An Adaptive Low-Power Listening Protocol for Wireless Sensor Networks in Noisy Environments

Thanh Dinh, Younghan Kim, *Member, IEEE*, Tao Gu, *Senior Member, IEEE*, and Athanasios V. Vasilakos, *Senior Member, IEEE*

***Abstract*—This paper investigates the energy consumption minimization problem for wireless sensor networks running low-power listening (LPL) protocols in noisy environments. We observe that the energy consumption by false wakeups (i.e., wakeup without receiving any packet) of a node in noisy environments can be a dominant factor in many cases while the false wakeup rate is spatially and temporarily dynamic. Based on this observation, without carefully considering the impact of false wakeups, the energy efficient performance of LPL nodes in noisy environments may significantly deviate from the optimal performance. To address this problem, we propose a theoretical framework incorporating LPL temporal parameters with the false wakeup rate and the data rate. We then formulate an energy consumption minimization problem of LPL in noisy environments and address the problem by a simplified and practical approach. Based on the theoretical framework, we design an efficient adaptive protocol for LPL (APL) in noisy environments. Through extensive experimental studies with Telosb nodes in real environments, we show that APL achieves 20%–40% energy efficient improvement compared to existing LPL protocols under various network conditions.**

***Index Terms*—Adaptive low power listening protocol, energy efficiency, energy optimization, noise environment, scheduling algorithm, wireless sensor network.**

I. Introduction

OVER the past few years, duty cycling [1], [2] and low-power listening (LPL) [3]–[5] have been greatly explored for energy saving in wireless sensor networks (WSNs). According to recent extensive surveys [1], [6], [7], LPL with duty cycling is one of the most popular energy efficient techniques for MAC protocols in constrained WSNs. The technique is used widely in real WSN deployments and in the default MAC protocols of TinyOS [8] and Contiki [9], the two common OS frameworks for constrained WSNs.

While LPL achieves energy efficiency to a large extent, its energy saving performance depends much on its temporal parameters such as sleep interval, wakeup period, and extended wakeup period. These temporal parameters determine how long a node should sleep or wake up, which also affect energy consumption of its sender nodes. An interesting observation is that energy consumption of nodes (i.e., a receiver and its sender nodes) shows different behaviors when LPL parameters vary. For example, if we decrease the sleep interval (I_s) of a receiver node, the energy consumption of the receiver will increase because it has to wake up more frequently; however, this may potentially reduce its sender's energy consumption as the senders' required preamble transmission duration can be reduced. Note that the transmission energy cost of a sender node depends on the sleep interval of its receiver (i.e., $I_s/2$ on average in case of Box-MAC [3]). Therefore, when determining optimal LPL parameters for a given node to minimize its energy consumption, we need to consider its senders' energy consumption as well.

In existing LPL protocols, LPL parameters of sensors are usually predetermined through empirical studies [3], [4] under certain network condition. For example, the sleep interval in the LPL MAC used in TinyOS [8] is set to 500 ms by default. These predefined values may achieve optimal performance in a certain network condition, but may work poorly in other conditions. Unfortunately, in real deployments, network condition such as noise and traffic load is highly dynamic. For example, the noise level in an indoor space may change over time. Moreover, even in the same network, network condition may be spatially nonuniform [10], [11]. This poses a big challenge to manually predefine the optimal LPL parameters for each sensor node at different locations over time. Therefore, designing an adaptive LPL protocol that allows sensor nodes to optimize their energy consumption dynamically is crucial.

There have been some prior works [12]–[19] that support adaptive duty cycling to improve energy efficiency. However, there is still a lack of a practical study on the impact of false wakeups to the performance of LPL in noisy environments and to optimize LPL's performance in such a scenario. False wakeups happen by environmental noise being detected as a channel activity, which triggers nodes spuriously wakeup in order to receive packets. However, there are actually no incoming packets.

Recent studies [10], [20], [21] show that noisy environments have become more popular, especially in indoor spaces, because the number of wireless devices, which coexist and share the same unlicensed spectrum 2.4 GHz (i.e., Wi-Fi, bluetooth, ZigBee, and microwave), is increasing significantly. This makes the above practical issue critical to be investigated. In noisy environments, energy consumption of sensor nodes can be seriously affected by false wakeups. However, existing works fail to adapt LPL temporal parameters in such noisy environments. Our experimental study shows that a false wakeup costs over 17 times

Manuscript received May 6, 2016; revised March 15, 2017; accepted June 18, 2017. This work was supported by the Ministry of Science, ICT and Future Planning, Korea, under the Information Technology Research Center Support Program (2017-0-01633) supervised by the Institute for Information and Communications Technology Promotion. *(Corresponding authors: Thanh Dinh; Younghan Kim.)*

T. Dinh and Y. Kim are with the School of Electronic Engineering, Soongsil University, Seoul 06978, South Korea (e-mail: thanhdcn@dcn.ssu.ac.kr; yhkim@dcn.ssu.ac.kr).

T. Gu is with the School of Computer Science, Royal Melbourne Institute of Technology University, Melbourne, VIC 3000, Australia (e-mail: tao.gu@rmit.edu.au).

A. V. Vasilakos is with the Department of Computer Science, Lulea University of Technology, Lulea 97187, Sweden (e-mail: vasilako@ath.forthnet.gr).

Digital Object Identifier 10.1109/JSYST.2017.2720781

more energy than a normal receive check, which is similar to the observation presented in [11]. In addition, our experiments with a real WSN network (presented in next section) indicate that the false wakeup rate of TinyOS-LPL in many locations may rise up to 60%. We show that the energy consumption of a sensor node caused by false wakeups in highly noisy environments can be a dominant factor compared to other sources such as transmitting and receiving, especially in low data rate applications. Importantly, we observe that the false wakeup rate of nodes in the same network is spatially dynamic, and the false wakeup rate of a node at the same location may change over time. Based on these observations, without carefully considering the impact of false wakeup, the performance of LPL may significantly deviate. We argue that it is crucial to study behaviors of LPL performance in such a realistic and popular scenario.

In this paper, we study the practical issue of false wakeup's impact on the performance of LPL protocols under noisy environments. Based on the observations from our experimental studies, we show the limitations of existing LPL protocols, and highlight a necessity to adapt LPL temporal parameters to achieve energy saving under noisy environments. We then formally address the limitations of existing LPL protocols by enabling sensor nodes to self-adapt their LPL parameters whenever the network condition is changed significantly. More specifically, a node optimizes the energy consumption of itself and its senders by adapting its LPL parameters over time. We propose a theoretical framework to capture energy consumption of a receiver node and its corresponding sender nodes in noisy environments. The framework incorporates LPL temporal parameters with the false wakeup rate and the data rate. We then formulate an energy consumption minimization problem of a node in noisy environments. Since our target is to design a practical protocol for constrained sensor nodes, we propose a simplified approach to solve the minimization problem using the extreme value theory. Based on our theoretical framework, we design a practical adaptive protocol for LPL (APL). In APL, a receiver node optimizes its LPL temporal parameters to minimize the total energy consumption by itself and its senders under dynamic network conditions. Through our comprehensive experimental studies, we show that APL achieves 20%–40% energy saving compared to existing LPL protocols under various network conditions.

In summary, this paper makes the following contributions.

1) We propose a theoretical framework, which incorporates LPL temporal parameters with the false wakeup rate and the data rate to optimize the total energy consumption of WSNs under dynamic network conditions (see Section IV).
2) We design a practical adaptive protocol for LPL based on the proposed theoretical framework, to enable a receiver node to dynamically optimize its LPL temporal parameters so that the total energy consumption of itself and its sender nodes is minimized (see Section V).
3) Through our comprehensive experimental studies with real sensors, we show that APL outperforms existing LPL protocols and achieves a significant improvement in terms of energy efficiency in noisy environments (see Section VI).

II. Related Work

LPL and duty cycling are common MAC-layer techniques for energy efficiency in WSNs, where sensors periodically wake up to check the wireless channel for incoming packets.

Duty-cycled MAC protocols for WSNs can generally be categorized into synchronous and asynchronous schemes. In the synchronous scheme [22], [23], MAC procedures work under an assumption of time synchronization among nodes. By synchronizing nodes' active time together, synchronous MAC protocols are normally designed to optimize the packet delivery latency. In this scheme, a node is required to exchange timing information periodically with neighbor nodes for time synchronization. High energy consumption and synchronized precision requirement are two remaining challenges for resource-constrained sensor nodes using synchronous MAC protocols.

Asynchronous MAC protocols [1], [2], [4], [6], [24]–[26] have been proposed to address the above-mentioned limitation. In the asynchronous scheme, the communication among nodes is enabled by LPL, thus eliminating the overhead for time synchronization. In particular, the sender transmits preambles to explicitly alert other nodes about its packet transmission. Other nodes, including the receiver, periodically samples the channel for activity detection. If any channel activity is detected, the nodes wake up in order to receive packets. Extensive survey for LPL-related MAC protocols can be found in [1], [6], [7], and [27], and our previous work [28].

While LPL achieves energy efficiency to a large extent, its energy saving performance depends much on its temporal parameters. Some prior works have been proposed to improve energy efficiency of LPL based on the duty cycling adaptivity. We categorize those prior works based on their adaptive approach. For instant, IDEA [12] uses a centralized approach to tune LPL parameters while GDSIC [13] uses a distributed method to achieve energy fairness. In MiX-MAC [14], Merlin and Heinzelman propose to improve energy efficiency by switching among various duty cycling protocols for different scenarios. In [15] and [16], heuristic methods are used to improve energy efficiency by adapting LPL based on the number of descendants and successfully received packets. ASLEEP [17] focuses on adapting sleep schedules of nodes to match the network demand of periodic data acquisition applications by forming a staggered topology. In [19], Ning and Cassandras propose to adapt sleep time based on objectives and constraints of the network while in [18], a desired sampling period is used as input information for adapting. Recently, several protocols [29]–[31] are proposed to adjust LPL parameters based on traffic patterns. In [11], although the study is not designed for tuning LPL temporal parameters, a new method is proposed to improve energy efficiency by adjusting CCA thresholds. In the literature, there is still a lack of a practical study on the impact of false wakeups to the performance of LPL in noisy environments, which is investigated in this paper.

III. Motivation

A. LPL Operations

LPL is a common mechanism, which has been greatly explored in designing energy-efficient MAC protocols. Although there are several different LPL implementations, their basic design is quite similar. In LPL, a node periodically wakes up (after a sleep interval I_s) to perform receive checks (CCA), as illustrated in Fig. 1(a). If there is no channel activity detected, the node then turns OFF its radio. If the channel is busy, the node wakes up fully and remains active for a wakeup period T_w to listen for incoming packets, as shown in Fig. 1(b) and

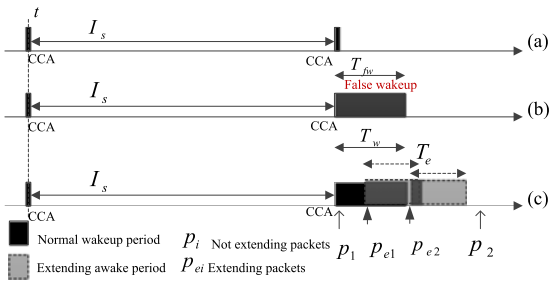

Fig. 1. LPL operations: (a) CCA checks, (b) CCA checks with false wakeup, and (c) CCA checks with received packets.

(c). Fig. 1(b) illustrates a case of false wakeup where a node wakes up, but there are no incoming packets. This false wakeup may be triggered by noise. In Fig. 1(c), the node receives several packets (i.e., p_1, p_{e1}, p_{e2}). Before transmitting a packet, senders transmit preambles until their receiver wakes up. For each packet received, a receiver extends its active time by an extended period T_e [see Fig. 1(c)] because there may be more than one incoming packet or sender. However, a packet may be an extending packet or not, depending on the packet sent time. This will be further analyzed in Section III.

B. LPL Parameters and Dynamically Noisy Environments

I_s: The value of I_s presents the frequency a node wakes up for receive checks. With a low value of I_s, a receiver wakes up more frequently to check for incoming packets, which results in high energy consumption, but this potentially shortens the preamble transmission duration of senders, thus reducing senders' energy cost. With a high value of I_s, the cost for receive checks of receivers is reduced, but the cost for sending packets increases. For energy efficiency, the value of I_s is selected to minimize the total energy consumption of senders and receivers [i.e., $\min(E_{\text{receive-check}} + E_{\text{receiving}} + E_{\text{sending}})$].

In noisy environments, false wakeups (wakeup without receiving any packet) of receivers may occur frequently because transmissions of other wireless devices (such as Wi-Fi, bluetooth, and microwave) also lead to high energy on the channel. According to the previous study [11], the energy consumption of a false wakeup is higher than a normal receive check by $17.3\times$.

In high false wakeup rate scenarios, if a low value of I_s is set, a node will perform receive checks frequently, which leads to a great number of false wakeups within a time period (i.e., T). As a result, a considerable amount of energy is consumed by false wakeups. Adjusting the value of I_s can reduce the number of false wakeups and energy consumption.

We deploy 63 Telosb sensor nodes running with TinyOS-LPL MAC [8] at different locations at our university (e.g., the central hall, Student HUB, and different buildings) to study the behaviors of the existing LPL's false wakeup rates. Fig. 2 presents results obtained from Student HUB [32]. The results show that false wakeup rates at different locations of the same network may be different.

We record the false wakeup rates of sensors at ten selected locations in Student HUB over time. We divide the time of a day into different time frames based on the similarity of obtained results, including the frame 1 (10 P.M.–7 A.M.), 2 (7–10 A.M.), 3 (10 A.M.–12 P.M.), 4 (12–3 P.M.), 5 (3–6 P.M.), 6 (6–8 P.M.), and 7 (8–10 P.M.). Results are reported in Fig. 3, which shows that the false wakeup rates in the same location may vary over time. The

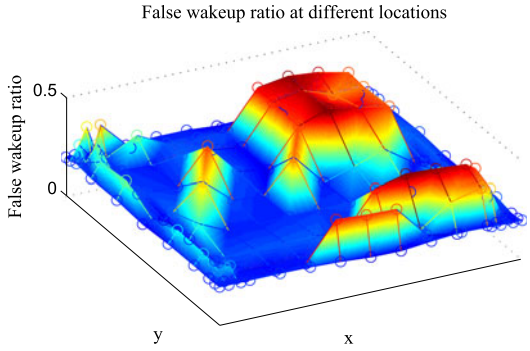

Fig. 2. False wakeup rates at different locations in Student HUB [32] with the width $x = 80$ m and the length $y = 95$ m.

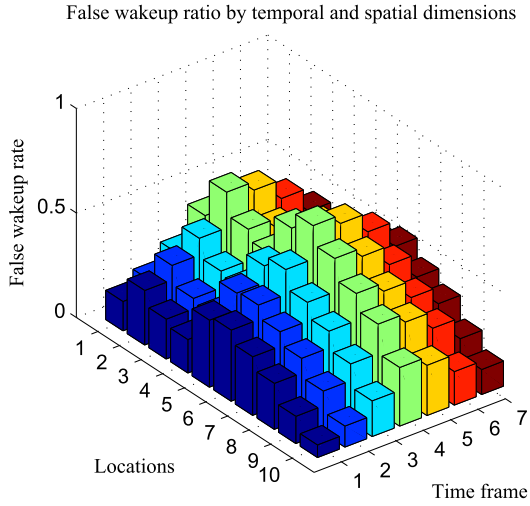

Fig. 3. False wakeup rate in Student HUB at different times.

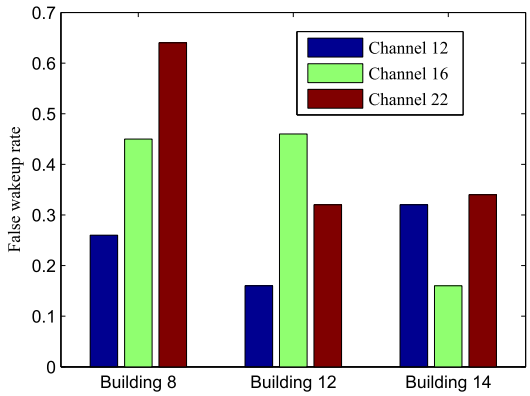

Fig. 4. False wakeup rate at different buildings and channels.

highest false wakeup rates are obtained at the time frames of 3 and 4 when Student HUB witnesses the most crowded students who may use their wireless devices for different applications and use microwave machines in the HUB during their lunch time. The false wakeup rate of sensors at different buildings on different channels is presented in Fig. 4. The figure indicates that the false wakeup rate also varies over different channels.

Due to the dynamic characteristics of false wakeup rate in both temporal and spatial dimensions, I_s should be dynamically adapted to optimize LPL's energy efficiency.

T_w and T_e: A high value of T_w may lead to high energy consumption of a node because the node remains active for a long period to listen for packets, but it potentially reduces its senders' energy cost due to a higher chance for receiver to detect their transmissions, and vice versa. The same reason applies to T_e. The energy consumption in this case depends on how frequent senders send packets, or in other words, the data rate that may vary over time. Therefore, both values T_w and T_e need to be adaptive to optimize the energy efficiency.

IV. Theoretical Framework

To optimize the energy consumption of LPL in dynamic network conditions, the temporal parameters need to be adapted over time. For that reason, a theoretical framework, which models the impact of the LPL temporal parameters incorporating with the network condition parameters (i.e., false wakeup and data rate) to the energy consumption of sensors, is required. In this section, a theoretical framework is established by analyzing and modeling the energy consumption of sensors (i.e., the receiver and its senders) in noisy environments as a function that incorporates false wakeup rate and data rate parameters with the three LPL temporal parameters I_s, T_w, and T_e. The theoretical framework is then used by the parameter optimizer, presented in the next section, to calculate the optimal LPL temporal parameters, which minimize the total energy consumption.

A. System Parameters

False wakeup ratio (R_{fw}): The ratio between the number of false wakeups and the total number of CCA checks in the same period of time.

Traffic rate (R_p): The number of incoming data packets in a unit of time (i.e., 1 s).

Sleep interval (I_s): The sleep period in a cycle.

Active period (E_a): The total wakeup period in a cycle, which depends on the following two parameters.

Periodic wakeup period (T_w): The period a node remains awake after waking up in every cycle if the node does not send or receive any packet.

Extended wakeup period (T_e): The extra period a node extends its wakeup time after receiving a packet.

Cycle length (T_{cycle}): The period between two consecutive sleep times. $T_{\text{cycle}} = I_s + E_a$.

Number of received packets (k): The number of packets a node receives in a cycle during its active period.

B. Energy Consumption Analysis

In this section, we compute the energy consumption of a receiver in a time window T_u, including energy consumed by the radio's start-up phase, CCA checks, false wakeups, and wakeup period for receiving packets. We also compute the energy consumption of senders for sending packets to the receiver. We mainly focus on the time cost of the radio wakeup of nodes, when most of energy is consumed. Based on the results computed, the receiver will optimize the energy consumption of itself and its senders.

1) Receiver's Cost:

Receive checks: In the time window T_u, a receiver performs N_{rc} receive checks on average (i.e., $N_{\text{rc}} = T_u/T_{\text{cycle}}$) in total. Assume the time cost of the radio start-up phase is $T_{\text{radio-on}}$ and of a receive check is T_{rc}. The total time cost for receive checks in T_u is as follows:

$$E_{\text{rc}} = (T_{\text{radio-on}} + T_{\text{rc}})T_u/T_{\text{cycle.}} \tag{1}$$

The result shows that the energy consumption by receive checks is inversely proportional to the cycle length.

False wakeups ($k = 0$): A wakeup is called as a false wakeup when a node wakes up fully, but there is no incoming packet. Assume R_{fw} and T_{fw} are the false wakeup ratio and the time cost for a false wakeup. The average number of false wakeups of a receiver in T_u is $N_{\text{fw}} = R_{\text{fw}}N_{\text{rc}}$. The total time cost for false wakeups is as follows:

$$E_{\text{fw}} = T_{\text{fw}}R_{\text{fw}}T_u/T_{\text{cycle.}} \tag{2}$$

This result also indicates that the energy consumption by false wakeups is inversely proportional to the cycle length. In other words, a node wakes up more frequently for receive checks likely to have more false wakeups.

Receiving packets ($k > 0$): We first compute the expected receiving time cost of a receiver in a cycle. Because a node dynamically extends its wakeup period when it receives a packet, the time cost of a receiver depends on the following parameters: 1) the value of T_w; 2) the value of T_e; and 3) the number of received packets and the interpacket interval between two consecutive packets. Note that not all received packets lead to an extended wakeup period of a receiver. We define the following concepts.

Extending packets: Upon receiving an extending packet, the receiver extends its wakeup period for a positive extended period. As a result, its wakeup period is greater than the case without receiving the packet.

Data packets with preamble transmission: Packets are transmitted with preamble when the receiver still sleeps. Those packets can be received immediately when the receiver wakes up.

Data packets without preamble transmission: Packets are sent at the receiver's wakeup period.

In a general case, the number of received packets and the interpacket interval are random variables. We thus compute the wakeup period of a receiver based on T_w, T_e, probability of *k* extending packets, and the expected interpacket interval. Based on the definition of an extending packet, we calculate the probability of *k* extending packets in two cases.

For $T_w \geq T_e$:

Theorem 1: An extending packet should be received after $t + I_s + (T_w - T_e)$.

Proof: When a receiver wakes up and does not receive any packet, its total wakeup period is $E_a = T_w$ from $t + I_s$ to $t + I_s + T_w$.

If a packet *p* is received at time t_1 before $t + I_s + (T_w - T_e)$ $(t \leq t_1 \leq t + I_s + (T_w - T_e))$, the receiver will extend its wakeup period until at least $t_1 + T_e$. However, $t_1 + T_e < t + I_s + (T_w - T_e) + T_e = t + I_s + T_w$. As a result, receiving the packet *p* does not result in an increase in the wakeup period of the receiver. In other words, *p* is not an extending packet.

If a packet p′ is received at time t_2 after $t + I_s + (T_w - T_e)$, the receiver will extend its wakeup period until at least $t_2 + T_e > t + I_s + (T_w - T_e) + T_e = t + I_s + T_w$. The extended period is equal to $t_2 + T_e - (t + I_s + T_w)$. Therefore, p′ is an extending packet.

Denote the number of received packets in a time period from t to t' by $N_t^{t'}$ and t_i is the arrival time of packet i_{th}. Wakeup period of a receiver is extended if it receives at least one packet

during the period from $t + I_s + (T_w - T_e)$ to $t + I_s + T_w$ ($N_{t+I_s+(T_w-T_e)}^{t+I_s+T_w} > 0$). ■

Theorem 2: The interpacket interval between two consecutive extending packets $P_{(i+1)}$ and P_i should not be greater than T_e.

Proof: After receiving an extending packet P_i at time t_i, the receiver will go to sleep if it does not receive any packet during the period from t_i to $t_i + T_e$.

If a packet P_{k+1} arrives at t_{k+1} with $t_{k+1} - t_k > T_e$, the receiver is unable to receive as it is sleeping, thus P_{k+1} is not an extending packet.

The probability for k extending packets is calculated as follows:

$$P_{T_w \geq T_e}(k) = P(N_{t+I_s+(T_w-T_e)}^{t+I_s+T_w} > 0) \bigwedge_{i=1}^{k-1}(t_{i+1} - t_i \leq T_e) \bigwedge(t_{k+1} - t_k > T_e). \tag{3}$$

We have $P_{T_w \geq T_e}(0) = P(N_{t+I_s+(T_w-T_e)}^{t+I_s+T_w} = 0)$.

We now calculate the expected interpacket interval between two consecutive extending packets

$$\overline{T_{\text{ip}}(T_{\text{ip}}^{\max})} = \int_0^{T_{\text{ip}}^{\max}} TP(T_{\text{ip}} = T | N_0^{T_{\text{ip}}^{\max}} > 0) dT \tag{4}$$

where $T_{\text{ip}}^{\max}$ is the maximum interpacket interval. In this case, $T_{\text{ip}}^{\max} = T_e$. $P(T_{\text{ip}} = T)$ is the probability of the interpacket interval of *T*.

As a result, we can compute the expected total wakeup period $E_{a1}(k)$ of a receiver with *k* extending packets as follows:

$$E_{a1}(k) = \begin{cases} T_w, & \text{if k = 0} \\ T_w + k\overline{T_{\text{ip}}(T_e)}, & \text{otherwise.} \end{cases} \tag{5}$$

For $T_w < T_e$: ■

Theorem 3: Any packet received during the wakeup period of the receiver is an extending packet.

Proof: If the receiver does not receive any packet in a cycle, its wakeup period is T_w from $t + I_s$ to $t + I_s + T_w$. If the receiver receives a packet at time t′ during its wakeup period ($t' \geq t + I_s$), the receiver will set its active period until at least $t' + T_e$. As $t' + T_e \geq t + I_s + T_w$, *p* is an extending packet. ■

In other words, the wakeup period of the receiver is extended if it receives at least one packet T_w ($N_{t+I_s}^{t+I_s+T_w} > 0$).

Theorem 2 is also applied for this case.

We then have the probability of *k* extending packets

$$P_{T_w < T_e}(k) = P(N_{t+I_s}^{t+I_s+T_w} > 0) \bigwedge_{i=1}^{k-1}(t_{i+1} - t_i \leq T_e) \bigwedge(t_{k+1} - t_k > T_e). \tag{6}$$

The expected interpacket interval is also calculated using (4). The expected total wakeup period $E_{a2}(k)$ of the receiver with *k* extending packets in a cycle is calculated as follows:

$$E_{a2}(k) = \begin{cases} T_w, & \text{if k = 0} \\ \overline{T_{\text{ip}}(T_w)} + (k-1)\overline{T_{\text{ip}}(T_e)} + T_e, & \text{otherwise} \end{cases} \tag{7}$$

where $\overline{T_{\text{ip}}(T_w)}$ is the expected period from the time the receiver wakes up to the time of receiving the first packet. In case $N_t^{t+I_s} > 0$, there is data packets with preamble transmission and the first extending packet may be received when the receiver wakes up, thus $T_{\text{ip}}(T_w)$ can be equal to 0.

From (5) and (7), we compute the expected total wakeup period of a receiver as follows:

$$E_a = \begin{cases} \sum_{k=0}^{\infty} E_{a1}(k) P_{T_w \geq T_e}(k), & \text{if } T_w \geq T_e \\ \sum_{k=0}^{\infty} E_{a2}(k) P_{T_w < T_e}(k), & \text{otherwise.} \end{cases} \tag{8}$$

We now have the expected duty cycle length

$$T_{\text{cycle}} = I_s + E_a. \tag{9}$$

From (1), (2), and (8), we calculate the total wakeup period of the receiver in a time window of T_u as follows:

$$E_{\text{receiver}} = (E_{\text{rc}} + E_{\text{fw}} + E_a) T_u / T_{\text{cycle.}} \tag{10}$$

2) Senders' Cost: : The time cost of senders to send packets to the receiver depends on the total number of packets including packets with preamble transmission (N_p) and nonpreamble transmission (N_{non}).

The expected number of packets with preamble transmissions depends on I_s and the traffic rate R_p, and is calculated as follows:

$$N_p = R_p I_s T_u / T_{\text{cycle.}} \tag{11}$$

The expected number of packets without preamble transmission (i.e., N_{non}) depends on the wakeup period of the receiver and probability of *k* received packets. In case of $T_w \geq T_e$, N_{non} consists of received packets in a period between $[t + I_s, t + I_s + (T_w - T_e)]$ and extending packets. In case of $T_w < T_e$, one of the received extending packets may be a packet with preamble transmission if $P(N_t^{t+I_s}) > 0$. Therefore, N_{non} is calculated as follows:

$$N_{\text{non}} = \begin{cases} \left\{ \sum_{x=0}^{\infty} x P(N_{t+I_s}^{t+I_s+(T_w-T_e)} = x) + \sum_{k=0}^{\infty} P_{T_w \geq T_e}(k) k \right\} T_u / T_{\text{cycle}}, & \text{if } T_w \geq T_e \\ \left\{ \sum_{k=0}^{\infty} P_{T_w < T_e}(k) k - P(N_t^{t+I_s} > 0) 1 \right\} T_u / T_{\text{cycle}}, & \text{otherwise.} \end{cases} \tag{12}$$

Assume that sending a packet without preamble transmission costs β s. The expected sending duration of a packet with preamble transmission is $I_s/2$. The total wakeup period for sending packets is then

$$E_{\text{senders}} = N_p I_s / 2 + (N_p + N_{\text{non}})\beta. \tag{13}$$

Expected energy consumption: We denote γ, η, and δ as the energy consumption rates for channel sensing, for listening/receiving, and for sending. The expected energy consumption to receive and send packets is calculated as follows:

$$f(I_s, T_w, T_e) = E = \gamma E_{\text{rc}} + \eta(E_{\text{fw}} + E_a) + \delta E_{\text{senders.}} \tag{14}$$

Our goal is to optimize *E* to achieve the minimum energy consumption of the receiver and senders. We use (14) as the guideline for our protocol design.

3) Illustration to Calculate E: *E* can be easily obtained based on a specific distribution of traffic. For illustration, we assume the traffic follows a Poisson distribution. As intervals between events y have the exponential distribution and considering the memorylessness as well as independence of

interpacket interval $y_i = t_{i+1} - t_i$, we have $f(y) = \lambda e^{-\lambda y}$. Follows the Poisson distribution, we also have the probability $P(N_{t+I_s}^{t+I_s+T_e} > 0) = 1 - e^{-\lambda T_e}$. From above results and (3), we have

$$P_{T_w \geq T_e}(k) = P(N_{t+I_s+(T_w-T_e)}^{t+I_s+T_w} > 0) \left(\prod_{i=1}^{k-1} \int_0^{T_e} f(y_i) dy_i \right) \int_{T_e}^{\infty} f(y_k) dy_k$$
$$= (1 - e^{-\lambda T_e})^k e^{-\lambda T_e} \quad (15)$$

$$P_{T_w < T_e}(k) = P(N_{t+I_s}^{t+I_s+T_w} > 0) \left(\prod_{i=1}^{k-1} \int_0^{T_e} f(y_i) dy_i \right) \int_{T_e}^{\infty} f(y_k) dy_k$$
$$= (1 - e^{-\lambda T_w})(1 - e^{-\lambda T_e})^{k-1} e^{-\lambda T_e}. \quad (16)$$

Similarly, we can calculate other values such as R_p and R_{fw} and finally obtain $E = f(I_s, T_w, T_e)$. We later show how to use (14) to optimize LPL's parameters to minimize E.

C. Energy Consumption Minimization Problem

From the result (14), we formulate the energy consumption minimization problem as follows.

Objective function:

$$\text{minimize } f(I_s, T_w, T_e). \quad (17)$$

Subject to:

$$I_s \geq 0 \quad (18)$$
$$T_w \geq 0 \quad (19)$$
$$T_e \geq 0. \quad (20)$$

We solve the minimization problem by using the extreme value theory to find the optimal values of the LPL temporal parameters (i.e., I_{s0}, T_{w0}, T_{e0}) so that the minimum energy consumption is achieved. We have the gradient vector of f as follows:

$$\overrightarrow{\nabla} f = \left(\frac{\partial f}{\partial I_s}, \frac{\partial f}{\partial T_w}, \frac{\partial f}{\partial T_e} \right) \quad (21)$$

which is a vector of first-order partial derivatives.

Because f achieves the extreme value at (I_{s0}, T_{w0}, T_{e0}), we have $\overrightarrow{\nabla} f(I_{s0}, T_{w0}, T_{e0}) = 0$. As a result, we have

$$\frac{\partial f}{\partial I_s}(I_{s0}, T_{w0}, T_{e0}) = 0 \quad (22)$$
$$\frac{\partial f}{\partial T_w}(I_{s0}, T_{w0}, T_{e0}) = 0 \quad (23)$$
$$\frac{\partial f}{\partial T_e}(I_{s0}, T_{w0}, T_{e0}) = 0. \quad (24)$$

By solving (22)–(24) under the constraints (18)–(20), we find the optimal values of the LPL temporal parameters (I_{s0}, T_{w0}, T_{e0}). We can check whether or not the obtained results lead to the minimum of the function f (i.e., the minimum energy consumption) by using the second derivation test with Hessian matrix (H) based on the extreme value theory

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial I_s^2} & \frac{\partial^2 f}{\partial I_s \partial T_w} & \frac{\partial^2 f}{\partial I_s \partial T_e} \\ \frac{\partial^2 f}{\partial T_w \partial I_s} & \frac{\partial^2 f}{\partial T_w^2} & \frac{\partial^2 f}{\partial T_w \partial T_e} \\ \frac{\partial^2 f}{\partial T_e \partial I_s} & \frac{\partial^2 f}{\partial T_e \partial T_w} & \frac{\partial^2 f}{\partial T_e^2} \end{bmatrix} \quad (25)$$

where the derivatives are evaluated at (I_{s0}, T_{w0}, T_{e0}).

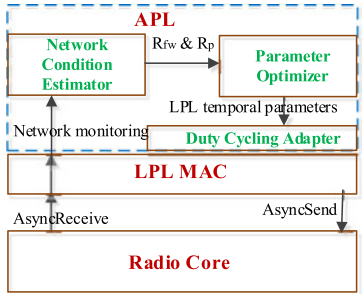

Fig. 5. Implementation of LPL-APL in TinyOS.

V. Implementation of the Adaptive Protocol

Based on the theoretical framework, we implement APL on top of the existing TinyOS-LPL MAC protocol [8] (LPL-APL). The implementation consists of three main components: network condition estimator, parameter optimizer, and duty cycling adapter, as illustrated in Fig. 5. The estimator operates based on the counters, as described below, to measure the traffic rate and the false wakeup rate. If the estimator detects a significant change in one of the two rates, it triggers to call the parameter optimizer. The parameter optimizer computes optimal values of LPL temporal parameters based on the information received from the estimator. If the optimizer finds any changes in the optimal setting of the LPL temporal parameter, it calls the adapter to adjust the duty cycling (i.e., when the false wakeup rate increases, the framework automatically extends the sleep interval to reduce the total energy consumption).

A. False Wakeup Rate Measurement

We use two counters to monitor states of the radio. One counter, named FalseWakeup counter, is used to count the number of false wakeups. After waking up and listening for a timeout without receiving any packet, a node knows that its wakeup is false. Another counter, named CCA counter, is used to count the number of receive checks. The false wakeup rate R_{fw} is calculated as a ratio between the two values of the FalseWakeup counter and the CCA counter.

B. Data Rate Measurement

To measure the traffic rate, we use a counter to count the number of incoming packets N_p in a time window T. We then obtain the estimation of the traffic rate R_p by $R_p = N_p/T$.

C. Preamble Transmission Timeout Selection

To enable a sender to transmit packets to a receiver, the sender has to know its preamble transmission timeout. Preamble transmission timeout determines the maximum preamble length a

node may transmit. There are generally two ways to set the timeout in LPL-APL. The first way is to enable each node to record I_s values of its neighbors and update those values whenever changes incur, by piggybacking them on data messages of the upper layers. In this way, each sender can use exact preamble transmission timeout when it communicates with a neighbor node. The second way is to select a preamble transmission timeout that works for every node. The timeout should be long enough to ensure that during such a preamble transmission period of a sender, its receiver should wake up at least once (i.e., the timeout value should be equal or greater than sleep interval of the receiver). Note that, compared to the preamble transmission timeout, actual preamble transmission periods of a node are normally shorter because preamble transmission of a sender is terminated once the target receiver is awakened. The analysis in our previous study [28], [33] and extensive experimental results [27] show that average preamble transmission period of a sender is only about a half of its receiver's sleep interval.

Using the second method can eliminate unnecessary LPL parameters exchanging among nodes, thus more efficient in term of energy. We implement LPL-APL using the second method by allowing a node to select its own preamble transmission timeout as follows. At the time of network deployment, to enable nodes to communicate with each other, all nodes are configured with the same I_s^{default} value. After operating, each node calculates its own LPL parameters. Assume the first calculated optimal sleep interval of node i is $I_s^{\text{i-init}}$. Theoretically, the maximum value of I_s of a node may double the initial value (i.e., when the false wakeup rate increases from 0% to 100%). In practice, the highest value of I_s witnessed in our experiments is only about $1.53 * I_s^{\text{init}}$ in case of the highest false wakeup rate during experimental periods. In addition, neighbor nodes, which communicate directly with each other, coexist in the same small space, thus having fairly similar network conditions. For those reasons, we implement each node i selects its timeout equal to $I_s^{i-\max} = 2 * I_s^{\text{i-init}}$. In the experimental part, we validate our selection and show that value of the timeout does not affect significantly to actual preamble transmission length as long as the timeout value is great enough to ensure the receiver waking up at least once during preamble transmission period of the sender.

D. Parameter Optimization and Adaptation

In our protocol design, a receiver node adapts its LPL temporal parameters to optimize the energy consumption of itself and its senders. Depending on applications, the calculation of LPL temporal parameters can be different. In constant bit rate applications, T_w and T_e can be predetermined because they mainly depend on the traffic rate and do not impact significantly on the false wakeup of a node. The reason is that those timers start only when nodes already wake up. In this case, optimization is simple by calculating only one parameter of I_s so that $\frac{\partial f}{\partial I_s} = 0$. This calculation can be performed by sensor nodes.

In other cases, solving (22)–(24) may introduce a significant overhead compared to the limited capability of sensors. A complex computation is energy consuming and unnecessary because in practice, a node only needs to adapt its parameters when significant changes happen, in order to ensure the stability of the system.

For this reason, we precompute the optimal values for those parameters under different values of R_{fw} and R_p (i.e., with an interval of 0.05). Each node then stores those values locally.

TABLE I
PARAMETERS

Parameter	Value	Parameter	Value
Data packet length	32 B	$I_s^{\text{TinyOS-LPL}}$	0.5 s
Preamble packet length	6–9 B	$T_w^{\text{TinyOS-LPL}}$	10 ms
T_{fw}	10 ms	T_u	1 s
Time window T	1 s	$T_e^{\text{TinyOS-LPL}}$	100 ms

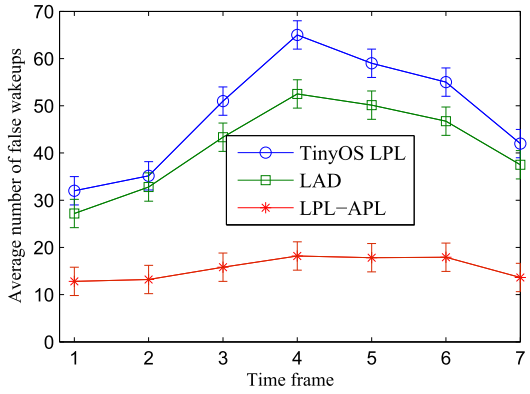

Fig. 6. Average number of false wakeup in 1 min.

When a node obtains new values of R_{fw} and R_p, it then approximates and searches for corresponding optimal values of the LPL temporal parameters. If the new optimal values are found, the adapter then adjusts the LPL temporal parameters to optimize energy consumption.

VI. Performance Evaluation

Table I gives the detailed parameters used in our experiments. We keep the default CCA checks of the TinyOS-LPL up to 400 times. The same number of sensor nodes (63 Telosb sensor nodes) with the experiments in part III is used for this evaluation. We compare performance of APL with current LPL protocol implemented in TinyOS and LAD [29], the most recent improvement version of LPL for energy efficiency. For a fair comparison, other parameters are set to the default values used in TinyOS-LPL and LAD. We use the same seven time frames as described in the preliminary experiments in Section II.

A. Constant Data Rate and Dynamic False Wakeup Rate

In this section, we evaluate the performance of LPL-APL in constant data rate applications. Each node generates a data packet every 30 s, which is reasonable for many WSN applications, and transmits packets toward the sink node (i.e., many-to-one traffic pattern) following a collection tree topology. Fig. 6 shows the average number of false wakeups in 1 min. Results are obtained at different time frames in a day. LPL-APL achieves the lowest number of false wakeups compared to other protocols. The graphs of TinyOs-LPL and LAD jump quickly from the first time frame to the fourth time frame, whereas that of LPL-APL shows only a slight change. The reason is that TinyOS-LPL and LAD are designed to be false wakeup aware. The temporal parameters in both TinyOS-LPL and LAD are fixed regardless the false wakeup rate. When the false wakeup rate increases, the

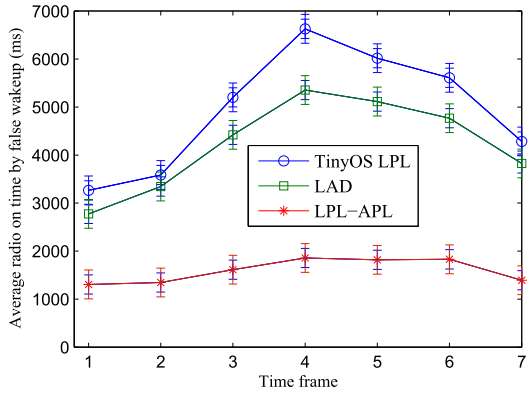

Fig. 7. Average radio on time by false wakeup in 1 min.

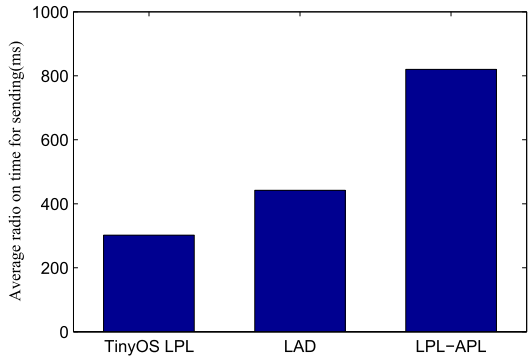

Fig. 8. Average radio on time for sending a packet.

number of false wakeups also increases quickly because the frequency of receive checks of the two protocols does not change compared to cases with a low false wakeup rate. Sensor nodes running LPL-APL are programmed to be aware of false wakeup and adapt their parameters when network condition changes. They control the number of false wakeup in balance with the cost for sending packets of sender nodes so that the total energy consumption is minimized. We translate the number of false wakeups into the average radio on time and present the results in Fig. 7. The figure shows that by adapting LPL temporal parameters based on the false wakeup rate, LPL-APL reduces a significant amount of energy consumed by false wakeups. LAD achieves a better result than TinyOS-LPL because LAD computes its parameters based on a given data rate.

Fig. 8 presents a tradeoff between the energy consumed by false wakeup and the energy consumed for sending a packet of LPL-APL in a high false wakeup rate scenario (i.e., at the fourth time frame). A sender may need to be awake for approximately 800 ms to send a packet in case of a high false wakeup rate because its receiver adjusts parameters to reduce the energy consumption by false wakeups. It is worth noting that a node aims to minimize the total energy consumption by both itself and its senders. In this testbed, the data rate is relatively low, so the total energy consumption for transmitting packets is not a dominant factor compared to others such as receive checks and false wakeups. The average radio-on time per a received packet of a receiver (including the cost for receive checks and false wakeups) is shown in Fig. 9, which indicates an opposite trend compared to that of Fig. 8. The energy consumption of receiver nodes running LPL-APL is much lower

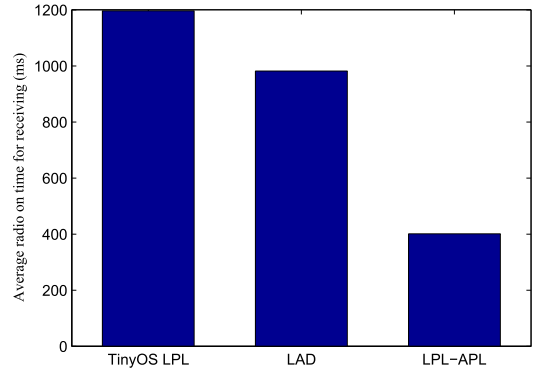

Fig. 9. Average radio on time per received packet.

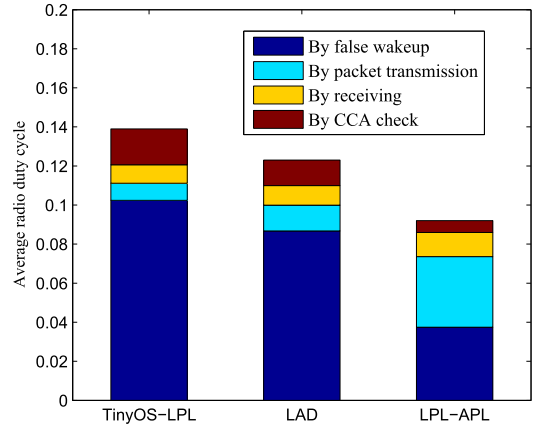

Fig. 10. Average radio duty cycle.

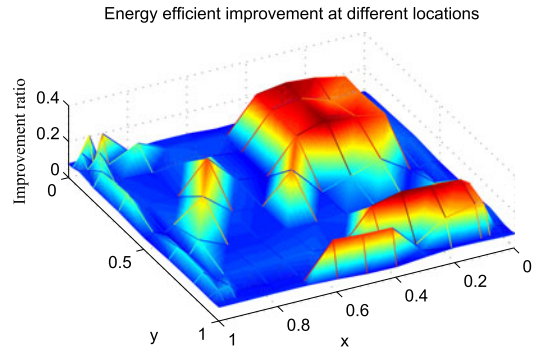

Fig. 11. Average energy efficient improvement of LPL-APL compared to TinyOS-LPL in different locations in Student HUB [32] with the width $x = 80$ m and the length $y = 95$ m.

compared to LAD and TinyOS-LPL. Fig. 10 shows the comparison of the overall energy consumption and the portion of each radio state in each protocol. LPL-APL achieves the lowest energy consumption, whereas TinyOS-LPL is the most expensive one. Although the energy consumption for sending states of nodes running LPL-APL is higher than TinyOS-LPL and LAD, LPL-APL reduces energy consumption caused by false wakeups considerably. TinyOS-LPL keeps the default LPL parameter setting regardless the false wakeup rate. Without the default values, LAD can achieve a low energy cost for sending packets but the energy cost for false wakeups is still very expensive.

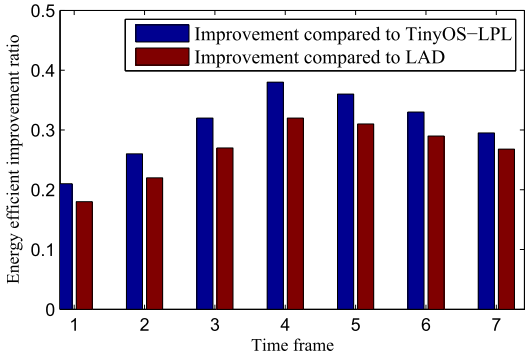

Fig. 12. Average energy efficient improvement of LPL-APL in different time frames.

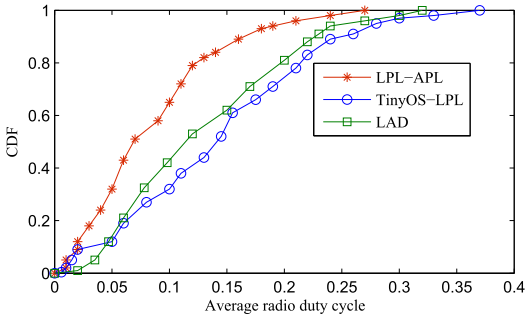

Fig. 13. Cumulative distribution of the average radio duty cycle under a low data rate.

Both Figs. 11 and 12 illustrate the characteristics of LPL-APL. LPL-APL brings more benefits in highly noisy environments. In particular, by comparing Figs. 11 and 2, we can see that the energy efficient improvement of LPL-APL in locations with a high false wakeup rate is higher than that locations with a lower false wakeup rate. Note that TinyOS-LPL with the default parameters may perform well in case of no false wakeups, so LPL-APL has less improvement in case of low false wakeups. Results are averaged from measurements at different times in a day. Fig. 12 presents the energy efficient improvement of LPL-APL compared to both TinyOS-LPL and LAD in the time dimension. The time frames used in this testbed are the same with the experiment in Fig. 3. The highest improvement is achieved at the fourth time frame when the false wakeup rate reaches its peak.

B. Dynamic Data Rate and Dynamic False Wakeup Rates

In this section, we evaluate LPL-APL with different data rates under a dynamic false wakeup rate. We select a relative low data rate of one packet per 60 s and a relative high data rate of one packet per 5 s to compare the performance of LPL-APL in low and high data rate applications.

Fig. 13 presents the cumulative distribution function (CDF) of average radio duty cycle in a low data rate scenario. We can see that LPL-APL outperforms other protocols in term of energy efficiency. In particular, in our protocol, more than 85% of nodes have a duty cycle lower than 15%, whereas that of others is lower than 60%. By dynamically adapting LPL temporal parameters, our protocol saves a significant amount of energy compared to other protocols. LAD performs slightly better than TinyOS-LPL in low data rate applications. Results in a high data rate scenario are shown in Fig. 14. We observe similar improvement of LPL-APL compared to TinyOS-LPL, but lower than LAD. The reason is that LAD also adapts parameters based on data rate. In addition, in a high data rate scenario, LPL-APL needs to consider more in balance between the energy consumption by false wakeups and the energy consumption for sending packets. This is illustrated in Figs. 15 and 16. Fig. 16 indicates that the energy consumption by false wakeups of LPL-APL in a high data rate scenario is even higher than in a low data rate scenario although the number of incoming packets is greater. In a high data rate, nodes running LPL-APL has a trend to wake up more frequently to check for incoming packets so that its energy consumption for sending a packet is reduced. In

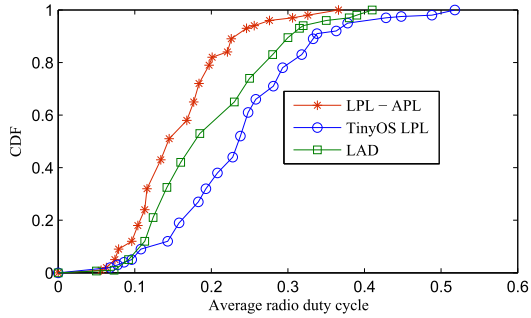

Fig. 14. Cumulative distribution of the average radio duty cycle under a high data rate.

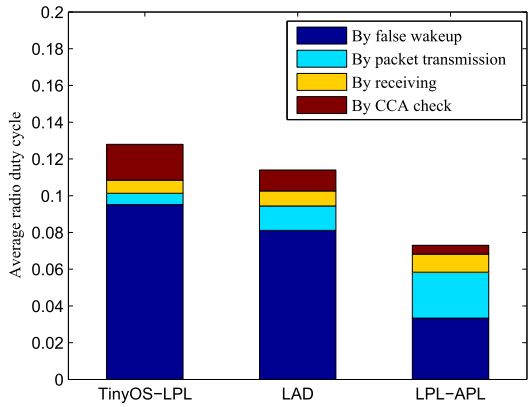

Fig. 15. Average radio duty cycle under a low data rate.

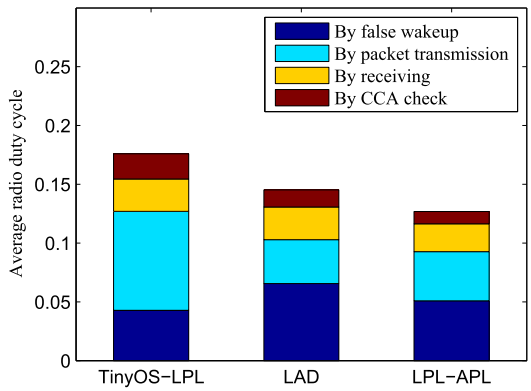

Fig. 16. Average radio duty cycle under a high data rate.

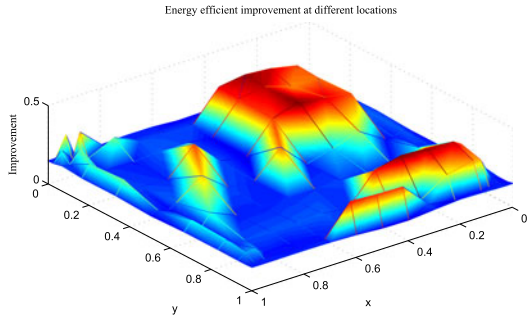

Fig. 17. Energy efficient improvement of LPL-APL compared to TinyOS-LPL under a low data rate.

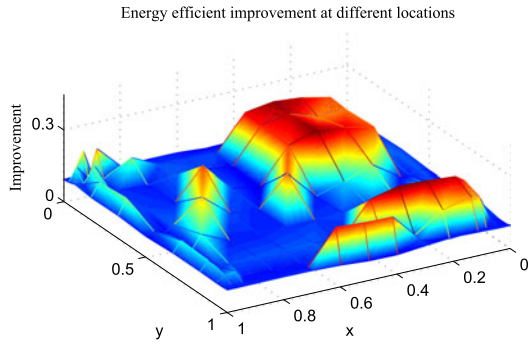

Fig. 18. Energy efficient improvement of LPL-APL compared to TinyOS-LPL under a high data rate.

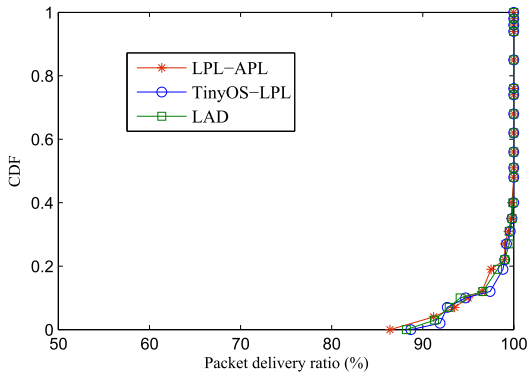

Fig. 19. Cumulative distribution of the PDR under a low data rate.

an opposite site, the energy consumption for false wakeups of TinyOS-LPL is decreased. This is not because the false wakeup rate is decreased, but the number of received packets is higher. The energy consumption for sending packets of TinyOS-LPL jumps quickly and in proportional with the data rate because its cost for sending a packet does not change over time.

Figs. 17 and 18 compare the energy efficient improvement of LPL-APL with TinyOS-LPL in both cases of a low data rate and a high data rate at different locations. The explanation for the results can be found in the discussion for Fig. 11. Overall, LPL-APL achieves higher improvement in a low data rate than a high data rate.

Figs. 19 and 20 show the packet delivery ratio (PDR) under both low data rate and high data rate. Under a low data rate, the three protocols achieve a similar PDR. However, TinyOS-LPL achieves the worst result under a high data rate. Fig. 20 shows

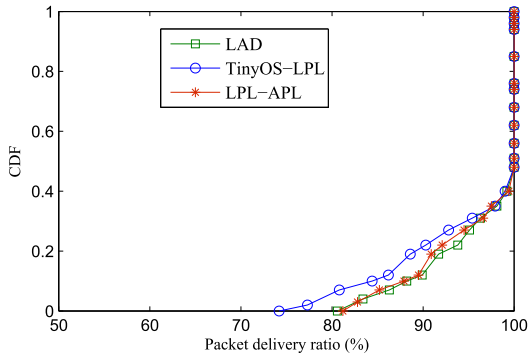

Fig. 20. Cumulative distribution of the PDR under a high data rate.

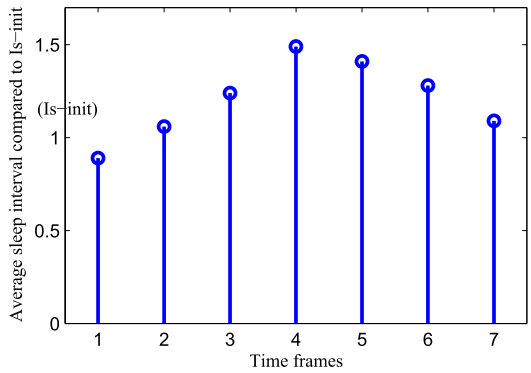

Fig. 21. Average sleep interval at different time frames compared to the initial sleep interval.

that TinyOS-LPL witnesses more than 20% of nodes with PDR lower than 90% and a significant number of nodes (i.e., 8%) with PDR lower than 80%, whereas in LPL-APL and LAD, all nodes achieve PDR higher than 80%. This is due to inefficient channel utilization of TinyOS-LPL in case of high data rate. A node in TinyOS-LPL occupies the channel longer than that in LPL-APL and LAD, which leads to a high collision probability under a high traffic load scenario. The result also highlights the importance of adaptation in term of packet reliability.

C. *Validating Preamble Transmission Timeout Selection*

In this section, we validate correctness of the selection method for preamble transmission timeout of LPL-APL. Note that preamble transmission timeout is just the maximum transmission period that a node may use. Normally, actual preamble transmission of a sender is much shorter than the timeout because a sender stops its preamble transmission when its receiver wakes up.

We compute the ratio between sleep intervals of nodes in the network at different time frames and their initial sleep interval based on recorded data. Statistical results for the ratio are presented in Fig. 21. Note that the network is deployed in the time frame T_2. Average sleep intervals of a node at different time are different. The ratio fluctuates from 0.89 to 1.48 times. The highest change of I_s witnessed in our experiments is only about $1.53 * I_s^{\text{init}}$, at a node at time frame T_4 when false wakeup rate is about 68%. This means that sleep interval changes in practice is far from $2 * I_s^{\text{init}}$. Therefore, selecting the timeout $I_s^{\max} = 2 * I_s^{\text{init}}$ is reasonable.

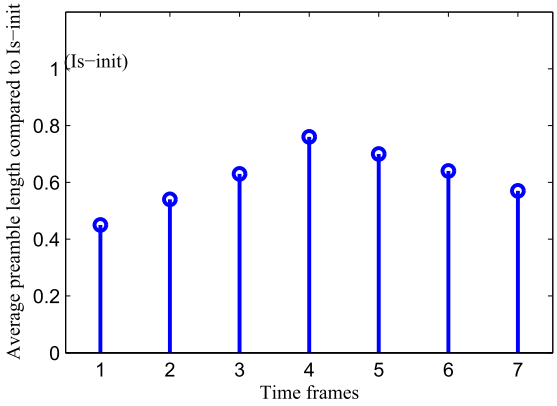

Fig. 22. Average preamble transmission period at different time frames with timeout ($I_s^{\max} = 2 * I_s^{\text{init}}$).

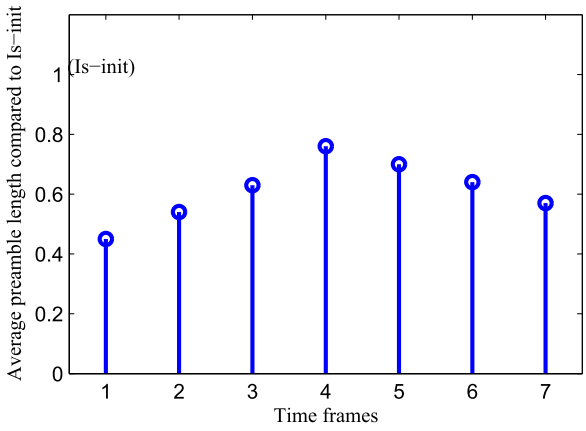

Fig. 23. Average preamble transmission period at different time frames with timeout ($I_s^{\max} = 3 * I_s^{\text{init}}$).

Fig. 22 shows average actual preamble transmission periods of nodes at different time frames. Actual preamble transmission periods of nodes are normally lower than I_s^{init}. Therefore, the selected timeout $I_s^{\max} = 2 * I_s^{\text{init}}$ is great enough to ensure that during such a preamble transmission period of a sender, its receiver should wake up at least once. During experiments, we witness only few preamble transmissions with length up to $1.4 * I_s^{\text{init}}$, possibly due to a receiver failed to detect preamble transmission of a sender. Their impact on the average period is insignificant.

We now increase the timeout value to $I_s^{\max} = 3 * I_s^{\text{init}}$, to see its impact on actual preamble transmission length. Obtained results are presented in Fig. 23. Average preamble transmission periods of nodes in this case remain the same as that with $I_s^{\max} = 2 * I_s^{\text{init}}$. This does mean that the value of the timeout does not affect significantly to actual preamble transmission period as long as the value is great enough. The results validate the timeout selection presented in the previous section.

D. Discussion

Experimental results show that APL contributes a significant improvement in energy efficiency of LPL in noisy environments. APL works more efficient in highly noisy environments compared to less noisy environments. Because sensor nodes are normally resource constraint, a complicated and high computational overhead theoretical framework is not suitable. For this reason, we establish a simplified model to ensure the practicality of our protocol. The implementation of APL fits well to Telosb sensor nodes. In case of dynamic data rate applications, the significant computational overhead for parameter optimization is resolved by precomputing optimal values, storing values, and searching operations. APL updates values of LPL parameters based on the significance of environment changes through setting of two intervals: false wakeup rate interval and data rate interval. This allows APL to ensure the stability of an LPL protocol while the optimal performance can be achieved (the deviation is insignificant).

VII. Conclusion

This paper investigates the energy consumption minimization problem of LPL protocols in noisy WSN environments where the energy consumption by false wakeups of a node can be a dominant factor. We present a theoretical framework and propose a practical solution to address the energy consumption minimization problem. Based on the proposed framework, we design an efficient adaptive protocol to enable a node to optimize its LPL temporal parameters to minimize the total energy consumption of itself and its sender. Experimental results prove that our adaptive protocol outperforms existing LPL protocols in term of energy efficiency.

References

[1] P. Huang, L. Xiao, S. Soltani, M. Mutka, and N. Xi, "The evolution of MAC protocols in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 101–120, Jan.–Mar. 2013.

[2] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst.*, 2004, pp. 95–107. [Online]. Available: http://doi.acm.org/10.1145/1031495.1031508

[3] D. Moss and P. Levis, "BoX-MACs: Exploiting physical and link layer boundaries in lowpower networking," Stanford Univ., Stanford, CA, USA, Tech. Rep. SING-08–00, 2008.

[4] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. 4th Int. Conf. Embedded Netw. Sensor Syst.*, 2006, pp. 307–320. [Online]. Available: http://doi.acm.org/10.1145/1182807.1182838

[5] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: A receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. 6th ACM Conf. Embedded Netw. Sensor Syst.*, 2008, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460414

[6] M. Doudou, D. Djenouri, and N. Badache, "Survey on latency issues of asynchronous MAC protocols in delay-sensitive wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 528–550, Apr.–Jun. 2013.

[7] R. C. Carrano, D. Passos, L. C. S. Magalhaes, and C. V. N. Albuquerque, "Survey and taxonomy of duty cycling mechanisms in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 181–194, Jan.–Mar. 2014.

[8] "TinyOS LPL MAC," 2013. [Online]. Available: http://www.tinyos.net/tinyos-2.x/doc/html/tep105.html

[9] "Contiki: The open source OS for the Internet of Things," 2013. [Online]. Available: http://www.contiki-os.org/

[10] M. Pakparvar, H. Gharibdoust, S. Pollin, and L. Tytgat, "Dynamic channel selection algorithms for coexistence of wireless sensor networks and wireless LANs," in *Proc. IEEE 9th Int. Conf. Wireless Mobile Comput., Netw. Commun.*, Oct. 2013, pp. 33–38.

[11] M. Sha, G. Hackmann, and C. Lu, "Energy-efficient low power listening for wireless sensor networks in noisy environments," in *Proc. 2013 ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, Apr. 2013, pp. 277–288.

[12] G. W. Challen, J. Waterman, and M. Welsh, "IDEA: Integrated distributed energy awareness for wireless sensor networks," in *Proc. 8th Int. Conf. Mobile Syst., Appl, Serv.*, 2010, pp. 35–48. [Online]. Available: http://doi.acm.org/10.1145/1814433.1814439
[13] Z. Li, M. Li, and Y. Liu, "Towards energy-fairness in asynchronous duty-cycling sensor networks," *ACM Trans. Sensor Netw.*, vol. 10, no. 3, pp. 138:–38:26, May 2014. [Online]. Available: http://doi.acm.org/10.1145/2490256
[14] C. Merlin and W. Heinzelman, "Schedule adaptation of low-power-listening protocols for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 5, pp. 672–685, May 2010.
[15] R. Jurdak, P. Baldi, and C. Lopes, "Adaptive low power listening for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 8, pp. 988–1004, Aug. 2007.
[16] C. Merlin and W. Heinzelman, "Duty cycle control for low-power-listening MAC protocols," *IEEE Trans. Mobile Comput.*, vol. 9, no. 11, pp. 1508–1521, Nov. 2010.
[17] G. Anastasi, M. Conti, and M. Di Francesco, "Extending the lifetime of wireless sensor networks through adaptive sleep," *IEEE Trans. Ind. Informat.*, vol. 5, no. 3, pp. 351–365, Aug. 2009.
[18] U. Colesanti, S. Santini, and A. Vitaletti, "DISSense: An adaptive ultralow-power communication protocol for wireless sensor networks," in *Proc. 2011 Int. Conf. Distrib. Comput. Sensor Syst. Workshops*, Jun. 2011, pp. 1–10.
[19] X. Ning and C. G. Cassandras, "Dynamic sleep time control in wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 6, no. 3, pp. 21:1–21:37, Jun. 2010. [Online]. Available: http://doi.acm.org/10.1145/1754414.1754417
[20] X. Chang *et al*., "Accuracy-aware interference modeling and measurement in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 2, pp. 278–291, Feb. 2016.
[21] T. Csoka and J. Polec, "Analysis of additive noise characteristics in indoor wireless sensor networks," in *Proc. IEEE Int. Conf. Comput. Tool*, Sep. 2015, pp. 1–6.
[22] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 3, pp. 493–506, Jun. 2004.
[23] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for tree-based data gathering in sensor networks: Research articles," *Wireless Commun. Mobile Comput.*, vol. 7, no. 7, pp. 863–875, Sep. 2007. [Online]. Available: http://dx.doi.org/10.1002/wcm.v7:7
[24] X. Shi and G. Stromberg, "SyncWUF: An ultra low-power MAC protocol for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 1, pp. 115–125, Jan. 2007.
[25] A. Bachir, D. Barthel, M. Heusse, and A. Duda, "Micro-frame preamble MAC for multihop wireless sensor networks," in *Proc. 2006 IEEE Int. Conf. Commun.*, Jun. 2006, vol. 7, pp. 3365–3370.
[26] A. El-Hoiydi and J.-D. Decotignie, *WiseMAC: An Ultra Low Power MAC Protocol for Multi-hop Wireless Sensor Networks (Ser. Lecture Notes in Computer Science)*, vol. 3121, S. Nikoletseas and J. Rolim, Eds. Berlin, Germany: Springer, 2004. [Online]. Available: http://dx.doi.org/10.1007/978–3-540-27820-7_4
[27] K. Langendoen and A. Meier, "Analyzing MAC protocols for low data-rate applications," *ACM Trans. Sensor Netw.*, vol. 7, no. 1, pp. 10:1–10:34, Aug. 2010. [Online]. Available: http://doi.acm.org/10.1145/1806895.1806905
[28] T. Dinh, Y. Kim, T. Gu, and A. Vasilakos, "L-MAC: A wake-up time self-learning MAC protocol for wireless sensor networks," *Comput. Netw.*, vol. 105, pp. 33–46, 2016.
[29] J. Wang, Z. Cao, X. Mao, and Y. Liu, "Sleep in the Dins: Insomnia therapy for duty-cycled sensor networks," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2014, pp. 1186–1194.
[30] K.-il. Hwang and G. Yi, "Adaptive low-power listening MAC protocol based on transmission rates," *Sci. World J.*, vol. 2014, no. 1, Aug. 2014, Art. no. 473132. [Online]. Available: http://doi:10.1155/2014/473132
[31] R. Kuntz, A. Gallais, and T. Noel, "Auto-adaptive MAC for energy-efficient burst transmissions in wireless sensor networks," in *Proc. 2011 IEEE Wireless Commun. Netw. Conf.*, Mar. 2011, pp. 233–238.
[32] "RMIT city campus map," 2012 [Online]. Available: http://mams.rmit.edu.au/ah5hshrpo0ky.pdf
[33] T. Dinh and T. Gu, "A novel metric for opportunistic routing in heterogenous duty-cycled wireless sensor networks," in *Proc. 23rd IEEE Int. Conf. Netw. Protocols*, Nov. 2015, pp. 1–11.

Thanh Dinh received the Master's degree from Soongsil University, Seoul, South Korea, in 2012, where he is currently working toward the Ph.D. degree at the Graduate School of Electronic and Telecommunication.

He was a Ph.D. Scholar of research with the Royal Melbourne Institute of Technology University. His research interests include the Internet of Things and cloud, mobility, NFV/SFC, and next-generation networks.

Younghan Kim (M'06) received the B.S. degree in electronic engineering from Seoul National University, Seoul, South Korea, in 1984, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1986 and 1990, respectively.

From 1987 to 1994, he was a Senior Research Engineer with the Digicom Institute of Telematics, where he developed ISDN and packet switching systems. Since 1994, he has been a Professor with the School of Electronics, Soongsil University, Seoul. His research interests include wired and wireless networking, QoS, mobile computing, and ubiquitous networking.

Tao Gu (S'03–M'07–SM'14) received the M.Sc. degree from Nanyang Technological University, Singapore, and the Ph.D. degree in computer science from the National University of Singapore, Singapore.

He is an Associate Professor with the School of Computer Science and Information Technology, Royal Melbourne Institute of Technology University, Melbourne, VIC, Australia. His 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 have been published in top journals such as the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING and top conferences such as the IEEE Conference on Computer Communications, IEEE International Conference on Pervasive Computing and Communications, and International Joint Conference on Pervasive and Ubiquitous Computing.

Dr. Gu serves as TPC member for many leading conferences such as the IEEE Conference on Computer Communications, IEEE International Conference on Pervasive Computing and Communications, etc. He has also served as a member of numerous Editorial Boards. He is a member of the ACM.

Athanasios V. Vasilakos (M'00–SM'11) is currently a Professor with the Lulea University of Technology, Lulea, Sweden. He has authored or coauthored more than 200 technical papers in major international journals and conferences.

Prof. Vasilakos was the General Chair and a TPC member for many international conferences. He has been an Editor or/and Guest Editor for many technical journals such as the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, the IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY, the IEEE TRANSACTIONS ON COMPUTERS, *ACM Transactions on Autonomous and Adaptive Systems*, the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, and *IEEE Communications Magazine*. He is the General Chair of the Council of Computing of the European Alliances for Innovation.