Ad Hoc Networks 73 (2018) 65–79

Contents lists available at ScienceDirect

Ad Hoc Networks

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

Flexible real-time transmission scheduling for wireless networks with non-deterministic workloads

Arda Gumusalan*, Robert Simon, Hakan Aydin

Department of Computer Science, George Mason University Fairfax, VA 22030, Virginia

ARTICLE INFO

Article history:
Received 10 October 2017
Accepted 3 February 2018
Available online 7 February 2018

Keywords:
Low power listening
Industrial control networks
Time division multiple access

ABSTRACT

Wireless Sensor Networks (WSNs) are increasingly used in industrial applications such as the Internet-of-Things, Smart City technologies and critical infrastructure monitoring. Industrial WSNs often operate in a cluster or star configuration. To ensure real-time and predictable performance, link access is typically managed using time-slotted superframe methods. These methods generally use static and potentially inefficient slot assignments. In this paper, we propose to dynamically readjust time slot lengths as a technique to minimize overall energy consumption. Our approach combines real-time performance guarantees with energy conservation methods through a set of dynamic modulation based adaptive packet transmission scheduling algorithms that are designed to *reclaim* unused slot times. To support our reclaiming method in a wireless environment we introduce a novel low-power listening technique called *reverse-low-power listening* (RLPL) as part of an overall Hybrid Low-Power Listening (HLPL) protocol. We evaluate our algorithms using Castalia simulator against an oracle-based approach, and show that our dynamic slot reclaiming approach, coupled with HLPL, can introduce substantial power savings without sacrificing real-time support which may be a new approach towards improving industrial wireless standards.

1. Introduction

Industrial network automation systems were traditionally installed with wires connecting communicating devices. Potential drawbacks to purely wired systems are higher costs for cables and maintenance and inflexibility in terms of deploying new nodes or reconfiguring existing systems. As a result industrial automation and control systems are increasingly being supported by wireless networks [1]. Wireless industrial systems are now appearing in application domains such as manufacturing, electrical generation, and chemical refining [2], along with Smart City and environmental monitoring applications [3]. Currently deployed industrial wireless protocols include IEEE 802.15.4e, WIA-PA, WirelessHART and ISA100.11a [4].

Low-power real-time wireless protocols typically work by organizing nodes in cluster or star topologies, and sometimes in multi-hop topologies. Variations of time division multiple access (TDMA) based scheduling for link access is the most widely used method to provide real-time guarantees on WSN [5]. TDMA systems generally have a coordinator that is in charge of distributing time slots to the nodes. Nodes in the system therefore share a logical superframe that is divided into timeslots. Each node has a pre-assigned time slot where it is allowed to transmit so that collisions are prevented. The work presented in [5] concludes that TDMA improves the performance of basic CSMA/CA protocols. Most current standards either use a fixed size or varying size but pre-computed slot lengths for their superframes. This may lead to an efficient or inflexible use of resources in the form of unused timeslots, especially if the workload is not fully predictable.

To address the above issues we propose to dynamically readjust time slot lengths in the superframe as a method to reduce overall energy cost and provide tight real-time guarantees. As noted, many existing protocols assume that the workload is fully deterministic and known in advance, which is not the case for many newer applications that can be supported by real-time wireless protocols [6]. An intuitive question then emerges — *is it possible to achieve both real-time performance and energy savings in the face of uncertain workloads?*

This paper aims to answer the above question through the design and analysis of adaptive, superframe based techniques designed to maintain real-time performance guarantees while minimizing energy consumption. In order to accomplish this goal, we adopted a well-known and widely-studied technique called *Dynamic Modulation Scaling (DMS)* [7,8]. DMS is a technique that exploits the trade-off between latency and energy consumption at a given modulation level. Higher modulation levels consume more

* Corresponding author.
E-mail addresses: agumusal@gmu.edu (A. Gumusalan), simon@gmu.edu (R. Simon), aydin@gmu.edu (H. Aydin).

https://doi.org/10.1016/j.adhoc.2018.02.002
1570-8705/

energy to transmit and receive the data but the transmission takes less time. It has been showed that DMS technique leads to reduced energy consumptions [8]. The basic idea in our approach is to assign nodes time slots in order to meet their communication transmission deadlines, but allow them to *proactively* wake up and determine if other nodes have transmitted all of their packets and no longer require some portion of their time slots. If this is the case, a node can begin packet transmission before its scheduled time, and conserve energy by transmitting at reduced modulation levels. In order to accomplish this, we have designed a new low-power listening protocol called *Hybrid Low-Power Listening*. The algorithmic and protocol challenge is to schedule packet transmissions in a manner that reduces energy while maintaining real-time performance, as compared to the traditional static TDMA approach.

Our work makes the following contributions. We first build a basic real-time superframe model, and then use DMS to opportunistically save energy. DMS, also known as *Adaptive Modulation*, is commonly used to increase throughput in hostile and unpredictable wireless communication settings [9]. For instance, in tactical military environments one current mobile handheld standard is called *JTRS*. Mobile handheld radios such as the Harris AN/PRC-15 implement adaptive modulation within the JTRS standard. For Low-Power and Lossy Networks, the TI CC1200 supports 2-FSK, 2-GFSK, 4-FSK, 4-GFSK, MSK, OOK modulations [10] whereas CC2500 supports 2-FSK, 2-GFSK, MSK and OOK [11]. The application of DMS within WSNs has been studied in [12–16].

We formulate the joint real-time and energy minimization as an optimization problem under the assumption of a workload that can only be known probabilistically. Since solving the optimization problem may be computationally complex, we then propose a set of polynomial-time algorithms to address the joint real-time energy optimization problem. To avoid excessive energy expenditures during the times nodes proactively wake up to see if they can prematurely transmit, we propose and analyze a novel Hybrid Low-Power Listening (HLPL) protocol. HLPL incorporates a new technique called *reverse-low-power listening* (RLPL). RLPL is a twist on traditional low-power listening (LPL) protocols, which are well-known methods used in low data rate duty cycling wireless sensor networks [17]. LPL protocols yield significant energy savings. To our best knowledge this is the first usage of a hybrid LPL technique in a joint TDMA-based real-time energy savings protocol, as well as the first one that combines the beacon-enabled superframe concept with low power listening.

Using HLPL, we evaluated our optimal and heuristic algorithms against an oracle-based approach, which has perfect workload knowledge, under a number of workload and deadline constraints. Our detailed evaluation under 802.15.4 IEEE standards shows that the hybrid HLPL approach, coupled with DMS, can achieve significant energy savings while maintaining real-time performance, as opposed to the traditional TDMA method.

2. Related work

A beacon-enabled superframe technique is presented as an amendment to IEEE 802.15.4 standards and is included in 802.15.4e [18]. Aimed at supporting real-time industrial systems, 802.15.4e provides real-time guarantees for wireless sensor networks (WSNs). This standard defines the contention-access-period, contention-free-period and guaranteed-time-slot formats. This basic approach is incorporated in industrial standards such as WirelessHART, ISA 100.11a and WIA-PA [4,19]. Our work is fully compatible with these standards.

WirelessHART made TDMA based scheduling an industrial standard using a formulation similar to the approach presented in [18]. It defines a superframe as the collection of fixed size time slots controlled by the network manager [19]. In order to avoid interference, it uses a frequency hopping spread spectrum mechanism across the 16 channels of 2.4 GHz ISM band [20]. ISA100.11a is another industrial standard for wireless networks. It also has a superframe concept similar to that in WirelessHART [4]. WIA-PA is a widely adopted system arhitecture and communication protocol standard for wireless networks. WIA-PA uses the beacon enabled superframe design as introduced in 802.15.4e [4]. As discussed in Section 3, we use a generic beacon-enabled superframe model that can be applied to any of these industrial standards. These standards emphasize supporting real-time performance in wireless networks. Our work enhances these standards by adopting dynamic time slot allocation and on-the-fly adjustment for the generic superframe structure. Moreover, we introduce a novel protocol called Hybrid Low-Power Listening (HLPL) as an efficient technique to eliminate the impact of neighborhood for superframe structures by combining the two seemingly contradicting ideas such as superframe and low-power listening.

Our work incorporates TDMA, CSMA, and Low Power Listening (LPL) MAC layer protocols; hence, it can be categorized as a hybrid MAC layer protocol. Hybrid MAC-layer approaches have been studied for a number of years [21]. Some well-known examples are Z-MAC, HyMAC, H-MAC, ER-MAC, and Queue-MAC [22–26]. Z-MAC [22] uses a randomized two-hop setup mechanism to avoid collisions where each node is assigned a time slot. When there is no transmission after a predefined time interval, the nodes can start communicating using CSMA for the duration of the time slot. A hybrid bandwidth-aware mechanism is presented in [27]. The work in [28] used a Markov Decision Process in a hybrid solution to resolve congestion issues and minimize energy consumption, while others considered approaches for reducing queue length [29]. HyMAC combined TDMA and FDMA where a base station assigns time slots and frequencies [23]. H-MAC is another hybrid MAC protocol that combines CSMA with the Aloha protocol [24]. ER-MAC [25], on the other hand, reduces the energy consumption of Z-MAC by allowing CSMA in only *emergency* situations. Queue-MAC [26] also combines TDMA with CSMA in conjunction with variable slot length. In Queue-MAC, each node transmits its load to its parent which in return adjusts the slot distribution in advance. One possible weakness in this protocol is that it assumes the load of the nodes are known prior to their transmission, which is not practical in many real-life scenarios [6].

A more recent hybrid MAC protocol called MMSMAC is proposed in [30]. This protocol operates in synchronous, asynchronous, and hybrid modes. MMSMAC groups the nodes into clusters according to their per-hop distances to the cluster head. In synchronous mode, the nodes are grouped into odd or even based on their cluster numbers and work in periodic active and sleep cycles. Only the nodes in active state are allowed to receive or transmit data (one node can transmit during an active state per cluster). This mode reduces energy consumption but increases delay. In asynchronous mode, the nodes compete for the channel which reduces delay but increases energy consumption. In hybrid mode, the nodes are set up just as synchronous mode but sensor nodes of the active cluster follow the asynchronous operation mode. The hybrid mode's performance is between those of asynchronous and synchronous modes. Another recent hybrid MAC protocol is TAH-MAC [31] which combines CSMA/CA,TDMA, and FDMA. This protocol uses CSMA/CA for lower traffic levels and TDMA/FDMA for high traffic and only TDMA for medium traffic. An adaptive TDMA scheduling for multi-cluster networks is proposed in [32]. This system divides time slots into three categories — IntraSend, InterComm, and IntraRecv — and requires each node to know its interference information and workload. The adjustment of time-slots are done accordingly and are static during the superframe interval. Lenka et al. [33] also proposed a distributed slot scheduling algorithm for hybrid CSMA-TDMA MAC layer. In this proto-

col, each node randomly selects a slot from the available slots list then broadcasts its request to acquire it. If all neighboring nodes agree, the slot is assigned. CSMA comes into play during slot re-allotment process where the system tries to reduce the number of allocated time slots. This re-allotment process may cause non-slot-owner nodes to collide and hence, CSMA is used by non-owners for channel assessment.

Our work differs from the above through our introduction of the Hybrid Low-Power Listening (HLPL) concept that allows efficient on-the-fly slot adjustments, even in the presence of interference where the nodes may overhear each others' communication, and our use of Dynamic Modulation Scaling (DMS) to save energy while maintaining transmission deadlines.

Low-power listening (LPL) is a commonly used MAC-layer protocol that reduces the energy consumption caused by idle listening to the channel for an activity. In LPL, nodes periodically wake up to detect the activities in the channel. LPL techniques are generally categorized as either *sender-initiated, receiver-initiated*, or *hybrid*. Another classification divides LPL protocols into *synchronous* or *asynchronous* solutions. Our proposed HLPL protocol is a sender-initiated, asynchronous LPL and includes a new technique to address high false-alert rates caused by overhearing observed traffic in the traditional LPL protocol frameworks [34]. The most common current approach in these scenarios is to use either TDMA to give QoS guarantees or LPL otherwise [35]. To our best knowledge, our work is the first to show that these can be combined in order to give QoS guarantees such as real-time performance and energy saving.

In our system DMS is simply a control knob that can be replaced with any other energy saving mechanism but it is an important concept for energy minimization and worth examining in greater detail. One of the earliest papers that applied DMS to real-time traffic is [36]. That work developed the concept of adjusting modulation scaling on-the-fly for general real-time purposes. However, WSNs are not the main focus of the paper. The authors in [37] studied the application of DMS on data gathering scheduling of wireless sensors in a real-time scenario. They have shown that DMS can achieve up to 90% energy savings. However, they have assumed the same constant packet workload for each node in the network. Our work differs from the above by considering a *probabilistic* workload and applying DMS to Time Division Multiple Access (TDMA) based scheduling. We believe a non-deterministic workload is more realistic for many applications of wireless sensors and is worth investigating. The work in IGCC'14 by Bandari et al. [38] considered joint DVS/DMS for a single wireless node with probabilistic workload, and suggested static speed scheduling solution for both DVS and DMS. In our work, we consider applying DMS to the task set of every node in the network as a whole where they all share the same deadline. Although the work by Bandari et al. [38] proves the benefits of DMS and DVS, it does not consider dynamic time slot readjustment, which is the main focus of this paper. This case creates problems such as the overhead of low-power listening and interference. In our work, we investigate this dimension in depth and evaluate the effects of low-power listening and neighborhood problem. We also propose a new protocol to overcome this problem.

3. System architecture

This Section describes our targeted system architecture, requirements and device model.

3.1. Application topologies and requirements

Our work focuses on nodes that form single-hop communication clusters. Each node is assumed to periodically generate some number of packets that it must transmit by a specific deadline. The actual number of packets changes over time, and is only known probabilistically. In order to meet these requirements we use a generic, beacon-enabled superframe architecture for real-time communication. We assume every node participating in the cluster can directly communicate with the coordinator. We also assume that the coordinator or clusterhead is not power-limited.

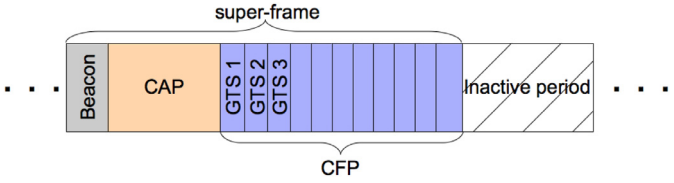


Fig. 1. The lay-out of superframe.

This model is shown in Fig. 1. As seen, at the beginning of each superframe the beacon is transmitted by the coordinator. The beacon contains management information such as TDMA slot assignments, and is received by all the nodes in the cluster. This is followed by a contention access period (CAP), allowing each node to talk to the coordinator via CSMA. During this phase a node may send future workload information, or may ask to join or leave the cluster. The contention free period (CFP) starts right after the CAP. The CFP consists of a series of Guaranteed Time Slots (GTSs), which are assigned to specific nodes. In order to provide real-time communication guarantees, each node is assigned a number of GTSs equal to its worst-case traffic workload.

The coordinator manages GTS assignments to avoid collisions and provide real-time guarantees. The CAP could followed by an inactive period during which the nodes can sleep. We assume that the coordinator manages the duty cycle and, without loss of generality, do not consider its impact in our work. A traditional approach is to have a node sleep during the GTSs assigned to other nodes. In this paper, we introduce a Hybrid Low-Power Listening protocol that allows nodes to *proactively* remain awake during time slots assigned to other nodes to attempt to opportunistically transmit their packets at reduced energy levels.

3.2. Communication

We assume that each node is equipped with a DMS-enabled radio capable of dynamically adjusting the modulation levels. We adopt the basic energy model presented in [7]. Specifically, the radio power consumption is divided into two parts. The first is *transmission power*, denoted as p_s, and the second is *electronic circuitry power*, denoted as p_e. These values can be expressed as $p_s = C_s \times \phi(b) \times R_s$ and $p_e = C_e \times R_s$, respectively. Here, R_s is the number of symbols transmitted per second and b is the modulation size. The values C_e and C_s are radio-specific; but C_s can be affected by the current environmental conditions, such as atmospheric noise, transmitter-receiver distance and temperature. In practice C_e and C_s can both be approximated as constants. Finally $\phi(b)$ is the convex scaling function of the modulation used, depending on the modulation scheme. For QAM, $\phi(b)$ function is $2^b - 1$ for even modulation levels and a close approximation for odd levels [7], which shows the exponential increase in power consumption in terms of the modulation level (p_e is assumed to be constant). The transmission time, on the other hand, is $\frac{1}{b \times R_s}$ which decreases linearly in terms of modulation level. As a result, the trade-off involved in DMS becomes an exponential increase (decrease) of transmission power compared to a linear decrease (increase) of transmission time for QAM [7].

Our HLPL protocol uses two schemes commonly used in asynchronous duty-cycled low-power MAC protocols, namely low-

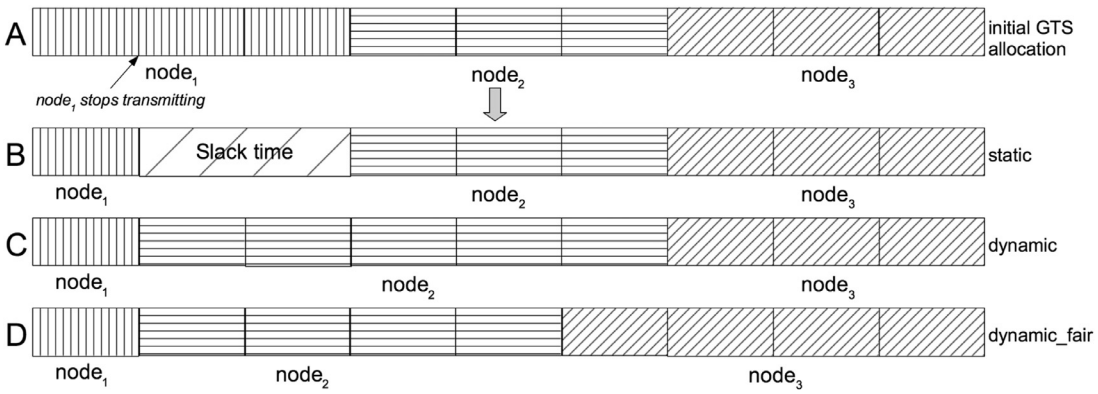


Fig. 2. Illustration of static and dynamic slack reallocation.

power listening (LPL) and embedding information in short preamble (physical layer) packets. A typical LPL protocol such as B-MAC [17] requires a sender to transmit a long preamble. Receivers wake up, sense the preamble, and stay awake to receive data. The duration of listening and sleeping schedules can be adjusted to, for example, maximize energy savings, maximize throughput, or minimize delay. Protocols such as X-MAC [39] extend this idea by using shorter preambles with embedded address information of the target node. The advantage of using preamble addressing is a reduction in the number of bits that need to be transmitted and a flexible, user configurable information that follows the preamble. X-MAC also showed embedding address information into preamble avoids the overhearing problem and saves energy on the non-target devices.

4. Hybrid low-power listening (HLPL)

The advantage of DMS is that the nodes can minimize overall energy consumption by using lower modulation levels. The drawback is that lower modulation levels require longer periods of time to transmit the same number of bits. In our environment this means that more GTSs are required. We assume that the nodes can bound their worst-case workload in terms of the number of packets to send, but the actual distribution is only known probabilistically. This means that the nodes may send fewer packets then their worst case estimate. Hence, a node might not use all of the GTSs assigned to it. It is therefore desirable to devise *dynamic* algorithms capable of assigning these unused slots to other nodes. Further, it is possible to use DMS techniques to extend transmission over unused slots, in order to reduce their energy expenditures, as long as the deadlines are maintained. We call the extra time available from unused slots “slack time”.

Fig. 2 shows how different algorithms may behave when slack time is available. We can generalize these slack reallocation algorithms as static and dynamic. Assume the initial GTS assignments are shown in the superframe labeled *A*. In this example $node_1$ has been assigned 3 packet-length GTS but it only transmits a single packet. Superframe *B* shows what happens under the traditional static approach. The GTS assignments for $node_2$ and $node_3$ do not change, and the available slack time remains un-utilized. The superframe *C* shows a possible dynamic approach where these slots are reallocated to $node_2$, which can lower its modulation level but still meet the deadline. Another possible dynamic approach is dynamic_fair, shown in superframe *D*. In this case the slack could be allocated among $node_2$ and $node_3$ equally. Detailed descriptions of these algorithms along with methods for determining modulation levels for the new GTS distribution are provided in Section 7.

For dynamic algorithms to succeed, the nodes need to be aware when the currently scheduled node prematurely finishes transmission. Our approach works by having the coordinator broadcast a relatively short **preamble** that contains the address of the next node to transmit and the modulation levels that the node will use. Nodes hear this preamble by using low-power listening. The selected node may in fact be granted permission to transmit early using the available slack time.

Fig. 3 shows the parameters of the listen-sleep cycle from the perspective of a single node. Here δ shows the *wait-duration* before the node starts its low-power listening (LPL) cycle. The node is entirely asleep during the period δ. Initial intuition is to set δ value to zero which means the nodes will start performing LPL as soon as the CAP period ends to ensure no preamble will be missed. As we will show in Section 7, performance improvements can be achieved with a careful choice of a non-zero δ value. However, this is possible only in the cases where we have a priori information about the packet workloads of the nodes. For the LPL phase, we use parameters α and γ, referred to as the *sleep-duration* and *listening-duration*, respectively. γ is the time during which the node is listening to the medium whereas α is the time during which the node is in the sleep mode.

The length of the coordinator’s preamble has to be greater than the LPL period of $\alpha+\gamma$. $L_{preamble} \geq \alpha+\gamma$ guarantees that the receiver will hear a portion of the preamble. However, it does not guarantee that the receiver will listen to the preamble as a whole.

Fig. 4 shows the possible intersections of the LPL period and preamble. Among these three possibilities, the first one is preferable, since the receiver gets the entire preamble. In the second and third cases, the receiver will hear the preamble; however, it will not know who the preamble is addressed to. As a result, the receiver will need to keep listening even after the preamble message is over, in order to learn the address of the preamble. This adds to the power consumption of the receiver. The coordinator needs to make sure that after sending the preamble message, the intended node starts transmitting. If not, the coordinator needs to re-send the preamble. We evaluate the effect of these parameters in Section 7.

An additional problem exists in that the absence of transmission activity being detected by a node does *not* necessarily mean that nodes are not transmitting in the cluster. Two nodes may be entirely out of each others’ radio range. Another possibility is that a node may be in another node’s interference range, but not its transmission range. This means that a node can hear another node transmit but cannot decode the transmission. Also, when a node is in the transmission range of another node, it overhears the communication. However, nodes are only interested in transmissions from the coordinator. Listening to the other nodes in addition to the coordinator increases the energy consumption. For our scheme, the practical impact is that a node may not be able to hear another node that is in the process of sending a packet to the coordinator,

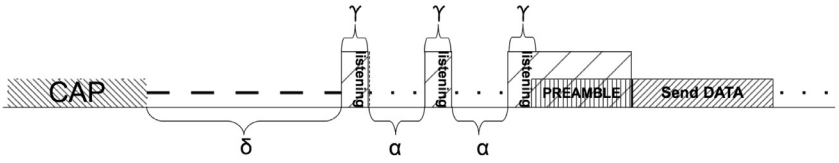


Fig. 3. Parameters of the sleep-listen cycle.

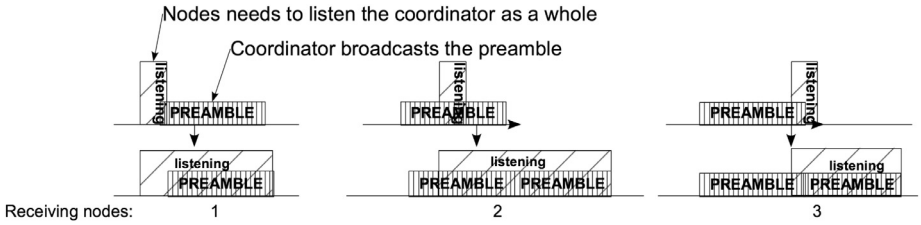


Fig. 4. Possible intersection of listening period and preamble.

and therefore cannot tell if the slot is used or idle. Further, it may overhear the unintended communications with additional energy cost. We refer to these issues as the *neighborhood* problem.

In order to overcome the *neighborhood* problem, we combine preamble addressing with our newly defined hybrid-low-power listening (HLPL) protocol. As described earlier HLPL is a combination of traditional low-power listening and a new scheme called *reverse-low-power listening* (RLPL). On the receiver side, the node needs to decide which LPL mode it needs to be in. In HLPL, if a node receives a preamble and learns that it is not scheduled next **and** it senses any transmission during its *first* wake-up after this preamble, the node goes into the RLPL (described below) stage. For non-zero wait-duration values, when the node wakes up, it listens to the channel for a preamble. If it senses any transmission and this transmission is not a preamble then the node goes into the RLPL stage. However, if this transmission is a preamble that is not addressed to itself and if it does not hear any transmission during its first wake-up after the preamble, it goes into the traditional LPL stage.

For zero wait-duration values the nodes (except for the first scheduled node) start with traditional LPL. During listening phases if they do not hear any transmission, they stay in the tradition LPL stage. However, when a node senses a transmission after a preamble, it goes into the RLPL stage. The logic behind this process is the fact that hearing a transmission during the first wake-up right after the "false" preamble indicates that there is an interference since the sleep-duration is smaller than the time needed for a single packet. If the node does not hear any transmission after a preamble addressed to another node means there is no interference. Meanwhile, the coordinator waits for sleep-duration amount of time before it broadcasts the preamble, unlike in traditional LPL, where a sender broadcasts the preamble as soon as the current node stopped transmitting. Waiting for *sleep-duration* amount of time ensures that all the nodes that are in RLPL mode are currently in the *wait-for-preamble* stage. Fig. 5 shows the flow chart for HLPL.

RLPL differs from traditional LPL in its conditions to transition between listening and sleeping stages. In RLPL when the node wakes-up and hears a transmission, it goes back to sleep. However, if it does not hear any transmission, it starts listening, which is different than the traditional low-power listening case. In RLPL, this listening phase is called *wait-for-preamble* stage. As explained, when a node stops transmitting, the coordinator waits for *sleep-duration* amount of time before it broadcasts the preamble. Hence, *wait-for-preamble* stage can last at least for the sleep-duration time. *Wait-for-preamble* guarantees that when the coordinator broadcasts the preamble, the nodes will be listening to the channel.

The core idea of HLPL is to save energy when there is *constant* traffic in the network. In the absence of this, HLPL behaves very similar to traditional LPL. Fig. 6 aims to clarify the difference between traditional LPL and RLPL during a constant traffic. Under traditional LPL, the node wakes up periodically and tries to sense a transmission. Then it stays awake long enough to conclude that the transmission is not from the coordinator. On the other hand, under RLPL, the node first listens to the channel, realizes that it is not a preamble, and goes to the RLPL stage. With RLPL, a node still periodically wakes up but stays awake enough to detect that there is *some* transmission. If so, the node goes back to the sleep mode. Otherwise, the RLPL stage concludes, and the node transitions to the wait-for-preamble stage. It stays in that stage until it hears a transmission.

5. Joint deadline-energy optimization problem

Based upon the number of nodes, the real-time constraints, and the actual workload, the question remains how to set the modulation levels to achieve all deadlines and conserve energy. We now show how to formulate this question as an optimization problem.

Earlier research in DMS has shown that there exists a *constant* optimal modulation level that minimizes the energy consumption while meeting all deadlines under *deterministic* workloads [7]. However, the work in [38] observed that under *probabilistic* workloads, this is not the case. Instead, the optimal solution to minimize the *expected* energy consumption consists in transmitting the first packets at low speed (modulation), and increasing the speed gradually for the subsequent packets when the deadlien approaches. This is based on the observation that in the more likely scenarios where the actual workload deviates from the worst-case, low modulation levels are sufficient to meet the deadline while saving significant energy. However, as more packets are transmitted, the modulation level is gradually increased to meet the deadline. The framework to find the optimal modulation levels given a deadline and probabilistic workload profile is called *speed scheduling* in [38] and we also adopt this approach.

In our target applications each node has a varying communication workload determined by a known probabilistic distribution. The $node_i$ can have from 1 to m_i packets to transmit in a given superframe. $p_i(k)$ represents the probability distribution function of $node_i$'s workload. Specifically, $p_i(k)$ denotes the probability that $node_i$ will transmit exactly k packets during a superframe (Table 1).

The energy needed to transmit a single packet, e_{packet}, is the product of time to send a single bit (t_{bit}), the length L of the maximum transmission unit (in bits), and the total power ($p_s + p_e$). Moreover, $t_{bit} = \frac{1}{b.R_s}$ where b indicates the modulation level and

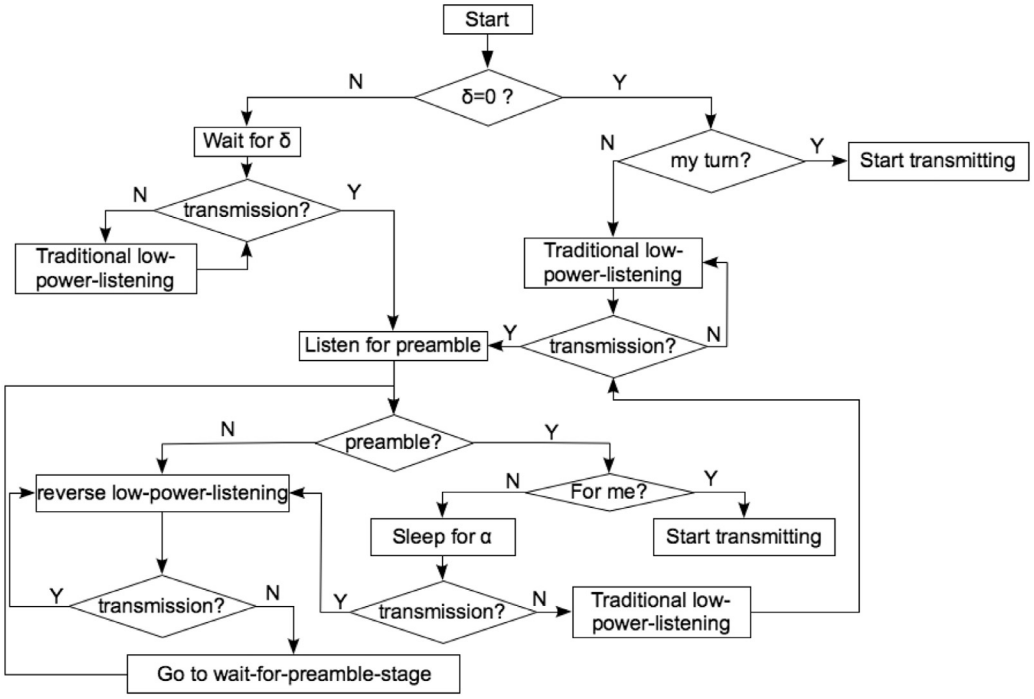


Fig. 5. The steps of HLPL.

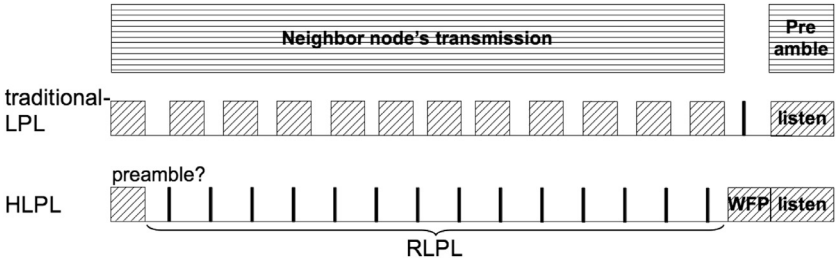


Fig. 6. Comparison of traditional LPL and RLPL.

Table 1
List of symbols

Symbol	Description
n	Number of nodes
m_i	Upper limit on the number of packets $node_i$ can send
D	length of superframe
$p_i(k)$	Probability that $node_i$'s workload is k packets
$y_i(k)$	Probability that $node_i$ sends k or more packets
$b_i(k)$	Modulation level used by $node_i$ to send its kth packet
L	Maximum transmission unit of the underlying communication protocol in bits
t_{bit}	Time to send a bit
t_{symbol}	Time to send a single symbol
R_s	Symboling rate
b_{min}	Minimum modulation level that a node can use
b_{max}	Maximum modulation level that a node can use
$\beta_l^{i,k}$	A binary indicator that equals 1 for the selected modulation level l for $node_i$'s kth packet
p_e	Power consumption of the electronic circuitry
p_s	Power consumption from transmission

R_s is constant. A typical value for R_s is 62,500 symbols/s for 802.15.4 [40]. By using the radio power consumption formula from Section 3, we get:

$$e_{packet} = L \cdot (p_s + p_e) \cdot t_{bit} = \frac{L \cdot (C_s \cdot \phi(b) + C_e)}{b} \tag{1}$$

Define $y_i(k)$ as the probability that node i will actually transmit the kth packet. Then $y_i(k) = \sum_{x=k}^{m_i} p_i(x)$. The total *expected* energy consumption is the sum of expected energy consumption of n nodes:

$$e_{expected} = \sum_{i=1}^{n} \sum_{k=1}^{m_i} e_{packet} \cdot y_i(k) \tag{2}$$

By denoting the modulation level of the kth packet of the node i by $b_i(k)$, we obtain:

$$e_{expected} = \sum_{i=1}^{n} \sum_{k=1}^{m_i} \frac{L.y_i(k)}{b_i(k)} .[C_s.\phi(b_i(k)) + C_e] \tag{3}$$

Note that the kth packet of node i can potentially be transmitted with any of the discrete modulation levels in the range $[b_{min}, \ldots, b_{max}]$. Let $\beta_l^{i,k}$ be a binary indicator variable $\in \{0, 1\}$ to represent whether the kth packet of node i is transmitted using the modulation level l or not. Then an integer programming formulation to minimize the expected energy can be obtained as:

$$\text{minimize} \quad \sum_{i=1}^{n} \sum_{k=1}^{m_i} \sum_{l=b_{min}}^{b_{max}} \beta_l^{i,k} \cdot \frac{L \cdot y_i(k)}{b_l} \cdot [C_s \cdot \phi(b_l) + C_e] \tag{4a}$$

$$\text{subject to} \quad \sum_{l=b_{min}}^{b_{max}} \beta_l^{i,k} = 1 \ \forall i, k \tag{4b}$$

$$\sum_{i=1}^{n}\sum_{k=1}^{m_i}\sum_{l=b_{min}}^{b_{max}} \beta_l^{i,k} \cdot \frac{L}{b_l.R_s} \leq D \tag{4c}$$

$$\beta_l^{i,k} \in \{0,1\}\ \forall i,k \tag{4d}$$

The objective function gives the sum of the energy consumption of all the packets over all the nodes, by considering their probability of being transmitted and all possible modulation levels. The constraints (4b) and (4d) indicate that exactly one modulation level will be assigned to each packet in the workload. The constraint (4c) enforces that all the modulation levels must be selected in a way that all the transmissions will be completed before the deadline (the end of the superframe). Although integer programming problems are known to be intractable in the general case, moderate-size instances can be solved using the existing optimization tools such as CPLEX.

6. Proposed algorithms

Section 4 showed how static and dynamic algorithms operate when slack time is available. Section 5 illustrated how to optimally calculate the modulation levels to achieve energy minimization and required performance. Now we will discuss the algorithms that combine both.

i.) Static: In this algorithm, assuming the worst-case workload for every node, the smallest possible modulation level with which the deadline can be met is assigned to all the nodes in uniform manner, statically. The assigned modulation levels do not change for the duration of the superframe, even though the actual workload of a node may deviate from the worst-case (i.e., the slack time is not reclaimed).

*ii.) Static**: This is similar to the *Static* algorithm in the sense that the modulation levels are computed statically and slack is not reclaimed. However, instead of assigning a constant modulation level to every node, the nodes use a speed schedule that gradually increases the modulation levels by exploiting the probabilistic workload profile. This is computed by solving the integer programming problem with the objective function given in Eq. (4a).

iii.) Dynamic: This algorithm makes the initial modulation level assignments as in *Static* but then dynamically adjusts the modulation levels in order to take advantage of the available slack time after the end of each node's transmission and allows only the following node to reclaim this slack time by adaptively reducing the modulation level. At slack reclamation times, each node uses the smallest feasible modulation level to use the duration of its originally allocated slots and reclaimed slots.

*iv.) Dynamic**: This algorithm enables the dynamic reclaiming of the unused slots by adaptively reducing the modulation level at run-time. However, the initial modulation levels are computed using *Static** and the node that reclaims the slack uses the speed scheduling solution to re-assign possibly different modulation levels to each of its packets.

v.) Dynamic_f: The *fair* version of the *Dynamic* algorithm in the sense that the available slack time is distributed evenly among *all* subsequent nodes rather than being assigned entirely to the next node. The modulation levels of all the subsequent nodes are dynamically adjusted after the end of each node's transmission to the lowest feasible modulation level.

Static, Dynamic, and *Dynamic_f* are polynomial-time algorithms. They only iterate over each modulation level (from 2 to 8) once and select the minimum feasible one. *Static** solves the Binary-Integer Programming Problem introduced in Section 5 but it is executed offline by the coordinator and only once unless the probability distribution changes. *Dynamic** also solves the same Binary Integer Problem but only for a single node. In practice, a look-up table can be constructed with the pre-computed modulation levels as a function of available slack.

Fig. 7 shows an example of possible slack reclamation of dynamic algorithms. In this example there are 5 nodes with maximum workload of 10 packets. Initially, each node is assigned slots with total length equal to 10 packets with the modulation level b where $b > b_{min}$. When it is $node_1$'s turn, it sends 6 packets using the modulation level b which yields a slack time of 4-packets long. *Dynamic* and *Dynamic** allocate this slack time to the next scheduled node, namely $node_2$. Now, $node_2$ has effectively additional slots, giving a transmission time equal to 14 packets. However, $node_2$ will transmit at most 10 packets so it can reduce its modulation levels. In the case of *Dynamic*, $node_2$ uses the lowest feasible modulation, b_D, where $b_D < b$ for each of its 10 probable packets. *Dynamic** uses the optimal modulation levels computed by solving Eq. (4a) only for its probabilistic workload and slot length. When it is $node_2$'s turn, it ends up transmitting 5 packets implying there is a slack time of 9 packets with modulation b. Similarly, *Dynamic* and *Dynamic** assign this slack time to the next scheduled node, namely $node_3$. In the *Dynamic* case, $node_3$ uses the lowest feasible modulation level, $b_{D'}$, where $b_{D'} \leq b_D \leq b$. $node_3$ uses the optimal solution computed for its own packets with its own deadline. In the *Dynamic_f* case, the 4-packet long slack time after $node_1$'s transmission is distributed among $node_2$, $node_3$, $node_4$, and $node_5$. These nodes have 11-packet long slack time with the modulation level b. The lowest feasible modulation level, b_{D_f}, that will meet with the deadline with 40 possible packets is computed where $b_{D_f} \leq b$. After $node_2$ stops transmitting, the 9-packet long slack is distributed among $node_3$, $node_4$, and $node_5$. The new lowest feasible modulation level $b_{D'_f}$ for all 30 possible packets to meet the deadline is computed where $b_{D'_f} \leq b_{D_f} < b$.

7. Performance evaluation

To evaluate the performance of the several variants of the proposed framework under different workload conditions, we simulated the system on Castalia framework of Omnet++ simulator. We simulated a system with a coordinator and 10 nodes arranged in star topology, and with communication range set to 30 m. The work done in [41] shows that DMS is effective for distances greater than 25 m. Each node's workload in a superframe varies between 1 and 10 packets and is derived from a probability distribution. We assumed DMS-capable systems (with QAM modulation) where the modulation levels can vary from 2 to 8.

The purpose of our simulation is to quantify, from an algorithmic perspective, the difference between DMS-aware and DMS-oblivious approaches in energy-aware super-frame management. In order to achieve this, we ran various simulations for different superframe lengths (deadlines) to analyze how the energy consumption varies. Furthermore, we have compared our proposed algorithms against an *Oracle* algorithm which is the *yardstick* scheme where the exact number of packets that each node will transmit is known in advance, at the beginning of each superframe. As a result, it does not need to assume the worst-case workload. *Oracle* does not require any LPL because it knows the exact time each node will stop transmitting. Hence, the overhead of LPL is also omitted. Although it is not a feasible algorithm in practice, it provides the minimum energy consumption that is theoretically possible for a given experiment.

The minimum deadline D_0 is assumed to be the superframe length necessary to allow the transmission of the worst-case workload (10 packets) by each node at the maximum modulation level,

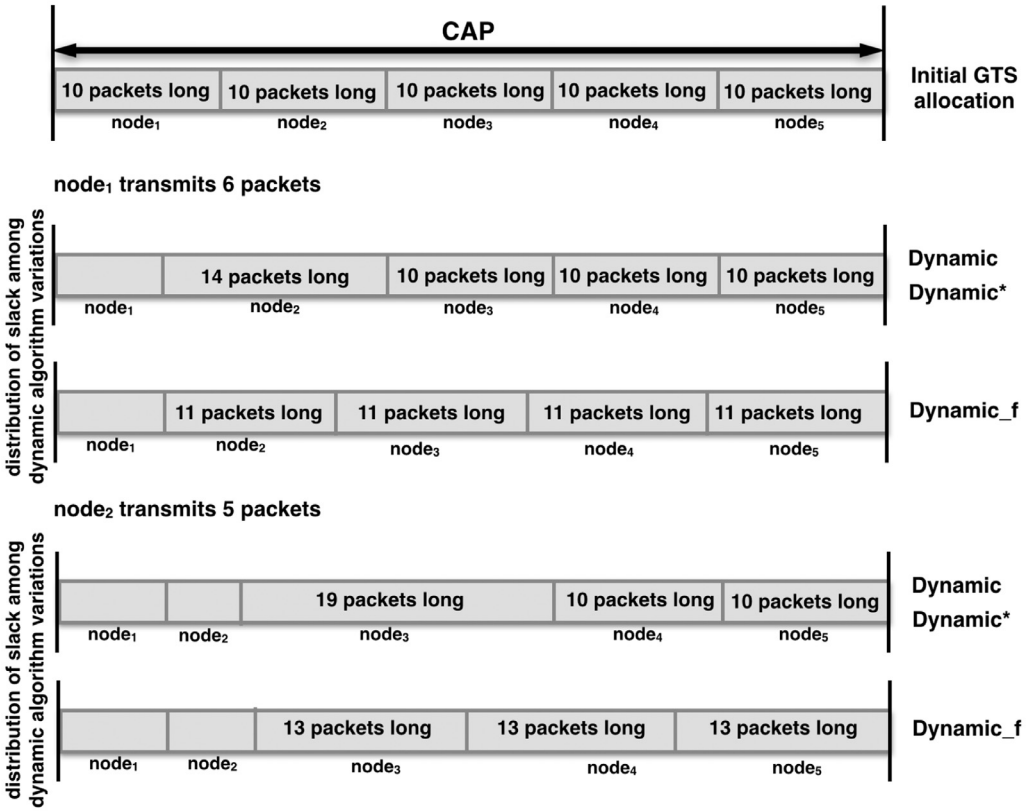


Fig. 7. An example for dynamic algorithms.

considered to be equal to 208 ms.[1] The actual deadline for a given experiment is then computed as $D = \frac{D_0}{load}$ where the system's *load* is in the range [0.1, 1.0].

For each *load* value, our simulator generated 600 workload instances for 10 nodes using a uniform distribution function. We also ran experiments with Normal, Pareto and Flipped-Pareto distributions for workload generation. We will present detailed simulation results for the Uniform distribution and underline the trends and relative ordering of the schemes for the other distributions.

Castalia is a high-fidelity simulator with an advanced channel model that incorporates log-normal path loss with temporal variations [42]. It is not platform-specific and allows reliable and realistic validation of wide range of algorithms and platforms [43]. The packet loss is computed according to collisions as well as by comparing the energy level of the received packet to the noise level in the environment. Our simulation implementation complies with 802.15.4-2006 standard, and allows data transfer during CAP period where nodes only use slotted CSMA/CA. In the slotted CSMA/CA, a node needs to wait for a random number of backoff slots to transmit data packet but the acknowledgement packet does not have to use slotted CSMA/CA. During the CFP period, the coordinator sends ACK packets after each successful data packet transfer. IEEE 802.15.4-2006 standard describes how the superframe intervals must be calculated. The sum of active and inactive period lengths must be equal to $BaseSuperframeDuration = NumberOfSuperframeSlots \times symbolTime$. $SymbolTime$ is calculated as

$$\frac{1}{physicalDataRate \times 1000/physicalBitsPerSymbol}$$

and then $BeaconInterval$ is calculated as $BaseSuperframeDuration \times 2^{BeaconOrder}$. Here, $BeaconOrder$ is a constant and is equal to 6 in our simulations. The active portion of the superframe is $ActiveInterval = BaseSuperframeDuration \times 2^{ActiveOrder}$. We have chosen 4 as our $ActiveOrder$ constant. Also, the number of time slots assigned to CAP period needs to be specified in order for the CAP length to be calculated. We set the CAP period to 2 GTS long. The transition cost in terms of energy and delay between RX, TX, and Sleep states are also included.

- $MaximumNumberOfTries\_CAP = 4$, $MaximumNumberOfTries\_CFP = 2$, $guardTime = 1$ ms
- Clear Channel Assessment related factors: IEEE 802.15.4-2006 specifies three modes of performing CCA. Castalia's radio module is built to provide *Mode 1* which checks whether the measured energy is above a threshold value or not. The default time duration to measure the energy level is set to 0.000128 ms which is independent of the radio that is being used. The default value of energy threshold is −95 dBm.
- Transition costs: We only consider the light sleep level which consumes 0.5 mW. The list of transition costs are; RX to TX = 32 mW, TX to RX = 32 mW, RX to Sleep = 1.4 mW, Sleep to RX = 1.4 mW, TX to Sleep = 1.4 mW, and Sleep to RX = 0.5 mW.
- Transition delays: Once again only with the light sleeping mode the transition delays are as follows: RX to TX = 0.01 ms, TX to RX = 0.01 ms, RX to Sleep = 0.05 ms, Sleep to RX = 0.194 ms, TX to Sleep = 0.05. These values are obtained from the CC2420 radio specification.
- Modulation level parameters: DataRate (kbps) is calculated as symbolRate*bitsPerSymbol where symbolRate is constant and 62,500 for 2450 MHz radios such as CC2420. Bandwidth, noiseBandwidth, noiseFloor, and Sensitivity values are taken from the CC2420 radio specification, and they are 20 MHz, 194 MHz, −100 dBm, and −95 dBm respectively. TX_dBm levels which af-

[1] As in low-power listening mode each node can miss up to 2 preambles before it can start transmitting, this duration as well as the transmission delay are included in D_0 to ensure feasibility.

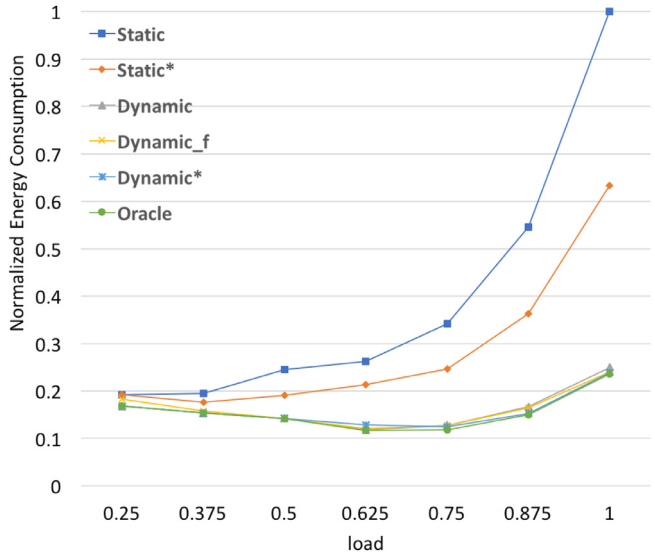


Fig. 8. Energy consumption in the ideal case. Here, we assume dynamic algorithms hypothetically know the exact time they need to wake up in order to start transmitting. Hence, LPL is not needed.

fect packet loss, CCA, and neighboring problem (see Section 4) are 5, 8, 11, 14, 17, 20, 23 dBm, respectively, for modulation levels 2–8.

All the proposed algorithms run on the coordinator. The computed modulation levels are transmitted to the nodes using beacons and preambles. Hence, the computation overhead on individual nodes is minimal. We set the preamble size to 14 bytes, which includes a 1 byte for sender address, a 1 byte for receiver address, 10 bytes for the calculated modulation levels, and 2 bytes for the CRC footer. For the *Static*, *Static**, and *Oracle* algorithms the beacon size is set to 124 bytes; while it is set to 24 bytes for the *Dynamic*, *Dynamic**, and *Dynamic_f* algorithms.

All the dynamic algorithms require the same amount of signaling. Furthermore, we assumed that the coordinator uses the maximum modulation level to broadcast the preamble. In our simulation settings, the *listening-duration* $\gamma = 0.000128$ ms and the *sleep-duration* $\alpha = t_{preamble} - \gamma$. For the C_e and C_s values described in Section 3, we have used 15×10^{-9} and 12×10^{-9} J, respectively, and $b_{min} = 2$, $b_{max} = 8$, after [7,38]. All the simulation results are presented at 95% confidence level. In all the plots presented, the energy consumption values of various schemes are normalized with respect to the energy consumption of *Static* at *load* = 1.0.

7.1. Analysis of the ideal case

In this section we evaluate the proposed algorithms' ideal case performances. In *ideal case*, we assume that the nodes have exact knowledge about the time at which they need to wake up, in advance. The static algorithms can incorporate this information in the beacon message. For dynamic algorithms we assumed the same beacon message structure. Obviously in this ideal case, the need for low-power listening disappears. Still, we believe the analysis of this case reveals some important patterns because it yields the upper bounds on the energy savings that each algorithm can provide with zero-overhead low-power listening.

Fig. 8 shows the normalized energy consumption of the proposed algorithms. We observe that on higher load values, the *Dynamic, Dynamic**, and *Dynamic_f* algorithms give significant energy savings compared to *Static* and *Static** algorithms. Moreover, *Dynamic* and *Dynamic** perform better than *Dynamic_f*. However, at lower load values, the dynamic algorithms provide only limited gains; this is because even the static algorithms are able to assign low modulation levels when the system has ample time to finish the workload.

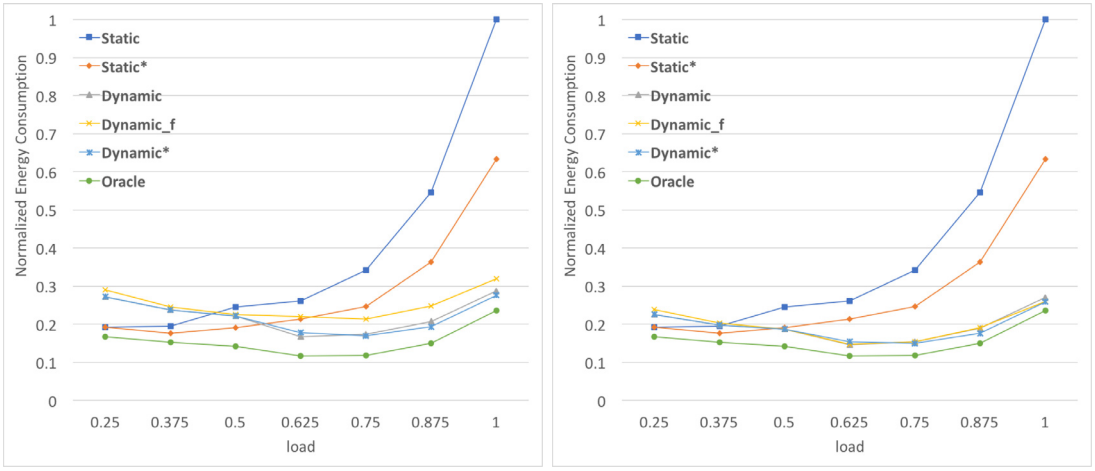
It is observed that the energy consumption is minimized for the load value 0.625. For the load values smaller than 0.625 the sleeping energy consumption becomes dominant and for the load values greater than 0.625 the transmission and reception energy consumptions become dominant.

7.2. Analysis of the effect of traditional LPL with no interference

In this section, we will show the effect of low-power listening on the proposed algorithms. The ideal case where the nodes know exactly when the previously scheduled node stops transmitting cannot be implemented in real-life scenarios. The nodes need to listen for a preamble from the coordinator to see when they can start transmitting. One possibility is to use the traditional low-power listening (without the HLPL enhancement described in Section 4) and our results in this section consider this case, by further assuming that the cross-node interference is negligible. In Section 7.3, we will re-analyze these settings within the HLPL framework by considering the impact of the interference.

Fig. 9a shows the normalized energy consumption of greedy low-power listening enabled algorithms. The compared algorithms are *greedy* in the sense that they use traditional low-power listening with the wait-duration δ set to zero. We need to recall that only *Dynamic, Dynamic**, and *Dynamic_f* require low-power listening. The remaining algorithms have pre-determined wake-up times. We can see that the dynamic algorithms perform poorly compared to static algorithms when $load \leq 0.6$. This is due to the fact that for lightly loaded systems, the gain from dynamic reclamation of the slack times is offset by the additional energy consumption due to energy overhead of traditional low-power listening activity. The dynamic algorithms' energy performance improves only when *load* approaches and exceeds 0.6 (this threshold is slightly larger for *dynamic_f*) – this is when the overhead of low-power listening (necessary to implement the reclaiming mechanism) becomes reasonably low compared to the gains of adaptive modulation downscaling at run-time. We also observe the energy consumption gap between dynamic algorithms becomes more significant where *Dynamic* and *Dynamic** performs very closely and outperforms *Dynamic_f*.

Another possibility for the implementation of the traditional low-power listening in these settings is to have each node wait for a time duration δ equal to the *expected time needed for the completion of the packet transmissions by the previous nodes*. The idea is to take advantage of the known probability distribution. Rather than letting nodes start low-power listening as soon as the collision-avoidance-period starts, the nodes calculate the expected number of packets that will be transmitted by the previously scheduled nodes based on the known probability distribution function. We call this scheme *smart-LPL*. The expected-number-of-packets before $node_i$ can start to transmit is $\sum_{k=1}^{i-1} \sum_{l=1}^{m_l} p_k(l) \times l$. The scheduling order is embedded into the beacon message. Two observations are in order here: *i*) if the node starts low-power listening before its actual turn then the node spends more energy for low-power listening but does not miss any of its slack time. However, if the node wakes up after its turn starts then the node loses some portion of the given slack time (the node could not reduce its modulation levels as much as it could have) but spends less energy on low power listening. Hence, there is a trade-off between the gain from low-power listening and loss from smaller slack times. *ii*) The modulation level assumed in the calculation of the expected-wait-time for the previous nodes is another critical variable. The nodes know the expected number of packets to be sent before their turn, but they do not know what modulation levels have been used by the pre-



(a) Energy consumption of greedy-LPL where $\delta = 0$

(b) Energy consumption of smart-LPL where δ =expected-wait-time

Fig. 9. Simulation results with no interference. We assume each node node only hears the coordinator and none of its neighbors. Hence, the neighborhood problem described in Section 4 hypothetically does not exist in this settings.

vious nodes (they cannot accurately map the expected number of packets to the expected amount of waiting time). In our simulation settings, we have used the modulation level calculated by *Static* to compute the expected-wait-time values.

Fig. 9b illustrates the normalized energy consumptions with smart-LPL. Our analysis reveals that with smart-LPL, the energy consumption of the dynamic algorithms is reduced compared to the greedy-LPL case. One important observation here is the effect of low-power listening on the dynamic algorithms. The increased gap between dynamic algorithms observed in Fig. 9a becomes less significant in Fig. 9b. This is a result of two factors: *i)* shorter GTS slots mean longer duration to listen for a preamble and hence lead to higher overhead caused by low-power listening; *ii)* For the case shown in Fig. 9b, the expected-wait-time values are calculated using the modulation level given by the *Static* algorithm. However, higher ideal case performance implies that the previously scheduled nodes have used smaller modulation levels than initially computed. This leads to less accuracy in predicting expected-wait-time and as a result, a longer duration for traditional low-power listening. This is a crucial result that shows how dynamic modulation levels can affect low-power listening and becomes one of the fundamental reasons necessitating the use of HLPL protocol.

7.3. Analysis of the impact of neighborhood/interference

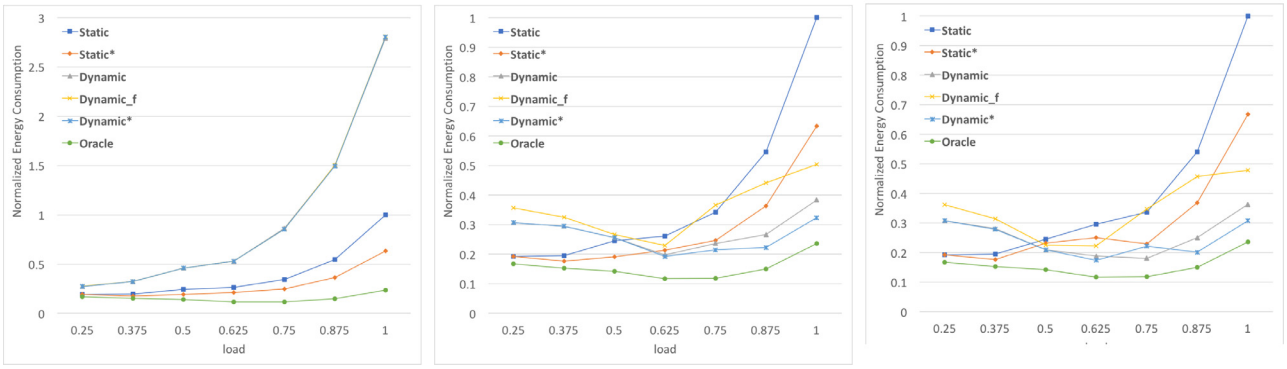
In this section, our aim is to evaluate the effect of neighborhood/interference on traditional low-power listening and also include our newly proposed HLPL in the comparison. In real settings when a node wakes up to check for a preamble, it has to listen to its neighbors' communications to make sure that the communication it is sensing is not a preamble. In order for a node to make sure that it is not receiving a preamble, it may need to listen the channel for up to two preamble transmission times as shown in Fig. 4.

Fig. 10a shows the normalized energy consumption with traditional low-power listening and possible interference. A striking observation is the significantly increased energy consumption of the dynamic algorithms for most of the spectrum, due to the prohibitive energy consumption of *false alerts* induced by the interference due to the naive application of the traditional low-power listening framework. In this case, the nodes receive *false alerts* from their neighbors and they need to verify the content of these transmissions. The length of preamble message is 14-byte long whereas the MTU of 802.15.4 is 127 bytes. This indicates that even for a single packet with the highest modulation level, the node has to consume an additional energy of listening up to 2/3 of a packet (which is 84 bytes) to see if there is or there is not a preamble addressed to itself. In a neighborhood of size 4, this may create and additional overhead up to 26 packets per node as can be seen from Fig. 10a.

The overhead created by the interference also depends on the values used for sleep-duration and *preamble size*. In our simulations, we have observed that larger *sleep-duration* values tend to decrease the overhead induced by the interference. However, longer sleep-duration has other consequences such as longer superframe lengths and larger losses in the available slack times. In order to ensure the deadlines, we have to account for the maximum time a node can miss before it hears a preamble. This maximum time needs to be added to the minimum feasible superframe length to ensure the feasibility of the system.

Some optimal values of preamble length and sleep-duration values that will minimize this overhead may exist. However, we believe even this minimized overhead will still be undesirable especially for lower utilization factors where *offline* algorithms perform well. Finding this minimized overhead value is left as a future work.

Fig. 10b shows the simulation results obtained after adopting greedy-HLPL. Comparing to Fig. 10a, one can see the drastic energy savings provided by the greedy-HLPL. *Dynamic* and *Dynamic** outperform the *Static* algorithm for load values higher than 0.52. For load value 0.6 and higher, we observe that *Dynamic* and *Dynamic** have less energy consumption than *Static**. *Dynamic_f* outperforms *Static** for load values roughly after 0.91. If we compare Fig. 10b with Fig. 9a, we can see that the performance of *Dynamic* and *Dynamic** algorithms in the presence of interference is rather close to the one in the no-interference case where *Dynamic_f* results in a more significant increase. Fig. 10c shows the normalized energy consumption of smart-HLPL when wait-duration is equal to expected-wait-time. This case further reduces the overall energy consumption of dynamic algorithms. In this case, *Dynamic* outperforms the static algorithms for load values roughly larger than 0.45.

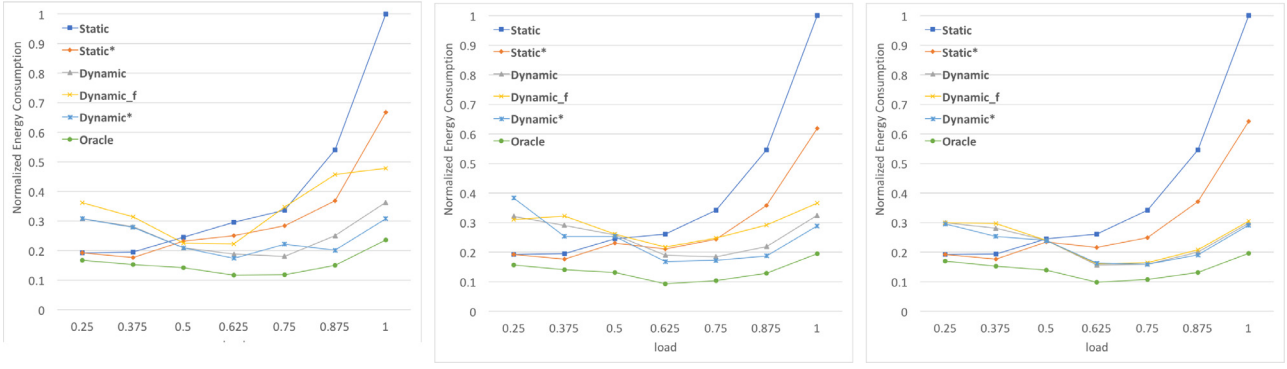


(a) Energy consumption with greedy-LPL and interference

(b) Energy consumption with greedy-HLPL and interference where $\delta = 0$

(c) Energy consumption with smart HLPL and interference where $\delta =$ expected-wait-time

Fig. 10. Impact of interference.



(a) Flipped-Pareto distribution

(b) Normal distribution

(c) Pareto Distribution

Fig. 11. Energy consumption of smart-HLPL with different probability distribution functions.

As can be seen, HLPL successfully addresses the neighborhood/interference problems and yields significant energy savings. Finally, the comparison of Fig. 8 with Fig. 10c shows that the results of HLPL are reasonably close to the ideal case, showing the potential of the framework.

7.4. Effect of different probability distribution functions

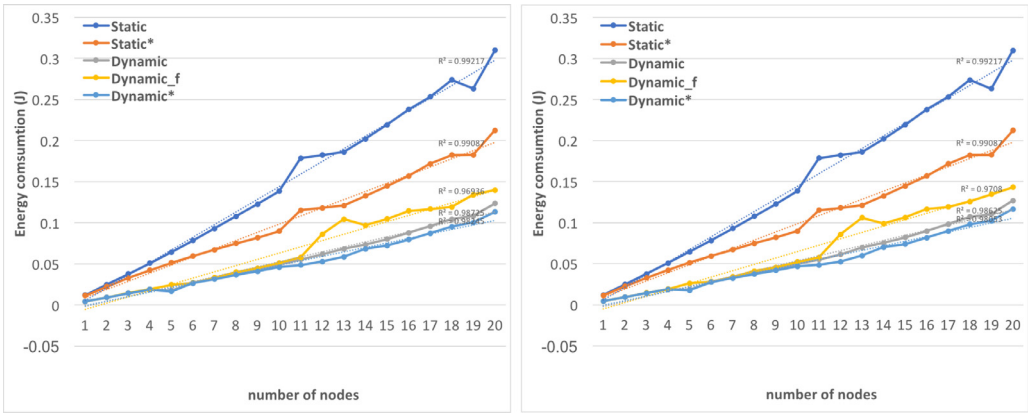
All the results presented so far were obtained under Uniform probability distribution for the packet workload. We have also repeated all the simulations with Normal Distribution (with the mean $\mu = 5$ and standard deviation $\sigma = 2$), Pareto and Flipped-Pareto distributions (with the shape parameter $k = 10$, scale parameter $\sigma = 3$, and threshold value $\theta = \frac{10}{3}$).

An important difference is in terms of the average energy consumption under different distributions. Our simulation results show that the Pareto distribution has the lowest average energy consumption followed by Normal, Uniform, and Flipped-Pareto distributions. This is expected due to the fact that each distribution function has different expected workload figures which are 3.22, 5.04, 5.5, 7.78 for Pareto, Normal, Uniform, and Flipped-Pareto distributions, respectively.

Fig. 11 shows the energy consumptions for different probability distributions. We can draw several conclusions: *i)* Distribution functions have limited impact on the results presented in previous sections; the ordering of the algorithms is still the same for each of the tested probability distribution functions. *ii)* For all the cases analyzed in previous sections: the gap between the average energy consumption values of the algorithms got smaller for the Pareto distribution case. The *dynamic* algorithms have performed very close for these load values greater than 0.5. Fewer number of packets led to limited difference in transmission energy consumption. For similar reasons, this gap became larger for Flipped-Pareto distribution function. We can say that when the nodes have higher workloads, the performance gaps between *dynamic* algorithms get larger and for the cases where the nodes have lower workloads, the gap between *Dynamic_f* and *Dynamic** as well as the gap between *Dynamic** and *Dynamic* get smaller. *iii)* In the Pareto distribution case, *Dynamic, Dynamic** and *Dynamic_f* outperformed *Static** for the load value roughly 0.5. These values are slightly lower than the results presented in previous sections.

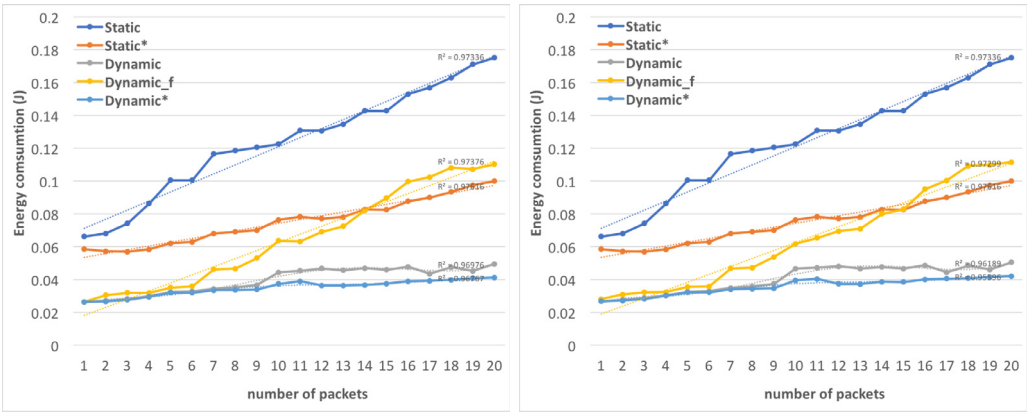
7.5. Analysis of scalability

We have analyzed the scalability of HLPL in terms of number of nodes, and number of packets. When assessing the impact of the number of nodes, we conducted simulations with 1 to 20 nodes each with uniformly distributed workload of 10 packets. When



(a) number of nodes smart-HLPL (b) number of nodes greedy-HLPL

Fig. 12. Scalability in terms of number of nodes.



(a) number of packets smart-HLPL (b) number of packets greedy-HLPL

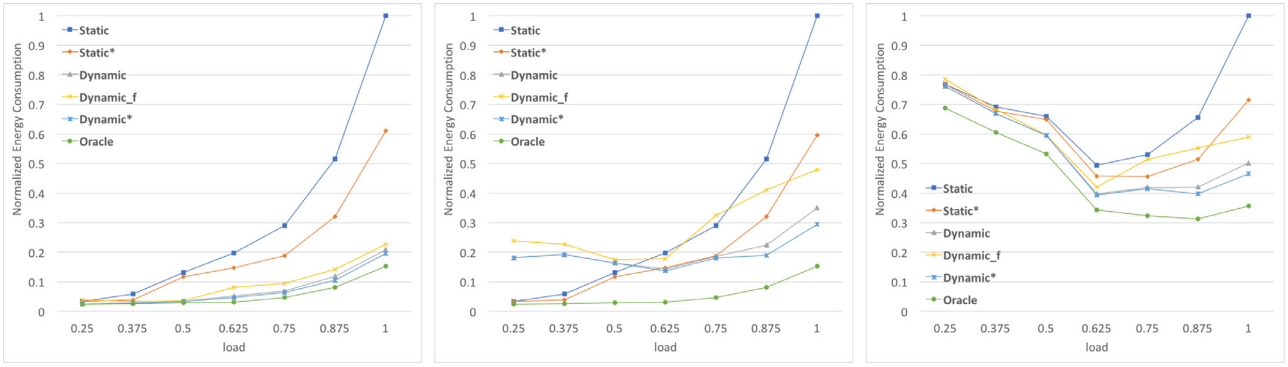
Fig. 13. Scalability in terms of the worst-case number of packets.

evaluating the impact of the worst-case number of packets, we assumed 10 nodes with uniformly distributed 1–20 packets of probabilistic workload. Figs. 12 and 13 show the scalability in terms of number of nodes and packets. The linear regression analysis shows that for all of the scheduling algorithms except *Dynamic* and *Dynamic** in Fig. 13a and b, the average energy consumption grows linearly with respect to number of nodes and number of packets where each of the regression analysis had an R-squared value of 0.96 or higher. For the mentioned *Dynamic* and *Dynamic** results, a 6th degree polynomial regression had R-squares value of 9.95 or higher. The general ordering of the algorithms has not changed in terms of number of nodes. However, we observed that *Static** outperforms *Dynamic_f* for 14 packet workloads.

7.6. Impact of radio hardware variations

In this section we will discuss the effect of different values of radio power consumption. We first ran a series of experiments with sleeping-power consumption of 0 mW and 3 mW. In the previously reported results, this value was set to 1.4 mW. Fig. 14 shows the effect of sleep power consumption with 10 nodes, maximum of 10 packets workload with uniform distribution. Fig. 14a corresponds to the ideal case (described in Section 7.1) with sleep power consumption assumed to be zero. Here, we observe a strict increase in energy consumption with increasing load value. This result is different than the one shown in Fig. 8 which has the minimum energy consumption at the load value of 0.625. This is because lower load values mean longer superframe duration and hence increased energy consumption from sleeping. When we take sleeping energy consumption out of the equation (recall that ideal case does not require low-power-listening), higher load values results in strictly higher energy consumption due to higher modulation levels. Fig. 14b shows the energy consumption with greedy-HLPL when sleep power consumption is set to 0. Comparing this with Fig. 10b, we see the energy consumption gap between the highest and the lowest load levels increases. Fig. 14c shows the case where sleep power consumption is set to 3 mW. The lowest load level results in the highest energy consumption except for *Static*. For the other algorithms, the sleeping energy consumption is higher than the energy savings from using lower modulation levels. This case also shows that the *Dynamic* and *Dynamic** outperforms static algorithms for every load value which further emphasizes the effectiveness of HLPL.

Next, we experimented with the CC2420 based power consumption values. The results are presented in Fig. 15. CC2420 only has a single modulation level of 4, which consumes 62 mW. If we assume the same exponential increase of initial test values, the transmission/reception power consumption can be estimated to be 15, 31, 62, 124, 248, 496, 992 mW for modulation levels 2, 3, 4, 5, 6, 7, 8 respectively. Fig. 15a shows the energy consumption results for the greedy-HLPL. These settings shows using *Static** consumes less energy for every load value hence, the best option. This is due to the very expensive clear channel assessment performed by the dynamic algorithms exceeding the energy savings from lowering

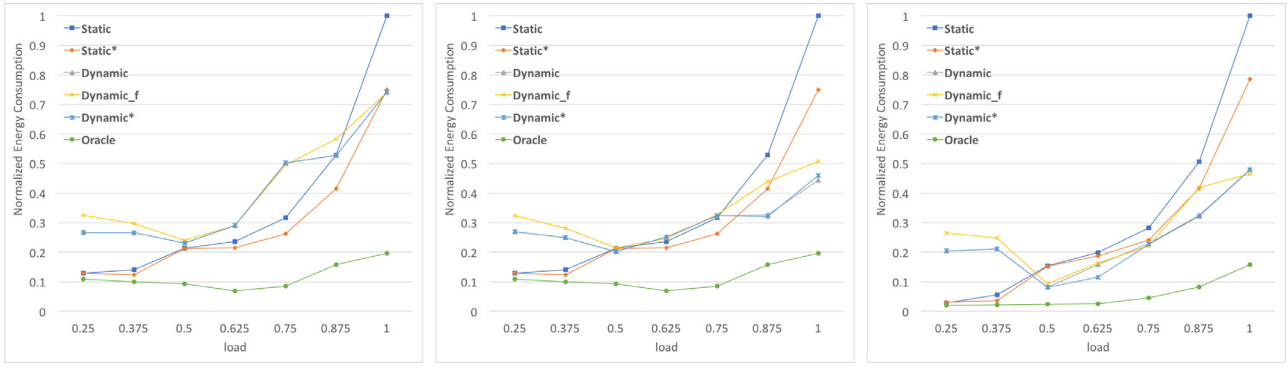


(a) Ideal case with sleep power = 0 mW.

(b) Greedy-HLPL with sleep power = 0 mW.

(c) Greedy-HLPL with sleep power = to 3 mW

Fig. 14. The effect of sleep power consumption.



(a) Greedy-HLPL with lightest sleep level

(b) smart-HLPL with lightest sleep level

(c) smart-HLPL with multiple sleep-levels

Fig. 15. Energy consumption with CC2420 based power consumption settings.

modulation levels. However, Fig. 15b gives significantly different results with smart-HLPL with only the lightest low-power-mode called *idle*. In this mode, CC2420 turns off its frequency synthesizer and draws 426 μA of current. The results showed that *dynamic* and *dynamic** outperforms *static** for load values of 0.8 or higher and *dynamic_f* does so for the load value of roughly 0.9. Moreover, CC2420 has two more low-power-modes namely *power down* and *power off*. In *power down* the crystal oscillator is turned off in addition to frequency synthesizer and draws 20 μA of current, and in *power off* the voltage regulator is also turned off with a total of 0.02 μA current draw. It takes 0.6 ms to transition from *power off* to *power down* and 1.0 ms from *power down* to *idle*. We assumed that during these transitions there is a current draw equal to the level with the higher current draw value. All the algorithms are adjusted to consider all low-power-modes and put the nodes into the deepest one wherever expected-wait-time exceeds the respective break-even point. The results shown in Fig. 15c indicate a significant improvement. The dynamic algorithms outperform the static ones for the load value of roughly 0.4 and greater. This further shows the benefits of smart-HLPL.

8. Conclusions

This paper addressed the problem of ensuring real-time guarantees while minimizing the overall energy consumption in wireless sensor networks. We focused on cluster-oriented superframe communication, the most widely adopted method for providing real-time guarantees in industrial wireless networks. Using Dynamic Modulation Scaling we studied static and dynamic algorithms for reallocation of slack times in a superframe. We analyzed the effect of interference and dynamic modulation levels on low-power listening. We also introduced a new low-power listening protocol called *hybrid-low-power listening* (HLPL) in order to overcome the interference problem caused by neighborhood. Using th Castalia Simulator we empirically assessed the performance improvements of DMS slack reclaiming and HLPL. Our experiments show that dynamic slot readjustment saves a significant amount of energy under highly loaded systems. They also indicate that HLPL overcomes the interference caused by other nodes in the cluster and significantly reduces the overall energy consumption of the system.

References

[1] V.C. Gungor, G.P. Hancke, Industrial wireless sensor networks: challenges, design principles, and technical approaches, Ind. Electron. IEEE Trans. 56 (10) (2009) 4258–4265.

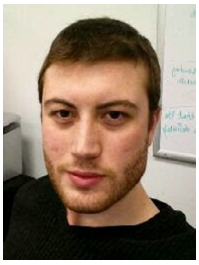
[2] B. Galloway, G. Hancke, Introduction to industrial control networks, IEEE Commun. Surv. Tutor. 15 (2) (2013) 860–880, doi:10.1109/SURV.2012.071812.00124.

[3] L. Filipponi, A. Vitaletti, G. Landi, V. Memeo, G. Laura, P. Pucci, Smart city: An event driven architecture for monitoring public spaces with heterogeneous sensors, in: Proceedings of Fourth IEEE International Conference on Sensor Technologies and Applications, 2010, doi:10.1109/SENSORCOMM.2010.50.

[4] W. Liang, X. Zhang, Y. Xiao, F. Wang, P. Zeng, H. Yu, Survey and experiments of wia-pa specification of industrial wireless network, IEEE Wireless Commun. Mobile Comput. 11 (8) (2011) 1197–1212.

[5] V. Cionca, T. Newe, V. Dadarlat, Tdma protocol requirements for wireless sensor networks, in: Proceedings of Second IEEE International Conference on Sensor Technologies and Applications, 2008, doi:10.1109/SENSORCOMM.2008.69.

[6] R. Szabo, Framework for smart city applications based on participatory sensing, in: Proceedings of 4th IEEE International Conference on Cognitive Infocommunications, 2013, doi:10.1109/CogInfoCom.2013.6719260.
[7] C. Schurgers, V. Raghunathan, M.B. Srivastava, Power management for energy-aware communication systems, ACM Trans. Embedded Comput. Syst. 2 (3) (2003) 431–447.
[8] V. Raghunathan, C. Schurgers, S. Park, M.B. Srivastava, Energy-aware wireless microsensor networks, IEEE Signal Process. Mag. 19 (2) (2002) 40–50.
[9] C. Dong, L.-L. Yang, L. Hanzo, Performance of buffer-aided adaptive modulation in multihop communications, Commun. IEEE Trans. 63 (10) (2015) 3537–3552, doi:10.1109/TCOMM.2015.2469287.
[10] http://www.ti.com/lit/ds/symlink/cc1200.pdf.
[11] http://www.ti.com/lit/ds/symlink/cc2500.pdf.
[12] M. Bandari, R. Simon, H. Aydin, On minimizing expected energy usage of embedded wireless systems with probabilistic workloads, Sustainable Comput. Inf. Syst. 11 (2016) 50–62.
[13] A. Gumusalan, R. Simon, H. Aydin, Adaptive transmission scheduling for energy-aware real-time wireless communication, in: Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks, in: EWSN, 2016.
[14] T. Hamachiyo, Y. Yokota, E. Okubo, A cooperative power-saving technique using dvs and dms based on load prediction in sensor networks, in: Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on, IEEE, 2010, pp. 7–12.
[15] B. Fateh, M. Govindarasu, Energy minimization by exploiting data redundancy in real-time wireless sensor networks, Ad Hoc Netw. 11 (6) (2013) 1715–1731.
[16] M.H. Anisi, G. Abdul-Salaam, M.Y.I. Idris, A.W.A. Wahab, I. Ahmedy, Energy harvesting and battery power based routing in wireless sensor networks, Wireless Netw. (2015) 1–18.
[17] J. Polastre, J. Hill, D. Culler, Versatile low power media access for wireless sensor networks, in: Proceedings of 2nd ACM International Conference on Embedded Networked Sensor Systems, 2004.
[18] I. 802.15.4e Standard, https://standards.ieee.org/findstds/standard/802.15.4e-2012.html, Last accessed on 2015-May-22.
[19] J. Song, S. Han, A.K. Mok, D. Chen, M. Lucas, M. Nixon, Wirelesshart: Applying wireless technology in real-time industrial process control, in: Proceedings of IEEE Real-Time and Embedded Technology and Applications Symposium, 2008.
[20] T. Lennvall, S. Svensson, F. Hekland, A comparison of wirelesshart and zigbee for industrial applications, in: Proceedings of IEEE International Workshop on Factory Communication Systems, 2008.
[21] C. Na, Y. Yang, A. Mishra, An optimal {GTS} scheduling algorithm for time-sensitive transactions in {IEEE} 802.15.4 networks, Comput. Networks 52 (13) (2008) 2543–2557. https://doi.org/10.1016/j.comnet.2008.05.012.
[22] I. Rhee, A. Warrier, M. Aia, J. Min, M.L. Sichitiu, Z-mac: a hybrid mac for wireless sensor networks, IEEE/ACM Trans. Netw. (TON) 16 (3) (2008) 511–524.
[23] M. Salajegheh, H. Soroush, A. Kalis, Hymac: Hybrid tdma/fdma medium access control protocol for wireless sensor networks, in: Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on, IEEE, 2007, pp. 1–5.
[24] S. Mehta, K.-S. Kwak, H-mac: a hybrid mac protocol for wireless sensor networks, ITC-CSCC: 2007 (2007) 755–756.
[25] L. Sitanayah, C. Sreenan, K. Brown, Er-mac: A hybrid mac protocol for emergency response wireless sensor networks, in: Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on, 2010, pp. 244–249, doi:10.1109/SENSORCOMM.2010.45.
[26] S. Zhuo, Y.-Q. Song, Z. Wang, Z. Wang, Queue-mac: A queue-length aware hybrid csma/tdma mac protocol for providing dynamic adaptation to traffic and duty-cycle variation in wireless sensor networks, in: Factory Communication Systems (WFCS), 2012 9th IEEE International Workshop on, 2012, pp. 105–114, doi:10.1109/WFCS.2012.6242552.
[27] Y.K. Rana, B.H. Liu, A. Nyandoro, S. Jha, Bandwidth aware slot allocation in hybrid mac, in: Proceedings of 31st IEEE Conference on Local Computer Networks, 2006.
[28] B. Shrestha, E. Hossain, K.W. Choi, Distributed and centralized hybrid csma/ca-tdma schemes for single-hop wireless networks, IEEE Wireless Commun. 13 (7) (2014) 4050–4065, doi:10.1109/TWC.2014.2327102.
[29] M.H.S. Gilani, I. Sarrafi, M. Abbaspour, An adaptive csma/tdma hybrid {MAC} for energy and throughput improvement of wireless sensor networks, Ad Hoc Netw. 11 (4) (2013) 1297–1304. https://doi.org/10.1016/j.adhoc.2011.01.005.
[30] M. Guerroumi, A.-S.K. Pathan, A. Derhab, N. Badache, S. Moussaoui, Mmsmac: a multi-mode medium access control protocol for wireless sensor networks with latency and energy-awareness, Wireless Personal Commun. (2016) 1–38.
[31] A. Rajasekaran, V. Nagarajan, Adaptive intelligent hybrid mac protocol for wireless sensor network, in: Communication and Signal Processing (ICCSP), 2016 International Conference on, IEEE, 2016, pp. 2284–2289.
[32] G. Ali, K.H. Kim, K.-I. Kim, Adaptive tdma scheduling for real-time flows in cluster-based wireless sensor networks., Comput. Sci. & Inf. Syst. 13 (2) (2016).
[33] M.R. Lenka, A.R. Swain, M.N. Sahoo, Distributed slot scheduling algorithm for hybrid csma/tdma mac in wireless sensor networks, in: Networking, Architecture and Storage (NAS), 2016 IEEE International Conference on, IEEE, 2016, pp. 1–4.
[34] 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. Sensor Netw. (TOSN) 8 (4) (2012) 30.
[35] C. Cano, B. Bellalta, A. Sfairopoulou, M. Oliver, Low energy operation in wsns: A survey of preamble sampling mac protocols, Comput. Netw. 55 (15) (2011) 3351–3363.
[36] C. Schurgers, V. Raghunathan, M. Srivastava, Modulation scaling for real-time energy aware packet scheduling, in: Proceedings of IEEE Global Telecommunications Conference, vol. 6, 2001, doi:10.1109/GLOCOM.2001.966363.
[37] Y. Yu, B. Krishnamachari, V. Prasanna, Energy-latency tradeoffs for data gathering in wireless sensor networks, in: Proceedings of Twenty-third IEEE Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 1, 2004, doi:10.1109/INFCOM.2004.1354498.
[38] M. Bandari, R. Simon, H. Aydin, Energy management of embedded wireless systems through voltage and modulation scaling under probabilistic workloads, in: Proceedings of IEEE Green Computing Conference, 2014, doi:10.1109/IGCC.2014.7039168.
[39] 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 4th ACM International Conference on Embedded Networked Sensor Systems, 2006.
[40] F. Easdy, Hands-On ZigBee: Implementing 802.15.4 with Microcontrollers, Newnes, 2007.
[41] B. Fateh, M. Govindarasu, Joint scheduling of tasks and messages for energy minimization in interference-aware real-time sensor networks, Mobile Comput. IEEE Trans. 14 (1) (2015) 86–98.
[42] A. Boulis, et al., Castalia: A simulator for wireless sensor networks and body area networks, NICTA: Natl. ICT Aust. 83 (2011).
[43] D. Pediaditakis, Y. Tselishchev, A. Boulis, Performance and scalability evaluation of the castalia wireless sensor network simulator, in: Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010, p. 53.



Arda Gumusalan received his B.S. degree from State University of New York in 2012 and M.S degree from George Mason University, Fairfax, VA in 2017, both in computer science. Currently, he is working towards his Ph.D. degree at the Department of Computer Science, George Mason University. His research interests include wireless sensor networks, internet of things, industrial control networks, and more recently blockchain networks.

Robert Simon (M'05) received the B.S. degree in history and political science from the University of Rochester, Rochester, NY, and the Ph.D. degree in computer science from the University of Pittsburgh, Pittsburgh, PA. He is a Professor of Computer Science at George Mason University, Fairfax, VA. His research has been supported by a number of agencies, including NSF, DARPA, the U.S. Department of Defense and private industry. His research interests include embedded systems, wireless and mobile computing, distributed systems and performance modeling and analysis, and distributed computing. He has published over 110 peer-reviewed journal and conference papers on these topics, and has received 6 best paper awards.

Hakan Aydin (M'08) received the B.S and M.S degrees in control and computer engineering from Istanbul Technical University, Istanbul, Turkey, and the Ph.D. degree in computer science from the University of Pittsburgh, PA. He is currently a Professor of Computer Science at George Mason University, Fairfax, VA. He has served on the program committees of several real-time and embedded systems related conferences and workshops. His research interests include real-time embedded systems, low-power computing, and fault tolerance. Dr. Aydin received the U.S National Science Foundation (NSF) Faculty Early Career Development (CAREER) Award in 2006. He was the Technical Program Committee Chair of the 2011 IEEE Real-Time Technology and the Applications Symposium (RTAS1).