Broadcast-Free Collection Protocol

Daniele Puccinelli
University of Applied Sciences of Southern Switzerland

Marco Zuniga
Delft University of Technology

Silvia Giordano
University of Applied Sciences of Southern Switzerland

Pedro José Marrón
University of Duisburg-Essen

Abstract

Asynchronous low-power listening techniques reduce the energy footprint of radio communication by enforcing link layer duty cycling. At the same time, these techniques make broadcast traffic significantly more expensive than unicast traffic. Because broadcast is a key network primitive and is widely used in various protocols, recently several techniques have been proposed to reduce the amount of broadcast activity by merging broadcasts from different protocols. In this paper we focus on collection protocols and investigate the more extreme approach of eliminating broadcast completely. To this end, we design, implement and, evaluate a Broadcast-Free Collection Protocol, BFC. We derive first-order models to quantify the costs of broadcasts, and evaluate the performance of BFC on a public testbed. Compared to the Collection Tree Protocol, the de facto standard for data collection, BFC achieves double-digit percentage improvements on the duty cycles. The specific benefits to individual nodes depend on the relative cost of unicast activity; we show that the nodes that benefit the most are the sink's neighbors, which are crucial for network lifetime extension. Eliminating broadcast also brings several other advantages, including extra flexibility with link layer calibrations and energy savings in the presence of poor connectivity.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms

Design, Experimentation, Performance

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SenSys'12, November 6–9, 2012, Toronto, ON, Canada.
Copyright © 2012 ACM 978-1-4503-1169-4 ...$10.00

Keywords

Collection, Sensor Networks, Routing, Broadcast, Energy Consumption

1 Introduction

Traditionally, the broadcast nature of the wireless channel has been viewed as an advantage of radio frequency (RF) communication. In a wireless medium, every transmission can be received by all nodes that lie within the sender's communication range. This important feature has been exploited in several studies, ranging from the construction of energy efficient routing trees [1, 2] to network coding [3]. However, the broadcast advantage requires all nodes to be awake at the same time.

In low power wireless networks with asynchronous link layer duty cycling, this underlying assumption does not hold true, because nodes are asleep for most of their time and periodically wake up with no coordination. In the presence of link-layer duty-cycling, broadcasts are, on average, twice as costly as unicasts [4]. In fact, the duration of a broadcast transmission must be stretched to cover the on time of all nodes within radio range. In contrast, for unicasts, it is enough to cover the on time of the intended receiver. Due to the energy cost gap between unicast and broadcast traffic, the low-power wireless community has recently devoted a lot of attention to the energy costs that come with broadcast communication [4, 5].

In spite of its high relative cost, broadcast is an important network primitive that is widely used by the control plane of several classes of network protocols. Data dissemination protocols utilize broadcasts to deliver information reliably to every node in a network [6, 7], neighbor discovery protocols utilize beacons to keep up to date information of the surrounding nodes [8, 9], and data collection protocols rely on broadcast messages to form and maintain their data gathering trees [10, 11]. The energy footprint of broadcast is particularly significant for ultra-low data rate scenarios; in [12], it is shown that the control overhead of broadcast-based collection greatly limits the energy benefits of model-driven data acquisition.

To limit the impact of broadcast across protocols, various techniques have been proposed to merge broadcasts [5, 13] or reduce their impact and reach [14]. In this paper we experiment with the drastic approach of eliminating broadcast

altogether. To this end, we focus on the key primitive of data collection and propose a Broadcast-Free Collection Protocol (BFC) that employs no broadcast traffic and only uses unicast traffic.[1] Instead of using dedicated broadcast packets to form the data gathering tree, BFC lets nodes discover routes by eavesdropping on the unicast transmissions from the surrounding neighbors.

We present the design of BFC and its TinyOS prototype implementation. We extensively illustrate and discuss the properties of BFC with a thorough experimental evaluation on the Motelab [15] testbed. We compare BFC to broadcast-based collection and use CTP [10], the *de facto* standard collection protocol for low-power wireless networks, as our benchmark. Some of our key results show that:

- *It is practical to perform data collection without broadcast control traffic*, achieving a double digit percentage reduction of the duty cycle of broadcast-based collection and preserving reliability while trading off latency in the route discovery process.
- Eliminating broadcasts has the *greatest impact on the sink's neighbors*. Our results show that BFC can improve their mean duty cycle by upwards of 70%.
- It is possible to perform *broadcast-free route maintenance*, and broadcast-free operation leads to *energy conservation in the presence of poor connectivity conditions*.

The rest of the article is organized as follows. First, we describe the limitations of broadcast communication in low power listening MACs and in data collection protocols, and use first-order models to quantify the benefits of eliminating broadcasts (Section 2). In Section 3, we describe the design and implementation of BFC, and in Section 4, we present the evaluation results obtained in the Motelab testbed [15].

2 Broadcast Transmissions in Low Power Wireless

In this section we describe the well-known energy issues with broadcast in low-power wireless and develop a first-order model to quantify the impact of broadcast on nodes depending on their role in the network. The model is a simplified version of more advanced models reported in the literature [16, 4].

2.1 Broadcasts with Low Power Listening

Generally speaking, radio communication is the primary source of energy consumption [17], and a large body of work has focused on reducing the on-time of radio transceivers. A key contribution in this direction is Low Power Listening (LPL [18]), a very popular solution for link-layer duty cycling. With LPL, all nodes are duty-cycled, and each transmitter must ensure that the transmission duration of each of its packets overlaps with the wakeup interval of the receiver. The most popular flavor of LPL is the one employed in TinyOS' standard MAC layer, BoX-MAC [19], which uses concepts from B-MAC [18] and X-MAC [20]. BoX-MAC

[1] Physically, unicast and broadcast transmission are both *broadcast* to the wireless medium. Unicast traffic is broadcast traffic logically disguised by way of aiming at a single intended receiver. BFC is free of logic broadcasts.

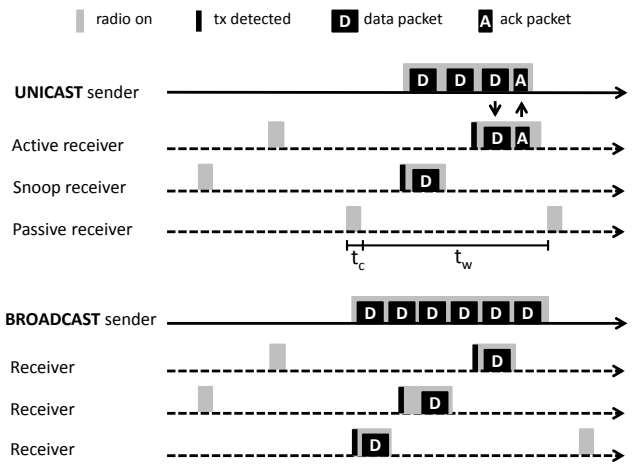


Figure 1. Unicast and broadcast transmissions in low power listening MACs. Broadcasts consume more energy because the duration of the packet train must match the entire wakeup interval of any potential receiver in the network. t_w represents the wakeup interval, and t_c the LPL periodic energy check time.

stretches the transmission duration of each packet by sending packet trains; unicast packet trains may be cut short by a link layer ack from the receiver, while broadcast packet trains must match the entire wakeup interval. The different impact of broadcast and unicast transmission is represented in Figure 1. If every node in the network has the same LPL wakeup interval, a broadcast packet is, on average, twice as costly as a unicast packet [4].

2.2 Broadcasts in Data Collection Protocols

Data collection protocols use unicasts for data traffic and broadcasts for control traffic to form the routing structure, which is typically a data gathering tree rooted at the sink [10, 11] or, more in general, a Destination-Oriented Directed Acyclic Graph [21, 22]. The routing structure is formed and maintained by broadcasts sent by all nodes. Once the routes are established, nodes use multi-hop unicast transmissions to forward their own data as well as the data of their descendants.

The standard approach to the management of broadcast control traffic in low-power wireless is based on the Trickle algorithm [6], whereby nodes use beacons aggressively to discover a route and converge to a fixed steady-state inter-beacon interval once the routes have been found. In the default implementation of CTP, the inter-beacon interval (t_{IBI}) follows an exponentially increasing pattern from 64 ms to roughly 8 minutes at steady state [12]. If a node needs fresh routing information due to topological instability in the network, the Trickle timer is reset to the minimum t_{IBI} to *pull* a route from the neighbors, *i.e.*, get them to broadcast a beacon with valid routing information.

We focus on steady-state operation and present a first-order analytical model to quantify the energy costs of broad-

cast communication. This model will enable us to quantify the expected energy benefits of eliminating broadcasts and to understand which classes of nodes can benefit the most.

Because RF communication is, in general, the most expensive operation in sensor networks, our analysis focuses on the duty cycle of the radio (the fraction of time that the radio is on). Employing the radio duty cycle as a proxy for energy consumption is a common technique in the wireless sensor network and low-power wireless literature [5][12][23].

2.3 Modeling the Duty Cycle

We model a periodic data collection scenario where nodes inject packets at regular intervals [10] with link-layer duty-cycling based on Low Power Listening (LPL) [18]. We assume that all nodes are duty-cycled with the same LPL wakeup interval except for the sink, which is always on (a common assumption also used in [10], because the sink is usually connected to a base station with access to a permanent power supply). In these scenarios, the duty cycle of the radio depends mainly on five parameters:

- t_w The LPL wake up interval.
- t_c The LPL periodic energy check time, equal to 11 ms in the default CC2420 LPL implementation.
- t_{rx} The packet reception time. In the standard TinyOS distribution, t_{rx} takes approximately 25 ms due to the radio on-time for packet reception (roughly 5 ms [24]) and a post-reception delay (set to 20 ms by default)[2].
- t_{IBI} The inter-beacon interval.
- t_{IPI} The inter-packet interval.

The interplay of these parameters defines the energy consumption of the radio. Overall, there are five main operations that affect the radio's on-time: (1) receive checks, (2) broadcast transmissions, (3) broadcast receptions, (4) unicast transmissions and (5) unicast receptions.

Receive checks occur once per wakeup interval, and their contribution to the duty cycle may be written as $\delta_{rc} = \frac{t_c}{t_w}$.

Broadcast transmissions occur every t_{IBI} and their duration is deterministic and equal to t_w (to wake up all potential receivers); their contribution to the duty cycle may be expressed as $\delta_{bt} = \frac{t_w}{t_{IBI}}$.

Receive checks and broadcast transmissions have the same impact on the duty cycle across all nodes. On the other hand, we will observe that broadcast receptions, unicast transmissions and unicast receptions depend on the local density of a node's neighborhood and on the node's depth within the data gathering tree.

Broadcast receptions depend on t_{IBI} and the number of neighbors. Denoting N_i as the number of neighbors of node i, at every t_{IBI} interval, node i receives N_i broadcast messages. Considering that the amount of time required to receive a packet is t_{rx}, the contribution of broadcast receptions to the duty cycle may be expressed as $\delta^i_{br} = \frac{t_{rx}}{t_{IBI}} N_i$.

[2]The post-reception delay is used to reduce delays on data transmissions. Instead of waiting until the next periodic energy check, a node remains awake for some time after receiving a packet in case other packets are being trasnmitted.

Unicast transmissions depend on the transmission load and the quality of the wireless channel. At every interval t_{IPI}, each node i has to transmit its own (locally generated) packet plus the packets received from its descendants. Let F_i denote the ratio of the total number of forwarded packets (both locally generated and relayed) per locally generated packet. If the channel from node i to its parent is lossy, each packet may require more than one transmission. Denoting Γ_i as the number of transmissions required for every successful reception (in other words, the measured ETX [25][26]), a node needs to transmit $F_i\Gamma_i$ packets at every interval t_{IPI}. Considering that node i starts a packet transmission uniformly at random across the LPL wakeup interval of the parent node, each unicast transmission takes an expected time of $t_w/2$ [4]. Together, these effects lead to a duty cycle contribution of $\delta^i_{ut} = \frac{t_w}{2t_{IPI}} F_i\Gamma_i$.

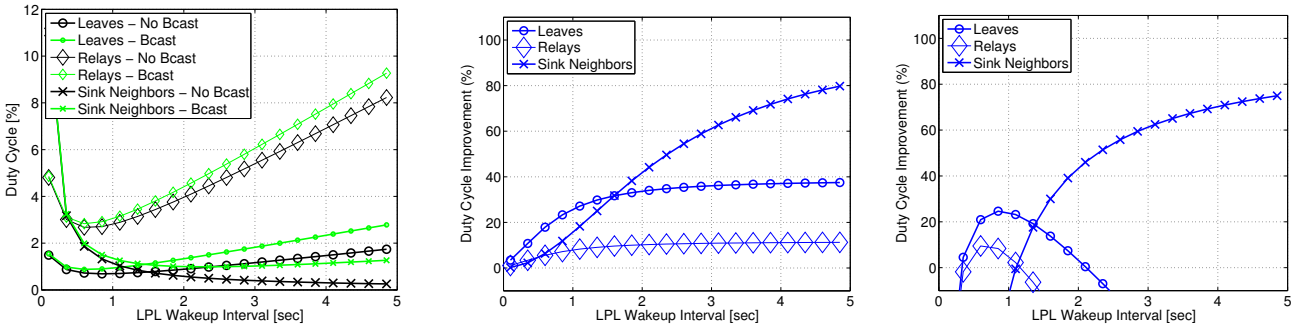
Unicast receptions are tricky to model. The radio on-time of a node depends not only on the packets addressed to that particular node, but also on the amount of time spent by that node snooping packets meant for other receivers. Denoting L_i as the total listening load of node i during the interval t_{IPI} (either intended and unintended receptions), the duty cycle due to unicast receptions can be expressed as $\delta^i_{ur} = \frac{t_{rx}}{t_{IPI}} L_i$

For any given node i, the duty cycle δ_i can be computed as the sum of the contributions illustrated above. This model is based on [16, 4] and also employs principles from [18]. In [16], the energy consumption of a set of basic radio primitives (receive checks and communication costs) is measured and employed to determine the energy consumption of different classes of nodes (relays and hosts – hosts are nodes with zero forwarding load, and we call them leaves in this paper). We use the same distinction but also consider the special case of the sink's neighbors [4], whose energy consumption profiles are fundamentally different if the sink's radio is not duty-cycled (*i.e.*, it is always on), as is often the case and as we assume throughout this paper. In particular, for sink neighbors the footprint of unicast transmissions, δ^i_{ut}, is typically three orders of magnitude smaller than for regular nodes because t_w (generally of the order of seconds) is replaced by the packet transmission time (of the order of ms).

The main simplification of our model is the use of radio on time without capturing the specific power consumption (accounted for in [16]); the ensuing loss of accuracy is acceptable for our purposes because we are only interested in trends and orders of magnitude. Note that we use the radio on time (and the duty cycle) as a proxy for energy consumption because our experimental work is entirely based on remote-access testbeds, where measuring the energy consumption of every node is not possible, while it is easy to measure the radio on time with software-based on-line energy estimation [27].

2.4 Insights Derived from the Model

Our model provides several important insights about the role of broadcasts in collection protocols. Figure 2(a) shows the expected duty cycle of three different types of nodes as a function of the LPL wake up interval (t_w). To asses the impact of broadcasts, we fix t_{IPI} to 5 minutes (a reasonable value for low data rate applications) and employ two t_{IBI} in-



(a) Duty cycles from model ($t_{\text{IPI}} = 5$ min)

(b) Improvement at each operating point

(c) Improvement relative to the optimal operating point

Figure 2. Eliminating broadcasts has a major positive impact on leaves and sink's neighbors.

tervals: 8 min (the steady-state value in the standard CTP implementation) and ∞ (*i.e.*, no broadcasts). In order to get realistic values for the parameters N_i, F_i, Γ_i, and L_i, we ran CTP during a 4-hour period in Motelab with the sink at one edge of the testbed and divided the nodes into three classes: leaves (nodes with $F_i < 2$ that are not within one hop of the sink), relays (nodes with $F_i \geq 2$ that are not within one hop of the sink), and sink's neighbors[3]. At the time of the experiments, there were 69 active nodes, and the breakdown of node roles was roughly 13% sink's neighbors, 27% relays, and 60% leaves. We measured an average forwarding load $\bar{F}$ of 7.2 for the sink's neighbors, 7 for the relays, and 1.2 for the leaves; an average ETX $\bar{\Gamma}$ of 1.1 for the sink's neighbors, 1.2 for the relays, and 1.4 for the leaves; an average neighboorhood size $\bar{N}$ of 14 for the sink's neighbors, 22 for the relays, and 18.5 for the leaves. As we are only after orders of magnitude, these calibrations are enough to ground the model into reality.

LPL has an optimal wake-up interval that minimizes the radio duty cycle. If the sleep time t_w is short, the duty cycle is high because the receive checks are more frequent. On the other hand, when t_w is long, transmissions dominate the energy consumptions. Figure 2(a) shows that, depending on the node's role, the optimal wake-up interval is in the $[0.5, 2]$ sec range (for the Bcast approach). This is a well-known trade-off that has been studied and illustrated in the literature [18][28][12] and is correctly captured by our model.

Eliminating broadcasts mostly benefits the lifetime of the sink's neighbors and leaf nodes. There are hardly any benefits for relays due to their high volume of unicast activity which dwarfs the impact of broadcast. On the other hand, leaves benefit a lot because broadcast accounts for a large portion of their energy usage. Their unicast activity is in fact limited to their own load, and if t_w is increased their unicast energy footprint is also increased. The sink's neighbors benefit the most, and their benefits grow as t_w is increased. In fact, broadcast is their primary source of energy usage: as the sink is always on, the cost of their unicast transmissions to the sink is reduced from hundreds or thousands of ms (depending on t_w) to a few tens of ms, because an ack is received after the transmission delay of one single packet. Figure 2(b) is based on Figure 2(a) and shows the relative gains achieved by each class of nodes in a broadcast-free scenario at each operating point. Figure 2(c) is also based on Figure 2(a) and shows the relative gains achieved by each class of nodes with respect to CTP's optimal operating point (at which the minimum the minimum duty cycle is achieved).

Eliminating broadcasts widens the optimal wakeup interval range. Overall, eliminating broadcasts provides a wider operational range for the duty cycle, making it possible to increase t_w beyond CTP's optimal setting. Increasing t_w means letting nodes sleep longer, but it also means making transmissions costlier. With broadcasts gone, only unicast transmissions are left, and the duty cycles become less sensitive to the calibration of t_w. Having the freedom to increase t_w is important for applications with very infrequent data traffic [12] and in scenarios with out-of-network interference. In fact, it has been shown that out-of-network interference may seriously affect the performance of LPL [24], causing nodes to wake up unnecessarily. A longer t_w means less frequent receive checks, thus decreasing LPL's exposure to interference [24].

3 The Design and Implementation of BFC

Conceptually, BFC uses a simple idea to eliminate broadcast communication in collection protocols: instead of sending periodic beacons to form the data gathering tree, nodes eavesdrop on the unicast transmissions in their neighborhoods and connect to a neighbor that already has a reliable path to the sink. The eavesdropping is conditioned on the duty cycle dictated at the link layer and does not exacerbate the overhearing problem: nodes only leverage unicast packets that they happen to overhear, so there is no extra energy cost for this form of passive snooping. We will observe that there are several challenges that need to be overcome in the design and implementation of a broadcast-free collection protocol. We build our prototype implementation on top of BoX-MAC [19]; in principle, BFC can work on top of a duty-cycled link layer as long as link layer acknowledgements are provided. The sink is assumed to be always on, as is customary in the literature [10]; this greatly reduces the burden on

[3] Strictly speaking, leaves should have $F_i = 1$, but in a real network leaves may occasionally act as relays for short periods of time; we use a threshold of 2 to filter out such nodes from the set of bona fide relays.

the sink's neighbors [4]. We assume a standard data collection application whereby every node injects traffic every t_{IPI} [10].

3.1 Route Discovery

3.1.1 *Initialization*

The goal of the initialization is for nodes within the sink's radio range to discover the sink. At startup, every node in the network sends one unicast data packet to the sink; in case a link layer ack is not received, a second attempt is made; if this second attempt also fails, the node infers to be outside of the sink's radio range and goes into a de facto hibernation state until it eavesdrops on unicast transmissions from a potential parent (refer to Figure 13 for an illustration). In the parentless state, LPL continues to check the medium every t_w but takes no action if radio activity is not detected. This approach is employed to fully identify all the sink's neighbors while maintaining the protocol 100% broadcast-free.

3.1.2 *Parent Selection*

Before the sink's neighbors can advertise themselves as viable parents, they employ data path validation [10] to gauge the stability of their link. Data path validation consists of two steps: assessing the route cost and setting the viability flag. To evaluate the route cost, each node measures the ETX to its parent by counting the number of transmissions needed to get one link layer ack from its parent [26, 29]. The cost of a route is computed by adding up the measured ETXs for all the links involved. We use the expression *measured ETX* to clarify that BFC measures the Required Number of Packets [30] to get an ack instead of simply estimating the ETX. Viability as a potential parent is advertised by setting a dedicated flag in the header of data packets. The viability flag is set if v consecutive unicast transmissions are acknowledged at their first attempt. The viability flag is reset as soon as an ack is missed. When a parentless node overhears a unicast transmission with a set viability flag from sink neighbor i, it selects i as its parent and initiates the data path validation process, eventually offering its services as a parent to upstream nodes. This way, routing information naturally propagates upstream with no need for broadcast beacons.

BFC attempts to build solid routes to minimize the need for local route repair. The parameter v represents the data path validation threshold and offers a tradeoff between stability and latency; increasing v makes routes more stable but causes nodes to take longer to join the network. Our empirical results indicate that $v = 3$ is a reasonable choice that works well across the wide variety of testbed scenarios that we tested. We have observed that setting $v < 3$ may occasionally result in instability (such as temporary loops) in networks with challenging connectivity conditions.

Due to the lack of dedicated control traffic, with BFC nodes have no global view of their radio neighborhood, and BFC's parent selection does not necessarily lead to an optimal routing selection, *i.e.*, does not necessarily minimizes the total number of transmissions between a node and the sink. With BFC, a node simply aims to select a *workable* parent that keeps it connected to the network.

3.1.3 *Best Effort Data Delivery*

Once a path towards the sink is found, the data delivery process may begin. In terms of reliability, BFC aims at best effort performance and, like CTP, does not provide end-to-end delivery guarantees. Each packet transmission must be validated by the reception of a link layer ack. The maximum number of retransmissions to an individual parent is denoted by N_{retx} (set to 6 by default). Note that the current parent is dropped after N_{retx} failed unicasts, but the packet is only dropped when the Time To Live (TTL) has expired after a total of $N_{\text{max}} = 32$ retransmissions, similarly to CTP [10]. In BFC, packets can be lost due to link failures (if the TTL is exceeded), buffer overflows (congestion), or false ACKs [31]. Because LPL increases the packet transmission duration, it makes hidden node effects more likely as t_w is increased [12] (recall that instead of sending a single packet, LPL sends a train of packets). To alleviate this problem, BFC jitters transmissions across all nodes at every t_{IPI}. The jitter also ensures that every node gets to hear from every neighbor over time, because it forces the duty cycles of each node pair to shift with respect to each other. At startup nodes draw their jitter from a uniform distribution in [0, m], and at each t_{IPI} they redraw their jitter from a uniform distribution in [0, s]. In general, $m \approx t_{\text{IPI}}$ and $s \ll t_{\text{IPI}}$. In the implementation, we employ the values $m = 2$ min and $s = 1$ sec based on empirical observations.

3.2 Route Maintenance

Route breakage occurs when a node no longer has a valid parent. This may happen due to the vagaries of the wireless channel or due to data traffic conditions, such as with congestion. BFC has mechanisms to detect and recover from these different types of route failures without injecting dedicated control packets.

3.2.1 *Route failure due to channel dynamics.*

Low-power wireless communication is notoriously affected by asymmetric and unreliable links [32]. Link asymmetries are automatically addressed by relying on layer two acknowledgments [26], because links are deemed valid only if transmissions are successful in both directions. On the other hand, if an existing link becomes unavailable, a route breakage is signaled if the maximum number of (re)transmissions is exceeded.

3.2.2 *Route failure due to traffic dynamics.*

Although BFC is designed for ultra-low data rates, congestion may still be an issue in case nodes have a high forwarding load and a weak link to their parents. In this case, the capacity of the outgoing link may not be sufficient to accommodate the received packets. BFC uses two mechanisms to prevent buffer overflows:

- When the buffer occupancy reaches a critical level (half of the buffer size), parent nodes reset their viability flag to inform their neighbors that they no longer offer a viable route.
- When the buffer occupancy reaches a very critical level (80% of the buffer size), link layer acknowledgements are shut down to emulate a route breakage. This form of cross-layer broadcast-free backpressure [33] has an impact on the energy consumption of the children (due to

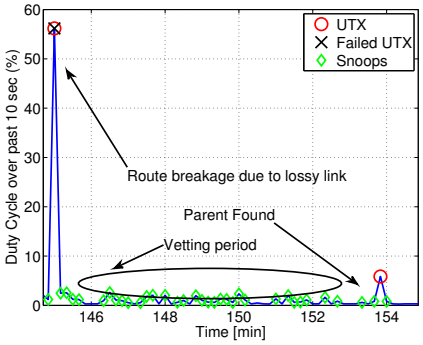


Figure 3. Route breakage (t_{IPI}=5min): parentless nodes typically recover within one t_{IPI}. To avoid instability, they tend to ignore snooped unicasts during the vetting period.

failed retransmissions), but it effectively prevents congestion losses.

Note that broadcast-free backpressure may not be employed if BFC coexists with other protocols, in which case shutting down all layer 2 acks is not an option. This limitations may be overcome in case a radio chip with selective layer 2 acks were to become available.

3.2.3 *Route Repair*

The centerpiece of the route repair process is a reactive mechanism governed by a vetting period. Right after a route breakage, nodes enter a period in which potential parents are subjected to a heavy scrutiny, which is essential to avoid the creation of loops in disconnected subgraphs. A potential parent is only accepted if it offers the same measured ETX as the old one. This conservative position is taken to avoid route instability due to the spatial correlation of link failures. If a link goes down, either due to a link failure or to a parent resetting its viability flag, other nearby links may also be down. Without our conservative approach, nodes could easily connect to siblings that advertise outdated information, potentially forming routing loops. Because of our conservative approach, the formation of loops is unlikely and has never been observed in the course of our experimental campaign. No explicit loop detection scheme (other than verifying that packets do not loop back to the node they originated from) has been employed in the prototype implementation.

If a new parent is found during the vetting period and the outgoing unicast packet is acknowledged, regular operation resumes. If no viable parent is discovered, the node limits its activity to periodic energy checks every t_w. In case no viable parent is found, a unicast packet is sent every t_{seek} to the latest viable parent on record. Every time a unicast transmission is attempted and not acknowledged, the ETX is increased by N_{retx}. Progressively, the increments in ETX loosen the initial constraints of the vetting period allowing nodes with higher ETX to serve as parents. If a parent ceases to be viable and then becomes viable again, its potential child nodes typically rediscover it by overhearing or by eventually trying to transmit to the latest viable parent on record.

The dynamics of the vetting mechanism are illustrated in Figure 3, which shows the instantaneous duty cycle of a node using a fixed window of 10 sec; we purposefully avoid using a sliding window to clearly mark each window with the key events it contains. This same visualization strategy is employed throughout the paper. As in this specific case, our empirical observations show that recovery from route breakage typically happens within one t_{IPI} (5 min in Figure 3), barring exceptional circumstances in which more rounds of t_{seek} are needed. In the prototype implementation, $t_{\text{seek}} = t_{\text{IPI}}$, but in practice it could be made much larger to save more energy during poor connectivity periods.

3.3 Adaptive Low Power Listening

By forgoing broadcast, BFC uses less energy to build and maintain a collection tree. The downside is that parent selection is a relatively uninformed process. Thanks to broadcast beacons, CTP allows nodes to consider the whole range of parenting options available and to consistently choose the parent that offers the lowest ETX to the sink. With BFC, nodes lack a comprehensive knowledge of all the available routing options because they rely on snooped unicasts, and they tend to lock to the most active parents which perform more unicasts than others and are more likely to be eavesdropped on. This process generally leads to an unbalanced tree compared to CTP. To counteract this effect, BFC employs built-in LPL adaptivity [34, 35]. Every node i monitors its forwarding load F_i and adapts its LPL wake up interval accordingly. If i has a heavy load, its parent j is bound to have a heavier load and adapts its LPL wakeup interval (halves it). This way, the cost of i's unicasts is also halved. If i also halves its wake up interval, the cost of the unicasts transmissions of its children also gets halved. Of course, i, j, and any other node that halves its LPL wakeup interval doubles its receive check cost, but the relative impact of receive checks is negligible compared to the impact of unicasts in heavily loaded nodes.

For our prototype implementation, we employ a very simple and relatively mild scheme whereby t_{w} is halved if node i has $B_i > 3$ (*i.e.*, it carries the equivalent of at least two other nodes) and is further halved if $B_i > 10$; t_{w} goes back to the default value if $B_i < 1.5$. These thresholds work well for network setups with upwards of 100 nodes and offered loads of $t_{\text{IPI}} \geq 1$ min, as we verified in the available public testbeds. A thorough study of adaptive LPL strategies to streamline broadcast-free operation will be the subject of future investigations.

3.4 Connectivity

Even though overhearing is not an exhaustive method to identify all neighbors, in our testbed experiments the number of nodes that joined the tree was consistently the same for both CTP and BFC. In Motelab, five nodes consistently failed to join the tree with both protocols, as we will illustrate in Section 4. We now present a simple probabilistic model to understand how eavesdropping leads to good connectivity.

First, let us consider a worst case scenario, when a node has only one potential parent available. The potential parent transmits its packet every t_{IPI}, and as explained in Section 2, the expected duration of a unicast transmission is roughly $t_{\text{w}}/2$. Given that a node wakes up every t_w, the probabil-

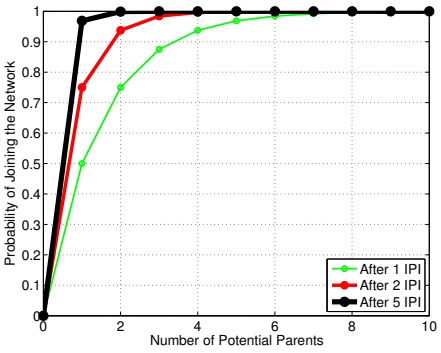


Figure 4. A node can typically join a BFC network within one IPI; it may take longer if the local node density is very low and there are not many potential parents.

ity of overhearing a packet is 1/2 if the link is lossless. If there are n potential parents, the probability of snooping a unicast is $1-0.5^n$. Since BFC jitters unicast transmissions (Section 3.1.3), the overhearing attempts at each IPI can be seen as independent events. Denoting h as the number of IPI intervals, the probability of successfully snooping a packet is given by

$$p_{\text{snoop}} = 1 - 0.5^{n\,h}. \quad (1)$$

Using this model, we can roughly assess how easily nodes can join a BFC network as a function of their local neighborhood density. Figure 4 shows the probability of finding a parent with parameters $t_{\text{IPI}} = 5$ min and $t_{\text{w}} = 2$ sec, which are reasonable for a low-data rate collection application. After five IPIs, the probability of identifying the only potential parent available is above 0.9. The probability is almost one with four potential parents in two IPIs.

It is also important to remark that the non-exhaustive search of overhearing does not lead to suboptimal routes (i.e., routes that are longer than the ones found in CTP). In our evaluation, we did not observe significant differences between the path-lengths formed by CTP and BFC.

3.5 Snapshots of BFC Operation

Figure 5 depicts snapshots of the operation of BFC and CTP on Motelab for a sink's neighbor (left column), a relay (center column) and a leaf (right column); as mentioned in Section 3.2.3, we sample the on time at 10 sec intervals and show the duty cycle over the past 10 sec, purposefully avoiding the use of a sliding window to get an idea of which events happen within each 10 sec time slice. The figure captures the key radio operations that contribute to the duty cycle. For the sake of clarity, and because of their lower footprint in energy consumption, unicast receptions are not shown. In this evaluation, we use t_{IPI} =5 min and we select $t_w = 2$ sec. for CTP and BFC

The trends of our empirical comparison follow the insights of our analytical model. First, sink neighbors benefit the most. Unicast transmissions are plentiful but cheap (due to the sink being always on) and broadcasts dominate the energy consumption in CTP, while in BFC the broadcast energy is spared. Second, in BFC, relays do not benefit much by eliminating broadcasts because unicast transmissions dominate the energy consumption in both protocols. Third, the leaves also benefit from the elimination of broadcasts because they utilize few unicast transmissions, which increases the relative cost of broadcasts.

4 Experimental Evaluation

4.1 Evaluation Methodology

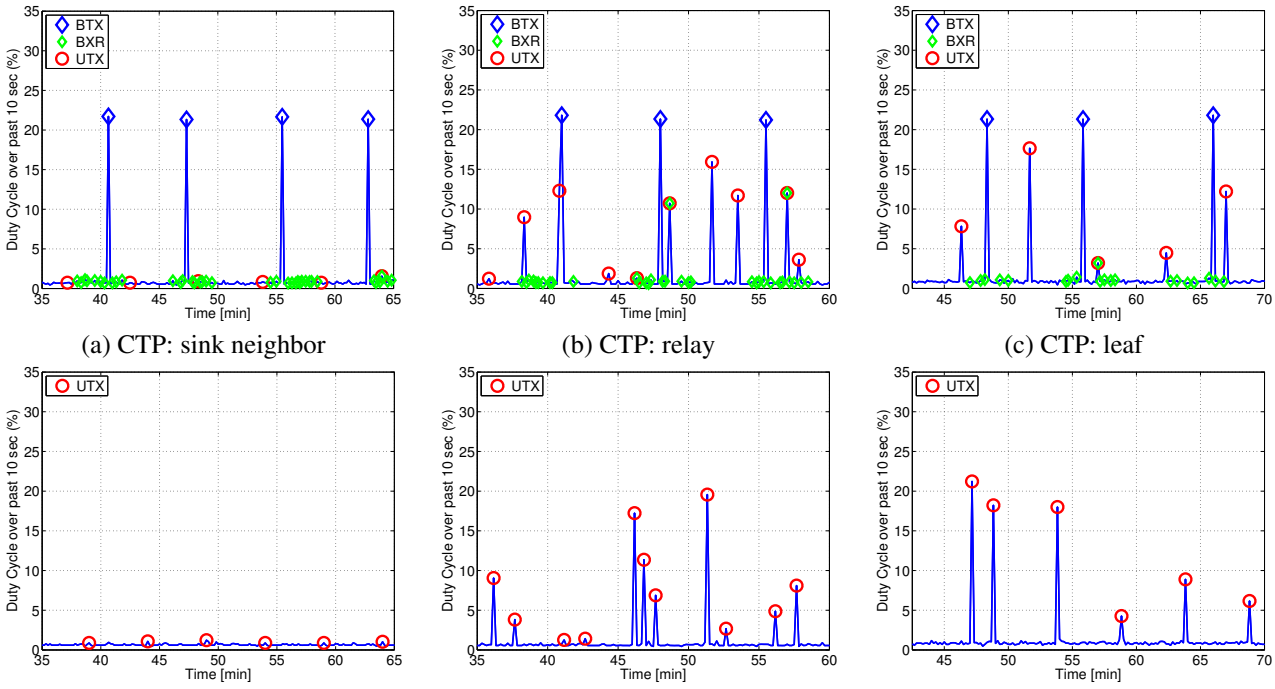
We considered three different testbeds to test our BFC implementation (Motelab [15] at Harvard, Indriya [36] at the National University of Singapore, and Twist [37] at TU Berlin). After evaluations in all three, we found Motelab to be the most challenging testbed due to the comparatively low density and relatively unstable link dynamics. For this reason, the results presented in this paper focus on Motelab. At the time of the experiments (2011-12), there were 69 active nodes in the testbed. A transmit power of 0 dBm on 802.15.4 channel 26 was used for all runs.

We compare BFC's performance to CTP, the de-facto standard for tree-based data collection in low-power wireless sensor networks. We ran several BFC and CTP experiments of 2.5-4 hours each. In our results, we took care of comparing data for periods where both protocols observe similar channel conditions. This step is extremely important to avoid misleading results. It is reported in [38] that, even on the same testbed, the connectivity of the network can change dramatically throughout the day. For as fair a comparison as possible, we measured Γ_i for each node i and computed our statistics over intervals with similar values of $\sum_{i\in\mathcal{N}}\Gamma_i$, where $\mathcal{N}$ denotes the set of nodes in the network.

At the time of the experiments, Motelab had 5 connectivity outliers on channel 26. Such outliers have no connectivity to other nodes and deliver no packets to the sink (no matter which node is chosen as the sink). This is a physical connectivity issue, also pointed out in [10], that is independent of the chosen routing protocol. We did not consider these nodes for the network statistics reported in Section 4.2, but we do report results that are specific to the outliers in Section 4.3.3.

We tested BFC and CTP using a periodic reporting application as in [10], where each node reports data every t_{IPI}. Our key figure of metric is the duty cycle, which we measure as a proxy for energy consumption using software-based online estimation [27] (implemented in TinyOS by measuring the on time of the radio). In terms of delivery rate (number of packets delivered to the sink over number of injected packets) we did not observe a significant difference compared to CTP; both protocols generally provided delivery rates above 0.99 across all operating points (other than the outliers, whose reliability was equally poor for both protocols, as expected). The same goes for steady-state throughput, although BFC incurs in a significant latency at startup, as illustrated in Section 4.3.4.

In Section 4.2 we focus on the steady-state performance and do not consider the set up phase for either protocol. With CTP, nodes use broadcast heavily at startup; in the default version available in the TinyOS tree in early 2012, the Trickle timer initializes at $T_{\text{m}} = 2^6 = 64$ ms and doubles its interval until it stabilizes at $T_{\text{M}} = 2^{19}\text{ms} \approx 8.7$ min; consider-



(a) CTP: sink neighbor (b) CTP: relay (c) CTP: leaf

(d) BFC: sink neighbor (e) BFC: relay (f) BFC: leaf

Figure 5. Radio activity for three representative classes of nodes with BFC and CTP (both with $t_w = 2$ sec) in Motelab: sink's neighbors (left column), relays (center column), and leaves (right column). Notice the absence of broadcast transmissions and receptions in the BFC nodes. The relative cost of broadcast is different for the different types of nodes. The sink neighbors benefit the most because their unicasts are cheap. Relays do not benefit much because unicasts dominate the costs. Leaves benefit significantly because unicasts are infrequent.

ing CTP's set-up phase for the energy results would unfairly favor BFC. CTP's startup broadcast footprint is only considered in the context of node insertion in 4.3.1. In the results reported in Sections 4.2 and 4.2.3, $T_{\text{IBI}} = T_{\text{M}}$ holds for CTP. With BFC, startup is a slower process with an ultra-low energy footprint; BFC's fundamental tradeoff between latency and energy usage is the object of Section 4.3.4.

Note that BFC uses adaptive LPL by default to counteract its load imbalance; CTP has no need for adaptive LPL because its tree is generally more balanced. Adaptive LPL should not be viewed as an add-on feature of BFC, but it represents a core component that cannot be decoupled from the rest of the system.

4.2 Performance as a function of the LPL Wakeup Interval

We begin by studying the performance of BFC as we vary the LPL wakeup interval t_w. We set $t_{\text{IPI}} = 5$ min, a reasonable value for low data rate applications, and we pick a node in the middle of the network as our sink. Since BFC uses adaptive LPL, for BFC the value of t_w represents the maximum wakeup interval employed by a node. In several parts of this section, we report the median and mean duty cycle.

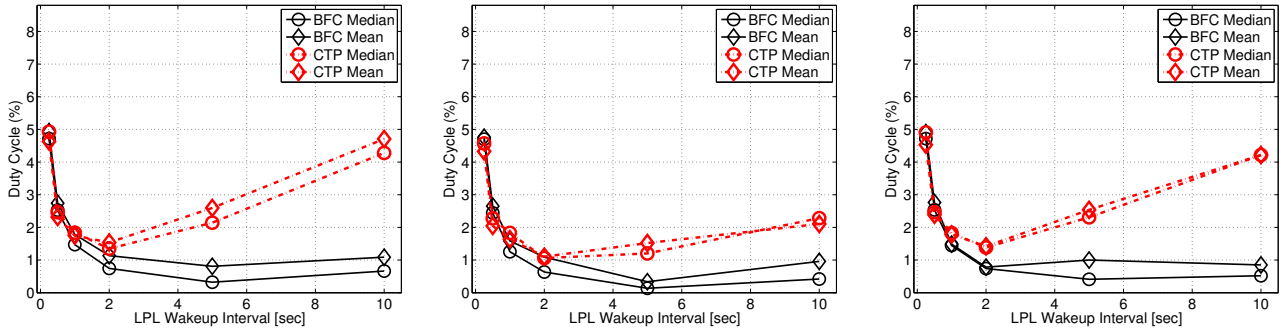
4.2.1 Median and mean for all nodes.

Figure 6(a) shows the median and mean duty cycle for various different values of the LPL wakeup interval. CTP has an optimal wakeup interval in the $[1,2]$ sec range; for higher values, its duty cycle increases due to the increasing cost of broadcasts, which grows linearly with t_w. On the other hand, BFC's duty cycle exhibits a much wider and flatter optimal operating region of t_w. The breakdown of the node classes with the sink in the middle was roughly 27% sink's neighbors, 21% relays, and 52% leaves. Figures 6(b) and 6(c) show the mean and the median duty cycle for, respectively, the sink neighbors (for which unicasts are cheaper) and the leaves (for which unicasts are rare). The duty cycle curve for the relays remains flat for increasing values of t_w because of the use of adaptive LPL.

4.2.2 Duty cycling savings.

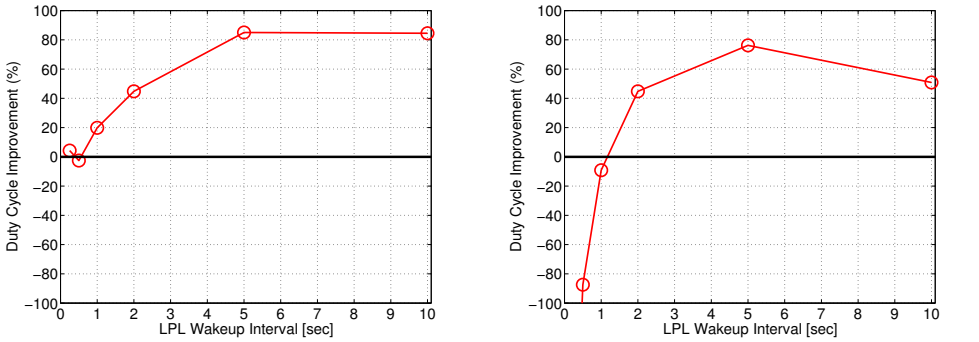
Figure 7(a) shows the median duty cycle improvement of BFC. This figure is derived from Figure 6 by normalizing the results with respect to the performance of CTP. For example, across all connected nodes and with t_w =5 sec, we observe in Figure 6(a) that the median duty cycle for CTP is just above 2%, while the median duty cycle for BFC is in the ballpark of 0.4%, which leads to a reduction of approximately 80% as depicted in Figure 7(a). The importance of this figure is that it quantifies the reduction in duty cycle obtained by BFC. At $t_w = 1$ sec, we obtain median reductions in the order of 25%.

In Figure 7(b) we take a more conservative approach for the comparison. We normalize the performance of BFC with respect to the *optimal* duty cycle in CTP. For example, at $t_w =$ 5 sec, we compare the median duty cycle of BFC across all



(a) All connected nodes. (b) Sink's neighbors. (c) Leaves.

Figure 6. Motelab: mean and median duty cycle with $t_{\mathrm{IPI}} = 5$ min and various values of t_w.



(a) At each operating point. (b) With respect to the optimal operating point.

Figure 7. Percentage improvement of the median duty cycle with BFC.

connected nodes (about 0.4%) with the minimum duty cycle achieved by CTP across all settings of t_w (around 1.5% with t_w =2 sec); the median duty cycle at this operating point improves by as much as 75%. Even in this unfavorable comparison, BFC performs better in most cases, except when $t_w < 1$ sec, in which case broadcast is so cheap that avoiding it provides no benefits.

4.2.3 Impact of the network structure.

The results shown so far have been obtained by placing the sink in the center of the testbed, which is expected to provide the widest possible range of connectivity options. Figure 8(a) shows a set of results obtained over all nodes with the sink placed at the edge of the network (on the uppermost floor in Motelab); in these experiments, the breakdown of the node classes was roughly 13% sink's neighbors, 27% relays, and 60% leaves, as reported in 2.4. We focus on the narrower t_w range [0.5, 5] sec. The optimal operating range in terms of t_w is, again, much wider than with CTP, but now we observe a wider spread between the mean and the median duty cycles: with fewer connectivity options, the energy consumption across all nodes becomes more unbalanced, and nodes that offer good connectivity are bound to be burdened with a higher load.

Even in this scenario, however, BFC provides a significant improvement over CTP. Figure 8(b) shows the "conservative comparison" between BFC and CTP (i.e. utilizing only CTP's optimal duty cycle). Note that load imbalance is not detrimental per se: if there is enough connectivity, there are leaves at all levels of hop count that can take over once unbalanced relays get used up. In any case, more advanced adaptive LPL schemes can be employed to counteract the imbalance.

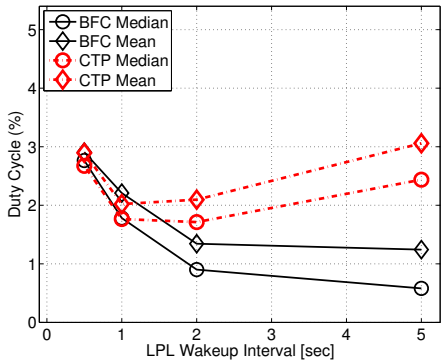
4.3 Performance outside of steady state

We now consider BFC's behavior with respect to node insertion and removal, poor connectivity conditions, and network startup.

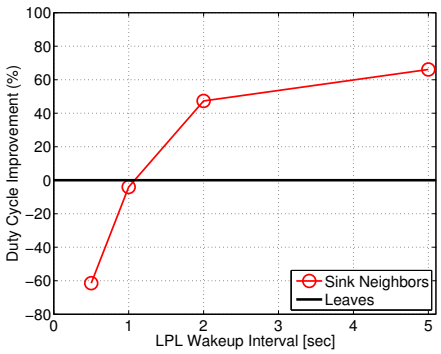
4.3.1 Node insertion

In principle, if the topology is stable, CTP could achieve quasi broadcast-free operation by letting $T_{\mathrm{M}} \to \infty$, or at least boost T_M and thus reduce BFC's competitive edge. The complete lack of broadcast, however, greatly favors BFC in dynamic scenarios where nodes get added or simply reboot.

Figure 9(a) shows what happens when a node is added to an existing CTP network, or, equivalently, when a node reboots: the node in question aggressively sends out broadcast as dictated by its Trickle timer in an effort to *pull* a route from its neighbors, *i.e.*, to receive a beacon with valid routing information. This way, a route is quickly discovered at the cost of several broadcast transmissions and receptions, resulting in an on time of $t_w(\log_2 T_{\mathrm{M}} - \log_2 T_{\mathrm{m}} + 1) + t_{\mathrm{rx}} N_i$ for node i.

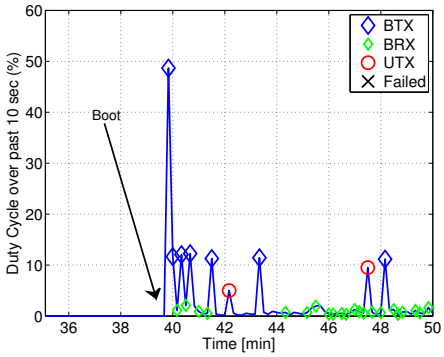


(a) Mean and median duty cycles.

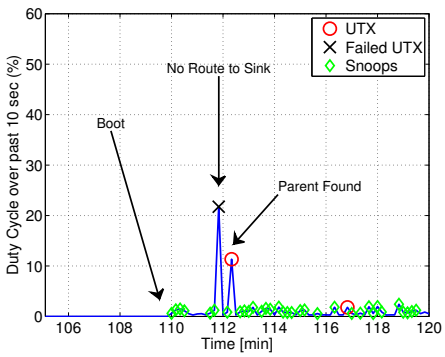


(b) Percentage improvement of the median duty cycle with respect to CTP's optimal operating point

Figure 8. Motelab, sink placed at the edge of the network: performance with $t_{\mathrm{IPI}} = 5$ min and various values of t_w.



(a) CTP



(b) BFC

Figure 9. Consequences of node insertion with CTP and BFC in Motelab with t_w =1 sec and t_{IPI} =5 min.

With BFC, node insertion is much cheaper in energy terms. Figure 9(b) shows a trace from a Motelab experiment also with t_w =1 sec and t_{IPI} =5 min where a node is added to an existing BFC network. The node is not within the sink's reach, so there is a waste of kT_w. After that, a route is picked up (typically within t_{IPI} if the network is already formed, as in this case) with no further energy usage other than the one for unicast eavesdrops (snoops), which, as our model shows and Figure 9 confirms, have a negligible footprint compared to broadcast transmissions. Without accounting for broadcast and unicast receptions, BFC's node insertion is $(\log_2 T_{\mathrm{M}} - \log_2 T_{\mathrm{M}} + 1)/k$ times cheaper than CTP's; with the default settings, this translates to a factor of 7. Also note that, in the case of the sink neighbors, BFC's node insertion has zero energy overhead.

4.3.2 *Node removal*

Node removal also has a small footprint with BFC. Figure 10 shows an example of what happens to a child node in Motelab when a very active relay is shut down. Once the maximum number of retransmissions N_{retx} is reached (6 in the default implementation employed here), route breakage is inferred and reacted to as we have seen in Section 3.1. The cost incurred with BFC is dominated by $N_{\mathrm{retx}} t_w$ plus the comparatively negligible cost of snooping unicasts. In such scenarios, CTP would switch to another parent, and if necessary send out a beacon to *pull* a route from a neighbor. Differently from node insertion, it is not easy to evaluate which approach is more energy-efficient due to the many variables at play (local node density, topological importance of the removed node, number of CTP retransmissions before switching, availability of an alternative parent in CTP's neighbor table, value of BFC's N_{retx}, value of t_w, among others).

4.3.3 *Poor connectivity .*

We have mentioned the presence of five connectivity outliers in Motelab that do not have any usable connections to the rest of the network on channel 26. Note that they do work on other channels, where other sets of nodes behave as outliers. Up to this point, we have consistently removed the outliers from all the results we presented; in this Section, however, we wish to focus on them to see how BFC handles instances of severely poor connectivity. Figure 11 shows the mean and median duty cycle achieved by the outliers with the sink in the central area of the testbed (similar trends are observed no matter where the sink is placed). In CTP, the outliers try aggressively to find a new parent because the Trickle timer commands intense broadcast activity in the hope to *pull* a route. On the other hand, BFC simply gives up for intervals equal to t_{seek} before they unicast to the

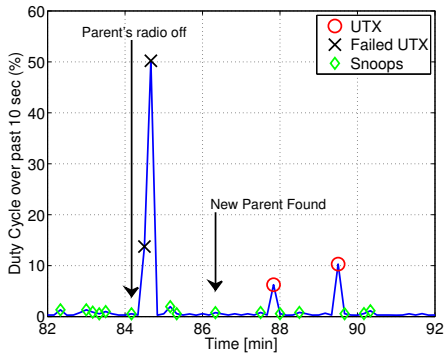


Figure 10. Consequences of node removal with BFC. Similarly to the case of route breakage due to lossy links, node removal is viewed as parent loss, and a new parent can be typically found within t_{IPI}.

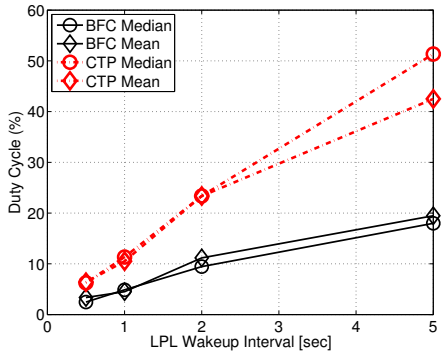


Figure 11. Motelab: mean and median duty cycle of the connectivity outliers with $t_{\text{IPI}} = 5$ min and various values of t_w.

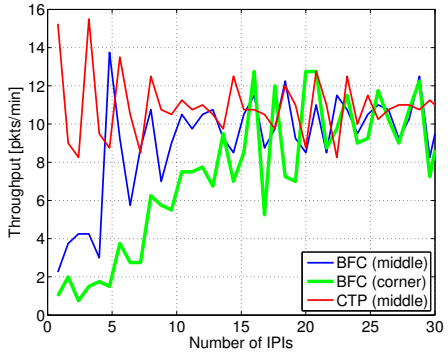


Figure 12. Latency vs. energy consumption tradeoff: at steady-state, BFC's throughput is similar to CTP's.

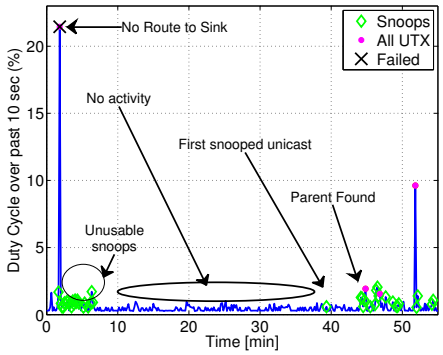


Figure 13. The tree discovery time is linear with the number of hops. Here (Motelab with $t_{\text{IPI}} = 5$ min and $d = 5$), a parent is elected after $8.5t_{\text{IPI}}$.

last node that has been heard from (note that the outliers do hear from other nodes from time to time, but can never reach any of them). The attempts of both protocols are useless because there is no connectivity, the end result is the same (the outliers never discover a route and no packets are delivered), but BFC's more patient approach halves CTP's duty cycle. Note that the prototype implementation employs the liberal value $t_{\text{seek}} = t_{\text{IPI}}$; individual applications can set t_{seek} to much larger values depending on their latency vs energy needs. This is an important result because it has been shown that even well-connected networks can have periods of poor connectivity [38], in which case BFC's approach would preserve the nodes' energy much more than CTP's.

4.3.4 Latency

The main trade-off of BFC is the longer time required to form the data gathering tree. Figure 12 depicts the time required by BFC and CTP, in Motelab, to reach a stable throughput for t_{IPI}=5 min. We use the evolution in throughput as a proxy for connectivity. The throughput reaches a steady state when all nodes have joined the tree, other than the outliers. The figure shows three curves, one for CTP with a sink located in the central area of the network, and the other two for BFC with sinks located in the middle and at the edge of the network. With the sink in the central area, the average hop count for both, CTP and BFC, is 2, while the depth is 4 (the farthest nodes from the sink are 4 hops away). With the sink at the edge of the network, the average hop count is 3.5, and the depth is 9. CTP reaches its steady-state throughput within the first IPI thanks to the use of broadcast; CTP's cost field propagation is so rapid that the same result is achieved with the sink at the network's edge (not shown). On the other hand, BFC incurs a significant delay that depends on three parameters: the depth of the tree d, the validation parameter v (Section 3.1.2) and the inter-packet interval t_{IPI}. Consider the leaf node that is the farthest from the sink whose time-averaged depth is equal to d. At each hop, a potential parent needs to have v continuous successful unicast transmissions before disseminating the routing information. Note that v continuous transmission does not imply a wait of $v \times t_{\text{IPI}}$. In fact, as a parent is found, the first packet is sent with just a short delay (jitter), and

the waiting time until the second packet follows a uniform distribution in $[0, t_{\text{IPI}}]$; for the following packets, the inter-packet interval becomes t_{IPI}. Hence, the farthest leaf node will be able to choose a parent approximately after a delay uniformly distributed in $[d \times (v-2) \times t_{\text{IPI}}, d \times (v-1) \times t_{\text{IPI}}]$. The expected value of the parent discovery delay is therefore $d \times (v-3/2) \times t_{\text{IPI}}$, which translates to 6 t_{IPI} with the sink in the middle and $13.5 t_{\text{IPI}}$ with the sink at the edge, as confirmed by Figure 12.

Figure 13 shows a Motelab trace for a BFC node with $t_{\text{IPI}} = 5$ min and $t_w = 1$ sec; because the node is located five hops away from the sink, its parent discovery delay is uniform in [25, 50] min, and in fact the value measured in the experiment shown in Figure 13 is approximately 43 min. In absolute terms, such delays are significant, but they are acceptable in the context of low-power sensor networking applications where energy conservation is essential and lifetimes are expected to be long (in the order of weeks or more). Also note that these delays are only incurred at the inception of network operation; once the tree has been formed, new nodes typically join the network within one IPI. In other words, these should be viewed as one-time delays that are experienced when the tree has to be built from scratch, but they are not applicable in the case of recovery from route breakage, as we have seen in Section 4.3.2.

Because BFC's latency grows linearly with the network depth, in large networks of hundreds or thousands of nodes BFC should be used in a hierarchical fashion. For instance, as proposed in [39] for Glossy-based collection [40], disjoint sets of nodes could be assigned to different channels and thus form different BFC clusters of manageable depth reporting to one cluster-head.

5 Related Work

Lately, the low-power wireless community has devoted a lot of attention to the comparatively high cost of broadcast communication on top of a duty-cycled link layer. Various solutions have been proposed, ranging from beacon coordination [5] to unified broadcasts [13] to the *politecast* primitive [14], which makes it possible for nodes to independently decide whether to stay awake to receive *politecast* messages. To the best of our knowledge, BFC is the first collection protocol that avoids the use of broadcast traffic by design. The idea of route discover through snooping is not new and was employed in [41] in a quasi-broadcast-free fashion (broadcast was only used at startup); however, a detailed quantitative exploration and evaluation of the idea was not provided. Most notably, the approach in [41] requires coordinated duty cycles, while BFC supports asynchronous low-power listening.

There exists a significant body of work on data-driven link estimation [42], which represents the centerpiece of link estimation schemes such as Four-Bit [26] and DUCHY [29]. Since BFC forgoes using broadcast traffic, data-driven link estimation is the natural choice. Similarly to [26], BFC measures the Expected Number of Transmissions (ETX [25]) by counting link layer acknowledgements. This is the data path validation mechanism at the heart of many collection protocols [10, 16, 11, 21, 22].

Link-layer duty cycling has also been studied extensively. The Low Power Listening protocol (LPL) was initially presented as part of B-MAC [18]. It is the most popular technique to counteract the idle listening problem and shift the burden of communication from the receiver to the transmitter. The basic idea of LPL is to send a packet train (long preamble) to match the wakeup interval of the intended receiver. A key refinement to basic LPL/B-MAC is X-MAC [20], which makes it possible for transmitters to cut their packet trains short as soon as a layer 2 acknowledgement is received. BoX-MAC [19] is the standard TinyOS MAC and merges X-MAC with basic B-MAC. Our BFC prototype implementation is built on top of BoX-MAC. We have presented a simple first-order model of LPL to estimate the energy footprint of broadcast on different categories of nodes. Our model is based on ideas and concepts presented in [18], [4], and [16].

In its present form, BFC takes advantage of specific features of BoX-MAC (its packet train and its purely asynchronous nature). At the moment, it remains unclear whether BFC may be applied to receiver-based MAC protocols [43, 44]. Investigating the use of BFC on top of different MACs is a research effort that will be considered in our future work.

In this paper, BFC needs to counteract the inherent load imbalance introduced by the tree construction process where, if possible, nodes tend to stick to parents, and a few parents tend to get overloaded. Explicit load balancing is not used because it may lead to suboptimal routes with hidden costs in terms of retransmissions [11]. Instead, BFC uses a basic form of adaptive LPL to achieve energy balancing without explicit load balancing. The advantages of adaptive LPL have only been explored by a few studies. An initial contribution to this important topic was given by [34], where ALPL (Adaptive LPL) is presented and implemented on MICA2 motes. The basic idea is to dynamically rebalance the burden of communication between transmitter and receiver by letting the LPL wakeup interval vary between 20 ms (the default setting of B-MAC) and 200 ms. Given that BoX-MAC uses a default LPL wakeup interval of 500 ms, ALPL's calibration of the wakeup interval range appears rather conservative. Schemes for LPL adaptivity have been tackled within the ZeroCal protocol [35] and the IDEA framework for energy-aware routing [45]; the techniques in IDEA and ZeroCal could be used to further boost the effectiveness of BFC. Recently, there has been an effort on the runtime parameter adaptation of low-power MAC parameter that has resulted in the pTunes framework [46]. pTunes makes model-based predictions of the impact of the MAC on the network-wide performance given the current network state and selects the MAC parameters at run time whose predicted performance matches the application requirements. pTunes and BFC are orthogonal efforts toward network lifetime extension that could both benefit from each other. On one hand, pTunes could be employed to optimize LPL adaptivity in BFC; on the other hand, BFC makes it more energy-efficient to use longer LPL wakeup intervals and would give more freedom of action and flexibility to pTunes.

BFC can be viewed as an extreme version of broadcast mitigation approaches like [5, 13, 14], because it tackles the

broadcast problem by removing its source. A completely different approach in the same direction is represented by collection protocols [39, 47] based on the Glossy paradigm [40], which only employs broadcast but dramatically reduces its cost by means of ultra-tight time synchronization that makes it possible to leverage constructive interference.

Related to BFC are the recent efforts on applying ExOR-style opportunistic routing [48] to low-power wireless, namely the ORW protocol presented in [23]. The basic principle of OWR is that when a packet is transmitted by node *i*, it gets forwarded by the first node *j* that wakes up and receives it and provides routing progress toward the destination (in which case *j* sends a layer two ack to *i*). In case a packet is received by multiple nodes, a lightweight coordination scheme is employed to determine a unique forwarder and avoid the propagation of duplicates. OWR uses broadcast very sparingly (for instance, at startup) and adopts an anycast approach to data collection. The testbed-based evaluation in [23], where OWR is benchmarked against CTP, shows a reduction of the duty cycle of the same order of the one achieved by BFC. We plan to address an in-depth comparison of BFC and OWR in our future work; merging ideas from both protocols might lead to a more energy-efficient solution with less latency. Another related effort is the Backpressure Routing Protocol (BRP)[49]; though not broadcast-free, BRP employs a data-driven approach whereby packets are forwarded to the neighbor with the lowest queue level. BRP addresses the opposite end of the collection design space compared to BFC, because it can only be applied when the system is saturated and nodes always have packets to forward.

6 Conclusion

We have challenged the conventional notion that broadcast traffic is necessary to steer the operation of multi hop data collection by presenting the design, implementation, and experimental evaluation of the Broadcast-Free Collection (BFC) protocol, to the best of our knowledge the first effort in this particular direction. Our motivation stems from the recent work [5, 13, 14] by the low-power wireless community on how to manage the extra cost of broadcast traffic compared to unicast on top of duty-cycled operation. Rather than concentrating on containing its extra cost, we take a radical approach and show that broadcast can be completely eliminated. In the specific case of data collection, which represents the focus of this paper, we show that route discovery and maintenance can be carried out solely based on unicast traffic, as long as the link layer acknowledgements are available. The benefits in energy terms are significant, both at steady state and in the presence of anomalies. The key benefit is that without broadcast longer wakeup intervals can be employed, and nodes can be allowed to sleep longer. Our evaluation on Motelab has shown percentage duty cycle benefits in the double digits. Nodes within radio range of an always on sink benefit the most because broadcast is by far their largest source of energy consumption; depending on the operating point, BFC cuts their duty cycle by upwards of 70% for the benefit of the whole network.

Acknowledgements

We wish to thank Omprakash Gnawali and Olaf Landsiedel for their invaluable input. We also wish to thank our shepherd, Prabal Dutta, the anonymous reviewers, and the maintainers of the testbeds. We gratefully acknowledge the support of GEBERT RÜF STIFTUNG under the SMACS Project (043-10) within the BREF program, the Mercatur Research Center Ruhr under the SEVERE project (Pr-2011-0014), and the European Commission under the projects SCAMPI (ICT-258414) and PLANET (FP7-257649-ICT-2009-5) and the Network of Excellence CONET (FP7-2007-2-224053).

7 References

[1] J. Wieselthier, G. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *19th IEEE International Conference on Computer Communications (INFOCOM '00)*, Tel Aviv, Israel, Mar 2000.

[2] M. Cagalj, J. Hubaux, and C. Enz. Minimum-energy broadcast in all-wireless networks: Np-completeness and distribution issues. In *8th Annual International Conference on Mobile Computing and Networking (MobiCom '02)*, Atlanta, GA, USA, Sep 2002.

[3] S. Katti, H. Rahul, Wenjun Hu, D. Katabi, M. Medard, and J. Crowcroft. Xors in the air: Practical wireless network coding. *IEEE/ACM Transactions on Networking*, 16(3):497 –510, June 2008.

[4] K. Langendoen and A. Meier. Analyzing MAC protocols for low data-rate applications. *ACM Transactions on Sensor Networks*, 7(1):1–34, November 2010.

[5] A. Dunkels, L. Mottola, N. Tsiftes, F. Österlind, J. Eriksson, and N. Finne. The Announcement Layer: Beacon Coordination for the Sensornet Stack. In *8th European Conference on Wireless Sensor Networks (EWSN '11)*, Bonn, Germany, February 2011.

[6] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks. In *1st USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI '04)*, San Francisco, CA, USA, March 2004.

[7] K. Lin and P. Levis. Data discovery and dissemination with dip. In *7th international conference on Information processing in sensor networks (IPSN '08)*, St Louis, MO, USA, 2008.

[8] P. Dutta and D. Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *6th ACM conference on Embedded network sensor systems (SenSys '08)*, Raleigh, NC, USA, 2008.

[9] A. Kandhalu, K. Lakshmanan, and R. Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN' 10)*, pages 350–361, Stockholm, Sweden, April 2010.

[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, Berkeley, CA, November 2009.

[11] D. Puccinelli and M. Haenggi. Reliable Data Delivery in Large-Scale Low-Power Sensor Networks. *ACM Transactions on Sensor Networks*, Nov. 2010.

[12] U. Raza, A. Camerra, A. Murphy, T. Palpanas, and G. P. Picco. What Does Model-Driven Data Acquisition Really Achieve in Wireless Sensor Networks? In *10th IEEE International Conference on Pervasive Computing and Communications (PerCom '12)*, Lugano, Switzerland, March 2012.

[13] M. Hansen, R. Jurdak, and B. Kusy. Unified broadcast in sensor networks. In *10th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'11)*, Chicago, IL, USA, April 2011.

[14] M. Lundén and A. Dunkels. The Politecast Communication Primitive for Low-Power Wireless. *ACM SIGCOMM Computer Communication Review*, April 2011.

[15] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: a Wireless Sensor Network Testbed. In *4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, Los Angeles, CA, USA, April 2005.

[16] J. Hui and D. Culler. IP is dead, long live IP for wireless sensor networks. In *6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, November 2008.

[17] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, pages 1567–1576, New York City, NY, USA, June 2002.

[18] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *2nd ACM Conference on Embedded Networked Sensor Systems (SenSys '04)*, Baltimore, MD, USA, November 2004.

[19] D. Moss and P. Levis. BoX-MACs: Exploiting Physical and Link Layer. Technical Report 08-00, Stanford University, 2008.

[20] M. Buettner, G. Yee, E. Anderson, and R. Han. X-MAC: a Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks. In *4th ACM Conference on Embedded Networked Sensor Systems (SenSys '06)*, Boulder, CO, USA, November 2006.

[21] J. Ko, S. Dawson-Haggerty, O. Gnawali, D. Culler, and A. Terzis. Evaluating the Performance of RPL and 6LoWPAN in TinyOS. In *Workshop on Extending the Internet to Low power and Lossy Networks (IP+SN '11)*, Chicago, IL, USA, April 2011.

[22] J. Ko, J. Eriksson, N. Stiftes, S. Dawson-Haggerty, A. Terzis, A. Dunkels, and D. Culler. ContikiRPL and TinyRPL: Happy Together. In *Workshop on Extending the Internet to Low power and Lossy Networks (IP+SN '11)*, Chicago, IL, USA, April 2011.

[23] O. Landsiedel, E. Ghadimi, S. Duquennoy, and M. Johansson. Low Power, Low Delay: Opportunistic Routing meets Duty Cycling. In *11th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '12)*, Beijing, China, April 2012.

[24] M. Sha, G. Hackmann, and C. Lu. Energy-Efficient Low Power Listening for Wireless Sensor Networks in Noisy Environments. In *The 31st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '06)*, Orlando, FL, USA, April 2012.

[25] D. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, San Diego, CA, USA, 2003.

[26] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four-Bit Wireless Link Estimation. In *6th Workshop on Hot Topics in Networks (HotNets-VI)*, Atlanta, GA, November 2007.

[27] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He. Software-based online energy estimation for sensor nodes. In *4th Workshop on Embedded Networked Sensors (EmNets '07)*, Cork, Ireland, June 2007.

[28] A. Rowe, R. Mangharam, and R. Rajkumar. RT-link: A time-synchronized link protocol for energy-constrained multi-hop wireless networks. In *3rd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON '06)*, Reston, VA, USA, September 2006.

[29] D. Puccinelli and M. Haenggi. DUCHY: Double Cost Field Hybrid Link Estimation for Low-Power Wireless Sensor Networks. In *5th Workshop on Embedded Networked Sensors (HotEmNets '08)*, Charlottesville, VA, USA, June 2008.

[30] A. Cerpa, J. Wong, M. Potkonjak, and D. Estrin. Temporal Properties of Low Power Wireless Links: Modeling and Implications on Multi-Hop Routing. In *4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, Los Angeles, CA, USA, April 2005.

[31] M. Wachs, J. Choi, J. Lee, K. Srinivasan, Z. Chen, M. Jain, and P. Levis. Visibility: A New Metric for Protocol Design. In *5th ACM Conference on Embedded Networked Sensor Systems (SenSys '07)*, Sydney, Australia, November 2007.

[32] M. Zuniga and B. Krishnamachari. An Analysis of Unreliability and Asymmetry in Low-Power Wireless Links. *ACM Transactions on Sensor Networks*, 3(2):1–30, June 2007.

[33] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating Congestion in Wireless Sensor Networks. In *2nd ACM Conference on Embedded Networked Sensor Systems (SenSys '04)*, Baltimore, MD, November 2004.

[34] R. Jurdak, P. Baldi, and C. Lopes. Adaptive Low Power Listening for Wireless Sensor Networks. *IEEE Transactions on Mobile Computing*, 6(8), August 2007.

[35] G. Challen, J. Waterman, and M. Welsh. IDEA: Integrated Distributed Energy Awareness for Wireless Sensor Networks. In *8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, San Francisco, CA, USA, June 2010.

[36] M. Doddavenkatappa, M.C. Chan, and A.L. Ananda. Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed. In *7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom '11)*, Shanghai, China, April 2011.

[37] V. Handziski, A. Koepke, A. Willig, and A. Wolisz. TWIST: A Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks. In *2nd International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (REALMAN '06)*, Florence, Italy, 2006.

[38] D. Puccinelli, O. Gnawali, S. Yoon, S. Santini, U. Colesanti, S. Giordano, and L. Guibas. The Impact of Network Topology on Collection Performance. In *8th European Conference on Wireless Sensor Networks (EWSN '11)*, Bonn, Germany, February 2011.

[39] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Low-Power Wireless Bus. In *10th ACM Conference on Embedded Networked Sensor Systems (SenSys '12)*, Toronto, Canada, November 2012.

[40] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient Network Flooding and Time Synchronization with Glossy. In *10th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '11)*, Chicago, IL, USA, April 2011.

[41] G. Barrentxea, F. Ingelrest, G. Schaefer, and M. Vetterli. SensorScope: Out-of-the-Box Environmental Monitoring. In *7th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '08)*, St Louis, MO, USA, April 2008.

[42] H. Zhang, L. Sang, and A. Aroraa. Comparison of Data-driven Link Estimation Methods in Low-power Wireless Networks. *IEEE Transactions on Mobile Computing*, 9(11), November 2010.

[43] Y. Sun, O. Gurewitz, and D. Johnson. RI-MAC: A Receiver-Initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Trafc Loads in Wireless Sensor Networks. In *6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, Rayleigh, NC, November 2008.

[44] P. Dutta, S. Dawson-Haggerty, Y. Chen, C. Liang, and A. Terzis. A-MAC: Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless. *ACM Transactions on Sensor Networks*, Aug. 2012.

[45] A. Meier, M. Woehrle, M. Zimmerling, and L. Thiele. ZeroCal: Automatic MAC Protocol Calibration. In *6th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2010)*, Santa Barbara, CA, USA, June 2010.

[46] M. Zimmerling, F. Ferrari, L. Mottola, T. Voigt, and L. Thiele. pTunes: Runtime Parameter Adaptation for Low-power MAC Protocols. In *11th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '12)*, Beijing, China, April 2012.

[47] F. Ferrari, M. Zimmerling, L. Thiele, and L. Mottola. The Bus goes Wireless: Routing-Free Data Collection with QoS Guarantees in Sensor Networks. In *4th International Workshop on Information Quality and Quality of Service for Pervasive Computing (IQ2S '12)*, Lugano, Switzerland, March 2012.

[48] S. Biswas and R. Morris. ExOR: Opportunistic Multi-Hop Routing for Wireless Networks. In *Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '05)*, Philadelphia, PA, USA, August 2005.

[49] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali. Routing without routes: The backpressure collection protocol. In *9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'10)*, Stockholm, Sweden, April 2010.