Energy-Efficient, Application-Aware Medium Access for Sensor Networks

Venkatesh Rajendran[*]
Email: venkat@soe.ucsc.edu
[*]Computer Engineering Department
University of California at Santa Cruz
Santa Cruz, CA 95064.

J. J. Garcia-Luna-Aceves[*†]
Email: jj@cse.ucsc.edu
[†]Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304.

Katia Obraczka[*]
Email: katia@cse.ucsc.edu

***Abstract*— We introduce FLAMA (FLow-Aware Medium Access), an energy-efficient medium-access control (MAC) protocol designed for wireless sensor networks. FLAMA achieves energy efficiency by preventing idle listening, data collisions and transmissions to a node that is not ready to receive packets. It adapts medium access schedules to the traffic flows exhibited by the application. FLAMA is simple enough so that it can be run by nodes with limited processing, memory, communication, and power capabilities. We evaluate the performance of FLAMA through simulations and test-bed experimentation. Simulation results indicate that, in terms of reliability, queuing delay and energy savings, FLAMA outperforms TRAMA, the first traffic-adaptive, schedule-based MAC proposed for sensor networks, and S-MAC, a contention-based energy-efficient MAC. FLAMA achieves significantly smaller delays (up to 75 times) when compared to TRAMA with significant improvement in energy savings and reliability, demonstrating the importance of application-awareness in medium access scheduling. Our simulation and test-bed results show that FLAMA achieves better end-to-end reliability with significant energy savings compared to S-MAC.**

I. Introduction

Sensor networks typically refer to arbitrarily large ensembles of interconnected sensing devices with limited processing, communication, and power capabilities. Self-organization, robustness, and energy efficiency are important goals in these networks because the deployment is often done in an ad hoc manner, nodes are battery powered (re-charging them may be either impossible or not cost-effective), and topology changes (e.g., additional nodes may be deployed or nodes may fail) are likely to occur.

Most existing energy-efficient MAC protocols for sensor networks have employed a contention-based approach. A notable example is the Sensor-MAC (or S-MAC) protocol [1]. The main drawback of contention-based MAC protocols is that the probability of collisions increases with the offered load, which degrades channel utilization and wastes energy. This motivates our research into distributed schedule-based medium access methods. Section II summarizes prior work in energy-efficient channel access based on contention schemes and scheduling schemes.

[1]This work was supported in part by the UCOP CLC under grant SC-05-33, by the Army Research Office under grant No. W911NF-05-1-0246, and by the Baskin Chair of Computer Engineering at the University of California, Santa Cruz. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the funding agencies.

In this paper, we introduce the FLow-Aware Medium Access (FLAMA) protocol, a schedule-based MAC protocol that leverages traffic predictability in sensor network applications. Traffic information can be determined by having the application explicitly specify its traffic characteristics, or by using traffic prediction techniques at each node. Depending on the application at hand, traffic prediction can be relatively simple. For instance, periodic data gathering (e.g., environmental monitoring) generates data streams over a collection tree rooted at the information sink and spanning all relevant nodes. When sending data, each node transmits to the upstream next-hop towards the sink. This information could be used to determine the next-hop node for a node's transmission.

Section III describes FLAMA in detail. FLAMA uses the concept of *flows* to characterize application traffic patterns. Flows represent one-hop traffic information and specify the transmitter, the receiver(s), and the rate at which packets are sent. FLAMA uses flow-based traffic information to determine transmission schedules, as well as when nodes should be in receive mode or can switch to low-power sleep state. Its main features are: (a) the distributed maintenance of energy-efficient, collision-free transmission schedules based on two-hop neighborhood information and implicit traffic information, (b) low transmission delays with limited processing and storage requirements, and (c) robust operation that accommodates topology changes.

We evaluate the performance of FLAMA through simulations and test-bed experimentation. Section IV presents simulation results comparing the performance of FLAMA against two other MAC protocols. We uses the QualNet network simulator [2] for our simulation experiments, and compared the performance of FLAMA against TRAMA [3], an existing schedule-based MAC, and S-MAC. The results from our simulation study show that FLAMA achieves significantly lesser delay (up to 75 times) when compared to TRAMA, with significant improvement in energy savings and reliability when compared to TRAMA and S-MAC, demonstrating the importance of application-awareness in medium access scheduling.

Section IV-D describes our implementation of FLAMA on TinyOS [4] for the Mica2 Motes platform [5] and presents experimental results comparing FLAMA and S-MAC in a sensor network test-bed. The results of our experiments show that FLAMA achieves 100% delivery compared to 75% for S-MAC at low offered loads (which are scenarios that favor contention-based MACs) and the average service time for FLAMA is an order of magnitude less than that of S-MAC. Finally, Section V concludes the paper with directions for future work.

II. Related Work

PAMAS [6] is one of the earliest contention-based proposals to address power efficiency in channel access. PAMAS saves energy by attempting to avoid over-hearing among neighboring nodes. To achieve this, PAMAS uses out-of channel signaling. Woo and Culler [7] address variations of CSMA tailored for sensor networks, and propose an adaptive rate control mechanism to achieve fair bandwidth allocation among sensor network nodes. In the power save (PS) mode in IEEE 802.11 DCF, nodes sleep periodically. Tseng *et al.* [8] investigated three sleep modalities in 802.11 DCF in multi-hop networks. The sensor-MAC protocol [1], or S-MAC, exhibits similar functionality to that of PAMAS and the protocol by Tseng *et al.*. Like the other approaches, S-MAC avoids overhearing and nodes periodically sleep. However, unlike PAMAS, S-MAC uses in-line signaling, and unlike modalities of the PC mode in 802.11 DCF, neighboring nodes can synchronize their sleep schedules. T-MAC [9] is an improvement over S-MAC that adapts the duty cycle based on the traffic. However, synchronized listen periods increases the channel contention significantly for S-MAC and T-MAC and also increases the overall noise floor during transmissions leading to degradation in link quality. D-MAC [10] is a medium access protocol designed specifically for data gathering applications using unidirectional trees. It schedules transmissions at each hop so that the latency in data collection is reduced. However, D-MAC assumes fixed topology and does not allow multiple data gathering trees. It cannot adapt to other sensor network applications.

The transmission schedule established in a wireless network can be topology independent or topology dependent [11]–[14]. The schedule-access MAC protocol described by Sohrabi and Pottie [15] uses a combination of TDMA and FDMA or CDMA for accessing the channel. The main drawback of this scheme is that, like most fixed scheduling mechanisms, time slots are wasted if a node does not have any data to send to the intended receiver.

The Traffic-Adaptive Medium Access (TRAMA) protocol [3] was the first proposal to implement energy-aware schedule-based medium access. TRAMA addresses energy efficiency by having nodes going into sleep mode if they are not selected to transmit and are not the intended receivers of traffic during a particular time slot. TRAMA uses traffic information to establish transmission schedules which are propagated to one-hop neighbors. This information is then used to define when nodes need to be in receive mode and when they can switch to low-power sleep mode. Besides its energy efficiency benefits, the use of traffic information also makes TRAMA adaptive to the sensor network application at hand. However, TRAMA's adaptiveness comes at a price, namely the complexity of its election algorithm and scheduling overhead for announcing traffic information. Schedule-based protocols exhibit inherently higher delivery delays when compared to contention-based approaches. In TRAMA, this is exacerbated by the need to propagate schedule information. FLAMA avoids explicit traffic information exchange and employs a much simpler election algorithm than TRAMA.

III. Flow-Aware Medium Access

FLAMA uses a simple traffic adaptive, distributed election scheme for energy-efficient channel access. It requires two-hop neighborhood and flow information in the neighborhood to perform the election. Using only two-hop neighborhood information makes FLAMA scalable. Time is organized in periods of random- and scheduled-access intervals as shown in Figure 1. We assume a single channel for data and signaling; however, FLAMA can be easily extended to handle multiple channels. Channel access is contention-based during random-access and time-slotted during scheduled-access periods. During random access, neighbor discovery, time synchronization and implicit traffic information exchange are performed. Data transmission happens during scheduled access. Using periodic random-access periods allows FLAMA to adapt to topology and traffic changes in the network.

Unlike previous attempts at achieving adaptive scheduling in sensor networks (e.g., TRAMA [3]), FLAMA does not require explicit schedule announcements during scheduled access periods. Alternatively, application-specific traffic information is exchanged among nodes during random access to reflect the driving application's specific traffic patterns, or *flows*. This allows FLAMA to still adapt to changes in traffic behavior and topology (e.g., node failure). FLAMA uses flow information to establish transmission schedules for each node. Additionally, FLAMA achieves traffic adaptiveness by assigning slots to a node depending on the amount of traffic generated by that node. This is accomplished by assigning *node weights* based on the incoming and outgoing flows. Nodes with more outgoing flows are given higher weights (i.e., more slots); the net effect is that nodes that produce/forward more traffic are assigned more slots.

The implementation of FLAMA we showcase in this paper is customized for data gathering applications, an important class of sensor network applications. In data gathering scenarios, the information sink(s) sends out a query for a given sensor reading. When relevant sensors reply, a tree rooted at the sink is established. FLAMA uses this tree to define the corresponding flows. We discuss FLAMA's flow discovery mechanism in detail (and illustrate it with examples) in the remainder of this section.

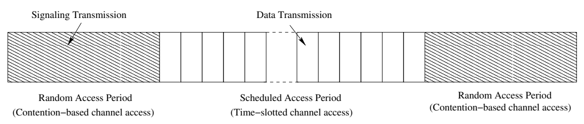


Fig. 1. FLAMA's time organization.

A. Application Overview

We assume that in data gathering applications, the sink initially sends out a query requesting data from sensing nodes. As the replies from the sensors are forwarded back, a tree rooted at the sink spanning all relevant nodes is established. Sensor nodes then sample readings periodically and send them to the sink over the collection tree. On its way to the sink, data might be aggregated [16] to minimize energy consumption. We use the sink as the synchronization point for the other nodes (e.g., the sink may be connected to a backbone network and is synchronized to it).

For this type of data gathering scenarios, traffic is predictable and exhibit regular patterns, which can be exploited when designing MAC layer protocols tailored for these application scenarios. Since data is sent back to the sink along the forwarding tree, nodes can easily determine incoming and outgoing flows. More specifically, a node has incoming flows from all its children in the tree and it has only one outgoing flow to its parent. The sink does not have any outgoing flows.

If R is the rate at which sensed data is generated at sensor nodes, all the nodes in the network except for the sink have an originating flow with data rate of R that exists for a period specified by the sink. A node has to either forward or aggregate flows that are incoming from its children. The outgoing flow rate is the sum of the incoming flow rates from the children and the originating flow rate R (if no data aggregation is employed [1]). FLAMA assigns node weights based on the resulting flow rates and performs traffic-adaptive scheduling. Section III-B describes how FLAMA acquires this information during the random access period.

B. Random-Access Period

In FLAMA, random access is used for time synchronization, exchanging neighbor information, and establishing flow information. In the specific case of data gathering applications, establishing flow information is essentially forming the data forwarding tree. The data gathering node, or sink, initiates tree formation and time synchronization. Every node in the tree synchronizes with its parent using a pair-wise time synchronization algorithm based on timestamps.

Hence, during the random access period the following tasks that are necessary for FLAMA's operation are performed: (1) network-wide time synchronization, (2) data forwarding tree formation, (3) traffic flow information exchange and weight computation for traffic-adaptive election, and (4) two-hop neighborhood information and corresponding node weight exchange.

Nodes running FLAMA start in random access mode and the radio is in either transmit or receive state. Control frames are exchanged every *SYNC_INTERVAL*. Two types of control frames (*SYNC* and *SYNC_REQ*) are exchanged during random access and channel access is based on carrier sensing. Figure 2(a) illustrates FLAMA's control frame format. Other than the source and destination information, the control frame also includes the node's outgoing flow weight , the node's parent, timestamp and a neighbor update list. The neighbor update list contains node identifiers for one-hop neighbors, their announced weights, and receive timestamps. In the case of data gathering applications, each node has only one outgoing flow towards the parent. Hence, it suffices to announce a single weight for the node. Other applications might need to announce multiple node weights based on the number of outgoing flows.

FLAMA requires time synchronization between two-hop neighbors. There are a number of known algorithms for time synchronization in ad hoc networks [17]–[20] that can provide accuracy in the order of microseconds. The basic idea behind all these algorithms is time-stamping the packet at the lowest possible level and using these timestamps to calculate clock drifts. We follow a similar approach to achieve time synchronization.

We employ a sender-initiated time synchronization mechanism where a node can send a *SYNC* frame only after synchronizing the clock with its parent. Otherwise, nodes send *SYNC_REQ* frames to discover parents. The sender (or the parent) initiates time synchronization by sending a *SYNC* frame with its local timestamp ($T1$). The receiver receives the frame at its local time $T2$. Now, $T2 = T1 + \delta + \tau$, where δ is the clock drift and τ is the propagation delay. The receiver replies with *SYNC_REQ* to the parent with its local timestamp ($T3$) and the sender receives the packet at its local time $T4$. Now, $T4 = T3 - \delta + \tau$. Using $T1, T2, T3,$ and $T4$, both δ and τ can be calculated. As we require the receiver to adjust its clock based on the sender, the sender sends back a *SYNC* frame announcing the timestamp $T4$ to the receiver. The receiver computes the clock drift δ using the following expression and adjusts its clock:

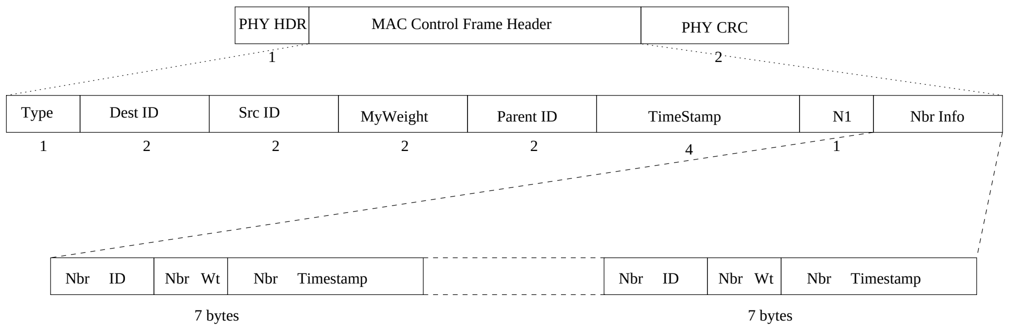
$$\delta = (T2 - T1 + T3 - T4)/2 \qquad (1)$$

Once a node becomes synchronized with its parent, it can start sending *SYNC* frames and synchronize downstream nodes. This process eventually synchronizes the entire network. Timestamps are generated at the physical layer to improve the accuracy. A node updates its child information whenever it receives *SYNC* frames with its node identifier as the parent. The length of the random access period is fixed based on the time required to complete the synchronization and tree formation processes.

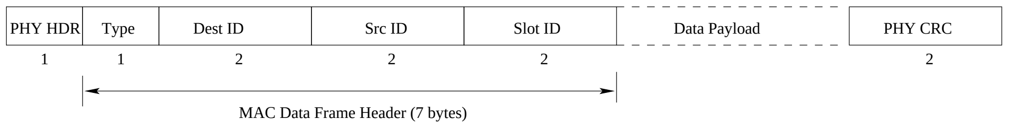
During random access periods, signaling packets may be lost due to collisions. Hence, the interval should be long enough to accommodate signaling retransmissions. In general, the length of the random access period is *NUM_RETX* × *SYNC_INTERVAL* × *NETWORK_RADIUS*, where *NUM_RETX* is the desired number of retransmissions and *NETWORK_RADIUS* is the network radius.

In our FLAMA implementation, we try to minimize state information exchanged and kept by nodes. Each node maintains the following neighborhood information: parent identifier (2 bytes), clock drift information ($T1$, $T2$, $T3$, $T4$, and $offset$,

[1]If data aggregation is used, then the outgoing flow rate remains constant at R.



(a) MAC control frame



(b) MAC data frame

Fig. 2. Frame formats.

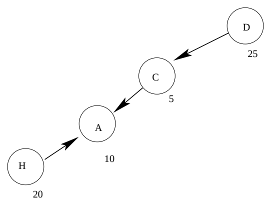


Fig. 3. Topology.

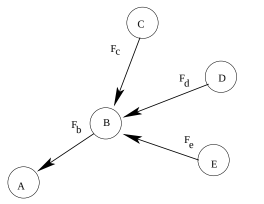


Fig. 4. Traffic Flows.

for a total of 20 bytes), one-hop neighbor table where each entry has 8 bytes of information (namely, node identifier, isChild flag, receive timestamp, node weight), and two-hop neighbor table (namely, node identifier, and node weight) with each entry having 3 bytes of information.

FLAMA uses node weights to adjust transmission schedules based on how much traffic individual nodes generate. Node weight calculation is illustrated using the example shown in Figure 4, where arrows represent traffic flows with certain rates. For example, node B has three incoming flows (from nodes C, D, and E with rates F_c, F_d, and F_e, respectively) and a single outgoing flow to node A with a rate F_b. The outgoing flow rate F_b is a function of incoming flow rates and is given by:

$$F_b = F_{origin} + c \times F_c + d \times F_d + e \times F_e \tag{2}$$

where c, d and e denote the fraction of the flow that is forwarded. If the flows are "terminal flows" than c, d and e are 0. F_{origin} denotes the rate of the originating flow (if any) from node B. Node weights are directly proportional to the outgoing flow rate. Hence, node B's weight is decided based on F_b and is announced during random access.

C. Scheduled-Access Period

Setting the Slot Size: During scheduled-access, channel access is time-slotted. The slot interval is fixed based on a maximum physical layer frame size. In our implementation we used a packet size of 128 bytes which is the maximum physical layer packet size for TinyOS's CC1000 physical radio module. A guard interval is added to the time slot duration to account for synchronization errors and radio mode switching, and is set to a multiple of the maximum possible clock drift. The data frame format is shown in Figure 2(b). The number of slots in the scheduled-access period is decided based on the duty cycle for scheduled access. The distributed election algorithm described below is used to decide the state of each node at every slot.

Distributed Election Algorithm: FLAMA uses a distributed election algorithm to schedule collision-free transmissions. The design of the election algorithm is driven by the assumption that sensing nodes are typically limited in terms of processing and memory resources. Essentially, for each node, the election algorithm decides which radio mode to use in the current slot. The choices are transmit, receive, or sleep. FLAMA ensures that there is only one transmitter in the two-hop neighborhood and thus avoids hidden-terminal collisions. FLAMA's election algorithm requires that each node maintains a list of one- and two-hop neighbors and their corresponding weights, and parent information.

A node can transmit if it has the highest two-hop priority for the given time slot and it has data to send. A node should be in receive mode if it is not the highest two-hop priority node and its highest one-hop priority node is a child. Otherwise, a node can go to sleep. While in receive mode waiting for data, the node can switch to sleep mode if it does not start receiving data for *PREAMBLE_INTERVAL*. Node weights computed during the random access period are incorporated into the election algorithm to provide more channel access for nodes with higher traffic rate. This makes FLAMA traffic-adaptive while maintaining the simplicity of the election algorithm.

Node priorities are calculated based on a pseudo-random function using the node identifier (*n*), time-slot identifier (*t*) and node weight (*weight*) as shown below:

$$prio(n,t,weight) = pseudorandom(n+t) + weight \times C \quad (3)$$

where C is a constant multiplier. The pseudo-random function could be implemented using linear shift registers and $(n+t)$ determines the initial state of the register.

FLAMA achieves collision-freedom by allowing only one transmitter in the two-hop neighborhood. Due to limited neighborhood information and the distributed nature of the algorithm, special care should be taken to prevent a node from sleeping when a neighbor is transmitting a data packet destined to this node.

For example, consider the network shown in Figure 3. The priority values computed for the nodes are shown next to the node. According the node H it has the highest priority in two-hop neighborhood and will transmit to node A. However, highest priority two-hop node in node A's neighborhood is node C. If node A decides to switch its radio to stand-by mode, it will miss the data transmission from node H. This leads to transmission to a sleeping node.

To prevent this, the election algorithm can identify highest priority one-hop flows that are hidden from the highest priority two-hop flow and listen if needed. However, to identify hidden flows, a node should maintain complete topology information for the two-hop neighborhood. This is expensive when the available processing and memory resources are low. Alternatively, a node can just listen for a short interval during the start of the time slot to determine whether the highest priority one-hop flow is an incoming flow. If the node receives a start symbol during this period, it continues to listen and receives the packet. Otherwise, the node switches it radio to sleep mode. This method is easily implementable in today's radios and does not require maintenance of complex state information. In our implementation of FLAMA for MICA2 Motes we use this optimization. In case of powerful nodes (more processing power and memory) one can improve the efficiency of scheduling by maintaining more state information.

In the absence of flow information, the election could be carried out with one- and two-hop node identifiers. In this case, the receiver for the elected transmission is not known as the nodes do not have flow information. Energy-efficiency could be still achieved by using the technique mentioned in the previous paragraph.

There are many other optimizations that are possible based on the available flow- and topology information to improve channel utilization. The election process can also be extended to take advantage of multichannel radios without much modifications. A flow's transmission channel could be calculated as a random-hash function using the transmitter's identifier and the time-slot identifier. Multiple flows can be elected within the two-hop neighborhood without collisions if they use different operating channel. These optimizations are planned as future work items.

IV. Performance Evaluation

FLAMA's performance is evaluated both by simulation and test-bed experimentation. The main goal of the simulation experiments is to highlight the importance of application-awareness in channel access scheduling. TRAMA is designed for general applications and hence, has to propagate traffic information explicitly and periodically. FLAMA, on the other hand, establishes flows based on traffic patterns exhibited by sensor network applications and need not propagate traffic information explicitly. S-MAC is also designed for general applications and does not account for application-specific traffic patterns. The main goal of our test-bed experiments is to establish the feasibility of implementing a TDMA-based, application-aware MAC protocols on sensor nodes and also to establish the advantages of FLAMA over traditional contention-based channel access protocols.

A. Performance Metrics

The following metrics are used to assess the performance of the protocols:

- **Average Packet Delivery Ratio** is the ratio of number of packets received at the sink to the number of packets sent by all sensor nodes. For broadcast traffic, a packet is counted to be received only if it is received by all the one-hop neighbors.
- **Percentage Sleep Time** is the ratio of the time spent in low-power sleep mode to the total experiment run time.
- **Latency** is computed as the average per-hop latency for the network.
- **Average Queue Drops** provides the average number of packets dropped at the MAC-layer queue.

B. Simulation Setup

To establish the importance of application-awareness, FLAMA's performance is compared against that of TRAMA and S-MAC. Qualnet [2] is used as the simulation platform. A physical layer model based on Mica2 motes' Chipcon CC1000 radio is implemented to accurately model the operating environment. The radio's data rate is $19.2Kbps$ and its range is around 300 feet.

Sensor network deployments for data gathering are often hierarchical, where there are some more capable data gathering nodes, each of which collect data from a subset of sensor nodes. We try to mimic this kind of deployment by using a grid topology with 16 nodes with the sink in the corner periodically issuing queries to the network to gather requested information. Nodes in the grid are separated by a distance of $75m$. All sensor nodes participating in the network report to the sink sending the requested information at the rate specified in the query. In our simulations, sensor nodes generate periodic 128-byte packets after an initial warmup time. This initial warmup

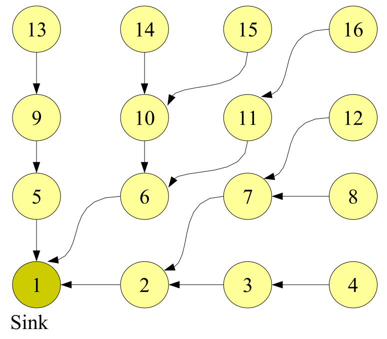


Fig. 5. Data Gathering Application

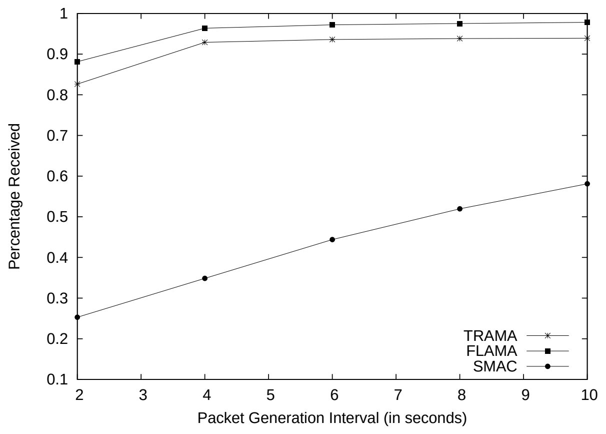


Fig. 6. Average delivery ratio.

period is needed to allow for neighbor discovery and is fixed at 50*S*. The data generation rate is varied over multiple trials.

In FLAMA, flow discovery is done during the random-access period and this effectively establishes the data gathering tree. Since TRAMA and S-MAC do not perform flow discovery, we hard-coded the data collection tree (shown in Figure 5) for the simulation experiments involving TRAMA and S-MAC. The duty cycle for S-MAC is fixed at 10% and nodes are allowed to do adaptive listen at end of data transmission. S-MAC's synchronization interval is set to 10*S* and the contention window for data and synchronization packets are set to 31 and 15 slots, respectively. The simulation is run for 2000 seconds and results are averaged over multiple runs.

C. Simulation Results

Figure 6 shows the average packet delivery ratio at the sink for different traffic generation intervals. FLAMA achieves better delivery ratio than TRAMA and S-MAC. This is due to the fact that FLAMA performs traffic adaptive scheduling without incurring much overhead. Nodes that are near the sink have a larger outgoing flow-rate and these nodes are favored in the election process. Whereas, TRAMA needs to propagate traffic information periodically and this is a significant overhead during scheduled access period. Hence, for the given simulation duration, FLAMA is able to service more packets than TRAMA.

We observed that the synchronized listen- and sleep cycles of S-MAC affect neighbor discovery and data throughput in multi-hop forwarding. This is because of the fact that S-MAC restricts transmitting or receiving packets to a specific (small) window of time. Depending on the contention window size for transmitting data and synchronization packets, collisions occur due to hidden terminals. This affects neighbor discovery significantly as the synchronization packets are sent by unreliable broadcasts. Hence, the average delivery ratio at the sink is significantly less for S-MAC when compared with scheduling-based protocols.

Figure 8 presents the average per-hop delay for FLAMA, TRAMA, and S-MAC. Overhead in periodic traffic announcements leads to higher queueing delay at intermediate nodes running TRAMA. The queueing delay for FLAMA is significantly lesser than that of TRAMA (up to 75 times). S-MAC achieves lesser delay than FLAMA in this topology. This is due to the delay involved in the election algorithm, which is dependent of the two-hop neighborhood size. However, it should be noted that FLAMA achieves much higher reliability than S-MAC. Hence, end-to-end application perceived delay is much higher for S-MAC due to retransmissions.

Energy efficiency for FLAMA, TRAMA, and S-MAC are shown in Figure 7. We observe that FLAMA achieves significant energy savings when compared to TRAMA and S-MAC. This is because FLAMA exchanges lesser information than TRAMA during scheduled access periods. For both FLAMA and TRAMA the energy savings are proportional to the offered load as expected. For S-MAC, energy savings depends on the fixed duty cycle.

D. Test-bed Experiments

We implement FLAMA on TinyOS for Mica2 motes and its performance is compared with that of S-MAC. Similarly to S-MAC, FLAMA is implemented on top of ISI's radio communication stack [21] for the Mica2 platform. ISI's S-MAC implementation is used for the experiments. Both for S-MAC and FLAMA, there is no MAC layer buffer to queue up frames. Hence, a frame from the application is dropped if the send buffer is full.

In the topology chosen for initial evaluation, a sink collects periodic data generated by sensor nodes and all the nodes are directly connected to the sink and the placements are such that hidden terminals exist. The main goals of the test-bed experiments are: to showcase that FLAMA can be implemented on sensor network platforms and also compare FLAMA's performance with S-MAC in a sensor network test-bed (instead of just through simulations).

During the experiments, each node maintains statistics about the number of data packets generated, number of data packets forwarded, number of data packets dropped due to buffer overflow, average service time for the packets, and radio

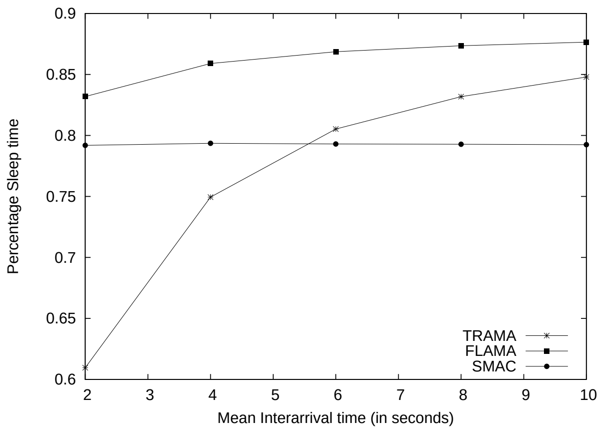


Fig. 7. Energy savings.

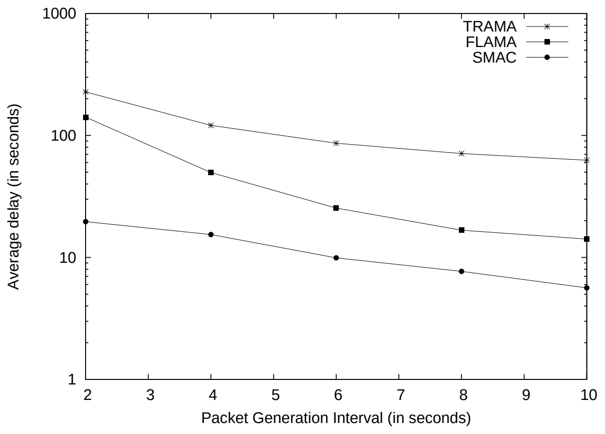


Fig. 8. Average queueing delay.

Packet generation Rate (seconds)	Average Delivery Ratio (FLAMA)	Average Delivery Ratio (S-MAC)
2	94 %	74.19 %
4	99.6 %	74.13 %
6	100 %	75.81 %

TABLE I
AVERAGE DELIVERY RATIO.

Packet generation Rate (seconds)	Energy savings (FLAMA)	Energy savings (S-MAC)
2	74.6 %	72.9 %
4	78.4 %	79.3 %
6	86.9 %	80.6 %

TABLE II
PERCENTAGE SLEEP TIME.

Packet generation Rate (seconds)	Average Drops (FLAMA)	Average Drops (S-MAC)
2	0	13.66
4	0	9
6	0	0.33

TABLE III
AVERAGE DROPS.

statistics (i.e., time spent in transmit, receive, and standby mode). Statistics information is sent to the sink periodically along with the data. The sink is connected to an end host, and forwards all packets received to the host. Statistics are collected and processed by the host computer for every packet received at the sink.

We considered 128 bytes of data payload for both S-MAC and FLAMA. The routing information is hard-coded for S-MAC and the experiments are run multiple times to average the results. Identical operating environments are ensured for both S-MAC and FLAMA to avoid measurement errors. For S-MAC we considered 90% duty cycle for sleeping and for FLAMA we used 90% duty cycle for the scheduled access period. The length of the random-access period is fixed to 55*s*. The experiments are run for 400*s* so that there is enough time spent in scheduled access for FLAMA.

We consider different data rates for traffic generation and the results of our experiments are summarized in Tables I, II, and III. As we can observe, FLAMA significantly outperforms S-MAC in terms of delivery ratio, drop rate, and energy efficiency. For this topology, the average service time for S-MAC is on the order of 700*ms*, while for FLAMA the service time is around 100*ms*. Hence, the number of packets dropped for S-MAC is significantly higher than that of FLAMA. This affects the end-to-end reliability measured at the sink. It should be noted that FLAMA's delay is dependent on the number of two-hop nodes.

For low data rates, FLAMA achieves perfect reliability while S-MAC's reliability is 75%. This is because FLAMA avoids collision and transmissions to sleeping node. Also it does not exchange any control packets during the scheduled access period and hence the channel contention level is less. On the other hand, even though S-MAC uses RTS/CTS handshakes to avoid hidden-terminal collisions, it loses data packets due to increased service time and also due to RTS/CTS handshake failures. Note that low offered loads tend to benefit contention-based protocols.

The average number of drops reported in the Table III reflects the number of packets dropped at the network layer (i.e., the buffer is full when a new packet arrives). In addition to these losses, there can be losses due to collisions, transmission errors, and RTS/CTS handshake failures. Hence, there is no direct correlation between the number of losses reported in Table III and the end-to-end delivery ratio. The results also indicate that FLAMA achieves energy savings comparable to S-MAC. This is in spite of the fact that FLAMA has the radio on during the entire random access period. This clearly demonstrates the importance of using an adaptive scheduling approach for channel access in sensor networks. In our future experiments, we will be testing more complicated topologies with more nodes and multiple hops.

V. Conclusions

This paper introduced an energy-efficient, scheduled-based, application-aware medium access control protocol specifically designed with sensor network applications in mind. The proposed protocol is named FLAMA for FLow-Aware Medium Access and uses application-specific traffic information to adapt to the sensor network scenario at hand. Using traffic information, FLAMA is able to establish transmission schedules as well as determine which nodes should be in transmit or receive mode, or can switch their radios to low-power sleep state. This feature is instrumental in achieving energy efficiency without compromising the simplicity of the protocol.

Using simulations and tested experimentation we evaluated the performance of FLAMA and demonstrated the importance of application-awareness in medium access. Simulation results indicate that FLAMA outperforms TRAMA and S-MAC in terms of reliability, and energy savings. FLAMA achieves significant improvement in delay performance for scheduling-based protocols. Our test-bed experiments showcase FLAMA's deployment on MICA2 Motes. They also show that FLAMA can achieve better end-to-end reliability with significant energy savings when compared to a contention-based protocol such as S-MAC.

References

[1] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *INFOCOM 2002*, june 2002.

[2] "Scalable networks, http://www.scalble-networks.com."

[3] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," in *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003, pp. 181–192.

[4] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for network sensors," in *ASPLOS 2000*, November 2000.

[5] "http://www.cs.berkeley.edu/ awoo/smartdust/."

[6] S. Singh and C. Raghavendra, "PAMAS: Power aware multi-access protocol with signaling for ad hoc networks," 1999. [Online]. Available: citeseer.nj.nec.com/460902.html

[7] A. Woo and D. Culler, "A transmission control scheme for media access in sensor networks," *ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom) 2001*, 2001.

[8] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," in *Proceed-ings of the IEEE Infocom*, June 2002.

[9] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003, pp. 171–180.

[10] G. Lu, B. Krishnamachari, and C. Raghavendra, "An adaptive energy-efficient and low-latency MAC for data gathering in sensor networks," in *Int. Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, Santa Fe, NM, Apr. 2004.

[11] I. Chlamtac and A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 23–29, February 1994.

[12] J. Ju and V. Li, "An optimal topology-transparent scheduling method in multihop packet radio networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 3, pp. 298–306, June 1998.

[13] S. Ramanathan, "A unified framework and algorithm for channel assignment in wireless networks," *Wireless Networks*, vol. 5, no. 2, pp. 81–94, 1999.

[14] L. Bao and J. J. Garcia-Luna-Aceves, "A new approach to channel access scheduling for ad hoc networks," in *The seventh annual international conference on Mobile computing and networking 2001*, 2001, pp. 210–221.

[15] K. Sohrabi and G. Pottie, "Performance of a novel self-organization protocol for wireless ad hoc sensor networks," *IEEE 50th. Vehicular Technology Conference*, pp. 1222–1226, 1999.

[16] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," 2001. [Online]. Available: citeseer.nj.nec.com/article/intanagonwiwat01impact.html

[17] J. Elson and D. Estrin, "Time synchronization for wireless sensor networks," in *IPDPS 2001*, April 2001. [Online]. Available: citeseer.nj.nec.com/elson01time.html

[18] K. R, "Time synchronization in ad hoc networks," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*. ACM Press, 2001, pp. 173–182.

[19] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003, pp. 138–149.

[20] H. Dai and R. Han, "Tsync: a lightweight bidirectional time synchronization service for wireless sensor networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 8, no. 1, pp. 125–139, 2004.

[21] W. Ye, J. Heidemann, and D. Estrin, "A flexible and reliable radio communication stack on motes," USC Information Sciences Institute, Tech. Rep., September 2002.