Computer Networks 105 (2016) 33–46

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

L-MAC: A wake-up time self-learning MAC protocol for wireless sensor networks

Thanh Dinh[a], Younghan Kim[a], Tao Gu[b,*], Athanasios V. Vasilakos[c]

[a] School of Electronic Engineering, Soongsil University, South Korea
[b] School of Computer Science and Information Technology, RMIT University, Australia
[c] Lulea University of Technology, Sweden

A R T I C L E I N F O

Article history:
Received 1 June 2015
Revised 30 March 2016
Accepted 17 May 2016
Available online 24 May 2016

Keywords:
Wireless sensor networks
Duty cycle
MAC protocols

A B S T R A C T

This paper analyzes the trade-off issue between energy efficiency and packet delivery latency among existing duty-cycling MAC protocols in wireless sensor networks for low data-rate periodic-reporting applications. We then propose a novel and practical wake-up time self-learning MAC (L-MAC) protocol in which the key idea is to reuse beacon messages of receiver-initiated MAC protocols to enable nodes to coordinate their wakeup time with their parent nodes without incurring extra communication overhead. Based on the self-learning mechanism we propose, L-MAC builds an on-demand staggered scheduler to allow any node to forward packets continuously to the sink node. We present an analytical model, and conduct extensive simulations and experiments on Telosb sensors to show that L-MAC achieves significant higher energy efficiency compared to state-of-the-art asynchronous MAC protocols and a similar result of latency compared to synchronous MAC protocols. In particular, under QoS requirements with an upper bound value for one-hop packet delivery latency within 1 s and a lower bound value for packet delivery ratio within 95%, results show that the duty cycle of L-MAC is improved by more than 3.8 times and the end-to-end packet delivery latency of L-MAC is reduced by more than 7 times compared to those of AS-MAC and other state-of-the-art MAC protocols, respectively, in case of the packet generation interval of 1 min. L-MAC hence achieves high performance in both energy efficiency and packet delivery latency.

1. Introduction

To support data transmission in Wireless Sensor Networks (WSNs), a Medium Access Control (MAC) protocol [1] which controls radio communication for each sensor node is carefully designed to achieve high energy efficiency and low packet delivery latency. Over the past few years, duty cycling has been greatly explored in designing energy-efficient MAC protocols. In duty cycling approaches, nodes wake up periodically to sense the communication channel for incoming data. If there are no packets received or to send, a node will go to sleep to save energy. However, despite much work having been done in the literature [1–10], there is still a lack of a practical solution to resolve the trade-off between energy efficiency and packet delivery latency in duty cycling MAC protocols, especially when they are applied to low data rate applications.

Duty cycled MAC protocols for sensor networks can generally be divided into two categories: synchronous [2,3,11,12] and asynchronous [4–8,13–15].

In the synchronous approach, the active periods of nodes are synchronized to overlap with that of their neighbor nodes [2,3] so that a communication link between a sender and its one-hop receiver can be established immediately during their wakeup period. As a result, synchronous protocols normally achieve low packet delivery latency. For example, D-MAC [3] achieves very low packet delivery latency by adopting a staggered wakeup pattern. However, synchronous MAC protocols require nodes to be fully synchronized, which is very expensive and even difficult to achieve in certain circumstances due to its complexity [1]. Efficient synchronization is still a challenging topic in duty-cycled WSNs because duty-cycled WSNs are normally partitioned, limited power, constrained computational capacity, and long delay (i.e., nodes may sleep most of the time). Especially, in low data rate applications, when the number of data packets is relatively small, the synchronization overhead O_{sync} can be dominant compared to that of data communication O_{data}. While a node may send/receive only one data packet to its parent in a cycle, it may need to receive/send multiple

* Corresponding author.
E-mail addresses: thanhdcn@dcn.ssu.ac.kr (T. Dinh), younghak@dcn.ssu.ac.kr (Y. Kim), tao.gu@rmit.edu.au (T. Gu), vasilako@ath.forthnet.gr (A.V. Vasilakos).

http://dx.doi.org/10.1016/j.comnet.2016.05.015
1389-1286/

timing packets from neighbor nodes for synchronization purpose. This results in a high ratio (i.e., $\partial = O_{sync}/O_{data}$), which is obviously not efficient.

Asynchronous protocols [4,16,17] have been proposed to address the above limitation, and they decouple the duty cycle schedules of different nodes and thus eliminate the overhead for synchronization to achieve higher energy efficiency compared to synchronous protocols. However, the major limitation of these protocols is that they typically have high end-to-end delay for packet delivery due to the sleep latency problem [1]. We conducted analysis over existing asynchronous MAC protocols, and discovered that their performance (i.e., delivery latency and energy efficiency) is negatively impacted when the length of wakeup interval is increased. Therefore, a node has to wake up frequently, even in the case of low data rate applications, to achieve a reasonable trade-off between the energy efficiency and the packet delivery latency. For example, in B-MAC [4], the optimal wakeup interval for both applications with a periodic reporting interval of 5 min and applications with a lower reporting interval of 20 min is lower than 500 ms; a larger interval leads to higher energy consumption and higher packet delivery latency. In both of the aforementioned applications, a node has to wake up frequently for listening (i.e., every 500 ms) even when only one packet is generated in either case (i.e., every 5 min or 20 min). It is obvious that most of these wakeups are unnecessary because no data packet is transmitted.

This paper focuses on the trade-off problem between energy efficiency and packet delivery latency of MAC protocols in low data rate and periodic reporting applications which are popular used in industrial automation. To better trade off packet delivery latency and energy efficiency in low data rate WSNs, we propose a novel and practical wakeup time self-learning receiver-initiated MAC protocol (L-MAC). L-MAC is designed for low data rate periodic reporting applications where a data collection tree is normally used to gather data from sensors. A child node in L-MAC learns to adapt its sleep period based on the relative wakeup time with its parent in a way so that it can maintain its wakeup time to be closely earlier than that of its parent. Importantly, the node measures the relative wakeup time by itself through reusing beacon messages which are typically used for probing purposes in the receiver-initiated MAC approach. L-MAC does not require synchronization or schedule information exchanging, and incurs no extra transmission overhead compared to other receiver-initiated MAC protocols. As a result of the wakeup time self-learning, not only the idle listening of the sender but also that of the receiver (i.e., parent node) are reduced significantly as their wakeup time are designed to be close to each other. Whenever a child node wakes up and has packets to send, it can send the packets quickly, thus improving the energy efficiency and one-hop packet delivery latency.

We also design L-MAC's staggered wakeup scheduler based on the above self-learning mechanism, allowing a node to forward packets continuously to the sink without strictly depending on length of the wakeup interval, thus further shortening end-to-end delivery latency and transmission overhead. Moreover, built on the staggered scheduler, L-MAC seeks to expand wakeup interval in proportion to data rate. The purpose is to allow nodes in low data rate applications sleep longer compared to those in higher data rate applications, without a significant negative effect on network performance such as latency and delivery ratio. With the same amount of active time in a wakeup interval, the larger the interval length is, the lower the duty cycle of a node achieves, hence further improving energy efficiency. As a result, L-MAC is able to achieve both low latency and high energy efficiency at the same time. Through our comprehensive analysis, extensive simulations and experiments on Telosb sensors using TinyOS, we show that L-MAC outperforms state-of-the-art protocols. In particular, under a context of setting an upper bound value for one-hop packet delivery latency within 1 s and a lower bound value for packet delivery ratio within 95%, results show that the duty cycle of L-MAC is improved by more than 3.8 times and the end-to-end packet delivery latency of L-MAC is reduced by more than 7 times compared to those of AS-MAC and other MAC protocols, respectively, in case of the packet generation interval of 1 minute. L-MAC hence achieves high performance in both energy efficiency and packet delivery latency.

In summary, this paper makes the following contributions.

- We analyze the trade-off issue between energy efficiency and latency in existing MAC protocols, and discover their limitations when applied to low data rate applications.
- We propose a novel self-learning mechanism which enables a node to coordinate its wakeup with its parent without requiring synchronization or exchanging schedule information. We also design a staggered wakeup scheduler to allow a node to forward packets continuously to the sink. The design of L-MAC is very simple and easy to implement in real scenarios.
- We provide a detailed theoretical framework to quickly analyze and evaluate performance of current MAC protocols. Through our comprehensive analysis, we show that L-MAC achieves significantly higher energy efficiency compared to other asynchronous MAC protocols and a similar result of latency compared to synchronous MAC protocols. We conduct extensive simulations and experiments with Telosb motes, and show that L-MAC allows nodes in low data rate applications to sleep longer compared to those in higher data rate applications by setting a larger wakeup interval, without a significant negative effect on latency. As a result, L-MAC achieves high performance in both energy efficiency and packet delivery latency.

The rest of this paper is organized as follows. Section 2 discusses related works. Section 3 gives the overview and the detailed design of L-MAC. The analytical model and analysis of the trade-off problem in existing MAC protocol as well as L-MAC are presented in Section 4. Section 5 describes our validation and evaluation. Finally, Section 6 concludes the paper.

2. Related works

In this section, we discuss the state-of-the-art in the literature, focusing on energy efficiency and packet delivery latency. Energy efficiency is a critical issue in both traditional WSNs [1,16] and energy harvesting WSNs [18–21], as well as in general machine-to-machine communication [22]. Therefore, duty cycling [16] has been greatly explored in designing energy-efficient MAC protocols.

Duty cycled MAC protocols for sensor networks can generally be categorized into synchronous and asynchronous schemes. In synchronous approach, MAC protocols are designed under an assumption of time synchronization among neighbor nodes. Sensor nodes are required to synchronize their active time together, as a natural solution to establish communication between two nodes. In this way, synchronous MAC protocols are normally designed to achieve a low packet delivery delay. D-MAC [3] is a notable synchronous protocol which achieves low packet delivery delay. Some MAC protocols use global synchronization [23,24], others exploit local synchronization [3,11,12,25,26]. In both approaches, a node is required to exchange timing information packets periodically with multiple neighbor nodes for synchronization purpose. Efficient and precise synchronization is a challenging topic in duty-cycled WSNs. The reason is that such a network is normally partitioned, long delay, limited power, and limited computational capacity, and nodes may sleep most of the time. Beside the cost of time synchronization, synchronous MAC protocols also require nodes exchanging their sleep/wakeup schedule, which adds up extra cost. As a result, synchronous MAC protocols are normally expensive in term of cost, especially when considering low data rate applications, and synchronization may even be difficult to achieve in certain circumstances due to its complexity [1].

In asynchronous approach, the communication among nodes is enabled by using sender-initiated low power listening [4,5,7] or receiver-initiated low power probing [8,9,27,28]. In sender-initiated MAC approach, the sender transmits preambles to explicitly notify other nodes that it has a pending need for communication. Meanwhile, other nodes, including the receiver, periodically wake up to listen for such a preamble transmission. The sender normally transmits preambles until the receiver wakes up and detects its transmission. After that, a communication link between the sender and the receiver is established. The limitation of the sender-initiated approach is that the preamble transmission occupies the channel in a long period of time and prevents neighboring nodes from transmission. In addition, cost for the long preamble transmission of senders is expensive. To address this limitation, in receiver-initiated MAC approach (e.g., RI-MAC [9] and A-MAC [28]), when a sender has packets to send, the sender wakes up and silently listens until it receives a beacon from its receiver. After that, the sender starts to transmit data packets. In receiver side, each receiver node periodically wakes up and transmits a short beacon message to notify other nodes that it is ready to receive packets.

Among the state-of-the-art asynchronous protocols, schedule learning is one of the most efficient techniques [16], which attempts to reduce energy consumption and/or delivery latency. AS-MAC [8] and PW-MAC [13], receiver-initiated MAC protocols, and WiseMAC [7], a sender-initiated MAC protocol, are notable examples [16]. In the schedule learning technique, nodes are guided in some way to learn or estimate the wakeup schedule of other nodes so that a sender node can adjust its sending time to reduce idle listening and preamble sending overhead. However, to enable the schedule learning, current protocols require nodes to exchange their schedule information which results in extra communication overhead and higher collision probability. In addition, even though these protocols achieve lower energy consumption compared to conventional asynchronous MAC protocols [5,6] as nodes may know other nodes' wakeup schedule, senders still suffer from a considerable sleep latency problem.

Despite much work having been done in the literature [8,9,16,22,27,29–34], there is still a lack of a practical solution to resolve the trade-off between energy efficiency and packet delivery latency in duty cycling MAC protocols, especially when they are applied to low data rate applications. This paper focuses on this trade-off problem for low data rate and periodic reporting applications which are popular used in industrial automation. In the next section, we describe our design of a novel MAC protocol, namely L-MAC, to resolve the problem, attempt to achieve both low packet delivery latency as synchronous approach and high energy efficiency as asynchronous approach. In the later part, we provide detailed analysis using a theoretical framework for the trade-off problem in existing MAC protocols and show how L-MAC can address the problem efficiently.

3. The design of L-MAC

This section presents the design of L-MAC. The Sections 3.1–3.5 describes components of our wakeup time self-learning algorithm . The subsection Section 3.6 describes how L-MAC build a staggered scheduler based on the wakeup time self-learning and how packets are transmitted in a staggered data collection tree. The purpose of building a staggered scheduler is to resolve the sleep latency problem by allowing a node to forward packets continuously to the sink. The staggered scheduler is created and maintained by the self-learning algorithm which incurs no extra communication overhead compared to conventional receiver-initiated MAC protocols. In this way, L-MAC achieves high energy efficiency and low delivery latency at the same time. In other words, L-MAC addresses the performance trade-off problem of duty-cycled MAC protocols.

3.1. Overview

L-MAC, a receiver-initiated MAC protocol, is designed to enable child nodes to coordinate their wakeup time with their parent node without requiring synchronization or exchanging schedule information, so that whenever a child node has data packets to send, it can send packets quickly to achieve both high energy efficiency and low delivery delay. This is done through a wakeup time self-learning algorithm in which a child node, instead of operating with a fixed wakeup interval, adapts its sleep period based on relative wakeup time with its parent, so that it can wake up closely earlier than its parent. Notably, the child node measures the relative wakeup time by itself through reusing beacon messages which are typically used for probing purposes in the receiver-initiated MAC approach [9]. Therefore, L-MAC incurs no extra transmission overhead compared to other receiver-initiated MAC protocols. As nodes in duty-cycled WSNs periodically wake up to listen for incoming packets, intuitively, if a child node wakes up before its parent node, and receives any tone (i.e., a beacon) when the parent wakes up, the node can measure the offset between its wakeup time and that of its parent. We re-use beacon messages used in receiver-initiated MAC protocols for that purpose. In each wakeup interval, a node measures and then compensates the offset by re-calculating its sleep period in a way so that its next wake up time is closely earlier than its parent node. As illustrated in Fig. 1, a child node N2 adjusts its sleep period (SP_{N2}^{i+1}) in the current wakeup interval T_{i+1} based on its previous sleep period SP_{N2}^{i}, the offset $O_{N2}^{N1}(i)$, the difference between its active period in the current interval $T_{N2}^{a}(i+1)$ and its active period in the previous interval $T_{N2}^{a}(i)$, and a guard time α. t_{N2}^{W} and t_{N2}^{S} denote wakeup time and sleep time of node N2, respectively. In this way, a child node N2 learns to maintain its wakeup time to be close to that of its parent N1.

3.2. The offset measuring mechanism

When a node i wakes up at time t_i^w, it firstly sends a short beacon to notify other nodes, following typical procedures of a receiver-initiated MAC protocol. The node then listens for incoming packets. Note that an intermediate node in a tree topology plays roles of both sender and receiver. Similarly, its parent also sends a beacon at the parent's wakeup time. When the node receives a beacon from its parent, it records receiving time t_p^{beacon} as the parent's wakeup time t_p^w. If a node p fails to send a beacon at the first time (i.e., channel busy or collision), it will attach its past active period T_p^{pap} ($T_p^{pap} = t^{current} - t_p^w$) into beacon messages in later retransmissions so that other nodes can estimate its wakeup time easily ($t_p^w = t_p^{beacon} - T_p^{pap}$). Node i then calculates the offset between its wakeup time and that of its parent using (1).

$$O_i^p = t_p^w - t_i^w \tag{1}$$

Note that how a child node can wake up before the parent node from the beginning is discussed in the setting-up mechanism subsection and a fault tolerance mechanism to cope with a case that a child node may wake up after its parent is also proposed.

3.3. The offset compensation mechanism

An actual interval of a node starts from the time it receives a beacon message from its parent. After a node i finishes its tasks

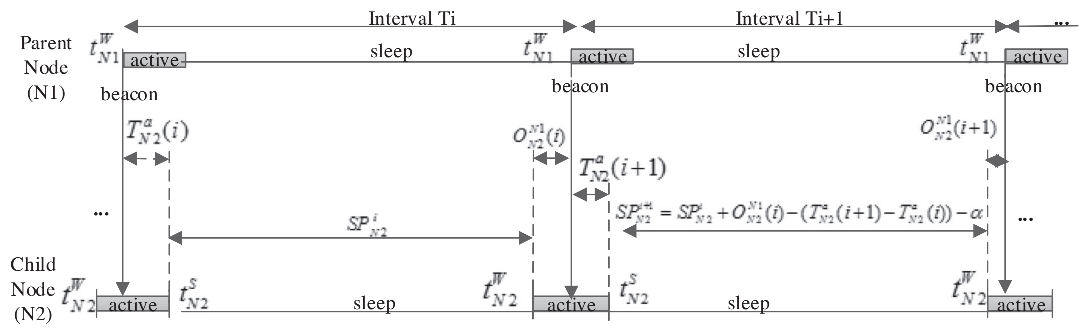


Fig. 1. Overview of wakeup time self-learning algorithm in L-MAC.

and prepares to go to sleep, it calculates its sleep period (SP_i) using (2). The purpose of calculating sleep period using (2) of node i is to compensate the offset and dynamically adapt its sleep period to adjust the time of its next wakeup.

$$SP_i = SP_i^{previous} + O_i^p - (T_i^a - T_i^{a-previous}) - \alpha \tag{2}$$

where $SP_i^{previous}$ is the sleep period of node i in the previous interval. T_i^a is the total active period of node i in the current interval ($T_i^a = t_{current} - t_p^w$). $T_i^{a-previous}$ is the active period of node i in previous interval. Due to clock drift, we add $\alpha = T_G = 2\rho T_W$ as a guard time to enable a node to wake up before its parent even when the maximum clock drift happens. T_W is the wakeup interval length. ρ is the maximum drift rate which is a constant given by the manufacturer a sensor device (i.e., 40 ppm for CC2420). Thus the maximum relative drift rate between a child node and its parent is 2ρ.

The node then sleeps for a period of SP_i which enables it to wake up closely earlier than its parent node next time. In this way, a node dynamically adapts its sleep period based on the offset and its task completion time (i.e., active period) to maintain the coordinated wakeup schedule. If a node completes its tasks earlier compared to the previous interval, it can sleep more with a longer calculated period and vice versa. It means that a node can adapt its sleep period based on its workload. This feature benefits specifically to intermediate nodes where incoming traffic may vary over time.

3.4. The fault tolerance mechanism

A node may not detect its parent's beacon as the node wakes up lately compared to its parent or there is errors in transmitting and receiving beacon messages (i.e., collision, interference). In this case, the node reduces its sleep period by doubling its guard time ($\alpha_{new} = 2\alpha_{previous}$) to quickly return its target state. The target state is that a node is always expected to wake up slightly before its parent. Note that after the fault is fixed, the normal guard (α) time is reset.

3.5. The setting-up mechanism

This subsection describes L-MAC's setting-up mechanism which is used for all nodes at the time of network deployment as well as for a node and its new parent node in case of dynamic networks. The mechanism is to enable a node to quickly achieve its target state.

We assume the data collection tree is available for the setting-up phase of L-MAC (i.e., predefined or constructed by a tree-based routing protocol). In this paper, we use CTP (Collection Tree Protocol) [35] for the network topology construction. The wakeup time coordination in the setting-up phase of L-MAC is executed after the setup phase of the upper layer protocol. Therefore, each node has already known its parent node. Because the sink node is always active, sink neighbor nodes are not required to coordinate their wakeup time with the sink node. Sink neighbor nodes can operate with their own wakeup interval T_W. As a result, the setting-up mechanism starts from sink neighbor nodes to leave nodes, following the top-down order which is as same as CTP's network topology setting-up phase. Because required information for nodes to calculate the offset value and to coordinate their wakeup time is not available at the time of network deployment, some extra information is attached to beacon messages to guide nodes to calculate their first sleep period so that the wakeup time coordination can be started. This is only required in the setup phase, thus it is trivial. We also highlight that only in the first interval, sink neighbor nodes sleep for a full wakeup interval T_W while they later will periodically wake up at the beginning of the interval.

First, each sink neighbor node sends a beacon containing its first sleep period ($= T_W$) to child nodes. After successfully sending the beacon, the sender will go to sleep for a period of T_W. To avoid collision and to distribute the sleep time of sink neighbor nodes differently, each node is required to perform a large contention window before broadcasting. It thus only one sink neighbor node within a transmission range can win the channel and send its beacon successfully at a point of time. A node with failed transmission is required to execute back-off and try again until it wins. As a result, sink neighbor nodes start sleeping at different point of time. As each branch of a data collection tree corresponds with a sink neighbor node, nodes in a branch coordinate their wakeup time with its corresponding sink neighbor node. As a result, each branch has different wakeup scheduling.

When a child node i receives a beacon from its parent (a sink neighbor node), it records the beacon receiving time as the sleep time of its parent t_p^s and the sleep period of the parent SP_p. The node also performs a contention window and broadcasts a beacon message to neighbor nodes attached with its sleep period SP_i which is calculated using (3). The sleep period allows the node to wakeup closely ealier than its parent next interval. If the transmission is successful, node i then goes to sleep. If not, node i tries to transmit the beacon again with an updated sleep period in real time until it succeeds.

$$SP_i = SP_p - (t_{current} - t_p^s) - \alpha \tag{3}$$

Similarly, when a child node of node i receives a beacon from its parent, it executes the same operations as discussed above and then sends out a beacon containing its calculated sleep period follows (3). After that, it goes to sleep. In the same way, downstream nodes calculate their sleep period and then sleep.

From next intervals, operations described in Sections 3.2 and 3.3 are used. The protocol only re-uses the typical beacon message of the receiver-initiated MAC [9] without requiring any extra information transmission. All information for calculating the sleep period of a node is measured locally by the node itself. In this way,

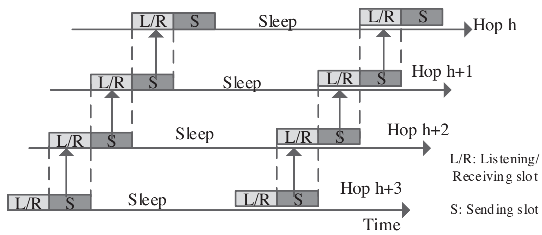


Fig. 2. L-MAC's staggered scheduler.

L-MAC enables a node to learn its wakeup time so that it wakes up closely earlier than its parent node.

3.6. Staggered collection tree

Based on our self-learning mechanism, this subsection describes how L-MAC extends the basic wakeup time coordination to build a staggered scheduler [3] for nodes on a route to the sink. The purpose is to enable a data packet can be delivered smoothly from any source to the sink node without suffering from a significant data forwarding interruption. In particular, the scheduler is designed in a way so that the sending period of a node overlaps its parent's listening/receiving period.

L-MAC establishes the staggered scheduler locally in a hop-by-hop fashion without requiring global information as in previous studies [16]. The staggered pattern can be easily achieved from the basic wakeup coordination above by adding a listening/receiving (L/R)slot and an on-demand sending (S) slot into the active period of a node. In previous studies [3], both L/R and S slots are assigned a fixed slot length of u which is enough to receive or transmit successfully one data packet (including size of a contention window). The L/R and S slots in L-MAC staggered scheduler also have the length of u. However, if each node has to listen for the whole L/R slot in every interval as in [3], it is inefficient in case of low data rate applications. The reason is that in these applications, the number of busy intervals (intervals with data packets to send/receive) is much smaller than the number of idle intervals. Therefore, instead of adding $(-u)$, we add only $(-u/2)$ into (2) and (3) to create a L-MAC staggered scheduler. In other words, a node is designed to wake up before its parent by $u/2$ to establish the staggered scheduler. Note that wakeup time learning operations remain the same as described in above sections, thus we do not repeat in this section which only focuses on description for operations to build a staggered scheduler and to transmit data packets in a staggered data collection tree.

As a node only needs to listen for a half of L/R slot if there is no incoming packet, the periodic listening overhead of a node in idle intervals can be reduced to almost a half compared to conventional designs [3]. The reason we use 'almost a half' is that as a child node is originally designed to wake up slightly earlier than its parent, the actual listening period of a node is typically longer than $u/2$. This also ensures a parent node can hear data packets from its child nodes. A full L/R slot of a node is only used on demand when the node has a packet to receive. If there is no incoming packet and a node has no packet to send, the node then goes to sleep after a timeout t_0. A drawback of this design is that a parent node may consume more energy in busy intervals if its child node starts sending at the end of the first half of its L/R slot. However, the benefit is greater than the drawback since the number of busy intervals is normally small in our target applications. A node uses an on-demand sending slot only when it has packets to send. The sending slot (if required) follows after the L/R slot as shown in Fig. 2. This enables a node to forward packets continuously to the sink node. In this way, L-MAC resolves the sleep latency problem efficiently to achieve low packet delivery latency.

Multi-packet sending mode: L-MAC also supports a multi-packet mode which is triggered when a node has more than one packet to send. For a source node with multi-packets to send, the node sends the first packet with a multi-packet flag attached to the packet header to request its parent and upper nodes for the multi-packet mode. Generally, L-MAC can support to transmit multiple packets continuously as a packet train by using m bits flag which can be used to specify the number of packets will be sent. In this way, a node can reduce the number of contention windows and ACK messages for multi-packet transmission. In addition, packets can be aggregated to reduce the number of transmissions. However, for a fair comparison, we do not assume any type of aggregation and use only one bit for the flag, which is obviously the worst-case scenario for energy efficiency of L-MAC. For a receiver, upon receiving a packet with a multi-packet flag, a receiver adds the flag into its ACK message as a response to the sender to accept the request and also to notify other nodes that the sender has reserved to transmit an additional packet. If there is any another child node having packets to send, it can take a short sleep before waking up again to send packets. After sending or relaying a packet with multi-packet mode, the sender or a forwarder pauses for a period of $3u$ before it starts for sending, receiving, or relaying additional packets. A period of $3u$ is to allow previous packets to be forwarded successfully out of the interference domain of the transmitter as the radio interference domain is normally twice the transmission range [10]. To support the case that there may be several child nodes having packets to send, L-MAC employs the data prediction scheme [3]. In particular, after a parent receives a normal data packet, it sets a schedule to sleep for a short period and then wake up again after a period of $3u$ to listen for possible incoming packets from other child nodes.

3.7. Collision avoidance

This section presents solutions to avoid collision effectively.

Inter-branch collision avoidance: As described in the setup phase, we use contention windows to distribute the sleep time of sink neighbor nodes. As a result, each sink neighbor and nodes in its branch wakes up at different point of time for communication compared to other branches, as nodes coordinate their wakeup time with their parent. Collision among nodes in different branches is thus avoided.

Intra-branch collision avoidance: In a branch, the wakeup and communication schedules of nodes on a route are staggered sequentially. Therefore, nodes at different hops are designed to transmit packets at different points of time. As a result, collision and overhearing can be avoided. In addition, we suppose that each node randomly picks up a time for its data sampling as L-MAC targets the periodic reporting application. In this way, nodes normally generate and transmit their data packets at different time to avoid collision.

Sibling collision avoidance and collision detection: As a node may have multiple child nodes, there is a chance that more than one child node sends data packets after receiving the parent's beacon. To avoid collision, before sending a data packet, senders execute a random backoff. If a sender detects another transmission during its backoff period, it cancels its transmission and overhears for an ACK. It decides to take a short sleep and retry later if a multi-packet mode was requested by another node, or it resumes with a new backoff in other cases. L-MAC employs a receiver-based collision detection technique. As a receiver listens to the channel for a period after sending a beacon, it measures the channel power level to detect bit patterns. If the measurement indicates an in-progress transmission, but a valid packet header is not detected,

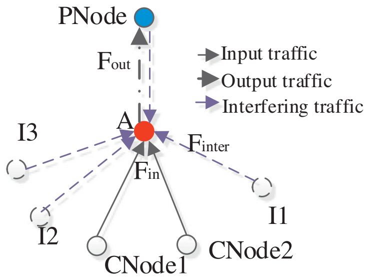


Fig. 3. A traffic model for MAC protocols.

the receiver then identifies a collision. In this case, the receiver performs a clear channel assessment (CCA) to detect if the channel is clear, then it rebroadcasts a beacon to notify senders about the collision and ask them to retransmit data packets.

Inter-flow collision avoidance: As described above, an inactive period ($3u$) is required to allow packets to be forwarded successfully to out of the interference domain of transmitters, thus avoid collision between flows.

4. Analytical model

In this section, we present a model to analyze the trade-off issue between energy efficiency and packet delivery latency of current MAC protocols and compare their performance with L-MAC. As the objective of this section is to focus on main ideas of each approach, we leave out many implementation details and simplify our model to allow for fast evaluation. Therefore, we make an analytical approach tractable in which latency and energy consumption are modeled as a function of key protocol parameters only, whereas comprehensive evaluations of protocols based on a full realistic model are given in the next part.

Application model: For analyzing the trade-off issue, we are interested in two performance metrics: energy efficiency and average latency. For simplicity, we do not model the queue overflow and collision, but we define constraints about the volume of network traffic. In addition, the impact of external interference is not considered, so we do not model the random packet loss and retransmission.

Traffic model: The traffic model, as shown in Fig. 3, is embedded in a tree topology where a node has a number of child nodes (CNode), a parent node (PNode), and C neighbor nodes (e.g., $I1$, $I2$). We assume the tree is constructed based on a minimum hop count scheme. In the model, a node (e.g., A) has input traffic F_{in}, output traffic F_{out}, and interfering traffic F_{inter} (traffic is sent by A's neighbor nodes, but not intended to A). For simplicity, we use a concentric circular ring (CCR) model with the sink node as the central point for the network deployment. Nodes communicate with each other based on a unit disk graph model. Nodes are uniformly deployed to achieve the same density with $D+1$ nodes per a unit disk (each node has D neighbors). Each CCR h consists of nodes with the same minimum hop count h to the sink. The number of nodes in the first ring simply equals to the number of neighbours of the sink. From that, we calculate the number of nodes Nh on the CCR h as Eq. (4).

$$N_h = \begin{cases} 1 & \text{if h = 0} \\ Dh^2 - D(h-1)^2 = D(2h-1) & \text{otherwise} \end{cases} \quad (4)$$

Nodes on CCR $h+1$ are children of nodes on CCR h. Because a node has only one parent node, we obtain the average number of child nodes of a node in level h by Eq. (5).

$$|C_h| = \begin{cases} 0 & \text{if h = } h_{max} \\ D & \text{if h = 0} \\ N_{h+1}/N_h = (2h+1)/(2h-1) & \text{otherwise} \end{cases} \quad (5)$$

where h_{max} is the maximum hop count to the sink (at leaves). We call F_{self} as a traffic rate generated by a node, and $F_{in}(h)$ as the average input traffic rate of a node in level h. The corresponding output traffic rate is the sum of F_{self} and $F_{in}(h)$.

$$F_{out}(h) = \begin{cases} F_{self} & \text{if h = } h_{max} \\ F_{in}(h) + F_{self} & \text{otherwise} \end{cases} \quad (6)$$

The input rate at a node in CCR h is the sum of output rate at its input links (from child nodes). We can then rewrite Eq. (6) as the cumulative self-generated traffic by nodes from leaves to level $h+1$ on its route and itself as follows.

$$F_{out}(h) = F_{self}(h_{max}^2 - h^2 + 2h - 1)/(2h-1) \quad (7)$$

We assume each interfering node have the same average traffic rate with the node, thus the average interfering traffic rate is:

$$F_{inter}(h) = (D - |C_h|)F_{out}(h) \quad (8)$$

We then define boundary conditions to safeguarding the contention-free operation of MAC protocols. We assume μ as the length of time required to receive or transmit one packet. *Boundary condition 1:* A node must not transmit more than one packet per μ_t at any time time t.

$$\mu_t F_{out}^t(h_i) \le 1 \quad \forall h, i \quad (9)$$

Boundary condition 2: For any disk unit $du(i)$, if node i transmits in period μ_t, other nodes must not transmit.

$$if\, \mu_t F_{out}^t(h_i) > 0 \to \mu_t F_{out}^t(h_j) = 0 \le 1 \quad \forall j, j \ne i \text{ and } j \in du(i) \quad (10)$$

Boundary condition 3: In one period μ_t, a node is unable to transmit and receive at the same time.

$$\mu_t F_{in}^t(h_i) > 0 \to \mu_t F_{out}^t(h_i) = 0\&\ vice - versa \quad (11)$$

Energy and latency models: We now model average end-to-end packet delivery latency and duty cycle of a node at hop h^{th} from the sink. Note that we use average radio duty cycle as an indicator for energy efficiency because most of energy in a sensor node is consumed by its radio module. Therefore, we only consider timing aspects for calculating the duty cycle (e.g., time for transmission). We skip the initialization phase to keep the model simple because its cost is negligible in long run. For D-MAC, WiseMAC, AS-MAC, we use F_S ($F_S = 1/T_{SI}$) to indicate the frequency of synchronization and schedule information exchange (T_{SI} is the corresponding interval). T_{pkt} denotes the time period to transmit a data packet ($T_{pkt} = (L_{hdr} + L_P + L_{ACK})t_B + sifs$). Similarly, T_S and T_{beacon} denote the time period to transmit a synchronization packet or a schedule information packet, and a beacon, respectively. T_G denotes the guard time which is specified by each protocol. Other parameters are described in Table 1. Note that guard time for clock drift, sleep latency, and preamble transmission are not required ($= 0$) for sink neighbor nodes ($h = 1$) because the sink is always active. The overall duty cycle (DC) of a node is calculated using (4) by simply adding duty cycles for each radio operation: listening (DC_{lx}), transmitting (DC_{tx}), receiving (DC_{rx}), overhearing (DC_{over}), and additional operations (DC_{add}) (if applicable).

$$DC = DC_{lx} + DC_{tx} + DC_{rx} + DC_{over} + DC_{add} \quad (12)$$

We conduct an extensive analysis for various MAC protocols, however only results of L-MAC are presented briefly due to page limit.

Table 1
Parameters.

Parameter	Meaning	Value
L_{hdr}	Packet header length	7 bytes
L_p	Payload length	32 bytes
L_{ACK}	ACK packet's length	10 bytes
$T_{simulation}$	Time period per a simulation	2 h
T_{rc}	Receive check period	2.5 ms
t_B	Time to TX/RX a byte	0.032 ms
CW	Contention window size	15
$sifs$	Short inter-frame space	192 µs
$T_{transition}$	Time to switch radio modes	167 µs
s θ	Maximum clock drift rate	40 ppm
Platform	Hardware, radio	Telosb, CC2420
D-MAC	Number of sleep slots	9–599
	Synchronization interval T_{SI}	30 s–300 s
	SYNC packet length L_S	20 bytes
	Slot length	10 ms
B-MAC	Wakeup interval	0.1 s–10 s
RI-MAC	Wakeup interval	0.1 s–10 s
A-MAC	Wakeup interval	0.1 s–10 s
WiseMAC	Wakeup interval	0.1 s–10 s
	Schedule information TX rate	$F_S = F_{in}$
AS-MAC	Wakeup interval	0.1 s–10 s
	Hello packet length L_S	18 bytes
	Hello interval T_{SI}	20 s–60 s
L-MAC	Wakeup interval	0.1 s–30 s
	Receiving/ sending slot	10 ms
	$(CW + T_{pkt} + T_{beacon})$	6 bytes
	Beacon length	

Results of other MAC protocols and detailed description can be found in [36].

L-MAC:

Latency: In each hop a message is delayed by a L/R slot with length μ on average and a possible offset with the maximum value equals to the guard time T_G. Note that T_G is not required at $h = 1$. Average delivery latency for a message generated by a node in hop hth is calculated as follows.

$$L_h = (h-1)(\mu + T_G) + \mu \tag{13}$$

Duty cycle:

Listening: In every interval, if there is no incoming and outgoing packets, a node wakes up to listen for a a half of L/R slot $\mu/2$ and T_G.

$$DC_{lx} = T_G/T_W + (\mu/2T_W - F_{in}\mu/2) \tag{14}$$

Transmitting: A node transmits a short beacon in its wakeup time and data packets when it has packets to send. Switching the radio to transmitting mode also consumes energy.

$$DC_{tx} = F_{out}(T_{pkt} + T_{transition}) + T_{beacon}/T_W \tag{15}$$

Receiving: A node receives data packets and beacons from its parent.

$$DC_{rx} = F_{in}T_{pkt} + T_{beacon}/T_W \tag{16}$$

L-MAC listens for $\mu/2$ for possible incoming packets once a node receives a data packet.

$$DC_{add} = F_{in}\mu/2 \tag{17}$$

RI-MAC [9]:

Latency: In each hop, a packet is delayed by a waiting period of $T_W/2$ on average, periods for beacon transmission, a contention window, and packet transmission.

$$L_h = (h-1)(T_W/2 + T_{beacon} + T_{CW} + T_{pkt}) + T_{CW} + T_{pkt} \tag{18}$$

Listening: In each interval, after waking up and sending out a probe message, a node stays awake to listen for a period of T_{lx}

$$T_{lx} = T_{pkt}^{max} + T_{transition} + T_{CW} \tag{19}$$

where T_{CW} is specified by a receiver. If a receiver does not indicate the value of T_{CW}, senders understand that back-off is not required. When a sender wants to send a packet packet to a receiver, it stays silently active to wait until receiving a beacon from the receiver. The average waiting period is about $T_W/2$. After that, it performs a contention window before it starts to transmit the data packet.

$$DC_{lx} = T_{lx}/T_W + F_{out}(T_W/2 + T_{transition}) + T_{CW} \tag{20}$$

Transmitting: A node transmits a short beacon when it wakes up, and data packets if the node has packets to send.

$$DC_{tx} = F_{out}(T_{pkt} + T_{transition}) + T_{beacon}/T_W \tag{21}$$

Receiving: A node receives data packets and beacon messages.

$$DC_{rx} = F_{in}T_{pkt} + F_{out}T_{beacon} \tag{22}$$

A-MAC [28]: A-MAC improves RI-MAC in term of quick decision to remain on or turn off the radio after sending a beacon. This is enabled by using auto-ACK packet in response to the receiver's beacon [A-MAC]. With this mechanism, a node can make a decision to go to sleep after a period of $T_{lx} = SIFS + T_{ACK}$ if it does not receive any ACK message. However, this comes at a cost for additional delay of the auto-ACK. Therefore, the latency and the duty cycle for A-MAC are computed as follows.

$$L_h = (h-1)(T_W/2 + T_{beacon} + T_{lx} + T_{CW} + T_{pkt}) + T_{CW} + T_{pkt} \tag{23}$$

$$DC_{lx} = T_{lx}/T_W + F_{out}(T_W/2 + +SIFS + T_{transition}) + T_{CW} \tag{24}$$

$$DC_{tx} = F_{out}(T_{autoACK} + T_{pkt} + 2T_{transition}) + T_{beacon}/T_W \tag{25}$$

$$DC_{rx} = F_{in}(T_{autoACK} + T_{pkt}) + F_{out}T_{beacon} \tag{26}$$

B-MAC [4]: In B-MAC, a node periodically wakes up and performs receive check for a period T_{rc}. A transmitter is required to transmit long preambles (T_W) before a data packet is sent. A receiver receives incoming messages and a half of preambles on average. Similar, we have:

$$L_h = (h-1)(T_{CW}/2 + T_W + T_{pkt}) + T_{CW}/2 + T_{pkt} \tag{27}$$

$$DC_{lx} = T_{rc}/T_W \tag{28}$$

$$DC_{tx} = F_{out}(T_W + T_{pkt}) \tag{29}$$

$$DC_{rx} = F_{in}(T_W/2 + T_{pkt}) \tag{30}$$

A node overhears a half of preambles on average and header of a packet before it goes back to sleep as the packet is not intended to it. Because sink neighbor nodes do not send preamble, the probability of overhearing a message from these nodes is the ratio of the packet transmission duration to T_W ($P_{over} = T_{pkt}/T_W$). We assume that sink neighbor nodes have a half of interfering nodes in the same level and a node overhears on average a half of a packet.

$$DC_{over} = \begin{cases} (F_{inter}/2)(T_W/2 + T_{hdr}) + \\ (F_{inter}/2)P_{over}(T_{pkt}/2) & \text{if h = 1} \\ F_{inter}(T_W/2 + T_{hdr}) & \text{otherwise} \end{cases} \tag{31}$$

D-MAC [3]:

Latency: A packet is delayed by a L/R slot and T_G in each hop. Average delivery latency is calculated as follows.

$$L_h = (h-1)(\mu + T_G) + \mu \tag{32}$$

Duty cycle:

Listening: A node periodically wakes up to listen for a full slot μ and T_G if there is no incoming packet.

$$DC_{lx} = T_G/T_W + (\mu/T_W - F_{in}\mu) \tag{33}$$

Transmitting: A node transmits data packets and synchronization messages. It also pays an overhead for switching the radio to transmitting mode.

$$DC_{tx} = F_{out}T_{pkt} + F_sT_s + (F_{out} + F_s)T_{transition} \tag{34}$$

Receiving: a node receives input traffic and synchronization messages from its neighbor nodes

$$DC_{rx} = F_{in}T_{pkt} + CF_sT_s \tag{35}$$

A node listens an additional slot to predict incoming data whenever it receives a message from its $|C|$child nodes.

$$DC_{add} = F_{in}\mu + |C|F_s\mu \tag{36}$$

AS-MAC and WiseMAC: AS-MAC [8] and WiseMAC [7] maintain a neighbor table's polling schedule by exchanging schedule information among nodes to reduce the preamble length or sender's idle listening. In each hop, a message is delayed by $T_W/2$ on average due to the sleep latency. Due to limited space, we only present results of WiseMAC.

$$L_h = (h-1)(T_W/2 + T_{CW} + T_G + T_{pkt}) + T_{CW} + T_{pkt} \tag{37}$$

$$DC_{lx} = T_{rc}/T_W + F_{out}(T_{CW}/2) \tag{38}$$

$$DC_{tx} = F_{out}(T_G + T_{pkt}) + F_{in}T_S \tag{39}$$

$$DC_{rx} = F_{in}(T_G/2 + T_{pkt}) + F_{out}T_S \tag{40}$$

The probability of overhearing a message is proportional with the length of its transmission ($P_{over} = (T_G + T_{pkt})/T_W$). Because WiseMAC sends a data packet train instead of preamble, thus a node will overhear $T_{pre} = min(T_G, (L_{hdr} + L_P) \times t_B)$ and the header of the adjacent packet before it realizes that the message is not for it. For the same assumption for sink neighbors as in B-MAC, a sink neighbor node only overhears a half of a message on average.

$$DC_{over} = \begin{cases} (F_{inter}/2)P_{over}(T_{pre}/2 + T_{hdr}) & \\ +(F_{inter}/2)P_{over}(T_{pkt}/2) & \text{if h = 1} \\ F_{inter}P_{over}(T_{pre}/2 + T_{hdr}) & \text{otherwise} \end{cases} \tag{41}$$

Analysis: The result (32) shows that the synchronous protocol, D-MAC, achieves a low packet delivery latency as the latency only depends on μ and T_G. However, results 33–(36) indicate that a node consumes a large proportion of energy for transmitting and receiving synchronization messages. While a node receives data packets only from its child nodes, it receives synchronization messages from all neighbor nodes for clock synchronization. As a result, the overall duty cycle of D-MAC is much higher than other protocols. In case of B-MAC, results for both energy efficiency and latency from (27) to (30) tightly depend on the wakeup interval T_W. A node is delayed at least T_W in each hop. When T_W increases, both of the energy consumption and the latency increase. In case of AS-MAC and WiseMAC 37–(41), although the latency result (37) is lower than that of B-MAC, it is still strictly proportional to $T_W/2$. A packet can only be forwarded one hop per an interval because the average delay per hop is greater than $T_W/2$. RI-MAC and A-MAC spend as a similar cost as L-MAC for transmitting beacon messages. However, their average one-hop latency is still greater than $T_W/2$. Therefore, the optimal setting for wakeup interval in these MAC protocols is within a limited range (e.g. one second), even for low data applications, to achieve a reasonable trade-off between energy efficiency and latency. This is obviously inefficient as nodes have to wake up frequently with many idle intervals. In case of L-MAC, the result (13) shows that the latency is fairly independent with T_W (note that $T_G < <T_W$). A node using L-MAC only sends and receives data packets and performs low power probing (beacons) following primitives of the receiver-initiated approach, as presented in results 14–(17), without paying extra overhead for synchronization or schedule information exchanging, whereas D-MAC, WiseMAC, and AS-MAC do. In B-MAC, long preamble transmission is required. The energy consumption of L-MAC is inversely proportional to wakeup interval T_W. To put it another way, when the wakeup interval increases, changes in delivery latency is negligible while energy consumption is reduced significantly. This permits to set a longer wakeup interval in low data rate applications, to allow nodes sleep longer compared to those in high data rate applications, without a significant negative effect on the latency. It thus greatly improves the network lifetime for low data rate applications compared with other MAC protocols.

5. Performance evaluation

We now move to evaluate L-MAC and conduct comparison studies. To compare energy efficiency, we select AS-MAC (i.e., a schedule learning MAC protocol) as the best representative for energy-efficient asynchronous schedule learning MAC protocol [16]. We do not compare PW-MAC since its pseudo-random function parameter has a problem for maintaining information consistence if we adjust parameters for optimization [16]. To compare delivery latency, we select D-MAC as the best representative for low delivery latency MAC protocols [1,16]. We also compare RI-MAC and A-MAC which are other two state-of-the-art receiver-initiated MAC protocols. Note that the issues related to probe beacon transmission such as overhead and collision in the receiver-initiated approach have been discussed and evaluated in both A-MAC and RI-MAC which show that the receiver-initiated MAC protocol provides more benefits than the sender-initiated protocol. Therefore, we do not repeat those evaluations. The performance of B-MAC and WiseMAC is also presented to show different steps in the evolution of MAC protocols in duty-cycled WSNs [16].

We validate the proposed MAC model by comparing model results with results of completed and time consuming simulations. We use simulation based on TOSSIM for large scale evaluation, and testbed on TelosB motes for small-scale experiments to validate the correctness of our simulations.

5.1. Implementation and system configuration

5.1.1. Implementation

We implemented L-MAC under the UPMA framework [37] in TinyOS for CC2420 Telosb motes, as illustrated in Fig. 4. The basic components include Radio Core and MacC. The MacC component consists of main modules and functions for L-MAC. LMACSchedulerC performs most of key functionalities of L-MAC, including wakeup/sleep scheduling, sleep period calculating, beacon generation, multi-packet mode trigger, retransmission and radio power control. The Radio Core component is used to manage packet transmission and reception. The L-MAC Adaptation Code is responsible for clear channel assessment and backoff control. A beacon retransmission is also triggered by this module if a failed beacon transmission is detected. L-MAC employs packet preloading functionality from RI-MAC but adapting it for beacon retransmission. The preloading functionality is not used for retransmitted beacons and control messages in the setting-up phase. The reason is that those messages are attached with the past active period T_{pap} and

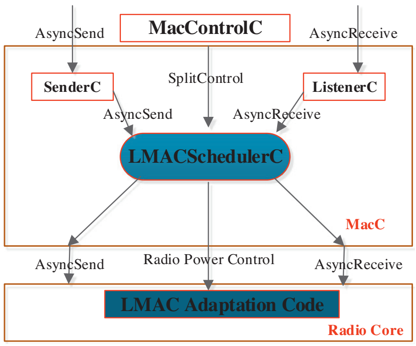


Fig. 4. Implementation of L-MAC in TinyOS.

the sleep period, respectively, which are computed in real time just before the packet is sent.

5.1.2. System configuration

Table 1 presents the detailed parameters used in our simulations and experiments. Other parameters are set to default values of TOSSIM's radio model for CC2420 (i.e., closest-fit pattern matching (CPM) noise model, meyer-heavy.txt noise trace). The wakeup interval of L-MAC is configured with a larger range than other protocols because only with L-MAC, obtained performance still satisfies the QoS requirements (defined in subsection V.C.1) in such a range. To measure the duty cycle, we record changes in the radio's states and use a counter to accumulate the time period using in each state. At the end of simulation, we calculate the average duty cycle and report average results of 5 runs. For the latency, we report the average end-to-end latency of packet generated at leave nodes.

5.2. Validation

First, we validate our analytical model by comparing with simulation results. The target of our model is to capture the main performance characteristics of each protocol. We leave out many implementation details and simplify our model to allow for fast evaluation and to keep the model tractable so that readers can easily understand the trade-off problem and the performance comparison of those MAC protocols. The matter is whether or not the analytical model is authentic enough. We carry out simulations on a binary tree topology with various number of nodes, hops (2–10 hops) and traffic rate (10^{-1}–10^{0}Hz). Obtained packet delivery latency and energy consumption results are compared with results from the analytical model. Fig. 5(a) and (b) shows correlation between simulation and model results. Fig. 5(a) shows that the end-to-end packet deliver latency results obtained by simulations is only about 12% on average (ranging from 5% to 19%) higher than that as estimated by the model. Fig. 5(b) shows that the duty cycle results obtained by simulations is within 18% on average (ranging from 13% to 31%) higher than that as predicted by the model. This is because the analytical model is simplified compared to the full simulations. Importantly, the performance trend calculated by our model has strong coherence with simulation results, which shows that our analytical model is accurate enough to capture main behaviors of each protocol's performance.

5.3. Simulation evaluation

We consider two scenarios for large-scale simulation: a concentric circular ring network which is described in our analytical model, and a grid network.

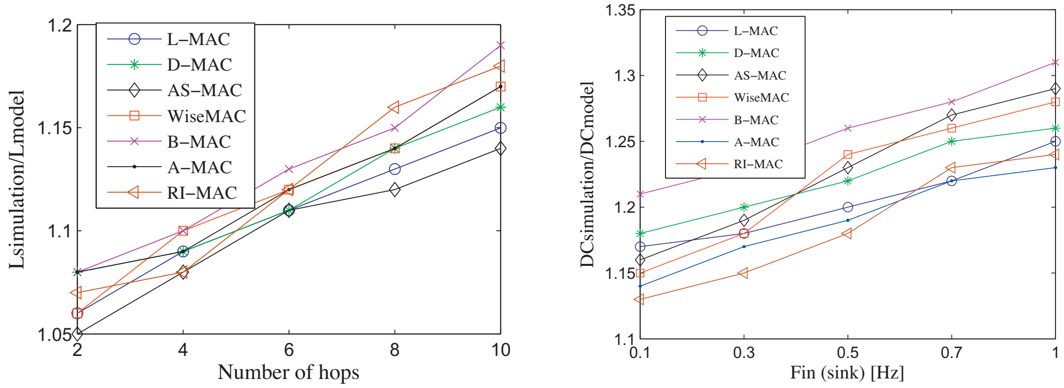
5.3.1. Concentric circular ring topology

We deploy a tree-based concentric circular ring network with 126 nodes. In particular, the network consists of a set of five rings (maximum hop = 5) with a uniform density of five neighbors per a node. The sink node is the central point and every leave node is a data source. Each source generates a data packet every 60 s and then forwards it to the sink.

As shown in Fig. 6(a) and (b), L-MAC achieves the lowest duty cycle compared to other protocols and similar packet delivery latency as D-MAC, with the same wakeup interval. When wakeup interval increases, the duty cycle of most MAC protocols decreases because the sleep period of sensor nodes are extended, except B-MAC. In B-MAC, the preamble transmission overhead is proportional to the length of wakeup interval, thus power consumption increases when wakeup interval increases. The duty cycle of D-MAC is higher than others as a large amount of energy is required for synchronization. AS-MAC and WiseMAC achieves a much lower duty cycle than B-MAC and D-MAC as their idle listening overhead and preamble transmission overhead are scaled down by enabling nodes to learn the wakeup schedule of others. Note that an overhead for exchanging schedule information is required. Similar to RI-MAC and A-MAC, a node in L-MAC spends small overhead for sending beacons, but the duty cycle of L-MAC is significantly smaller than those of RI-MAC and A-MAC. The reasons are: (1) the idle listening overhead of a sender in both RI-MAC and A-MAC are high as the sender has to remain active to wait until its parent wakes up; the waiting period is about $T_W/2$ on average; (2) the idle listening overhead of a node in L-MAC is reduced considerably because a node self coordinates to wakeup earlier than its parent. When wakeup interval increases, the duty cycle in L-MAC, RI-MAC and A-MAC is reduced as overhead for sending beacons degrades rapidly. The duty cycle of L-MAC, AS-MAC and WiseMAC is lower than 1% when wakeup interval is greater than 2 s.

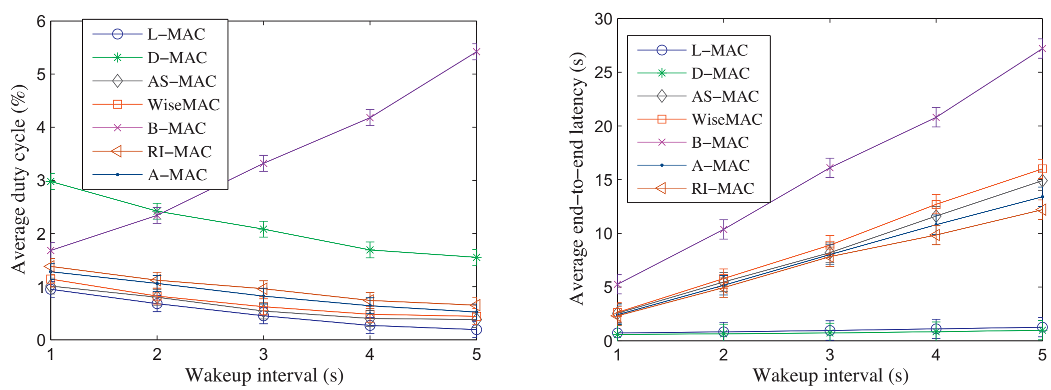
Fig. 6(b) shows a different trend in end-to-end packet delivery latency when comparing L-MAC to AS-MAC and WiseMAC. The latency in both AS-MAC and WiseMAC increases rapidly to 15 s when wakeup interval increases as their sleep latency is proportional to the length of wakeup interval. On the contrary, the latency of L-MAC remains stably around 1s. L-MAC achieves the second lowest end-to-end delivery latency, just slightly higher than that of D-MAC. This is due to the fact that child nodes in L-MAC are designed to self-adapt their wakeup time to be closely earlier than their parent node. Furthermore, nodes in a route to the sink schedule their wakeup time following a staggered pattern. As a result, packets from any node are forwarded continuously to the sink, thus reducing delivery latency. This is the key feature of L-MAC to resolve the sleep latency problem.

To give a comprehensive picture about the trade-off of different MAC protocols and compare their energy efficiency under the **same QoS requirement**, we carry out the following experiments. We first assume a QoS requirement including an upper bound value for average one-hop latency $L_{one-hoplatency}^{upper-bound}$ (i.e., the maximum acceptable latency) and a lower bound value for packet delivery ratio $PDR^{lower-bound}$ (i.e., the minimum acceptable PDR). In experiments, we use $L_{one-hoplatency}^{upper-bound} = 1$ s and $PDR^{lower-bound} = 95\%$ which are reasonable QoS requirements for many common WSN applications. Following the requirement, the maximum acceptable latency for a 5 hops end-to-end route is 5 s. We then run simulations with various wakeup intervals for each MAC protocol to obtain duty cycle results and their corresponding packet delivery latency results which satisfy the QoS requirement. The purposes are to (1) discover the lowest duty cycle that a MAC protocol can achieve while the requirements for packet delivery latency and packet delivery ratio are still satisfied; (2) find out that at a certain degree of energy efficiency (i.e., duty cycle) of a MAC protocol, how long it



(a) Delivery latency (b) Duty cycle

Fig. 5. Correlation between model and simulation results ($L_{simulation}$ and L_{model} stand for latency results, while $DC_{simulation}$ and DC_{model} stand for duty cycle results, obtained by the simulation and the model, respectively).



(a) Average radio duty cycle (b) Average end-to-end delivery latency

Fig. 6. Results with concentric circular ring topology as the wakeup interval varies.

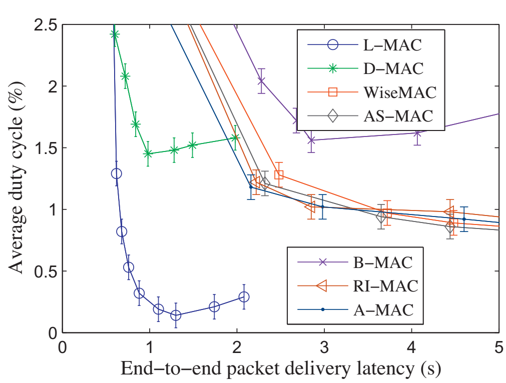


Fig. 7. Average duty cycle vs. corresponding end-to-end packet delivery latency values under QoS requirements of $PDR^{lower-bound} = 95\%$ and $L^{upper-bound}_{one-hoplatency} = 1$ s.

requires to deliver a packet (i.e., packet delivery latency). The former is to compare energy efficiency of MAC protocols while the later is used to find which protocol achieves a better trade-off between energy efficiency and latency. The results are shown in Fig. 7. Each point (y, x) in the figure presents two values: average duty cycle (i.e., the energy efficient indicator) and its corresponding packet delivery latency, respectively.

The figure shows that L-MAC achieves a significant improvement in term of energy efficiency compared to other MAC protocols. In particular, while the lowest duty cycle achieved by L-MAC is 0.14%, that of AS-MAC, WiseMAC, RI-MAC, and A-MAC is 0.8%, 0.82%, 0.89%, and 0.85%, respectively. This figure can also be interpreted in another way as follows. The origin $O(0, 0)$ indicates the minimum energy consumption and the minimum latency values (i.e., (0, 0)). A graph, which is closer to the origin, presents a better trade-off between energy efficiency and packet delivery latency (i.e., achieving better latency within the same or lower duty cycle). By comparing distances to the origin from points with similar duty cycle values in each graph, we see that L-MAC achieves the best trade-off among the MAC protocols.

In particular, the results show that L-MAC achieves low duty cycle as well as low packet delivery latency at the same time; D-MAC achieves low packet delivery latency with a tradeoff of higher energy consumption while AS-MAC, WiseMAC, D-MAC, RI-MAC, A-MAC, and B-MAC achieve a lower duty cycle compared to D-MAC, but higher packet delivery latency is a trade-off. Note that the figure shows **only points** with duty cycle and packet delivery latency values of simulations which **satisfy the QoS requirement**. Other results, which do not satisfy the QoS requirement, are not plotted to make results of the MAC protocols comparable. For example, in case of L-MAC, after achieving the lowest duty cycle 0.14% corresponding the end-to-end packet delivery latency of 1.3 s, the duty cycle and latency values of L-MAC start increasing as shown in Fig. 7, and its packet delivery ratio starts decreasing significantly as shown in Fig. 8 (i.e., from the wakeup interval of 5–7 s). When the packet delivery ratio of L-MAC does not satisfy the QoS requirement (i.e., lower than 95%), its corresponding duty cycle and latency values are also invalid and not shown in Fig. 7.

Fig. 8 shows that the packet delivery ratio of L-MAC is reduced to 92.4% and 87.3% corresponding with a wakeup interval of 6 s

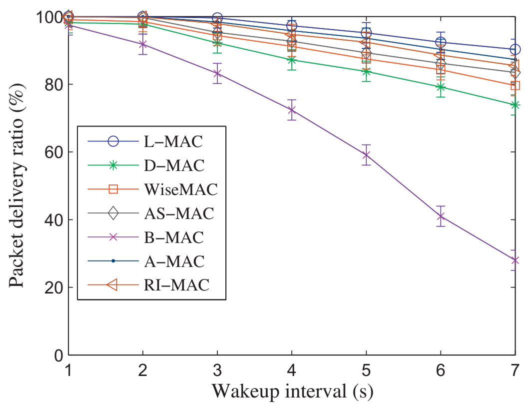


Fig. 8. Packet delivery ratio results with concentric circular ring topology.

and 7 s, respectively. This is due to the fact that a larger wakeup interval under a fixed data rate leads to a higher number of packets need to be forwarded in each wakeup time of a node, resulting in a higher probability of collision and longer queue delay. This characteristic is used to determine the lowest duty cycle of L-MAC. Similar phenomenon is also observed in cases of AS-MAC, WiseMAC, D-MAC, RI-MAC, A-MAC, and B-MAC. However, by carefully considering collision avoidance, L-MAC achieves a fairly good result for packet delivery ratio compared to others. The result of B-MAC is very low in case of large wakeup intervals because the probability of collision increases proportionally with its preamble transmission duration.

5.3.2. Grid topology

A 100-node (10 × 10) grid network is deployed. Each node is 70 m from its neighbors, and has a transmission range of 100 m. The sink node is placed in the center. We use CTP [35] as the upper layer protocol to create a data collection tree. The packet generation interval varies from 30 s to 300 s. We define **communication cost** as the overhead which a node spends for sending/receiving data packets when both sender and receiver are active, including the costs for sending/receiving, retransmission, collision avoidance, and back-off; **duty cycle cost** as the overhead a node which spends for other operations to enable its communication with other nodes, including idle listening, receive check, synchronization, schedule information exchanging, beacon/preamble transmission, etc; **sleep latency** as the delay from the time when a sender has packets to send to the time when both sender and its receiver wake up; and **communication latency** as the delay from the time when both sender and its receiver wake up to the time when packets are delivered successfully to the receiver.

We apply the QoS requirement (i.e., $PDR^{lower-bound} = 95\%$ and $L^{upper-bound}_{one-hoplatency} = 1\,\text{s}$) and run simulations with various wakeup intervals for each protocol to determine the minimum achieved duty cycle of each protocol. As shown in Fig. 9(a), when packet generation interval increases, duty cycle decreases for all the protocols as traffic load is cut down. The corresponding wakeup intervals of AS-MAC, WiseMAC, RI-MAC, A-MAC and B-MAC are lower than 2 s as their latency is over $L^{upper-bound}_{one-hoplatency}$ within that setting for wakeup interval. The tradeoff limits their achievement in term of energy efficiency. On the contrary, L-MAC achieves much higher energy efficiency compared to other MAC protocols as a higher value of wakeup interval is used for lower data rate scenarios without a significant negative effect on delivery latency. As a result, not only traffic load but also listening and beacon transmission overhead in low data rate scenarios are reduced significantly. In all cases, L-MAC achieves the highest energy efficiency.

We provide deep insight into duty cycle for each protocol by analyzing the relationship between duty cycle cost and communication cost at a packet generation interval of 60 s. The results are presented in Fig. 9(b). We observe that in all the protocols duty cycle cost is dominant compared to communication cost. While their communication costs are similar, L-MAC achieves the lowest duty cycle cost. The main reasons are as follows: (1) L-MAC applies a longer wakeup interval to achieve the same QoS requirement, thus nodes in L-MAC wake up less frequently compared to other MAC protocols; (2) a sender (i.e., a child node) is designed to wake up only slightly earlier than its receiver (i.e., a parent node), so its idle listening period and transmission period are shortened significantly. Although nodes in L-MAC pay an overhead for sending short beacons, the benefit gained from decreasing idle listening and transmission periods is more significant.

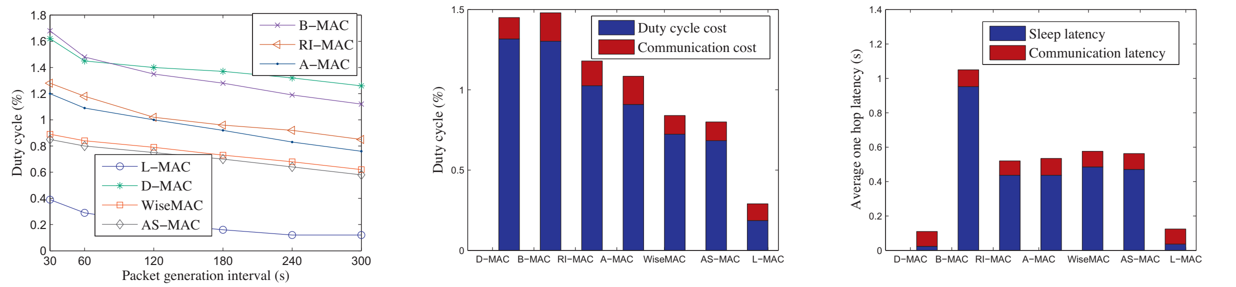
Fig. 9(c) presents an insight of one-hop delivery latency of at a wakeup interval of 1 s. While sleep latency in cases of B-MAC, RI-MAC, A-MAC, AS-MAC, and WiseMAC is dominant compared to communication latency, sleep latency in L-MAC and D-MAC is even lower than communication latency. This clearly demonstrates the advantage of our staggered scheduler.

5.4. Testbed experiments

We set up our testbed experiments in three scenarios: a chain topology which consists of 10 TelosB nodes indexed from 0 (the sink node) to 9, a binary tree topology which consists of 30 TelosB nodes rooted by the sink node, and a random deployment of 30 nodes with the CTP routing protocol running on the top to create the network topology. All the nodes, except the sink node, generate a data packet every 30 s. Therefore, multiple flows of data packets can interfere each other. We run various MAC protocols including L-MAC, RI-MAC, A-MAC, AS-MAC, and B-MAC on the same topology and measure their performance. By tracking the L-MAC set up phase completion time, we observe that each L-MAC node completes its L-MAC set up phase just a moment after its CTP set up phase. Records show that leave nodes in all topologies complete their L-MAC set up phase within a cycle later compared to their CTP set up phase. In addition, within two cycles on average from the setup completion time, each node achieves a small and fairly stable offset compared to its parent wakeup time (i.e., approximately 1.5 ms). The time overhead for the setup phase is negligible compared to the network lifetime.

To validate our simulation results, we repeat the same tests as in Fig. 9(b) and (c) for a testbed with the binary tree topology and the grid topology. Note that both simulation and testbed share the same TinyOS source code. Results obtained from the topologies are similar and comparable with simulation results, so we present only results obtained from the binary tree topology. The results are presented in Fig. 10(a) and (b). We observe slightly higher duty cycle and delivery latency in testbed than simulation, especially communication cost and communication latency. This is due to hardware delay and differences between simulation and real deployment. However, the relative performance among MAC protocols remains the same. This clearly validates our simulation results.

Fig. 10(c) shows a comparison of duty cycle between L-MAC and RI-MAC. Although a sender in RI-MAC does not occupy the channel when it has packets to send, the sender still spends a considerable overhead in idle listening to wait for its receiver waking up. As a result, the duty cycle of the sender is much higher than that of the receiver. On the contrary, in L-MAC, a sender wakes up closely earlier than its receiver, thus the sender's duty cycle is cut down to a similar amount of the receiver. It is obvious that L-MAC achieves better energy balancing between sender and receiver than RI-MAC.

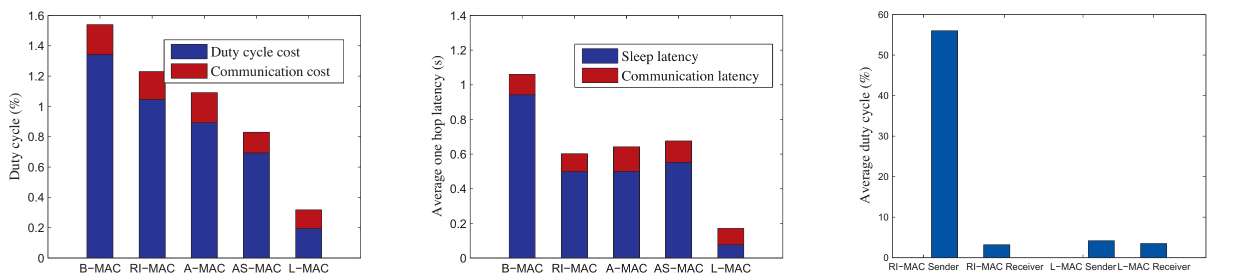


(a) Minimum achieved duty cycle under QoS requirements of $PDR^{lower-bound} = 95\%$ and $L^{upper-bound}_{one-hoplatency} = 1s$

(b) Duty cycle cost vs. communication cost

(c) One-hop sleep latency vs. communication latency

Fig. 9. Results with grid topology as the packet generation interval varies.

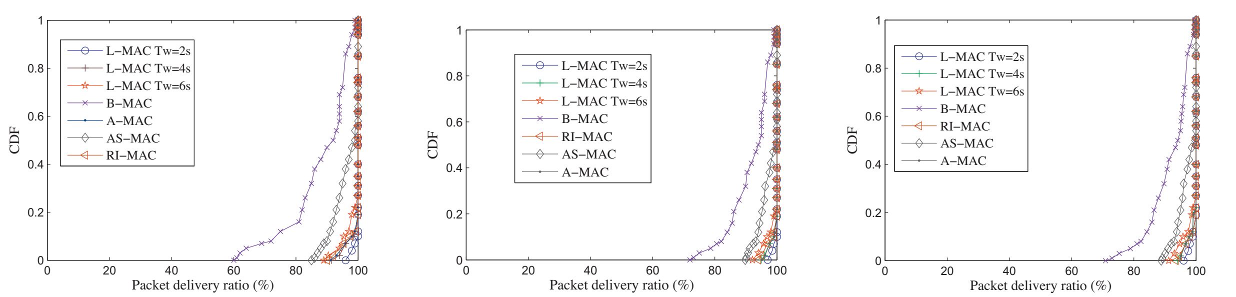


(a) Duty cycle cost vs. communication cost

(b) One-hop sleep latency vs. communication latency

(c) A comparison of sender's and receiver's duty cycle between L-MAC and RI-MAC.

Fig. 10. Results obtained from testbed with the binary tree topology.



(a) CDF of delivery ratio in the chain topology

(b) CDF of delivery ratio in the binary tree topology

(c) CDF of delivery ratio in random deployment

Fig. 11. Performance for packet delivery ratio.

We also conduct 20 experiments for each topology to explore the average packet delivery ratio of L-MAC, RI-MAC, A-MAC, AS-MAC, and B-MAC under interference scenarios of multiple data flows in intra-branch (chain topology) and inter-branch (binary tree topology) of data collection trees. We run L-MAC with different wakeup intervals of 2 s, 4 s, and 6 s. For other MAC protocols, wakeup interval values corresponding with the minimum duty cycle results from the experiment in Fig. 10(a) are used (2 s for both AS-MAC, RI-MAC, and A-MAC, 0.5 s for B-MAC). The results are presented in Fig. 11(a), (b) and (c). The packet delivery ratio of MAC protocols in case of the chain topology is worst than that of the binary tree topology and random deployment. This is because there is a higher chance of collision among packet flows in one route in the chain topology. The collision probability is also proportional to the channel occupied period of senders and receivers to send a data packet. For this reason, B-MAC presents the lowest packet delivery ratio in both topologies. A-MAC, RI-MAC, and L-MAC achieves a fairly good delivery ratio as a senders do not occupy the channel until the sender receives a beacon from its receiver. L-MAC with a lower wakeup interval (i.e., 2 s) achieves a higher delivery ratio compared to L-MAC with a higher wakeup interval (i.e., 6 s), as the traffic load per wakeup at a wakeup interval of 6 s is fairly high. This characteristic helps determine the lowest duty cycle of L-MAC under a specific QoS requirement.

6. Conclusion

This paper presents the comprehensive analysis, design, and evaluation of L-MAC, which enables child nodes to coordinate their wakeup time to their parent without synchronization or exchanging schedule information. L-MAC is designed to resolve the sleep latency problem to allow nodes in low data rate applications to sleep longer to save energy without a significant negative effect on delivery latency. Notably, the design of L-MAC is very simple and easy to implement. We implemented L-MAC in TinyOS within the UPMA framework. Through our analysis and evaluation, we show that L-MAC outperforms current asynchronous protocols in terms of energy efficiency and packet delivery latency. In particular, experimental results, obtained under a predefined QoS requirement as described above, show that L-MAC achieves a significant improvement of 3.8 times in term of energy efficiency and of 7 times in term of end-to-end packet delivery latency, compared to those of state-of-the-art MAC protocols. Experimental results indicate that L-MAC achieves a better trade-off between energy efficiency and latency compared to existing studies. In addition, experiments under various network scenarios , also reveal that the lower the application data rate is, the higher the improvement L-MAC achieves.

Acknowledgment

This work is supported by Soongsil University, and partially funded by RMIT University, and by National Science Foundation of China grants number 61272456 and 61472312.

References

[1] G. Anastasi, M. Conti, M.D. Francesco, A. Passarella, Energy conservation in wireless sensor networks: a survey, Ad Hoc Netw. 7 (3) (2009) 537–568. http://dx.doi.org/10.1016/j.adhoc.2008.06.003.

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

[3] G. Lu, B. Krishnamachari, C.S. Raghavendra, An adaptive energy-efficient and low-latency mac for tree-based data gathering in sensor networks: research articles, Wireless Commun. Mob. Comput. 7 (7) (2007) 863–875, doi:10.1002/wcm.v7:7.

[4] J. Polastre, J. Hill, D. Culler, Versatile low power media access for wireless sensor networks, in: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, in: SenSys '04, ACM, New York, NY, USA, 2004, pp. 95–107, doi:10.1145/1031495.1031508.

[5] M. Buettner, G.V. Yee, E. Anderson, 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, in: SenSys '06, ACM, New York, NY, USA, 2006, pp. 307–320, doi:10.1145/1182807.1182838.

[6] D. Moss, P. Levis, BoX-MACs: Exploiting Physical and Link Layer Boundaries in LowPower Networking, Technical Report, 2008.

[7] A. El-Hoiydi, J.-D. Decotignie, WiseMAC: An Ultra Low Power MAC Protocol for Multi-hop Wireless Sensor Networks, Lecture Notes in Computer Science, vol. 3121, Springer, Berlin, Heidelberg, 2004, doi:10.1007/978-3-540-27820-7_4.

[8] B. Jang, J.B. Lim, M.L. Sichitiu, An asynchronous scheduled {MAC} protocol for wireless sensor networks, Comput. Netw. 57 (1) (2013) 85–98. http://dx.doi.org/10.1016/j.comnet.2012.09.002.

[9] Y. Sun, O. Gurewitz, D.B. Johnson, Ri-mac: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks, in: Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, in: SenSys'08, ACM, New York, NY, USA, 2008, pp. 1–14, doi:10.1145/1460412.1460414.

[10] J. Li, C. Blake, D.S. De Couto, H.I. Lee, R. Morris, Capacity of ad hoc wireless networks, in: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, in: MobiCom '01, ACM, New York, NY, USA, 2001, pp. 61–69, doi:10.1145/381677.381684.

[11] W. Ye, F. Silva, J. Heidemann, Ultra-low duty cycle mac with scheduled channel polling, in: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, in: SenSys '06, ACM, New York, NY, USA, 2006, pp. 321–334, doi:10.1145/1182807.1182839.

[12] T. van Dam, K. Langendoen, An adaptive energy-efficient mac protocol for wireless sensor networks, in: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, in: SenSys '03, ACM, New York, NY, USA, 2003, pp. 171–180, doi:10.1145/958491.958512.

[13] L. Tang, Y. Sun, O. Gurewitz, D. Johnson, Pw-mac: an energy-efficient predictive-wakeup mac protocol for wireless sensor networks, in: INFOCOM, 2011 Proceedings IEEE, 2011, pp. 1305–1313, doi:10.1109/INFCOM.2011.5934913.

[14] W. Pak, K.-T. Cho, J. Lee, S. Bahk, W-mac: Supporting ultra low duty cycle in wireless sensor networks, in: Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE, 2008, pp. 1–5, doi:10.1109/GLOCOM.2008.ECP.79.

[15] S. Liu, K.-W. Fan, P. Sinha, Cmac: An energy efficient mac layer protocol using convergent packet forwarding for wireless sensor networks, in: Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON '07. 4th Annual IEEE Communications Society Conference on, 2007, pp. 11–20, doi:10.1109/SAHCN.2007.4292813.

[16] P. Huang, L. Xiao, S. Soltani, M. Mutka, N. Xi, The evolution of mac protocols in wireless sensor networks: a survey, Commun. Surv. Tut. IEEE 15 (1) (2013) 101–120, doi:10.1109/SURV.2012.040412.00105.

[17] M. Doudou, D. Djenouri, N. BADACHE, Survey on latency issues of asynchronous mac protocols in delay-sensitive wireless sensor networks, Commun. Surv. Tut IEEE 15 (2) (2013) 528–550, doi:10.1109/SURV.2012.040412.00075.

[18] V. Esteves, A. Antonopoulos, E. Kartsakli, M. Puig-Vidal, P. Miribel-CatalÃ, C. Verikoukis, Cooperative energy harvesting-adaptive mac protocol for wbans, Sensors 15 (6) (2015) 12635, doi:10.3390/s150612635.

[19] A. Ibarra, A. Antonopoulos, E. Kartsakli, C. Verikoukis, Heh-bmac: hybrid polling mac protocol for wbans operated by human energy harvesting, Telecommun. Syst. 58 (2) (2015) 111–124, doi:10.1007/s11235-014-9898-z.

[20] T. Dinh, T. Gu, A novel metric for opportunitic routing in heterogeneous duty cycled wireless sensor network, in: Network Protocols (ICNP), 2015 23rd IEEE International Conference on, 2015, pp. 1–10, doi:10.1109/ICNP.2015.6733599.

[21] S. Sudevalayam, P. Kulkarni, Energy harvesting sensor nodes: Survey and implications, Commun. Surv. Tut. IEEE 13 (3) (2011) 443–461, doi:10.1109/SURV.2011.060710.00094.

[22] E. Kartsakli, A.S. Lalos, A. Antonopoulos, S. Tennina, M.D. Renzo, L. Alonso, C. Verikoukis, A survey on m2m systems for mhealth: a wireless communications perspective, Sensors 14 (10) (2014) 18009, doi:10.3390/s141018009.

[23] G. Halkes, K. Langendoen, Crankshaft: An energy-efficient mac-protocol for dense wireless sensor networks, in: K. Langendoen, T. Voigt (Eds.), Wireless Sensor Networks, Lecture Notes in Computer Science, vol. 4373, Springer, Berlin Heidelberg, 2007, pp. 228–244, doi:10.1007/978-3-540-69830-2_15.

[24] I. Rhee, A. Warrier, M. Aia, J. Min, M. Sichitiu, Z-mac: A hybrid mac for wireless sensor networks, Netw. IEEE/ACM Trans. 16 (3) (2008) 511–524, doi:10.1109/TNET.2007.900704.

[25] Y. Sun, S. Du, O. Gurewitz, D.B. Johnson, Dw-mac: A low latency, energy efficient demand-wakeup mac protocol for wireless sensor networks, in: Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, in: MobiHoc '08, ACM, New York, NY, USA, 2008, pp. 53–62, doi:10.1145/1374618.1374627.

[26] S. Du, A. Saha, D. Johnson, Rmac: a routing-enhanced duty-cycle mac protocol for wireless sensor networks, in: INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE, 2007, pp. 1478–1486, doi:10.1109/INFCOM.2007.174.

[27] Y. Peng, Z. Li, D. Qiao, W. Zhang, Delay-bounded mac with minimal idle listening for sensor networks, in: INFOCOM, 2011 Proceedings IEEE, 2011, pp. 1314–1322, doi:10.1109/INFCOM.2011.5934914.

[28] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J.M. Liang, A. Terzis, A-mac: a versatile and efficient receiver-initiated link layer for low-power wireless, ACM Trans. Sen. Netw. 8 (4) (2012) 30:1–30:29, doi:10.1145/2240116.2240119.

[29] M. Doudou, D. Djenouri, N. Badache, Survey on latency issues of asynchronous mac protocols in delay-sensitive wireless sensor networks, Commun. Surv. Tut. IEEE 15 (2) (2013) 528–550, doi:10.1109/SURV.2012.040412.00075.

[30] K. Langendoen, A. Meier, Analyzing mac protocols for low data-rate applications, ACM Trans. Sen. Netw. 7 (1) (2010) 10:1–10:34, doi:10.1145/1806895.1806905.

[31] R. Carrano, D. Passos, L. Magalhaes, C. Albuquerque, Survey and taxonomy of duty cycling mechanisms in wireless sensor networks, Commun. Surv. Tut. IEEE 16 (1) (2014) 181–194, doi:10.1109/SURV.2013.052213.00116.

[32] Z. Li, M. Li, Y. Liu, Towards energy-fairness in asynchronous duty-cycling sensor networks, ACM Trans. Sen. Netw. 10 (3) (2014) 38:1–38:26, doi:10.1145/2490256.

[33] L. Tang, Y. Sun, O. Gurewitz, D.B. Johnson, Em-mac: a dynamic multichannel energy-efficient mac protocol for wireless sensor networks, in: Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing, in: MobiHoc '11, ACM, New York, NY, USA, 2011, pp. 23:1–23:11, doi:10.1145/2107502.2107533.

[34] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J.M. Liang, A. Terzis, Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless, in: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '10, ACM, New York, NY, USA, 2010, pp. 1–14, doi:10.1145/1869983.1869985.

[35] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, Collection tree protocol, in: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '09, ACM, New York, NY, USA, 2009, pp. 1–14, doi:10.1145/1644038.1644040.

[36] T. Dinh, T. Gu, The performance trade-off analysis for MAC protocols in WSNs, Technical Report, RMIT, 2014.

[37] K. Klues, G. Hackmann, O. Chipara, C. Lu, A component-based architecture for power-efficient media access control in wireless sensor networks, in: Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, in: SenSys '07, ACM, New York, NY, USA, 2007, pp. 59–72, doi:10.1145/1322263.1322270.

Thanh Dinh received his B.S. degree in information technology from University of Science – Vietnam National University in HoChiMinh city, Vietnam in 2010 and Master degree from Soongsil university, Korea in 2012. He has been pursuing a Ph.D. degree at Soongsil university, Korea. He is currently a Ph.D. candidate in School of Computer Science and Information Technology, RMIT University. His current research interests include wireless sensor and actor networks, Internet of Things, mobile computing and next generation network.

Younghan Kim received the B.S. degree in electronic engineering from Seoul National University, Korea in 1984. He received the M.S. and Ph.D. degrees in electrical engineering from KAIST, Korea in 1986 and 1990, respectively. From 1987 to 1994, he worked in Digicom Institute of Telematics as a senior research engineer, where he developed ISDN and packet switching systems. Since 1994, he has been a professor in the School of Electronics at Soongsil University in Korea. His research interests include wired and wireless networking, QoS, mobile computing and ubiquitous networking. He is a member of IEICE and IEEE.

Tao Gu is an Associate Professor in the School of Computer Science and Information Technology at RMIT University. He received his Bachelor degree from Huazhong University of Science and Technology, and M.Sc. from Nanyang Technological University, Singapore, and Ph.D. in computer science from National University of Singapore. His current research interests include mobile and pervasive computing, wireless sensor networks, distributed network systems, sensor data analytics, cyber physical system, Internet of Things, and online social networks. His publications appear in top journals such as JSAC, TMC, TPDS, and TKDE and top conferences such as INFOCOM, PerCom, and UbiComp. He serves as TPC members for many leading conferences such as INFOCOM, PerCom, etc, and also Journal editorial board members. He is a senior member of the IEEE and a member of the ACM.

Athanasios V. Vasilakos is currently a professor at the Lulea University of Technology,Sweden. He has authored or coauthored more than 200 technical papers in major international journals and conferences. He is the author/coauthor of five books and 20 book chapters in the areas of communications. He served as the general chair, technical program committee chair, and the TPC member for many international conferences. He was an editor or/and guest editor for many technical journals, such as the IEEE Transactions on Network and Service Management, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics,IEEE Transactions on Information Technology in Biomedicine, IEEE Transactions on Computers, ACM Transactions on Autonomous and Adaptive Systems, and theIEEE Journal on Selected Areas in communications special issues of May 2009, January 2011, and March 2011, the IEEE Communications Magazine, ACM/Springer Wireless Networks, and the ACM/SpringerMobile Networks and Applications. He is the founding editor-inchief of theInternational Journal of Autonomous and Adaptive Communications Systemsand theInternational Journal of Arts and Technology. He is the general chair of the Council of Computing of the European Alliances for Innovation. He is a senior member of the IEEE.