Opportunistic Routing in Low Duty-Cycled Wireless Sensor Networks

Euhanna Ghadimi, KTH - Royal Institute of Technology, Sweden
Olaf Landsiedel, Chalmers University of Technology, Sweden
Pablo Soldati, Huawei Technologies Sweden AB
Simon Duquennoy, SICS - Swedish Institute of Computer Science, Sweden
Mikael Johansson, KTH - Royal Institute of Technology, Sweden

Opportunistic routing is widely known to have substantially better performance than unicast routing in wireless networks with lossy links. However, wireless sensor networks are heavily duty-cycled, i.e. they frequently enter sleep states to ensure long network life-time. This renders existing opportunistic routing schemes impractical, as they assume that nodes are always awake and can overhear other transmissions. In this paper we introduce ORW, a practical opportunistic routing scheme for wireless sensor networks. ORW uses a novel opportunistic routing metric, EDC, that reflects the expected number of duty-cycled wakeups that are required to successfully deliver a packet from source to destination. We devise distributed algorithms that find the EDC-optimal forwarding and demonstrate using analytical performance models and simulations that EDC-based opportunistic routing results in significantly reduced delay and improved energy efficiency compared to the traditional unicast routing. We compare the performance of the ORW protocol with other alternatives in both simulations and testbed-based experiments. Our results show that ORW reduces radio duty cycles on average by 50% (up to 90% on individual nodes) and delays by 30% to 90% when compared to the state of the art.

Categories and Subject Descriptors: C.2.1 [**Network Architecture and Design**]: Wireless Communication; C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms: Design, Algorithms, Performance, Experimentation

Additional Key Words and Phrases: Wireless sensor networks, Energy Efficiency, Opportunistic Routing, Duty cycle

ACM Reference Format:
E. Ghadimi, O. Landsiedel, P. Soldati, S. Duquennoy, and M. Johansson. 2013. Opportunistic Routing in Low Duty-Cycled Wireless Sensor Networks. ACM Trans. Embedd. Comput. Syst. 0, 0, Article 0 (20xx), 35 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

In Wireless Sensor Networks (WSNs), forwarding of packets to their intended destination is typically done in a two-step process: first, the routing protocol determines the next hop node using a routing metric, often computed based on data from link estimators, and information about the routing progress offered by neighboring nodes; second, the MAC protocol waits for the intended next-hop node to wake up and to successfully receive the packet.

In this paper, we depart from this unicast design paradigm. Instead, we transmit packets opportunistically in a fashion tailored to duty-cycled sensor networks: A packet is forwarded by the first awoken neighbor that successfully receives it and offers routing progress towards the destination

This work was partially funded by the Swedish Research Council (VR) and the Swedish Foundation for Strategic Research (SSF). Author's addresses: E. Ghadimi, and M. Johansson, School of Electrical Engineering, KTH - Royal Institute of Technology, Sweden, E-mails:{euhanna, mikaelj}@ee.kth.se; O. Landsiedel, Computer Science and Engineering, Chalmers University of Technology, Göteborg, Sweden, E-mail: olafl@chalmers.se; P. Soldati, Huawei Technologies Sweden AB, E-mail: pablo.soldati@huawei.com; S. Duquennoy, The Swedish Institute of Computer Science (SICS), E-mail: simonduq@sics.se.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

(see Fig. 1). As a result, we significantly improve energy efficiency, reduce end-to-end delay, and increase resilience to wireless link dynamics when compared to traditional unicast routing in WSNs.

1.1. Significance and Distinction

Low-power links in WSNs are highly dynamic [Srinivasan et al. 2008; Srinivasan et al. 2010]. Link estimation [Woo et al. 2003; Fonseca et al. 2007] allows for WSN routing protocols (such as RPL [Winter (Ed.) et al.] and CTP [Gnawali et al. 2009]) to restrict forwarding attempts to links of consistently high reliability, hereby ensuring stable topologies. Our main departure from this work is that the opportunistic nature of our approach explicitly utilizes all neighbors, i.e., both stable and unstable links, for packet forwarding. As a result, we show significant improvements in terms of energy efficiency, delay, and resilience to link dynamics.

Originally, opportunistic routing [Larsson 2001; Choudhury and Vaidya 2004; Biswas and Morris 2005; Chachulski et al. 2007] was developed to improve throughput in multi-hop, mesh networks. These designs exploit the fact that in wireless mesh-networks radios are always on and hence can overhear messages at practically no additional cost. In contrast, sensor networks are commonly duty-cycled to ensure long node and network lifetime, limiting the use of overhearing for opportunistic routing. Moreover, WSN applications demand high energy efficiency and low delays rather than high throughput. The main distinction of this work over existing ones on opportunistic routing is that it adapts the concept of opportunistic routing to WSNs, accounting for critical hardware limitations and the specific demands of sensor networks and their applications.

1.2. Contribution

This paper has four contributions: First, it presents Opportunistic Routing in Wireless sensor networks (ORW). ORW adapts the concept of opportunistic routing to the particular requirements and challenges in WSNs by focusing on energy as a key metric and tailoring the design to duty-cycled nodes. Second, it introduces a detailed analytical model to compute the expected number of duty cycled wakeups as an indicator of the average end-to-end delay in WSNs. Analytical insight from the model allows us to formulate a novel anycast routing metric that approximates the exact expressions for the expected number of duty-cycled wakeups. The metric has several nice properties that enables efficient construction of an anypath routing gradient and determination of forwarder sets for opportunistic routing. We present a light-weight distributed routing algorithm and show that it is loop free and converges to the optimality of the metric in a few number of steps. Third, we introduce a lightweight, coarse-grained link estimator that reduces the probe traffic and the state information. It reflects the reduced requirements of opportunistic routing in terms of timeliness and accuracy in link estimation. Fourth, we present a practical realization of opportunistic routing and evaluate its benefits in both simulation and TinyOS-based testbed experiments. We show that ORW reduces radio duty cycles on average by 50% (up to 90% on individual nodes) and delays by 30 to 90% when compared to the state of the art. Additionally, we show an increased stability to link dynamics and node failures. This work builds upon our two previous papers [Landsiedel et al. 2012; Ghadimi et al. 2012]. In the present work, we align the theoretical and practical design perspectives of previous papers and provide an unified overview of a opportunistic routing in WSNs with all its building blocks.

The remainder of this paper is structured as follows: Section 2 provides the required background on opportunistic routing and introduces the basic concept of ORW. Next, we tailor opportunistic routing to the specific demands of WSNs and detail mechanisms for forwarder selection, our anycast routing metric, and link estimation (Section 3). We compare our design to the state of art in Section 4 and discuss related work in Section 5. Section 6 concludes.

2. ORW DESIGN OVERVIEW

In this section, we provide the required background on opportunistic routing in mesh networks and discuss why it cannot be directly utilized in wireless sensor networks. Next, we introduce the basic

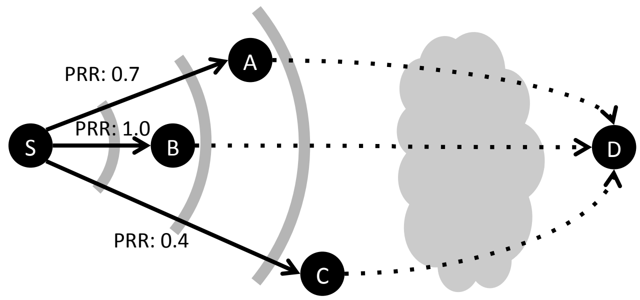
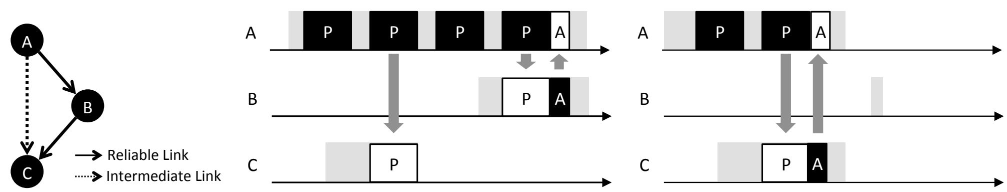


Fig. 1: Opportunistic routing in ORW: The first awoken neighbor (A to C) that successfully receives a packet from S and provides routing progress, forwards it to the destination D. It utilizes all neighbors that provide routing progress independent of link quality.



(a) Sample topology: Node A reaches C via B on reliable links or directly on an unreliable link.

(b) Traditional unicast routing in WSNs: Although C might overhear some transmission from A, packets are addressed to B to ensure stable routing.

(c) Opportunistic Routing in ORW: The first node that wakes up, receives a packet, and provides sufficient routing progress acknowledges and forwards it.

Fig. 2: Basic idea of ORW: Utilizing the first woken neighbor as forwarder, ORW reduces energy consumption and delay. This exploitation of spatial and temporal link diversity also increase resilience to link dynamics.

concepts of our opportunistic routing scheme, motivate them by simple examples, and outline how they are tailored to the particular demands of wireless sensor networks.

2.1. Preliminaries

Opportunistic routing [Biswas and Morris 2005; Chachulski et al. 2007; Larsson 2001; Choudhury and Vaidya 2004] improves network throughput in the context of multi-hop, mesh networks such as city-wide wireless networks. In contrast to traditional unicast routing, the underlying concept of opportunistic routing is to delay the forwarding decision until after the transmission so as to maximal use of the spatial diversity of the radio channel. For example, in ExOR [Biswas and Morris 2005] each packet is addressed to a set of potential forwarding nodes, prioritized by routing progress. Based on their priority, each node in the forwarder set is assigned a time slot for forwarding, which it only utilizes if it did not overhear the packet being forwarded in a previous time slot. Relying on such a arbitration protocol or other approaches [Chachulski et al. 2007], opportunistic routing avoids duplicate forwarding. By leveraging on spatial diversity, opportunistic routing ensures high routing progress and limits the impact of link dynamics. This leads to a significant throughput improvement when compared to traditional routing schemes [Biswas and Morris 2005; Chachulski et al. 2007].

2.2. Opportunistic Routing in WSNs

Wireless sensor networks and their applications pose special requirements, such as low power consumption and severe resource constraints, that distinguish them from traditional multi-hop mesh networks. These limit the direct applicability of existing opportunistic routing protocols in three key aspects:

Energy Efficiency vs. Throughput as Performance Metric: Opportunistic routing is designed to improve network throughput. However, WSN applications commonly demand reliable forwarding at high energy efficiency and not high throughput. In this paper, we show how opportunistic routing can be adapted to improve energy efficiency compared to traditional WSN routing.

Duty Cycled Radios: WSNs are commonly duty-cycled to ensure long node and network lifetime. Hence, nodes are in deep sleep states most of the time, with their radios turned off. Duty-cycling limits the number of nodes that concurrently overhear a packet (assuming no prior synchronization). As a result, it limits the spatial reuse in the forwarding process, one of the key benefits of opportunistic routing. However, we show in this paper that opportunistic routing brings low latency to duty-cycled networks: Instead of waiting for a given forwarder to wake up, the anycast primitive allows a node to send to the first awoken parent.

Low-Complexity Mechanisms for Unique Forwarder Selection: Commonly, opportunistic routing relies on a consensus protocol to determine a unique forwarder among the receiving nodes. For example, each packet in ExOR contains a list of potential forwarders and their priorities. Due to the small packet size in sensor networks such forwarder lists are not feasible. Similarly, assigning time slots to each potential forwarder poses implementation challenges. We introduce a lightweight algorithm for unique forwarder selection tailored to the resource constraints in WSNs.

In this paper we argue that the concept of opportunistic routing, i.e., delaying the decision of selecting a forwarder until the packet has been received, is well suited for the large node densities and high link dynamics in WSNs. However, many aspects of its realization need to be revisited and adapted to the specific requirements of WSNs.

2.3. Basic Idea of ORW

ORW targets duty-cycled protocol stacks. For simplicity we illustrate the basic concept of ORW in the context of an asynchronous low-power-listening MAC, such as in X-MAC [Buettner et al. 2006][1]. In low-power-listening a sender transmits a stream of packets until the intended receiver wakes up and acknowledges it (see Fig. 2b). To integrate opportunistic routing into duty cycled environments, in ORW we depart from the traditional unicast forwarding scheme in one key aspect: The first node that (a) wakes up, (b) receives the packet, and (c) provides routing progress, acknowledges and forwards the packet, see Fig. 2c. For example, in Fig. 2a node A can reach node C either directly via an unreliable link or via B. Commonly, traditional routing ignores the unreliable link $A \to C$ and relies on $A \to B \to C$ for forwarding. ORW extends this, by also including $A \to C$ into the routing process: If $A \to C$ is temporary available and C wakes up before B, ORW utilizes it for forwarding. This reduces the energy consumption and delay (see Fig. 2c).

Our design enables an efficient adaptation of opportunistic routing to the specific demands of wireless sensor networks: (1) In contrast to opportunistic routing in mesh networks, forwarder selection in ORW focuses on energy efficiency and delay instead of network throughput: It minimizes the number of probes until a packet is received by a potential forwarder. (2) It integrates well into duty-cycled environments and ensures that many potential forwarders can overhear a packet in a single wakeup period. Thereby, ORW exploits spatial and temporal link-diversity to improve resilience to wireless link dynamics. (3) The fact that only a small number of nodes receive a probe at a specific point in time simplifies the design of a coordination scheme to select a single forwarder. This limits overhead of control traffic.

[1]The concepts in ORW are generic and apply also to both phase-locking and receiver-initiated schemes.

3. DESIGN

After having reviewed the basic ideas and concepts of ORW, we now present its core mechanisms. Specifically, we introduce a variety of key algorithmic and system level elements that should work in harmony to realize anycast routing idea in duty cycled WSNs. We highlight the differences between ORW and traditional routing in WSNs and discuss key challenges such as stability and avoiding routing loops, duplicates, and asymmetric links. Before presenting the technical details of ORW let us briefly feature its main design elements:

Analytical Model for Expected Duty Cycled Wakeups: The existing routing solutions often rely on the average number of end-to-end (re)transmissions as an indicator for the cost of delivery [De Couto et al. 2003; Biswas and Morris 2005; Zhong and Nelakuditi 2007]. We find it inefficient for duty cycled WSNs, since these do not include –the essential factor– radio-on time into account. For this reason, the first element of the design is devoted to a framework that calculates the cost of forwarding in the presence of duty-cycling. After a brief description of the notion of Expected Number of Duty Cycled Wakeups (EDC), in section 3.2 we develop a unified theoretical framework to study the delay in terms of the average number of wakeups required for end-to-end packet delivery. Later in Section 4, this model is utilized as a baseline measure for comparisons between different routing metrics.

EDC Metric: While the analytical model provides a complete picture of the expected number of wakeups required for the end-to-end packet delivery in duty cycled WSNs, it provably includes complex computations. The EDC metric is the second building block of ORW that aims at simplifying the comprehensive analytical model yet being able to find near-optimal routing solutions.

Forwarder Sets: Forwarder set constructs the routing table of a sender in ORW. Each of nodes in the set are allowed to relay the traffic from the sender upon a possible packet reception. Including more neighbors in the forwarding set helps to reduce the required time until one of potential forwarders wakes up to receive. On the other hand, irrationally expanding the forwarding set increases the risk of choosing inefficient routs or even routing loops. Therefore, it is not inordinate to refer the forwarder set selection as the brain of ORW. We develop localized agile schemes to construct the forwarder sets as well as the formal proofs of optimality with respect to the EDC metric in Section 3.4. Later, Section 4.1.3 shows that our forwarder set selection algorithm is highly accurate compared to the optimal routing solutions constructed via exhaustive searches.

Link Estimator and Neighbor Discovery: As any other routing scheme, ORW requires mechanisms for neighbor discovery and link quality estimation. However, ORW utilizes a pool of forwarders, where each packet eventually takes various routs. Therefore, the influence of a temporary individual link failure on the overall forwarder set performance is quite limited compared to the traditional unicast routing schemes. For this reason, we have tailored a light-weight link estimator and neighbor discovery scheme that relies on the overhearing. We explain it in details in Section 3.5

Unique Forwarder Selection: ORW implements a coordination protocol to prohibit multiple nodes to relay simultaneously the same received packet. Specifically, via light-weight localized mechanisms, this protocols: (1) determines the number of receivers of a packet, and (2) guarantees a unique forwarder in case of multiple receivers. We present the details of this protocol in Section 3.6

In next, we present the detailed description of above design building blocks. The system integration details of ORW will is discussed in Section 3.7.

3.1. Expected Number of Duty Cycled Wakeups (EDC)

In ORW, a packet is forwarded by the first awoken neighbor that provides routing progress. As a result, the routing topology towards a destination is not a tree anymore as in traditional unicast-based routing protocols. Instead, it is a Directed Acyclic Graph (DAG) with a single destination (such DAGs are often called Destination Oriented DAGs, DODAGs). In this DODAG, ORW allows each packet to traverse on a different route to the destination (anycast). Note that DODAGs are sometimes used instead of trees even in unicast-based routing protocols, such as RPL [Winter (Ed.) et al.]. In that case, a single parent is selected *before* transmitting any packet.

ORW introduces EDC (Expected Duty Cycled wakeups) as a routing metric. EDC is an adaptation of ETX [De Couto et al. 2003] to energy-efficient, anycast routing in duty-cycled WSNs. In ORW, nodes use asynchronous duty cycling and Low Power Listening (as implemented by *e.g.* X-MAC [Buettner et al. 2006] or BoX-Mac [Moss and Levis 2008]), and a forwarding node retransmits a packet repeatedly until a receiver wakes up and acknowledges it. In this setting, EDC describes the expected duration, i.e., number of wakeups of duty-cycled nodes, until a packet has reached its intended destination, possibly across multiple hops.

The EDC routing metric is an indicator of both the radio-on time to reach a destination and the end-to-end packet delay. Thus, choosing routes with low EDC lead to reduced energy consumption and small delays, both key metrics in sensor networks. The multiple routing choices offered by anycast routing decreases the waiting time until one of the potential forwarders wakes up and successfully receives the packet, hereby lowering the end-to-end delay and overall energy consumption. In the following we introduce an analytical model for EDC and show that it depends on two key parameters: the number of potential forwarders and the quality of the wireless links to these. We then derive a simplified model for EDC, that reflects the resource constraints of wireless sensor nodes.

3.2. Analytical Model for Expected Number of Duty Cycled Wakeups

In this section, we introduce an analytical model for our anycast routing-metric EDC. We represent the topology of the network as a labeled directed graph $\mathcal{G} = \{\mathcal{N}, \mathcal{L}, \mathcal{P}\}$ with node set $\mathcal{N} = \{1, 2, \ldots, N\}$, link set $\mathcal{L} \subseteq \{(i, j) \ |i \in \mathcal{N}, j \in \mathcal{N} \backslash i\}$ and link parameters $\mathcal{P} = \{p(i, j) \mid (i, j) \in \mathcal{L}\}$. The link parameters model the success probabilities on links. Specifically, the analytical model assumes that packet loss process on each link $(i, j) \in \mathcal{L}$ follows a Bernoulli process with success probability $p(i, j)$, and is independent of the packet loss processes on the other links in the network.

Let $\mathcal{F}(i)$ be the set of forwarders of node i. For the moment, we assume that the forwarder set of each node is fixed. Later, in Section 3.4, we address how individual nodes can perform this forwarder selection. Further, we will assume that nodes use the same radio-on time length T. At wakeup interval w and before going to dormant state, node i draws the next wakeup time t_w uniformly in the interval $[0, T]$; *i.e.* node i wakes up at $t_0, T + t_1, 2T + t_2$, etc. In this setting, we are interested in analyzing the expected number of duty cycled wakeups for a transmission from source to destination. To this end, let DC be the random number of duty cycled wakeups required to complete the end-to-end transmission, and note that we can divide it into two independent components. First, the number of wakeups required for a single-hop transmission from the source node to one of its forwarders, and second, the number of wakeups the packet takes to reach from the forwarder node to the sink. Since link losses are assumed to be independent, we can write

$$\mathbf{E}\{\mathrm{DC}(i)\} = \mathbf{E}\{\mathrm{DC}_s(i)\} + \mathbf{E}\{\mathrm{DC}_m(i)\}, \tag{1}$$

where $\mathbf{E}\{\mathrm{DC}_s(i)\}$ is the expected number of duty cycled wakeups until the packet has been received by one of the forwarders, and $\mathbf{E}\{\mathrm{DC}_m(i)\}$ is the expected number of remaining duty cycled wakeups it takes to complete the multi-hop transmission. The number of wakeups required for a single-hop transmission can be seen as the sum of two independent random variables, $\mathrm{DC}_s(i) = X_s(i) + Y_s(i)$ where $X_s(i)$ is the number of failed intervals (in which all forwarders wake up and fail to receive the transmission from node i) and $Y_s(i)$ is the waiting time within a wake up period of a node containing the successful transmission. A failed interval requires that all transmissions between i and $j \in \mathcal{F}(i)$ fail, hence $X_s(i)$ follows a geometric distribution with pdf

$$\Pr\{\mathrm{X(i)} = \tau\} = \prod_{\mathrm{j} \in \mathcal{F}(\mathrm{i})} (1 - \mathrm{p(i,j)})^{\tau} \left(1 - \prod_{\mathrm{j} \in \mathcal{F}(\mathrm{i})} (1 - \mathrm{p(i,j)})\right),$$

and mean

$$\mathbf{E}\{X_s(i)\} = \sum_{\tau=0}^{\infty} \tau \Pr\{\mathrm{X(i)} = \tau\} = \frac{\prod_{\mathrm{j}\in\mathcal{F}(\mathrm{i})}(1-\mathrm{p(i,j)})}{1-\prod_{\mathrm{j}\in\mathcal{F}(\mathrm{i})}(1-\mathrm{p(i,j)})}. \quad (2)$$

To characterize $Y_s(i)$, note that for node $j \in \mathcal{F}(i)$ to be the forwarder, (1) it must experience a successful transmission and (2) transmissions to all other nodes that wake up before j must fail. Moreover, the underlying event is conditioned on that at least one node will successfully receive in the current duty cycle. Hence, the probability of reception for $j \in \mathcal{F}(i)$ in an arbitrary duty cycle conditioned on at least one reception is given by

$$p_s(j) = \frac{p(i,j)}{1-\prod_{f\in\mathcal{F}(i)}(1-p(i,f))}. \quad (3)$$

In general, this event can happen for $2^{|\mathcal{F}(i)|-1}-1$ cases which is based on exploring the probabilities of having all nodes included in the possible subsets of forwarders $\{\forall f \subset \mathcal{F}(i)\backslash j\}$ have failed before node j. Let $\mathcal{F}_{k\backslash j}$ denote the set of all the subsets of forwarders of node i with cardinality equal k not containing node j. The probability of having exactly k failed transmission excluding node j is given by

$$p_f(k\backslash j) = \sum_{l\in\mathcal{F}_{k\backslash j}} \prod_{m\in l}(1-p(i,m)). \quad (4)$$

Due to continuity of the random variable, the probability of having two nodes with the same activation time is zero. Thus, the mean waiting time is achieved by iterating among all the nodes $j \in \mathcal{F}(i)$ with i.i.d uniform wakeups and different link qualities:

$$\mathbf{E}\{Y_s(i)\} = \frac{1}{T}\int_0^{\infty} x \sum_{j\in\mathcal{F}(i)} \sum_{k=0}^{|\mathcal{F}(i)|-1} \frac{p_s(j)}{T} p_f(k\backslash j) \left(\frac{x}{T}\right)^k \left(\frac{T-x}{T}\right)^{|\mathcal{F}(i)|-k-1} dx.$$

Since $\mathbf{E}\{Y_s(i)\}$ does not have dimension -it is a portion of duty cycle- we normalize the above equation by dividing it by T. As an example consider node i with n forwarders with the same success probabilities $p(i,j) = 1$. It turns out that in this case, $\mathbf{E}\{\mathrm{DC}_s(i)\}$ has a closed form. Note that $\mathbf{E}\{X_s(i)\} = 0$ and $\mathbf{E}\{\mathrm{DC}_s(i)\} = \mathbf{E}\{Y_s(i)\} = \frac{1}{T}\int_0^T n\frac{x}{T}(\frac{T-x}{T})^{n-1}dx = \frac{1}{n+1}$. The mean single-hop waiting time decreases hyperbolically with increasing number of neighbors.

On the other hand, $\mathbf{E}\{\mathrm{DC}_m(i)\}$ is the expected number of duty cycled wakeups which it takes to send the packet from the forwarder set $\mathcal{F}(i)$ to the sink given that a successful transmission has already took place between i and $\mathcal{F}(i)$. Hence, $\mathbf{E}\{\mathrm{DC}_m(i)\}$ is given by

$$\mathbf{E}\{\mathrm{DC}_m(i)\} = \sum_{j\in\mathcal{F}(i)} \Pr\{\text{j is the forwarder}\}\mathbf{E}\{\mathrm{DC(j)}\}, \quad (5)$$

where

$$\Pr\{\text{j is the forwarder}\} = \sum_{k=0}^{|\mathcal{F}(i)|-1} \frac{1}{\binom{|\mathcal{F}(i)|-1}{k}} \sum_{l\in\mathcal{F}_{k\backslash j}} \prod_{m\in l} (1-p(i,m)) \frac{p_s(j)}{|\mathcal{F}(i)|}.$$

In words, this corresponds to the probability of node j being the first successful receiver given that at least one forwarder receives in the current duty cycle. Up to now, we have developed an analytical framework to measure the cost of packet delivery for each sender in terms of the average number of duty cycles. With the exception of a few cases -like when all link reliabilities are equal to 1- the analysis and even simulations tend to be intractable with respect to increased network density. Thus, the expected number of duty cycled wakeups $\mathbf{E}\{\mathrm{DC}(i)\}$ cannot be used for selecting the optimal forwarder set in practical opportunistic routing protocols.

Let us quickly comment on the assumptions that have been made to simplify the model. We have assumed that (1) packet loss probabilities are independent of each other, and (2) that the wake up times are continuous and hence, there is no multiple receivers. However, in reality even though nodes wake up in different times their ACKs can still collide. Later, in Section 3.6.1 we account for the possibility of having multiple receivers.

3.3. Simplifying Expected Duty-cycled Wakeups as Routing Metric

We have argued that an accurate evaluation of the expected number of duty cycled wakeups under opportunistic forwarding is quite complex, even when the forwarder sets are fixed. While the protocol does not need an analytical formula for the expected number of wakeups, it does need a procedure for selecting the optimal forwarder sets. To this end, we need to define a lightweight metric that captures the essential features of opportunistic forwarding, yet allows us to develop provably correct algorithms for distributed forwarder set selection.

3.3.1. Introducing EDC as Routing Metric. We have found that a metric which we call EDC strikes an appealing balance between effectiveness in forwarder set selection and simplicity of ORW analysis. The EDC of a node i can be computed recursively via

$$\mathrm{EDC}(i) = \frac{1}{\sum_{j \in \mathcal{F}(i)} p(i,j)} + \frac{\sum_{j \in \mathcal{F}(i)} p(i,j) \cdot \mathrm{EDC}(j)}{\sum_{j \in \mathcal{F}(i)} p(i,j)} + w \tag{6}$$

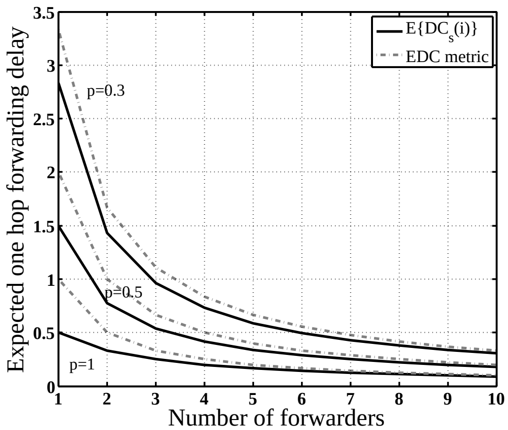
The first term is per-hop cost of forwarding because it approximates the expected one-hop forwarding delay $\mathbf{E}\{\mathrm{DC}_s(i)\}$ while the second term attempts to capture the essential features of the subsequent delay from forwarders to the sink. The third term, w, adds a weight to reflect the cost of forwarding. Please note, for single path forwarding and a weight w of 0, EDC equals to ETX and leads to the same topologies. As a result, we view EDC as an extension of ETX to anycast routing.

Per-hop cost of forwarding shares many important features of the analytical model: it is a hyperbolic function of the link reliabilities of the forwarders, and adding a forwarder or increasing the link reliabilities decrease the single-lop cost. Fig. 3 illustrates a situation with homogeneous links ($p(i,j) = p$ for all j) and compares $\mathbf{E}\{\mathrm{DC}_s(i)\}$ with the proposed hyperbolic approximation in EDC metric. We see that the analytical model and approximation tend to agree with each other when the number of forwarders increases.

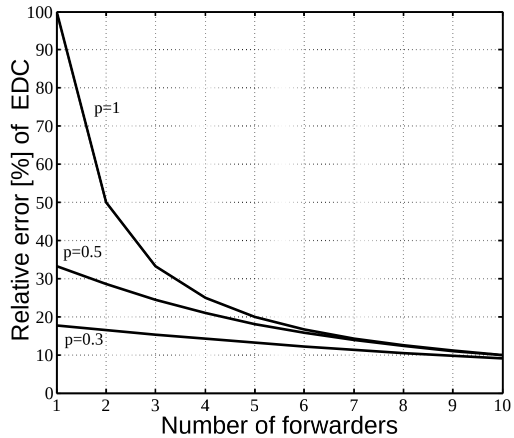
In the analytical model, the complexity of the multi-hop cost comes from the fact that the probability of j being a forwarder depends on the link reliabilities of the other nodes (the probability of a node j being a forwarder not only depends on that it could successfully decode the transmission when it is awake, but it also depends on the probability that nodes that woke up earlier all failed to decode the packet from i). The EDC metric simply assumes that the probability that j is a forwarder is directly proportional to $p(i,j)$. With this assumption, the probability of being forwarder for each node j is independent of others. However, since probability of forwarders must add up to one, we normalize each individual forwarding probability. Hence, in EDC metric, the probability that node j with reliability $p(i,j)$ being forwarder is $p(i,j)/(\sum_{j \in \mathcal{F}(i)} p(i,j))$. Our evaluation results in section 4.1.2 confirm the accuracy of EDC metric compared with the analytical scheme under real network scenarios.

3.3.2. Cost of Forwarding: w. We include the cost for forwarding, w, in EDC for a simple, practical reason: Processing and forwarding a packet consumes energy on a sensor nodes and adds delay. Thus, each additional hop increases both delay and energy consumption. As a routing metric, EDC models energy and delay and we argue that it is important to also include the cost of forwarding a packet into the routing metric.

We model w as a constant value and it describes the cost of forwarding a packet over one hop (see Eq. 6). It is important to note that the choice of w impacts the resulting routing topology: Increasing w increases the routing progress that a forwarding neighbor j of a node i is required to provide to be included in the forwarder set $\mathcal{F}(i)$. We discuss forwarder sets in detail in Section 3.4 where



(a) $\mathbf{E}\{\mathrm{DC}_s(i)\}$ versus EDC metric



(b) Relative error of EDC metric with respect to the analytical model $\mathbf{E}\{\mathrm{DC}_s(i)\}$

Fig. 3: Comparison of the analytical model versus EDC metric for $\mathcal{F}(i) = [1, 10]$ forwarders. The success probabilities p are equal for each forwarder. The plot shows that the relative error of EDC metric with respect to the analytical model $\mathbf{E}\{\mathrm{DC}_s(i)\}$ is bounded by 10% for the practical numbers of forwarders in WSNs.

we argue that increasing w leads to a smaller forwarder set. Based on different choices of w, one faces the following trade-offs: (1) A high value of w limits the forwarding to nodes that provide high routing progress and leads to fewer hops until a packet reaches the destination. (2) However, reducing the size of the forwarder set increases delay and energy consumption for packet delivery. (3) A too low choice of w increases the risk of temporary routing loops, as packets are forwarded by nodes that provide even minimal routing progress (see Section 4.2). Given these effects, a proper tuning of w allows ORW to balance delay and energy with routing progress and stability.

Please note, that w is independent of the actual routing metric that is in use. For example, it is applicable to traditional unicast routing based on ETX in a similar way. By default, ETX merely focuses on link reliability and not the forwarding cost. However, with adding w to ETX, one can extend it to include the cost of forwarding. As a result, paths with fewer hops would be preferred over paths with many hops in order to reduce the cost of processing etc.

In our evaluation we determine a range of values for w that ensure both stable and efficient routing (see Sec. 4.2) with EDC. From this we choose a default configuration that provides both high performance and high stability. We show that this default choice is independent from individual deployments and holds across all our evaluation scenarios.

3.4. Forwarder Sets

We define the forwarder set $\mathcal{F}(i) \subseteq \mathcal{N}(i)$ of a node i as the subset of its neighbor set that is taken into account to compute EDC (i). Two key factors impact the forwarder set: (1) adding more neighboring nodes to the forwarder set reduces the time until one of the potential forwarders wakes up to receive. Hence, it decreases the single-hop EDC of the node and improves the spatial diversity. However, (2) adding too many neighboring nodes to the forwarder set may decrease its average routing progress, as typically not all neighbors provide good progress (see Eq. 6).

In this section, we describe how to maintain the EDC metric in the network. Our key contribution is a distributed algorithm for forwarder selection that minimizes EDC. In particular, we show that after the appropriate ordering of potential forwarders, nodes can use a greedy algorithm to find the optimal forwarder set, and that this algorithm is loop free.

3.4.1. Distributed, Optimal Forwarder Selection. In the forwarder selection problem, each node i is given a set of potential forwarders $\mathcal{N}(i)$, their EDC metrics $\mathrm{EDC}(j)$ for $j \in \mathcal{N}(i)$, and the probability $p(i,j)$ of successful transmission from i to $j \in \mathcal{N}(i)$, and should determine the subset of forwarders $\mathcal{F}^\star(i) \subseteq \mathcal{N}(i)$ that minimizes $\mathrm{EDC}(i)$. To develop our algorithm, we first review some rudimentary properties of the EDC metric.

LEMMA 3.1. *Let $p(i,j) \geq 0$ and $c_j > 0$ for every $j \in \mathcal{N}(i)$. Define the set functions $f_i^{(1)}$: $2^{|\mathcal{N}(i)|} \mapsto \mathbf{R}$ and $f_i^{(2)} : 2^{|\mathcal{N}(i)|} \mapsto \mathbf{R}$ with $f_i^{(1)}(\emptyset) = \infty$ and $f_i^{(2)}(\emptyset) = \infty$ as*

$$f_i^{(1)}(\mathcal{A}) = \frac{1}{\sum_{j\in\mathcal{A}} p(i,j)}, \quad f_i^{(2)}(\mathcal{A}) = \frac{\sum_{j\in\mathcal{A}} c_j \cdot p(i,j)}{\sum_{j\in\mathcal{A}} p(i,j)}$$

for $\mathcal{A} \subseteq \mathcal{N}(i)$. Then $f_i^{(1)}(\mathcal{A})$ is strictly decreasing in $p(i,j)$ and $f_i^{(2)}(\mathcal{A})$ is strictly increasing in c_j.

PROOF. See appendix for this and all other proofs. □

In above lemma, c_j in $f_i^{(2)}(\mathcal{A})$, maps to $\mathrm{EDC}(j)$ in the definition of EDC metric, and it has been introduced for the ease of notation. Let $f_i(\mathcal{A}) = f_i^{(1)}(\mathcal{A}) + f_i^{(2)}(\mathcal{A}) + w$, then the following lemma presents a necessary and sufficient condition for when the insertion of a new member $k \in \mathcal{N}(i)\backslash\mathcal{A}$ in the forwarder set decreases $\mathrm{EDC}(i)$,

LEMMA 3.2. *Let $p(i,k) > 0$. If $c_k < f_i(\mathcal{A}) - w$ then $f_i(\mathcal{A} \cup \{k\}) < f_i(\mathcal{A})$ and vice versa.*

In next, we observe the behavior of $f_i(\mathcal{A})$ after adding a new member into the forwarding set $\mathcal{F}(i)$.

LEMMA 3.3. *If $c_k < f_i(\mathcal{A})$ and $p(i,k) > 0$ then $f_i(\mathcal{A} \cup \{k\}) > c_k$.*

Up to now, we have concluded that it is beneficial for node i to add a new neighbor k to the forwarder set if its EDC is less than the EDC of node i minus a positive offset w. Moreover, after adding node k, the updated EDC of node i is greater than $\mathrm{EDC}(k)$. The next Theorem characterizes the optimum forwarder set of node i.

THEOREM 3.4. *Let π be an ordering of the nodes in $\mathcal{N}(i)$ such that $\{c_{\pi(1)} \leq c_{\pi(2)} \leq \cdots \leq c_{\pi(|\mathcal{N}(i)|)}\}$. Then, the optimal forwarder set that minimizes the EDC value is $\mathcal{F}^\star(i) = \{\pi(1), \ldots, \pi(k)\}$ where k satisfies $c_k < f_i(\mathcal{F}^\star(i)) - w$ and $c_{k+1} > f_i(\mathcal{F}^\star(i)) - w$.*

Given a network topology $\mathcal{G} = \{\mathcal{N}, \mathcal{L}, \mathcal{P}\}$, Lemmas 3.2-3.3 and Theorem 3.4 suggest a greedy algorithm to compute the EDC metric in the network and to locally construct the set $\mathcal{F}^\star(i)$ at each node. Starting from the sink with $\mathrm{EDC}(\mathrm{sink}) = 0$, each node i in the network sorts its neighbors in the set $\mathcal{N}(i)$ in increasing order of their EDCs. A potential forwarder $j \in \mathcal{N}(i)$ is added to the set of forwarders $\mathcal{F}^\star(i)$ of node i if $\mathrm{EDC}(j) < \mathrm{EDC}(i) - w$, upon which $\mathrm{EDC}(i)$ is updated on the basis of the new set $\mathcal{F}^\star(i) \cup \{j\}$. The procedure repeats until the forwarding list and the EDC values of all the nodes remain unchanged. This procedure is described in Algorithm 1. The stopping criteria of the algorithm is distributed in the sense that each node stops updating its own EDC value (and its forwarding list) when all of it's neighbors EDC values remain unchanged.

One question here is what happens if two or more neighbors have equal EDC's but different link radiabilities. The next lemma shows that Algorithm 1 first picks the node with higher reliability.

LEMMA 3.5. *if $c_k = c_{k'} < f_i(\mathcal{A}) - w$ and $p(i,k) > p(i,k') > 0$ then*

$$f_i(\mathcal{A} \cup \{k\}) < f_i(\mathcal{A} \cup \{k'\}) < f_i(\mathcal{A}).$$

Under the hypotheses of Algorithm 1 and Lemma 3.5, the optimal forwarder selection procedure first picks the node k because it has a higher link reliability compared to k'. After adding k into the optimal forwarding set, i.e., $\{k\} \in \mathcal{F}^\star(i)$, by Lemma 3.3 we have $c'_k = c_k < f_i(\mathcal{F}^\star(i))$. Now, whether we should include k' in the optimal forwarder set or not depends on the value of w. This is due to the forwarder insertion criteria specified in Lemma 3.2 that requires $c_{k'}$ to be less than $f_i(\mathcal{F}^\star(i)) - w$. Note that for $w = 0$, such requirement is automatically satisfied by Lemma 3.3 and the optimal forwarder set either includes both k and k' or none.

ALGORITHM 1: EDC-based opportunistic routing.

Input: $\mathcal{G} = \{\mathcal{N}, \mathcal{L}, \mathcal{P}\}$, neighbor set $\mathcal{N}(i)\ \forall i \in \mathcal{N}$.
Initialize: $\mathrm{EDC(Sink)} \leftarrow 0$, $\mathrm{EDC(i)} \leftarrow \infty$.
repeat
  Set $\mathcal{F}^\star(i) = \emptyset\ \forall i \in \mathcal{N}$
  for all $i \in \mathcal{N}$ **do**
    Sort $\mathcal{N}(i) = \{n_1, \ldots, n_{|\mathcal{N}(i)|}\}$ s.t. $\mathrm{EDC}(n_j) < \mathrm{EDC}(n_{j+1})$.
    for $j = 1 \rightarrow |\mathcal{N}(i)|$ **do**
      if $\mathrm{EDC(j)} < \mathrm{EDC(i)} - \mathrm{w}$ **then**
        a. Update: $\mathcal{F}^\star(i) = \mathcal{F}^\star(i) \cup \{n_j\}$.
        b. Update: $\mathrm{EDC(i)}$ based on eq. (6).
      end if
    end for
  end for
until EDC of all nodes are unchanged.

Each iteration k of Algorithm 1 produces a new routing topology $\mathcal{R}^{(k)} = \{\mathcal{N}, \mathcal{L}^{(k)}, \mathcal{E}^{(k)}\}$ where $\mathcal{L}^{(k)}$ consists of links (i, j) from a node i to all its forwarders $j \in \mathcal{F}^\star(i)$, and $\mathcal{E}^{(k)}$ is a network-wide set of updated EDC. We next prove that each routing topology $\mathcal{R}^{(k)}$ is loop-free.

LEMMA 3.6. *Any routing topology $\mathcal{R}^{(k)} = \{\mathcal{N}, \mathcal{L}^{(k)}, \mathcal{E}^{(k)}\}$ produced by Algorithm 1 is a directed acyclic graph (DAG).*

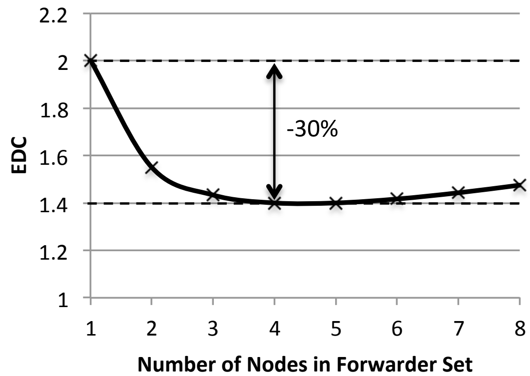
Our next result states that opportunistic routing algorithm converges in a finite number of steps.

THEOREM 3.7. *If $\mathcal{G} = \{\mathcal{N}, \mathcal{L}, \mathcal{P}\}$ remains constant through the execution of Algorithm 1, the algorithm terminates after at most $|\mathcal{N}|$ passes of the outer loop.*

Later in Section 4.2.5, we will show that the EDC construction algorithm converges fairly fast. In early iterations of the algorithm, the nodes closer to the sink will stabilize their EDC values. Afterward, nodes that are located far from the sink start to add closer ones in their forwarders and reach the optimality of the metric and this procedure propagates to the entire network.

3.4.2. Practical Considerations and Discussion. Practically, the forwarder set $\mathcal{F}(i)$ is computed by adding neighboring nodes sorted by their EDC – starting with the lowest EDC – to the forwarder set and determining the set with the minimum EDC (see example in Fig. 4). $\mathcal{F}(i)$ defines the $\mathrm{EDC}(i)$ of a node i and all neighboring nodes j that provide routing progress, i.e., $\mathrm{EDC}(j) < \mathrm{EDC}(i) - w$, are utilized as potential forwarders. As a node only selects nodes that provide strictly more progress than itself, the resulting topology forms a loop free graph, i.e., is a DAG (see Lemma 3.6 for a proof).

However, a slow spreading of updates, such as new EDC values due to link reliability changes of neighboring nodes, can lead to temporary loops until an update reaches all neighbors. This is common in routing protocols and not specific to ORW. We address it with three standard techniques: (1) when a parent node is downgraded to a child node, a node observes this and forwards its packets without dropping duplicates. Additionally, these packets are delayed shortly to allow the topology



Node	EDC	Use
1	1	✔
2	1.1	✔
3	1.2	✔
4	1.3	✔
5	1.4	✗
6	1.5	✗
7	1.6	✗
8	1.7	✗

Fig. 4: **Example: Optimal forwarder set.** Employing the practical algorithm discussed in Sec. 3.4.2 the forwarding node computes an EDC of 1.4 for itself and includes 4 neighbors with an EDC strictly lower than 1.4 in its forwarder set. Smaller or larger forwarder sets would lead to a worse EDC. When compared to single path routing, it reduces the number of expected wakeup periods to reach the sink by 30%. For simplicity, PRR is 1 and w is 0 in this example.

to stabilize. (2) A TTL field in each packet avoids infinite loops. (3) The forwarding cost w (see Sec. 3.3.2) ensures a threshold between forwarding nodes to avoid oscillating packets.

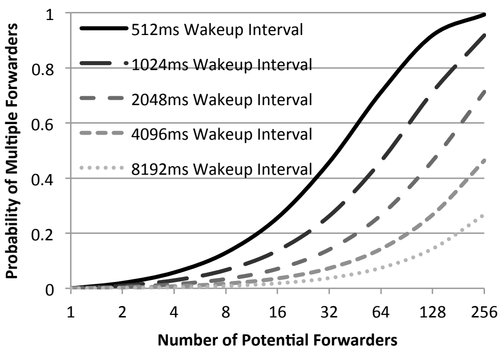
This design is in contrast to the traditional opportunistic routing such as ExOR or MORE [Chachulski et al. 2007] in the following key aspect: Commonly, opportunistic routing notes a prioritized forwarder set in the packet header while in ORW all nodes providing routing progress potentially forward the packet. This leads to two important benefits: (1) instead of long address lists in the packet header denoting a forwarder set, which is not feasible in resource constrained WSNs, a single value describes the EDC that a forwarder must provide at least. Upon receiving a message, each potential forwarder decides whether it provides sufficient routing progress. (2) It allows ORW to utilize spurious neighbors and neighbors it did not yet discover for forwarding. This reflects the link dynamics that are common in low-power, wireless networking.

3.5. Link Estimation and Discovery

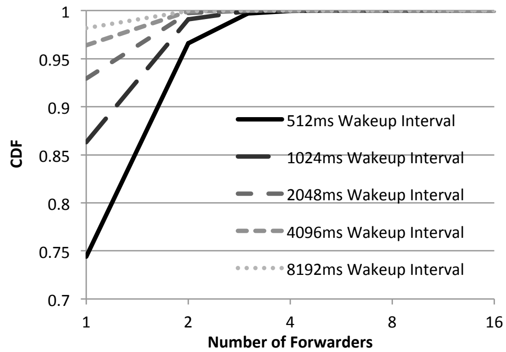
Anycast routing in ORW utilizes a pool of forwarders, where each packet potentially travels on a different route. Hence, when links to individual forwarders temporary fail or show reduced reliability, their impact on the overall quality of the forwarder set is limited. As a result, ORW does not require up-to-date estimates to each candidate forwarder, as traditional unicast routing.

Hence, we tailor link estimation and neighbor discovery in ORW to these specific demands. It mainly relies on overhearing: When a duty-cycled node in ORW wakes up to check for energy on the channel and subsequently receives a packet it (1) forwards it when providing routing progress, and (2) it updates its link quality estimate. For link estimation, a node maintains the link reception ratio from each neighbor. To this end, packets in ORW contain a header field that denotes the average rate at which a node is forwarding data. The link quality is obtained by dividing the rate of packets overheard from a neighbor by the forwarding rate of the same neighbor noted in the header field. As ORW operates with large forwarder sets and targets coarse grained link estimation, we argue that individual, asymmetric links have limited impact on the estimation and simplify link estimation by assuming $p(i, j) = p(j, i)$. This design defers from the traditional link estimation used for unicast routing in WSNs in the following two key points:

Stability. While agility is a key design criteria for modern link estimators as they shall adapt quickly to changes in link quality, ORW may not even recognize when a link temporary fails, assuming it utilizes multiple links for forwarding. This is a design goal: as long as the aggregate of the



(a) **Probability:** Probability that multiple forwarders receive the same packet for typical wakeup periods in ORW. For example, for 16 neighbors that provide routing progress, this probability is between 2% and 26% depending on the wakeup period.



(b) **CDF:** Cumulative distribution: TODO conclude in case of mutiple forwarders 2 or 3 forwarders is most common., plot in case of 16 potential forwarders.

Fig. 5: TODO

neighbors performs stably, the dynamics of individual links will be masked. Aging slowly removes broken links from the forwarder set and neighbor table.

Limited Use of Probing. Traditional link estimation employs probing to determine the link qualities to neighboring nodes. In contrast, ORW commonly relies on overhearing during wakeups to update its neighbor table and link estimates. Probing is only utilized when not a single route is available. In our evaluation, we show that the routing topology in ORW converges quickly after boot-up and node failures without the need for extensive probing. This reduces the overhead of control traffic in ORW.

Eventually, the bootstrap of a network using ORW is as follows: First, the nodes with no known parent probe their neighbors via a broadcast. The probing node i receives responses from a subset of its neighbors, depending on the link quality. For each received response from node j, node i adds the following entry to its neighbors table: (j, $\mathrm{EDC}(j)$, $p(i, j)$), where $p(i, j) = 1$. The link quality estimation, entirely based on overhearing, will refine the value of $p(i, j)$ and may add new entries to the neighbor table, when overhearing a neighbor that didn't answer to the original probe. Node i won't use probing anymore, unless all its entries reach an estimated link quality of 0.

3.6. Unique Forwarder Selection

Once a packet has been received by one or more nodes in the forwarding set, the next step is to ensure that only a single one forwards it. In this section, we first show that in the majority of the cases a packet is only received by a single forwarder. Next, we introduce a lightweight coordination protocol to determine a unique forwarder in case the packet was received by multiple nodes.

3.6.1. Probability of Multiple Receivers. In ORW, a packet is forwarded by multiple nodes, if (1) multiple nodes are awake while the packet is transmitted and (2) more than one of these awake nodes successfully receives it and provides routing progress. This probability of multiple forwarders depends on two factors: the node density and the wakeup rate of each node. Both a high node density and a high wakeup rate increase the probability of a packet being received by multiple forwarders.

To have a better understanding of the possibility of having multiple forwarders we consider the following scenario. Let n be the number of forwarders that wake up exactly once uniformly in a interval $[0, T]$. Assume that once node i wakes up it will remain active for a period a (typically a few milliseconds) listening for the incoming traffic and if there is no packet destined to i, it will switch to dormant state. For simplification assume that success probabilities are equal to 1. Having a forwarder with a packet in hand, we want to calculate the probability of having at least a second forwarder that receives the same packet. This event happens when the active period of at least one forwarder (out of remaining $n - 1$ forwarders) intersects with the current forwarder. The probability $q(n)$ that someone in a set of n forwarders receives the same packet as a particular forwarder (assuming fully reliable links) is then

$$q(n) = 1 - (1 - \frac{a}{T})^{n-1} \quad (7)$$

Fig. 5 depicts the probability of a packet being received by multiple forwarders for typical wakeup periods in ORW. For example, if 16 neighbors provide routing progress, the probability of multiple forwarders concurrently overhearing the same packet is between 2% and 28% for wakeup periods from 8192 to 512 ms, respectively (assuming fully reliable links). In this figure, the active period a is set to 10 ms for all cases. The probability of multiple forwarders decreases with increasing wakeup intervals, a key benefit, as ORW targets low-power networking utilizing large wakeup intervals. Energy on the channel such as other data transmissions or noise, may extend the duration that a node is listening and hence may increase the risk of multiple receivers.

It is worth pointing out that we leave the choice of the wakeup rate as a choice for the designer of the WSN since it has a direct relation to the application requirements. However, Fig. 5 indicates that at low wakeup rates as the main target of this paper, a packet is received by only a small number of nodes, and commonly only by a single forwarder. For moderate wakeup rates where one expects multiple simultaneous forwarders we design a lightweight coordination protocol for ORW to determine a unique forwarder out of multiple nodes, which we introduce next.

3.6.2. Coordination Algorithm. The coordination protocol in ORW fulfills two tasks: (1) it determines whether a packet was received by more than one potential forwarder, and (2) it ensures a unique forwarder in case of multiple receivers. Our goal is to achieve a duplicate rate similar to traditional unicast routing schemes. Due to the already low probability of multiple forwarders for a single packet at low wakeup intervals and traffic rates targeted by ORW (see Sec. 3.6.1), we aim for a practical, lightweight design. It relies on three mechanisms:

Demanding a Single Acknowledgment. Potentially, a sender receives multiple acknowledgments, one from each receiver[2]. This indicates multiple forwarders and hence the sending node resolves this as follows: It retransmits the packet and the forwarders will (potentially) receive the packet again. Receiving a link-layer duplicate, forwarders send a second acknowledgement only with 50% probability to reduce the number of duplicate acknowledgments. Eventually, only one acknowledgement is sent. Upon on receiving a single acknowledgment, the sender concludes that a single forwarder has been selected and does not not initiate further retransmissions. Once a forwarding node does not receive further retransmissions it concludes that it is the sole forwarder for a packet and hence proceeds to send the packet itself. The same mechanism is applied if acknowledgments collide, i.e., no acknowledgement is received (after a timeout) by the source. The calibration of acknowledgment probability in presence of link-layer duplicates to 50% is chosen based on the following observation: If if have multiple forwarders, two or three forwarders are the most common. Moreover, in practice link dynamics and asymmetry can lead to the sender only receiving one of the acknowledgments sent by the multiple forwarders. In ORW, we address this with the following two mechanisms.

Data Transmission Overhearing. When one nodes overhears another node forwarding the same packet while waiting for a clear channel, it cancels its own transmission. This mechanism is common in traditional opportunistic routing schemes. For example, ExOR [Biswas and Morris 2005] uses overhearing as key mechanism to ensure a single forwarder. In ORW, we add it as additional

[2]The delay between a frame and its acknowledgement is bounded by the time for the receiver to take the forwarding decision, and an additional small random time we inject, to guarantee non-constant timing and make acknowledgement collision more unlikely.

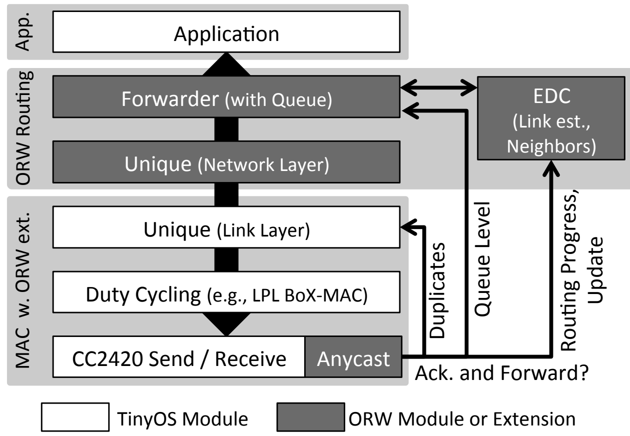


Fig. 6: Cross-layer control flow in ORW: Before acknowledging and forwarding a packet, ORW checks whether (1) the node provides the requested routing progress, (2) has space in the queue, and (3) the packet is not a duplicate.

mechanism to further detect multiple forwarders and reduce the number of duplicates. Note that adding this mechanism comes as practically zero cost, as nodes waiting for a clear channel are in receive mode anyway.

Network-Layer Duplicate Detection. Finally, network-layer duplicate detection serves as fall-back in case a packet slipped through the other mechanisms. Each node keeps track of the sequence numbers and source addresses of recently forwarded packets and filters any duplicates. Assuming a sufficiently large history, this mechanism detects all duplicates. However, as packets in ORW potentially travel along different routes, duplicate detection will be delayed until the forwarding path of two duplicates merges.

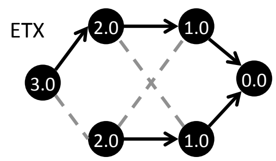
Please note that in the dynamic wireless environment with asymmetric or unstable links our practical design cannot guarantee a unique forwarder and packets may slip through. If they take different routes, this duplicate will only be detected at the sink. However, our evaluation shows that the lightweight mechanisms of ORW are sufficient to keep the duplicate rate at a level similar to traditional unicast routing such as CTP [Gnawali et al. 2009].

3.7. System Integration

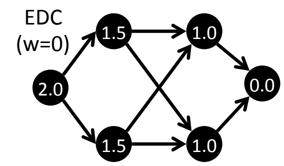
ORW acts as replacement of the unicast forwarding logic of WSN collection protocols. As a case study we integrated ORW into CTP, the de-facto standard for collection protocol in TinyOS. In this section we discuss system integration, and the portability of our design.

ORW provides the same interfaces as CTP to the application, and uses its protocol headers, and in part its TinyOS modules such as the forwarder. Anycast routing in ORW relies on two headers: The EDC of a node and the required routing progress is stored as two 8-bit values in the 802.15.4 MAC header instead of the 16-bit destination address, which is not required for anycast routing. Thus, we allow EDC values from 0.0 to 25.5 at a granularity of 0.1. Overall, the integration into the 802.15.4 header allows a node to decide whether it provides the required routing progress after reading merely the header. Hence, it reduces energy consumption and ensures that 802.15.4 acknowledgments are triggered timely.

Additionally, we extend the CTP routing header with one field: we add a weighted average of the transmission rate to facilitate link estimation (see Sec. 3.5). ORW places a small interface between routing and MAC layer (see Fig. 6): To decide whether to accept, i.e., acknowledge and forward, a packet, it determines whether (1) the node provides routing progress, (2) has space in its queue, and



(a) **Tree:** Building a routing tree with ETX as metric.



(b) **DODAG:** Building a DODAG with EDC as metric.

Fig. 7: **Topology I:** We depict the resulting tree and DODAG when using ETX and EDC, respectively, to build the routing topology. The toy topology highlights that a DODAG build with EDC leads to a dense routing topology when compared to the tree built with ETX. For simplicity, all links are assumed to be reliable, i.e., have a PRR of 1. The values depicted on the nodes represent the routing estimates computed by the respective metrics.

(3) the packet is not a duplicate (see Sec. 3.6.1). Hence, although ORW places functionality on both the routing layer and the MAC layer, its design is not bound to a specific routing protocol, MAC layer, or duty cycling scheme. ORW including link estimation requires slightly less RAM and ROM than CTP and its link estimator 4BitLE [Fonseca et al. 2007] which is mainly due to our simplified link estimator.

3.8. Examples: Building Topologies for Opportunistic Routing with EDC

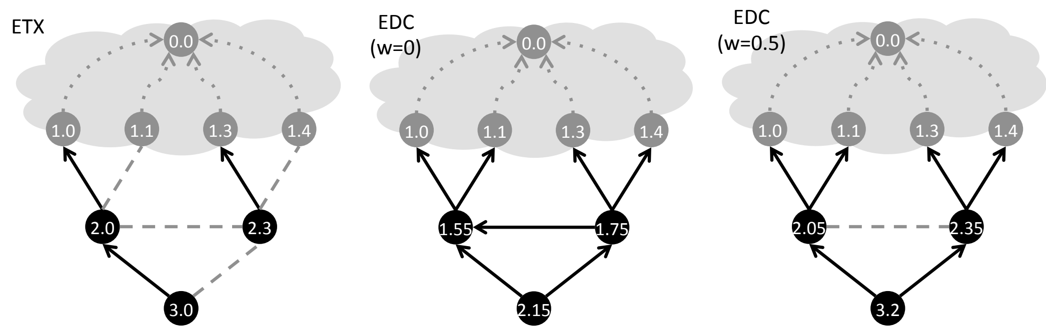
In this section we discuss how DODAGs built with EDC as routing metric differ from the trees formed with the ETX metric (see Section 3.3 and Equation 6). We show two sample topologies to highlight the following key differences between these two metrics: (1) By using EDC as routing metric, nodes tend to utilize more links when compared to the ETX metric. (2) EDC, unlike ETX, is able to establish a trade-off between the number of hop counts and the robustness against link dynamics.

In principle, the ETX metric describes the expected number of (re)transmissions until a packet reaches its destination. In contrast, EDC computes the estimated end-to-end delay (in duty cycles). Hence, while the resulting topologies from these metrics are comparable, for example, in terms of links utilized for routing, the values computed by the respective metrics are not directly comparable.

The routing topologies for ETX and EDC are depicted in Figures 7a and 7b, respectively. They form a simple toy-topology of 6 nodes. The figure highlights that a DODAG built with EDC utilizes more links than a tree constructed with ETX, leading to a dense routing topology. This translates into more stability against link dynamics and node failures. Additionally, it reduces the average end-to-end delay in duty-cycled networks by exploiting the spatial diversity of routes.

Figure 8 illustrates a subset of a larger network. It shows how EDC based topologies can tune the number of active links when compared to ETX based topologies. For example, if we set $w = 0$, all links in the sample topology are utilized (see Figure 8b) while a routing tree with ETX utilizes less than half of the links (see Figure 8a). As a consequence, the DODAG of EDC ensures a high stability to the dynamics of low-power wireless links and nodes. Additionally, the increased number of forwarding choices reduces the average end-to-end delay.

EDC as routing metrics leads to a large number of forwarding choices. Thus, it is not optimized in terms of hops counts and (re)transmissions when compared to the ETX metric. As a result, some packets in the DODAG may reach their destination via more hops than in the ETX tree. On the other hand, for values of w larger than 0 the degree of connectivity of the DODAG reduces (see Figure 8c) at the cost of an increased average end-to-end delay. Overall, theses examples show that the EDC metric is able to maintain a nice balance between the stability to wireless link dynamics and the number of hops required to reach a destination. Please note, for values of w larger than 0



(a) **Tree:** ETX as routing metric. (b) **DODAG:** EDC with $w = 0$ as routing metric. (c) **DODAG:** EDC with $w = 0.5$ as routing metric.

Fig. 8: **Topology II:** For a given set of parent nodes (depicted in gray) offering a different routing progress each to the destination, we depict the resulting routing topology for ETX, EDC with $w = 0$, and EDC with $w = 0.5$. The figures highlight the impact of the forwarding cost w on the number of active links. For the values of w larger than zero, EDC is able to tune the number of utilized links to avoid the packets from veering across the network. For simplicity, all links are assumed to be reliable, i.e., have a PRR of 1. The values depicted on the nodes represent the routing estimates computed by the respective metrics.

Evaluation Result	Type	Section
Analytical model validation	Simulation	4.1.2
EDC matches the analytical model	Simulation	4.1.3
EDC outperforms other metrics in a duty-cycled WSN	Simulation	4.1.4
Calibration of forwarding cost w	Testbed implementation	4.2.2
ORW improves the duty-cycles and delays versus CTP	Testbed implementation	4.2.3
ORW better resists to the node failure compared to CTP	Testbed implementation	4.2.4
ORW stabilizes quickly without expensive probing	Testbed implementation	4.2.5
ORW can operate at much lower wakeup intervals than CTP	Testbed implementation	4.2.6

Table I: The summary of our evaluations.

the metric depicted in the figures shows the sum of the estimated end-to-end delay and w and is not a direct representative of the delay. Therefore, it can not be directly compared with ETX values.

4. EVALUATION

In this section we evaluate ORW. First, we validating our analytical model via extensive monte-carlo simulations on an exemplary setup. Second, we focus on the EDC metric and evaluate its routing solution accuracy by comparing it to an optimal any-path routing topology constructed by exhaustive search based on the analytical model. Third, our simulation study aims at EDC metric and its differences to ETX, and two alternative any-path routing metric [Dubois-Ferrié 2006; Zhong and Nelakuditi 2007]. Finally, we move to real hardware implementations, specifically we use two large testbeds for a detailed, experimental comparison of ORW and CTP. We focus on four key metrics: radio duty-cycle, end-to-end delay, reliability and transmission counts. Convergence time of the routing algorithm, impact of node failures and the choice of wakeup intervals on the performance, and spatial diversity of ORW are investigated and we discuss the advantages and limits of our method compared to the state of the art. Table I presents a summary of our evaluation results as well as pointers to the appropriate sections.

4.1. Simulation: Anycast EDC

In our simulation-based evaluation, we explore the potential of anycast routing with EDC in terms of delay, and hops when compared to alternatives in unicast (ETX) and anycast routing [Dubois-Ferrié 2006; Zhong and Nelakuditi 2007]. Also, we explore the impact of network density on the performance of EDC-based routing and we evaluate the influence of the transmission cost w on EDC (see Sec. 3.3).

4.1.1. Simulation Setup. In our simulations, we solely focus on different routing mtrics and not individual protocol implementations. Thus, we compare two idealized protocols in this section: anycast routing with EDC and unicast routing with ETX.

We use two sets of simulation data: (1) PRR traces from testbeds, and (2) randomly generated topolgies. To ensure a fair and realistic evaluation our network profile is based on PRR traces from two testbeds: Twist [Handziski et al. 2006] and Motelab [Allen et al. 2005] with 96 and 123 nodes, respectively. The Tx power for both testbeds are set to 0dBM. Moreover, we picked the nodes with ids 229 and 1 to be the sink in Twist and Motelab testbeds, respectively. In addition, we use randomly generated topologies ranging from 100 to 1000 nodes to explore the impact of network density on the performance of EDC as routing metric. Nodes are placed in a fixed area and employ the Friis transmission model for radio propagation in a custom simulator; results are averaged over 100 random topologies per data point. For each topology we determine the neighbor sets of all nodes and link qualities, i.e., PRR, between them. On top of these, we deploy our protocol models, i.e., idealized protocols. Hence, in this simulation-based evaluation we deliberately exclude protocol mechanisms outside of the routing metric itself such as link estimation or neighbor discovery. Additionally, this allows us to avoid protocol artifacts such the slow spreading of route updates or packet collisions.

Overall, our goal is to evaluate the underlying performance of our routing metric EDC independent of a specific protocol implementation, before we compare EDC-based anycast routing in ORW to CTP in our testbed based evaluation (see Sec. 4.2).

4.1.2. Validating the Analytical Model. We first compare the analytical model versus extensive monte-carlo simulation and EDC metric in a small example. The goal of this section is to show the match between the analytical model presented in Section 3.2 and a fine-grained numerical simulation that calculates the average of the number of duty-cycled wakeups for end-to-end delivery. In the next section we will study the match between the EDC metric and the analytical model via thorough simulations based on the testbed traces. But for the moment, we consider a toy example with a sender node that has 1 to 10 forwarders with fixed average wakeups and link reliabilities. We start the experiment with a sender and the first forwarder and calculate (both analytically and numerically via simulations) the average number of wakeups for the sender to deliver a packet to a hypothetical sink given the fixed average wakeups and link reliability of the forwarder. Then, we add the second forwarder while keeping the first one and re-calculate the average wakeups for the sender. We repeat the same procedure until all 10 forwarders are added to the forwarding set.

The results of the experiment is depicted in Fig. 9. Each data point in the curves corresponds to the updated y-axis values after adding a new forwarder. The y-axis value of each squared point in the plot is the average of one hundred thousand numerical simulation instances. The first observation is the accurate match between theoretical model and the simulation of the $\mathbf{E}\{\text{DC}\}$. Leveraging on this match, we will continue our evaluations while keeping analytical model as a baseline. Second, we also have included EDC metric values for the comparison. Note that with growing number of forwarders, EDC values eventually tend to the theoretical values. We intentionally have sorted forwarders in increasing order of average wakeups. As a result, the plot represents the behavior of the forwarder selection algorithm. Starting from the forwarder with the minimum EDC, the sender inserts a new member into forwarding set if the EDC of the new forwarder is less than current value of y-axis, in which in this example happens until the third forwarder.

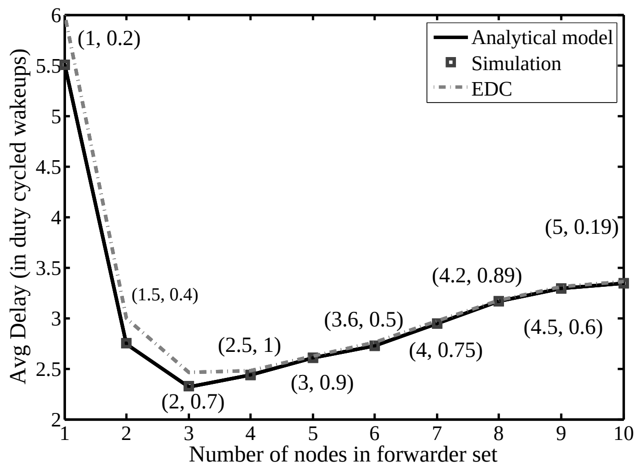


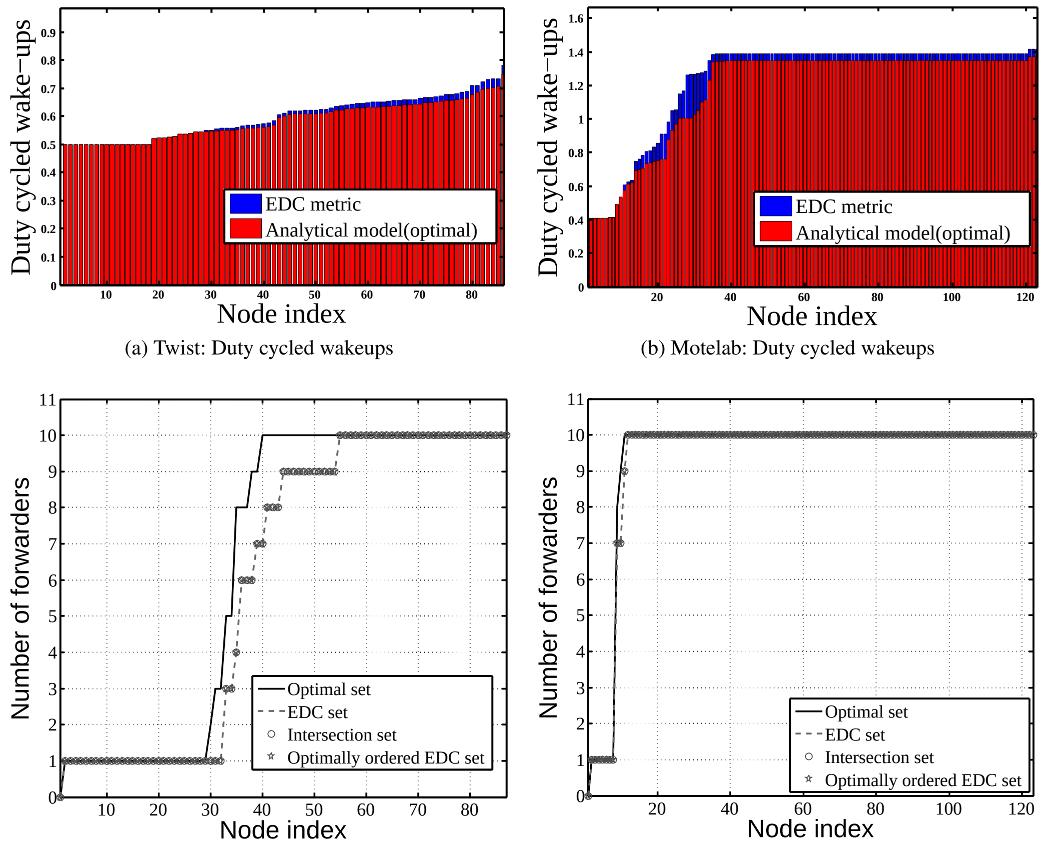
Fig. 9: Comparison of analytical model versus simulation and EDC metric for a node with $[1, 10]$ forwarders. Forwarder set grows incrementally with new members which are shown by tuple (EDC, reliability).

4.1.3. EDC Versus Analytical Model . To evaluate the EDC metric versus the analytical model, we perform several simulations over Twist and Motelab data traces. Specifically, we run the forwarder set construction mechanism (Algorithm 1) that utilizes EDC metric as its core and compare it against an exhaustive search based on analytical model (1). The number of forwarders for each node is limited up to 10 nodes. This restriction is made due to the exponential complexity of the analytical model as well as the exhaustive search over the forwarder candidates: with a state of the art PC, current setup requires 2 and 4 hours of simulations for Twist and Motelab profiles, respectively.

Fig. 10a and Fig. 10b show optimal duty cycles ($\mathbf{E}\{DC\}$) versus EDC metric for each node of the network. We note that EDC values are very close to the analytical values. Fig. 10c and Fig. 10d illustrate the number of forwarders versus node index. For each node, the number of optimal forwarders (given by exhaustive search) and the numbers from the EDC metric are depicted. Moreover, the number of common forwarders in these two schemes as well as the number of truly ordered common forwarders are plotted. We observe that the last three curves coincide to each other. In other words, EDC metric for each node, picks a subset of forwarders with the same order as the optimal set. Roughly speaking, for each node, the best forwarder in optimal set (the one with lowest $\mathbf{E}\{DC\}$) is also the first forwarder that EDC metric picks and so on.

The EDC values of nodes based on different restrictions on the number of forwarders is illustrated in Fig. 11. For some nodes (the ones closer to the sink) the EDC value does not change. The reason is that due to good paths towards the sink, they do not add more forwarders. In contrast, the nodes farther to the sink (the ones with higher EDC values) will benefit with having more forwarders. We observe that the values for these profiles does not change significantly after 20 forwarders.

4.1.4. EDC Versus Other Metrics. After evaluating the accuracy of EDC and the impact of routing table size, we next compare EDC with the other metrics. Different routing metrics are devised for various design contexts such as general purpose wireless networks, wired or battery-powered WSNs. A thorough performance comparison between the metrics requires a full system level implementation that is not the aim of this section. Here, we conduct the comparison in the context of the duty cycling; i.e. all the nodes are battery powered and only activate for a short interval followed by a random wakeup. So the underlying assumption is to utilize the ORW as a design core and then comparing the performance of different routing metrics in the duty-cycled environments. The comparison is carried out based on the following items. The Average number of wakeups required for



(a) Twist: Duty cycled wakeups

(b) Motelab: Duty cycled wakeups

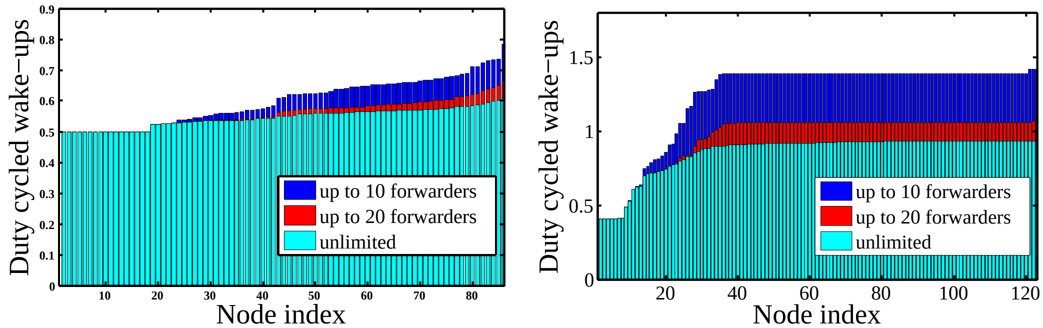
(c) Twist: Forwarder set

(d) Motelab: Forwarder set

Fig. 10: Per node comparison of EDC values and forwarder set in Motelab and Twist profile. The maximum number of forwarders is restricted to 10. The plots show high accuracy of EDC metric in both duty cycles and forwarder selection compared with optimal solution derived by exhaustive search.

the end-to-end packet delivery as an indicator of the delay. The average number of forwarders in the routing tables as an indicator of the overhead of route computation. Finally, the average number of hops performed by the packet to reach the (single) destination.

For the baseline of the comparison, we opted the expected number of duty-cycled wakeups, $\mathbf{E}\{\mathrm{DC}\}$, that has been developed in Section 3.1 and verified with monte-carlo based simulations in Section 4.1.2. The first counterpart is the widespread routing metric Expected Transmission Count (ETX) [De Couto et al. 2003]. Note that routing protocols using ETX aim to minimize the transmission count. In contrast to EDC, ETX does not take duty cycling into account. Hence, reducing transmission counts in ETX does not necessarily lead to low delays nor does it reduce radio on-time. The second and third counterparts of EDC are selected from the literature of anycast routing in wireless networks. To the best of our knowledge there is no metric that considers joint effects of duty cycling and link failure probability in the routing cost model. However, [Kim et al. 2008; Dubois-Ferrié 2006; Dubois-Ferrié et al. 2011; Zhong and Nelakuditi 2007] include either link reliability or duty cycling into their model. The popular EAX [Zhong and Nelakuditi 2007] accounts



(a) Twist: Duty cycled wakeups

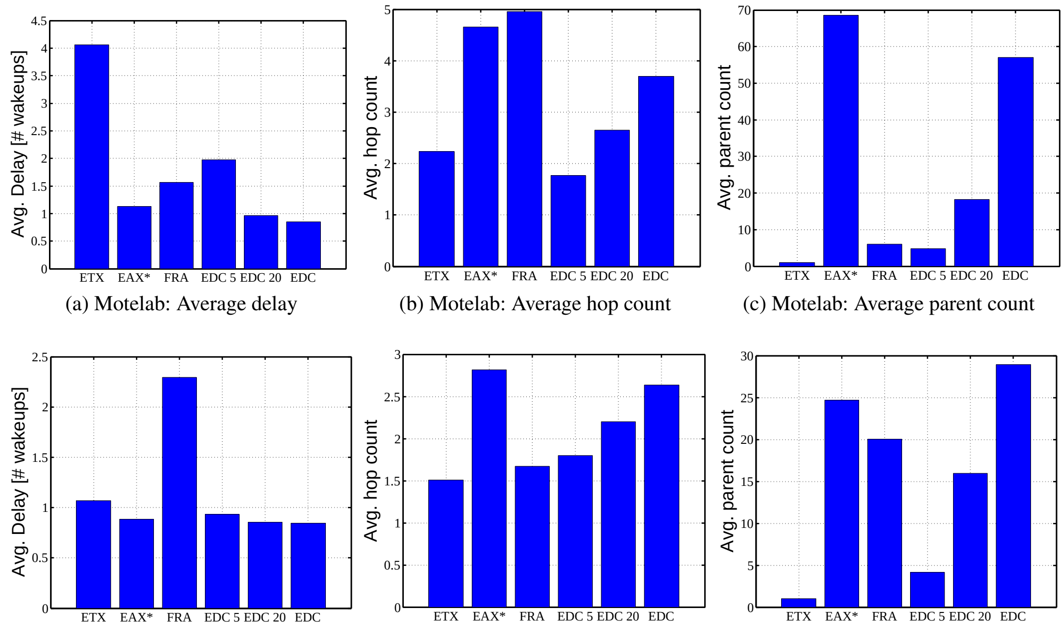
(b) Motelab: Duty cycled wakeups

Fig. 11: Per node comparison of EDC values in motelab and twist profile with different number of forwarders.

link-failure probabilities in its opportunistic routing metric while [Dubois-Ferrié 2006] provides an opportunistic metric for the duty-cycled WSNs with the full link-reliability assumption.

We note that aforementioned metrics are not directly comparable with EDC in their original design format. The EAX metric, assumes all the potential receivers being active while the sender transmits and it requires to maintain a priority list that dictates which parallel receiver has to relay the received packet. However, in a duty-cycled environment the multiple forwarder availability assumption of EAX does not hold. On the other hand, [Dubois-Ferrié 2006] employs preambles in a duty-cycled link layer while ORW replaces preambles with the actual packet (re)transmissions to gain a higher performance [Buettner et al. 2006]. Hence, the original equations for the cost calculation in [Dubois-Ferrié 2006] includes preamble details and are not directly applicable for ORW implementations. However, it is interesting to investigate if the performance benefits of the EAX metric prevails compared to ETX once it is applied in the duty-cycled ORW. Or whether it is important to consider both link-reliabilities and wakeups in the EDC metric. For this reason, based on above metrics, we have tailored two opportunistic alternatives for EDC. EAX* has the same metric values of EAX [Zhong and Nelakuditi 2007] but operates under the realm of ORW. The anycast routing with full reliability assumption (FRA) metric is the same as EDC with the assumption that all links are fully reliable.

Fig. 12 illustrates the comparison between EDC, ETX, EAX*, and FRA for two reference testbeds. Several comments are in order: First, the plots show that EDC reduces the delivery delay when compared to alternatives. In both testbeds, FRA and ETX have the worst average delay. EDC with 20 maximum allowed forwarders achieve a better delay compared to EAX* in both testbeds while the average number of forwarders for EAX* in Motelab and Twist are about 3.5 and 1.5 times more than EDC 20, respectively. Second, we show that depending on the size of the forwarding table, EDC achieves hop counts similar to ETX (see Fig. 12b and Fig. 12e). In some situations it even outperforms ETX. Third, compared to ETX, EDC-based routing explicitly utilizes all neighbors: instead of waiting for one specific neighbor to wake up as ETX based routing, EDC utilizes the first neighbor that wakes up and provides routing progress. As a result, EDC outperforms ETX in terms of delay, while leading to more hops.Forth, ignoring either link reliability or radio on-time in the metric design of the duty-cycled WSNs causes the performance degradation. On the other hand, relying merely on the hop count as an indicator of delay is not efficient since a sender may waste both time and energy by waiting for a best neighbor to wake up. However, with EDC we are able to limit the aggressive forwarding and force nodes to select a small number of good parents (see Fig. 12c and Fig. 12f). In this way we trade delay for hop counts and maintenance time of the forwarder set: by letting more candidates in the forwarder set we experience less delay while the



(a) Motelab: Average delay (b) Motelab: Average hop count (c) Motelab: Average parent count

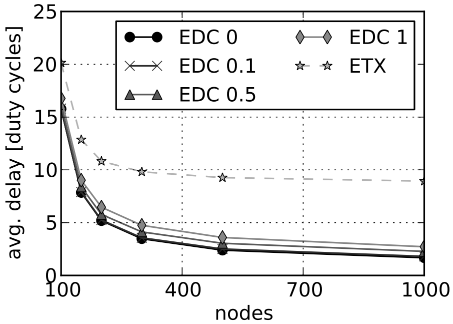
(d) Twist: Average delay (e) Twist: Average hop count (f) Twist: Average parent count

Fig. 12: Comparing EDC versus ETX, EAX*, and anycast routing with full reliability assumption (FRA) on Motelab and Twist traces. The EDC 5 and the EDC 20 data bars refer to the EDC metric when its number of forwarders are limed to 5 and 20, respectively. The EDC metric outperforms the three metrics in terms of delay. Additionally, EDC can be tuned so that it keeps the same average delay while reduces the hop count and the number of forwarders.

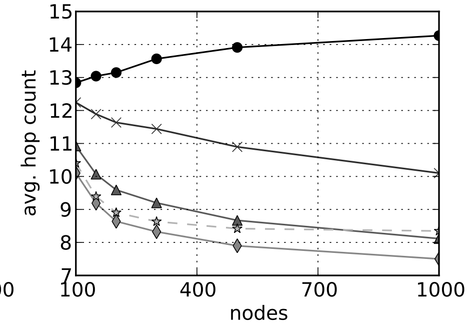
routing algorithm requires more steps to stabilize. As a final remark note that even though EDC requires a larger forwarder set than ETX and FRA (but often less than EAX*), the overhead of routing table management remains unchanged. In practice, other methods also need to keep the same number of neighbors as EDC to keep track the best routing option.

We also investigate the impact of the network density on the performance of the routing mechanisms. Fig. 13 shows that anycast routing with EDC benefits from an increased network density much more than unicast-based ETX. Compared to ETX, EDC reduces the average delay for packet delivery by factors of 1.3 and 6 for network sizes of 100 and 1000 nodes, respectively (see Fig. 13a). We later show that this leads to energy savings on a similar scale (see Sec. 4.2). As before, EDC leads to hop counts larger than ETX (see Fig. 13b).

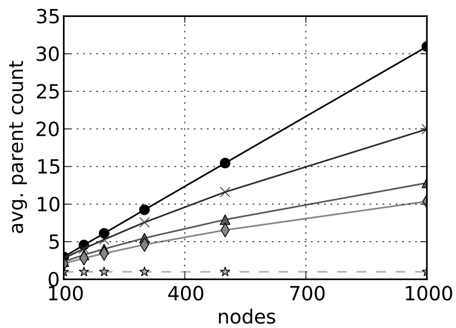
Finally, we evaluate how the per-hop forwarding cost w impacts the hop-count in the routing topology. Fig. 13c and Fig. 13d show that with $w = 0$, EDC utilizes about half of the neighboring nodes as parents. This decreases to 15% of the neighbors for $w = 1$ and it outperforms ETX in this case. EDC with $w = 0$ utilizes all forwarders that provide even the smallest routing progress. Hence, while minimizing delay, this increases the hop count when network density increases and as a result the number of parents increases, too. This aggressive forwarding, also makes this configuration of EDC sensitive to link dynamics and potentially leads to temporal routing loops, as we show in Section 4.2. In the experimental evaluation we next show that a configuration of w slightly above 0, such as 0.1, avoids these problems while utilizing many parents and providing low delay and high energy-efficiency.



(a) **Delay:** EDC outperforms ETX by a factor of 1.3 to 6, depending on network size and choice of w. For w of 0, EDC shows the lowest delay.



(b) **Hops:** For w of 1, EDC outperforms ETX in hops. For other configurations it trades lower delay for higher hop counts.



(c) **Parents:** Increasing network size and density allows EDC to utilize more parents for forwarding. Increasing w reduces their number.

Nodes	Density
100	6.0
150	9.2
200	12.3
300	18.6
500	31.0
1000	62.0

(d) **Density:** average node densities.

Fig. 13: Simulation-based evaluation: The results show that anycast routing with EDC benefits from increased density much more than unicast routing with ETX. Nodes are placed in an area of fixed size, leading to increased network density when the number of nodes increases.

4.2. Testbed Based Evaluation of ORW

In this section we evaluate ORW on real-world testbeds and compare its performance to CTP, the de-facto standard collection protocol in TinyOS. Please, note that the concepts in ORW are generic and independent of the chosen duty cycling scheme: They apply to both asynchronous and synchronous (phase-locked) MAC schemes as well as receiver-initiated ones (see Section 2.3). However, to ensure a fair comparison with CTP, we use BoX-MAC in this evaluation. It is the default MAC in both TinyOS and CTP; and resembles a combination of X-MAC and B-MAC [Polastre et al. 2004]. We also show results for CTP with A-MAC [Dutta et al. 2010], a state-of-the art, receiver-initiated MAC. We do not compare to SCP-MAC [Ye et al. 2006], a synchronous MAC. Although it promises high energy efficiency, it is not available for current TinyOS releases, and recent work [Vanhie-Van Gerwen et al. 2010] indicates that its energy efficiency in multi-hop collection trees is well below BoX-MAC.

4.2.1. Testbeds and Metrics. We base our experimental evaluation on two testbeds: Indriya [Doddavenkatappa et al. 2011] and Twist [Handziski et al. 2006], with 120, 96 and 123 nodes, respectively. For the each testbed we use two levels of transmission power, resulting in four evaluation scenarios (see Table II). We use the following setup for both ORW and CTP: Every node generates

Testbed	Size nodes, m^3	Sink id	Tx Power dBm	Diameter hops
Indriya	120,	1	0	5.6
	50 x 25 x 20	1	-10	9.1
Twist	96,	229	0	3.4
	30 x 13 x 17	229	-25	7.1

Table II: We use five evaluation scenarios: Each testbed contains about 100 nodes and the diameter ranges from 3 to 9 hops.

Testbed	Duty Cycle					Delay			Tx		
	Trace [%]		Improve [%]			Trace [s]		Impr. [%]	Trace [#]		Impr. [%]
	ORW	CTP	Avg.	Max.	Min.	ORW	CTP	Avg.	ORW	CTP	Avg.
Indriya, 0 dBm	1.1	2.2	50	79	-19	0.8	2.0	58	3.3	3.0	-11
Indriya, -10 dBm	1.6	2.8	41	88	-30	2.1	3.8	44	4.4	4.5	0
Twist, 0 dBm	0.8	2.0	57	82	0	0.1	1.2	91	1.8	2.0	10
Twist, -25 dBm	1.4	2.1	33	76	-39	1.8	2.4	29	4.1	3.8	-10

Table III: **Testbed Experiments Summary:** ORW decreases average duty cycles up to 57% and delays up to 90%, while achieving similar, but slightly higher transmission count than CTP.

a packet randomly with an average interval of 4 minutes, and the network forwards it to the sink. Unless explicitly mentioned, we use wakeup interval of 2 seconds (with our settings, this leads to the optimal duty cycle in CTP, see Sec. 4.2.6). As sink nodes, we use corner nodes 1 for Indriya and Motelab and 229 for Twist, respectively. We adopt the CTP default setting of having the sink node always on. For all experiments, we use channel 26 of 802.15.4 to avoid 802.11 impacting our results and potentially leading to an unfair comparison.

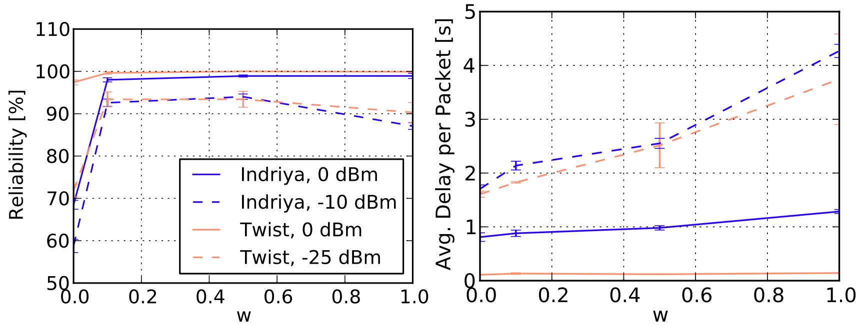
We evaluate energy consumption through the average duty cycle in the network, i.e., the portion of time spent with the radio chip turned on. The average duty cycle is a good proxy for energy consumption because (1) the radio chip consumes far more power than the other hardware components involved in our experiments and (2) low-power radio chips in sensor motes have a comparable power draw when transmitting or listening. This metric provides us with results that are independent from environmental conditions, hold across different hardware platforms, and are therefore reproducible. For a fair comparison we also skip the first two minutes when measuring duty cycles, as CTP shows a high duty cycle in this time due to its initial link probing.

For each data point, experiments are executed for a minimum of 30 minutes and are repeated three times; Experiments are executed at random times of the day, but back-to-back to ensure fairness. We display average results and error bars show standard deviations. Overall, the results shown are based on more than 300 individual experiments, each between 30 minutes and 2 hours.

4.2.2. System Calibration: w. We start the testbed-based evaluation by calibrating the forwarding cost w. Fig. 14 shows that in all four scenarios, a low w leads to the best performance in terms of delay and duty cycle. A larger w reduces hop counts at the price of increased delay and duty cycle. However, reliability shows a sharp drop for w of 0. Our logs indicate that a w of 0 increases the risk of routing loops and duplicate packets, which increase packet drops and reduce reliability. Please note, that ORW limits us to evaluate w at a granularity of 0.1 (see Sec. 3.7).

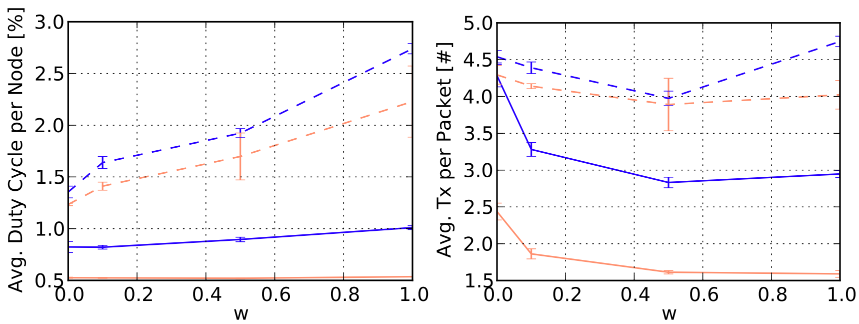
Overall, all of our four evaluation scenarios show that a value of 0.1 for w provides a stable balance between reliability, delay, and duty cycle. As these scenarios cover a wide range of network densities and network diameters (see Table II), we believe that a w of 0.1 is a good choice in general and use it as default in ORW.

4.2.3. Per Node Comparison of ORW to CTP. Next, we compare the performance of ORW and CTP in our four evaluation scenarios (see Table II). Fig. 15 and Table III show that ORW significantly improves duty cycles and delay. On average, ORW roughly doubles the energy efficiency,



(a) **Reliability:** A w of 0 is prone to routing loops, causing packet loss.

(b) **Delay:** Increasing the transmission penalty w increases the delay.



(c) **Duty Cycle:** Increasing w increases the duty cycle.

(d) **Tx:** The transmission count decreases when increasing w.

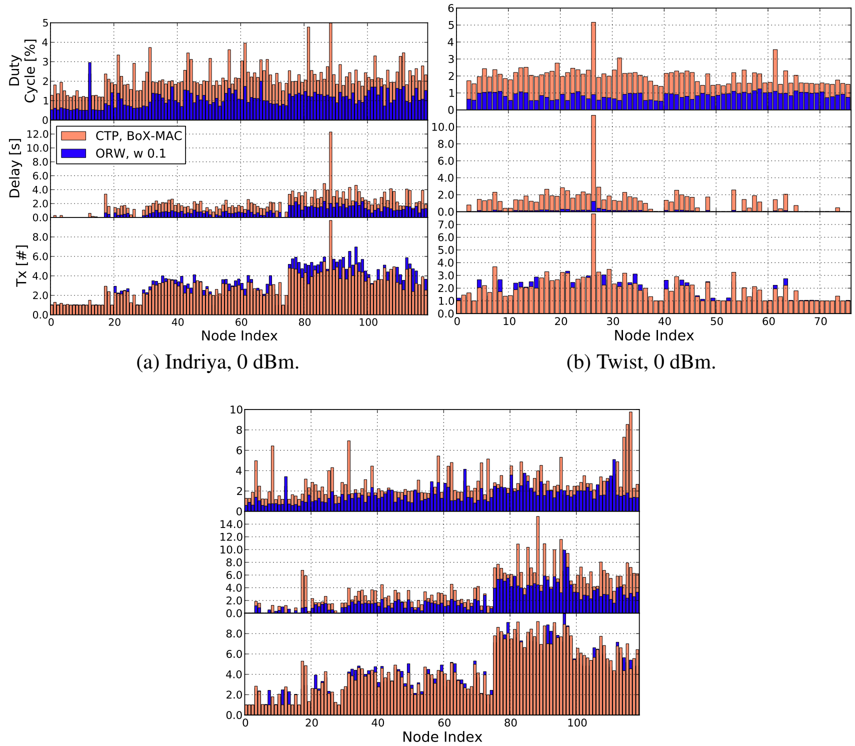
Fig. 14: **System Calibration**: a forwarding cost w of 0.1 is a good balance between energy efficiency, delay, and reliability.

individual nodes show improvements up to 88%. The results show that ORW strongly benefits from network density: it shows the best results on the dense Twist deployment at a transmission power 0 dBm. Additionally, it improves delay by 30% to 90% depending on network density and achieves (re)transmission counts (unicast or anycast) that are similar, but slightly higher when compared to CTP.

In ORW, nodes in dense networks and the ones further away from the sink benefit the most from spatial diversity in anycast forwarding (see Fig. 16). We define spatial diversity as the number of different, i.e., unique, forwarders that are utilized per hop on the path from a node to the sink during the course of the experiment. As another benefit of anycast forwarding, ORW removes outliers both in terms of duty cycles and delay (see Fig. 15). This has two key advantages: (1) It reduces the time until the first node runs out of energy and hence, ensures that sensor coverage can be maintained longer. (2) In terms of delay it allows us to switch to lower duty cycles in delay sensitive applications, what in turn reduces energy consumption even further.

4.2.4. Impact of Node Failures on ORW and CTP. We evaluate the impact of node failures on both ORW and CTP. Each 15 minutes, we remove on average 10 nodes from the network. Throughout the course of two hours this reduces the number of nodes from 120 to about 30 on the Indriya testbed.

Fig. 17 depicts the impact of node failures on the key metrics of reliability, transmissions, duty cycle, and delay. Both protocols show spikes of reduced reliability in presence of node failures. For increased node failure rates these spikes grow strongly for CTP. Hence, ORW maintains connectivity



(a) Indriya, 0 dBm. (b) Twist, 0 dBm.

(c) Indriya, -10 dBm.

Fig. 15: **Per Node Comparison of ORW and CTP:** ORW improves duty cycles and delays while achieving slightly higher hop counts than CTP.

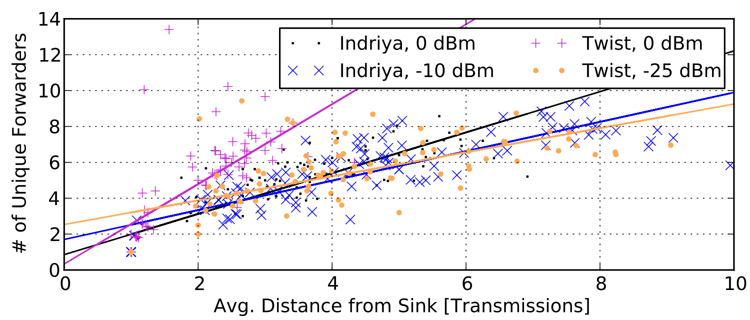


Fig. 16: **Spatial Diversity (Number of Unique Forwarders):** nodes in dense networks and the ones further away from the sink exploit spatial diversity, i.e., using different forwarders, in anycast routing the most. We plot data points and their linear regression.

much longer in the resulting sparse network. Additionally, it shows the benefits of anycast routing in ORW over unicast routing in CTP: node failures have only minimal impact on the duty cycle, delay, or transmission of ORW, while these show sharp peaks in CTP.

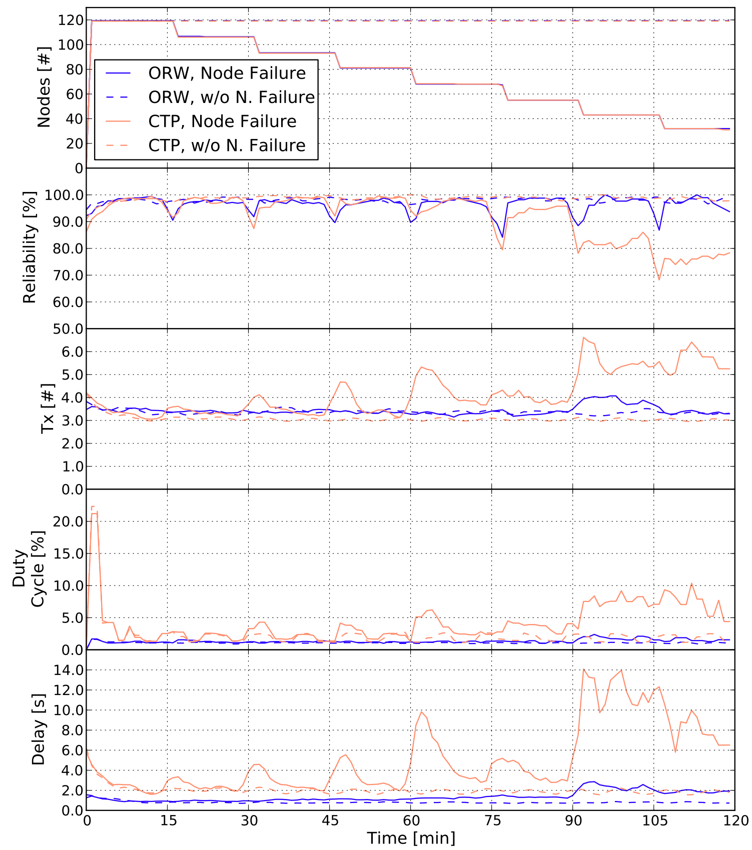


Fig. 17: **Node Failures (Indriya, 0 dBm):** While both ORW and CTP achieve similar reliability in presence of node failures, CTP pays a higher price in terms of energy, delay, and transmissions.

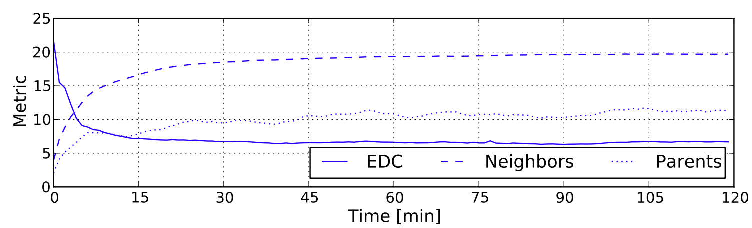
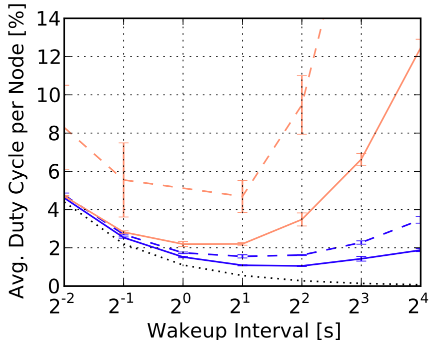
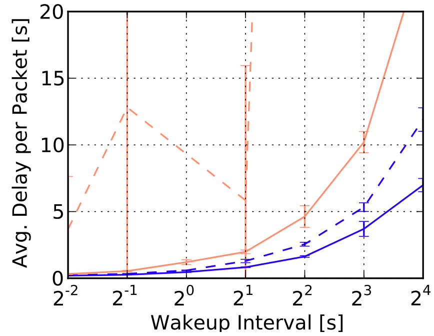


Fig. 18: **Convergence of ORW (Indriya, 0 dBm):** Average per node values for EDC, neighbors in the routing table and parents selected in the forwarder set.

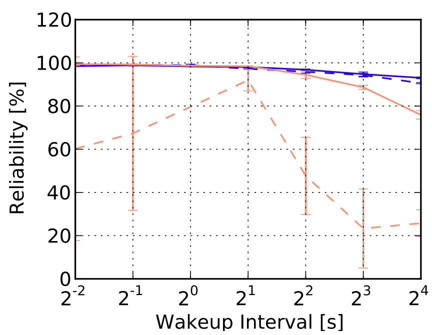
4.2.5. Convergence of ORW. Earlier in Section 3.4.1 we demonstrated that ORW in lack of link reliability changes converges in finite time. As ORW essentially operates without probing for link estimates (see Sec. 3.5), we evaluate its convergence in this section. We track the evolution of the average EDC as well as number of neighbors and parents per node. Fig. 18 shows that the routing metric EDC reaches an initial stable point within the first five minutes without the need for extensive beaconing or detailed link estimation. Over time it optimizes slightly.



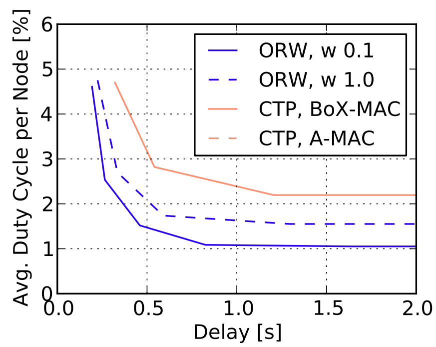
(a) **Duty Cycle**, Indriya



(b) **Delay**, Indriya



(c) **Reliability**, Indriya



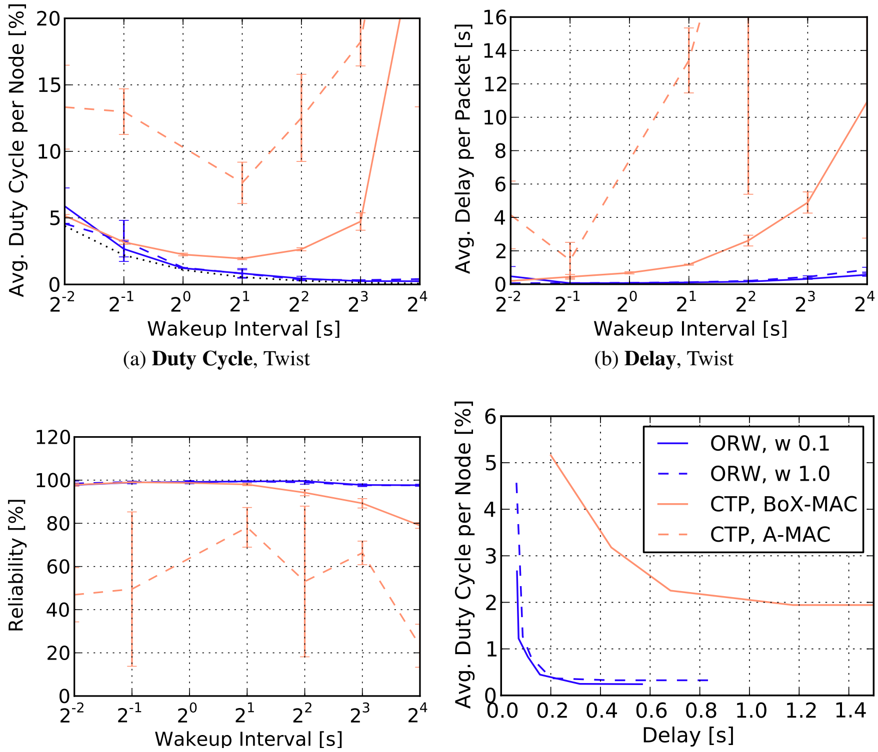
(d) **Duty Cycle over Delay**, Indriya

Fig. 19: **Choice of Wakeup Intervals (Indriya Testbed):** Performance of ORW and CTP on Indriya (with TX power of $0dBm$) for wakeup intervals between 0.25 and 16 seconds: ORW can operate at much lower wakeup intervals than CTP while achieving low delays and high reliability. Some data points for A-MAC are omitted due to inconsistent results. The dotted line in Figure (a) depicts the idle line of BoX-MAC.

Similarly, nodes in ORW continue to add new neighbors for routing. However, these lead to only minimal improvements, as we see only minimal changes to the duty cycle and delay over time (see Fig. 17). Overall, the results show that ORW stabilizes quickly without the need for expensive probing for link estimation and neighbor discovery. Additionally, Fig. 17 shows that ORW maintains this stability even in presence of node failures.

4.2.6. Choice of Wakeup Interval. The previous experiments used a wakeup interval of 2 seconds, i.e., a node wakes up every two seconds to receive data from neighboring nodes. We used this interval to ensure a fair comparison, as it leads to the optimal duty cycle in CTP and its default BoX-MAC at an inter-packet interval of 4 min (see Fig. 19a).

In this section, we discuss the impact of the wakeup interval on duty cycle, delay, and reliability. Fig. 19a and Fig. 20a show that ORW benefits much more than CTP from reduced wakeup intervals. The figures also depict the idle duty cycle, i.e., the energy that is consumed by just the wakeups of BoX-MAC without any data transfer. This baseline defines the lower bound for the duty cycle. For Indriya (see Fig. 19a) ORW stays closer to this line than CTP and in the dense Twist testbed (see Fig. 20a) ORW is marginally above the baseline throughout all experiments. These results show that ORW efficiently exploits network density. Delay increases with increased wakeup intervals (see Fig. 19b and 20b). However, the increase for ORW is significantly lower than for CTP with



(a) **Duty Cycle**, Twist

(b) **Delay**, Twist

(c) **Reliability**, Twist

(d) **Duty Cycle over Delay**, Twist

Fig. 20: **Choice of Wakeup Intervals (Twist Testbed):** Performance of ORW and CTP on Twist (with TX power of $0dBm$) for wakeup intervals between 0.25 and 16 seconds: ORW can operate at much lower wakeup intervals than CTP while achieving low delays and high reliability. Some data points for A-MAC are omitted due to inconsistent results. The dotted line in Figure (a) depicts the idle line of BoX-MAC.

BoX-MAC. Fig. 19d and Fig. 20d show the resulting duty cycle for a given average delay. This underlines that ORW can operate at much lower duty cycles for a given average delay.

Both CTP and ORW show high reliability (see Fig. 19c and 20c). However, at high wakeup intervals reliability of CTP decreases due to queue overflows on individual nodes. In contrast, ORW avoids this by using multiple forwarders. The results for other inter-packet intervals, radio channels, and testbeds show similar performance gains of ORW over CTP. As reference, the figures also depict results for CTP on A-MAC: However, it does not reach the performance of BoX-MAC for multihop routing with CTP. While we cannot conclude on the exact reasons, our traces indicate that its probes and loss of synchronization at low wakeup-rates increase duty cycles and delays.

4.3. Discussion and Limitations

After evaluating our anycast routing scheme in simulation and comparing it to the state of the art CTP collection protocol in testbeds, we reflect on the results and discuss limitations in this section.

4.3.1. Discussion. ORW improves duty cycles and delays significantly while achieving similar reliability and transmission counts when compared to the state of the art. Our results show an average decrease in duty cycle by about 50%, individual nodes improve up to 90%. Similarly, it decreases delay by a 30% to 90% depending on network density.

Anycast forwarding allows ORW to forward a packet faster than traditional unicast routing. Overall, such a design works best at high network densities, as this gives the most choices for forwarding. As a result, ORW shows the best results for dense topologies, i.e., in both testbeds at high transmission power. Similarly, its optimal duty cycle is at lower wakeup rates when compared to CTP. Hence, our results show that ORW can operate at much lower wakeup rates without major impact on reliability or delay (see Fig. **??**). At lower densities, ORW still outperforms CTP, but its benefits decrease (see Table III). Similarly, at high wakeup rates, the delay and energy advantages of ORW decrease. We believe that ORW can strongly benefit from the integration of adaptive duty-cycling [Jurdak et al. 2007; Meier et al. 2010; Hansen et al. 2012; Puccinelli et al. 2012]: Exploiting its anycast forwarding, it can efficiently adapt wakeup rates to traffic load and network density.

Additionally, its opportunistic nature allows ORW to take the state of wireless link into account and delay the decision of selecting a forwarder until the packet has been received. As a result, it reflects temporal and spatial diversity of wireless links (see Fig. 16) and increases the resilience of routing to link dynamics and node failures (see Fig. 17).

4.3.2. Limitations. ORW targets applications with lifetime demands in the order of month or years, which are typical deployment scenarios in WSNs. Commonly, such applications rely on duty-cycled low-power networking with wakeup rates in the order of seconds. Our evaluation shows that ORW achieves the strongest improvement at such low wakeup rates when compared to CTP (see Fig. **??**). At higher wakeup rates the baseline cost of the MAC layer accounts for the majority of the cost, and both CTP and ORW show similar performance in terms of energy and delay (see idle line in Fig. 19a and 20a). Additionally, ORW focuses on collection applications with low data rates. Thus, we believe that its design is not well suited for high throughput settings such as bulk transfers.

While ORW is agnostic to the underlying MAC scheme, it shows the strongest improvements for asynchronous MAC layers. For phase-locking MAC layers, we expect ORW to show similar improvements for delay but limited benefits in terms of energy. However, in dense deployments such as Twist (see Fig. 20a) the duty cycle in ORW closely approaches the idle base line of the MAC layer. This is also the cost of an ideal synchronous MAC, without considering its overhead in terms of time synchronization and guard times.

To ensure a fair comparison with CTP, our current implementation of ORW is tailored to collection applications with a single sink. Thus, we currently do not support mesh routing (or multiple sinks). However, the design of ORW is generic: When applications require it, this can be directly integrated by adding an extra header field that notes the intended destination next to the already existing requested routing progress (see Sec. 3.7).

5. RELATED WORK

In this section we discuss related work on opportunistic and adaptive routing in WSNs. Opportunistic routing itself is discussed in the preliminaries in Sec. 2.1.

GeRaF [Zorzi and Rao 2003b; Zorzi and Rao 2003a] pioneered the concept of anycast routing in duty-cycled wireless sensor networks. It utilizes geographic routing to determine routing progress of its neighboring nodes and a busy tone protocol to ensure a unique forwarder. CMAC [Liu et al. 2009; Liu et al. 2007] combines the concepts of GeRaF and ExOR: It includes prioritized forwarders, slotted acknowledgments and overhearing of acknowledgments to determine a unique forwarder as in ExOR. Relying solely on geographic routing, both do not address the key challenges for opportunistic routing in duty-cycled WSNs such as anycast routing metrics and wireless link dynamics.

The application of opportunistic routing in WSNs also received great attention from a more theoretical perspective. Similar to our work, [Kim et al. 2010; Dubois-Ferrié et al. 2011; Mao et al. 2011; Schaefer et al. 2009; Ashref et al. 2010; Basu and Chau 2008; Lu and Wu 2009; Xue et al. 2010; Zhong and Nelakuditi 2007; Kim and Liu 2008] consider anycast routing in WSNs. Their models and simulation results show that opportunistic routing can improve energy efficiency and delay when compared to traditional unicast routing. Their results strongly motivated our work. However, they omit the real-world challenges in terms of duty cycling and link dynamics that this paper addresses.

For example, LCAR [Dubois-Ferrié et al. 2011] assigns a relay candidate-set to each node in order to minimize the expected cost of forwarding a packet to the destination. The expected cost is recursively constructed by assuming that the relay nodes already know their own forwarding cost to the destination. Mao *et al*. [Mao et al. 2011] study the selection and prioritizing the forwarding list to minimize the overall energy consumption of WSNs. Similar to LCAR, it calculates an expected cost for every node to send a packet to a target. An optimal forwarder set in which minimizes this cost is constructed with a greedy algorithm. In contrast to our work, these do not consider the duty cycling in their theoretical model.

Joint study of anycast forwarding and duty cycling has been done in [Kim et al. 2010; Dubois-Ferrié 2006]. Kim *et al*. [Kim et al. 2010] investigates the optimal anycast forwarding policy for a poisson wakeup model to minimize the expected end-to-end delay in the event-driven WSNs. On the other hand, [Dubois-Ferrié 2006] considers a preamble based duty-cycled WSN and provides routing metrics for synchronous and asynchronous wakeups. However, in both solutions the forwarder selection merely depends on the wakeup process and does not take the probability of link failure into account: We consider this a key requirement, as due to the low-power nature of the WSNs, their links are highly dynamic [Srinivasan et al. 2008]. Other approaches to opportunistic forwarding [Liu et al. 2010; Autenrieth and Frey 2011; Pavković et al. 2011; Unterschütz et al. 2012] are routing agnostic and do not include energy efficient routing metrics nor tailor link estimation to anycast routing.

Commonly, routing protocols employ link estimation to identify long-term stable links [Fonseca et al. 2007; Woo et al. 2003] and to utilize these for routing. Recent approaches suggest to employ highly-adaptive link estimation to identify dynamic links [Srinivasan et al. 2008; Srinivasan et al. 2010] and to predict whether an intermediate link is temporary reliable [Becher et al. 2008; Liu and Cerpa 2011; Liu and Cerpa 2012]. For example, BRE [Alizai et al. 2009] employs short-term link-estimation [Becher et al. 2008] to reduce hop counts: when a far ranging link of intermediate quality becomes temporary available, BRE uses it to short-cut in the routing tree. In duty-cycled environments, BRE and related approaches such as 4C [Liu and Cerpa 2011] and Talent [Liu and Cerpa 2012] show two key limitations: (1) Routing short-cuts are only stable for a couple of milliseconds, making it difficult to exploit them in low traffic scenarios. (2) Their link estimators need to overhear data traffic to determine possible short-cuts. In heavily duty-cycled systems were nodes are asleep most of the time, this is not practical. Similarly, to these approaches, ORW exploits intermediate links for packet forwarding.: Utilizing the first awoken neighbor that provides sufficient routing progress, ORW employs both stable and dynamic links for packet forwarding. However, due its anycast nature, it does not demand for bursty traffic to make short-cuts in the routing tree. Thus, its benefits are also present in both heavily duty-cycled networks as well as systems with non-bursty and low-traffic rates.

Adaptive and low-power routing in WSNs proposes dynamic change of parents in routing. DSF [Gu and He 2007] selects the next hop of a packet based on the sleep schedule of neighboring nodes and different metrics such delay, reliability, and energy consumption. Similar to ORW, DSF shows strong improvements over unicast routing in these metrics. However, it focuses on synchronized networks. Furthermore, it requires iterative message exchanges to stabilize the forwarding schedules of all nodes, leading to control traffic overhead in the presence of dynamic links. The Backpressure Routing Protocol, BRP [Moeller et al. 2010], forwards packets to the neighbor with the lowest queue level. This improves throughput when compared to traditional unicast routing, while increasing delay. However, BRP can only be applied when the overall system is saturated, i.e., nodes always have packets to forward. This is rare in WSN deployment scenarios as these commonly show low traffic rates.

6. CONCLUSIONS

This paper introduced Opportunistic Routing for Wireless sensor networks (ORW), targeting applications with low duty cycles. A packet in ORW is forwarded by the node that first wakes up, successfully receives the packet and detects that it offers routing progress. This provides two key benefits over alternative routing protocols: by utilizing all neighbors as possible next hops, ORW reduces delay and energy consumption significantly when compared to unicast routing; additionally, it improves the resilience to link dynamics and node failures. Overall, ORW tailors the concept of opportunistic routing to the specific demands of WSNs and duty cycling. It integrates three key technologies: (1) the EDC anycast routing metric, inspired from the precise analytical expressions for the expected number of duty-cycles required for opportunistic forwarding, (2) mechanisms for selecting optimal forwarder sets in duty-cycled opportunistic routing and (3) coarse-grained, long-term link estimation. Our results show that ORW doubles energy efficiency in dense networks. It reduces duty cycles on average by 50% and delays by 30% to 90% while achieving reliability and transmission counts similar to the state of the art.

APPENDIX

A.1. Proof of Lemma 3.1

The first result can be verified by inspecting $f_i^{(1)}(\mathcal{A} \cup \{x\}) - f_i^{(1)}(\mathcal{A})$. For second result one can check that $f_i^{(2)}(\mathcal{A} \cup \{k\}) - f_i^{(2)}(\mathcal{A}) > 0$ given that $c_k > c_j$ for all $j \in \mathcal{A}$.

A.2. Proof of Lemma 3.2

The sign of $c_k - f_i(\mathcal{A}) + w$ is the same as the sign of $f_i(\mathcal{A} \cup \{k\}) - f_i(\mathcal{A})$, since

$$f_i(\mathcal{A} \cup \{k\}) - f_i(\mathcal{A}) = \frac{p(i,k) \cdot (c_k - f_i(\mathcal{A}) + w)}{\sum_{j \in \mathcal{A}} p(i,j) + p(i,k)}$$

and $p(i,k) > 0$. Our result follows.

A.3. Proof of Lemma 3.3

Consider

$$f_i(\mathcal{A} \cup \{k\}) - c_k = \frac{f_i(\mathcal{A}) - c_k + w \frac{p(i,k)}{\sum_{j \in \mathcal{A}} p(i,j)}}{1 + \frac{p(i,k)}{\sum_{j \in \mathcal{A}} p(i,j)}}$$

Our assumptions imply that the right-hand side is positive, hence $f_i(\mathcal{A} \cup \{k\}) > c_k$.

A.4. Proof of Theorem 3.4

Assume that $\mathcal{F}^\star(i) = \{\pi(1), \ldots, \pi(k)\} \backslash \pi(m)$ for some $m < k$, *i.e.* a node m with $c_m < c_k$ has been excluded from the forwarder set. Note that $c_j \leq f_i(\mathcal{F}^\star(i) \backslash \pi(j)) - w$, $\forall j \in \mathcal{F}^\star(i)$ because otherwise according to Lemma 3.2 $f_i(\mathcal{F}^\star(i)) > f_i(\mathcal{F}^\star(i) \backslash \pi(j))$ resulting that $\mathcal{F}^\star(i)$ is not the optimum. Moreover, from $c_j \leq f_i(\mathcal{F}^\star(i) \backslash \pi(j)) - w$ and Lemma 3.3 we conclude $c_j \leq f_i(\mathcal{F}^\star(i)) - w$, $\forall j \in \mathcal{F}^\star(i)$. Lemma 3.2 ensures that adding m with $c_m < c_k$ will decrease $f_i(\mathcal{F}^\star(i))$, *i.e.*, $f_i(\mathcal{F}^\star(i) \cup \{c_m\}) < f_i(\mathcal{F}^\star(i))$ so $\mathcal{F}^\star(i)$ is not optimal and a contradiction is achieved.

A.5. Proof of Lemma 3.5

The proof follows from the Lemma 3.2 and inspecting the inequality

$$f_i(\mathcal{A} \cup \{k'\}) - f_i(\mathcal{A} \cup \{k\}) = \frac{(p(i,k) - p(i,k'))\,(f_i(\mathcal{A}) - c_k - w)}{\Big(p(i,k) + \sum_{j \in \mathcal{A}} p(i,j)\Big)\,(\frac{p(i,k')}{\sum_{j \in \mathcal{A}} p(i,j)} + 1)} > 0.$$

A.6. Proof of Lemma 3.6

According to Lemma 3.2, at each iteration k Algorithm 1 a new member $j \in \mathcal{N}(i)$ is added to the forwarder set $\mathcal{F}^\star(i)$ for node $i \in \mathcal{N}$ if $\mathrm{EDC(j)} < \mathrm{EDC(i)} - \mathrm{w}$. By Lemma 3.3, this event can only happen since at iteration $k-1$ node $i \notin \mathcal{F}_j^\star$, i.e. otherwise $\mathrm{EDC(j)} > \mathrm{EDC(i)}$. Furthermore, upon adding j to $\mathcal{F}^\star(i)$ we have $\mathrm{EDC}(i) = f(\mathcal{F}^\star(i) \cup \{j\}) > \mathrm{EDC}(j)$ by Lemma 3.3, which ensures that node i cannot be added to the optimal forwarder set $\mathcal{F}_j^\star$ of j in future iterations of the Algorithm. Thus any two nodes $i, j \in \mathcal{N}$ can only be connected by either an arc (i, j) or (j, i) in each rooting topology $\mathcal{G}^{(k)}$.

Let now consider a path of arbitrary length $P_i = \{i, j_1, j_2, \ldots, j_n\}$ from node i to the sink in $\mathcal{G}^{(k)}$, with $j_1 \in \mathcal{F}^\star(i)$, $j_2 \in \mathcal{F}_{j_1}^\star$, and so on. Let assume that P_i has a loop involving node i, i.e. there exist a $j_k \in P_i$ such that $i \in \mathcal{F}_{j_k}^\star$, hence by Lemma 3.2 $\mathrm{EDC}(i) < \mathrm{EDC}(j_k) - w$. However, by applying Lemma 3.3 to each hop of the path P_i we have that $\mathrm{EDC}(i) > \mathrm{EDC}(j_n),\ \forall j_n \in P_i$ and a contradiction is achieved.

A.7. Proof of Theorem 3.7

Let π be an ordered set of nodes with cardinality $n = |\mathcal{N}|$ in which nodes are sorted increasingly based on their EDC values. It is sufficient to prove that in $m \geq 1$-th pass of the for loop in Algorithm 1 at line 5, at least one node will find its optimal forwarder list and remains unchanged for the rest of the algorithm; i.e. the first m nodes with lower EDC values in π converge to their optimality (in the sense of the EDC metric) after $m-1$ executions of for loop. We prove this by induction. The inductive basis is trivial since the first node is the sink, which has the lowest EDC. Now, assume that our claim holds for the first $m-1$ step. During the m-th pass, the first m node will not update their EDCs and corresponding forwarders set since they are converged to their optimality. However the remaining $n-m$ nodes still may insert beneficial forwarders in ascending order of their EDC's. After the m-th iteration, there will be one (couple of) node(s) with the same minimum value of EDC between remaining $n-m$ nodes. For the moment assume there is only one node with this condition denoted by i_{m+1}. We claim that i_{m+1} will converge to optimality in m-th iteration. Because of monotonically increasing property stated in Lemma 3.3, the EDC of $n-m$ remaining nodes will not be less than the first m nodes in π. It is easy to see that i_{m+1} after m-th iteration selects forwarders only from the first m nodes according to the strict monotonicity property stated in Lemma 3.2. Hence, $\forall j \in \mathcal{F}_{i_{m+1}}$, $\mathrm{EDC}(j)$ is optimal due to the inductive hypothesis. Since $\mathrm{EDC}(i_{m+1})$ can not be improved in upcoming iterations (due to monotonically increasing property stated in Lemma 3.3) we conclude that i_{m+1} converges to the optimality of the metric. For the case of multiple nodes with minimum EDC values after m-th iteration, all of them converge to optimality due to the fact that all of them are only utilizing first m nodes in π for their forwarders set and adding each other or remaining $n-m$ nodes will not decrease their EDC value.

REFERENCES

ALIZAI, M. H., LANDSIEDEL, O., BITSCH LINK, J. A., GÖTZ, S., AND WEHRLE, K. 2009. Bursty Traffic Over Bursty Links. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

ALLEN, G., SWIESKOWSKI, P., AND M.WELSH. 2005. MoteLab: a wireless sensor network testbed. In *IPSN: Proc. of ACM/IEEE Int. Conference on Information Processing in Sensor Networks*.

ASHREF, F., KRAVETS, R. H., AND VAIDYA, N. H. 2010. Exploiting Routing Redundancy using MAC Layer Anycast to Improve Delay in WSN. *SIGMOBILE Mob. Comput. Commun. Rev. 14*.

AUTENRIETH, M. AND FREY, H. 2011. PaderMAC: A Low-Power, Low-Latency MAC Layer with Opportunistic Forwarding Support for Wireless Sensor Networks. In *ADHOC-NOW'11: Proc. of the 10th Int. Conference on Ad-Hoc, Mobile, and Wireless Networks*.

BASU, P. AND CHAU, C.-K. 2008. Opportunistic Forwarding in Wireless Networks with Duty Cycling. In *CHANTS: Proc. of the ACM Workshop on Challenged Networks*.

BECHER, A., LANDSIEDEL, O., KUNZ, G., AND WEHRLE, K. 2008. Towards short-term wireless link quality estimation. In *Hot EmNetS'08: Proceedings of Fifth ACM Workshop on Embedded Networked Sensors*.

BISWAS, S. AND MORRIS, R. 2005. ExOR: Opportunistic Multi-Hop Routing for Wireless Networks. In *SigComm: Proc. of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*.

BUETTNER, M., YEE, G. V., ANDERSON, E., AND HAN, R. 2006. X-MAC: a Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

CHACHULSKI, S., JENNINGS, M., KATTI, S., AND KATABI, D. 2007. Trading Structure for Randomness in Wireless Opportunistic Routing. In *SigComm: Proc. of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*.

CHOUDHURY, R. R. AND VAIDYA, N. H. 2004. MAC-Layer Anycasting in Ad Hoc Networks. *SIGCOMM Comput. Commun. Rev. 34,* 1.

DE COUTO, D. S. J., AGUAYO, D., BICKET, J., AND MORRIS, R. 2003. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *MobiCom: Proc. of the ACM Int. Conference on Mobile Computing and Networking*.

DODDAVENKATAPPA, M., CHAN, M. C., AND ANANDA, A. 2011. Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed. In *TridentCom: Proc. of the Int. ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*.

DUBOIS-FERRIÉ, A. 2006. anypath routing. Ph.D. thesis, EPFL.

DUBOIS-FERRIÉ, A., GROSSGLAUSER, M., AND VETTERLI, M. 2011. Valuable Detours: Least-Cost Anypath Routing. *IEEE/ACM Trans. Netw. 19,* 2.

DUTTA, P., DAWSON-HAGGERTY, S., CHEN, Y., LIANG, C.-J. M., AND TERZIS, A. 2010. Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

FONSECA, R., GNAWALI, O., JAMIESON, K., AND LEVIS, P. 2007. Four Bit Wireless Link Estimation. In *HotNets: Proc. of the Workshop on Hot Topics in Networks*.

GHADIMI, E., LANDSIEDEL, O., SOLDATI, P., AND JOHANSSON, M. 2012. A metric for opportunistic routing in duty cycled wireless sensor networks. In *SECON'12: Proceedings of the 9th IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

GNAWALI, O., FONSECA, R., JAMIESON, K., MOSS, D., AND LEVIS, P. 2009. Collection Tree Protocol. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

GU, Y. AND HE, T. 2007. Data Forwarding in Extremely Low Duty-Cycle Sensor Networks with Unreliable Communication Links. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

HANDZISKI, V., KÖPKE, A., WILLIG, A., AND WOLISZ, A. 2006. TWIST: a Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks. In *Proc. of the Int. Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality*.

HANSEN, M., KUSY, B., JURDAK, R., AND LANGENDOEN, K. 2012. AutoSync: Automatic duty-cycle control for synchronous low-power listening. In *SECON: 9th IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

JURDAK, R., BALDI, P., AND VIDEIRA LOPES, C. 2007. Adaptive low power listening for wireless sensor networks. *IEEE Transactions on Mobile Computing 6,* 8.

KIM, D. AND LIU, M. 2008. Optimal stochastic routing in low duty-cycled wireless sensor networks. In *Proceedings of the 4th Annual International Conference on Wireless Internet*. WICON '08.

KIM, J., LIN, X., SHROFF, N. B., AND SINHA, P. 2008. On Maximizing the Lifetime of Delay-Sensitive Wireless Sensor Networks with Anycast. In *InfoCom: Proc. of the IEEE Int. Conference on Computer Communications*.

KIM, J., LIN, X., SHROFF, N. B., AND SINHA, P. 2010. Minimizing Delay and Maximizing Lifetime for Wireless Sensor Networks with Anycast. *IEEE/ACM Trans. Netw. 18*.

LANDSIEDEL, O., GHADIMI, E., DUQUENNOY, S., AND JOHANSSON, M. 2012. Low power, low delay: Opportunistic routing meets duty cycling. In *IPSN'12: Proceedings of the 11th ACM/IEEE International Conference on Information Processing in Sensor Networks*.

LARSSON, P. 2001. Selection Diversity Forwarding in a Multihop Packet Rradio Network with Fading Channel and Capture. *SIGMOBILE Mob. Comput. Commun. Rev. 5*.

LIU, S., FAN, K.-W., AND SINHA, P. 2007. CMAC: An Energy Efficient MAC Layer Protocol Using Convergent Packet Forwarding for Wireless Sensor Networks. In *SECON: Proc. of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

LIU, S., FAN, K.-W., AND SINHA, P. 2009. CMAC: An Energy-Efficient MAC Layer Protocol using Convergent Packet forwarding for Wireless Sensor Networks. *ACM Trans. Sen. Netw. 5*.

LIU, S., SHA, M., AND HUANG, L. 2010. ORAS: Opportunistic Routing with Asynchronous Sleep in Wireless Sensor Networks. In *ICFCC'10: Proc. of the 2nd Int. Conference on Future Computer and Communication*.

LIU, T. AND CERPA, A. 2011. Foresee (4C): Wireless link prediction using link features. In *In proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN 2010)*.

LIU, T. AND CERPA, A. 2012. TALENT: Temporal adaptive link estimator with no training. In *In proceedings of the 10th ACM conference on Embedded Network Sensor Systems (SenSys 2012)*.

LU, M. AND WU, J. 2009. Opportunistic routing algebra and its applications. In *InfoCom: Proc. of the IEEE Int. Conference on Computer Communications*.

MAO, X., TANG, S., XU, X., LI, X., AND MA, H. 2011. Energy-efficient opportunistic routing in wireless sensor networks. *IEEE Trans. Parallel and Distributed Systems 22,* 11.

MEIER, A. F., WOEHRLE, M., ZIMMERLING, M., AND THIELE, L. 2010. Zerocal: Automatic mac protocol calibration. In *Proc. 6th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*.

MOELLER, S., SRIDHARAN, A., KRISHNAMACHARI, B., AND GNAWALI, O. 2010. Routing without Routes: the Backpressure Collection Protocol. In *IPSN: Proc. of the ACM/IEEE Int. Conference on Information Processing in Sensor Networks*.

MOSS, D. AND LEVIS, P. 2008. BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking. Tech. Rep. SING-08-00, Stanford.

PAVKOVIĆ, B., THEOLEYRE, F., AND DUDA, A. 2011. Multipath Opportunistic RPL Routing over IEEE 802.15.4. In *MSWiM: Proc. of the ACM Int. Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*.

POLASTRE, J., HILL, J., AND CULLER, D. 2004. Versatile Low Power Media Access for Wireless Sensor Networks. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

PUCCINELLI, D., ZUNIGA, M., GIORDANO, S., AND MARRON, P. J. 2012. Broadcast-Free Collection Protocol. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

SCHAEFER, G., INGELREST, F., AND VETTERLI, M. 2009. Potentials of Opportunistic Routing in Energy-Constrained Wireless Sensor Networks. In *EWSN: Proc. of the European Conference on Wireless Sensor Networks*.

SRINIVASAN, K., JAIN, M., CHOI, J. I., AZIM, T., KIM, E. S., LEVIS, P., AND KRISHNAMACHARI, B. 2010. The κ Factor: Inferring Protocol Performance using Inter-Link Reception Correlation. In *MobiCom: Proc. of the ACM Int. Conference on Mobile Computing and Networking*.

SRINIVASAN, K., KAZANDJIEVA, M. A., AGARWAL, S., AND LEVIS, P. 2008. The β Factor: Measuring Wireless Link Burstiness. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

UNTERSCHÜTZ, S., RENNER, C., AND TURAU, V. 2012. Opportunistic, Receiver-Initiated Data-Collection Protocol. In *EWSN: Proc. of the European Conference on Wireless Sensor Networks*.

VANHIE-VAN GERWEN, J., DE POORTER, E., LATRÉ, B., MOERMAN, I., AND DEMEESTER, P. 2010. Real-Life Performance of Protocol Combinations for Wireless Sensor Networks. In *Proc. of the IEEE Int. Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*.

WINTER (ED.), T., THUBERT (ED.), P., AND TEAM, R. A. Rpl: Ipv6 routing protocol for low power and lossy networks. Internet Draft draft-ietf-roll-rpl-19, work in progress.

WOO, A., TONG, T., AND CULLER, D. 2003. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

XUE, Y., VURAN, M., AND RAMAMURTHY, B. 2010. Cost Efficiency of Anycast-Based Forwarding in Duty-Cycled WSNs with Lossy Channel. In *SECON: Proc. of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

YE, W., SILVA, F., AND HEIDEMANN, J. 2006. Ultra-Low Duty Cycle MAC with Scheduled Channel Polling. In *SenSys: Proc. of the ACM Int. Conference on Embedded Networked Sensor Systems*.

ZHONG, Z. AND NELAKUDITI, S. 2007. On the Efficacy of Opportunistic Routing. In *SECON: Proc. of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

ZORZI, M. AND RAO, R. R. 2003a. Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Latency Performance. *IEEE Trans. on Mobile Computing 2*, 349–365.

ZORZI, M. AND RAO, R. R. 2003b. Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Multihop Performance. *IEEE Trans. on Mobile Computing 2*, 337–348.