Predictable MAC-level Performance in Low-power Wireless under Interference

Mathieu Michel
Computer Science Department
University of Mons, Belgium
mathieu.michel@umons.ac.be

Thiemo Voigt
Uppsala University and SICS
Swedish ICT
thiemo@sics.se

Luca Mottola
Politecnico di Milano and SICS
Swedish ICT
luca@sics.se

Nicolas Tsiftes
SICS Swedish ICT
nvt@sics.se

Bruno Quoitin
Computer Science Department
University of Mons, Belgium
bruno.quoitin@umons.ac.be

Abstract

Predictable performance is key for many WSN applications. Recent efforts use models of the environment, the employed hardware, and protocols to predict network performance. Towards this end, we present an intentionally simple model of ContikiMAC, Contiki's default MAC layer, targeting worst-case bounds for packet delivery rate and latency. Our experiments reveal problems in the performance of ContikiMAC, which make the protocol perform much worse than predicted, and hence prohibit predictable performance with the current ContikiMAC implementation. We show that the reason for this performance degradation is that ContikiMAC loses phase-lock. To solve this problem, we add fine-grained timing information into the acknowledgment packets. We show that this mechanism solves these problems and enables predictable performance with ContikiMAC even under high external interference.

1 Introduction

Born as stand-alone tools to harvest fine-grained environmental data [3, 24, 34, 38], Wireless Sensor Networks (WSNs) are growing into the communication substrate of cyber-physical systems employed in safety-critical and high dependability applications [19]. Examples are WSN-based systems installed in factories [29], transportation [11], and healthcare [14].

Predictability. In the settings above, *predictable* performance is key. Predictability here intuitively indicates the ability to accurately understand the system's performance before it is deployed, or to forecast its evolution while it is running [2]. This information allows one to precisely dimension systems and installations, and gives users a sound foundation to rely on stated performance guarantees.

Wireless networking determines the system's performance to a large extent in the above applications. To achieve predictability, the role of protocol models thus becomes fundamental. Accurately modeling WSN protocols, however, is long known as a challenge [39]. Part of the reason for this is the intimate relation between the WSN operation and the surrounding environment, which greatly affects low-power wireless communications. Among the existing environmental factors, *external radio interference* [9, 23] is arguably one of the most significant, because of the fragile nature of low-power wireless transmissions. External interference complicates both the modeling of protocols and their operation.

The case of ContikiMAC. In this paper, we consider a specific instance of the issues above that revolves around the ContikiMAC protocol [16], the default MAC protocol in the Contiki operating system [17].

ContikiMAC represents the state of the art in many respects. As further described in Section 2, ContikiMAC employs a sender-initiated mechanism to achieve low-power operation. Receivers periodically probe the channel in search of potential transmissions, whereas senders continuously transmit a packet until the transmission overlaps with one of the receiver's channel checks. This allows the latter to align with the next packet transmission, which conveys the actual data. In ContikiMAC, this mechanism is combined with a "phase-lock" technique: the sender learns the receivers' channel check patterns, so as to minimize the number of useless packet transmissions waiting for the receiver to wake up.

As illustrated in Section 3, we first devise a performance model for ContikiMAC that, without taking external interference explicitly into account, becomes instrumental to understand the protocol's behavior in the latter setting. Next, we discuss the applicability of the model under external interference, eventually recognizing that *i)* as long as some key model assumptions continue to hold, the model still provides worst-case performance indications on communication latency and packet reliability, *ii)* the root cause for the modeling assumptions not to apply is the case when ContikiMAC

International Conference on Embedded Wireless Systems and Networks (EWSN) 2016
15–17 February, Graz, Austria

ISBN: 978-0-9949886-0-7

loses phase-lock with the intended receiver.

After experimentally verifying case *ii)* above with ad-hoc experiments, in Section 4 we present a simple, yet effective modification to ContikiMAC's operation that appreciably ameliorates the risk of losing phase-lock. Our modification increases the accuracy of the information employed by ContikiMAC to learn a receiver's channel check patterns. As this information becomes more robust, the chances that a sender loses phase-lock with the intended receivers diminishes even under interference. To this end, we switch the handling of packet acknowledgments from the radio chip to the software layers, gaining increased control on them.

The net result of these modifications is that ContikiMAC does not lose phase-lock, and thus its performance becomes more resilient to external interference. Section 5 provides experimental evidence of these claims. Our results show that our new version of ContikiMAC is able to completely avoid the phase-lock losses encountered by the default version. Our results also show that the model provides worst-case performance predictions when ContikiMAC does not lose phase-lock.

We end the paper by surveying related work in Section 6, and with brief concluding remarks in Section 7.

2 ContikiMAC

ContikiMAC [16] is an asynchronous duty-cycling protocol. A duty-cycling protocol aims to reduce energy consumption by forcing nodes to switch their radio transceiver between short *listen* periods and long *sleep* periods. ContikiMAC is asynchronous because nodes do not synchronize their *listen* periods.

When transmitting, a node repeatedly sends the full data packet until either an ACK is received or the transmission times out after a duration equal to a wake-up interval. The data packets that are part of this transmission are commonly called *strobes*. In other sender-initiated MAC protocols such as X-MAC [13] the strobes are small packets that contain the identifier of the receiver. The packet destination field in ContikiMAC's strobes allows nodes to go back to sleep early when receiving a packet not addressed to them.

In order to receive a packet, a node running ContikiMAC wakes up periodically for a short time to check the medium. When waking up, it performs two successive Clear Channel Assessments (CCA) to determine whether there is an incoming transmission. If at least one of the CCAs succeeds (clear channel), the node goes back to sleep immediately. If the CCAs fail (busy channel), it stays awake to receive an incoming packet. A certain procedure, called *fast sleep optimization*, determines if the CCA failure was caused by noise rather than an incoming frame. This procedure is based on the ContikiMAC timing constraints. Noise is considered in the three following cases:

1. The radio activity period is longer than the maximum packet size.
2. The radio activity period is followed by a silence period that is longer than the interval between two strobe transmissions.
3. The radio activity period is followed by a silence period with a correct length but no packet start delimiter can be detected right after the silence period.

In addition, a mechanism called *phase-lock* allows senders to learn the wake-up schedule of a neighbor, as depicted in Fig. 1. This mechanism, first introduced in

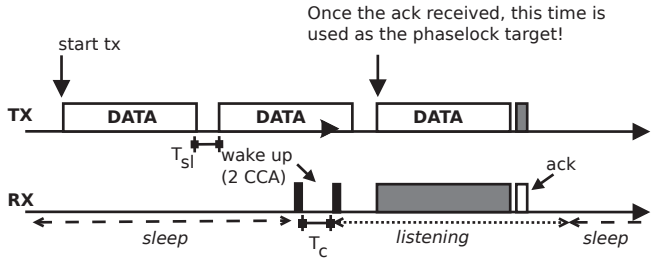

Figure 1. When an ACK is received, the sender can estimate the wake-up schedule of the receiver.

WiseMAC [18], allows a node to make use of its knowledge of the receiver's wake-up schedule to decrease the number of strobes required to achieve a transmission (Fig. 2). Once

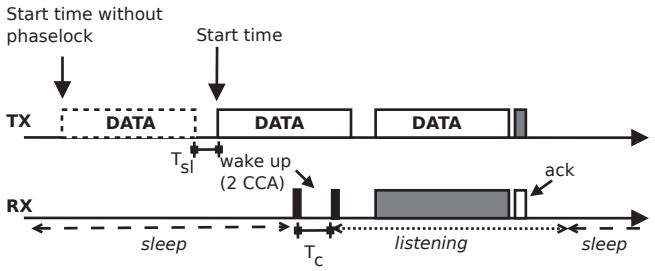

Figure 2. After the neighbour schedule is learned, the phase-lock mechanism reduces the length of the transmission.

the phase of the destination is learned, the sender can target a time close to the destination's wake-up. When the sender knows the neighbour's wake-up time and hence phase-lock is established, on average two data frames are sufficient: one strobe to wake-up the receiver and one strobe that the destination successfully receives. A side benefit of phase-lock is a reduction in channel utilization, which decreases the risk of collisions [25].

3 A Performance Model for ContikiMAC

In this section, we present our ContikiMAC model. We start by presenting a model that does not consider interference. Then we discuss the implications of interference. As discussed earlier, our model targets lower bounds. We focus on packet reception rate and latency since King et al. have already presented a ContikiMAC model for lifetime estimation [22].

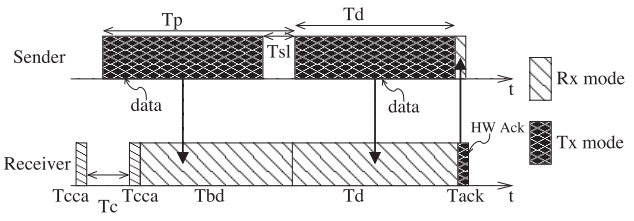

Figure 3. Unicast transmission in ContikiMAC.

3.1 ContikiMAC model without interference

Figure 3 shows a successful unicast transmission in ContikiMAC [16]. Nodes that expect to receive a packet wake up periodically. The time interval between wake-ups is denoted by T_{off}, i.e., T_{off} denotes the duty-cycle period. Each wake-up consists of two fast CCA checks of duration T_{cca}, with a short sleeping time (T_c) in between these checks unless the first check is already successful. Contrary to X-MAC, which transmits short strobe packets, ContikiMAC transmitters repeatedly transmit the whole packet. Once one of the receiver's CCA checks has detected a packet transmission, the receiver stays awake until it receives the next transmitted packet. We define T_{bd}, as the time from the CCA-based detection of a packet until the time the receiver starts receiving the packet. In the worst case, only the second CCA check is successful and happens in the beginning of the sender's packet that wakes up the receiver. Hence, we get $T_{bd} = 2 \cdot T_{cca} + T_d$ where T_{cca} is the time to perform a channel check and T_d is the time for transmitting or receiving a data packet. Note that T_{cca} is very short: 0.192 ms for the CC2420 radio [16].

As described in the previous section, a node that wishes to send a packet repeatedly transmits the data packet. Before transmitting, the sender performs a CCA check, and refrains from sending if the medium is busy.

ContikiMAC uses a phase-lock mechanism [16], where nodes keep track of their neighbors' wake-up time and start transmitting data packets briefly before the expected wake-up time of the intended receiver. We call the time between two transmissions of the same data packet T_{sl}. T_{sl} is short (0.4 ms in the current ContikiMAC implementation for the Tmote Sky). Note that it must be shorter than T_c, the time between the two consecutive CCA checks of the receiver, to prevent that both CCA checks fall into T_{sl}.

Once the receiver has successfully received the packet it sends an acknowledgment. We define T_{ack} as the transmission time of a data acknowledgement (d-ack).

In the X-MAC implementation, pTUNES [40] considers several variables that are adjustable, including T_{on}, T_{off}, and N, the maximum number of retransmissions. T_{on} is the time the receiver is on to listen to data strobes but ContikiMAC uses CCA checks instead. In ContikiMAC only N and T_{off} have a direct implication on the energy consumption for low data rates. Note that our model considers the steady case where the wake-up phase of the receiver has been learned. The current ContikiMAC implementation re-learns the wake-up phase only if no ACK has been received for 16 tries or for 30 seconds.

Per-hop reliability. In this paragraph we determine $p_{s,l}$, that is, the probability that a single unicast transmission from node n to its parent m succeeds. For this, it is required that node m can correctly detect the presence of a frame transmission, and successfully receive, and acknowledge it, afterwards. We define p_{CCA} as the probability that one of the performed CCA checks succeeds. Further, we define p_{cs} as the probability that the sender finds the radio medium idle, that is, there are no ongoing transmissions and no interference. The data packet transmission succeeds with probability p_l, the ACK with probability p_a. These probabilities depend on several factors such as packet size and environment, for example, interference. Based on the above consideration, and assuming that the events of data and d-ack packet transmission success are uncorrelated, we have:

$$p_{s,l} = p_{cs} \cdot p_{CCA} \cdot P_l, \tag{1}$$

where P_l denotes the probability of successful frame transmission (p_l) and acknowledgment (p_a),

$$P_l = p_l \cdot p_a. \tag{2}$$

In ContikiMAC, the transmitter keeps sending the data frame until it is successfully acknowledged or a maximum of additional frames probes have been sent. The number of additional frame probes is limited by a time threshold T_m. This corresponds to an additional number of frame strobes, $N_m = \lfloor \frac{T_m}{T_d} - 1 \rfloor$. Based on this model, the probability of successful link transmission during a single attempt will be:

$$P_l = \left[(p_l \cdot p_a) + \sum_{k=1}^{N_m} (1 - p_l)^k \cdot (p_l \cdot p_a) \right]. \tag{3}$$

Per-hop latency. In this paragraph we determine $T_{ftx,l}$, and $T_{stx,l}$, the expected times spent for failed and successful one-hop transmissions under ContikiMAC. A link transmission attempt may fail on four cases: *i*) the sender does not transmit as it senses the medium as busy, or *ii*) the receiver's CCA trials fail to wake-up the node, or *iii*) the frame transmission fails, or *iv*) the acknowledgment fails.

The latency for each transmission attempt has a fixed part, which includes the data and d-ack packet durations, T_d, and T_{ack}, respectively.

For a successfully received packet, the latency includes, additionally, a delay term, T_w, that reflects the time between the packet reception from the upper layer and the first transmission attempt. This delay appears due to the considered phase-lock mechanism of ContikiMAC. This time delay appears, additionally, before each re-transmission attempt[1]. As we consider the worst case, we assume that a new packet from the upper layer arrives just after the time when the receiver has woken up and hence the transmission must be delayed for almost T_{off}. Hence, the expected delay becomes

$$T_w = T_{off} + 2T_{cca} + T_c + T_x \tag{4}$$

In the current ContikiMAC implementation, there is a delay in form of a number of additional CCA checks a sender per-

[1]However, if back-off time is a multiple of the channel check interval, subsequent transmission attempts have correlated success rates.

forms before transmitting the first strobe. We call this delay T_x.

In case of link transmission failure, the transmitter backs-off for T_b before retransmitting the frame. Assuming that ContikiMAC would allow only the transmission of one data packet (after the first packet that wakes up the receiver) per wake-up cycle, the expressions for the expected per-hop latency become:

$$T_{ftx,l} = \overline{T}_w + T_d + T_{ack} + T_b + T_{bd} \quad (5)$$

$$T_{stx,l} = \overline{T}_w + T_d + T_{ack} + T_{bd} \quad (6)$$

Since ContikiMAC allows a maximum of $N_m + 1$ data frame strobes per wake-up cycle, we replace $T_d + T_{ack}$ in (5), i.e., the one-hop latency for a failed transmission, with $(T_d + T_{sl}) \cdot (N_m + 1)$. Similarly, in (6) we replace $T_d + T_{ack}$ with

$$\sum_{k=0}^{N_m+1} (T_d + T_{ack})(1 - (p_l \cdot p_a))^k (p_l \cdot p_a). \quad (7)$$

The actual value of T_b, i.e., the back-off latency, depends on the actual retransmission scheme of the link-layer protocol module. Here, we give the expressions for T_b, when considering the CSMA-based retransmission scheme in Contiki. In Contiki, the actual value of the back-off delay grows exponentially with the number of retransmissions. In general, it holds,

$$T_b = \sum_{i=1}^{N} T_{b|i} \cdot Pr\{\text{retry occurs after the i-th trial}\}, \quad (8)$$

where N denotes the maximum number of packet retransmissions. The expected back-off latencies, $T_{b|i}$ are given by the MAC retry scheme configuration. We compute the probabilities in (8) based on the relative frequencies of the retry occurrences at each retransmission *index*. We consider only packets requiring retransmissions, otherwise T_b is zero. The rate of packets requiring exactly k retries is

$$Pr\{\text{exactly } k \text{ retries}\} \triangleq P^{(k)} = (1 - p_{s,l})^{k-1} \cdot p_{s,l} \quad \forall k = 1, \ldots, N-1, \quad (9)$$

while $P^{(N)} = (1 - p_{s,l})^{N-1}$, as we consider both eventually transmitted and dropped packets. Consequently, the total rate of retry occurrences after the i-th try will be:

$$Pr\{\text{retry occurs after the i-th trial}\} = \frac{\sum_{j=i}^{N} P^{(j)}}{\sum_{k=1}^{N} k \cdot P^{(k)}}. \quad (10)$$

3.2 ContikiMAC model under interference

External interference has a high impact on several parameters of the ContikiMAC model: the probability of positive CCA checks at both the receiver (p_{cca}) and the sender (p_{cs}) as well as the probability p_l. The latter describes the probability that a receiver having its radio turned on will successfully receive a transmitted packet.

During interference, p_{cca} actually increases while p_l decreases as the sender's packet might be corrupted during transmission. Note that the CCA check may also be a false positive, meaning that the positive CCA check was caused by interference, which makes p_{cca} increase. An increase of the latter only has a negative impact on lifetime, but not on the per-hop reliability and per-hop latency, which are the two metrics we consider in this paper. p_{cs} also decreases under interference.

Independence of Data packet and ACK transmissions.

When computing the probability of a successful frame transmission and acknowledgment using Equation 2, we assume independence of the probabilities of successful transmissions of data packets and acknowledgments, as in pTUNES, i.e., $P_l = p_l \cdot p_a$. When packet loss is caused by interference, this assumption might not hold and we would need to change this equation to $P_l = p_l \cdot (p_a|p_l)$. The purpose of our modeling activity is to provide a lower bound. Hence, if $p_l \cdot p_a \leq p_l \cdot (p_a|p_l)$, the assumption of independence always constitutes a lower bound. From $p_l \cdot p_a \leq p_l \cdot (p_a|p_l)$ follows $p_a \leq (p_a|p_l)$. Therefore, in order to verify that we are modeling a lower bound by assuming independence, we have taken existing traces that are used in the COOJA simulator to simulate realistic radio interference [7], and shown that in all of them $p_a \leq (p_a|p_l)$. There are four such traces: one with Bluetooth interference, which leads to low packet loss around 10%, and three with WiFi interference, which have higher packet loss. The WiFi activity is caused by radio streaming, a file transfer, and video streaming. For all these traces $p_a \leq (p_a|p_l)$ is valid, which verifies that the assumption of independence does indeed present the lower bound.

Initial Results: Loss of Phase-Lock.

We performed preliminary experiments to validate our model. In these experiments we discarded packets above the MAC layer, that is, before they were handed to the application. This gave us fine-grained control over many variables such as p_{cs}. During these experiments it turned out that our model was quite accurate when we discarded packets above the MAC layer. Our results changed, however, significantly on real hardware even though we carefully controlled interference to only affect the receiver which also gave us control over, *e.g.*, p_{cs}. When we investigated the problem closer, we realized that ContikiMAC lost phase-lock very often, in particular when exposed to high interference. ContikiMAC's loss of phase-lock was also visible in COOJA's timeline [31]. Section 5 will show the negative impact of losing phase-lock on our performance metrics.

3.3 Discussion

We have modeled ContikiMAC on a single link. Our model, however, fits well into existing frameworks such as pTUNES: a framework for runtime adaptation of low-power MAC protocol parameters developed by Zimmerling et al. [40]. The goal of pTUNES is to adapt and optimize the MAC parameters towards application-level metrics specified by the user. Towards this end, pTUNES breaks up the modeling of low-power MAC protocols into three layers. The upper layer defines a set of application-level metrics (Reliability R, Latency L, Lifetime T) as functions of link and node-specific metrics (R_l, L_l, T_n). The middle layer expresses these metrics in a protocol-independent manner. It also provides the entry point for the modeling of a concrete MAC protocol. This modeling represents the lowest layer.

pTunes has provided MAC layer models for X-MAC [13] and LPP [26] . In contrast to the sender-initiated MAC protocols X-MAC and ContikiMAC, LPP is a receiver-initiated MAC protocol.

In the pTunes framework, ContikiMAC could be added as an additional MAC layer. Our model, however, provides lower bounds rather than average performance which pTunes uses for run-time adaptation. Nevertheless, we could use the pTunes framework to extend our model from individual links to a whole network since the choice of average or worst-case performance only has an impact on the model of the MAC layer. pTunes middle layer, for example, takes the per-hop reliabilities to compute the reliability of a path and the per-hop latencies to compute the latency of a path. Hence, for the middle layer it is not relevant if the MAC layer model targets worst-case or average performance.

One complication is that our model's most important input is the success probability of a single data packet over a link to a certain receiver. As Brown et al. note, however, "if interference is measured in one place it is obviously not guaranteed that interference measured at a different place in the same area is similar" [12]. Since we target worst-case performance one possibility is to make measurements at the location with maximum interference rather than using an average over all links that are expected to be part of a deployment. This could be the location which is expected to be most exposed to WiFi interference by having the maximum number of 802.11 base stations in its vicinity.

The goal of our model is to establish a lower bound using a simple but efficient model that does not require a lot of measurements to acquire the input data. In that respect, we know that by using p_l, we will significantly underestimate the probability of a successful data packet transmission since the probability that the destination successfully receives data packet increases when the sender has performed a CCA check before the transmission. In ContikiMAC, however, the sender may transmit a packet train of up to five data packets after a CCA check. Among those, the first is only used to wake up the receiver.

Table 1. Success probability based on position in packet train after successful CCA check (Interference level 33%). For a packet without CCA check, p_l is 75%.

Train Pos.	1	2	3	4	5
Prob. (%)	99	94.7	89.3	83.3	79.6

We have performed measurements that confirm two aspects: *(i)* the probability of a successful transmission decreases when the data packet is farther back in the packet train; *(ii)* p_l indeed underestimates the probability of a successful data packet transmission. In our experiments, the fifth packet after a CCA check has a probability similar to p_l which means that p_l is a lower bound. Table 1 shows an example with an interference level of 33%. In this case, the probability of a successful transmission p_l, i.e., a transmission without a CCA check is slightly lower than the success probability of the fifth packet after a CCA check. The table also shows that as expected, the probability of a successful transmission decreases with the position in the packet train.

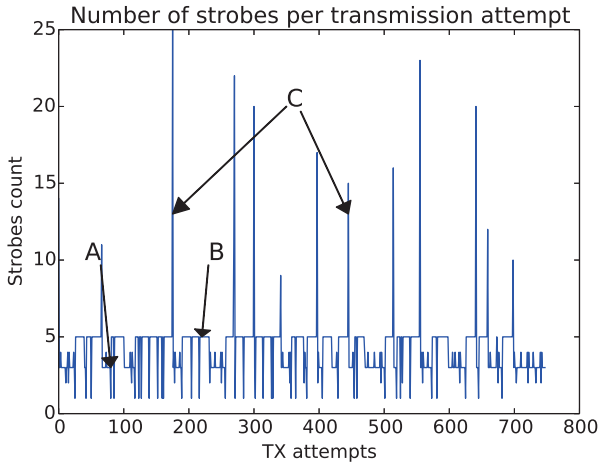

Figure 4. Number of strobes used for each transmission attempt with the original design of ContikiMAC and an interference level of 20%.

While one could of course strive for more complex and accurate input data, we decided to opt for a simpler model. Besides concretely facilitating the implementation of the model into different optimization frameworks [32, 40], model simplicity plays in favor of its use at run-time [5]. In turn, the ability to leverage model-based reasoning while the system executes enables sophisticated adaptation functionality [37]. Because of the unpredictable environment dynamics expected in the target deployment settings, these functionality are expected to become an asset as low-power wireless systems are increasingly deployed [36].

4 Maintaining Phase-lock

If ContikiMAC loses phase-lock, the performance model above stops applying. For example, not one, but multiple packet transmissions are required until the receiver eventually aligns with the sender. The inaccuracy of the phase-lock mechanism is known in the general case [25], yet external interference bears a great influence on it.

Problem. As the acknowledgments normally used to learn the receiver's phase are lost because of interference, senders are left with stale information about a receiver's wake-up patterns. In the original ContikiMAC design, this eventually leads to a situation whereby the sender gives up using phase-lock information.

Figure 4 provides experimental evidence of this behavior, showing the number of packet strobes for each transmission attempt in a situation with a moderate degree of interference[2]. The number of strobes should remain close to two if reliable phase information are available at the sender, as in point **A** of the chart. The transmissions needing three strobes can be explained by the use of the default ContikiMAC guard time. However, the plot indicates that often this information turns out unsound, and ContikiMAC first increases the number of packet strobes up to five, as in point **B**, trying to re-gain

[2]We discuss in Section 5 how we can quantify interference levels in the experiments and how we ensure their repeatability.

phase information. Eventually, it completely gives up using phase information, and resets the receiver's state information in the neighbor table. This entails that subsequent information see an even further increase in the maximum number of packet strobes, like the two nodes were never in touch before, as in point **C**.

Solution. We improve the accuracy of phase information, inherently increasing their robustness, by augmenting the acknowledgment packets with additional time information. The acknowledgment now carries a field including the time interval between the effective wake up of the receiver and the start of the reception of the data packet. Subtracting this value from the time when the packet is sent gives a sender a more accurate input to estimate the receiver's wake-up schedule. Compared to the original design, this also allows one to reduce the guard times compared to the default ContikiMAC, ultimately improving energy efficiency.

To make this modification possible, we switch from using hardware acknowledgments in ContikiMAC to software ones, which allows us to insert arbitrary information in acknowledgment packets. In principle, this loses efficiency. However, existing studies carried out in settings similar to ours indicate that the overhead is quite limited [1]. In the following, we refer to this implementation as *soft-ACK*.

5 Evaluation

We conduct an experimental evaluation of our model and the phase-lock maintenance mechanism (soft-ACK) for ContikiMAC. Firstly, we investigate whether our soft-ACK mechanism in ContikiMAC avoids the loss of phase-lock even under interference. Secondly, we examine whether the model can provide worst-case performance predictions on communication latency and packet reliability under interference. Thirdly, we evaluate whether our soft-ACK mechanism can improve the accuracy of the phase-lock information, and thereby increase the predictability of ContikiMAC's performance.

5.1 Experimental Setup

Our testbed consists of a sender-receiver couple and an interferer mote that we place in an office environment close to each other ($< 1m$). All threes nodes are Zolertia Z1 platforms that feature a CC2420 radio and run Contiki 2.7. Each experiment lasts for 30 minutes during which we transmit around 900 packets. We use IEEE 802.15.4 channel 26 since it does not overlap with WiFi [27]. The packet size is 127 bytes, which is the maximum transmission unit in IEEE 802.15.4. We have chosen this packet size as it represents the worst case since long packets are more likely to be corrupted than short ones.

To create interference, we use the CC2420's test mode to generate a continuous carrier [6]. We create semi-periodic interference as described by Boano et al. [10] using a two-state process that consists of a clear channel state and an interference state. By varying the time in which the process stays in each of these states we can achieve different levels of interference. In our experiments, we stay in the interference state for on average 500 milliseconds and vary the duration in which we stay in the clear channel state. We have also performed experiments with other durations but the results were not significantly different.

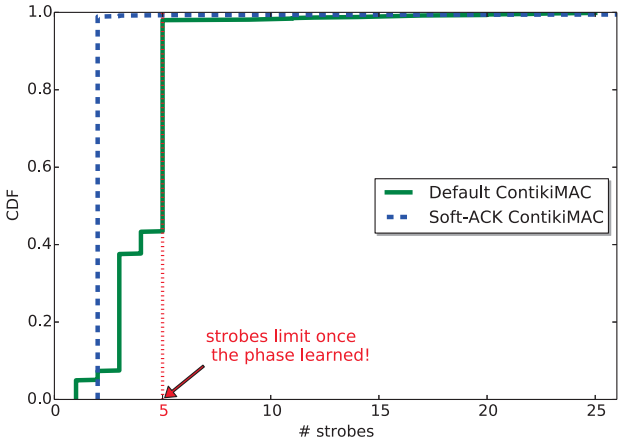

Figure 5. Number of strobes sent by transmission attempt with interference during 20% of time. Soft-ACK ContikiMAC mostly uses only two attempts demonstrating its effectiveness.

In the experiments, we vary the following factors.

1. The following **interference levels** are used: 50%, 33%, 20%, 11% and 7%. We have also run experiments without any interferer mote to have a baseline result.
2. Each experiment is made both with the **default** ContikiMAC implementation and our **soft-ACK** based version.
3. We run ContikiMAC with and without CSMA. CSMA allows retransmissions of failed transmissions, whereas running without CSMA[3] entails that a packet is dropped if the initial transmission fails.

The input to the model are the probabilities of success for two different packet sizes: packets of 127 bytes for p_l as defined in Section 3, and packets of 5 bytes for p_a, which are also defined in the same section. We measure these probabilities using the same setup as described above. Furthermore, we vary p_{cca} and assign it three different high values: 0.9, 0.95, and 0.99. We set p_{cs} to 1-(100%-interference level)/100%.

5.2 Maintaining Phase-lock

In order to compare the efficiency of our updated phase-lock mechanism, we first look at the number of strobes used per transmission attempt. A correct implementation of the phase-lock mechanism should allow the sender to achieve most of the transmissions with a maximum of two strobes [16].

Fig. 5 shows the cumulative distribution function of the number of strobes for an experiment without CSMA with interference during 20% of time. The figure shows that the number of strobes is more uniform with the soft-ACK version of ContikiMAC. Most transmissions are achieved with only two strobes: the first one wakes up the receiver and

[3]The NULLMAC layer in Contiki.

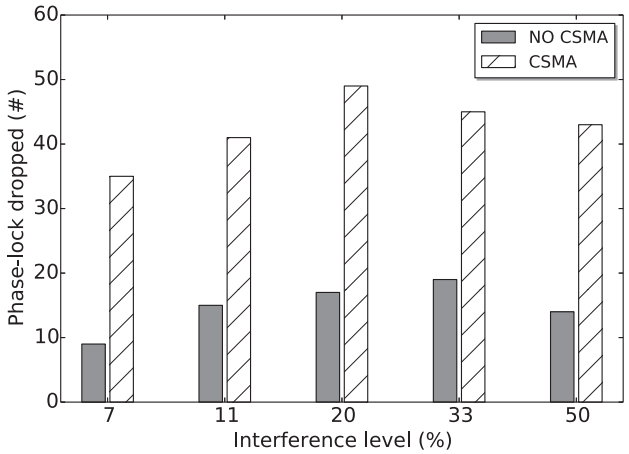

Figure 6. Number of phase-lock losses when using default ContikiMAC with and without CSMA.

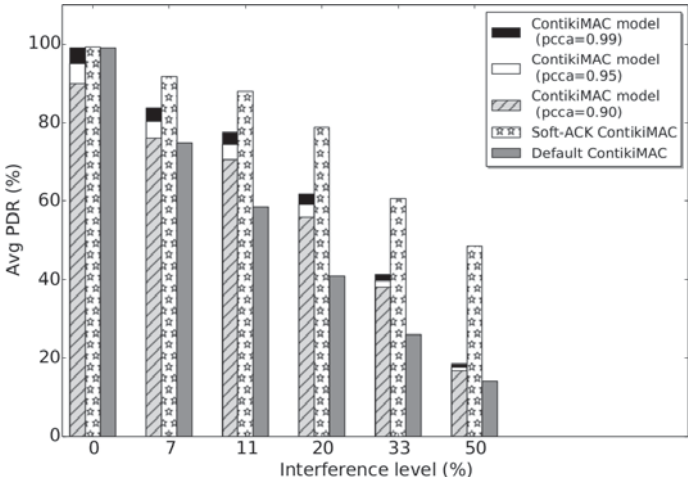

Figure 7. PDR results without CSMA. Soft-ACK ContikiMAC increases performance by avoiding loss of phase-lock. Our model establishes a lower bound.

the second one is the data packet acknowledged by the destination after reception. The few transmissions with more strobes are the initial transmission when the phase is not yet locked (1/604) and some due to interference triggered after the CCA check (6/604).

As explained in Sec. 4, the occurrence of five successive strobe transmissions shows that the default ContikiMAC is not able to maintain phase-lock. This is confirmed by Fig. 6 that shows the number of phase-locks lost and then dropped by the default implementation of ContikiMAC. In contrast, the soft-ACK ContikiMAC version did not encounter any phase-lock losses once the sender had learned the receiver's wake-up schedule.

5.3 Packet Reception and Latency

In the next series of experiments, we show that our model indeed presents a lower bound for the performance of ContikiMAC under interference when ContikiMAC does not lose phase-lock.

5.3.1 Packet Delivery Ratio

Fig. 7 and Fig. 8 summarize the packet delivery ratio (PDR) we get without CSMA and with CSMA compared to the value provided by the model. Each result presents the average of five runs. We do not include the variance in the graphs since it is very low.

The figures show that the soft-ACK version of ContikiMAC without CSMA is able to achieve a higher PDR than ContikiMAC, regardless of the interference level. This result can be explained by the fact that, unlike the default ContikiMAC, the soft-ACK version does not face phase-lock losses and therefore can keep the PDR steady. The figures show that when we set p_{cca} to 0.99 the model is closer to the actual values than with lower values for p_{cca}. This is expected since ContikiMAC for platforms with the CC2420 radio are the most mature and well-tuned ContikiMAC implementations: the timing of the receiver's CCA checks is well optimized and hence these CCA checks are likely to be successful. Furthermore, as discussed in Section 3.2 the probability that the CCA checks of the receiver fail decreases with interference.

The role of Contiki's CSMA layer is to reschedule failed transmissions. The occurrences of phase-lock losses involves failed transmissions, which need to be rescheduled. Those retransmissions are bounded by a maximum number of three failed retransmissions, afterwards the packet is dropped. Compared to the default version of ContikiMAC, the lack of phase-lock losses within the soft-ACK version involves fewer retransmissions. For example, with an interference level of 33% this number is four times lower for soft-ACK ContikiMAC which reduces the number of packet losses and explains the PDR differences between the two protocol versions.

The figure also shows that the model reaches its objective to provide a lower bound for PDR regardless of the interference level. Nevertheless, there is a difference between the lower bound provided by the model and the testbed values. As discussed in Section 3.3 the reason for this discrepancy is the conservative input values we are using for the model: we know that p_l underestimates the probability of a successful data packet transmission since it does not consider the sender's CCA check that ensures that a packet is only transmitted when the medium is idle.

5.3.2 Latency

Fig. 9 provides the latency results with CSMA with $p_{cca} = 0.95$. We do not show the results without CSMA since most of the latency stems from the back-off delay for retransmissions. As discussed in Section 3, the actual value of the back-off delay grows exponentially with the number of packet retransmissions. Therefore, a low PDR leads to more retransmissions and hence higher latency which explains the increased latency of the default ContikiMAC compared to the soft-ACK ContikiMAC. Due frequent phase-lock losses, the default ContikiMAC experiences more retransmissions in particular at high interference levels. The figure also shows that our model provides an upper bound

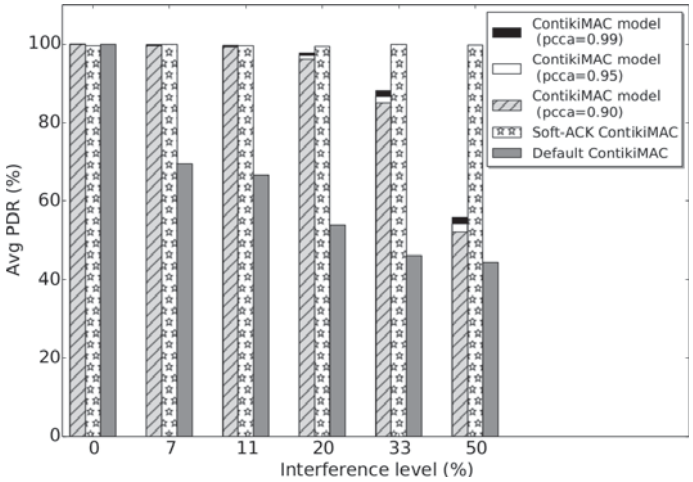

Figure 8. PDR results with CSMA. Soft-ACK ContikiMAC increases performance and our model establishes a lower bound even under high interference.

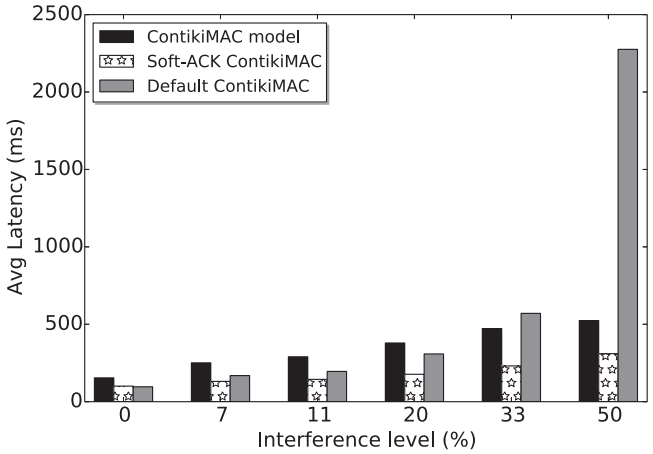

Figure 9. Latency results with CSMA. Avoiding loss of phase-lock has a huge impact on latency. Our model provides an upper bound (worst case) for soft-ACK ContikiMAC.

(worst-case) on the latency for all interference levels. The difference to the testbed results stems from the model's lower PDR compared to the soft-ACK ContikiMAC.

5.4 Energy

While we do not provide a model for the energy consumption, we expect that not losing phase-lock reduces the energy consumption since phase-lock avoids unnecessary strobing. Towards this end, we measure the radio duty cycle averaged over sender and receiver. Our results show that overall, interference has a large negative impact on the energy consumption. When maintaining phase-lock, this negative impact decreases. For example, for an interference level of 10%, maintaining phase-lock reduces the duty cycle from 10.3% to only 5.7% in our experiments. When there is no interference, ContikiMAC's duty cycle is only slightly above 1%.

6 Related Work

Several studies have tackled the challenge of predicting the performance of sensor networks under interference. Oppermann et al. [30] provide a framework to automate the parameter settings of protocols to achieve dependability in Internet of Things applications. The framework uses models of the environment, the target hardware platform, and the protocols to predict the performance when challenged by temperature variations and interference. They demonstrate the effectiveness of their approach using TempMAC [8]. TempMAC is a temperature-aware extension of ContikiMAC that dynamically adapts the CCA threshold in response to temperature variations. Our work is orthogonal to that of Oppermann et al. in that our model can be plugged into their framework.

Brown et al. [12] discuss how to predict the packet reception rate in a deployment area. Towards this end, they measure the probability distribution function of idle-period lengths by using fast sampling on standard sensor nodes. In our work, an estimate of this metric is the most important input parameter to our model. We estimate the packet reception rate in a more direct fashion by measuring it explicitly, but we could also use their model. King et al. [22] have presented a ContikiMAC model for lifetime estimation in interference environments. Our work complements their study in that we present ContikiMAC models for packet reception rate and latency under interference.

There are other approaches to establish predictable performance under interference prior to deployment. Pöttner et al. [33] derive schedules for time-critical data delivery in multi-hop networks based on measured data. Their scheme targets TDMA-based MAC layers, whereas ContikiMAC is contention-based. Noda et al. [28] present a channel quality metric that quantifies the spectrum usage. Based on this metric they optimize packet size and the application of erasure codes to improve reliability, energy consumption and throughput.

PTUNES [40] aims at run-time adaptation of low-power MAC protocol parameters. As discussed in Section 3.3 our model can be seen as an addition to the existing MAC layers that PTUNES models. In contrast to PTUNES, we model worst-case rather than average performance.

Recently, Despaux et al. [15] provided a Markov chain model for ContikiMAC that is "obtained through the analysis and instrumentation of its reference implementation". They use their model to estimate the end-to-end delay in multi-hop transmissions with static routing. By contrast, our model includes both delay and packet reception rate, and also considers interference. Others have provided accurate models for the Distributed Coordination Function (DCF) mode of 802.11. Bianchi [4] modeled the DCF mode of 802.11 as a discrete Markov process, where the back-off and retransmission mechanisms are represented as discrete states. Based on this model, Garetto and Chiasserini [20] developed a simpler Markov process by merging back-off states. Our model targets worst-case performance under interference.

Orthogonal to our work are those that study other important factors that have an impact on the performance of a deployment. Among those is the selection of the least inter-

fered channels. Several approaches exist, such as the one by Iyer et al. [21]. They are able to identify multiple channel activities that would impair a sensor network's communication on a certain channel. Shariatmadari et al. [35] share the same goal but build a model that ranks channels based on the estimated packet delivery rate which is computed using signal level, interference and noise characteristics.

7 Conclusions

Wireless sensor networks are increasingly being deployed as an underlying communication infrastructure for cyber-physical systems that require high dependability. In such settings, predictable performance is key. Predicting performance is difficult, however, since the performance is affected by external factors such as external radio interference, which complicates protocol modeling. In this paper, we have devised models for the packet reception rate and latency of ContikiMAC, Contiki's default MAC layer. We have shown that under interference the model does not hold when ContikiMAC loses phase-lock, as the current implementation sometimes does. By adding fine-grained timing information into ContikiMAC's acknowledgments, we avoid the loss of phase-lock. Our experiments on real hardware have shown that our modifications to ContikiMAC indeed avoid loss of phase-lock even under strong interference. Our results also demonstrate that our model provides worst-case performance predictions regarding communication latency and packet reliability under interference when using our improved version of ContikiMAC.

Acknowledgments

We would like to thank Ioannis Glaropoulos for his contributions to the model. This work was partly supported by the distributed environment Ecare@Home funded by the Swedish Knowledge Foundation 2015-2019. It was also partially supported by the European Commission with contract INFSO-ICT- 317826 (RELYonIT) and by the Italian Ministry for University and Research through Cluster Projects "Zero-energy Buildings in Smart Urban Districts" (EEB), "ICT Solutions to Support Logistics and Transport Processes" (ITS), and "Smart Living Technologies" (SHELL).

8 References

[1] M. Anwander and T. Braun. A reliable, traffic-adaptive and energy-efficient link layer for wireless sensor networks. In *Proceedings of IFIP Networking 2013 Conference*, 2013.

[2] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dependable Secur. Comput.*, 1(1), 2004.

[3] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle, et al. Permadaq: A scientific instrument for precision sensing and data recovery in environmental extremes. In *Proceedings of the International Conference on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, pages 265–276. IEEE Computer Society, 2009.

[4] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, 2000.

[5] G. Blair, N. Bencomo, and R. B. France. Models@ run. time. *Computer*, 42(10), 2009.

[6] C. A. Boano, Z. He, Y. Li, T. Voigt, M. Zuniga, and A. Willig. Controllable Radio Interference for Experimental and Testing Purposes in Wireless Sensor Networks. In *Proceedings of the 4th International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, Zurich, Switzerland, Oct. 2009.

[7] C. A. Boano, K. Römer, F. Österlind, and T. Voigt. Realistic simulation of radio interference in COOJA. In *Adjunct Proceedings of the 8^{th} European Conference on Wireless Sensor Networks (EWSN), demo session*, pages 36–37, Feb. 2011.

[8] C. A. Boano, K. Römer, and N. Tsiftes. Mitigating the adverse effects of temperature on low-power wireless protocols. In *Proceedings of the 11^{th} IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS)*, Oct. 2014.

[9] C. A. Boano, T. Voigt, C. Noda, K. Römer, and M. A. Zúñiga. JamLab: Augmenting sensornet testbeds with realistic and controlled interference generation. In *Proceedings of the 10^{th} International Conference on Information Processing in Sensor Networks (IPSN)*, pages 175–186, apr 2011.

[10] C. A. Boano, T. Voigt, N. Tsiftes, L. Mottola, K. Römer, and M. A. Zúñiga. Making sensornet mac protocols robust against interference. In *Proceedings of the 7th European conference on Wireless Sensor Networks*, EWSN'10, pages 272–288, Berlin, Heidelberg, 2010. Springer-Verlag.

[11] M. Bottero, B. Dalla Chiara, and F. P. Deflorio. Wireless sensor networks for traffic monitoring in a logistic centre. *Transportation Research Part C: Emerging Technologies*, 26:99–124, 2013.

[12] J. Brown, U. Roedig, C. Boano, and K. Roemer. Estimating packet reception rate in noisy environments. In *Proceedings of the 39th IEEE Conference on Local Computer Networks, 2014. LCN 2014.* IEEE, 2014.

[13] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Boulder, Colorado, USA, 2006.

[14] J. M. Caldeira, J. J. Rodrigues, and P. Lorenz. Toward ubiquitous mobility solutions for body sensor networks on healthcare. *IEEE Communications Magazine*, 50(5):108–115, 2012.

[15] F. Despaux, Y.-Q. Song, and A. Lahmadi. Modelling and performance analysis of wireless sensor networks using process mining techniques: Contikimac use case. In *Distributed Computing in Sensor Systems (DCOSS), 2014 IEEE International Conference on*, pages 225–232. IEEE, 2014.

[16] A. Dunkels. The ContikiMAC Radio Duty Cycling Protocol. Technical Report T2011:13, Swedish Institute of Computer Science, 2011.

[17] A. Dunkels, B. Grönvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the 1^{st} International Workshop on Embedded Networked Sensors (EmNetS)*, Tampa, FL, USA, nov 2004.

[18] A. El-Hoiydi, J.-D. Decotignie, C. C. Enz, and E. L. Roux. wiseMAC, an ultra low power MAC protocol for the wiseNET wireless sensor network. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, 2003.

[19] M. L. Fairbairn, I. Bate, J. Stankovic, et al. Improving the dependability of sensornets. In *Proceedings of Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2013.

[20] M. Garetto and C.-F. Chiasserini. Performance analysis of 802.11 wlans under sporadic traffic. In *NETWORKING 2005*. Springer, 2005.

[21] V. Iyer, F. Hermans, and T. Voigt. Detecting and avoiding multiple sources of interference in the 2.4 ghz spectrum. In *Proceedings of the European Conference on Wireless Sensor Networks (EWSN)*. Springer, 2015.

[22] A. King, J. Brown, J. Vidler, and U. Roedig. Estimating node lifetime in interference environments. In *40th IEEE Conference on Local Computer Networks, 2015*. IEEE, 2015.

[23] C. Liang, N. Priyantha, J. Liu, and A. Terzis. Surviving wi-fi interference in low power zigbee networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Zurich, Switzerland, Nov. 2010.

[24] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *First ACM Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, Atlanta, GA, USA, Sept. 2002.

[25] M. Michel and B. Quoitin. ContikiMAC vs X-MAC performance analysis. Technical report, University of Mons, 2015.

[26] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis. Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks. In *Proceedings*

of the International Conference on Information Processing in Sensor Networks (ACM/IEEE IPSN), St. Louis, Missouri, USA, 2008.

[27] R. Musaloiu-E. and A. Terzis. Minimising the Effect of WiFi Interference in 802.15.4 WSN. *IJSNet*, 3(1):43–54, Dec. 2007.

[28] C. Noda, S. Prabh, M. Alves, and T. Voigt. On packet size and error correction optimisations in low-power wireless networks. In *10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 212–220. IEEE, 2013.

[29] T. O'Donovan, J. Brown, F. Büsching, A. Cardoso, J. Cecílio, P. Furtado, P. Gil, A. Jugel, W.-B. Pöttner, U. Roedig, et al. The ginseng system for wireless monitoring and control: Design and deployment experiences. *ACM Transactions on Sensor Networks (TOSN)*, 10(1):4, 2013.

[30] F. J. Oppermann, C. A. Boano, M. A. Zúñiga, and K. Römer. Automatic protocol configuration for dependable internet of things applications. In *Proceedings of the* 10^{th} *International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*. IEEE, Oct. 2015.

[31] F. Österlind, J. Eriksson, and A. Dunkels. Cooja timeline: a power visualizer for sensor network simulation. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Zurich, Switzerland, 2010.

[32] P. Park, P. Di Marco, P. Soldati, C. Fischione, and K. H. Johansson. A generalized markov chain model for effective analysis of slotted ieee 802.15. 4. In *International Conference on Mobile Adhoc and Sensor Systems*, pages 130–139. IEEE, 2009.

[33] W.-B. Poettner, H. Seidel, J. Brown, U. Roedig, and L. Wolf. Constructing schedules for time-critical data delivery in wireless sensor networks. *ACM Transactions on Sensor Networks*, 10(3), 2014.

[34] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter. Luster: Wireless sensor network for environmental research. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Sydney, Australia, 2007.

[35] H. Shariatmadari, A. Mahmood, and R. Jantti. Channel ranking based on packet delivery ratio estimation in wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 59–64. IEEE, 2013.

[36] J. Stankovic, I. Lee, A. Mok, R. Rajkumar, et al. Opportunities and obligations for physical computing systems. *Computer*, 38(11), 2005.

[37] L. Subramanian and R. H. Katz. An architecture for building self-configurable systems. In *Mobile and Ad Hoc Networking and Computing, (MobiHOC)*, 2000.

[38] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, 2005.

[39] M. Zimmerling, F. Ferrari, L. Mottola, and L. Thiele. On modeling low-power wireless protocols based on synchronous packet transmissions. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on*, pages 546–555. IEEE, 2013.

[40] M. Zimmerling, F. Ferrari, L. Mottola, T. Voigt, and L. Thiele. ptunes: runtime parameter adaptation for low-power mac protocols. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 173–184, 2012.