Ann. Telecommun. (2010) 65:777–794
DOI 10.1007/s12243-010-0173-3

A load-aware energy-efficient and throughput-maximized asynchronous duty cycle MAC for wireless sensor networks

Muhammad Mostafa Monowar · Muhammad Mahbub Alam · Md. Obaidur Rahman · Choong Seon Hong · Sungwon Lee

Received: 25 December 2009 / Accepted: 12 April 2010 / Published online: 7 May 2010

Abstract Being a pivotal resource, conservation of energy has been considered as the most striking issue in the wireless sensor network research. Several works have been performed in the last years to devise duty cycle based MAC protocols which optimize energy conservation emphasizing low traffic load scenario. In contrast, considering the high traffic situation, another research trend has been continuing to optimize both energy efficiency and channel utilization employing rate and congestion control at the MAC layer. In this paper, we propose A *L*oad-aware *E*nergy-efficient and *T*hroughput-maximized Asynchronous Duty Cycle MAC (LET-MAC) protocol for wireless sensor networks to provide an integrated solution at the MAC layer considering both the low-and high-traffic scenario. Through extensive simulation using ns-2, we have evaluated the performance of LET-MAC. LET-MAC achieves significant energy conservation during low traffic load (i.e., no event), compared to the prior asynchronous protocol, RI-MAC, as well as attains optimal throughput through maximizing the channel utilization and maintains lower delay in regard to the CSMA/CA-like protocol during a high volume of traffic (i.e., when an event occurs).

Keywords Wireless sensor networks · Energy conservation · MAC protocol

M. M. Monowar · M. M. Alam · Md. O. Rahman · C. S. Hong (✉) · S. Lee
Computer Engineering Department, Kyung Hee University, Seoul, South Korea
e-mail: cshong@khu.ac.kr

M. M. Monowar
e-mail: monowar@ieee.org, hemal.cu@gmail.com

M. M. Alam
e-mail: mahbub@networking.khu.ac.kr

Md. O. Rahman
e-mail: mdobaidurrahman@gmail.com

S. Lee
e-mail: drsungwon@khu.ac.kr

1 Introduction

The Prodigious proliferation of Micro-Electro-Mechanical Systems (MEMS), [1] as well as wide adoption of wireless networking technologies, has created a great opportunity for wide spread utilization of various innovative sensor network applications in the foreseeable future. The growing demand of these applications also necessitates designing effective communication protocols for Wireless Sensor Networks (WSNs). However, designing such protocols faces various challenges due to the resource constraints of WSNs; more importantly, energy constraint and bandwidth constraint. Therefore, along with higher energy efficiency, a protocol also needs to achieve higher throughput, utilizing the maximum channel capacity.

A broad range of remote sensing applications (structural and habitat monitoring, fire detection, target recognition and tracking) necessitate WSNs to operate in a dormant state (i.e., absence of event) for a long period of time and in crisis state for short periods (i.e., occurrence of event) [2]. During the dormant period, nodes generate data at a lower rate (i.e., heartbeat messages), while, in crisis state, nodes report the event at a higher rate.

Considering energy as the most crucial resource, several duty cycle-based MAC protocols have been

proposed [3, 4], aiming to maximize the energy conservation. These protocols focus on reducing the *idle listening* (unnecessarily turning on the radio while no transmission/reception is going on) as much as possible through keeping the nodes asleep periodically, during the long dormant period. However, these studies overlooked the consequences of the high rate event traffic within the limited channel capacity. In contrast, there exist a number of works in the literature, which aim to handle the high rate event traffic adopting both the rate and congestion control in the MAC layer [5–8]. These works try to avoid the packet losses at high traffic, and optimize both the energy consumption and channel utilization. As a result, most of the existing works address the design of the MAC layer for energy-conserving low traffic, and throughput maximizing high-traffic situations separately. However, because a single application requires support for both these traffic situations, a traffic-aware integrated solution is, therefore, required at the MAC layer, which to the best of our knowledge, remains un-addressed.

In this paper, we focus to design a MAC protocol that, on one hand, maximizes the energy conservation at low traffic, and, on the other hand, maximizes the network throughput at high traffic with minimum possible energy consumption. The design of such a protocol, however, needs to handle two conflicting goals: i) maximizing the sleeping time of a node to increase the duty cycle, which might keep the channel idle, and ii) maximizing the channel utilization to increase the network capacity, and to support high data rates reducing the sleeping time. Furthermore, we aim to adapt the duty-cycling at high traffic along with avoiding the packet loss to achieve high-energy efficiency. Therefore, it is necessary to balance the energy conservation and network throughput based on the observed network traffic condition.

Existing duty-cycle based MAC protocols can be categorized as synchronous and asynchronous protocols based on their wake-up schedule for communication. Synchronous protocols [9–12] incur high cost for synchronization and maintain fixed wake-up intervals for communications. Asynchronous protocols [13–15] usually avoid this cost by employing a preamble-based transmission approach. Because the synchronous protocols allow the nodes to wake up simultaneously (and an increased collision rate at high traffic), and most of the asynchronous protocols use a long preamble (and hence, unnecessarily keep the medium busy), they usually achieve a very low medium utilization. Therefore, these protocols are not suitable to handle a high traffic situation. However, a recent work proposes a receiver-initiated duty-cycle MAC to minimize the control overhead on medium occupancy [16]. This work has motivated us to extend the duty cycle-based MAC for gaining higher throughput. Moreover, fairness is one of the core problems that need to be handled at the MAC layer during a crisis state. More specifically, here, we focus on attaining fair throughput among the nodes during a crisis state so that the sink can get unbiased information about the monitored area.

This paper introduces LET-MAC, a novel MAC protocol which integrates the objective of the duty-cycle MAC with MAC-based rate control protocols using a unified mechanism. Handling both the situations using a single mechanism has paid exiguous attention so far. We devise a receiver-driven medium access mechanism which provides higher energy efficiency, high medium utilization, and reduces the control overhead. We introduce a novel bi-directional rate control mechanism with the aim of achieving optimal performance considering the traffic situation, which, in one direction, controls the beacon-sending rate of a node based on the upstream node's request, and, in another direction, controls this rate along with reporting rate of the sources based on the contention and congestion situation observed around the node itself. Moreover, we perform extensive simulations using ns-2 to evaluate the performance of LET-MAC.

The remainder of the paper is organized, as follows: Section 2 summarizes the related works. Some preliminaries and design goals of LET-MAC are stated in Section 3. The detailed design of LET-MAC is presented in Section 4. Section 5 demonstrates the performance of our protocol, evaluated using ns2, and finally, in Section 6, we present our concluding remarks.

2 Related work

Energy-efficient duty cycle MAC protocols have been a very prominent research area in wireless sensor networks. Among the asynchronous protocols, B-MAC [13], developed at University of California, Berkley, is a contention-based CSMA like-approach which exploits Low Power Listening (LPL) and an extended preamble for energy-efficient communication. Nodes maintain a fixed periodic wake-up schedule to check for channel activity. If any activity is detected during channel sampling, it powers up and receives the data. The sender, on the other hand, sends a long preamble which lasts longer than the receiver's wake-up interval to ensure the reception of data by the receiver. B-MAC exhibits higher energy efficiency in low-traffic loads, since, after each wake-up, it performs very short channel activity checking. However, it suffers from an

overhearing problem and shows higher latency due to the long preamble transmission.

X-MAC [14], is the optimized version of B-MAC that overcomes the overhearing problem with the introduction of short-strobed preambles and inclusion of the target receiver address in the preamble. This facilitates the reply of early acknowledgement to the sender, and, thereby conserves energy with reduced per-hop latency. Yet, at each wake-up, every node has to perform long clear channel assessment (CCA) check in LPL, (longer than the ACK waiting period), which causes energy wastage if no data is available at the sender. Moreover, X-MAC also occupies the channel for a long time with preamble transmission.

BoostMAC [15] is another optimized version of B-MAC that provides an adjustable interface to achieve ultra-low power operation. Considering Habitat monitoring as the motivating application, BoostMAC adapts with the changing traffic load through adjustment of two parameters, polling interval and preamble length. The polling interval adjustment is based on selecting some predetermined polling time values stored in an array using Additive increase Multiplicative decrease (AIMD) techniques rather than computing the actual channel polling times. It exploits machine learning to dynamically adjust the preamble length. Despite the efforts in adapting with the traffic load, BoostMAC fails to achieve high channel utilization due to employing preamble based approach. Moreover, it does not consider the consequences of high traffic situation, and lacks the functionalities such as rate control, congestion control, without which the maximum sustainable throughput cannot be achieved during high traffic situation.

RI-MAC [16] handles the problems of long channel occupancy by the senders observed in B-MAC and X-MAC, introducing receiver-initiated transmission in which data transmission is initiated through a beacon transmission by the receiver. RI-MAC achieves better energy efficiency, delivery ratio, and lower latency, compared to X-MAC. However, like the prior asynchronous protocols, RI-MAC also shows non-adaptive behavior in maintaining the wake-up interval, and the channel utilization of RI-MAC during extreme conditions is still unknown which we have addressed while designing LET-MAC.

PTIP [17] and Koala [18] also use the receiver-initiated transmission for WSN; however, the former one is designed for infrastructure-based wireless sensor networks, and the latter one uses this concept using the gateway node to wake up the sensor nodes for downloading bulk data instead of actual data transfer. Besides WSN, the receiver-initiated MAC concept was proposed by Garcia-Luna-Aceves et al. [19] for general wireless networks in which collision handling had been given foremost importance above energy efficiency.

Synchronous approaches, such as S-MAC [9], T-MAC [20], R-MAC [10], DW-MAC [12], and also hybrid approaches, like SCP-MAC [21], maintain a fixed wake-up interval for communication, and require strict time synchronization among the nodes to perform synchronous wake-up and sleep. Communication is performed during the short active period of an operational cycle, reducing the idle listening problem. However, the maintenance of synchronization among the nodes is a drawback in terms of overhead and supporting network scalability. Therefore, due to the simplicity and scalable feature, asynchronous approaches are particularly attractive in the context of WSNs.

The current literature regarding the congestion/rate control protocol for WSNs reduces the energy wastage caused by packet drops by imposing a sustainable rate to the sources and utilizing the maximum channel capacity. Due to the dependency of MAC layer information for congestion control, MAC-based distributed congestion control protocols are developed such as CODA [6], FUSION [22], IFRC [5], CCF [8], and, PCCP [7] and are more effective in the wireless sensor network environment. Nonetheless, these protocols are developed based on the CSMA/CA-like "Always on" MAC protocol. The effect of duty-cycling on congestion/rate control in the crisis state has not been considered in those protocols, which could be a serious problem in detecting and handling congestion due to the sleep schedule maintenance in that situation. Therefore, LET-MAC integrates this functionality for gaining higher network efficiency, along with providing duty-cycling.

3 Preliminaries and design goals

We have studied the channel utilization of duty cycle MAC, in comparison with full-active CSMA/CA through performing a simulation. The general simulation setup is explained in Section 5. Unless otherwise mentioned, we consider a multi-hop network throughout the paper where some nodes act as sources (which generate data), some act as forwarding nodes (forward other node's data) and some play both roles, and the nodes forward data to the sink. Twenty nodes are chosen from the margin as sources, and per node average goodput is measured varying the data rate of the sources. The per node average goodput is defined as the average number of successful packet reception per second by the sink from each source node. In this

study, as a synchronous MAC, we choose S-MAC with adaptive listening, and X-MAC and RI-MAC are chosen as asynchronous protocols, where the former uses a preamble-based, and the latter employs a receiver-initiated mechanism. As the Fig. 1 shows, full-active CSMA/CA achieved the maximum per node goodput, which is 2.2 pps. RI-MAC achieved the second-best per node goodput (1 pps), which is around 54% less than CSMA/CA. X-MAC obtained the lowest average goodput (0.05 pps), whereas S-MAC achieved goodput in between RI-MAC and X-MAC (0.75 pps).

The study gives the following observations:

- Preamble based asynchronous approach saturates the network for a long duration of preamble transmission, which not only increases the medium access delay but also increases collision loss and congestion loss during a high traffic load.
- As the offered load increases, synchronous-based approaches suffer from huge contention and thereby collision losses due to synchronous wake-up by the nodes. The small active period for maintaining a higher duty cycle exacerbates the contention situation during the high traffic rate, resulting in lower goodput.
- A receiver-initiated asynchronous approach shows better performance than the other two during high traffic load due to the avoidance of unnecessary channel occupancy and provision of back-to-back data transmission; however, the achievable goodput is still far from full-active CSMA/CA. The reason is that, during high traffic, a huge number of beacon and data packet collisions occur, which forces the receiver to go into the sleep state. Moreover, the long sleep interval, irrespective of the traffic situation, builds up the congestion at the sender side.

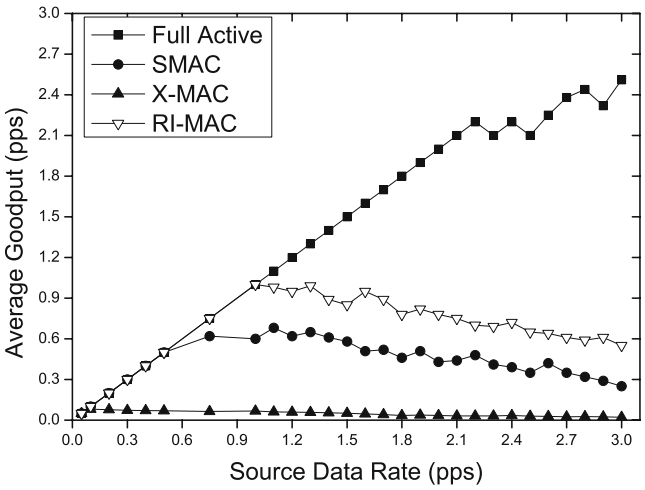


Fig. 1 Average goodput vs offered load

The observations direct us to reach the following conclusion:

In comparison with the existing duty cycle MAC protocols, the receiver-initiated asynchronous approach could be the better choice to achieve higher throughput. However, the lower utilization of the channel capacity of this approach, as compared to full-active CSMA/CA in a crisis state necessitates a mechanism which will reduce the collision, avoid the congestion, as well as perform adaptive rate control along with duty cycle adjustment to achieve near optimal throughput while keeping the nodes in sleep as much as possible.

Nonetheless, adopting any existing rate control protocol on the top of RI-MAC also will not work; since these protocols are designed based on the full-active CSMA/CA-like protocols and could detect false congestion and perform unnecessary rate control, keeping the channel capacity underutilized due to the sleep schedule maintenance by the receiver, while data is available to the sender.

Furthermore, the existing duty cycle protocols (synchronous or asynchronous) do not maintain their duty cycle based on the availability of traffic, which cause unnecessary wake up and energy wastage. Hence, the overall design goals of LET-MAC are:

1. *Energy Efficiency.* Our primary goal is to achieve significant improvement in energy savings during very low traffic load, compared to the current duty cycle protocols, as well as to reduce the energy dissipation, avoiding packet drops at high traffic load.
2. *Application Fidelity Enhancement.* Our second goal is to boost the application fidelity while it is required at the time of event occurrence and nodes generate data at a high rate. This situation demands to attain a certain delivery ratio, higher throughput, and lower latency.
3. *Network Efficiency.* Another major goal of LET-MAC is to run the network at an efficient operating point. More specifically, we intend to avoid the congestion collapse, as well as reduce the collision losses, keeping the rate as high as possible during the crisis state through efficient rate control.
4. *Robustness.* We further wish LET MAC to be robust to routing dynamics and nodes joining and leaving the network. Thus, in addition to event occurrence, traffic load also could vary due to the route change as well as node location change. Our proposed protocol aims to adapt these changes dynamically.

4 Design of LET- MAC

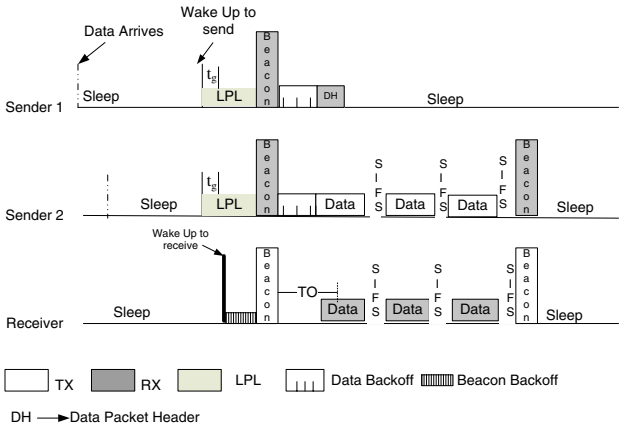
To achieve the design goals, we devise several mechanisms of LET-MAC as discussed in the subsequent sections.

4.1 Receiver-driven medium access

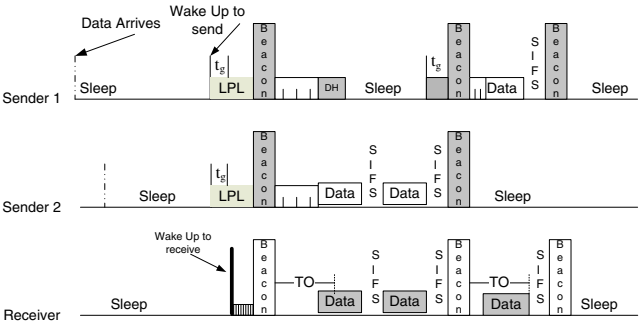
We adopt the receiver-driven mechanism for medium access, as proposed in RI-MAC. However, the proposed medium access mechanism differs, in particular, in terms of packet reception, apposite sleeping time determination, number of control packets disseminated, as well as in back-off strategy. To efficiently portray the mechanism we describe it according to the role the nodes play either as a receiver or a sender. The medium access mechanism is portrayed in Fig. 2.

Receiver's Operation Each node periodically wakes up according to its own wake-up interval to receive packets from the upstream nodes. After turning on the radio, a node broadcasts a beacon (referred as a primary beacon), after performing a short backoff (BO_b), followed by a CCA check, to avoid the collision from simultaneous beacon packet transmissions by the neighboring nodes. It then waits to receive packets from any of its upstream senders until a time-out (TO) occurs. The receiver goes to sleep if it detects the channel busy during beacon transmission or detects collision during waiting for packets or no packets arrived after TO occurs. If a node finds the channel busy for a consecutive number of times, it might shift its wake-up time to avoid the ongoing transmission, which coincides with the wake-up time of that node.

We opt for multiple-packet reception at each reception round (which starts after transmitting the beacon) for high-medium utilization. The number of allowable packets received for a reception round is denoted as N_r. The receiver embeds this value into the beacon packet. The successful reception of the packets from a particular sender is acknowledged through another beacon, denoted as ACK-Beacon, which acts as a block acknowledgement. However, if the receiver receives fewer number of packets than the specified value, then it updates the remaining number of packets to be received in the ACK-Beacon packet, which invites other potential senders having data to send packets to utilize that reception round. Thus, the ACK-Beacon serves two purposes; first, it acts as an acknowledgement (block-ack), and second, it solicits data packets to utilize the reception round. A receiver goes to sleep after full utilization of the reception round.



(a) The value of N_r is set as 3 as for example. Sender 2 wins the channel and utilizes the reception round of the receiver fully. Sender 1 goes to sleep after overhearing the first packet header of Sender 2.



(b) The value of N_r is set as 3 as for example. Sender 2 is the wining node but having only 2 packets in the queue. Sender 1 goes to sleep after overhearing the first packet header and again wakes up with a guard time. It sends the remaining number of packet in that reception round and goes to sleep again. Besides, sender 2 goes to sleep just after finishing its transmission.

Fig. 2 Medium access mechanism

Sender's Operation A node having data in the queue functions as a sender. Every sender is aware of its receiver's wake-up schedule. After waking up at that time, a sender waits in LPL for a receiver's beacon. Upon detecting the beacon preamble it powers up and receives the entire beacon packet.

A receiver i appends the remaining duration until its next wake-up, t_r^i, in the primary beacon or ACK-Beacon to inform the upstream senders its wake-up schedule. It is calculated as- $t_r^i = d_w^i - \left(t_b^i - t_w^i\right)$; where, d_w^i is the wake-up interval of node i, t_b^i is the time when the primary beacon or ACK-Beacon has been sent, and t_w^i is the wake-up time of the node i, and $t_b^i - t_w^i$ denotes an interval. Using the value, t_r^i, a sender, j, determines its wake-up schedule for sending data, $t_w^j(send)$ as-

$$t_w^j(send) = t_{curr}^j + \left(t_r^i - t_g\right) \tag{1}$$

where, t^j_{curr} is the current time at sender j, t_g is the guard time for clock drift and is measured as- $t_g = 2 \times R_{drift} \times t^i_r$ and, R_{drift} is the maximum clock drift rate.

Notably, if a sender misses the sending operation in its pre-scheduled sending time due to the receiver's beacon failure or collision or if it is a joining node, it continues waiting in LPL for the receiver's beacon instead of sleeping.

Upon beacon reception, the senders start their random back-off (BO_d) within a fixed contention window. Based on the current queue size, q_{curr}, and the i^{th} receiver's requisition, N^i_r, a wining node, j in the contention sends N^j_s number of packets with a gap of Short Interframe Space (SIFS) between successive packets according to-

$$N^j_s = \begin{cases} q_{curr}, & if q_{curr} \leq N^i_r \\ N^i_r, & otherwise. \end{cases}$$

The wining node goes to sleep immediately after receiving the ACK-Beacon. In contrast, the losing nodes will be in the sleep state if they find $N^j_s = N^i_r$ through overhearing the first data packet header (which contains N^j_s). However, for the case, $N^j_s < N^i_r$, the losing nodes remain awake instead of sleeping if $N^j_s < threshold$, to reduce the energy cost caused by frequent turning on/off the radio. We set the threshold as 2. If $N^j_s \geq threshold$, then the losing nodes go to sleep. They wake up again just before the ACK-Beacon reception time (intended for the wining sender), with a guard time t_g, and wait for it in LPL. The nodes can easily estimate the sleeping duration before ACK-Beacon comes through, overhearing the value, N^j_s, from the packet header. In every case, whenever a node looses the channel, it freezes its backoff to gain higher priority in accessing the channel for the next contention, which addresses the fair access of the medium.

4.2 Bi-directional rate control

The major challenges of LET-MAC is to obtain the maximal energy savings during the idle period of the sensors and achieve the near optimal throughput during crisis state with minimum energy wastage. We address these challenges through introducing a novel rate control mechanism, referred to as bi-directional rate control (BRC), which is a combination of the following two mechanisms;

- *Forward Rate Control(FRC).* The beacon sending rate of the downstream node is controlled according to the request by the upstream nodes.
- *Backward Rate Enforcement(BRE).* The downstream node enforces reduction in the forwarding rate of the upstream nodes, as well as creates backpressure for source rate reduction.

Since, in our proposed receiver driven medium access mechanism, a node broadcasts its primary beacon after wake up at a particular interval, hence controlling the wake-up interval is analogous to the control of its beacon interval (or beacon rate), and vice-versa. Therefore, throughout the paper, the term wake-up interval and beacon interval have been used interchangeably.

4.2.1 Forward rate control

With receiver-driven transmission technique, the forwarding rate of the upstream nodes depends on the beacon sending rate of the downstream receiver. As we employ multiple-packet reception at each reception round, thus, this number also affects the forwarding rate. In this mechanism, both of these factors are controlled according to the request performed by the upstream nodes, based on the prevailing traffic situation. This mechanism includes the following operations:

A. *Adaptive Multiple Packet Reception* A receiver receives multiple packets in succession according to the load observed at its upstream nodes. During low traffic load, a receiver might receive all of the queued packets at its upstream senders. As the traffic load at the upstream sender increases, it demands excessive packet reception, which, however, might increase the medium access delay and, thereby, collision. Hence, a maximum limit should be provided, addressing those factors denoted as N^{max}_r. Therefore, as long as the load at the upstream senders persists within this limit, a receiver could receive all of the packets for higher medium utilization.

Algorithm 1 exhibits the adjustment of multiple-packet reception for the i^{th} node (N^i_r) in a particular reception round. The initial value of the N^i_r field is set as N^{max}_r to force the upstream senders transmitting all the queued packets in order to get an idea of the current load (Line 1 in Algorithm 1). The senders include their corresponding current load in terms of packet rate, L_{curr} in the packet header, which is calculated as-

$$\begin{aligned} L_{curr} &= Packet\,Reception\,Rate + Packet\,Generation\,Rate \\ &= \frac{N_r}{d_w} + \frac{1}{T_d} \end{aligned} \quad (2)$$

where d_w is the wake-up interval of a node and T_d is the data generation interval. A receiver i then esti-

mates the observed load per wake up at its upstream senders as-

$$OL^i = \left\{ \sum_{j \in S(i)} L^j_{curr} \right\} \times d^i_w \tag{3}$$

where, $S(i)$ is the set of upstream senders of node i and d^i_w is the wake up interval of node i. Node i sets the

Algorithm 1 N^i_r Value Adjustment

1. *Set*, $N^{init}_r = N^{max}_r$
2. Set OL^i according to Eq. 3
3. **if** $(OL^i \geq N^{max}_r)$ **then**
4. $N^i_r = N^{max}_r$
5. **else**
6. $N^i_r = OL^i$
7. **end if**

value of N^i_r as N^{max}_r, when the observed load exceeds or is even with N^{max}_r, otherwise, the current observed load is selected as N^i_r.

B. *Traffic Rate Aware Wake-up Scheduling* The Traffic Rate Aware Wake-up Scheduling (TWS) determines the wake-up/beacon interval of downstream nodes based on the upstream node's request. The fixed [9, 13, 14] or random wakeup interval [16], observed in the literature failed to reflect the current traffic load which achieved neither higher energy efficiency during very low traffic nor obtained the higher throughput during the high traffic situation. Hence, we introduce TWS which selects the wake-up interval of a node i, d^i_w as-

$$d^i_w = \min\left(d^j_w\right);\ j \in S(i) \tag{4}$$

For this operation, upstream senders append the data generation interval (if it is source node only) or wake-up interval (if it is forwarding node only) or minimum of the packet generation interval and wake-up interval (if it acts both as source and forwarding node) in the interval field of the data packet header while sending packets to the receiver. We assume that the MAC layer knows the data generation interval. Thus, the wake-up interval is selected according to the minimum wake-up interval or data generation interval of the upstream senders, according to Eq. 4. However, initially, each node in the network could choose their wake-up interval according to the maximum data generation interval (decided by the application) or any random value and eventually update it according to TWS.

TWS achieves the maximum energy efficiency during low traffic load because nodes remain in the sleep state for a long period of time during the dormant state and expect to receive a packet from any of its upstream senders at each wake-up, thus avoiding unnecessary wake-up. On the other hand, during the crisis state, nodes wake up at a high rate, which also increases the beacon sending rate to achieve higher throughput. Furthermore, this reduces the per-hop latency, and, for multi-hop network, TWS affirms the non-increasing wake-up interval selection by the downstream nodes to maintain lower end-to-end delay.

C. *Towards Maximal Throughput Avoiding Queue Build Up* During high traffic situation, TWS effectively adjusts the wake-up interval of a node to achieve higher throughput. It also prevents queue build-up at the upstream nodes while the observed load is in a tolerable situation $(OL^i \leq N^{max}_r)$. However, when $OL^i > N^{max}_r$, to inhibit in building up the queue at the upstream senders, the wake-up interval needs to be adjusted for achieving maximum throughput. Hence, in that situation, to accommodate the extra traffic prevailing at its upstream senders, node i estimates required beacon sending rate $R^i_{BS}(req)$ as

$$R^i_{BS}(req) = \frac{R^i_{BS}(curr) \times OL^i}{N^{max}_r} \tag{5}$$

In Eq. 5, $R^i_{BS}(curr)$ is the current beacon sending rate of node i, and OL^i is the current observed load obtained from the algorithm 1. Based on Eq. 5, a node sets its new beacon sending rate, $R^i_{BS}(new)$ as-

$$R^i_{BS}(new) = R^i_{BS}(curr) + \alpha\left(R^i_{BS}(req) - R^i_{BS}(curr)\right) \tag{6}$$

Here, α is the stabilizing parameter. If $\alpha = 0$, then it nullifies the effect of the required rate. Again, if $\alpha = 1$, then $R^i_{BS}(new) = R^i_{BS}(req)$, which could result a sharp increase of the beacon sending rate instantaneously, perhaps creating an oscillating situation. Therefore, the value should be, $0 < \alpha < 1$, and based on the simulation, we set the value as 0.35.

While the new rate approaches the required rate eventually, the beacon sending rate of a node increases with the aim of gaining highest possible throughput during crisis state. This situation also allows the nodes to sleep, although for short duration. However, to provide energy efficiency, a node might remain awake instead of a sleep if $E^{d_w}_l \leq E_{onoff}$ holds true; where, $E^{d_w}_l$ is the energy consumption for listening up to the wake-up interval, and E_{onoff} denotes the energy consumption for turning the radio on and off.

4.2.2 Backward rate enforcement

Unlike the forward rate control, where the downstream node's packet reception rate is adjusted based on the traffic situation at upstream nodes, the backward rate enforcement mechanism enforces the reduction of the forwarding rate of the upstream nodes and the reduction of source rate through creating back-pressure considering the current network status observed at the downstream node. The related operations of this mechanism include:

A. *Rate Enforcement at High Contention* The forward rate control mechanism might fail to utilize the measured beacon rate due to experiencing collision. Hence, this is not the optimal wake-up/beacon interval considering the contention scenario. Therefore, a node adjusts its beacon interval during high contention, as shown in Algorithm 2. In this mechanism, every node maintains a

Algorithm 2 Beacon Rate Adjustment in High Contention

1. Initialize CRC
2. **while** $(CRC \geq 0)$ **do**
3. **if** $R^i_{BS}(current) changes$ **then**
4. go to 1
5. **end if**
6. **if** Experience Collision **then**
7. N_f++
8. $CRC--$
9. **end if**
10. **end while**
11. **if** $(N_f \geq threshold)$ **then**
12. $R^i_{BS}(new) = R^i_{BS}(curr) - \gamma$
13. $R^i_{BS}(curr) = R^i_{BS}(new)$
14. $d^i_w(curr) = \frac{1}{R^i_{BS}(curr)}$
15. $j = 0$
16. **end if**
17. **if** $(N_f < threshold\ and\ j \leq m)$ **then**
18. $j++$
19. **if** (j==m) **then**
20. call FRC()
21. **end if**
22. **end if**

contention resolution counter (CRC) and observes the number of failed transmissions, N_f, within this counter value (line 1–9 of Algorithm 2). A failed transmission is defined as each time a node wakes up but immediately goes to sleep as it experienced collision after sending the beacon. A node reinitializes its CRC value if its current beacon interval changes. If the value of N_f becomes greater than a threshold, μ (80% of CRC), then it immediately performs an additive decrease of the beacon sending rate (line 11–12 in Algorithm 2). Additive decrease is used to avoid the aggressive rate reduction which could be caused by multiplicative decrease mechanism.

In contrast, while the value of N_f is less than μ and prolong for m consecutive times (line 17–20 of Algorithm 2), a node updates its beacon interval according to forward rate control mechanisms.

B. *Source Rate Control at Congestion* The bounded packet reception rate due to high contention and limited capacity of the wireless network could cause queue build up at a node which, in turn, causes packet loss due to overflow. Thus, source rates need to be decreased which, in turn, increases the beacon/wake-up interval of the nodes along the path to the sources.

We measure the instantaneous average queue length of a node to detect the level of congestion following the usual trend of congestion control mechanisms. [5, 6, 22]. The average is taken using the exponentially weighted moving average (EWMA) formula as

$$Avg^i_q = (1 - \varepsilon) \times Avg^i_q + \varepsilon \times inst_q \tag{7}$$

Where, Avg^i_q is the average queue length of node i, $inst^i_q$ is the instantaneous queue length of that node, and ε is the tuning parameter. The average queue length is updated after each reception round of node i.

Whenever the value of Avg^i_q exceeds a certain threshold, λ, a node sets the ECN bit in the primary beacon and ACK-Beacon packet. Subsequently, all the nodes along the path set ECN bit in the beacon packets to notify the source nodes regarding congestion, thus creating back-pressure. A node resets the ECN bit upon relinquish of congestion. Sources employ the commonly followed Additive Increase Multiplicative Decrease (AIMD) approach for rate control. Since all the sources along the path to the congested node reduce the data generation rate and increases at the same rate afterwards; fairness is maintained among the sources [23].

The detection of the congestion level and thereby performing source rate control might not avoid the packet drops for the nodes which are far away from the sources. As we employ a receiver-initiated mechanism, to fully avoid the packet drops, a receiver, i, could stop sending beacon if it detects that the queue is full. It resumes sending the beacon again when its queue has space to receive at least one packet. In this case, it sets the value, N_r, as the remaining space of its queue.

4.3 Analysis

This section presents an energy and latency model for the basic LET-MAC protocol.

A. *Energy Analysis.* Our energy analysis is based on the work presented at [21]. In this analysis, we assume a network of n nodes, where all nodes are in the transmission range of each other. For simplicity, we consider only TWS operation for duty cycle adjustment. We assume that every node generates data at a fixed interval, denoted as T_d. Thus, $d_w = T_d$. Here, we intend to perform the analysis of radio energy consumption as it is the most dominant source of energy consumption for WSNs. [3, 4]. Typically, a radio has four different states: listen, transmit, receive, and sleep; the power consumption of each can be denoted as P_l ,P_t, P_r, and P_s. Since channel polling is different from typical listening [21], we present the energy consumption for polling, P_p, separately. We can measure the energy consumption of a radio device by determining the time it stays in each state, denoted as T_l, T_t, T_r, T_p, and T_s. Thus, the expected energy consumption per node for LET-MAC can be modeled as

$$\begin{aligned} E &= E_l + E_t + E_r + E_p + E_s \\ &= P_l T_l + P_t T_t + P_r T_r + P_p T_p + P_s T_s \end{aligned} \tag{8}$$

In this analysis, we use the power and time values of CC2420 radio [24] for actual representation of the model, as shown in Table 1. The value of the beacon and data packet length, along with the corresponding back-off periods, are taken from the simulation parameters, as to be discussed in Section 5.

Table 1 Typical power and time values used in 802.15.4 radio (CC2420)

Symbol	Meaning	Value
P_l	Power in listening	56.4 mW
P_t	Power in transmitting	52.2 mW
P_r	Power in receiving	56.4 mW
P_l	Power in polling	12.3 mW
P_l	Power in sleeping	3μ W
T_g	Average guard time	3 μs
T_w	Average waiting time after sending beacon	5.11 ms
T_b	Time to transmit or receive a byte	32 μs
T_d	Data generation interval	*varying*
d_w	Wakeup/Beacon interval	*varying*
T_{bb}	Average beacon backoff time	1.3 ms
T_{db}	Average backoff time for data packet	5.1 ms
L_d	Data packet length	32 bytes
L_b	Beacon packet length	14 bytes

$$\begin{aligned} T_l &= \frac{T_{bb} + T_w}{d_w} + \frac{T_{db}}{d_w} \\ &= \frac{1}{d_w}\left(T_{bb} + T_{db} + T_w\right) \end{aligned} \tag{9}$$

$$T_p = \frac{T_g + T_{bb}}{d_w} \tag{10}$$

$$\begin{aligned} T_t &= \frac{L_d T_b}{d_w} + 2\left(\frac{L_b T_b}{d_w}\right) \\ &= \frac{1}{d_w}(L_d T_b + 2(L_b T_b)) \end{aligned} \tag{11}$$

$$T_r = \frac{(n-1)L_d T_b}{d_w} + 2\left(\frac{L_b T_b}{d_w}\right) \tag{12}$$

$$T_s = 1 - T_l - T_p - T_t - T_r \tag{13}$$

Equation 9–13 delineates the expected staying time of a node in different states. Expected listen time, T_l, incorporates the carrier sensing performed by the receiving node for the T_{bb} period before sending a beacon packet at each wake-up interval, along T_w, the waiting time for data packet arrival after sending a beacon. It also includes the listening time for the sending node, which lasts up to T_{db} period after every beacon reception. T_p in Eq. 10 is for only the senders as each sending node wakes up to send according to the receiver's wake-up schedule and to poll the channel for beacon reception. The expected transmission time in Eq. 11 is derived for the sending nodes which sends data after each beacon reception, as well as for the receiver, which transmits the primary beacon and Ack-Beacon at each reception round. Equation 12 signifies the multiple-packet reception in which, at each beacon interval, a node receives data packets from its $(n-1)$ neighbor nodes. It also includes the primary and Ack beacon reception by the sending node. Here, we assume, $N_r = (n-1)$. Finally, the expected sleeping time can be formulated by measuring the inactive time of a node, as in Eq. 13.

Substituting Eq. 9–13 into Eq. 8, we get the expected energy consumption per node for our proposed LET-MAC. Notably, the energy consumption largely depends on the primary beacon interval or wake-up interval of a node. In particular, the energy consumption is inversely proportional to the beacon interval. Again, since, this parameter is determined based on the data generation interval of a node, a similar relationship exists between energy consumption and the traffic load offered in the network.

B. *Delay Analysis.* The per-hop latency can be formulated as

$$Lat_{perhop} = \frac{d_w}{2} + T_{bb} + T_{db} + L_b T_b + L_d T_b \qquad (14)$$

Assuming the data packet length and beacon length to be fixed, per hop latency also depends on the beacon interval of the receiver. Since, with the increase of the data rate the beacon interval decreases, during high traffic situation the per-hop latency also decreases. Moreover, for the multi-hop case, the value d_w shows a non-increasing feature due to the TWS mechanism, which also exhibits lower end-to-end latency in the multi-hop scenario.

Figure 3 illustrates the relation of the wake-up interval with energy consumption and latency, based on our energy and latency model. In particular, it shows the lower energy consumption sacrificing the delay in light traffic load, while higher energy consumption with lower delay in the case of heavy traffic loads. However, considering energy as a pioneer resource for wireless sensor network, an optimal beacon interval can be obtained satisfying $\frac{d}{dd_w}(E) = 0$ in case of higher traffic load to prolong the battery lifetime, while the energy consumption exceeds a certain limit with the sacrifice of latency, which can be used as minimum limit in the beacon interval.

4.4 Parameter selection

This section discusses the determination of the value of different parameters we used in designing our protocol.

The value of N_r^{max} is a system parameter and completely depends on the topology and application requirement. We consider the channel busy probability to select this value, which also reflects the medium access delay and collision. We derive the maximum value of N_r^{max} based on the analysis performed in [25] and [26] for the IEEE 802.15.4 CSMA/CA mechanism. From their results, we formulate the channel busy probability, denoted as β, due to the transmission from neighboring nodes, as

$$\beta = \frac{n \times T_t^{avg}}{d_w - T_t^{avg}} \qquad (15)$$

Where, T_t^{avg} is the average transmission time required to transmit N_r packets, which can be obtained as

$$T_t^{avg} = N_r \times L_d T_b + (N_r - 1) \times SIFS + 2L_b T_b \qquad (16)$$

We use the parameter values for L_d, $SIFS$, L_b, and T_b, as used in simulation discussed in Section 5. Assuming 200ms[1] as d_w, keeping the channel busy probability as 75%, and putting the average number of backlogged neighbors as 5 (depends on the topology), and solving the Eq. 15 using Eq. 16, we obtain, $N_r^{max} = 3$ and use that value in the simulation.

We intend to keep the value of congestion detection threshold, λ (as discussed in Section 4.2.2 B), lower to detect an early congestion and initiate the source rate control immediately for congestion avoidance. However, a very low threshold could result in lower network utilization and a very high value also could cause delay in detecting congestion. From the simulation, we obtain that 60–70% of the maximum queue size gives a better result for early handling of congestion, which leads to setting the value of λ as 9. Hence, when the average queue length reaches 9, the source rate control mechanism is initiated.

The determination of the value of ε used in Eq. 7 should reflect both the transient congestion situation and actual queue size. If it is too large, it could overestimate the congestion situation, and a very lower value also might not reflect the actual queue size. The relationship between congestion detection threshold and the tuning parameter has been explored in [27]. Based on their analysis, and through extensive simulation, we set the value of ε as 0.1. This value detects congestion when the actual queue size reaches 63% of maximum queue length (40 as set in the simulation). Moreover, we intuitively set the value of CRC and m in the simulation, which justifies the selection.

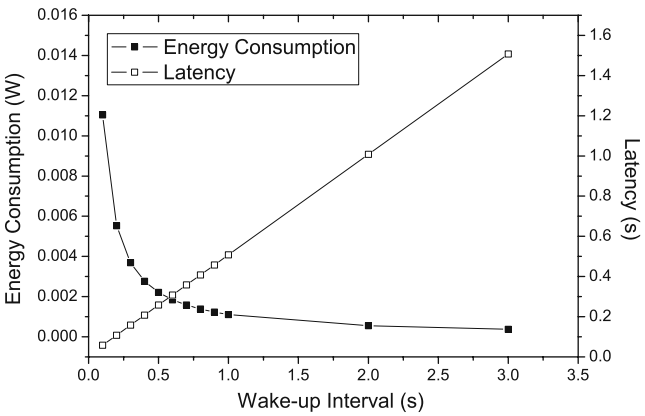


Fig. 3 Analytical results of energy and latency model varying wake-up interval

[1]The value could be taken from the maximum allowable data generation rate or minimum allowable data generation interval decided by the application.

5 Performance evaluations

This section discusses the performance evaluation of the LET-MAC protocol. The results show that LET-MAC clearly outperforms others in achieving its design goals.

5.1 Simulation environment

We perform extensive simulation of our protocol using the ns-2 network simulator. We used version 2.33 of the ns-2 simulator using the Two Ray Ground propagation model in the air and a single Omni-directional antenna commonly used with ns-2. A network of area 100m x 100m is used with 100 nodes deployment in uniform random distribution. We evaluated our protocol using two models with this network environment: a correlated event traffic model and a periodic traffic load model. In this study, we compare LET-MAC with RI-MAC and a fully active CSMA/CA protocol. Full-active CSMA/CA has been chosen, as it achieves the highest possible throughput since packet transmission is initiated as soon as it arrives to the queue, and no duty cycle is maintained. We further omit preamble-based asynchronous protocols, as RI-MAC showed its supremacy upon those protocols in their simulation. We exclude routing traffic to simplify our evaluation and assume that a routing functionality exists which selects the shortest path between any two nodes. Table 2 represents different parameters, and their corresponding values we used in our simulation. Some of the parameters (SIFS, Slot time, Data Rate, CCA Check delay) are from the data sheet of CC2420 radio [24]. The beacon size varies from minimum 96 to maximum 113 bits, depending on the Block-Ack size, along with the presence/absence of congestion control information. On the contrary, the beacon size of RI-MAC is 48–72 bits. The data packet of LET-MAC also includes 4 bytes additional data in the MAC header. This overhead, however, facilitates in performing both rate and congestion control, integrated with duty cycling without incorporating as additional component for those purposes. The sleep interval of RI-MAC is set to a random value between 0.5 to 1.5 second. We used the power consumption values of sensor nodes in different states according to Table 1. Each simulation has been performed for 60 seconds, and we averaged the values obtained for 30 random runs.

5.2 Performance metrics

In this study, we measure the effectiveness of LET-MAC, compared to the other relevant protocols using the following metrics:

Energy Consumption. The total energy consumption, E, is calculated as

$$E = \sum_{i=1}^{n} P_l T_l^i + P_t T_t^i + P_r T_r^i + P_p T_p^i + P_s T_s^i \qquad (17)$$

Where n is the number of deployed nodes. The average value of E is taken after 30 simulation runs.

Average Duty Cycle. The duty cycle is the ratio of a node's active time and the summation of active and sleeping time.

End-to-End Latency. End-to-End latency of a packet is measured as the time difference between the packet generation time and the time when it is received by the sink. Delays experienced by distinct data packets are averaged over the total number of distinct packets received by the sink.

Delivery Ratio. It is the ratio of the total number of packets received by the sink to the total number of packets generated by the source nodes.

Average Goodput. We measure the average goodput of a source node as the average number of successful packet receptions per second by the sink from each source node.

Buffer Drop Rate. It is the ratio of dropped packets due to buffer overflow to the number of transmitted packets, expressed as a percentage.

Collision Drop Rate. It is the ratio of the dropped packets due to collision to the number of transmitted packets, expressed as a percentage.

Table 2 Parameters and their values used in the simulation

Parameter	Value	Parameter	Value	Parameter	Value
Data rate	250 kbps	Beacon length	96–113 bits	N_r^{max}	3 packets
SIFS	192μs	MAC header length	14bytes	λ	9 packets
Slot time	320μs	Payload length	32bytes	ε	0.1
Tx range	30 m	BO_b	8	CRC	5
Carrier sensing range	67 m	BO_d	32	μ	4
CCA check delay	128μs	Queue size	40packets	m	3
PHY header	6 bytes	Retry limit	5	γ	1

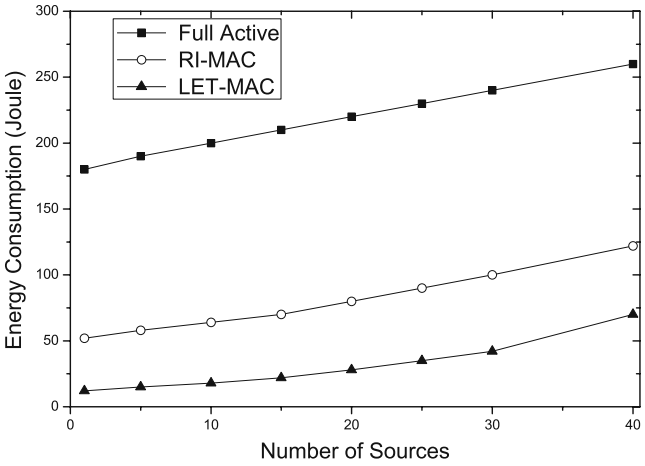


Fig. 4 Average energy consumption for different numbers of sources

5.3 Simulation results

This section presents the results obtained through evaluating LET-MAC with others using the metrics stated above. The results are shown in two different traffic models.

5.3.1 Correlated event traffic model

The correlated event traffic model grabs the effect of spatially correlated events in a sensor network. The remote sensing applications generate a high volume of traffic when an event occurs, and the number of sources along with the traffic generation rate signifies the traffic load situation. To reflect this scenario, this model picks a random location for the generation of an event. The event radius is varied from 6 m to 36 m and, thereby, got the number of sources[2] from 1 to 40. Moreover, we assume that an event occurred after the 10 seconds from the beginning of the simulation, and this crisis situation lasts for 20 seconds. During the crisis state, the data generation interval for the sources is set as 0.05 seconds, whereas, in the dormant state, it is set as 10 seconds (assuming the maximum data generation interval). In this study, we vary the traffic load by varying the number of sources.

Figure 4 shows the total energy consumption among the sensor nodes as the traffic load increases. Typically, full-active CSMA/CA shows the highest energy consumption, as no duty cycle mechanism is employed, and most of the energy is consumed for idle listening. LET-MAC and RI-MAC both exhibit significantly lower energy consumption, compared to full-active CSMA/CA due to the duty cycling mechanism; however, LET-MAC conserves energy far better than RI-MAC, because of its load-aware duty cycle maintenance using bi-directional rate control. Hence, during the dormant period, nodes in LET-MAC stay in the sleeping state much longer than RI-MAC. Moreover, during high traffic load, although the wake-up interval of LET-MAC decreases with the increase of the packet sending rate, unlike RI-MAC, the receivers still perform a sleep operation after receiving N_r^{max} packets (a receiver in RI-MAC continues receiving packets as long as it gets packets from its upstream senders), and senders are also tuned to the receiver's wake-up time, thus avoiding idle listening, which, still exists for the senders of RI-MAC. For the same reasons, as shown in Fig. 5, LET-MAC shows lower duty cycle than RI-MAC as the traffic load increases. In contrast, due to a lack of any sleep mechanism, full-active CSMA/CA has a 100% duty cycle irrespective of the traffic load. Figure 6 illustrates the energy consumption of the sensor nodes at different periods of the simulation. The result is shown for 40 source nodes. During 10 to 30 seconds of the simulation time, while the event occurs, a dramatic increase in energy consumption is observed for all the protocols. However, during the dormant state, while data generation interval is very low (10 seconds), LET-MAC shows significantly lower energy consumption than the other protocols. At this period, RI-MAC also outperforms the full-active CSMA/CA due to the duty cycle maintenance.

We evaluate the average end-to-end latency per packet during crisis state with increasing traffic load

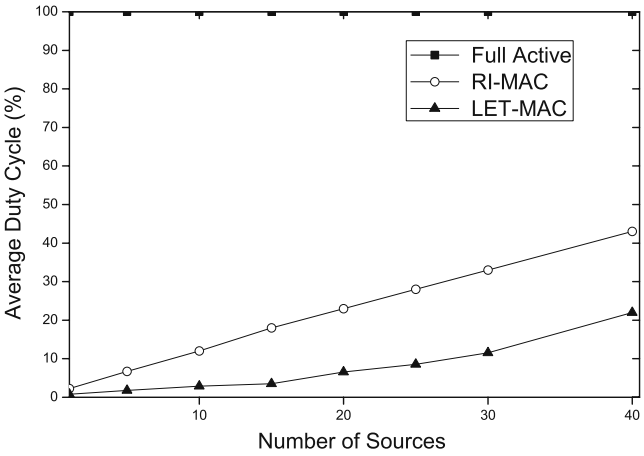


Fig. 5 Average duty cycle for different numbers of sources

[2]No of sources= $\rho\pi R_s^2$ where, $\rho = \frac{100}{100\times100}$, $\pi = 3.14$, $R_s =$ *sensing range*

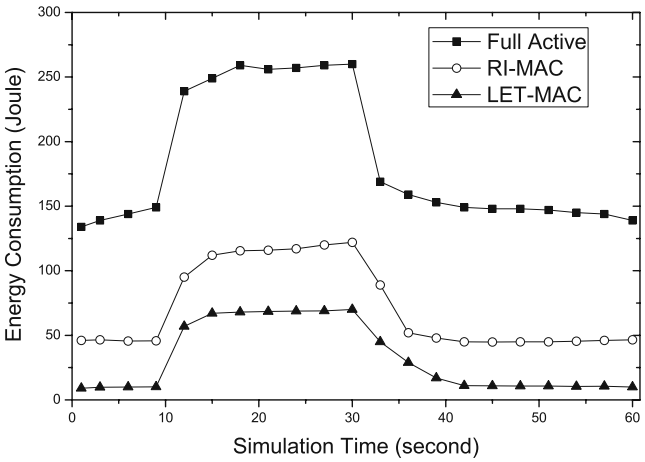


Fig. 6 Average energy consumption versus simulation time

as the number of sources increase, shown in Fig. 7. Full-active CSMA/CA shows the best performance for latency since it is exempted from any sleep delay or beacon waiting delay for data transmission. However, with the increase of the traffic load, the end-to-end latency increases due to the retransmission increases caused by collisions. A similar trend is observed for all the cases shown in this figure. LET-MAC delineates the closest performance with full-active CSMA/CA. The reason is that, although it incurs some sleep delay due to the maintenance of duty cycle, the beacon rate adjustment technique during high traffic load decreases the wake-up interval considerably, and the backward rate enforcement mechanisms reduce the retransmission of data packets caused by both high contention and congestion. RI-MAC exhibits the worst delay performance

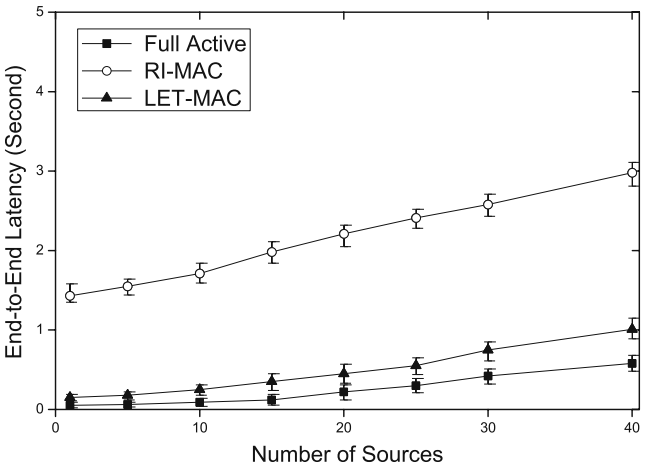


Fig. 7 Average end-to-end latency for different number of sources

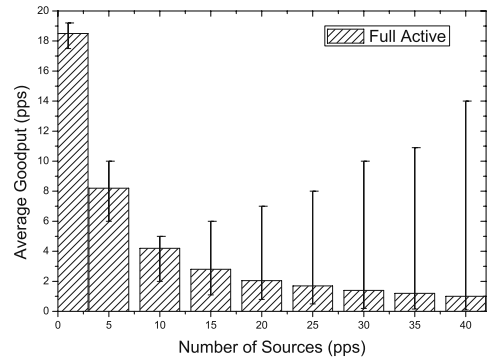
due to the absence of load aware duty cycle maintenance and suffers from huge collision and thereby retransmission during a crisis state. For example, in an extremely high traffic situation when the number of sources is 40 the end-to-end latency is improved by 66% for LET-MAC as compared to RI-MAC.

Figure 8 demonstrates the average goodput achieved per node with different numbers of sources. In this study, we implement a MAC-based distributed rate control mechanism (IFRC) with full-active CSMA/CA to know the behavior using a rate control mechanism. The error bars indicate the maximum variation in node goodput. As the Fig. 8 shows, the average goodput per node decreases as the number of sources increases for all of the protocols since the traffic load exceeds the network capacity. Although full-active CSMA/CA achieves higher goodput per node, the variation in node goodput is also higher which exhibits unfair behavior among the nodes (Fig. 8a). With the use of IFRC, full-active CSMA/CA shows fairness, although the average goodput decreases slightly because of aggressive congestion avoidance by IFRC (Fig. 8b). In contrast, for the lack of rate control along with non-adaptive duty cycle maintenance RI-MAC shows the worst performance both in achievable goodput and fairness (Fig. 8c). Due to the bi-directional rate control mechanisms LET-MAC performs closer to full-active CSMA/CA in achieving higher goodput with the additional benefit of fairness among the source nodes (Fig. 8d).

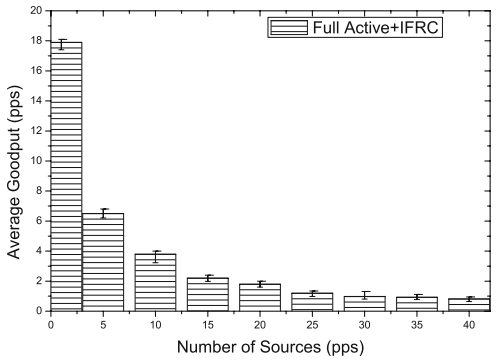
Considering the delivery ratio, both LET-MAC and full-active CSMA/CA with IFRC achieves an almost 100% delivery ratio due to the rate control techniques as shown in Fig. 9. However, the absence of rate control mechanism prevents a higher delivery ratio for both full-active CSMA/CA without rate control and RI-MAC. Comparatively, RI-MAC shows the worst performance due to a huge number of collision and congestion loss for maintaining non-adaptive duty cycle in a crisis state.

5.3.2 Periodic traffic load model

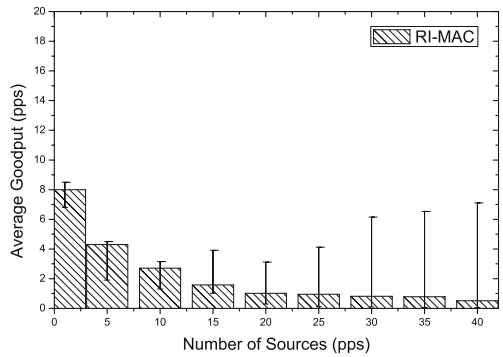
The periodic traffic load model represents the periodic application (i.e. monitoring application), in which sensors generate data at a fixed interval. In this model, each sensor sources traffic at a particular offered load, as well as forward other nodes traffic through multi-hop manner. Here, we randomize the initial data generation of the sensors to avoid the synchronized periodic reports of the sensors. In this study, the traffic load is varied by changing the data generation interval of the nodes.



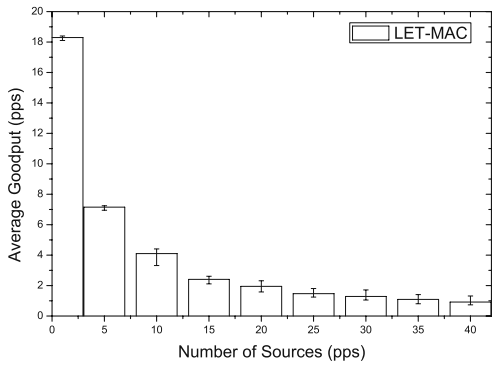
(a) Average Goodput for full-active CSMA/CA



(b) Average Goodput for full-active CSMA/CA with IFRC



(c) Average Goodput for RI-MAC



(d) Average Goodput for LET-MAC

Fig. 8 Average goodput achieved varying number of sources

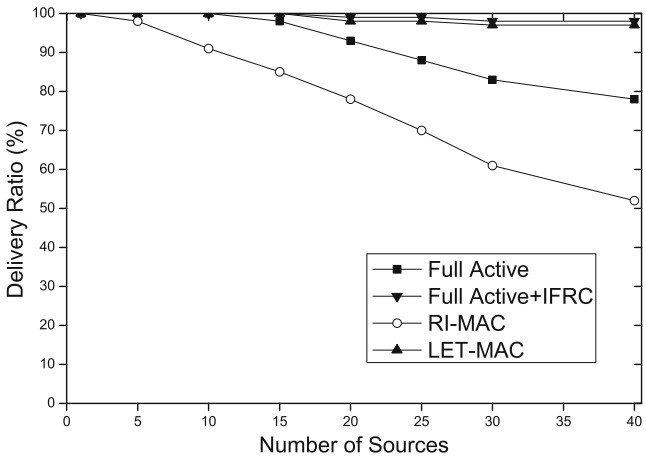


Fig. 9 Delivery ratio varying number of sources

The energy usage varying data generation interval is illustrated in Fig. 10. As the figure shows, for a high rate periodic application, the energy depletion is significantly higher than the applications which generate packets at moderate intervals for all of the protocols. During the high traffic rate, the energy consumption for transmission and reception is dominating rather than idle listening. Hence, after a certain data generation interval, the energy consumption falls for all of the protocols at which idle listening dominates the energy consumption. Through handling idle listening, the asynchronous duty cycle protocols, RI-MAC and LET-MAC show considerable energy conservation compared to full-active CSMA/CA. Comparatively, LET-MAC shows better performance in both situations, especially with low traffic scenario, since it

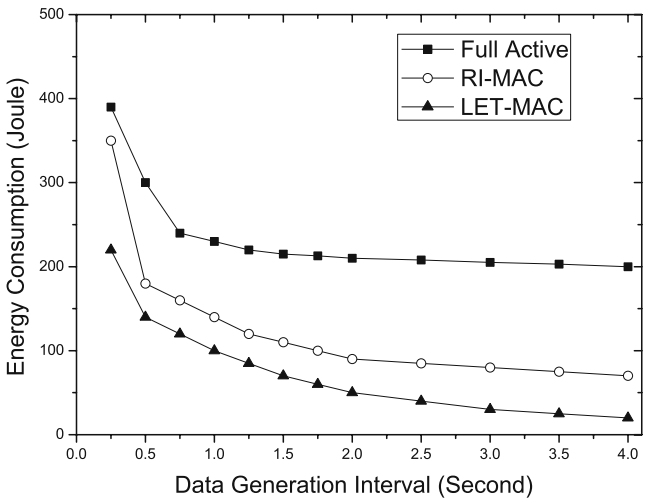


Fig. 10 Energy consumption versus data generation interval

reduces the idle listening and unnecessary wake-up significantly with the increase of the data generation interval along with avoidance of energy wastage due to packet drops through a bi-directional rate control mechanism.

As far as the average duty cycle is concerned, LET-MAC shows considerable improvement in maintaining very low duty cycle as the wake-up interval increases with the reduction of data generation rate, as illustrated in Fig. 11. On the other hand, after a certain interval, the duty cycle for RI-MAC shows steady behavior since, although sender's active time is reduced at low traffic load, receivers maintain their duty cycle in a fixed manner. As usual, full-active CSMA/CA possesses 100% duty cycle irrespective of the traffic load.

Figure 12 delineates the effect of the data generation interval on average end-to-end latency per packet. While high offered load is given to the source nodes, LET-MAC performs well with full-active CSMA/CA having lower latency. Due to the huge collision and retransmission occurred, as well as the long sleep interval during that period, RI-MAC exhibits higher latency per packet. However, as the data generation interval increases, the end-to-end latency of LET-MAC also increases due to the long sleep delay. This reflects the design objective of LET-MAC, in particular, achieving lower latency during the high traffic load offered, which shows similar performance to full-active CSMA/CA, while for lower traffic load gaining high energy efficiency. Hence, concerning latency, LET-MAC is suitable for the periodic applications that generate data at a high rate, which is quite common in practice for WSNs. (i.e. Structural monitoring, Habitat Monitoring).

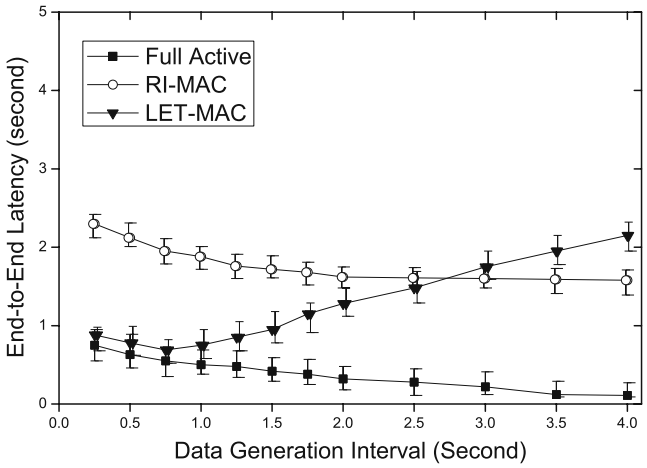


Fig. 12 Average end-to-end latency versus data generation interval

The impact of offered traffic load over delivery ratio has been illustrated in Fig. 13. This study also includes IFRC with full-active CSMA/CA. During high traffic rate, both RI-MAC and full-active CSMA/CA achieve a quite lower delivery ratio (below 50%) because of the absence of any rate control algorithm causing high contention and congestion loss. However, a sharp increase in delivery ratio is observed as the traffic rate decreases for both of the protocols. On the other hand, LET-MAC and full-active CSMA/CA with IFRC achieves a higher delivery ratio (near 100 %) at almost all of the traffic load offered to the sources, except in a very extreme condition while it is reduced to 90% because of the collision. This finding signifies that, for high rate periodic applications, LET-MAC surpasses both full-active CSMA/CA (when no rate control mechanism is

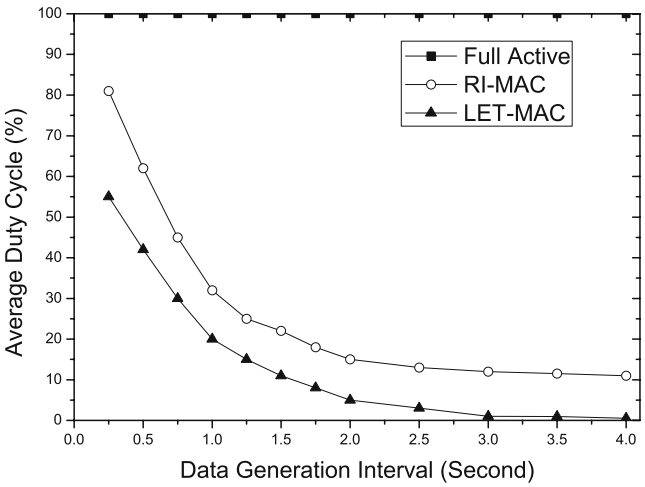


Fig. 11 Average duty cycle versus data generation interval

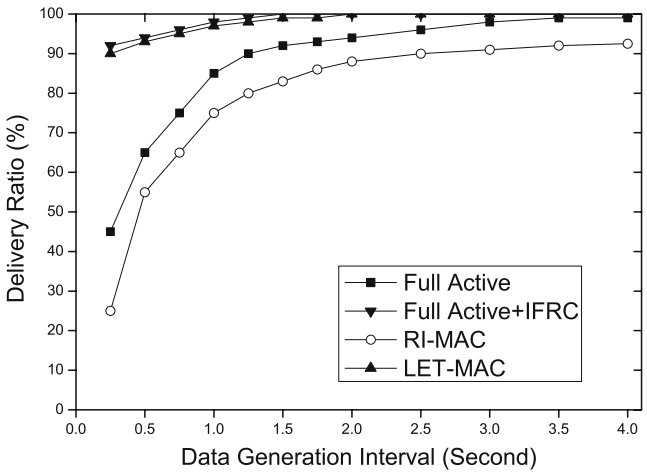


Fig. 13 Delivery ratio versus data generation interval

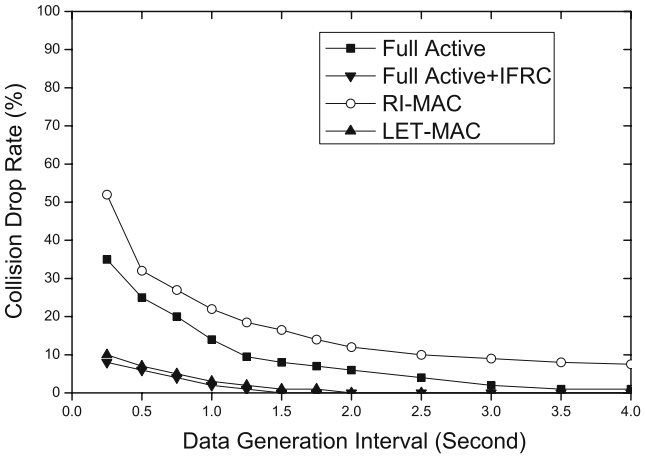


Fig. 14 Collision drop rate versus source data rate

used), as well as asynchronous MAC protocols like RI-MAC, in terms of achieving higher delivery ratio.

The sources of packet drops at different data generation interval is portrayed in Figs. 14 and 15, respectively, while the former shows the collision drop rate and the latter exhibits the packet drop rate due to buffer overflow. As the figures show, more of the drops are caused due to the number of collisions than buffer overflow as the source data rate increases. With the presence of contention and congestion handling mechanisms, LET-MAC excels for full-active CSMA/CA and RI-MAC by far in terms of collision drop rate, as shown in Fig. 14. However, with the use of IFRC, full-active CSMA/CA also shows lower collision drop rate. For the congestion avoidance technique, no buffer drops occurred for either LET-MAC or full-active CSMA/CA with IFRC, as shown in Fig. 15. However, congestion

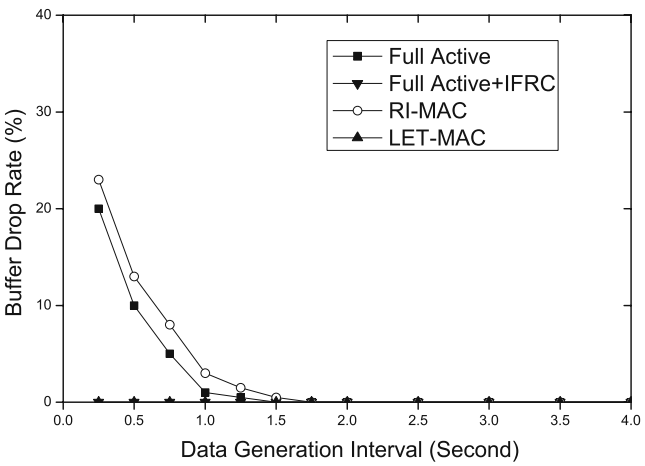
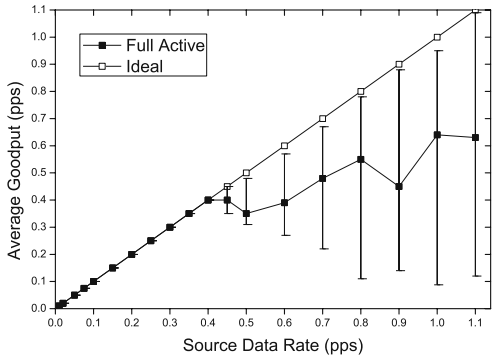
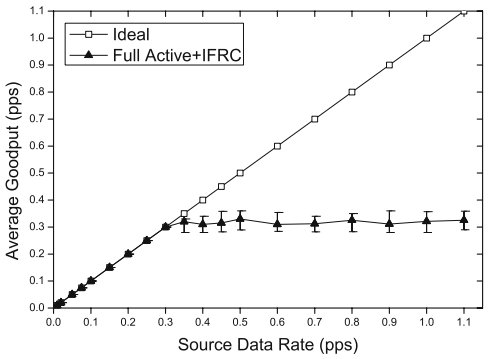


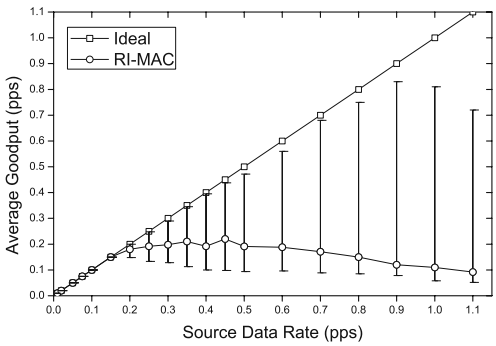
Fig. 15 Buffer drop rate versus source data rate



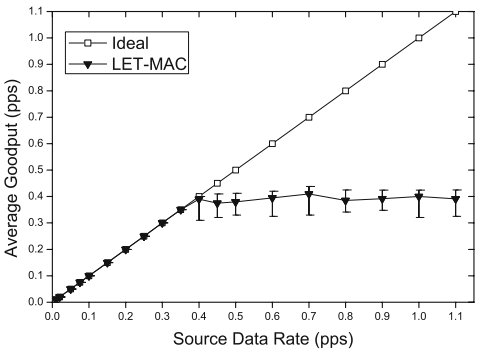
(a) Average Goodput for full-active CSMA/CA



(b) Average Goodput for full-active CSMA/CA with IFRC



(c) Average Goodput for RI-MAC



(d) Average Goodput for LET-MAC

Fig. 16 Average goodput achieved varying the Source data rate

drops are noticed in the case of RI-MAC and in that of full-active CSMA/CA without rate control.

Figure 16 plots the average goodput achieved over all nodes at different offered loads for full-active CSMA/CA Fig. 16a, full-active CSMA/CA with IFRC Fig. 16b, RI-MAC Fig. 16c, and LET-MAC Fig. 16d. For each case, the diagonal line represents the achievable rate with infinite capacity. The error bars parallel to the y-axis indicates the maximum variation in node goodput at each offered load. As Fig. 16a shows, the maximum achievable fair rate for full-active CSMA/CA with no rate control mechanism is 0.42 pps, after which, although the average goodput increases, the variability in goodput also increases. The high variation in error bars indicates the low fairness achieved among the nodes. In particular, nodes closer to the sink achieve higher goodput than the nodes which are far the away. While IFRC is used, full-active CSMA/CA shows fair per node goodput, although the maximum achievable goodput decreases (0.3 pps). LET-MAC also demonstrates fair goodput per node. However the maximum achievable goodput for LET-MAC (0.38 pps) is better than full-active CSMA/CA with IFRC because of bi-directional rate control mechanisms, which controls the rate less aggressively than IFRC does, considering both contention and congestion. In contrast, the maximum sustainable fair rate for RI-MAC is only 0.15 pps, as it lacks non-adaptive duty cycle maintenance with no rate control functionalities.

6 Concluding remarks

This paper proposes LET-MAC, a novel rate control integrated asynchronous duty cycle MAC protocol for WSNs. With the proposed bi-directional rate control mechanisms exploiting the receiver initiated transmission, LET-MAC achieves optimal performance in different traffic load. In particular, during low traffic situation, it achieves higher energy efficiency, while in the crisis state, it obtains maximum sustainable throughput.

We measured the performance of LET-MAC using ns-2 which revealed that, compared to RI-MAC, LET-MAC improves about 71.5% in energy conservation, while the traffic rate is low (0.25 pps), considering a 100 node network, with all acting as sources. Moreover, while fair achievable goodput and end-to-end latency is concerned in the crisis state, LET-MAC achieves fair goodput and lower end-to-end latency near the full-active CSMA/CA protocol and far outperforms RI-MAC in fairness, achievable goodput, and latency.

Acknowledgement This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by MEST (No. 2009-0083838).

References

1. Warneke B, Pister K (2002) Mems for distributed wireless sensor networks. In: 9th international conference on electronics, circuits and systems, 2002, vol 1, pp 291–294
2. Kang J, Zhang Y, Nath B (2007) Tara: topology-aware resource adaptation to alleviate congestion in sensor networks. IEEE Trans Parallel Distrib Syst 18(7):919–931
3. Anastasi G, Conti M, Di Francesco M, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Ad Hoc Netw 7(3):537–568
4. Halkes GP,van Dam T, Langendoen KG (2005) Comparing energy-saving mac protocols for wireless sensor networks. Mob Netw Appl 10(5):783–791
5. Rangwala S, Gummadi R, Govindan R, Psounis K (2006) Interference-aware fair rate control in wireless sensor networks. SIGCOMM Comput Commun Rev 36(4):63–74
6. Wan C-Y, Eisenman SB, Campbell AT (2003) Coda: congestion detection and avoidance in sensor networks. In SenSys '03: proceedings of the 1st international conference on embedded networked sensor systems. ACM, New York, pp 266–279
7. Wang C, Li B, Sohraby K, Daneshmand M, Hu Y (2007) Upstream congestion control in wireless sensor networks through cross-layer optimization. IEEE J Sel Areas Commun 25(4):786–795
8. Ee CT, Bajcsy R (2004) Congestion control and fairness for many-to-one routing in sensor networks. In SenSys '04: proceedings of the 2nd international conference on Embedded networked sensor systems. ACM, New York, pp 148–161
9. Ye W, Heidemann J, Estrin D (2004) Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Trans Netw 12(3):493–506
10. Du S, Saha A, Johnson D (2007) Rmac: a routing-enhanced duty-cycle mac protocol for wireless sensor networks. In: INFOCOM 2007. 26th IEEE international conference on computer communications. IEEE, Piscataway, pp 1478–1486
11. Lu G, Krishnamachari B, Raghavendra CS (2007) An adaptive energy-efficient and low-latency mac for tree-based data gathering in sensor networks: research articles. Wirel Commun Mob Comput 7(7):863–875
12. Sun Y, Du S, Gurewitz O, Johnson DB (2008) Dw-mac: a low latency, energy efficient demand-wakeup mac protocol for wireless sensor networks. In MobiHoc '08: proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing. ACM, New York, pp 53–62
13. Polastre J, Hill J, Culler D (2004) Versatile low power media access for wireless sensor networks. In SenSys '04: proceedings of the 2nd international conference on Embedded networked sensor systems. ACM, New York, , pp 95–107
14. Buettner M, Yee GV, Anderson E, Han R (2006) X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In SenSys '06: Proceedings of the 4th international conference on embedded networked sensor systems. ACM, New York, pp 307–320
15. Stone K, Colagrosso M (2007) Efficient duty cycling through prediction and sampling in wireless sensor networks. Wirel Commun Mob Comput 7(9):1087–1102

16. Sun Y, Gurewitz O, Johnson DB (2008) Ri-mac: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks. In SenSys '08: proceedings of the 6th ACM conference on Embedded network sensor systems. ACM, New York, pp 1–14
17. El-Hoiydi A, Decotignie J-D (2005) Low power downlink mac protocols for infrastructure wireless sensor networks. Mob Netw Appl 10(5):675–690
18. Musaloiu-E R, Liang C-J, Terzis A (2008) Koala: ultra-low power data retrieval in wireless sensor networks. In: IPSN '08. International conference on information processing in sensor networks, 2008, pp 421–432
19. Garcia-Luna-Aceves JJ, Tzamaloukas A (1999) Reversing the collision-avoidance handshake in wireless networks. In MobiCom '99: proceedings of the 5th annual ACM/IEEE international conference on mobile computing and networking. ACM, New York, pp 120–131
20. van Dam T, Langendoen K (2003) An adaptive energy-efficient mac protocol for wireless sensor networks. In SenSys '03: proceedings of the 1st international conference on embedded networked sensor systems. ACM, New York, pp 171–180
21. Ye W, Silva F, Heidemann J (2006) Ultra-low duty cycle mac with scheduled channel polling. In SenSys '06: proceedings of the 4th international conference on embedded networked sensor systems. ACM, New York, pp 321–334
22. Hull B, Jamieson K, Balakrishnan H (2004) Mitigating congestion in wireless sensor networks. In SenSys '04: proceedings of the 2nd international conference on Embedded networked sensor systems. ACM, New York, pp 134–147
23. Jian Y, Chen S (2008) Can csma/ca networks be made fair? In MobiCom '08: proceedings of the 14th ACM international conference on Mobile computing and networking. ACM, New York, pp 235–246
24. Chipcon inc. CC2420 datasheet. http://www.chipcon.com
25. Kim TO, Park JS, Chong HJ, Kim KJ, Choi BD (2008) Performance analysis of ieee 802.15.4 non-beacon mode with the unslotted csma/ca. IEEE Commun Lett 12(4):238–240
26. Na J, Lim S, Kim C-K (2008) Dual wake-up low power listening for duty cycled wireless sensor networks. EURASIP Journal on Wireless Communications and Networking 2008(11)
27. Floyd S, Jacobson V (1993) Random early detection gateways for congestion avoidance. IEEE/ACM Trans Netw 1(4): 397–413