM Conference on Embedded Networked Sensor Systems (SenSys 2006), November 2006.

[18] W. Ye, F. Silva, and J. Heidemann," Ultra-Low Duty Cycle MAC with Scheduled Channel Polling" In Proc. of 5th ACM Conference on Embedded Networked Sensor Systems (SenSys 2006), November 2006.

[19] Yanjun Sun, Omer Gurewitz, and David B. Johnson. "RI-MAC: A receiver initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks". In Proc. of 6th ACM Conference on Embedded Networked Sensor Systems (SenSys'08), 2008.doi:10.1145/1460412.1460414

[20] Lei Tang, Yanjun Sun, O. Gurewitz, D.B. Johnson, "PW-MAC: An energy-efficient predictive-wakeup MAC protocol for wireless sensor networks," In Proc. of the 30th IEEE International Conference on Computer Communications ((INFOCOM 2011), pp.1305-1313, 10-15 April 2011, doi: 10.1109/INFCOM.2011.5934913

[21] Beakcheol Janga, Jun Bum Limb, Mihail L. Sichitiuc, "An asynchronous scheduled MAC protocol for wireless sensor networks" In Elsevier Journal on Computer Networks, 2012

Zaher M. Merhi received the B.Sc. degree and the M.Sc. degree in computer engineering from the University of Balamand, Lebanon, in 2002 and 2005. He also received the M.Sc. and the PhD degree in computer engineering from the University of Louisiana at Lafayette, USA, in 2007 and 2010. Since 2010 he joined the Lebanese International University (LIU) as an assistant professor at the computer and communication engineering (CCE) department. Currently he is the associate chairperson of the CCE department at LIU. His research interests include low power embedded systems, distributed DSP applications for wireless sensor networks, data fusion techniques and medium access controls (MAC) for wireless sensor networks.

Mohamed A. Elgamel received the B.Sc. degree in computer science from Alexandria University, Alexandria, Egypt, in 1991, the M.Sc. degree in computer engineering from Arab Academy for Science and Technology (AAST), Alexandria, Egypt, in 1998, the M.Sc. and the PhD degrees in computer engineering from University of Louisiana at Lafayette, Louisiana, USA, in 2000 and 2003 respectively. He is currently an Associate Professor at the Faculty of Computing and Information Technology, AAST. In 2013, he was a visiting professor at The International Telematic University (UNINETTUNO), Rome, Italy. From 2003-2010, he has been a Graduate Faculty at The Center for Advanced Computer Studies, University of Louisiana at Lafayette, USA. From 1999 to 2003, he served as a Research Assistant and Adjunct Faculty at the Center for Advanced Computer Studies, University of Louisiana at Lafayette, USA. From 1991 to 1999, he was a Senior Analyst Programmer

at Arab Academy for Science and Technology, Egypt. His research interests include CAD tools, Embedded Systems, Wireless Sensors Networks, Software Engineering, and digital video processing. His research was supported by funds from federal agents like DOE, NSF and the Louisiana Governor ITI. Dr. Elgamel is a senior IEEE member. He is also a member on the IEEE Technical Committee on Communication and IEEE-TC on VLSI. He served on the program committees and chaired sessions of several conferences and as a reviewer/Reviewer Committee Member for several conferences and journals. Dr. Elgamel received the 2002 Richard E. Merwin Student Scholarship for demonstrating outstanding involvement in an IEEE-CS and excellence in academic achievement. He was the first place winner in the student paper contest in the 46th IEEE International Midwest Symposium on Circuits and Systems, (MWSCAS2003). He was the president of the IEEE Computer Society student chapter at the University of Louisiana at Lafayette.

Samih Abdul-Nabi received his B.E. degree in Electrical Engineering in 1990 from the Lebanese University, Beirut, Lebanon and his M.S. degree in Computer Sciences in 1993 from the University of Montreal, Canada. Since 1993, Mr Abdul-Nabi worked in many related engineering fields in both Canada and the U.S. His last industrial assignment was at Diebold Inc. as Senior Systems Engineer providing technical support to one of the largest banks in Canada. His is currently a lecturer at the Lebanese International University working on some research topics that includes Network Coding and encryption.

Amin A. Haj-Ali received his B.E. degree in Computer and Communications Engineering in 1993 from the American University of Beirut, Beirut, Lebanon, his M.S. degree in Electronics and Computer Control Systems in 1994, and his Ph.D. in Electrical Engineering (Systems and Soft Computing) and Computer Science (Minor) in 2002 from Wayne State University, Detroit, USA. Dr. Haj-Ali earned a Masters in IT Project Management from George Washington University, Washington DC, USA. Since 1996, Dr. Haj-Ali worked in many related engineering fields in both Lebanese and international corporations. His last industrial assignment was at DaimlerChrysler Corporation Scientific Labs and Proving Grounds, Auburn Hills, MI USA where has registered a patent (US07043963B2). Currently Dr. Haj-Ali is the Associate Dean of the School of Engineering and Associate Professor at the Lebanese international University. His research interest includes Machine Vision, Embedded Systems, Fuzzy Systems and Neural Networks.

Magdy A. Bayoumi received the B.Sc. and M.Sc. degrees in electrical engineering from Cairo University, Cairo, Egypt, the M.Sc. degree in computer engineering from Washington University, St. Louis, and the Ph.D. degree in electrical engineering from the University of Windsor, Windsor, ON, Canada. He is the Director of the Center for Advanced Computer Studies (CACS) and Department Head of Computer Science Department, University of Louisiana, Lafayette. He is also the Z.L. "Zeke" Loflin Eminent Scholar Endowed Chair at the Center for Advanced Computer Studies, University of Louisiana, Lafayette, where he has been a faculty member since 1985. His research interests include VLSI design methods and architectures, low power circuits and systems, digital signal processing architectures, parallel algorithm design, computer arithmetic, image and video signal processing, neural networks, and wideband network architectures Dr. Bayoumi was the vice president for the technical activities of the IEEE Circuits and Systems Society and the chairman of the Technical Committee on Circuits and Systems for Communication and the TC on Signal Processing Design and Implementation. He was a founding member of the VLSI Systems and Applications Technical Committee and was its chairman. He is a member of the Neural Network and the Multimedia Technology Technical Committees. He was an Associate Editor of the Circuits and Devices Magazine, the IEEE Transactions on Very Large Scale Integration (VLSI) Systems, the IEEE Transaction on Neural Networks, and the IEEE Transaction on Circuit and Systems II: Analog and Digital Signal Processing and Integration. He was also the chairman of many conferences and workshops including MWSCAS'94, GLSVLSI'98, MWSCAS'03, ISCAS'07, and ICIP'09.