Data Prediction + Synchronous Transmissions = Ultra-low Power Wireless Sensor Networks

Timofei Istomin[1], Amy L. Murphy[2], Gian Pietro Picco[1], Usman Raza[3]
[1] University of Trento, Italy {timofei.istomin, gianpietro.picco}@unitn.it
[2] Bruno Kessler Foundation, Italy murphy@fbk.eu
[3] Toshiba Research Europe Limited, UK usman.raza@toshiba-trel.com

ABSTRACT

Data prediction in wireless sensor networks replaces the commonly used (periodic) data reporting with a model, updated (infrequently) at the sink to accurately reproduce real data trends. This technique abates up to 99% of *application* messages; yet, recent work has shown it achieves "only" up to a 7x lifetime improvement when executed atop a mainstream *network* stack (e.g., CTP + BoX-MAC), as the idle listening and topology maintenance in the latter are ill-suited to the sparse traffic induced by data prediction. This paper presents a novel network stack designed for data prediction, CRYSTAL, that exploits synchronous transmissions to quickly and reliably transmit model updates when these occur (infrequently but often concurrently), and minimizes overhead during the (frequent) periods with no updates. Based on 90-node experiments in the Indriya testbed and with 7 public datasets, we show that CRYSTAL unleashes the full potential of data prediction, achieving per-mille duty cycle with perfect reliability and very small latency.

CCS Concepts

•**Networks** → **Network protocol design;**

Keywords

Synchronous transmissions; wireless sensor networks; data prediction; energy efficiency.

1. INTRODUCTION

Extensive work in wireless sensor networks (WSNs) has focused on energy savings by relying on efficient hardware and routing protocols agnostic of the data being transmitted. Instead, data prediction [15] has emerged as an application-level technique to reduce the amount of data generated and transmitted by WSN nodes. In this approach, each node constructs a mathematical model, shared by the node and the sink, to approximate future sensed data. As long as sensor readings fall within an application-defined tolerance of the value predicted by the model, no data is transmitted. When significant deviations occur, a new model is generated and sent to the sink. Data prediction is particularly effective when applied to environmental data such as light and temperature, suppressing up to 99% of message transmissions.

Recent work [27], however, has shown that this significant reduction of application messages does not lead to analogous savings when considering the lifetime of the system as a whole; e.g., when applying data prediction over a typical stack of CTP and BoX-MAC, only a 7-fold lifetime was achieved. Idle listening in the MAC layer and routing overhead prohibit further lifetime improvements.

These findings motivated us to evaluate the potential of placing synchronous transmissions, made popular by the Glossy [13] protocol, at the core of a new network stack expressly designed to support the traffic patterns data prediction induces when used in conjunction with data collection. Protocols based on synchronous transmissions neither maintain a topology nor rely on a duty-cycled MAC, yet offer low latency, high reliability, and extremely low duty cycle. While work has been done to exploit Glossy for collection of periodic data [12, 29], the traffic resulting from data prediction is aperiodic, making these approaches inapplicable.

Indeed, inspection of the traffic profiles induced by data prediction (Section 2) reveals long periods of inactivity when models accurately predict the data; occasionally, these models must be updated and are transmitted by nodes to the sink. However, the time interval between updates and the number of updates to be communicated concurrently are irregular, not known globally, and ultimately unpredictable.

These observations lead to two key requirements (Section 3) for our new stack, CRYSTAL[1]. First, when there is nothing to transmit, the *network overhead must be minimized*. Second, model updates themselves must be delivered in both a *timely and reliable manner* despite their unpredictable nature in terms of distribution over time and number of concurrent transmissions.

CRYSTAL approaches these conflicting requirements by using Glossy network flooding as a primitive to build reliable data collection with data prediction at the topmost layer. CRYSTAL inherits the aforementioned properties of Glossy to broadcast a single message, and offers a simple but effective mechanism to provide reliability of the unpredictable and possibly concurrent model transmissions arising from data prediction. The core of CRYSTAL is a periodic, flexible sequence of synchronized slots organized in pairs, providing

[1]Crystal balls are often associated with the ability to predict the future; further, a crystal is a beautiful, *glossy* object.

SenSys '16, November 14-16, 2016, Stanford, CA, USA

 ISBN 978-1-4503-4263-6/16/11…$15.00

DOI: http://dx.doi.org/10.1145/2994551.2994558

a network-wide transport protocol of sorts. In the first slot, all nodes with data to transmit send it by initiating a Glossy flood. In the second slot, it is the sink that initiates a Glossy flood, acknowledging which packet it has received, if any. Due to the capture effect [18], a property of IEEE 802.15.4 radios, the sink is highly likely to receive one of the packets even when there are multiple, concurrent transmitters. The alternate execution of these two slots is repeated continuously inside a reporting interval; a distributed termination policy allows all network nodes to determine when the transmission sequence is complete, i.e., all data has been received by the sink, and nodes can safely go to sleep.

As CRYSTAL relies on Glossy, we offer a concise primer about it (Section 4) followed by a complete CRYSTAL protocol description (Section 5). An analytical model (Section 6) provides the foundation to analyze the energy consumption of CRYSTAL. An extensive set of 90-node experiments in the Indriya testbed [8] enable us to characterize the operation of CRYSTAL (Section 7) by determining experimentally a few key parameters that determine the accuracy of the model, allowing us to identify a good configuration and experimentally validate the model itself. Finally, we close the circle by using *both* our model *and* our implementation of CRYSTAL to determine the duty cycle it can achieve on 7 publicly-available real-world datasets (Section 8). We confirm our claim that CRYSTAL *achieves per-mille duty cycle* and lower, and show experimentally that this translates into improvements up to 80x over the CTP + BoX-MAC baseline, therefore bringing low energy consumption to levels hitherto possible only via specialized hardware.

We end the paper by surveying related work (Section 9), followed by brief concluding remarks (Section 10).

2. DATA PREDICTION: NETWORK IMPLICATIONS

Data prediction enables the suppression of a remarkable number of periodic data reports, greatly reducing the need for communication in WSNs and improving significantly their lifetime. Nevertheless, the potential benefits brought by this application-level technique can be reaped only to some extent when applied to mainstream network stacks; these are designed for periodic traffic and therefore are ill-suited for the aperiodic, sparse traffic induced by data prediction. In this section we discuss qualitatively and quantitatively these issues, for a specific data prediction technique.

2.1 Derivative-Based Prediction

Several prediction techniques exist [15], with varying degrees of complexity and accuracy. In this paper, we adopt Derivative Based Prediction (DBP) [27], in which each node constructs a linear model to predict the data. The model is formed by taking a sequence of m sensor values and approximating the slope (the derivative) of the data by a line formed by two points, respectively the average of the initial and final l values in the sequence. This model is used to predict the subsequent sensor values. As long as the actual, sensed value is within a certain value tolerance of the predicted value, no data is sent. Instead, if the sensed value falls outside the value tolerance for a given time tolerance, a new model is generated from the last m sensed values and sent to the sink.

We choose DBP as it is the most recent in the literature, shown to perform equivalently or better than the state of the art. Further, it is the only one that has been evaluated on a real WSN platform and for which the interplay with the underlying network stack has been evaluated. As summarized next, this constitutes the motivation for this paper.

2.2 Data Prediction on a Staple WSN Stack

The authors of [27] report message suppression rates up to 99% w.r.t. periodic collection, based on several publicly-available datasets we also use here for comparison; they are summarized in Table 1 and discussed in Section 2.3.

Nevertheless, in the same work the authors also show that the savings on the system as a whole are not as significant when the staple WSN stack constituted by CTP [14] and BoX-MAC [21] is used. The maximum improvement is seen by exploiting the characteristics of the sparse traffic induced by data prediction inside the configuration of the underlying network stack. Specifically, the infrequent transmissions generated by model updates enable in BoX-MAC the use of a sleep interval much longer than in the periodic case. This sleep interval can be further increased if, at the same time, a much longer maximum beaconing interval is used in CTP.

Differently from [27], in this paper we use the Indriya testbed. Therefore, to establish a baseline for CRYSTAL, we apply the same experimental methodology and datasets in Indriya, validating the results of [27] in this setting. Figure 1 shows results for the INDOOR temperature dataset. The lowest duty cycle $DC = 4.778\%$ for periodic reporting is achieved with a MAC sleep interval of 500 ms, while data prediction yields the lowest $DC = 1.146\%$ (4.17x improvement) with a sleep interval of 2.5 s. The cross-layer configuration of data prediction, MAC, and routing achieves the lowest $DC = 0.743\%$ (6.4x improvement) with a sleep interval of 3 s and a maximum beaconing interval of 4000 s, instead of the default 500 s. Both data prediction configurations achieve 100% data yield, thanks to reduced contention, while the plain periodic configuration achieves 98.3%.

The sparse traffic induced by data prediction implies that energy consumption is dominated by network overhead. In the best CTP/DBP configuration, 65% of the energy is spent in idle listening, 25% in transmitting beacons for tree maintenance, and only 10% in transmitting model updates. This observation motivates the approach presented in this paper that, based on synchronous transmissions, entirely removes the need for a duty-cycling MAC and the maintenance of a routing topology.

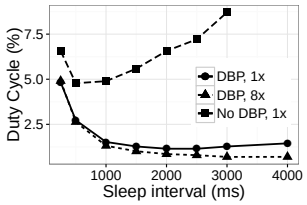


Figure 1: Exploring the best configuration for CTP, BoX-MAC, and DBP-based data prediction in Indriya.

2.3 Traffic Patterns with Data Prediction

We mentioned that data prediction induces a very sparse traffic w.r.t. periodic collection. We now quantify this statement based on the same publicly-available datasets used in [27], concisely summarized in Table 1.

Without data prediction, each WSN node reports a sample during each reporting period, hereafter called *epoch*. Epochs are not synchronized; nevertheless, if we were to discretize time based on their duration, each epoch would "see"

Table 1: Datasets characteristics.

Application & Dataset		Epoch	Nodes	Samples	Description
INDOOR	temperature	30 s	54	2,303,255	A 36-day dataset from an indoor WSN deployment in Intel Berkeley Research Lab; one of the earliest publicly available datasets, and therefore used by many papers on data prediction, e.g., [24, 30].
	humidity	30 s	54	2,303,255	
	light	30 s	54	2,303,255	
SOIL	air temperature	10 minutes	10	225,360	A 225-day dataset from the Life Under Your Feet (LUYF) project [19], in which WSN nodes are deployed in forests to study soil properties.
	soil temperature	10 minutes	4	77,904	
TUNNEL	light	30 s	40	5,414,400	The 47-day dataset used in the WSN-based closed-loop control system for road tunnel lighting described in [5].
WATER	chlorine	5 minutes	166	715,460	A dataset from a sensor network monitoring a water distribution system, simulated via the EPANET 2.0 [11] tool. Like INDOOR, this dataset is used in several previous works (e.g., [2, 25]).

Table 2: Data prediction applied to the INDOOR temperature dataset.

	updates sent in a given epoch															
	TOT	0–1	≥ 2	0	1	2	3	4	5	6	7	8	9	10	11,12	13
epoch occurrences	103K	99.8K	2.9K	84.3K	15.5K	2.2K	432	131	43	21	13	5	3	4	0	1
% over #epochs	100	97.2	2.8	82.1	15.1	2.2	0.4	0.1	0.04	0.02	0.01	0.005	0.003	0.004	0	0.001
#updates	22.3K	15.5K	6.8K	N/A	15.5K	4.4K	1.3K	524	215	126	91	40	27	40	0	13
% over #updates	100	69.4	30.6	N/A	69.4	19.9	5.8	2.3	1	0.6	0.4	0.2	0.1	0.2	0	0.1

a number of messages equal to the number of nodes, e.g., 54 in the INDOOR dataset. The message suppression achieved by data prediction in DBP dramatically changes this behavior, as shown in Figure 2. Discretizing the model updates in our traces over the epoch size shows that the vast majority of epochs contains *no* message transmission (Figure 2a); further, epochs with *at most* one message transmission are common in several datasets (Figure 2c). On the other hand, epochs with *multiple* message transmissions in the same epoch do exist, although their frequency depends on the phenomena at hand. For instance, Figure 2a shows that most datasets show a sort of "exponential decay", where the maximum number of transmissions in the same epoch is between 2 (SOIL temperature) and 19 (INDOOR light). The only exception is the WATER dataset (Figure 2b) where, due to the dynamics of the underlying physical phenomena sensed and the long epoch duration of 5 minutes, *i)* a negligible number of epochs contain at most one update, and *ii)* up to 38 message transmissions in the same epoch are observed.

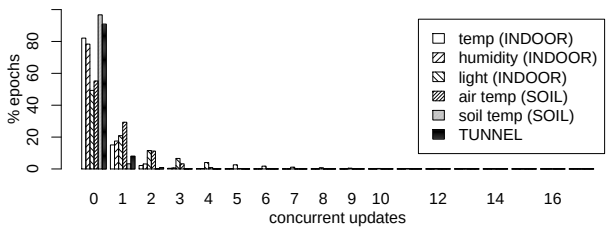
Table 2 offers a closer look at the number of epochs with multiple updates for the INDOOR temperature dataset, the one analyzed in-depth in [27]. The table also illustrates the number of updates occurring with some other in the same epoch—an important factor for protocol design, as discussed next. For instance, the table shows that although only 2.80% of the epochs see $u > 2$ updates, these "concurrent" updates are 30.57% of all those to be disseminated. Figure 2c provides a similar view for all datasets, showing the fraction of updates that are *not* isolated in an epoch. Again, the WATER dataset is the exception: almost all updates occur concurrently with at least one other update.

3. CRYSTAL: DESIGN RATIONALE

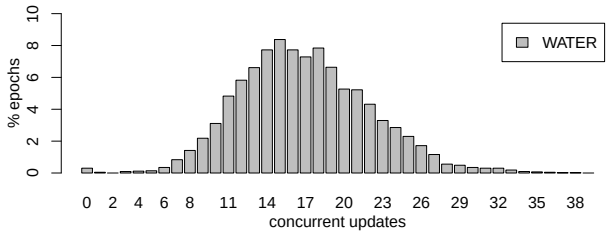
The quantitative considerations in Section 2.2–2.3 allow us to distill the **goals** that inspired the design of CRYSTAL:

Goal 1. *Minimal network overhead in the control plane.* An obvious goal of our design is to harvest the potential gains offered by the message suppression of data prediction, which drastically reduces data transmissions. As discussed in Section 2.3, this creates a traffic pattern with *no* message transmissions during the majority of reporting epochs. Section 2.2 shows this clashes with the operation of a perfectly tuned, mainstream WSN stack due to control overhead.

Goal 2. *Timely and reliable dissemination of unpredictable model updates.* We mentioned that the distribution of model updates across epochs is not known a priori; at the beginning of an epoch we cannot know if there will be several



(a) Percentage of epochs in which a given number of concurrent updates occur (all datasets except WATER).



(b) Percentage of epochs in which a given number of concurrent updates occur (only WATER).

Application & Dataset		%epochs with $u \leq 1$	%updates *not* isolated
INDOOR	temperature	97.19	30.57
	humidity	95.84	36.32
	light	70.19	84.13
SOIL	air temperature	84.51	55.52
	soil temperature	99.94	3.42
TUNNEL	light	99.00	20.77
WATER	chlorine	0.34	99.99

(c) Percentage of epochs where at most one update occurs and, dually, fraction of updates occurring with others in a given epoch.

Figure 2: Distribution of model updates over epochs, in the datasets of Table 1.

updates or none. Still, in the former case, applications demand that *all* pending updates are disseminated *within the epoch in which they were generated*. Deferring the update to a later epoch or, worse, losing an update, causes the sink to become unaware of changes in the actual data sensed at the nodes. This is exacerbated if the network is part of a control system, whose actions may be delayed or even incorrect.

Our **solution** to tackle these goals is a network stack based on synchronous transmissions that, requiring neither a MAC layer nor topology maintenance, is in line with Goal 1. The fundamental communication primitive is network-wide flooding, performed by exploiting physical properties of wireless communication (i.e., constructive interference and the capture effect [18]) yielding rapid and reliable packet dissemination, in line with Goal 2.

Nevertheless, this choice has consequences. Synchronous transmissions require all nodes to be simultaneously awake to help disseminate the flooded packet. For us, this requires a global schedule to ensure nodes are awake when an update must be disseminated. However, this schedule must consider that Goal 1 and Goal 2 pose conflicting concerns. On one hand, the desire to minimize control overhead (Goal 1) implies that *nodes should normally be awake as briefly as possible during an epoch, as in most cases no transmissions occur*. This argues for a very short schedule. At the other extreme, the need for timely and reliable dissemination of unpredictable model updates (Goal 2) demands that *nodes with a pending update have enough opportunities to transmit and recover from rare packet loss within a single epoch*. This argues for a long-enough schedule accommodating all nodes with updates, whose number or even presence is impossible to ascertain without (very expensive) global knowledge.

CRYSTAL reconciles these conflicting goals with the mechanics of synchronous transmissions by essentially providing a *network-wide transport protocol* atop Glossy. In a nutshell, nodes with data simultaneously attempt to transmit their updates. After each transmission, the sink must acknowledge which packet it has received, if any. When all transmissions have completed, the network returns to sleep. Before going into the details of CRYSTAL in Section 5, we offer a concise primer on synchronous transmissions.

4. SYNCHRONOUS TRANSMISSIONS

Simply put, Glossy offers network-wide packet flooding and high-accuracy synchronization. In Glossy, a single node initiates a flood with a single transmission. Neighboring nodes receive it and immediately retransmit it, with their neighbors doing the same. While such a straightforward approach seems to lead to an inordinate number of collisions with many nodes transmitting simultaneously, Glossy observes that such concurrent transmissions need not be negative. In fact, they can be exploited due to a phenomenon of IEEE 802.15.4 radios called the *capture effect*. In these radios, a node can receive a packet despite interference from other transmitters when the signal of that packet is stronger than other signals or when the node begins to receive the packet sufficiently earlier than other signals. Further, if multiple transmissions initiate with a tiny temporal difference (smaller than 0.5 µs), the transmissions *constructively interfere*, increasing the probability of reception.

Glossy builds on these two phenomena, carefully controlling the timing of retransmissions to encourage constructive interference and to reliably flood a packet from a single initiator throughout the network. Experiments show that, with a single initiator, a Glossy flood reaches all network nodes with reliability >99% in few milliseconds, depending on the configuration parameters. To increase the flooding reliability, Glossy allows nodes to retransmit packets multiple times, denoting this with N.

Table 3: CRYSTAL parameters.

Parameter	Description
E	Epoch duration (reporting period)
W_S, W_T, W_A	Glossy maximum listening interval (slot duration) for the S, T, A phases
N_S, N_T, N_A	Number of Glossy transmissions in S, T, A phases
G	Guard time before the S, T, A slots
R	Number of consecutive silent TA pairs indicating the end of communication
Z	Number of consecutive unacknowledged T slots, or consecutive T and A slots with zero packets

5. CRYSTAL: PROTOCOL DESCRIPTION

We recall that our goals are to keep nodes asleep as much as possible and to disseminate the model updates in a timely, reliable fashion. CRYSTAL accomplishes these goals by using Glossy for rapid, highly reliable flooding. CRYSTAL itself is periodic, with the epoch determining when communication is possible toward the sink. Each epoch is formed by a very short active portion in which all nodes participate in data collection, and a much longer sleep portion when nodes consume very little power. The intricacies of CRYSTAL lie in how we guarantee that all updates are reliably received during the active portion using only Glossy transmissions.

In a nutshell. A CRYSTAL epoch starts with a synchronizing Glossy transmission from the sink, ensuring all nodes are temporally aligned and ready to participate in data collection. Subsequently, any node with a data packet to send transmits it with a Glossy flood. Due to the capture effect, at least one of these packets is highly likely to reach the sink, which then sends an acknowledgement via a Glossy flood, announcing the ID of the packet it received. With high reliability, all senders receive this acknowledgement, and if their data packet was not acknowledged, they simply try again by transmitting data then listening for the acknowledgement. This repeats until all transmitting nodes have received an acknowledgement for their data, then all nodes go to sleep.

An example. Figure 3 offers a sample of the active portion of a single CRYSTAL epoch, shown as a sequence of Glossy

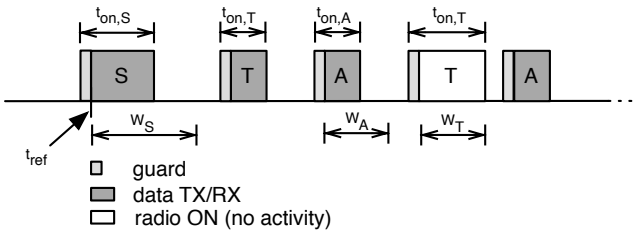


Figure 3: The active portion of a sample CRYSTAL epoch with one sender ($u = 1$) whose data is immediately acknowledged by the sink. For simplicity, $R = 1$.

transmissions. Table 3 offers the key parameters. The first slot, S, contains a synchronization message from the sink, and serves the purpose of preparing the network nodes for communication following the previous, long sleep interval. This is followed by some number of TA pairs in which T represents a data transmission slot for use by nodes transmitting data and A is an acknowledgement slot for use by the sink. The number of TA pairs in each epoch varies depending on the number of nodes with data to transmit and the desired reliability. To identify the end of the active portion of the epoch, we define a distributed termination policy, detailed later. Note that although we refer to this as the *active* portion, when a node is not involved in communication (either receiving or transmitting) its radio is off. Slots have a duration W, defining the maximum interval a node listens on the channel to detect an ongoing Glossy flood. When the latter occurs, it normally completes before the end of W, as seen by comparing the two T slots in Figure 3.

Detailing TA. Inside a single TA pair, all nodes with data to send become Glossy initiators, meaning they initiate floods in the T slot, which are then carried out concurrently throughout the network. All non-initiators act as forwarders. Although Glossy is a flooding protocol, we focus on packet reception *at the sink*, as our goal is data collection. Glossy was designed to work with only a single initiator, but our experiments in Section 7 show that, due to the capture effect, one of the concurrent transmissions reaches the sink with a probability close to 1. In this case, the T slot is *successful*, and the sink floods a positive acknowledgment in the next A slot. If, instead, the sink does not receive a packet, it floods a negative acknowledgement. Turning our attention back to the network, if the positive acknowledgement reaches the corresponding sender, its data is known to have been received at the sink and the sender will not attempt retransmission.

Distributed termination. All TA pairs follow this structure with the expectation that, in each subsequent TA, there will be fewer initiators until, eventually, some number of consecutive TA pairs have no data and only negative acknowledgements, triggering termination of the active portion of the epoch. We call these TA pairs without data *silent pairs*.

The number R of *consecutive* silent pairs is key in determining termination, based on three conditions. First, the sink goes to sleep after R consecutive T slots without data. Therefore, if a data packet did not get through in an earlier T slot, a node has R-1 additional attempts to transmit before the sink stops listening. Second, at a network node, when R consecutive negative acknowledgements are received the node goes to sleep because it knows that the sink will be asleep due to the first termination condition. Increasing R decreases the probability of falsely detecting the end of the transmissions, at the expense of energy consumption.

While these two conditions are nearly always sufficient, due to the occasional loss of acknowledgements we cannot rely on the second condition alone to put network nodes to sleep. Therefore, we define a third condition that *i)* puts a node to sleep if it is mistakenly awake due to the loss of a negative acknowledgement, but *ii)* simultaneously keeps a node awake for some additional time in noisy conditions when it is unable to detect activity in the network. We distinguish whether the node still has unacknowledged data to send. If yes, it goes to sleep when it misses Z acknowledgments from the sink in a row. In this case, the node's data might remain undelivered during the epoch, however transmission can be attempted in the next epoch. Alternately, if the node has no unacknowledged messages, it goes to sleep only when it detects Z consecutive slots with *zero* packets, i.e., neither data in T nor acknowledgment in A. Intuitively, this condition expresses the fact that a node "keeps trying" until the sink is likely asleep or inaccessible.

Synchronization. It is critical that CRYSTAL ensures all nodes are properly aligned to wake up and participate in data collection. This is particularly challenging in applications with long epochs, e.g., WATER. CRYSTAL accomplishes time alignment by beginning the epoch with a Glossy synchronizing packet and prepending this S slot with a sufficiently long guard time G to compensate for clock drift. For applications with long epochs, or systems composed of nodes with significant clock drift, this approach can lead to large guard times and increased consumption. Therefore, our CRYSTAL implementation includes a mechanism to *learn* the clock skew at each node, and adjust the wake-up period accordingly. While this works remarkably well, guards are still needed due to imperfect estimation and changes in clock skew over time. In addition to a guard at the beginning of the epoch, each T and A slot also includes a guard time; this compensates for clock drift should the synchronization packet be lost.

All nodes expect to receive the synchronizing Glossy message from the sink within $G+W_S$. By starting the CRYSTAL epoch with a synchronization packet from the sink, we expect to spread with high probability the correct reference start time t_{ref} to all nodes. Nevertheless, our implementation allows both G and W_S to grow with the number of consecutive losses of this synchronization packet.

Further, if the number of consecutive transmitters is large, the TA sequence can become similarly long, increasing the risk that nodes lose synchronization in the middle. To combat this, CRYSTAL makes every A slot a synchronizing Glossy slot, bringing all nodes back in line.

Glossy reliability. Inside a single flood, the Glossy protocol allows packets to be repeated a variable number of times N. A higher number of repetitions increases reliability but also power consumption, as we show in Section 7. CRYSTAL leaves the number of repetitions for each slot type, N_S, N_T, and N_A, to be configured to meet application requirements. For instance, Figure 3 shows the transmission in S longer than the transmissions in the T and A slots, a choice ensuring that the synchronization message at the beginning of the epoch is more reliable than the other transmissions, as discussed in Section 7.3.

6. ANALYTICAL MODEL

We derive an analytical model for CRYSTAL, estimating the average, network-wide radio-on time T_{on} within an epoch, a key constituent to estimate duty cycle and therefore lifetime. From Section 5, it is evident that T_{on} depends on the number u of concurrent updates, as this determines the minimum number of TA pairs necessary for their dissemination. We estimate $T_{on}(u)$ in two ways: an upper bound that uses only CRYSTAL's configuration parameters, and a much more accurate model that requires a few measurements of some constituents of CRYSTAL.

Upper bound. The average radio-on time across the entire

network is (over)approximated by:

$$T_{on}(u) = W'_S + (u + R)(W'_T + W'_A) \quad (1)$$

where $W'_x = G+W_x$ is the slot duration in Glossy augmented by the short guard time discussed in Section 5, and R is the number of silent TA pairs. The equation above is a strict upper bound for T_{on} because, when an actual transmission takes place in a slot, the actual average per-slot radio-on time is $t_{on} < W'$. Knowing this value t_{on} for each slot type enables us to derive a much more accurate estimate, discussed next. Eq. (1) assumes perfectly reliable dissemination, potentially underestimating T_{on} when retransmissions occur. However, as discussed next and shown empirically in Section 7.4, retransmissions are extremely rare; the underestimation caused by neglecting them is amply overcome by the overestimation caused by considering W' in place of t_{on}.

Model. Determining the values of t_{on} for each slot type, hereafter referred to as $t_{on,S}$, $t_{on,T}$, $t_{on,A}$, is necessary to obtain accurate estimates of $T_{on}(u)$. The values of $t_{on,S}$ and $t_{on,A}$ can be safely assumed constant w.r.t. u, as transmission in the S and A slots is performed only by the sink; they are effectively a normal Glossy dissemination. This does not hold for T slots, in which multiple update senders may compete, and for which the value of u affects the radio-on time $t_{on,T}(u)$, as shown experimentally in Section 7.2.

Interestingly, the value of t_{on} (regardless of the slot type) also implicitly depends on W. Indeed, while the strict inequality $t_{on} < W'$ holds on average, this is not true for a single update transmission; if the expected number of packets N is not received, a node remains awake for the entire slot. If we know the fraction σ of nodes that complete their flood before the end of the slot, and the average time $\hat{t}_{on}$ it takes, we can formalize the dependency of t_{on} on W as:

$$t_{on} = \sigma\hat{t}_{on} + (1 - \sigma)W' \quad (2)$$

Empirical knowledge of all these parameters, which we acquire in Section 7.2, allows us to determine an accurate estimate of the average per-epoch radio-on time as:

$$T_{on}(u) = t_{on,S} + u\rho(t_{on,T}(u) + t_{on,A}) + R(W'_T + t_{on,A}) \quad (3)$$

where ρ is the average number of TA pairs required to successfully deliver an update to the sink. This parameter also implicitly defines the probability that an update is successfully disseminated in a single pair, easily computed as $\frac{1}{\rho} = p_T p_A$, where p_T is the probability that the update is received at the sink in a T slot, and p_A the probability that the acknowledgment sent by the sink is received in the subsequent A slot. In practice, as we show in Section 7.2, these probabilities *i)* depend on the number N of Glossy retransmissions *ii)* are both very high, causing ρ to be very small.

7. CHARACTERIZATION

We have implemented Crystal atop the original publicly-available version of Glossy based on ContikiOS for the TMote Sky platform. In this section, we analyze the operation of our Crystal implementation with the double goal of identifying and quantifying the main factors affecting its performance, as well as of measuring the parameters necessary to inform the model in Eq. (3).

After describing our experimental setup (Section 7.1) we focus on the mechanics of the S, T, A slots (Section 7.2). We then use this information to identify the best configuration for running Crystal in the Indriya testbed (Section 7.3) and use it in Section 7.4 to characterize the performance of Crystal at the level of a single round of execution within an epoch. This data is used in Section 7.5 to validate our model, which is then exploited in Section 8 to derive duty cycle estimates based on the datasets of Section 2.3.

7.1 Experimental Setup

We ran our experiments in the Indriya testbed that, at that time of writing, had 88–92 operational nodes. We generated two topologies with different transmit power levels, 0 dBm (power 31) and -15 dBm (power 7), yielding an average network diameter of 4 and 7 hops, respectively.

We tested Crystal during the night and also during the day, when the interference from Wi-Fi networks is significantly higher. We chose two channels, 20 and 26, which are believed to have respectively high and low influence from Wi-Fi. To assess the actual interference during the experiments, our Crystal test application sampled and logged noise (RSSI) in the inactive portion of each epoch. Additionally, we ran regular, isolated network connectivity tests probing all links individually, without any concurrent transmissions. This identified two nodes (71 and 83) unpredictably losing connectivity, which we removed from our analysis.

Crystal showed very similar performance on both channels during the night runs; however, the daytime results were inconsistent and difficult to assess. For example, while the majority of tests on channel 20 during the day yielded perfect reliability as in nighttime runs, in some others the packets from a handful of nodes were sometimes lost on the way to the sink. For instance, in one run 5 nodes showed a packet loss between 25% and 40%. Closer inspection revealed that these nodes were exposed to an average noise of -70 dBm and higher. Although we conjecture that the resilience built into Crystal is an asset in these harsh conditions, a full analysis and comparison w.r.t. the state of the art requires the ability to control and reproduce interference patterns. Therefore, in this paper we report the results only from night runs on channel 26. This is not to say that these experiments are interference-free: the average noise is between -90 and -95 dBm, while the maximum noise is often above -70 dBm and as high as -30 dBm for several nodes. This holds for both channel 20 and 26, despite the fact that the latter is often purported to be interference-free.

As for the scheduling of transmissions, for every test a unique table of nodes that should send packets in any given epoch was randomly generated and "replayed" cyclically by the nodes. Logging was performed via serial line during the inactive portion of an epoch. The logs contained information about transmission and reception of every message, and other vital information for each node and epoch. Node-level statistics (over all epochs) and network-wide ones (over all epochs and nodes) have been calculated offline from the logs.

7.2 Dissecting a Crystal Slot

Knowledge about key metrics of Crystal slot types is fundamental to inform the model we defined in Section 6, enabling a correct configuration of the system as well as accurate duty cycle estimates given an update traffic profile.

Setup. To measure these parameters, we run specialized "benchmarks", where each slot type is measured in isolation. We used S phases only (pure Glossy) or a combination of S

and a single TA pair. We used $E = 250$ ms to increase the sampling rate. We set $W = 20$ ms to ensure that all floods have enough time to complete. The results we show here are based on 1500 epochs for each individual point on the plot.

S and A phases. These phases are pure Glossy disseminations, as they are always performed by the sink. Figure 4 shows that the packet delivery rate (PDR) of the S phase, defined as the percentage of nodes that correctly receive the packet sent by the sink; PDR is very high, in line with results reported in the literature [12, 13, 16]. Furthermore, Figure 4 also confirms that, given a value of N, the corresponding reliability decreases as the network diameter increases, and is therefore lower in the low power case.

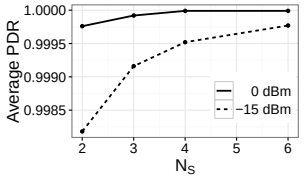
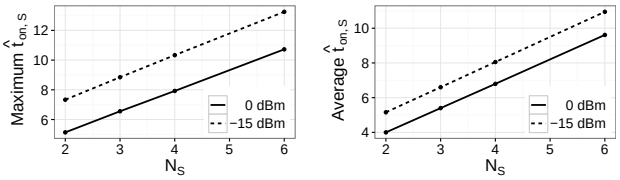


Figure 4: S phase: average PDR.

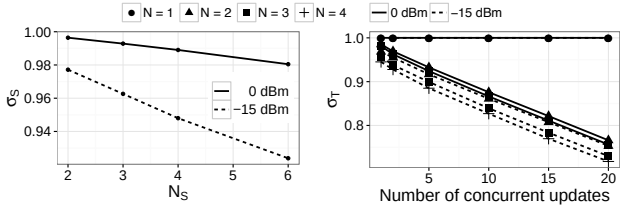
Figure 5 shows the per-slot radio-on time $\hat{t}_{on,S}$, computed only for the fraction σ_S of nodes receiving *all* N Glossy transmissions. $\hat{t}_{on,S}$ is crucial to dimension properly the slot duration W_S; if the latter is shorter than $\hat{t}_{on,S}$, the Glossy dissemination may not reach distant nodes. Therefore, W_S should be larger than the maximum value of $\hat{t}_{on,S}$, shown in Figure 5a.

On the other hand, W_S cannot be too large. Even if the disseminated packet is received with high probability (as shown in Figure 4), some of the individual N transmissions may be lost, causing the corresponding nodes to stay awake for the entire W_S, wasting energy. As shown in Figure 6a, the fraction σ_S of nodes for which this does *not* happen (i.e., those for which $\hat{t}_{on,S}$ is computed) decreases as N increases and is lower for the larger-diameter low power case, because both these factors increase the chance of individual packet losses. The average value of $\hat{t}_{on,S}$, shown in Figure 5b, is relevant for computing the overall $t_{on,S}$ according to Eq. (2).



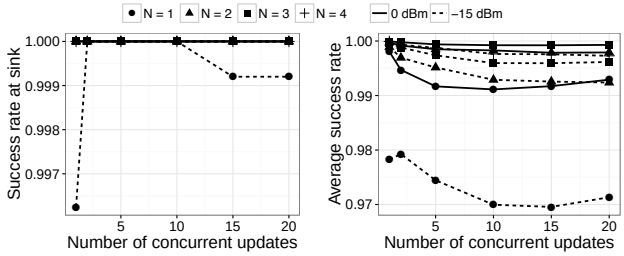
(a) Maximum. (b) Average.

Figure 5: S phase: per-slot radio-on time, $\hat{t}_{on,S}$. Note the different y-axis scale.



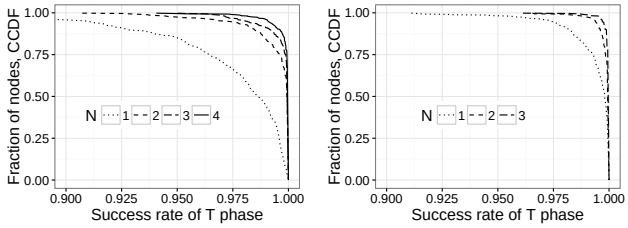
(a) S phase: σ_S. (b) T phase: σ_T.

Figure 6: Fraction of nodes completing transmission within the slot duration W. Note the different scale on both axes.



(a) To the sink, only. (b) Across the entire network.

Figure 7: T phase: average probability of successful transmission. Note the different y-axis scale.



(a) Low power (-15 dBm). (b) High power (0 dBm).

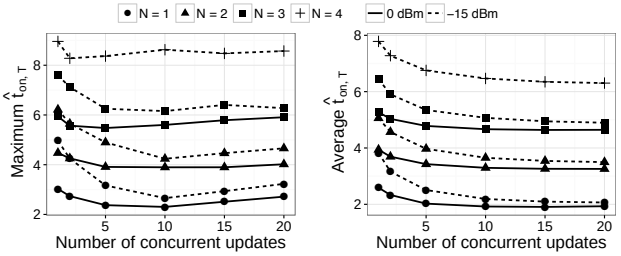
Figure 8: CCDF for Figure 7b.

The results above hold also for A phases, as their mechanics is exactly the same, apart from the different packet size, 8 B for S vs. 9 B for A. This yields a minimal difference on the radio-on time: on average, $\hat{t}_{on,A} = \hat{t}_{on,S} \times 1.020$.

T phase. Unlike S and A, the T phase behaves as in Glossy only if $u = 1$, i.e., there is only one sender in the network. Otherwise, if $u \geq 2$, multiple senders act as Glossy initiators, and compete during dissemination; as described in Section 4, the capture effect determines which packet among those concurrently broadcast is received, if any.

This mode of dissemination is inherently more unreliable than standard Glossy; however, the built-in redundancy inherited from the latter still yields a rather high probability that *at least one* update among those concurrently sent in the T slot is correctly received at all nodes. Figure 7b shows this probability of success, computed across the entire network, as a function of N and the number of concurrent updates u. The chart shows that, as the number of concurrent updates increases from $u = 1$, competition among senders causes transmissions to increasingly fail—up to a given point. As u increases, in fact, the probability that a node is close to one of the senders, and therefore receives its packet with high probability, increases. The chart also includes curves with $N = 1$ that, not surprisingly, provides the worst reliability especially in the low power case. This value was not included in the analysis of the S and A phases; these are used for time synchronization, and in this case Glossy requires $N \geq 2$.

On the other hand, the T phase in Crystal is devoted to communication *towards the sink*. The probability of successful transmission to the sink in our experiments, shown in Figure 7a, is always at 100% except for the configuration with low power and $N = 1$. The reader may be led to think that this is the result of conveniently selecting the sink node. However, Figure 8 shows that a significant fraction of



(a) Maximum. (b) Average.

Figure 9: T phase: per-slot radio-on time, $\hat{t}_{on,T}$.

nodes similarly enjoy 100% reception rate when chosen as a sink. The chart shows the complementary cumulative distribution function (CCDF) of the probability of success per node (over all values of u), based on the same data shown in Figure 7b; effectively, this allows us to compute the probability of success for each node, in case it were chosen as the sink. For $N = 2$ and high power, in the worst case ($u = 15$) 43% of the nodes have perfect reception rates, and 53% have $> 99.9\%$; for $N = 3$ at low power, 61% of the nodes have 100% reception. This confirms that our choice of sink is not biased.

Finally, Figures 9 and 6b show the average values of $\hat{t}_{on,T}$ and σ_T for the T phase. Figure 9b shows that the value of $\hat{t}_{on,T}$ decreases rapidly as concurrent updates increase from $u = 1$; this is due to the increased likelihood of finding a closer sender, which therefore yields a shorter radio-on time. As u further increases, however, the density of senders increases and therefore the likelihood that their transmissions result in a packet loss. Since, as already mentioned, in Glossy a node remains awake (up to the end of W) until N transmissions of the same packets have been received, the increase in packet loss yields an increase in the average and, especially, maximum values of $\hat{t}_{on,T}$. This phenomenon is mirrored by the linear decrease in the fraction σ_T of nodes that complete dissemination within W, shown in Figure 6b.

7.3 Configuring Crystal

The results in Section 7.2 enable us to determine a reasonable configuration for Crystal, i.e., one that strikes an appropriate balance between reliability and energy consumption. We later use this configuration, shown in Table 4, to further analyze the inner characteristics of Crystal and its overall performance against our datasets. We distinguish the configuration based on the power, as we have seen that this is the parameter that most affects the configuration.

Slot configuration: High power. N is the critical parameter affecting reliability, as already discussed. For the S and A phases, Figure 4 shows that $N = 3$ provides 99.99% reliability. Larger values further approach perfect reliability but induce higher energy costs, due to higher radio-on time $\hat{t}_{on,S}$ (Figure 5) and fraction $1-\sigma_S$ of nodes remaining awake for the entire slot (Figure 6a). Therefore, $N = 3$ is a good tradeoff for both S and A. As for T, similar considerations motivate $N_T = 2$. In principle, $N_T = 1$ could further reduce $\hat{t}_{on,T}$ and energy consumption; however, this would make sink selection more critical, as shown in Figure 8.

The slot duration W should be chosen for each phase by looking at the maximum radio-on time t_{on}. For the S and A phases, a value $N = 3$ implies $W \geq 7$ ms (Figure 5a). We use this value for W_A, while we use a higher $W_S = 10$ ms, given that the S phase is crucial for synchronization. In any case, unlike for the T and A phases, the impact of W_S on duty cycle is limited, given that *i)* the S phase occurs less frequently than T and A, for $u > 0$, and *ii)* $\sigma_S > 99\%$ for $N_S = 3$ (Figure 6a), therefore the impact of W_S (which comes into play only for the remaning 1% of the S phases) is negligible. A similar reasoning for the T phase, based on Figure 9a and the chosen $N_T = 2$, yields $W_T = 5$ ms.

Slot configuration: Low power. The configuration for low power is determined based on analogous reasoning. For the S and A phases, we choose $N = 4$. Although it does not yield reliability as good as its high power counterpart (Figure 4), higher values of N would significantly increase energy consumption due to the dissemination time $\hat{t}_{on}$. For the T phase, $N_T = 1$ is not an option, as it does not guarantee reliable transmission to the sink (Figure 7a). Our choice of $N_T = 3$ approaches the overall reliability of the high power counterpart (Figure 7b) and limits the impact of the sink placement.

For S and A, a slot duration $W = 12$ ms is sufficient to guarantee that $W > \max(\hat{t}_{on})$ (Figure 5a). Therefore, we use this value for the A phase, and a higher value $W_S = 14$ ms, coherently with the high power case. We then set $W_T = 8$ ms based on similar reasonings (Figure 9a).

Slot configuration: Guards. Selecting W is not enough; we must determine also the duration of the guard G preceding it. We verified that $G = 150$ μs yields good reliability for all types, even with $E = 5$ minutes, as discussed in Section 7.4.

Terminating a Crystal round. As discussed in Section 5, two additional parameters govern the behavior of Crystal, specifically concerning the termination of the sequence of TA pairs in a single round: the number of silent pairs R, and the number of missed acknowledgments Z.

For the former, we use $R = 2$ as we determined experimentally that *i)* higher values do not bring additional benefits w.r.t. reliability, while they obviously greatly and negatively affect energy consumption *ii)* using $R = 1$ in general negatively affects reliability, although we show in Section 7.4 that the energy savings it enables can be exploited in some cases.

Finally, we use $Z = 4$ as we determined experimentally that this yields good reliability, and that different values bear a limited impact on the performance of Crystal.

Table 4: Crystal configuration parameters. W_x and G values are expressed in ms.

Power	N_S	N_T	N_A	W_S	W_T	W_A	G	R	Z
High	3	2	3	10	5	7	0.15	2	4
Low	4	3	4	14	8	12	0.15	2	4

7.4 Dissecting a Crystal Epoch

We now focus on the mechanics of Crystal operation inside an epoch, i.e., the entire sequence of S, T, A phases necessary to disseminate a given number u of updates.

Setup. We use $E = 2$ s to accommodate long TA sequences, as we need to explore u values in the range 0–40. We use $R = 2$ and the slot durations in Table 4. For every point in the parameter space, we gathered data for 450 epochs.

Reliability. For both high and low power, the configuration in Table 4 yields 100% reliability, for all values of u we con-

Table 5: Average TA pairs required for each update, $\rho(u)$.

	1, 2, 5, 10	15	20	30	40
High power	1	1.0004	1.0003	1.0001	1.0004
Low power	1	1	1	1	1.0002

sider. In other words, despite the fact that at most one out of the u updates is delivered in a single T slot, and that occasional packet loss may occur even for $u = 1$, the network-wide transport mechanism we devised is very effective in ensuring reliability. Re-transmissions do occur however, as shown in Table 5; these are more frequent with high power, consistent with the larger collision domain. Their number is however very small; even for the maximum $u = 40$ considered, only 7 times in 450 epochs an extra TA pair was needed.

Impact of R on the reliability of TA chains. The results we just presented are derived with $R = 2$ silent pairs; for a TA chain to break prematurely (i.e., before u TA pairs have been executed) it must happen twice in a row that the packet transmitted in a T slot is not received at the sink, which therefore replies with an empty acknowledgment in the corresponding A slot. The probability of this event is extremely low in practice, yielding the aforementioned very high reliability.

Nevertheless, while it does not make sense to explore $R > 2$, the question remains about the impact of a lower value $R = 1$, which would enable energy savings. We focus again on $u = 40$ concurrent updates, as this defines a challenging test to reliability. Figure 10 shows the results for high power, with $R = 1$, over 1800 epochs. The chart shows that the majority of chains terminate correctly upon the 41^{st} TA pair; a few terminate slightly after, due to retransmissions. However, a few chains terminate earlier, therefore causing the loss of one or more updates; the probability of this happening increases towards the end of the chain. This is explained by the fact that the "stronger" a sender is the earlier its transmission succeeds; therefore, the end of the chain is usually populated by the "weakest" senders, for which the probability of packet loss is higher. Nevertheless, in a few cases, the TA chain breaks even halfway, around the 20^{th} position, causing the loss of half the updates. Therefore, a value $R = 1$ cannot be used *for an entire chain*.

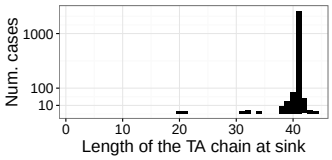


Figure 10: Analyzing TA chains, $u = 40$, $R = 1$.

Opportunity for optimization: Dynamic R. On the other hand, the positive side of the previous argument is that the TA chain in Figure 10 does not break until the 20^{th} position, with $R = 1$. This observation enables significant savings in energy consumption without prejudicing reliability.

Recall from Section 2.3 that the message suppression achieved by data prediction yields traffic profiles where the majority of epochs see at most one update. This holds for all our profiles except WATER (Figure 2c).

Therefore, optimizing the case with $u = 0$ is of paramount importance. In the $R = 2$ configuration we used thus far, CRYSTAL unfolds a transmission schedule with 5 slots, arranged in a STATA sequence. This was motivated by the reasoning that when using $R = 1$ (i.e., STA), the loss of a sent update would cause the sink to mistakenly believe that no update is disseminated, and send an empty acknowledgment in the A phase, effectively putting the entire network to sleep. However, Figure 10 shows that, in practice, *a chain is extremely unlikely to break on the first TA pair*. This is corroborated by Figure 7a showing that, at both powers, when any number u of updates are sent concurrently in a T slot, one *always* reaches the sink.

This leads to a strategy with a *dynamic* R value, using $R = 1$ for the first TA pair and, if a transmission occurs in it, switch to $R = 2$ for the rest of the epoch. In the cases where no update is actually transmitted, this dynamic assignment of R saves an entire TA pair, enabling substantial savings in energy consumption, as we further illustrate in Section 8.

Assessing the impact of the epoch duration. Until now, we executed experiments with an epoch $E = 2$ s, motivated by the need to reduce experiment time while exploring several combination of parameters. However, this is a rather small duration if compared with the epoch typically adopted in sensing applications; Table 1 shows that, in our real-world datasets, E ranges between 30 s and 10 minutes. Are the results we derived thus far applicable to these long epochs?

In principle, the inner working of CRYSTAL is determined by the execution of the schedule, which is the same irrespective of the epoch length. Therefore, all of our estimates still hold unchanged; we verified this experimentally, although we omit the results due to space limitations.

A threat to this statement comes from time synchronization, to ensure the correct operation of the underlying Glossy layer. The S phase serves this purpose; however, if E is very large, the clock drift among nodes may grow to a point where the synchronization packet in the S slot is lost, causing inefficiencies due to node de-synchronization. However, this can be easily solved by choosing a larger guard for the S slot.

In our experiments, we verified that the value $G = 150$ μs we use for all slot types is enough to reliably maintain time synchronization for epochs up to 5 minutes. This enables us to apply the model of Section 6 to the duty cycle computation by simply changing the value of the epoch. An investigation of longer epochs is beyond the scope of this paper.

7.5 Validating the Model

We now focus on validating the accuracy of the CRYSTAL model we presented in Section 6. The upper bound of Eq. (1) can be determined solely by knowledge of the CRYSTAL configuration parameters shown in Table 4. Instead, the more accurate estimate of Eq. (3) requires also knowledge of the parameters $t_{on,S}$, $t_{on,T}$, $t_{on,A}$, and ρ.

We have two options for determining these parameters. The first one is to derive them from the slot-centric analysis in Section 7.2. The parameter values in this case are less accurate, as they are derived from specialized, slot-centric *benchmarks* instead of a full CRYSTAL run. These benchmarks are faster to gather than CRYSTAL runs and, as shown in Section 7.3, useful to choose the CRYSTAL configuration; it is therefore interesting to see what error they introduce in estimating T_{on}. The second option is instead to acquire these parameters directly from the full CRYSTAL experimental *runs* in Section 7.4. In this case, the parameters are obviously more accurate. These two model variants, derived from benchmarks and from runs, are then compared to the

Table 6: Validating the model: per-epoch radio-on time $T_{on}(u)$, in milliseconds.

	number of concurrent updates						
	0	1	2	5	10	15	20
	High power						
Maximum from Crystal runs	30.6	41.51	51.82	81.54	130.18	180.13	231.33
Upper bound	34.75	47.05	59.35	96.25	157.75	219.25	280.75
Over-approximation (%)	13.57	13.34	14.52	18.04	21.18	21.71	21.36
Average from Crystal runs	27.41	37.01	46.26	73.95	119.15	164.74	210.08
Model from benchmarks	26.82	36.32	45.38	72.25	117.52	163.96	211.69
Model from Crystal runs	27.26	36.87	46.16	73.9	119.18	164.86	210.25
Error vs. benchmarks (%)	-2.15	-1.87	-1.91	-2.3	-1.37	-0.48	0.77
Error vs. runs (%)	-0.56	-0.38	-0.23	-0.07	0.03	0.07	0.08
	Low power						
Maximum from Crystal runs	48.86	71.59	85.37	138.4	224.44	305.04	395.61
Upper bound	54.75	75.05	95.35	156.25	257.75	359.25	460.75
Over-approximation (%)	12.05	4.84	11.69	12.89	14.84	17.77	16.47
Average from Crystal runs	41.81	58.23	73.62	118.14	189.88	259.08	327.64
Model from benchmarks	41.75	56.81	70.92	112.61	182.81	254.47	328.06
Model from Crystal runs	41.65	56.76	71.43	114.79	185.79	254.7	323.1
Error vs. benchmarks (%)	-0.68	-0.65	-1.13	-2.08	-1.61	-0.03	1.63
Error vs. runs (%)	-0.92	-0.74	-0.42	-0.18	0.003	0.06	0.09

actual T_{on} in the CRYSTAL runs of Section 7.4.

The results are shown in Table 6, for both high and low power, as a function of the number u of concurrent updates. In the top part of the table, for both high and low power, we find confirmation that Eq. (1) is indeed an upper bound for T_{on}, by comparing against the maximum T_{on} in the CRYSTAL runs. The over-approximation introduced by neglecting the fact that $t_{on} < W$ grows with u, as expected. However, it always remains below 22%; therefore, the upper bound is still a valid design tool to get a first rough estimate of T_{on}.

In the rest of the table we assess (both variants of) the model by comparing its estimates against the average $T_{on}(u)$. We consider the error $\epsilon = \frac{real-model}{real}$, in percentage. As expected, the error of the benchmarks variant is higher, yet $|\epsilon| \leq 2.3\%$. The runs variants has much higher accuracy, as expected, always achieving $|\epsilon| < 1\%$.

Therefore, we can conclude that our model is a very good approximation of the real behavior of CRYSTAL, and we can exploit it next for computing the duty cycle over long datasets, without introducing a significant error.

8. ULTRA-LOW POWER WIRELESS SENSOR NETWORKS: A REALITY

The premise of this paper is that by combining the power of data prediction with a network stack efficiently supporting the traffic patterns it induces, it is possible to achieve ultra low-power WSNs. To verify the extent to which we achieve this goal we need to ascertain the duty cycle CRYSTAL achieves on the datasets we illustrated in Section 2.3. We divide our evaluation in two complementary parts. In the first one, we apply our model to the datasets, therefore estimating for all of them the duty cycle achievable over a long time span—impractical to reproduce in a testbed. In the second one, we instead measure *directly* the duty cycle in 2-hour experimental sessions where we compare the performance of CRYSTAL against the staple CTP-based stack, concerning not only duty cycle but also reliability.

In both cases, the duty cycle for each dataset is given by:

$$DC = \frac{\sum_{u=0}^{N} T_{on}(u)e(u)}{E\sum_{u=0}^{N} e(u)}$$

where $e(u)$ is the number of epochs in which u updates are transmitted concurrently, and $T_{on}(u)$ is the per-epoch radio-on time. The value of $e(u)$ is known for all datasets; see Table 2 for the INDOOR temperature one. As for $T_{on}(u)$, in the first part we use the model estimates, while in the second part we use directly the measured value. All results, computed and measured, are reported for the CRYSTAL variants with fixed and dynamic R, and for both high and low power.

***Computing* the duty cycle from datasets**. Table 7 shows the results of applying the model to our datasets, using the more accurate estimates resulting from the parameters derived from experimental runs, as described in Section 7.5.

For the INDOOR dataset and high power, the upper bound estimate already places the duty cycle of CRYSTAL around our per-mille target. For the temperature and humidity datasets the upper bound for DC is slightly above 0.1% (i.e., 1‰) with a fixed R, and slightly below with a dynamic one. The light dataset has a slightly higher DC, as it has the highest number of concurrent updates among INDOOR datasets. However, the more accurate estimates provided by the model show that, by using a dynamic R, DC is reduced to slightly above 1‰ for light and as low as 0.7‰ for temperature and humidity. These best DC values translate to remarkable improvements w.r.t. CTP: up to 70x for temperature.

The DC of TUNNEL is ∼0.1‰ lower than the INDOOR temperature and humidity datasets; this is reasonable, as these three datasets have a similar epoch and traffic pattern, as illustrated in Section 2.3. However, given the very small values at stake, this minuscule difference translates in an additional 8–10x improvement w.r.t. CTP.

The SOIL datasets also have a similar traffic pattern but a much longer (∼20x) epoch. This brings the upper bound of DC for soil temperature to reach a stunning 0.06‰—i.e., 60 ppm. The more accurate model estimate brings this value down to 50 ppm, and dynamic R further reduces it to a tiny $DC = 30$ ppm. In this case, the table does not report any comparison against CTP. This would require finding the right configuration for epochs this long, and would anyway yield an exorbitant amount of control traffic w.r.t. the application one. However, our model reports an improvement of 955.6x for air temperature and 1592.67x for soil temperature.

Table 7: CRYSTAL duty cycle for the datasets of Table 1, and comparison with plain CTP (i.e., without data prediction). The datasets with the "*" are artificially normalized to a 30-second epoch.

Application & Dataset		Epoch	Duty Cycle (%), high power					Duty Cycle (%), low power				
			upper bound		model (runs)		vs.	upper bound		model (runs)		vs.
			$R=2$	$R=1,2$	$R=2$	$R=1,2$	CTP	$R=2$	$R=1,2$	$R=2$	$R=1,2$	CTP
INDOOR	temperature	30 s	0.125	0.091	0.098	0.068	**70.26x**	0.197	0.142	0.15	0.104	**64.9x**
	humidity	30 s	0.127	0.095	0.1	0.071	**67.3x**	0.201	0.148	0.153	0.109	**61.93x**
	light	30 s	0.17	0.15	0.132	0.114	**41.91x**	0.272	0.238	0.204	0.176	**38.35x**
SOIL	air temperature	600 s	0.007	0.006	0.006	0.005	N/A	0.011	0.009	0.009	0.007	N/A
	soil temperature	600 s	0.006	0.004	0.005	0.003	N/A	0.009	0.006	0.007	0.004	N/A
TUNNEL	light	30 s	0.12	0.083	0.094	0.061	**78.33x**	0.189	0.128	0.144	0.093	**72.58x**
WATER	chlorine	300 s	0.081	0.081	0.061	0.061	N/A	0.133	0.133	0.094	0.094	N/A
SOIL*	air temperature	30 s	0.143	0.12	0.112	0.092	**51.93x**	0.227	0.19	0.172	0.141	**47.87x**
	soil temperature	30 s	0.117	0.078	0.092	0.057	**83.82x**	0.185	0.119	0.141	0.087	**77.59x**
WATER*	chlorine	30 s	0.809	0.809	0.608	0.608	**7.86x**	1.327	1.326	0.939	0.939	**7.19x**

Table 8: Measuring reliability and duty cycle from Indriya experiments. The model values in italics are those that exhibit a (very small) change w.r.t. those in Table 7, when scaling down to a 2-hour traffic profile.

Network Stack	INDOOR, **temperature**				INDOOR, **light**				WATER*			
	yield (%)	DC (%)	gain vs. CTP no DBP	gain vs. CTP DBP	yield (%)	DC (%)	gain vs. CTP no DBP	gain vs. CTP DBP	yield (%)	DC (%)	gain vs. CTP no DBP	gain vs. CTP DBP
	High power											
CTP (no DBP)	98.33	4.778	1x	N/A	98.33	4.778	1x	N/A	98.33	4.778	1x	N/A
CTP (DBP, best configuration)	100	0.743	6.4x	1x	99.60	0.912	5.2x	1x	99.09	2.372	2x	1x
CRYSTAL (model, fixed R)	N/A	*0.097*	49.3x	7.7x	N/A	*0.131*	36.5x	7x	N/A	0.608	7.9x	3.9x
CRYSTAL (measured, fixed R)	100	0.098	49.0x	7.6x	100	0.131	36.4x	7x	100	0.606	7.9x	3.9x
CRYSTAL (model, dynamic R)	N/A	0.068	70.3x	10.9x	N/A	*0.113*	42.3x	8.1x	N/A	0.608	7.9x	3.9x
CRYSTAL (measured, dynamic R)	100	0.068	70.3x	10.9x	100	0.114	42x	8x	100	0.606	7.9x	3.9x
	Low power											
CTP (no DBP)	97.13	6.750	1x	N/A	97.13	6.750	1x	N/A	97.13	6.750	1x	N/A
CTP (DBP, best configuration)	100	0.825	8.2x	1x	99.60	1.087	6.2x	1x	98.99	2.997	2.3x	1x
CRYSTAL (model, fixed R)	N/A	*0.149*	45.3x	5.5x	N/A	*0.202*	33.4x	5.4x	N/A	0.939	7.2x	3.2x
CRYSTAL (measured, fixed R)	100	0.154	43.8x	5.3x	100	0.211	31.9x	5.1x	100	0.923	7.3x	3.2x
CRYSTAL (model, dynamic R)	N/A	*0.103*	65.5x	8x	N/A	*0.175*	38.6x	6.2x	N/A	0.939	7.2x	3.2x
CRYSTAL (measured, dynamic R)	100	0.106	63.7x	7.8x	100	0.173	39x	6.3x	100	0.923	7.3x	3.2x

The WATER dataset is somehow in the middle w.r.t. the other datasets. Its epoch is 5 minutes (half of SOIL, 10x more than INDOOR and TUNNEL) but, as discussed in Section 2.3, it exhibits a much higher frequency of concurrent updates. Therefore, the upper bound is $DC = 0.8‰$, while the actual one is as low as $DC = 0.60‰$. Interestingly, using a fixed or dynamic R bears a negligible impact on WATER. This is not surprising given that only 0.34% of the epochs have $u \leq 1$ (Figure 2c), i.e., 15 epochs out of the 4310 in the dataset.

In illustrating the results, we focused for simplicity on the high power ones; low power increases the network diameter, leading to a slightly higher DC. However, the right-hand side of Table 7 clearly shows that CRYSTAL achieves a DC around our 1‰ target and improves significantly over CTP.

This analysis allows us to put the duty cycle that can be achieved by CRYSTAL into the context of typical parameters for the real-world applications from which the datasets were obtained. In this respect, *a duty cycle below per-mille, and in some cases of parts-per-million, is several orders of magnitude smaller than what is achieved by the state of art.*

On the other hand, the epoch duration has a strong impact on the overall duty cycle. Therefore, the bottom of Table 7 offers an alternative view where the duty cycle of the various datasets is normalized to the lowest 30-second epoch of INDOOR and TUNNEL. Clearly, this is *purely speculative*, as in reality changing the epoch duration would actually change the probability of concurrent updates: the shorter the epoch, the lower this probability. In other words, we are *artificially defining a more challenging setup for* CRYSTAL.

Table 7 shows that, with this artificial normalization, the DC achievable for SOIL is in line with the other datasets that are not normalized; SOIL temperature actually achieves an improvement of 83.82x over CTP, the highest in our comparison. This is not the case for WATER, whose upper bound is 8‰, and best DC is 6‰. Nevertheless, considering the peculiar pattern shown in Figure 2, where essentially *every* epoch in WATER has concurrent updates, and the fact that even in these conditions CRYSTAL achieves a 7.86x improvement w.r.t. CTP, we argue this is actually a remarkable result.

***Measuring* the duty cycle (and reliability) from testbed experiments**. We now report about experiments that enable us to *measure* the duty cycle, therefore validating the findings we obtained by computing DC from the datasets with our model. In addition, this enables us to also evaluate the reliability of CRYSTAL, not captured by our model.

For these experiments, we "scale down" the traffic profiles of our datasets to reproduce their trends over a much shorter interval. The latter is determined by the maximum length of Indriya experiments (2 hours, i.e., 240 epochs of 30 s) minus a "burn-in" time for the CTP topology to stabilize, yielding experiments that are 200 epochs long.

Given a traffic profile, scaling is performed by simply multiplying by 200 the fraction of epochs with a given number u of updates. For instance, for INDOOR temperature in Table 2, the number of epochs with $u = 0$ updates becomes $0.8211 \times 200 = 164$. This obviously removes some values of u for which very few epochs exist (e.g., $u = 13$ in Table 2) but faithfully preserves the dominant trends of the profile. The latter is then reproduced by choosing, at each epoch, u

nodes at random to serve as the update senders.

We focus only on three representative datasets, based on the estimates in Table 7: *i)* INDOOR temperature, for which CRYSTAL achieves good performance; *ii)* INDOOR light, the most challenging among the INDOOR datasets; *iii)* WATER normalized to a 30 s epoch because, albeit artificially generated, serves as a very challenging case for CRYSTAL.

We repeat the experiments for each dataset with both high and low power. For each combination, we compare the estimate given by our model against the sum of the T_{on} values we measure in each epoch divided by 200, the total number of epochs. Moreover, we measure reliability as the data yield at the sink. We repeat all experiments both with a fixed and dynamic R. Finally, for each combination we also compare against plain CTP (no prediction) as in Table 7, and also with the best configuration (for the combination of power and dataset) for CTP with DBP data prediction.

Table 8 shows the results. In all of our experiments CRYSTAL achieved 100% reliability. Plain CTP achieved the lowest reliability, as expected due to the higher traffic. However, even the CTP/DBP combination, despite the much sparser traffic induced by data prediction, achieved 100% only in 1/6 of the cases, namely, INDOOR temperature at high power.

The duty cycle values in Table 8 agree closely with those in Table 7. The scaled down profiles induce tiny changes across the two tables, marked in italics in Table 8. The measured DC is always very close to the model estimates, therefore corroborating the results derived in Table 7, including the improvement over plain CTP. Moreover, Table 8 shows that, in all combinations, CRYSTAL significantly improves (up to 10.9x) also against the best configuration of CTP/DBP; even in the challenging WATER* dataset, CRYSTAL achieves a 3.2x improvement, confirming that CRYSTAL offers a significant advancement w.r.t. the state of the art.

9. RELATED WORK

Data prediction [1, 15, 28] is applicable to a large number of real-world applications, in which it greatly abates data traffic. Prior work [27] showed the limitations of a staple network stack in taking full advantage of data prediction, opening the way for CRYSTAL to remove limitations from idle listening and collection topology maintenance.

Routing Optimization. Accurate link estimation with broadcast beacons is a major overhead source in CTP. Some protocols mitigate this by using overhearing to estimate link quality [17, 26] or queue status [20] but, by relying on periodic traffic to update estimates, they compromise their accuracy and therefore usefulness in the extreme, low-traffic scenarios CRYSTAL targets. CRYSTAL also removes the low-power listening MAC, thus abating idle listening costs.

Ultra low-power data collection. A number of protocols achieve extremely low duty cycles by crossing the line between MAC and routing. Dozer [3] and Koala [22] do so by accepting extremely long latencies, minutes or days respectively, allowing nodes to sleep as long as possible between communication events. DISSense [6] reduces latency with a synchronous wake-up mechanism somewhat similar to CRYSTAL, but it only reaches per-mille duty cycle for reporting intervals of 60 minutes. CRYSTAL, instead, achieves per-mille duty cycle even with a short 30 s reporting interval, making it applicable to control-loop applications.

Concurrent Transmissions. Glossy [13] pioneered work on exploiting constructive interference to achieve millisecond level network-wide flooding from a single sender, and is at the core of other protocols supporting multiple senders.

In LWB [12] and Choco [29], Glossy floods are used to collect node requests for transmission slots and to distribute a global transmission schedule computed at the sink. Changes in traffic require regeneration and dissemination of the schedule. These solutions are effective for periodic data streams where a schedule is used for several transmissions, but inapplicable to the unpredictable and aperiodic traffic induced by data prediction, which would require a continuous rescheduling for each new transmission. Instead, CRYSTAL exploits the capture effect to build a network-wide transport protocol that involves sink-based acknowledgments, effectively allowing transmissions to compete and "self-schedule" based on the contingent communication needs.

Chaos [16] also supports multiple senders but allows concurrent, unscheduled transmissions by relying on the capture effect. Packets from different senders are merged before forwarding to compute, over several iterations, a network-level aggregate (e.g., the maximum value transmitted). CRYSTAL also relies on the capture effect, but with the opposite goal of delivering reliably to the sink *all* the data transmitted by nodes, bringing a different set of challenges and solutions.

Studies [23, 32] show that the Glossy flooding reliability degrades as the density and number of nodes increases. This problem is mitigated by limiting the number of concurrent transmitters [4,32–34], or improving synchronization between the transmitters by compensating for radio processing and signal propagation delays [31]. These enhancements are orthogonal to CRYSTAL and can improve its performance.

Concurrent transmissions have also been explored for bulk data transfer [7,9,10]. The use of channel and spatial diversity allows them to push larger amounts of data, however, their complexity and higher power consumption make them unsuitable for the ultra-low data rates of data prediction.

10. CONCLUSIONS

We demonstrated that the synergy between the high message suppression rates offered by data prediction and the lightweight, reliable, energy-efficient, fast communication enabled by synchronous transmissions bring the performance of WSNs to unprecedented levels. Our system, CRYSTAL, can disseminate multiple concurrent packets—the model updates generated infrequently, unpredictably, and aperiodically by data prediction—very fast and with perfect reliability, making it amenable to applications where WSNs are part of a control loop. Further, CRYSTAL achieves per-mille duty cycle, improving on the staple WSN stack (CTP + BoX-MAC) up to a factor of 80x. This remarkable reduction in energy consumption is achieved by neither compromising on the data reporting period nor using specialized hardware. On the contrary, the very small duty cycle achieved by CRYSTAL may offer a large leap towards energy-neutrality, e.g., in combination with energy harvesting techniques hitherto considered insufficient to power WSN nodes under real-world profiles like those we considered in this paper.

11. REFERENCES

[1] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[2] J. Berry, W.E. Hart, and C.A. Phillips. Sensor placement in municipal water networks. *J. Water*, 131, 2003.

[3] N. Burri, P. von Rickenbach, and R. Wattenhofer. Dozer: Ultra-low power data gathering in sensor networks. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2007.

[4] D. Carlson, M. Chang, A. Terzis, Y. Chen, and O. Gnawali. Forwarder selection in multi-transmitter networks. In *Proc. of the Int. Conf. on Distributed Computing in Sensor Systems (DCOSS*, 2013.

[5] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, S. Guna, G.P. Jesi, R. Lo Cigno, L. Mottola, A.L. Murphy, M. Pescalli, G.P. Picco, D. Pregnolato, and C. Torghele. Is there light at the ends of the tunnel? Wireless sensor networks for adaptive lighting in road tunnels. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2011.

[6] U. M. Colesanti, S. Santini, and A. Vitaletti. DISSense: An adaptive ultralow-power communication protocol for wireless sensor networks. In *Proc. of the Int. Conf. on Distributed Computing in Sensor Systems (DCOSS)—Workshops*, 2011.

[7] M. Doddavenkatappa, M. Chan, and B. Leong. Splash: Fast Data Dissemination with Constructive Interference in Wireless Sensor Networks. In *USENIX Symp. on Networked Systems Design and Implementation (NSDI)*. USENIX, 2013.

[8] M. Doddavenkatappa, M.C. Chan, and A.L. Ananda. Indriya: A low-cost, 3D wireless sensor network testbed. In *Proc. of the Int. Conf. on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom)*. 2011.

[9] M. Doddavenkatappa and M. Choon. P3: A practical packet pipeline using synchronous transmissions for wireless sensor networks. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2014.

[10] W. Du, J.C. Liando, H. Zhang, and M. Li. When Pipelines Meet Fountain: Fast Data Dissemination in Wireless Sensor Networks. In *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2015.

[11] EPANET. www.epa.gov/nrmrl/wswrd/dw/epanet.html.

[12] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Low-power wireless bus. In *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2012.

[13] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient Network Flooding and Time Synchronization with Glossy. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2011.

[14] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2009.

[15] N.Q.V. Hung, H. Jeung, and K. Aberer. An evaluation of model-based approaches to sensor data compression. *IEEE Trans. on Knowledge and Data Engineering*, 25(11):2434–2447, 2013.

[16] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and Efficient All-to-all Data Sharing and In-network Processing at Scale. In *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2013.

[17] O. Landsiedel, E. Ghadimi, S. Duquennoy, and M. Johansson. Low Power, Low Delay: Opportunistic Routing Meets Duty Cycling. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2012.

[18] K. Leentvaar and J. Flint. The capture effect in fm receivers. *IEEE Trans. on Communications*, 24(5):531–539, May 1976.

[19] Life Under Your Feet Project. lifeunderyourfeet.org/en/src.

[20] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali. Routing without routes: The backpressure collection protocol. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2010.

[21] D. Moss and P. Levis. BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking. Technical Report SING-08-00, Stanford Information Networks Group, 2008.

[22] R. Musaloiu-E., C.-J.M. Liang, and A. Terzis. Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks. In *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2008.

[23] C. Noda, C. Pérez-Penichet, B. Seeber, M. Zennaro, M. Alves, and A. Moreira. On the scalability of constructive interference in low-power wireless networks. In *Proc. of the European Conf. on Wireless Sensor Networks (EWSN)*, 2015.

[24] T. Palpanas, M. Vlachos, E.J. Keogh, and D. Gunopulos. Streaming time series summarization using user-defined amnesic functions. *IEEE Trans. on Knowledge and Data Engineering*, 20(7), 2008.

[25] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *Proc. of the Int. Conf. on Very Large Data Bases (VLDB)*, 2005.

[26] D. Puccinelli, S. Giordano, M. Zuniga, and P.J. Marrón. Broadcast-free collection protocol. In *Proc. of the Int. Conf. on Embedded Network Sensor Systems (SenSys)*, 2012.

[27] U. Raza, A. Camerra, A.L. Murphy, T. Palpanas, and G.P. Picco. Practical data prediction for real-world wireless sensor networks. *IEEE Trans. on Knowledge and Data Engineering*, 27(8):2231–2244, 2015.

[28] M. A. Razzaque, C. Bleakley, and S. Dobson. Compression in Wireless Sensor Networks: A Survey and Comparative Evaluation. *ACM Trans. Sensor Networks*, 10(1):5:1–5:44, December 2013.

[29] M. Suzuki, Y. Yamashita, and H. Morikawa. Low-power, end-to-end reliable collection using glossy for wireless sensor networks. In *Proc. of the Vehicular Technology Conference (VTC Spring)*, June 2013.

[30] D. Tulone and S. Madden. An energy-efficient querying framework in sensor networks for detecting node similarities. In *Proc. of the Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2006.

[31] Y. Wang, Y. He, D. Cheng, Y. Liu, and X. y. Li. Triggercas: Enabling wireless constructive collisions. In *Proc. of the Int. Conf. on Computer Communications (INFOCOM)*, 2013.

[32] Y. Wang, Y. He, X. Mao, Y. Liu, and X. y. Li. Exploiting constructive interference for scalable flooding in wireless networks. *IEEE/ACM Trans. on Networking*, 21(6):1880–1889, Dec 2013.

[33] D. Yuan, M. Riecker, and M. Hollick. Making 'glossy' networks sparkle: Exploiting concurrent transmissions for energy efficient, reliable, ultra-low latency communication in wireless control networks. In *Proc. of the European Conf. on Wireless Sensor Networks (EWSN)*, 2014.

[34] J. Zhang, A. Reinhardt, W. Hu, and S. Kanhere. RFT: Identifying Suitable Neighbors for Concurrent Transmissions in Point-to-Point Communications. In *Proc. of the Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2015.