Load-Aware Energy-Efficient Medium Access Control for Wireless Sensor Networks

Chih-Min Chao[1] and Yi-Wei Lee
Department of Computer Science and Engineering
National Taiwan Ocean University, Taiwan
[1] Email:cmchao@ntou.edu.tw
Tel: +886-2-24622192 ext 6651, Fax: +886-2-24623249

Abstract

In wireless sensor networks, sensors are usually battery-powered. Therefore, it is essential to reduce energy consumption to prolong network lifetime. In traditional power saving MAC protocols, such as IEEE 802.11 Power Saving Mode and S-MAC, sensor nodes must wake up at every beacon interval to check if they are involved in any transmission. Such a fixed active/sleep mechanism fails to adjust a sensor node's sleep duration based on its traffic load. Thus, these protocols suffer from either higher energy consumption or higher latency. In this paper, we propose a Load-aware Energy-efficient MAC protocol (LE-MAC) which utilizes fuzzy control and the quorum concept. The LE-MAC protocol utilizes fuzzy control to dynamically adjust the active/sleep ratio of each sensor node. The duration for a node to stay in sleep mode is increased to conserve energy when its traffic load is light. On the contrary, the ratio of operating in active mode is increased to reduce transmission latency as traffic load is getting heavier. The quorum concept is adopted to guarantee that two sensor nodes with different active/sleep ratios can meet and communicate with each other. Simulation results verify that our LE-MAC can adapt to various network traffic loads. Furthermore, LE-MAC achieves significant performance improvements in energy consumption, transmission successful ratio, and transmission delay.

Keywords: Wireless Sensor Networks, Fuzzy Control, Energy-Efficient MAC protocol, Quorum Systems.

1 Introduction

Plenty of potential applications for wireless sensor networks (WSNs) have been discussed lately. These applications include environment and habitat monitoring, mobile object tracking, and military surveillance. A WSN typically consists of many inexpensive wireless sensor nodes, each capable of collecting, processing, and storing environmental information. These nodes are normally deployed in an ad hoc manner and operate distributedly to coordinate with each other to fulfill a common task. The sensor nodes are battery-powered and are often not feasible to recharging. Thus, it is necessary to design energy-efficient protocols for wireless sensor networks. In the literature, many protocols have been proposed to extend the network lifetime of sensor networks in their deployment protocols [12, 18, 21], routing protocols [2, 9, 17], and power-efficient MAC protocols [3, 4, 5, 7, 13, 15, 20, 23, 26].

The many-to-one communication model is adopted in many wireless sensor applications where all sensor nodes constantly report data to a single sink node. In such an environment, sensor nodes close to

the sink deplete their energy faster. This is referred to as the energy hole problem. To solve this, node deployment protocols [12, 18, 21] suggest distributing more nodes around the sink. However, sometimes only random (uniform) node distribution is possible due to environment limits. In such situations, node deployment protocols are incapable of prolonging network lifetime.

Designing energy-efficient MAC protocols is another way to prolong network lifetime. Since idle listening has been identified as a major reason for energy wastage, several solutions [3, 5, 7, 13, 15, 20, 23, 26] managed to reduce the time a sensor node spends in idle listening. Some of them [5, 13, 20, 23, 26] required time synchronization among sensor nodes. Since time synchronization is essential for many sensor applications, it is natural for a synchronous MAC protocol to be used. In general, these protocols maintain a schedule that indicates when a sensor should be awake in order to check transmission activity. Energy consumption is reduced since nodes keep awake only at a specified time. However, these synchronous protocols either suffer from long delay or fail to adapt to individual node's traffic well. In asynchronous solutions [3, 7, 15], sensor nodes independently schedule their awake period. In transmitting data, the source node sends a preamble that is long enough for the destination node to detect. When a preamble is detected, the destination node will remain awake to receive the data that follows the preamble. These protocols avoid the synchronization overhead. However, long preambles introduce long latency and extra energy consumption. A more detailed review of representative power-saving protocols is shown in Section 2.

We focus on the synchronous MAC protocol design in this paper. Most existing synchronous MAC proposals wake up all the sensor nodes at every cycle. Such a regular wake up frequency is not a satisfactory design since different nodes may have different traffic loads. There exist some proposals [5, 26] that aim to produce different active/sleep schedule for different sensor nodes. However, they either fail to provide good enough performance or cannot adjust the schedule dynamically. In this paper, we proposed a load-aware energy-efficient MAC Protocol (LE-MAC). In LE-MAC, the energy conservation is obtained through the quorum-based wake up schedule. This schedule is dynamically adjusted through fuzzy control, according to each sensor node's traffic load. When the load is heavy, nodes will wake up frequently

to delivery their traffic. When the load is light, nodes will wake up less frequently to reduce energy consumption. Simulation results verify that the LE-MAC can properly adjust nodes' schedule according to their traffic condition changes.

The rest of this paper is organized as follows. Related work reviews and problem statement are in Section 2. In Section 3, preliminaries are described. Section 4 describes the details of the proposed protocol. Simulation results are presented in Section 5. Our conclusion and plans for future work are drawn in Section 6.

2 Related Work and Problem Statement

First we review some asynchronous protocols. B-MAC [15] achieves low power operation by using lower power listening and a long preamble. To transmit data reliably, a sender uses a preamble that is long enough to notify the receiver. For example, if the receiver checks the channel every 50 ms, a preamble will be longer than 50 ms. Once a preamble is recognized, the receiver will stay awake to receive the packet. The extended preamble in B-MAC produces excess energy consumption when compared with synchronous solutions. Besides, non-target nodes will waste a lot of energy since they have to stay awake until the end of the preamble to check if they are the recipient. This long preamble mechanism also generates long delays.

Similar to B-MAC, WiseMAC [7] sends a preamble before any transmission. A receiver running WiseMAC notifies its neighbors the time of its next awake period through an extra field in the ACK packet. This information enables a sender to start a preamble just before the receiver wakes up. Such a mechanism reduces the energy consumption of sending long preambles at the expense of an extra field in each ACK packet and memory space for recording each neighbor's schedule. WiseMAC does not provide a mechanism to adapt to changing traffic conditions.

X-MAC [3] is another improvement over B-MAC. X-MAC replaces the long preamble in B-MAC by a series of short preamble packets that are separated by small pauses. The pauses enable the target receiver to send an early ACK to cease the preamble earlier. The target address is contained in each short

preamble to alleviate the energy wastage problem of non-target nodes. Short preamble packets reduce delay and energy wastage when compared with B-MAC. However, they may still consume more energy when compared with synchronous protocols since more preamble packets are delivered.

For the synchronous MAC protocols, S-MAC [23] is a cluster-based one that avoids idle listening by allowing sensor nodes to go to sleep periodically if they are not involved in any communication. This is similar to the IEEE 802.11 power saving mode where nodes wake up at the beginning of each beacon interval to check if they need to remain awake. Nodes running S-MAC exchange synchronization and schedule information with their direct neighbors at the beginning of each listen period to ensure that neighboring nodes would listen and go to sleep simultaneously. A virtual cluster consists of nodes that follow to the same schedule. A node that receives two different schedules follows both and this node belongs to two different virtual clusters. To reduce latency, the authors also introduce adaptive listening [24]. Sensor nodes can wake up briefly at the end of the transmission, if a CTS is overheard, to possibly act as the next hop. S-MAC reduces each sensor node's energy consumption by keeping the duty cycle low; however, it still has some flaws. For example, its low duty cycle may produce long transmission latency. Besides, it is hard to adapt to an individual's traffic well since nodes in a virtual cluster adhere to the same schedule. The same schedule may not be optimal for all nodes because each node has a different traffic load.

T-MAC [20] is an extension of S-MAC that adopts an adaptive duty cycle. A sensor node in listen mode will stay awake until there is no activity for a duration of T_A. Such a power-down strategy may produce the early sleeping problem wherein potential receivers go to sleep too early. This implies the number of hops a message can travel in a time frame is limited. T-MAC suffers from long transmission latency although it finds a way to determine a node's active duration.

DMAC [13] also uses an adaptive duty cycle. In DMAC, a data gathering tree is built and transmission latency is reduced by staggering active times of nodes along the tree. With this staggered wakeup schedule, DMAC achieves better performance in transmission latency, throughput, and energy conservation when compared with S-MAC. However, DMAC still wakes up all the sensor nodes at every cycle. This may

produce excess energy consumption for light-loaded nodes because they may remain idle in most cycles.

PMAC [26] is another protocol that can adjust its duty cycle. In PMAC, each sensor node individually chooses its active/sleep pattern according to its own traffic condition. A sensor node with more data is allowed to generate a pattern with more awake periods. This pattern, indicating that the node intends to sleep or not at the upcoming time slots, is shared with all the neighbors. The actual schedule for each sensor node is constructed based on its own pattern and those of its neighbors. PMAC enables nodes to adaptively construct schedules based on their own traffic conditions. However, its functionality deeply relies on pattern exchanges among sensor nodes. Two sensor nodes may be unable to meet each other if they do not receive the other's schedule correctly. This results in idle listening and fruitless transmissions. Besides, nodes running PMAC may also experience long transmission latency.

QMAC [5] enables sensor nodes to have different active/sleep frequencies. Nodes running QMAC are classified into different coronas according to their hop count distances to the sink. A node with shorter distance to the sink is supposed to have heavier load and thus must keep active longer. In QMAC, the active/sleep frequency for nodes in a particular corona is determined by the grid quorum size associated with the corona. Each sensor node randomly and independently selects one row and one column as its quorum time frames wherein it must stay awake. For nonquorum time frames, a node that does not have pending packets to transmit/receive can switch to sleep mode to conserve energy. A smaller grid size is assigned to a corona that is closer to the sink. This enables nodes located in the inner coronas to wake up more frequently. The nonempty intersection property of quorum systems is adopted to guarantee that nodes with different active/sleep frequencies can communicate with each other. QMAC has better flexibility since it does not wake up all the sensor nodes at every time frame. The cost of this is increased latency since it takes longer time for two nodes to be active simultaneously. To reduce this transmission delay, the next hop group concept is also introduced in [5] which produces the QMAC_LR protocol. The concept of next hop group can be illustrated by a four-corona network, labelled from C_1 to C_4, as shown in Fig. 1. Instead of requesting a specific inner corona node to relay its traffic, node X selects a set of candidates as its next hop group members which are capable of relaying its traffic. With next hop group,

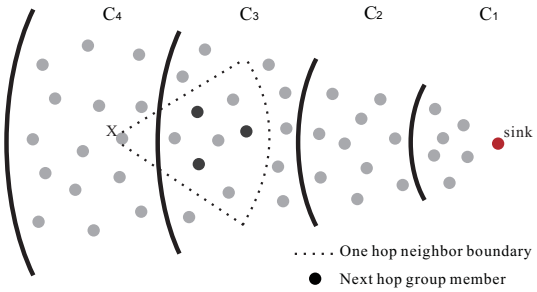

Figure 1: Next hop group

a node has much chance to transmit its traffic faster. QMAC and QMAC_LR work well in an environment where sensor nodes are deployed uniformly and traffic is generated evenly and periodically. However, it fails to adjust each sensor node's schedule dynamically in a more practical network where traffic load for each node changes over time.

2.1 Problem Statement

The above mentioned protocols fail to provide each node a schedule that is robust and dynamically adjustable. In fact, in a network where nodes and traffic are not uniformly distributed, a better schedule is the one that can be adjusted dynamically. However, choosing a proper schedule for each node is not easy. Issues such as the number of neighbors, pending packets in each node, and experienced transmission delay should be considered. The purpose of this work is to solve the problem of how to dynamically determine a proper active/sleep ratio for each sensor node. Specifically, we aim to design a load-aware MAC mechanism that can arrange nodes' active/sleep schedules dynamically according to their traffic condition changes.

3 Preliminaries

The proposed protocol utilizes fuzzy control and the quorum system concept. We introduce these two components briefly in this section.

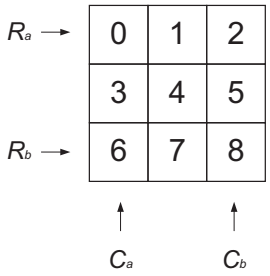

Figure 2: A consecutive nine beacon intervals can be represented by a 3×3 grid.

3.1 The Quorum System

A quorum is a request set to enable some actions, if permission is granted. There are nonempty intersections between any two quorum sets. There exist many kinds of quorum, such as the grid-based [6, 14], tree-based [1], majority-based [19], and others [8, 10, 11, 22]. In this paper, a quorum set represents the time frames wherein a sensor node has to wake up. For nonquorum time frames, sensor nodes can enter sleep mode for the entire time frame to save energy. Based on quorum's properties, any two nodes are guaranteed to wake up and meet each other at some time frame. Without loss of generality, we explain our protocol through a grid-based quorum. In a grid-based quorum, one row and one column are selected in an $n \times n$ grid. This concept can be shown in Fig. 2. Hosts A and B select row R_a, column C_a and row R_b, column C_b as their quorums, respectively. There are two intersections between A and B, one for R_a and C_b and the other for C_a and R_b. As we let sensor nodes wake up at their chosen quorum time frames, both nodes will wake up at these intersections.

Fig. 3 is an example of representing nine continuous time frames by a 3×3 grid in a left-to-right and top-to-bottom way. A sensor node randomly selects one row and one column as its quorum time frames. In this example, node A picks the first row and the first column as its quorum while host B selects the third row and the third column. This means node A wakes up at time frames 0, 1, 2, 3, and 6 while node B wakes up at time frames 2, 5, 6, 7, and 8. The intersections occur at time frames 2 and 6, when both nodes A and B keep awake.

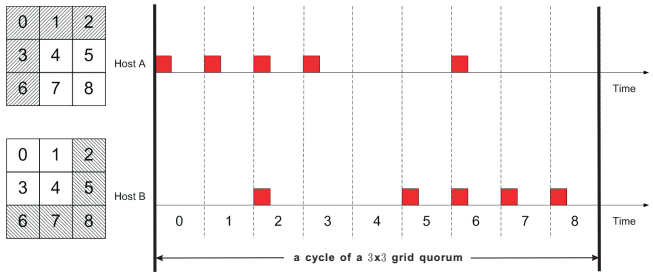

Figure 3: An example of intersections. Host A and host B meet each other at intervals 2 and 6.

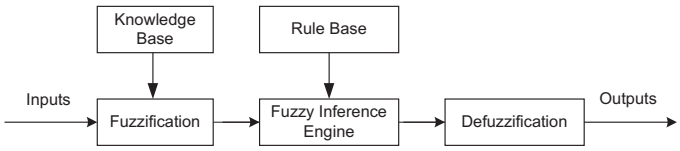

Figure 4: Block diagram of fuzzy control.

3.2 Fuzzy Control

The concept of fuzzy set is first introduced by Zadeh [25]. Through the efforts of many researchers, it is extended to be a theory of fuzzy logic. Fuzzy logic handles degree of truth: truth values range from "completely true" to "completely false". Fuzzy logic is able to make precise conclusions from vague or imprecise linguistic information (such as "very hot", "a little busy", and "quite large"). That is, it provides a easier way to explain linguistic implicit information than traditional two-valued Boolean logic.

The fuzzy control systems use the concept of fuzzy logic to make a controlling decision. The *if-then* rules and the associated inference engine are utilized to simulate human decision making: work from approximate data and find precise solutions. Fig. 4 [16] shows the block diagram of fuzzy control. The operation of a fuzzy control system can be summarized as the following steps:

1. Define the input and output variables.
2. Classify these variables into a number of fuzzy sets. Each fuzzy set has a linguistic label.
3. Assign a membership function for each of these fuzzy sets.
4. Build the rule-based inference engine by assigning the fuzzy relationships between input/output variables and fuzzy sets.

5. Fuzzify the inputs.

6. Determine the (fuzzy) outputs. Each rule generates one output.

7. Integrate these outputs.

8. Defuzzify the output to obtain a crisp decision.

4 LE-MAC (Load-Aware Energy-Efficient MAC Protocol)

In this paper, we make the following assumptions:

- Time is divided into a series of time frames.
- All nodes are time synchronized.
- Each sensor node has a unique ID.
- Sensor nodes are randomly and uniformly distributed in the network area. We assume that all the sensor nodes are distributed in a circular area centered at the sink node as in [21]. The area is divided into several roughly equal-width coronas according to their hop count distance to the sink node, as shown in Fig. 5. During the network initialization phase, a control packet NET_INIT with a field $hopcount = 1$ is sent from the sink to create the coronas. Upon receiving this packet, each node rebroadcasts the packet with the $hopcount$ field increased by one. A node belongs to corona C_i if it receives an NET_INIT with $hopcount$ field equals to *i*. If multiple NET_INIT packets are received, only the one with the least $hopcount$ value is handled. Sensor nodes in corona C_i are *i* hops away from the sink and rely on nodes in C_{i-1} to relay their data.
- Each sensor node reports their data to the common sink node.
- All of the sensor nodes have the same transmission range.
- All of the sensor nodes are static after deployment.

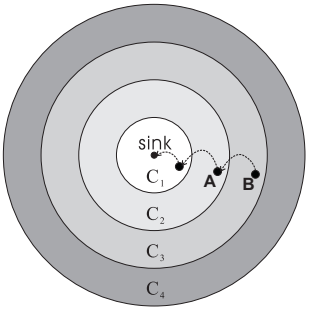

Figure 5: The network is divided into adjacent coronas centered at the sink node. The i-th corona is denoted as C_i.

LE-MAC follows the medium access control mechanism of QMAC. LE-MAC extends QMAC in that it aims to dynamically adjust each sensor node's grid size to fit its traffic load. LE-MAC adopts fuzzy control to achieve this goal. For a node using a grid of $n \times n$, each continuous n^2 time frames are called a *cycle*. A node running LE-MAC applies the fuzzy control mechanism at the end of every cycle to determine the proper grid size for the upcoming cycle. To accomplish this, two input variables are defined for LE-MAC. The first one is *Actual Wake-up Ratio (AWR)* which represents the ratio of actually wake-up time frames to scheduled ones. For a sensor node i using an $n \times n$ grid, the number of scheduled wake-up time frames is $2n - 1$. Specifically, the AWR is defined as

$$AWR = \frac{N_{a_i}}{2n-1}, \;\; 2n - 1 \leqslant N_{a_i} \leqslant n^2 \tag{1}$$

where N_{a_i} is the number of actually time frames that node i wakes up in a cycle. The AWR has a minimum value of 1. AWR helps identify whether the grid size currently being used is proper or not. A large AWR value implies the grid size being used is too big in that some more extra wake up time frames are necessary to deliver/receive packets. To handle this, it is nature to reduce the grid size being used. On the contrary, a small AWR is more likely to induce a need of grid size increase. It should be noted that AWR alone is unable to model each host's traffic load well. A node in a key or a junction position is typically responsible of relaying packets. However, it may also temporarily experience light traffic which produces a smaller AWR and grid size increase. To make such a node reacts promptly for incoming traffic, the grid size being used should not be too big. To address this issue, we need another variable to manage the grid size selection.

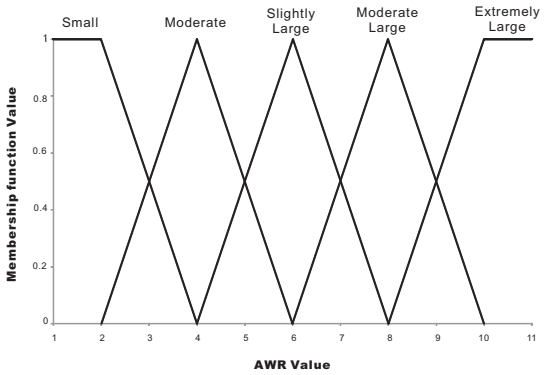

Figure 6: Membership function of AWR

TABLE 1: LE-MAC CONTROL RULES

DRR	**AWR**				
	SMALL	MODERATE	SLIGHTLY LARGE	MODERATE LARGE	EXTREMELY LARGE
LIGHT	EXTREMELY SUFFICIENT	VERY SUFFICIENT	MODERATE SUFFICIENT	SLIGHTLY SUFFICIENT	MODERATE
MODERATE	VERY SUFFICIENT	MODERATE SUFFICIENT	SLIGHTLY SUFFICIENT	MODERATE	SLIGHTLY INSUFFICIENT
SLIGHTLY HEAVY	MODERATE SUFFICIENT	SLIGHTLY SUFFICIENT	MODERATE	SLIGHTLY INSUFFICIENT	MODERATE INSUFFICIENT
MODERATE HEAVY	SLIGHTLY SUFFICIENT	MODERATE	SLIGHTLY INSUFFICIENT	MODERATE INSUFFICIENT	VERY INSUFFICIENTT
EXTREMELY HEAVY	MODERATE	SLIGHTLY INSUFFICIENT	MODERATE INSUFFICIENT	VERY INSUFFICIENT	EXTREMELY INSUFFICIENT

The second input variable is *Data Receiving Ratio (DRR)* which is defined as the ratio of receiving time frames in a cycle. For a node i using an $n \times n$ grid, the DRR can be obtained as follows:

$$DRR = \frac{N_{r_i}}{n^2}, \;\; 0 \leqslant N_{r_i} \leqslant n^2 \tag{2}$$

where N_{r_i} is the number of time frames that node i has received a data packet in a cycle. The range of the DRR values is between 0 and 1. We also define one output variable, *Grid-size Adjustment Value (GAV)*, which represents the amount of grid size to be adjusted.

For the input variable AWR, we define five linguistic variables representing each node's actual wake up ratio: *small*, *moderate*, *slightly large*, *moderate large*, and *extremely large*. For the input variable DRR, we also define five linguistic variables standing for each sensor's packet receiving condition: *light*, *moderate*, *slightly heavy*, *moderate heavy*, and *extremely heavy*. For the output variable GAV, nine linguistic variables are defined: *extremely sufficient*, *very sufficient*, *moderate sufficient*, *slightly sufficient*, *moderate*, *slightly insufficient*, *moderate insufficient*, *very insufficient*, and *extremely insufficient*. The membership function of AWR, DRR, and GAV is shown in Fig. 6, Fig. 7, and Fig. 8, respectively.

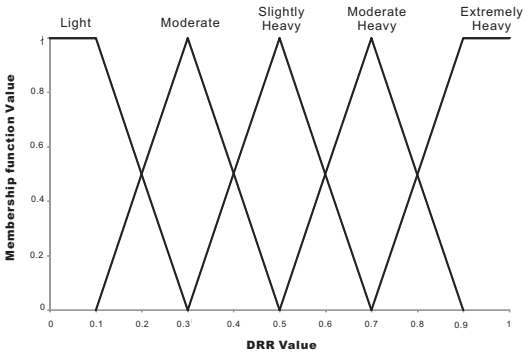

Figure 7: Membership function of DRR

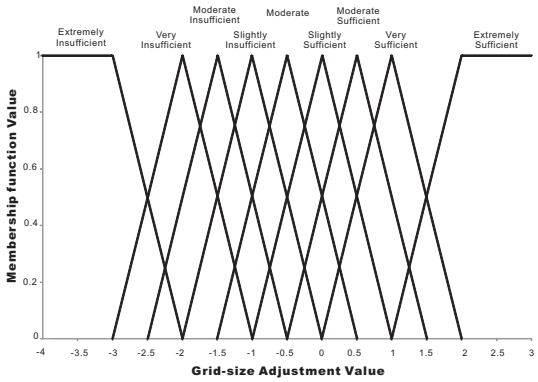

Figure 8: Membership function of GAV

The fuzzy inference engine of LE-MAC consists of a number of *if-then* rules in which antecedents and consequents are linguistic variables. An example rule is

If *AWR is "extremely large" and DRR is "extremely heavy"*
Then *sensor node is "extremely insufficient"*.

Since there five linguistic variables for each input variable, totally 25 rules are defined as shown in Tab. 1.

With these rules, the next job is to translate them into fuzzy relations. Then, through inferencing and defuzzificaiton, we can obtain the final output value to adjust the grid size. Specifically, to obtain the output, the following four steps should be taken.

Step. 1 For each input value, find the associate linguistic variable(s) that has nonzero membership value(s). For example, assume that a node has the input values of AWR = 7 and DRR = 0.75 at the end of a cycle. For AWR = 7, as shown in Fig. 9, there are two linguistic variables that have nonzero membership value: *slightly large* (0.5) and *moderate large* (0.5). For DRR = 0.75, there are also two linguistic variables: *moderate heavy* (0.75) and *extremely heavy* (0.25).

Step. 2 For associate linguistic variables found in Step 1, find the corresponding fuzzy rule(s). Follow the example mentioned in Step 1, there are two linguistic variables for each of AWR and DRR. Thus, totally four fuzzy rules will be found. One of them is **If** *AWR is "slightly large" and DRR is "moderate heavy"* **Then** *sensor node is "slightly insufficient"*.

Step. 3 For each selected rule, find the output variable membership value. Here we use the MIN method (also called the AND method) which selects the minimum value from the inputs. For example, the GAV membership value is Min(0.5, 0.75) = 0.5 for the rule showed in the previous step. According to this value, we obtain a marked area as shown in Fig. 10(a). One GAV membership value can be found for each of these rules and thus totally four areas can be obtained as showed in Fig. 10(b).

Step. 4 Defuzzify each GAV membership value obtained in the previous step and output the average of these defuzzified values, round off after the decimal. There are several defuzzification methods such as

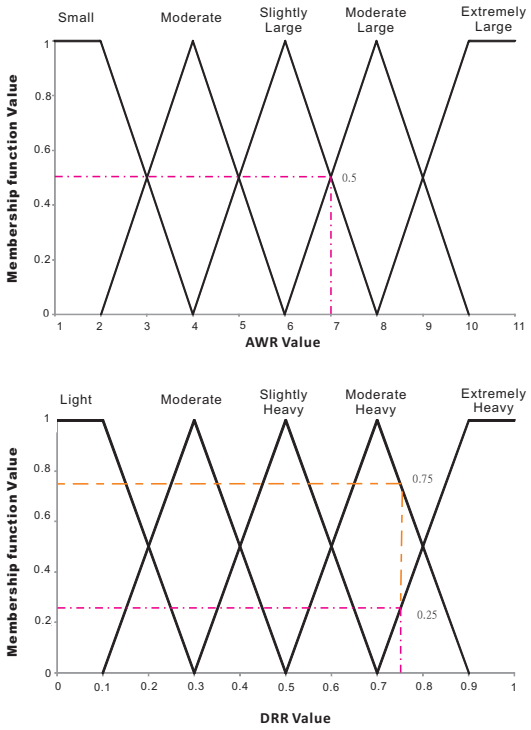

Figure 9: Associate membership values for AWR=7 and DRR=0.75

the centroid, bisector, mean of maximum, smallest of maximum, and largest of maximum method. We use the centroid method for defuzzification because it is the most widely used one to derive averages from input factors. The centroid of each area in Fig. 10(b) is calculated and corresponding GAV for each centroid, from left to right, is -2, -1, 0, and 1, respectively. After averaging these four values (round off after the decimal), we have the final output of -1 which means the grid size should be decreased by one for the coming cycle.

Here we use an example to illustrate the grid size adjustment of LE-MAC. Suppose that a node A uses a grid size of 16×16 and, in the previous cycle, wakes up 248 time frames with 192 out of which being receiving time frames. This means AWR = 248/31 = 8 and DRR = 192/256 = 0.75. Referring to Fig. 6, the associated linguistic variable for AWR = 8 is *moderate large* with membership value of 1. Similarly, referring to Fig. 7, the associated linguistic variables for DRR = 0.75 is *moderate heavy* and *extremely heavy* with membership value of 0.75 and 0.25, respectively. With these linguistic variables, we can find two applicable fuzzy rules. The GAV membership value for the rule **If** *AWR is "moderate large" and DRR is "moderate heavy"* **Then** *sensor node is "moderate insufficient"* is given by MIN(1, 0.75) = 0.75 while

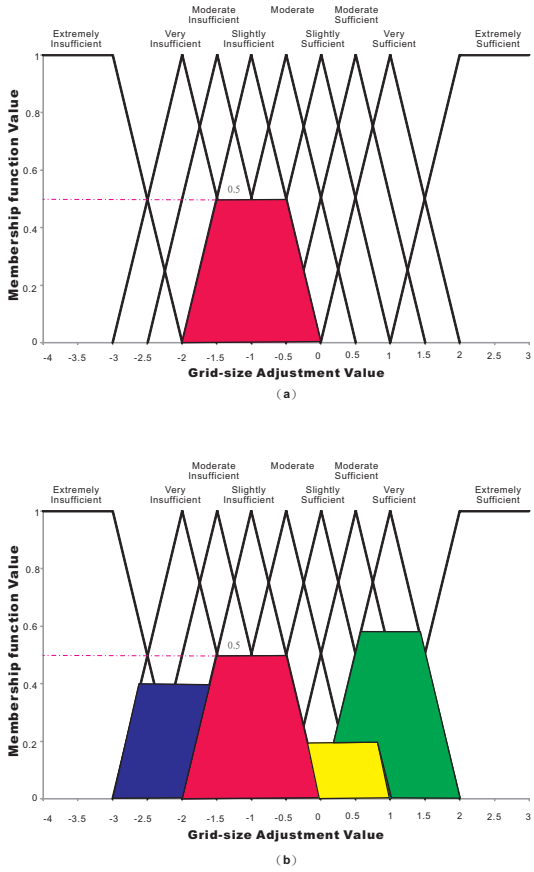

Figure 10: (a)The area obtained from a fuzzy rule by using the Min method (b)The four areas obtained from four different rules

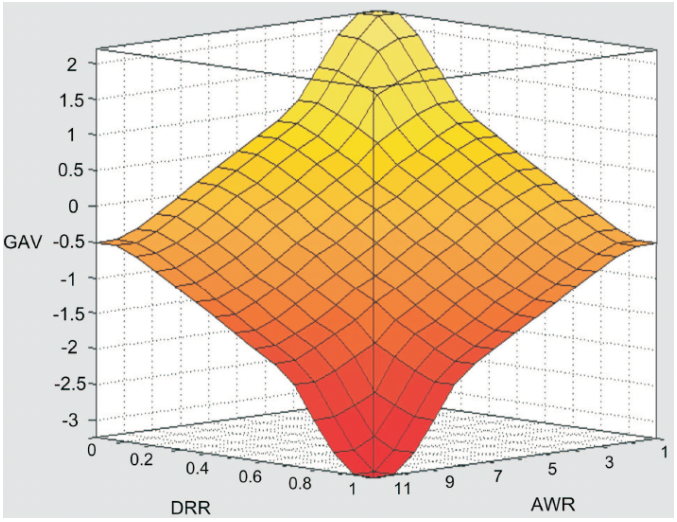

Figure 11: Effect of AWR and DRR on GAV

the GAV membership value for the rule **If** *AWR is "moderate large" and DRR is "extremely heavy"* **Then** *sensor node is "very insufficient"* is MIN(1, 0.25) = 0.25. A marked area similar to Fig. 10(a) can be found for each of the GAV membership vales. The corresponding GAV, mapping from the centroid of the marked area, is -1.5 and -2 for linguistic variables *moderate insufficient* and *very insufficient*, respectively. The final output is -2 which indicates node A will use a 14×14 grid for the next cycle.

We have verified the correctness of our design through MATLAB 7.0 Fuzzy Inference System. The GAV for different AWR and DRR is shown in Fig. 11. As expected, higher AWR and DRR produce negative GAV such that the grid size being used will be reduced while lower AWR and DRR produce positive GAV to increase the grid size being used. An improper inference design will produce a surface that is rough and uneven. The smooth surface in Fig. 11 indicates a correct and reasonable design of the LE-MAC protocol.

5 Performance Evaluation

We have implemented a simulator using ns-2 to evaluate the performance of the proposed LE-MAC protocol. The QMAC_LR protocol was also implemented for comparison purposes. In our simulations, 400 sensor nodes were randomly placed within a circular area of radius 250 meters. The maximum transmission range of a sensor node is 75 meters which makes the network consisting of four coronas. The channel

capacity is 10 kbps. Each packet is 64 bytes long. The initial grid size for nodes in the coronas C_1 to C_4 is 2×2, 8×8, 17×17, and 42×42, respectively. A time frame is set to 100 milliseconds long. We employed the energy consumption model described in [13], where the power consumption for transmit, receive, idle, and sleep modes was 0.66, 0.395, 0.35, and 0 W, respectively. Each sensor node has an initial energy of 100 Joules. A spot in the following figures shows the average of 50 simulation runs with each simulating 1000 seconds.

We define two reporting models in our simulations:

- Periodical reporting: Each node generates a packet every 5 seconds. This model produces a heavy-loaded network.
- Dynamic regional reporting: The network is evenly partitioned into six fan-shaped regions. One to three regions can be selected every 100 seconds. A packet is generated every 5 seconds for a node located in the selected region(s). In our 1000-second simulation time, the number of reporting regions is sequentially set to 3, 1, 3, 1, 2, 3, 2, 3, 1, and 2.

In the following, we observe the performance of different protocols from three aspects:

A) Impact on End-to-End Delay: The results of the average end-to-end delay a packet experienced for different reporting models are shown in Fig. 12. In the periodical reporting model, as shown in Fig. 12(a), the differences between LE-MAC and QMAC_LR are not large. This indicates that QMAC_LR performs well in a uniform and stable network environment. The gap between LE-MAC and QMAC_LR enlarges after simulation time 600 seconds. At that time, some nodes in the first corona (C_1) deplete their energy, which burdens alive nodes in C_1 with extra loads. QMAC_LR does not react properly to such changes and thus produces increased end-to-end delay. On the other hand, the increased delay produced by LE-MAC is limited because of the alive nodes' increased wake-up frequencies. In the dynamic regional reporting model, the delays produced by QMAC_LR fluctuate along with traffic load changes as shown in Fig. 12(b). On the contrary, the delays produced by LE-MAC remain steady.

It should be noted that LE-MAC produces higher delays at the first 100 simulation seconds because it takes a little time for each node to find a proper active/sleep frequency.

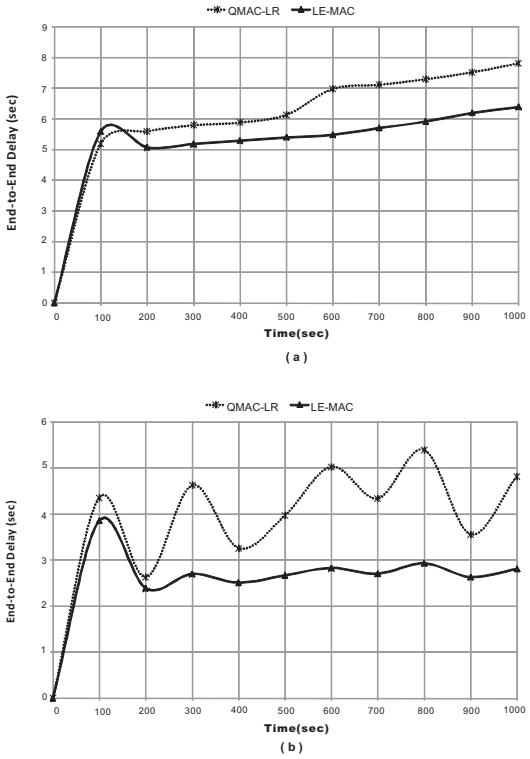

Figure 12: Impact on end-to-end delay of (a) periodical reporting and (b) dynamic regional reporting

B) Impact on Energy Consumption: Next, we observe average energy consumption to successfully transmit a packet in different protocols. The results can be found in Fig. 13. As expected, LE-MAC performs better since it enables sensor nodes to adjust their active/sleep frequencies according to their experienced traffic loads. In Fig. 13(a), we found that the energy consumption increases in proportional to simulation time. We believe this increasing results from more collisions produced by the accumulated traffic. In the dynamic regional reporting model, as shown in Fig. 13(b), the energy consumption of QMAC_LR also fluctuates. At simulation time 200, 400, and 900 seconds, since only one region generates traffic, more energy is wasted due to a lot of idle listening. On the contrary, LE-MAC consumes energy in an efficient and stable way. LE-MAC avoids idle listening by allowing nodes to reduce their active time frames when they have little traffic.

C) Impact on Successful Transmission Ratio: In this experiment, we examine the successful transmission ratio achieved through different protocols. This ratio is defined as the number of successful received packets to that of transmitted ones. As shown in Fig. 14, the differences between LE-MAC and

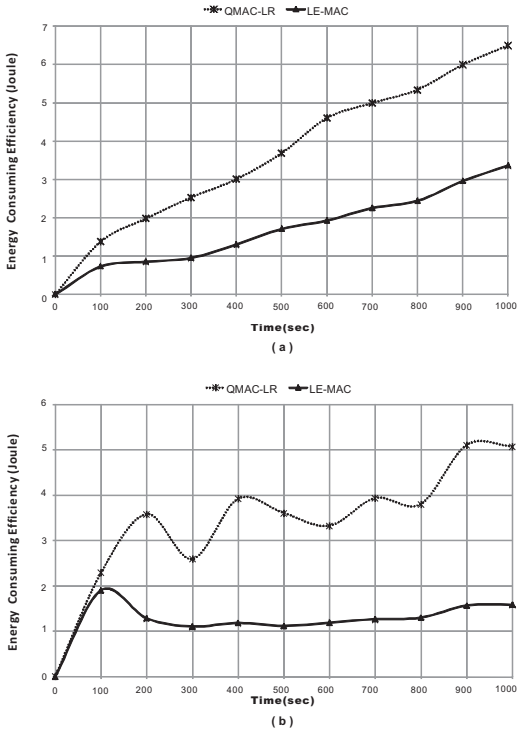

Figure 13: Impact on energy consumption of (a) periodical reporting and (c) dynamic regional reporting

QMAC_LR are not large. This implies that LE-MAC achieves better performance in end-to-end delay and energy consumption without sacrificing successful transmission ratio.

6 Conclusion

To conserve energy is essential in wireless sensor networks. A fixed active/sleep frequency does not provide enough flexibility to fit in practical network environments where traffic loads may vary over time. In this paper, we provide a load-aware mechanism to conserve energy. The proposed LE-MAC protocol utilizes the fuzzy control mechanism to dynamically adjust the frequency of active time frames. This enables a sensor node to conserve more energy when it is lightly loaded and to reduce end-to-end delay when it is heavily loaded. Simulation results verify the superiority of our LE-MAC. We consider the proposed protocol is a promising one to be applied in wireless sensor networks.

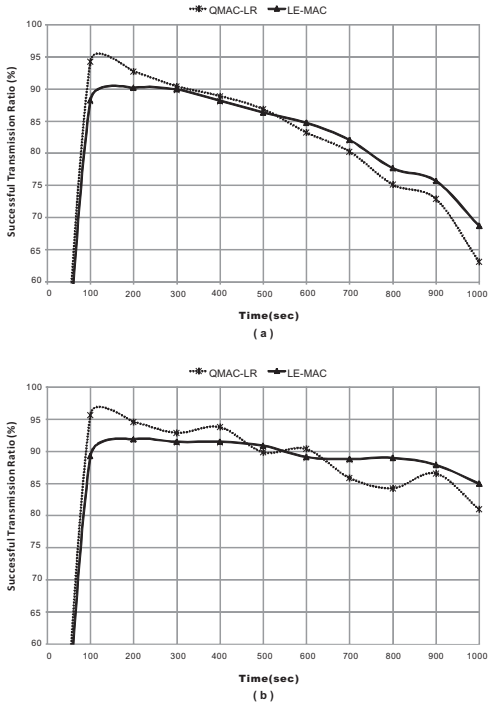

Figure 14: Impact on successful transmission ratio of (a) periodical reporting and (c) dynamic regional reporting

References

[1] D. Agrawal and A. El Abbadi. An Efficient and Fault-Tolerant Solution for Distributed Mutual Exclusion. *ACM Transactions on Computer Systems*, 9(1):1–20, Feb. 1991.

[2] D. Braginsky and D. Estrin. Rumor Routing Algorithm For Sensor Networks. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, WSNA '02*, pages 22–31, Atlanta, 28 Sept. 2002.

[3] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, Sensys '06*, pages 307–320, Boulder, 31 Oct. - 3 Nov. 2006.

[4] C.-M. Chao and Y.-C. Chang. A Power-Efficient Timing Synchronization Protocol for Wireless Sensor Networks. *Journal of Information Science and Engineering*, 23(4):985–997, July 2007.

[5] C.-M. Chao and Y.-W. Lee. A Quorum-Based Energy Saving MAC Protocol Design for Wireless Sensor Networks. *IEEE Transactions on Vehicular Technology*, 59(2):813–822, Feb. 2010.

[6] S. Y. Cheung, M. H. Ammar, and M. Ahamad. The Grid Protocol: A High Performance Scheme for Maintaining Replicated Data. *IEEE Transactions on Knowledge and Data Engineering*, 4(6):582–592, Dec. 1992.

[7] A. El-Hoiydi and J.-D. Decotignie. WiseMAC: An Ultra Low Power MAC Protocol for the Downlink of Infrastructure Wireless Sensor Networks. In *Proceedings of the Ninth International Symposium on Computers and Communications 2004, ISCC '04*, volume 2, pages 244–251, Alexandria, Egypt, 28 June - 1 July 2004.

[8] J.-R. Jiang, Y.-C. Tseng, C.-S. Hsu, and T.-H. Lai. Quorum-Based Asynchronous Power-Saving Protocols for IEEE 802.11 Ad Hoc Networks. *Mobile Networks and Applications*, 10(1-2):169–181, Feb. 2005.

[9] W. R. H. J. Kulik and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom '99*, pages 174–185, Seattle, 15-20 Aug. 1999.

[10] Y.-C. Kuo and S.-T. Huang. A Geometric Approach for Constructing Coteries and k-Coteries. *IEEE Transactions on Parallel and Distributed Systems*, 8(4):402–411, April 1997.

[11] S. Lang and L. Mao. A Comparison of Two Torus-Based K-Coteries. In *Proceedings of the 1998 International Conference on Parallel and Distributed Systems, ICPADS '98*, pages 300–306, Tainan, Taiwan, 14-16 Dec. 1998. IEEE Computer Society.

[12] J. Li and P. Mohapatra. An Analytical Model For The Energy Hole Problem In Many-To-One Sensor Networks. In *Proceedings of the IEEE 62nd Vehicular Technology Conference*, volume 4, pages 2721–2725, Dallas, 25-28 Sept. 2005.

[13] G. Lu, B. Krishnamachari, and C. S. Raghavendra. An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks. In *Proceedings of the 18th International Parallel and Distributed Processing Symposium, IPDPS '04*, pages 224–231, Santa Fe, 26-30 April 2004.

[14] M. Maekawa. A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems. *ACM Transactions on Computer Systems*, 3(2):145–159, May 1985.

[15] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, Sensys '04*, pages 95–107, Baltimore, 3-5 Nov. 2004.

[16] T. J. Ross. *Fuzzy Logic with Engineering Applications, 2nd Edition*, pages 481–485. John Wiley & Sons, Inc., Hoboken, New Jersey, USA, 2nd edition, 2004.

[17] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie. Protocols for Self-Organization of a Wireless Sensor Network. *IEEE Personal Communications*, 7(5):16–27, Oct. 2000.

[18] I. Stojmenovic and S. Olariu. *Handbook of Sensor Networks: Algorithms and Architecture*, chapter Data-Centric Protocols for Wireless Sensor Networks, pages 417–456. John Wiley & Sons, Inc., 2005.

[19] R. H. Thomas. A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases. *ACM Transactions on Database Systems*, 4(2):180–209, June 1979.

[20] T. van Dam and K. Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, Sensys '03*, pages 171–180, Los Angeles, 5-7 Nov. 2003.

[21] X. Wu, G. Chen, and S. K. Das. Avoiding energy holes in wireless sensor networks with nonuniform node distribution. *IEEE Transaction on Parallel and Distributed Systems*, 19(5):710–720, May 2008.

[22] Y.-T. Wu, Y.-J. Chang, S.-M. Yuan, and H.-K. Chang. A New Quorum-Based Replica Control Protocol. In *Proceedings of the 1997 Pacific Rim International Symposium on Fault-Tolerant Systems, PRFTS '97*, pages 116–121, Taipei, Taiwan, 15-16 Dec. 1997. IEEE Computer Society.

[23] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom)*, pages 214–226, New York, June 2002.

[24] W. Ye, J. Heidemann, and D. Estrin. Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, June 2004.

[25] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965.

[26] T. Zheng, S. Radhakrishnan, and V. Sarangan. PMAC: An adaptive energy-efficient MAC protocol for Wireless Sensor Networks. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, IPDPS '05*, pages 65–72, Denver, 4-8 April 2005.