

link quality estimation strategy for RPL and its impact on topology management,
Computer Communications, Volume 112, 2017, Pages 1-13, ISSN 0140-3664,
<https://doi.org/10.1016/j.comcom.2017.08.005>.

A Reinforcement Learning-based Link Quality Estimation Strategy for RPL and its Impact on Topology Management

Emilio Ancillotti^a, Carlo Vallati^{b,*}, Raffaele Bruno^a, Enzo Mingozzi^b

^a*Institute for Informatics and Telematics (IIT) – CNR, V. Giuseppe Moruzzi 1, 56124 Pisa, ITALY*

^b*Dipartimento di Ingegneria dell'Informazione, University of Pisa, Via Diotisalvi, 2, 56122 Pisa, ITALY*

Abstract

Over the last few years, standardisation efforts are consolidating the role of the Routing Protocol for Low-Power and Lossy Networks (RPL) as the standard routing protocol for IPv6-based Wireless Sensor Networks (WSNs). Although many core functionalities are well defined, others are left implementation dependent. Among them, the definition of an efficient link-quality estimation (LQE) strategy is of paramount importance, as it influences significantly both the quality of the selected network routes and nodes' energy consumption. In this paper, we present RL-Probe, a novel strategy for link quality monitoring in RPL, which accurately measures link qualities with minimal overheads and energy waste. To achieve this goal, RL-Probe leverages both synchronous and asynchronous monitoring schemes to maintain up-to-date information on link qualities and to promptly react to sudden topology changes, e.g. due to mobility. Our solution relies on a reinforcement learning model to drive the monitoring procedures in order to minimise the overhead caused by active probing operations. The performance of the proposed solution is assessed by means of simulations and real experiments. Results demonstrated that RL-Probe helps in effectively improving packet loss rates, allowing nodes to promptly react to link quality variations as well as to link failures due to node mobility.

Keywords: RPL, topology and mobility management, link quality estimation, experimental evaluation.

1. Introduction

The future Internet of Things (IoT) foresees information systems seamlessly integrated with smart objects, i.e. daily objects empowered with computation and communication capabilities. This vision will require sensors and actuators deployed on a large scale for ubiquitous sensing and remote control of physical systems. In this context, Wireless Sensor Networks (WSNs) will represent a key enabler to guarantee low-cost and rapid deployment of IoT devices exploiting multi-hop data delivery over wireless links. Efficiency and reliability of multi-hop data forwarding will be essential to support future IoT systems and guarantees their robustness [1].

In general, the main objective of WSN routing protocols is to enable reliable communication while minimising resource consumptions [2]. This is particularly important in low-power and lossy networks (LLNs), because they are characterised by unreliable links whose quality might significantly fluctuate over time influenced by external interference or obstacles. In this context, the selection of the optimal path is, however, challenging, as gathering topology information requires communication and processing

overhead that contrasts with the limited computational and energy capabilities available to constrained devices that are usually battery powered.

Recently, significant efforts were put into defining a common routing protocol for IP-based WSNs, which have led to the standardisation of RPL, a gradient-based routing protocol that aims at building a robust multi-hop mesh topology over lossy links with minimal state requirements [3]. However, several studies have demonstrated that RPL is affected by reliability issues. The root cause of this unreliability is the lack of responsiveness to variations of network conditions that arise in real-life scenarios due to node mobility, or environmental factors, such as interference, multi-path effects, irregular radios patterns among the others [4]. Hence, many extensions have been recently proposed for the original RPL specification to support routing optimisations and more efficient mechanisms for route discovery and topology repair, especially under mobility [5] [6] [7] [8] [9] [10] [11]. However, *tweaking routing procedures does not solve the core problem of how to proactively maintain up-to-date information about links quality and network routes in a highly efficient manner.*

More generally, the availability of a highly efficient, accurate, and adaptive link-quality estimation (LQE) framework is essential to allow any routing protocol to select the best routing path under time-varying network conditions. Thus, LQE in wireless sensor networks has received significant attention over the past years [12] [13]. Unfortunately, there

*Corresponding author

Email addresses: emilio.ancillotti@iit.cnr.it (Emilio Ancillotti), carlo.vallati@iet.unipi.it (Carlo Vallati), raffaele.bruno@iit.cnr.it (Raffaele Bruno), enzo.mingozzi@iet.unipi.it (Enzo Mingozzi)

are several limitations in using existing LQE techniques with RPL. In particular, broadcast-based probing is commonly adopted for LQE in low-power wireless networks because it incurs a lower overhead than unicast-based probing. However, RPL employs the Trickle algorithm [14] to dynamical control the dissemination of routing control information, which makes complicate the implementation of broadcast-based probing without affecting normal RPL operations. Furthermore, even with broadcast-based probing the measurement overhead increases almost linearly with the number of neighbours and the probing frequency, thus consuming precious energy resources and worsening network congestion. On the other hand, infrequent or on-demand probing would yield poor measurement accuracy and reduce routing ability to adapt to the link dynamics in real time. A few approaches have been proposed to support adaptive LQE in multi-hop wireless networks to minimise measurement overhead. Unfortunately, these existing techniques cannot be used in LLNs because they require the maintenance of large link-state tables or exploit cross-traffic overhearing [15].

To address the above issues, in this paper we propose a novel *lightweight link monitoring scheme*, called RL-Probe, which ensures a good trade-off between routing reliability, responsiveness to variable network conditions, and low overheads. The salient features of RL-Probe can be summarised as follows. First, RL-Probe *combines synchronous and asynchronous monitoring mechanisms* to maintain up-to-date information about link qualities and their temporal variations while promptly reacting to sudden and unpredictable changes in network and link conditions. Secondly, RPL-probe uses *a reinforcement learning technique based on the multi-armed bandit model* [16] to dynamically control the probing procedures in order to automatically minimise measurement overheads without affecting responsiveness to route changes. Thirdly, RL-Probe preserves backward compatibility with standard RPL, i.e., enabling the interoperability of standard and enhanced nodes in the same network. It is also important to point out the objective of RL-Probe is not to provide a better routing metric for RPL, but a new measurement methodology that can be applied to generic link metrics, such as ETX [17].

To evaluate the effectiveness of our solution, we have integrated RL-Probe within the Contiki RPL/6LowPAN protocol stack. The proposed solution has been compared against the standard RPL configuration, in which no active probe is employed, and RPL with periodic probing, in which active probe traffic is exploited to monitor links and handle channel quality variations. In addition to static scenarios, experiments involving mobile nodes have been run to analyse the performance of RL-Probe in challenging conditions. Indeed, mobile nodes are characterised by rapidly-changing link quality conditions that are usually managed using ad-hoc extensions of RPL rather than standard LQE solutions. In order to offer a fair comparison, mRPL, a state-of-the-art RPL enhancement explicitly designed to cope with mobility, is also considered in the

scenarios in which mobility is involved. We conduct both simulations and experiments in an indoor testbed in a broad range of static and dynamic scenarios. Results obtained under static conditions show that RL-Probe can handle link quality variations with a significant advantage over existing solutions. The results of the experiments involving mobility show that RL-Probe can rapidly detect sudden link quality variations on the mobile node representing a viable alternative to handle mobility through accurate LQE. Specifically, experiment results show that RL-Probe guarantees similar performance compared mRPL with an overhead that is significantly lower in terms of control messages and energy consumption.

The remaining of this paper is organised as follows. Section 2 provides an overview of related work. In Section 3 we present RL-Probe and we evaluate its performance in Section 4. Finally, in Section 5 we draw our conclusions.

2. Background and Related Work

In the following subsections, we first provide background information on RPL, then, we overview the works that are the most relevant to our study. Specifically, we review both LQE approaches for RPL aimed at monitoring links quality, and mechanisms proposed to improve mobility support in RPL.

2.1. Background on RPL

RPL is an IPv6-based routing protocol specifically designed for lossy environments and resource-constrained embedded devices [3]. Specifically, RPL employs a distance vector routing algorithm which builds a logical topology on top of the physical network. In particular, the topology is a *Destination Oriented Directed Acyclic Graph, DODAG* for short. The root node of the DODAG initialises the DODAG formation by emitting DODAG *Information Object* messages (hereafter *DIOs* for short). Non-root nodes listen for DIOs and use the included information to join a DODAG. As a node joins a DODAG, it starts advertising its presence through the emission of DIO messages. Each DIO message specifies the rank of the sender, which is a scalar measure of the distance of that node from the root [1]. To avoid loops in the logical topology the rank must monotonically decrease along an upward path towards the DODAG root.

As DIO messages are received from the neighbours, each node updates its view of the topology. In particular, node's neighbours with lower rank are selected to form a *parent set* which is used for data forwarding. Among them, a *preferred parent* is selected to forward traffic towards the root. Other important RPL control messages are the: (i)

¹The rank of each node is computed on the basis of an *Objective Function* (*OF* for short), which also defines how nodes select parents. Although the rank is not meant as a path cost, it is typically obtained from path metrics that are somehow function of the distance from the root, e.g., number of hops or end-to-end packet delays.

DODAG Information Solicitation (DIS), which is a multicast message used to trigger the transmission of DIO messages from neighbours; and (ii) *Destination Advertisement Object (DAO)*, which is propagated upward (along the DODAG) to build the downward routes. To reduce the overhead associated to routing signalling RPL use a Trickle-based strategy [18, 14], an adaptive beaconing scheme that exponentially increases the transmission timers when the network conditions are considered stable. Thus, the periodic transmission of control packets, scheduled by the Trickle algorithm, is the main mechanisms employed by RPL to detect topological changes. In addition, RPL uses the Neighbour Discovery Protocol (NDP, defined in RFC 4861 [19]) to determine if a neighbour is no longer reachable on a link. However, in NDP a router advertises its presence flooding the network with special messages, the router advertisements, which may easily lead to excessive protocol overheads. Therefore, in 6LoWPAN networks router advertisements are typically sent only in response to router solicitation messages, which limits the responsiveness of the neighbour unreachability detection. Finally, when a network inconsistency is detected (e.g., a network loop or preferred parent change) a repair procedure is triggered. This repair mechanism can be local (e.g., the node detaches from the DODAG and tries to reconnect) or global (the current DOAG is invalidated and a new DODAG is built).

2.2. LQE in RPL

There is a large body of research on LQE in WSNs and it is out of the scope of this paper to overview the several link quality metrics that exist in the literature (the interested reader is referred to the comprehensive survey [13]).

From a general perspective, a LQE framework consists of three components: *i)* *link monitoring*, which is the mechanism used to collect link measurements; *ii)* *link measurement*, which specifies the information to be retrieved, and *iii)* *metric evaluation*, which defines the metric to assess the link quality. First RPL implementations employed simple passive monitoring techniques, which leverage on statistics of transmission failures for the links used by data traffic [20]. However, data-driven link estimation methods do not apply to idle links, and the reactivity of such methods heavily depends on the network traffic patterns. Furthermore, over-hearing is required to monitor links to neighbouring nodes that are not the preferred parent [21]. Thus, recent RPL implementations prefer active monitoring to measure link qualities through probes. The Contiki 3.0 RPL implementation, for instance, includes an optional probing mechanisms (dubbed RPL-PP), in which neighbours are probed periodically through unicast DIO messages. The target of the probe is selected following a round robin strategy, except the link to the preferred parent that pre-empts the other links to neighbouring nodes in the probing schedule if its quality has not been updated in a given interval.

It is generally agreed that unicast-based probing provides accurate link measurements [22]. The downside is that neighbours are assessed individually, which results

into long convergence times for link quality estimation, especially when large or dense networks are considered. In order to reduce the measurement delay, a broadcast-based probing has been proposed in many RPL extensions. This is implemented in different ways, e.g. forcing the periodicity of routing control messages [23], or sending bursts of RPL control messages during specific phases of network operations (e.g. at network formation) or at the occurrence of certain events (e.g. during parent changes) [8]. Those schemes are sender-initiated, as the transmitting node generates the probe packets. On the other hand, some RPL variants use receiver-side probing schemes, which use information from received probes to estimate link quality. For instance, in [7] each node triggers the link monitoring by sending a multicast DIS message to its neighbours, which reply with a train of unicast DIS messages. The advantage is that multiple neighbours can be monitored in parallel even if unicast probes are used. One potential drawback of this approach is that the metric computation is performed on the opposite direction to data transmission. However, many experimental studies show the symmetry of wireless links in real use cases of WSNs [12, 24].

All the above-described mechanisms adopt basic measurement schemes. However, there are a few examples of more sophisticated solutions for LQE, which use a hybrid approach by combining multiple complementary methods. For instance, a link-quality measurement framework is proposed in [15], which combines three measurement techniques: passive, cooperative, and active monitoring. However, this framework cannot be easily applied in low-power wireless networks because it requires the maintenance of large link-state tables and leverage on cross-traffic overhearing. Differently from this prior work, RL-Probe proposes a lightweight framework that can be implemented in constrained devices in which different methods for LQE are adopted.

2.3. Machine Learning and LQE

Machine learning techniques are frequently used for LQE, also in low-power wireless networks. For instance, fuzzy logic is used in [9] and [25], and fuzzy rules are defined to combine multiple link metrics while compensating for the uncertainties in the wireless channel conditions. Supervised and online regression classifiers are proposed in [26] and [27, 28], respectively, to predict the quality of a wireless link in the near future based on historical information. Differently from prior work, in this study we do not apply machine learning methods to the metric evaluation but to the process of link measurement.

2.4. RPL Mobility Extensions

A survey of recent RPL extensions and modifications to improve mobility support is provided in [10]. Prior work has shown that RPL provides a fast network setup but that mobility support is not adequate and it should be

improved [29]. Most of existing solutions for mobility management in RPL focus on modifying native RPL mechanisms to improve RPL ability of detecting link failures, or to speed up the handoff and local repair procedures.

In ME-RPL [5] nodes are separated as *mobile* nodes (i.e. nodes with unstable links) and *fixed* nodes, with mobile nodes forced to act only as leaf nodes to avoid network paths through them. Then, mobile nodes must advertise their mobility status in control messages and generate frequent DIS messages to update their parent set information. The time between DIS messages is computed based on the frequency of preferred parent changes using a simple multiplicative increase/multiplicative decrease algorithm.

MoMoRo [9] quickly reacts to packet transmission failures by immediately resending the lost packet, and starting a new route search (by broadcasting beacons and collecting updated routing information from the neighbours) if the second attempt also fails. Furthermore, MoMoRo uses a fuzzy estimator that combines different link metrics (ETX, average RSS, symbol error rate variance) to classify links based on their probability of disconnection.

As for ME-RPL, the authors in [6] proposes that mobile nodes only connect as leaves in the DODAG. Furthermore, a reverse Trickle timer – the timer starts from the maximum value and halves the DIO sending intervals after each new DIO transmission – is used by the preferred parents of mobile nodes to trigger DIO messages. The implicit assumption of this approach is that mobile nodes’ stability decreases with time. An adaptation of the Trickle timer algorithm for better mobility support is also proposed in mod-RPL [11]. Specifically, mod-RPL takes into account the trajectory and velocity of mobile nodes when selecting the sending interval of DIO messages, and it dynamically adapts the timer to the distance between the mobile node and its preferred parent.

KP-RPL is a position-based extension of RPL to provide mobility support [30]. Standard RPL is used for routing between fixed nodes, while mobile nodes make their routing decisions using positioning information obtained by applying a Kalman filter on RSSI measurements. Furthermore, mobile nodes generate a blacklist to discard neighbours that could not be reachable due to positioning errors.

In conclusion, all the above-described schemes are specifically designed to handle mobility through ad-hoc mechanisms that usually do not perform accurate LQE. Differently from such solutions, our proposed approach is not restricted to support only node mobility, but it can seamlessly manage node mobility through fine-grained LQE, without requiring modifications of standard RPL or a-priori knowledge of which nodes are mobile.

Overview of mRPL. We present mRPL [8] in more details as it will be used as benchmark in the following evaluation.

Basically, mRPL integrates smart-Hop, a handoff scheme for low-power networks [31], in RPL. As in [5, 6, 30] mobile and fixed nodes are separated. Then, mobile nodes

monitor the link quality by receiving DIO messages from their preferred parents. A mobile node disconnects from the preferred parent and enters a discovery phase if the average received signal strength is below a given threshold or if no packets are received by the parent before a connectivity timer (T_C) expiration. To avoid that the high variability of wireless links causes frequent handoffs a hysteresis mechanism is applied to this RSSI threshold. During the discovery phase, the mobile node multicasts DIS messages to all neighbouring nodes and collects their unicast DIO replies to decide which new preferred parent to select based on the average RSSI level. As an additional stability mechanism, a mobile node repeats the discovery procedure m times after switching to a different preferred parent to check the stability of that node.

mRPL introduces additional timers to increase handoff efficiency and reliability. In particular, the mobility detection timer (T_{MD}) is used to detect connection losses due to the existence of external objects (obstacles between the sender and the receiver). The handoff timer establishes the transmission period of DIS messages to the neighbouring parents. Finally, a reply timer (T_R) is used to select the time instant at which a parent should reply to the mobile node to reduce the probability of collision between control messages during the handoff process.

3. The RL-Probe framework

This section describes RL-Probe in details. First, we provide background material on the multi-armed bandit model. Then, we explain the design rationale behind RL-Probe. Finally, we detail the main mechanisms and algorithms used in RL-Probe.

3.1. Multi-armed Bandit Background

Generally, a multi-armed bandit (MAB) model is used to describe a learning problem in which an agent must repeatedly choose among different options, or actions [16]. After each choice, the agent receives a reward chosen from an *unknown* stationary probability distribution that depends on the selected action. The objective of the agent is to maximise the expected total reward over a time horizon. The MAB model is so named by analogy to a slot machine that has multiple levers, with different but unknown probabilities of hitting the jackpot. Then, the player should try to find the best levers that maximise the winnings by repeatedly playing.

More formally, let us assume that time is discretised and time slots are numbered as $n = 1, 2, \dots$. Then, let \mathcal{U} be the set of actions (or arms). At round n , the arm pulled is $u^{(n)}$ and the reward received is $W^{(n+1)}$. We assume that the reward provided by arm u follows a random distribution $F_u(x)$, which is unknown. Now, let $s^{(n)}$ be the representation of the system state at round n and let \mathcal{S} be the discrete set of possible states. The history of the system at a given stage is the sequence of decisions, observed states

and collected rewards. For the sake of tractability, MAB models usually assume the *Markov property*, i.e., rewards depend only on the current state and the current action and not on the full history of previous actions and states. Then, the core of a MAB model is the policy π , namely the mapping function between states and actions, which should maximise the amount of rewards the agent receives over time. Several methods have been proposed in the literature to learn the optimal policy without requiring a model of the system behaviours, but leveraging only on the experience obtained by iteratively interacting with the system. As discussed more in depth in the following sections, these learning techniques have to cope with the *exploration/exploitation dilemma*, which implies balancing immediate gains (i.e., selecting the action with the maximum expected reward) with knowledge creation to make better decisions in the future (i.e., selecting actions that appear to be worse but could potentially be the best). Typically, a probabilistic learning strategy is defined that assigns a probability to each possible action in a state according to an estimation of the current state value.

3.2. Overview of RL-Probe

One of the main distinct features of RL-Probe over existing probing strategies for LLNs is that it adopts a *hybrid* approach to adaptively combine *synchronous* and *asynchronous* LQE techniques. More precisely, the synchronous LQE technique relies on unicast probes to provide accurate link quality measurements. However, we design two novel mechanisms to make unicast-based probing more adaptive and responsive, without introducing excessive probing overheads. First, we formulate the selection of the probing period as a multi-armed bandit problem to dynamically adjust the probing frequency to the link variability in real time. Our learning-based approach provides a good trade-off between overhead and responsiveness to varying link qualities. Secondly, we cluster neighbours into groups and we assign different probing priorities to each group. The clustering and the priority selection are based on the importance of each node in RPL route maintenance and recovery procedures. The rationale behind this approach is that wireless link correlation (e.g., due to cross-network interference under shared medium) has been observed in many recent studies [32]. Consequently, independent estimates of link qualities are likely to be superfluous, especially in dense networks.

Our asynchronous LQE mechanism is designed to efficiently handle sudden and disruptive link variations. To this end, we integrate in RL-Probe a receiver-side probing method first proposed in [33], which allows to rapidly assess the quality of the links from a node to all its neighbours with a sufficient accuracy. In principle, asynchronous probing could facilitate the isolation of faulty nodes/links, or the detection of preferred parent unavailability due to mobility. Clearly, asynchronous probing must be activated on-demand when it is most likely needed because it is costly in terms of energy and bandwidth consumption. Thus, we

Algorithm 1 Main procedure of RL-Probe

Require: \mathcal{N}_i ▷ set of neighbours

```

1: loop
2:   if (received packet  $p$  from  $j$ ) then
3:      $rss_i \leftarrow \text{GETRSSI}(p)$ ;
4:      $rss_{i,j}[] \leftarrow \text{add}(rss_i)$ ;
5:      $rssiTrend_{i,j} \leftarrow \text{CALCRSSITREND}(rss_{i,j}[])$ ;
6:     if ( $p = \text{ACK}$  and  $j = pp$ ) then
7:        $\Delta rss_{i,j} = \frac{rcvTh - rss_i}{rcvTh}$ ;
8:       if ( $rssiTrend_{i,j} < 0$  and  $\Delta rss_{i,j} \leq \alpha$ ) then
9:         do receiver-side probing;
10:        for all  $j \in \mathcal{N}_i$  do
11:           $\text{UPDATELQE}(i,j)$ ;
12:        end for
13:      end if
14:     else if ( $p = \text{NACK}$  and  $j = pp$  and
15:        $cv_{i,j}[] \text{.get}(\text{last}) \leq \beta$ ) then
16:       do receiver-side probing;
17:       for all  $j \in \mathcal{N}_i$  do
18:          $\text{UPDATELQE}(i,j)$ ;
19:       end for
20:     else if (timer  $T_p$  expires) then
21:        $\text{UPDATECLUSTERS}(\mathcal{P}_i, \mathcal{O}_i)$ ;
22:        $x \leftarrow \text{Uniform}(0, 1)$ ;
23:       if ( $x \leq \epsilon$ ) then ▷ exploitation phase
24:          $u \leftarrow \underset{x}{\text{argmax}} \left( W^{(x)}[] \text{.get}(\text{last}) \right)$ ;
25:       else ▷ exploration phase
26:          $u \leftarrow \text{random}(D_1, D_2, D_3)$ ;
27:       end if
28:       if ( $x = D_1$ ) then
29:          $j \leftarrow \text{BESTNODE}(\mathcal{P}_i)$ 
30:       else if ( $x = D_2$ ) then
31:          $j \leftarrow \text{BESTNODE}(\mathcal{O}_i)$ 
32:       end if
33:       if ( $x \neq D_3$ ) then
34:         do unicast probing to  $j$ ;
35:          $\text{UPDATELQE}(i, j)$ ;
36:          $\text{UPDATEUTILITY}(i, j)$ ;
37:       end if
38:        $\text{UDAPTEREWARD}(i,j,u)$ 
39:     end if
40:     if (sent data packet to  $pp$ ) then
41:        $\text{UPDATELQE}(i, pp)$ ;
42:        $\text{UPDATEUTILITY}(i, pp)$ ;
43:     end if
44:   end loop

```

also propose specific triggering mechanisms for our asynchronous LQE technique, which are based on the trend of received signal strength indicator (RSSI) values and link qualities. Our solution reduces the cost of recovering RPL connectivity by ensuring quick detection of network disruptions without depleting the limited resources of the devices.

For the sake of clarity, the pseudocode of the main mechanisms of RL-Probe are provided in Algorithm 1 and Algorithm 2.

3.3. Asynchronous Probing

As outlined above, the proposed asynchronous probing scheme exploits the measurements of RSSI values to detect sudden link variations. More precisely, whenever a node i receives a packet from a neighbour j , it obtains the RSSI value from the wireless transceiver (line 3 in Algorithm 1). A list of recent RSSI values is maintained for each link $l_{i,j}$ (line 4), which is used to estimate the trend in RSSI variations (line 5). For the sake of computational efficiency, in our implementation we estimate the RSSI trend using a simple moving average (SMA) filter over the last three measurements of RSSI differences between consecutively received packets. Then, our asynchronous probing has both a *proactive* and *reactive* phase. The proactive phase tries to anticipate topology changes and it is activated when a packet is successfully transmitted to the preferred parent pp (i.e., the sender receives a MAC ACK from the preferred parent). If both the RSSI trend is negative and the last RSSI sample is close to the receiver sensitivity $rcvTh$, we predict that the link quality of $l_{i,j}$ is worsening (line 8). This condition triggers our receiver-side probing² to update the link qualities to all neighbours and to quickly search for alternative links (lines 9-11). On the other hand, the reactive phase is designed to facilitate local repair operations after unexpected network disruptions. In this case, the receiver-side probing is activated when there is a transmission failure (i.e., the sender receives a MAC NACK from the preferred parent) on a link that has a stable link quality. The link quality stability is measured using the coefficient of variation of the link quality (defined as the ratio between the standard deviation $\sigma_{i,j}$ and the mean $\mu_{i,j}$), which is maintained by the function $UPDATELQE(i, j)$. More precisely, a link is considered stable if $cv_{i,j} \leq \beta$ (line 14). Note that the selection of threshold β is dictated by the variability of the wireless links. Links with $cv_{i,j}$ lower than one are considered low-variance. Summarising, both a successful transmission on a rapidly degrading link and a packet loss on a stable link activates the asynchronous probing for updating the link quality to all nodes in the neighbourhood (lines 15-17).

²We recall that with receiver-side probing a node sends a multicast DIS message and its neighbouring nodes reply with a train of unicast DIS messages.

3.4. Synchronous Probing

Differently from conventional unicast-based probing schemes, our synchronous probing does not use a fixed probing interval, which would cause unacceptable convergence delays for the link quality estimation, especially in dense networks. On the contrary, RL-Probe clusters nodes into separate groups and adaptively adjusts the probing period for each group. Different approaches for such link clustering can be devised, taking also advantage of cross-layer information from the network layer. For instance, in this study we define a link clustering that considers the importance of a neighbour to maintain good RPL connectivity. More precisely, let us denote with \mathcal{N}_i the set of neighbours of node i . We define the set \mathcal{P}_i that contains the best m_p parents of node i , i.e. parents that have the lowest path cost to the sink if selected as next hop by node i . Similarly, we define the set \mathcal{O}_i that contains up to m_o nodes from the set $\mathcal{N}_i \setminus \mathcal{P}_i$, which have the lowest path cost to the sink if selected as next hop by node i . The m_p and m_o parameters should be selected as a trade-off between reliability and responsiveness. Large m_p and m_o values provide coarse grained information about the links and decrease the responsiveness of the system. On the other hand, low m_p and m_o values reduce the possibility to discover good alternative paths in case of loss of the preferred parent. In general, $m_o > m_p$ as it is more critical to have detailed information on nodes of the parent set.

The decision-making process that determines which set to probe and how frequently to probe it is formalised through a MAB model. Specifically, we define three possible actions as follows:

D_1 : probe a node in \mathcal{P}_i ;

D_2 : probe a node in \mathcal{O}_i ;

D_3 : skip the probing.

Each of the above decisions corresponds to an independent arm in the MAB problem. The reward associated to each probing decision, say $W^{(x)}$ with $x \in \{D_1, D_2, D_3\}$, corresponds to the potential *gain* provided by probing a node in a specific group. For instance, the gain can be a measure of the improved network responsiveness to link quality variations. Details on reward estimation are provided later in this section.

First of all, let us explain which is the policy used by node i to select the probing action given the knowledge of the average rewards. In this study, we adopt the well-known ϵ -greedy algorithm, which selects with probability ϵ the action with the maximum accumulated reward, or *greedy action* (lines 23-24), and with probability $(1 - \epsilon)$ selects a random action (lines 25-26). By properly tuning the ϵ parameter it is possible to balance exploration and exploitation phases to fast converge to the optimal action selection. If action D_1 or D_2 are selected it is necessary to decide which neighbour to probe in the set \mathcal{P}_i and \mathcal{O}_i , respectively. As anticipated above, we exploit a measure

Algorithm 2 Functions for reward computation

```
1: function UPDATEUTILITY( $i, j$ )  $\triangleright l_{i,j}$  probed link
2:    $\omega_{i,j}[] \leftarrow \omega_{i,j}[] \cdot \text{get}(\text{last}) + \sigma_{i,j}[] \cdot \text{get}(\text{last});$ 
3:    $\Delta\omega_{i,j}[] \leftarrow \omega_{i,j}[] \cdot \text{get}(\text{last}) - \omega_{i,j}[] \cdot \text{get}(\text{last} - 1);$ 
4:   if ( $\Delta\omega_{i,j} \cdot \text{get}(\text{last}) > 0$ ) then
5:      $U_{i,j} \leftarrow U_{i,j} + |\Delta\omega_{i,j} \cdot \text{get}(\text{last})|;$ 
6:   else
7:      $U_{i,j}(n) \leftarrow 0;$ 
8:   end if
9: end function

10: function COMPUTEREWARD( $i, j, u$ )  $\triangleright u$  MAB action
11:   if ( $u == D_1$ ) then
12:      $W_i^{(D_1)}[] \leftarrow \max \left[ 0, \max_{j \in \mathcal{P}_i} U_{i,j}(n) - C_1 \right];$ 
13:   else if ( $u == D_2$ ) then
14:      $W_i^{(D_2)}[] \leftarrow \max \left[ 0, \max_{j \in \mathcal{O}_i} U_{i,j}(n) - C_2 \right];$ 
15:   else
16:      $W_i^{(D_3)}[] \leftarrow \max [0, G_{np} - U_{i,pp}(n)];$ 
17:   end if
18: end function
```

of the utility of a node for the RPL topology maintenance procedure. Note that this utility measure is also used in the reward computation. In the following we formalise the algorithms used in RL-Probe to compute the utility and reward metrics.

Utility and reward computation. In RL-Probe, the reward function is mainly used to estimate the trends in link quality variations (e.g., quality degradation for an interfered link). To this end, we follow the same approach as in [34] and we use the *mean* and the *standard deviation* of the link quality metric. More formally, let us assume that each node maintains a list of the estimated values of the mean $\mu_{i,j}$ and standard deviation $\sigma_{i,j}$ of link $l_{i,j}$.³ We introduce an aggregate measure $\omega_{i,j}$ of the link quality variability, as the sum of $\mu_{i,j}$ and $\sigma_{i,j}$ (line 2 in Algorithm 2). We can easily compute the incremental variation of the link quality variability as the difference of two consecutive samples of $\omega_{i,j}$ (line 3). Intuitively, it might be appropriate to monitor more frequently links that are showing a clear *trend*, in order to timely identify a link that is quickly degrading (e.g., due to an external interference) or improving. Thus, we associate a high utility to links that show a consistent link quality variation in the last two probes (lines 4-5), while we assign a null utility to links that are not characterised by a steady (positive or negative) trend (line 7). Now, we can also clarify how the BESTNODE(\mathcal{A}) function chooses the neighbour to probe in set \mathcal{A} . In the simplest case, it could select the node with the highest utility. However, this would make impossible to check, even infrequently, links with small utilities (i.e., more variable links). Following

the same line of reasoning of the above-discussed ϵ -greedy exploration strategy, the BESTNODE(\mathcal{A}) function selects the node with the highest utility with probability ϵ , while a random link in the set \mathcal{A} in the other cases.

Commonly, reward functions for learning problems should include a positive term and a negative term to be well specified. The positive term measures the gain of performing that action. For the case of actions D_1 and D_2 the gain is given by the highest utility value in the group (lines 12 and 14). Thus, the reward of a link cluster is high if there is at least one link in the set with a consistent variability pattern for its link quality. Intuitively, the cost for actions D_1 and D_2 should be a measure of the cost of a unicast probe. Since we want to give higher priorities to probing parents than other neighbours, we have that $C_1 \geq C_2$. For the same reason, we assume that skipping a probing phase provides a gain G_{np} , in terms of saved node and network resources (line 16). As a cost for the action D_3 we use the utility of the link with the preferred parent because if the link with the preferred parent is not stable a node should keep looking for alternative links.

Group management. As explained above link clustering is controlled using the path cost. It is important to point out that link quality variations, especially for neighbours with intermediate link qualities, may yield to frequent changes in the cluster composition. However, this can negatively affect the convergence of the learning algorithm. Therefore, we define a *hysteresis margin* for the sojourn time of a node in the set \mathcal{P}_i . Specifically, when a node j in the set \mathcal{P}_i is not anymore among the best m_p neighbours for node i it would be removed only if this condition persists for at least a time t_{hyst} . This check is implemented in the function UPDATECLUSTERS($\mathcal{P}_i, \mathcal{O}_i$) (line 21 of Algorithm 1).

Preferred parent monitoring. In RL-Probe the link quality to the preferred parent is estimated by passively monitoring the data traffic, as in legacy RPL. For this reason, the preferred parent is not part of the set \mathcal{P}_i . However, the link quality measurements are still used to update the utility estimates for the preferred parent (line 42 of Algorithm 1).

4. Performance Evaluation

In order to implement and evaluate RL-Probe, we opted for the Contiki 3.0 operating system (OS). The main reasons for selecting Contiki are: *i*) the support of Cooja simulator, which allows to easily port the software on real hardware, *ii*) the availability of a standard RPL implementation that is widely used, and *iii*) the availability of several plugins that already implement mobility models, interference models and probing techniques. Table 1 summarises the RL-Probe parameters used in both Algorithm 1 and Algorithm 2, which have been fine-tuned with extensive simulations.

In this section, we evaluate RL-Probe against standard implementations of legacy RPL, RPL-PP and mRPL using

³An exponential moving average (EMA) filter with smoothing factor 0.8 is used to update these estimates.

Table 1: RL-Probe Protocol Parameters.

Parameter	value
α	3%
β	1
m_p/m_o	3 / 10
C_1/C_2	1 / 5
G_{np}	10
ϵ	0.7
t_{hyst}	10 minutes
T_p	1 minute

both simulations and experiments. Specifically, we consider a basic RPL implementation that measures the quality of links through passive monitoring of the links used by data traffic [20]. Secondly, we consider RPL-PP, included in Contiki 3.0 RPL implementation, as a solution specifically designed to handle link quality variations. Finally, mRPL is also considered as a term of comparison, to verify the efficacy of RL-Probe in handling mobility. To this aim, we ported the mRPL implementation described in [8] and available for Contiki 2.6.1 to the latest version of the Contiki OS. It is important to point out that mRPL needs to differentiate between mobile nodes and fixed nodes (called Access Points) as mobile nodes are forced to act only as leaf nodes in the routing tree. For this reason, we compare mRPL with RL-Probe only in scenarios in which mobility is involved, or there are many unstable links.

Metrics. We consider three main performance metrics in our evaluation. First, we measure the packet loss ratio (PLR) at the sink, defined as the percentage of failed packet transmissions over the total number of packets sent by a node. Secondly, we consider the packet overhead measured as the sum of the RPL control messages, including probe packets. Thirdly, we measure the total energy consumed during the experiments.

4.1. Simulation Analysis

A simulation study is needed to investigate the performance in controllable and easily reproducible network conditions. Thus, we use Cooja to simulate Tmote Sky nodes. In WSNs, a radio duty cycling (RDC) mechanism is typically implemented to switch on and off the radio transceiver in order to save energy. ContikiMAC is the default RDC scheme used in our tests [35]. To model realistic radio propagation and interference we use the Multipath Ray-tracer Medium (MRM), which supports multi-path effects [36]. We have configured MRM parameters to achieve a 100% success rate at 10 meters and an interference range of 20 meters. Figure 1 shows the packet loss rate as a function of node distance for a link under the MRM model. The traffic flows generate a Constant Bit-Rate (CBR) traffic consisting of small UDP messages (40 bytes) sent every one minute from all the nodes to the sink. We simulate 24 hours of network operations and 95% confidence intervals are computed by replicating each simulation five times with different random seeds.

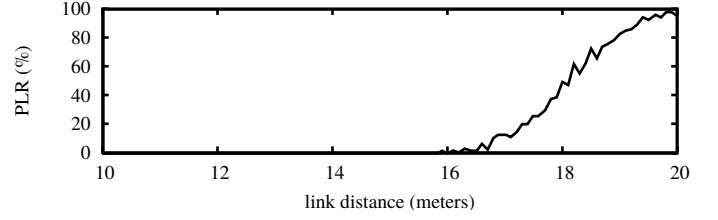


Figure 1: Link characterisation in Cooja simulator under the MRM model.

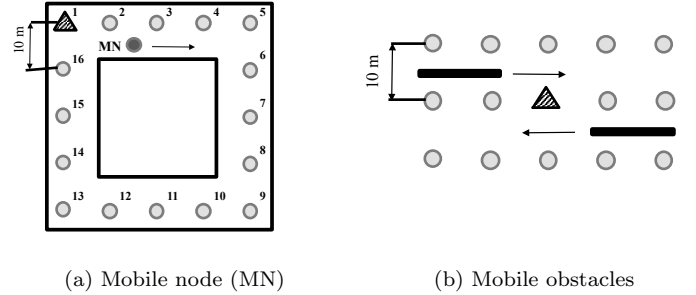


Figure 2: Simulation scenarios (the triangle is the root node).

To evaluate the proposed scheme, we consider two network scenarios. The first one is depicted in Figure 2a, and it exemplifies a corridor monitored with fixed and mobile nodes. Specifically, 16 sensor nodes are deployed following a square layout at a distance of 10 meters each. Then, a mobile node moves at a constant speed v from one corner to the following one, and every time it reaches the location of a fixed node it pauses for p minutes. The second scenario is depicted in Figure 2b, and it exemplifies sensor nodes deployed in a challenged industrial environment (e.g., an assembly line) with large moving obstacles that can impair wireless communications. Specifically, we have three parallel rows of 5 sensor nodes each with one large obstacle that moves from the left to the right corners and back, and another large obstacle that moves in the opposite direction. We assume that each obstacle moves to the next node in the row and remain fixed for a time p . Furthermore, each obstacle is able to completely filter out wireless transmissions between adjacent nodes. It is important to remind that in mRPL mobile nodes are forced to act only as leaf nodes. This implies that mobile nodes cannot forward traffic from other nodes. However, in a network in which there are many mobile nodes, or many nodes experiencing unstable links, it might be difficult to build an optimal network topology using mRPL. Clearly, assuming that there is only a single mobile node as in Figure 2a corresponds to consider a best case for mRPL. On the other hand, in the network scenario depicted in Figure 2b nodes are static but link qualities vary significantly. In this case, it is not straightforward to properly configure mRPL. To fairly compare mRPL with RL-Probe we firstly conducted an extensive set of simulations by varying the number of nodes that

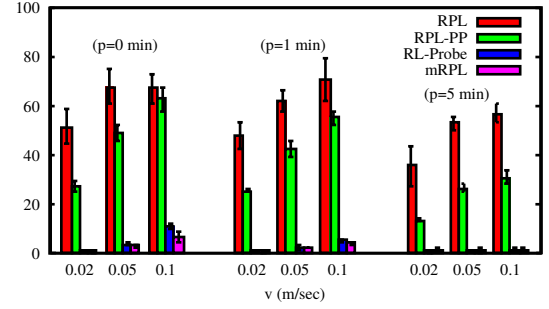
are configured as mobile in mRPL. Then, we selected the configuration that provided the best performance in terms of PLR. Finally, nodes that are configured as mobile in RL-Probe are forced to be leaf nodes also when RL-Probe is used to avoid that the higher flexibility of RL-Probe in topology construction might provide an advantage over mRPL.

4.1.1. Results with mobile nodes

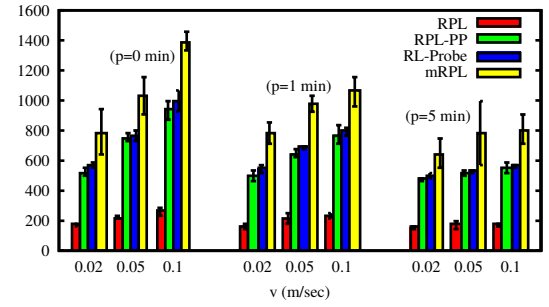
In this section, we report the results for the scenario illustrated in Figure 2a. First of all, Figure 3a shows the average packet loss rate of the mobile node for various speeds and pause times. Important conclusions can be drawn from these results. First, as expected packet loss rates increase when increasing speed and decrease when increasing pause times for all considered schemes. This is due to more frequent handoffs. Secondly, passive monitoring is unable to promptly cope with topology changes and packet loss rates range from 35% to 65% in the considered scenarios. Thirdly, unicast-based probing improves RPL ability to detect handoffs but packet loss rates still range from 18% to 55%. On the contrary, RL-Probe and mRPL have similar performance and they dramatically improve communication reliability, with packet loss rates that now range from 2% to 12%. An in-depth explanation of the root cause of such improvement is provided later in this section.

Figure 3b shows the protocol overhead in terms of the total number of RPL control messages sent during 24 hours. Clearly, there is a significant increase in packet overheads when active probing is used. As expected, the higher the speed and/or the shortest the pause time (i.e., the faster the network dynamics), the higher the protocol overheads. Interestingly, mRPL generates the highest protocol overheads among the considered routing schemes, while RL-Probe has similar overhead performance as RPL-PP. Analysing more in details the results we found out that the adaptive beaconing of RL-Probe reduces the number of unicast-based probing that are generated with respect to RPL-PP. However, these protocol overhead savings are compensated by the receiver-side probing, which generates trains of consecutive probes. On the other hand, handoff process in mRPL is quite aggressive as it generates long trains of multicast DIS messages to neighbouring nodes.

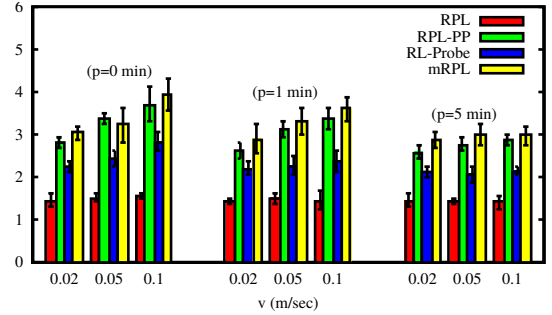
It may be argued that an increase in protocol overheads would severely affect the node energy consumption. To verify this conjecture, Figure 3c shows the total energy consumption as estimated by Cooja. Clearly, RPL-PP, mRPL and RL-Probe consume more energy than basic RPL without probing. However, RPL-PP and mRPL have similar energy consumption while RL-Probe consumes up to 30% less energy than the other protocols, although RL-Probe and RPL-PP have similar protocol overheads. This counterintuitive result can be explained by observing that retransmissions have a great impact on the energy consumption. Thus, avoiding the use of lossy links can



(a) PLR



(b) Packet overhead



(c) Total energy consumption (Wh)

Figure 3: Simulation analysis of MN's performance for different speed values and pause times.

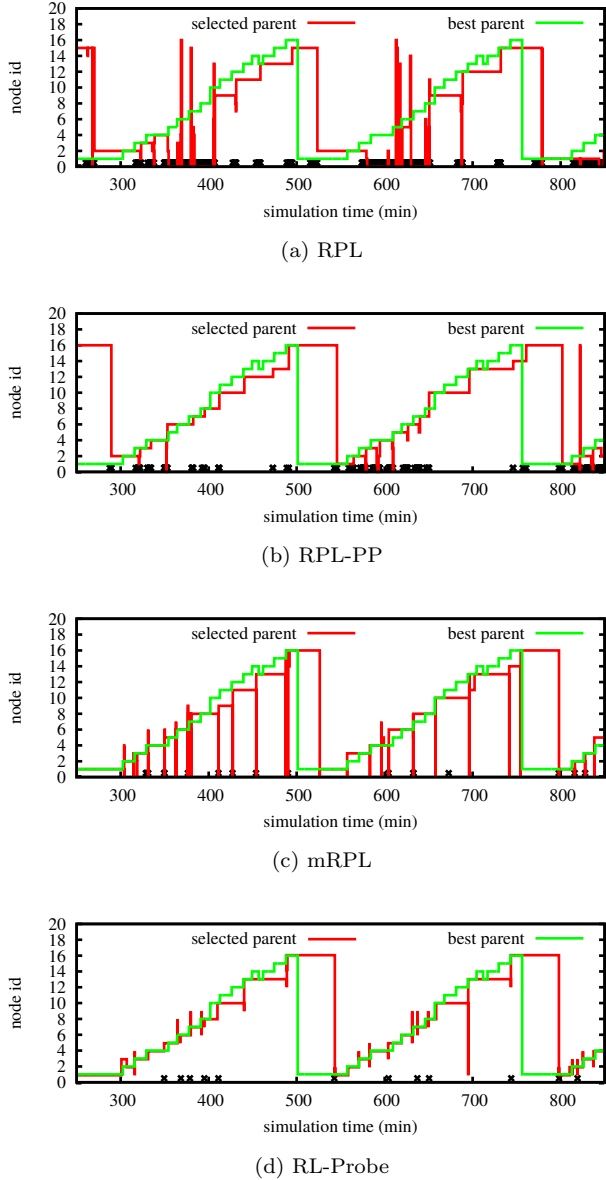


Figure 4: Handoff events and packet losses (crosses) for MN when $p = 0.01$ m/sec and $p = 5$ minutes.

balance the additional energy consumption due to active probing.

To explain more in details the key advantages of RL-Probe, Figures 4 show the sequence of handoff events and packet losses for the case $v = 0.02$ m/sec and $p = 5$ minutes (the best case for both RPL and mRPL). The results indicate that standard RPL with passive link monitoring frequently changes the preferred parent, and it rarely selects the optimal one. On the contrary, RPL-PP is able to follow more closely the best parent. However, handoffs are mainly triggered by packet losses and round-robin periodic probing in RPL-PP causes a burst of packet losses before discovering the new optimal parent. A similar issue is also observed in mRPL, which triggers the discovery phase after a packet loss. Furthermore, the handoff delays in mRPL also cause short disconnections of the mobile node (pre-

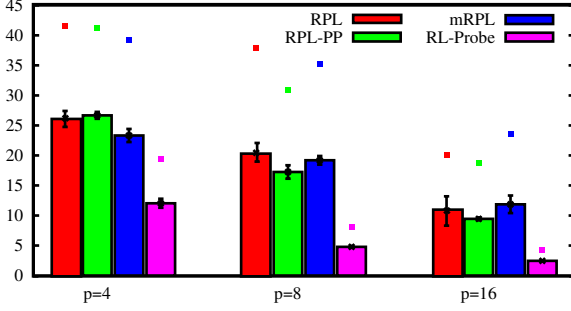
ferred parent id equal to 0). On the contrary, *the analysis of link trends allows RL-Probe to anticipate changes of link characteristics* and to timely switch to a better preferred parent. Finally, it is interesting to note that a higher number of packet losses occurs when the mobile node is close to the sink. This can be explained by observing that an inaccurate ETX estimation of the link to the neighbours has a greater effect on the rank computation of nodes close to the sink than on nodes far from the sink.

4.1.2. Results with mobile obstacles

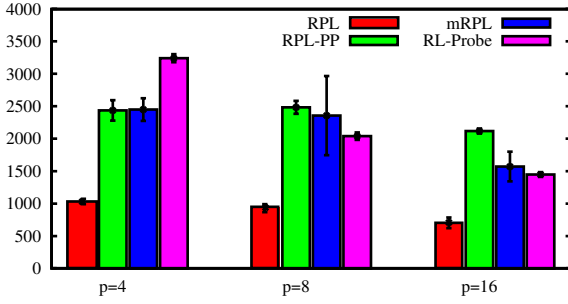
In this section, we report the results for the scenario illustrated in Figure 2b where network topology changes are due to variations of link conditions caused by mobile obstacles and not handoffs. Figure 3a shows the average (bars) and maximum (squares) PLR of *all* nodes for different pause times ($p = 4, 8, 16$ minutes). Results indicate that RL-Probe achieves a three-fold decrease of both average and peak PLRs with respect to the other considered schemes, including mRPL. On the contrary, mRPL performs similarly to RPL and RPL-PP. Several factors contribute to mRPL inefficient behaviour. First, the hysteresis margin in mRPL assumes that the transitional region of links is quite wide [31]. However, this decreases the ability of mRPL to detect sudden changes of link qualities that occur within the transitional region. Furthermore, if the quality of the link to the preferred parent is stable mRPL does not trigger discovery phases, which are needed to quickly detect if the quality of the links to neighbouring nodes is suddenly improved (e.g., because an obstacle has moved). Figure 5b shows the protocol overhead in terms of RPL control messages. We can observe that RL-Probe rapidly limits protocol overheads as the network conditions become less variable (i.e., pause times increase). Thus, RL-Probe generate higher protocol overheads than mRPL for $p = 4$ minutes, but the overheads are similar for $p = 16$ minutes. The same trend can be observed also for the total energy consumption (see Figure 5c). Summarising, in case of link quality variability due to changes in network conditions RL-Probe outperforms mRPL in terms of communication reliability with similar network overhead and energy waste.

4.2. Experimental Evaluation

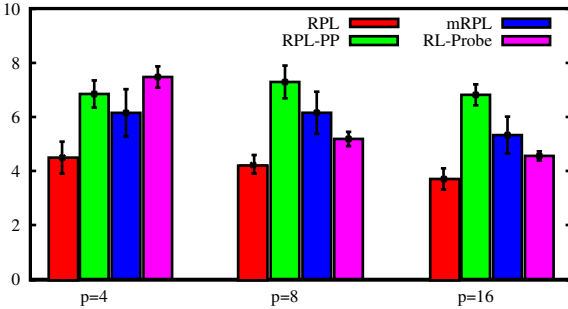
In this section, we report the results obtained from real experiments conducted in an indoor IoT testbed [37]. Specifically, our low-power wireless network is composed of 23 wireless sensor nodes deployed in office spaces, student labs, and corridors on two floors in the Department of Information Engineering of the University of Pisa. Figure 6 shows the layout of the testbed. Sensor nodes are TelosB motes, equipped with an MSP430 micro-controller that can run a wide range of Operating Systems for sensors. Thus, the same Contiki code used for Cooja simulations is loaded on the testbed. IEEE802.15.4 connectivity is provided through the cc2420 wireless chip equipped with an external



(a) PLR (average and maximum)



(b) Packet overhead



(c) Total energy consumption (Wh)

Figure 5: Simulation analysis with mobile obstacles and different pause times.



Figure 6: Map of the testbed

Table 2: p -th percentiles of the average ETX of the wireless links in the testbed.

	p value				
	10%	25%	50%	75%	90%
Average ETX	1.0	1.0	1.025	1.245	2.776

5dBi antenna. The maximum number of active links in the network is 178. Table 2 reports the main percentiles of the average ETX across all links, as measured by unicast-based probing without any data or control traffic in the network.

The first set of results is obtained considering a static scenario in which there are not mobile nodes. However, we emphasise that our testbed is deployed in a dynamic environment and the experiments have not been run at special times to avoid interference. Thus, our testbed is susceptible to changes in radio channel conditions due to interference (e.g., from other 802.15.4 radios and from 802.11 radios), and this interference is highly time-varying. In this scenario all nodes are configured to generate a CBR traffic consisting of 40-bytes UDP messages sent every minute to node 1. The underlying MAC protocol is CSMA and ContikiMAC is used as RDC layer. The radio transmission power is set to the maximum value (0 dBm) to maximise the network density and the number of available links. Figure 7 shows the average packet loss ratio and energy consumption measured for RPL, RPL-PP and RL-Probe during experiments that last three hours. We omit mRPL as we already know that it is not optimised for static scenarios. The experimental results confirm the trends observed in simulations. Specifically, RL-Probe and RPL-PP help reducing the PLR compared to legacy RPL because they provide faster routing adaptation to channel fluctuations. Such improvement is paid with an additional energy consumption overhead, caused by the active probing traffic. Note that the absolute values of PLRs are low. This is explained by observing that there is a careful planning of the sensor locations and, the use of the maximum transmission power ensures the existence of several good links.

The second set of results is obtained in a scenario in which there is a mobile sensor node. Figure 8 illustrates the trajectory of this mobile node. Specifically, after an initial set-up phase of 5 minutes, the node moves at 0.5 m/s from one specified point to another, pausing at each location for 2 minutes. The path is covered round-trip, i.e.

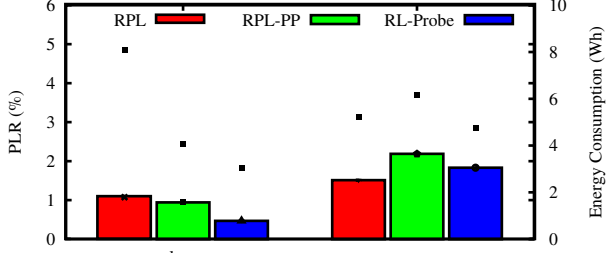


Figure 7: Experimental analysis, average packet loss and energy consumption in the static scenario.

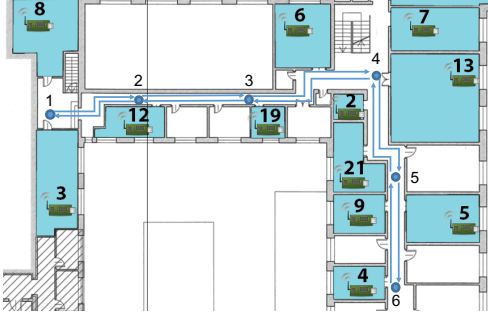


Figure 8: Trajectory followed by the mobile node during the mobility experiments.

from point 1 to 6 and then back from 6 to 1. Traffic is only originated by the mobile node towards node number 1, which is selected as sink and RPL root node. A CBR traffic is employed i.e. a 40-byte UDP message is emitted every 10 seconds. In this scenario, RDC is disabled, i.e. each node keeps its wireless transceiver on all the time, in order to avoid any possible influence of duty cycling on handoff procedures. Furthermore, the radio transmission power is lowered to -7 dBm to make the network topology sparser so as to reduce the number of neighbours of the mobile node. This guarantees that each movement of the mobile node results into a change of the parent node. Each experiment lasts 30 minutes.

Figure 9 shows the average packet loss experienced by the mobile node with different strategies. We can observe that RL-Probe slightly outperforms mRPL, which confirms the effectiveness of multicast probing in obtaining a rapid assessment of the link quality when multiple neighbours appear/disappear at the same time. On the other hand, standard RPL experiences many packet losses every time the mobile node changes its location, due to the lack of an active strategy for LQE. RPL-PP achieves better performance than basic RPL but it is less efficient than both mRPL and RL-Probe. This can be explained by considering that unicast probing assesses links individually and therefore more time is needed to discover a better preferred parent when moving.

Figure 10, instead, quantifies the overhead produced by each strategy. Specifically, the figure reports the average number of RPL control packets (including both probe and response packets) per second generated by the nodes in the network. We distinguish between the overhead generated

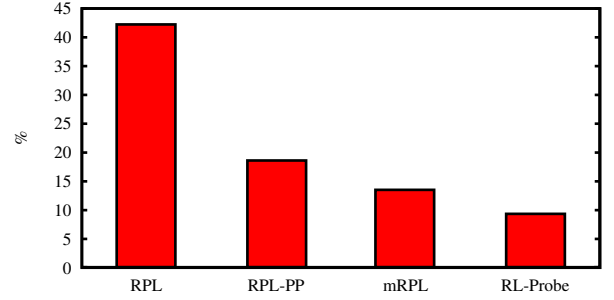


Figure 9: Experimental analysis, average packet loss in the mobile scenario.

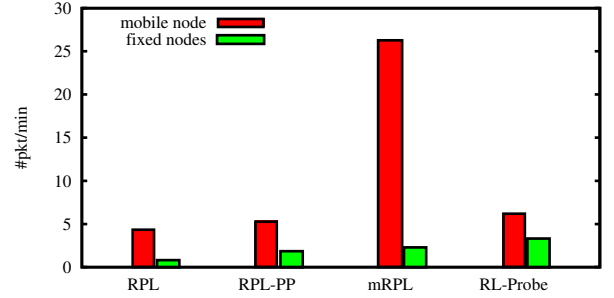


Figure 10: Experimental analysis, control message overhead in the mobile scenario.

by static nodes and the overhead generated by the mobile node. As expected, legacy RPL is the strategy characterised by the least overhead, as nodes only transmit RPL control packets for topology discovery without any probe packet. RPL-PP, instead, shows a slight increase in the overhead as a light unicast probe traffic is employed. Both mRPL and RL-Probe are characterised by the highest overhead due to the active probe traffic generated by each node. However, the overhead generated by the mobile node using mRPL is four times the overhead provided by the same mobile node when using RL-Probe. This clearly shows that the responsiveness of mRPL to node mobility is obtained at the cost of introducing frequent probing. On the contrary, RL-Probe does not penalise the mobile node, who requires a minimal additional overhead with respect to fixed nodes to detect link failures.

5. Conclusion

In this paper, we have proposed RL-Probe, link quality estimation strategy for RPL-based WSNs. RL-Probe employs synchronous and asynchronous monitoring schemes to maintain up-to-date information on link quality towards the neighbours and react to sudden topology changes. RL-Probe achieves a trade-off between low probing overheads and responsiveness to changing network conditions by leveraging on a lightweight reinforcement learning technique to control the active probing operations. This is crucial to minimised energy consumptions of tiny, resource-constrained

devices. Furthermore, we have integrated our solution in the RPL implementation that is included in the Contiki operating system for embedded devices. A performance evaluation based on both simulations and real-world experiments has been carried out, demonstrating how the proposed approach guarantees better performance with respect to state-of-the-art LQE techniques for RPL. In particular, results show that the proposed approach does not only properly react to link quality variations, but it is also effective to handle topology variations due to mobility.

References

- [1] J. A. Stankovic, Research directions for the internet of things, *IEEE Internet of Things Journal* 1 (1) (2014) 3–9. [doi:10.1109/JIOT.2014.2312291](https://doi.org/10.1109/JIOT.2014.2312291)
- [2] M. Zhao, I. Ho, P. H. J. Chong, An energy-efficient region-based rpl routing protocol for low-power and lossy networks, *IEEE Internet of Things Journal* 3 (6) (2016) 11319–11333.
- [3] T. Winter, P. Thubert, A. Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, *IETF RFC 6550* (March 2012).
- [4] E. Ancillotti, R. Bruno, M. Conti, RPL Routing Protocol in Advanced Metering Infrastructures: an Analysis of the Unreliability Problems, in: *Proc. of IFIP SUSTAINIT'12*, 2012, pp. 1–10.
- [5] I. E. Korbi, M. B. Brahim, C. Adjih, L. A. Saidane, Mobility Enhanced RPL for Wireless Sensor Networks, in: *Proc. of IEEE NOF'12*, 2012, pp. 1–8.
- [6] C. Cobârzan, J. Montavont, T. Noël, Analysis and performance evaluation of rpl under mobility, in: *Proc. of IEEE ISCC'14*, 2014, pp. 1–6.
- [7] E. Ancillotti, R. Bruno, M. Conti, Reliable Data Delivery With the IETF Routing Protocol for Low-Power and Lossy Networks, *IEEE Transactions on Industrial Informatics* 10 (3) (2014) 1864–1877. [doi:10.1109/TII.2014.2332117](https://doi.org/10.1109/TII.2014.2332117)
- [8] H. Fotouhi, D. Moreira, M. Alves, mRPL: Boosting mobility in the Internet of Things, *Ad Hoc Networks* 26 (2015) 17–35. [doi:http://dx.doi.org/10.1016/j.adhoc.2014.10.009](http://dx.doi.org/10.1016/j.adhoc.2014.10.009)
- [9] J. Ko, M. Chang, MoMoRo: Providing Mobility Support for Low-Power Wireless Applications, *IEEE Systems Journal* 9 (2) (2015) 585–594.
- [10] A. Oliveira, T. Vazão, Low-power and lossy networks under mobility: A survey, *Computer Networks* 107 (2016) 339–352.
- [11] F. Gara, L. B. Saad, E. B. Hamida, B. Tourancheau, R. B. Ayed, An adaptive timer for RPL to handle mobility in wireless sensor networks, in: *Proc. of IEEE IWCMC'16*, 2016, pp. 678–683.
- [12] L. Tang, K. C. Wang, Y. Huang, F. Gu, Channel Characterization and Link Quality Assessment of IEEE 802.15.4-Compliant Radio for Factory Environments, *IEEE Transactions on Industrial Informatics* 3 (2) (2007) 99–110. [doi:10.1109/TII.2007.898414](https://doi.org/10.1109/TII.2007.898414)
- [13] N. Baccour, A. Koubâa, L. Mottola, M. A. Zúñiga, H. Youssef, C. A. Boano, M. Alves, Radio Link Quality Estimation in Wireless Sensor Networks: A Survey, *ACM Transactions on Sensor Networks* 8 (4) (2012) 34:1–34:33. [doi:10.1145/2240116.2240123](https://doi.org/10.1145/2240116.2240123)
- [14] J. H. O. G. P. Levis, T. Clausen, J. Ko, The Trickle Algorithm, *IETF RFC 6206* (March 2011).
- [15] K.-H. Kim, K. G. Shin, On Accurate and Asymmetry-aware Measurement of Link Quality in Wireless Mesh Networks, *IEEE/ACM Transactions on Networking* 17 (4) (2009) 1172–1185. [doi:10.1109/TNET.2008.2008001](https://doi.org/10.1109/TNET.2008.2008001)
- [16] N. Cesa-Bianchi, G. Lugosi, *Prediction, Learning, and Games*, Cambridge University Press, 2006.
- [17] D. S. J. De Couto, D. Aguayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless routing, in: *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, MobiCom '03*, ACM, New York, NY, USA, 2003, pp. 134–146. [doi:10.1145/938985.939000](https://doi.org/10.1145/938985.939000) URL <http://doi.acm.org/10.1145/938985.939000>
- [18] P. Levis, N. Patel, D. Culler, S. Shenker, Trickle: A Self-regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks, in: *Proc. of USENIX NSDI'04*, 2004.
- [19] T. Narten, E. Nordmark, W. Simpson, S. H., Neighbor Discovery for IP version 6 (IPv6), *IETF RFC 4861* (September 2007).
- [20] S. Dawans, S. Duquennoy, O. Bonaventure, On Link Estimation in Dense RPL Deployments, in: *Proc. of IEEE SenseApp'12*, 2012.
- [21] H. Zhang, L. Sang, A. Arora, Comparison of Data-Driven Link Estimation Methods in Low-Power Wireless Networks, *IEEE Transactions on Mobile Computing* 9 (11) (2010) 1634–1648.
- [22] H. Zhang, A. Arora, P. Sinha, Link Estimation and Routing in Sensor Network Backbones: Beacon-Based or Data-Driven?, *IEEE Transactions on Mobile Computing* 8 (5) (2009) 653–667.
- [23] O. Gaddour, A. Koubâa, R. Rangarajan, O. Cheikhrouhou, E. Tovar, A. Abid, Co-RPL: RPL routing for mobile low power wireless sensor networks using corona mechanism, in: *Proc. of IEEE SIES'14*, 2014, pp. 200–209.
- [24] K. Srinivasan, P. Dutta, A. Tavakoli, P. Levis, An Empirical Study of Low-power Wireless, *ACM Transactions on Sensor Networks* 6 (2) (2010) 16:1–16:49. [doi:10.1145/1689239.1689246](https://doi.org/10.1145/1689239.1689246)
- [25] N. Baccour, A. Koubâa, H. Youssef, M. Ben Jamâa, D. do Rosário, M. Alves, L. B. Becker, F-LQE: A Fuzzy Link Quality Estimator for Wireless Sensor Networks, in: *Proc. of EWSN'10*, 2010, pp. 240–255.
- [26] Y. Wang, M. Martonosi, L.-S. Peh, Predicting Link Quality Using Supervised Learning in Wireless Sensor Networks, *SIGMOBILE Mob. Comput. Commun. Rev.* 11 (3) (2007) 71–83. [doi:10.1145/1317425.1317434](https://doi.org/10.1145/1317425.1317434) URL <http://doi.acm.org/10.1145/1317425.1317434>
- [27] T. Liu, A. E. Cerpa, Temporal Adaptive Link Quality Prediction with Online Learning, *ACM Transactions on Sensor Networks* 10 (3) (2014) 46:1–46:41.
- [28] T. Liu, A. E. Cerpa, Data-driven Link Quality Prediction Using Link Features, *ACM Transactions on Sensor Networks* 10 (2) (2014) 37:1–37:35.
- [29] N. Accettura, L. A. Grieco, G. Boggia, P. Camarda, Performance analysis of the rpl routing protocol, in: *Proc. of IEEE ICM'11*, 2011, pp. 767–772.
- [30] M. Barcelo, A. Correa, J. L. Vicario, A. Morell, X. Vilajosana, Addressing Mobility in RPL With Position Assisted Metrics, *IEEE Sensors Journal* 16 (7) (2016) 2151–2161.
- [31] H. Fotouhi, M. Alves, M. Z. Zamalloa, A. Koubâa, Reliable and Fast Hand-Offs in Low-Power Wireless Networks, *IEEE Transactions on Mobile Computing* 13 (11) (2014) 2620–2633.
- [32] S. Wang, A. Basalamah, S. M. Kim, G. Tan, Y. Liu, T. He, A Unified Metric for Correlated Diversity in Wireless Networks, *IEEE Transactions on Wireless Communications* PP (99). [doi:10.1109/TWC.2016.2581821](https://doi.org/10.1109/TWC.2016.2581821)
- [33] E. Ancillotti, R. Bruno, M. Conti, E. Mingozzi, C. Vallati, Trickle-l2: Lightweight link quality estimation through trickle in rpl networks, in: *World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2014 IEEE 15th International Symposium on a, 2014, pp. 1–9. [doi:10.1109/WoWMoM.2014.6918951](https://doi.org/10.1109/WoWMoM.2014.6918951)
- [34] C. E. Koksal, H. Balakrishnan, Quality-Aware Routing Metrics for Time-Varying Wireless Mesh Networks, *IEEE Journal on Selected Areas in Communications* 24 (11) (2006) 1984–1994. [doi:10.1109/JSAC.2006.881637](https://doi.org/10.1109/JSAC.2006.881637)
- [35] A. Dunkels, The ContikiMAC Radio Duty Cycling Protocol, *Tech. Rep. T2011:13*, SICS (December 2011).
- [36] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, T. Voigt, Cross-Level Sensor Network Simulation with COOJA, in: *Proc. of IEEE LCN'06*, 2006, pp. 641–648.
- [37] C. Vallati, E. Ancillotti, R. Bruno, E. Mingozzi, G. Anastasi, Interplay of link quality estimation and rpl performance: An experimental study, in: *Proceedings of the 13th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, &*

Ubiquitous Networks, PE-WASUN '16, ACM, New York, NY,
USA, 2016, pp. 83–90. doi:10.1145/2989293.2989299
URL <http://doi.acm.org/10.1145/2989293.2989299>