Received: 5 April 2019 | Revised: 14 August 2019 | Accepted: 14 October 2019
DOI: 10.1002/ett.3806

RESEARCH ARTICLE

WILEY

TAMU-RPL: Thompson sampling-based multichannel RPL

Pedro Henrique Gomes | **Bhaskar Krishnamachari**

Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles, California

Correspondence
Pedro Henrique Gomes, Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, EEB 300, 3740 McClintock Avenue, Los Angeles, CA 90089.
Email: pdasilva@usc.edu

Funding information
National Science Foundation (NSF), Grant/Award Number: AST-1248017 and CCF-1423624

Abstract

For the success of critical applications in the IoT, there is a need to counteract the effects of external interference, especially when the unlicensed spectrum is used. One way of improving the performance is with a routing protocol that quickly reacts to changes in the environment and avoids path and/or frequencies with higher interference. In this paper, we propose an optimization to the RPL protocol, called TAMU-RPL, that can keep a more accurate estimation of link quality to the neighbors and quickly react to degradation on the links. TAMU-RPL also uses the quality of links at different frequencies and opportunistically avoids the ones with bad quality. It is evaluated through simulations with connectivity traces from a 40-node test bed and in a real 5-node deployment. We compare TAMU-RPL with a baseline RPL protocol and a Dijkstra-based shortest-path algorithm. Results show that TAMU-RPL can successfully explore the neighbors and obtain ETX values much closer to the ones obtained with a shortest-path tree, even when link quality changes over time. In the simulation evaluation, TAMU-RPL was able to double the number of packets received at the sink compared to RPL and reduced the average delay of packets (in time slots) by more than 10%. In the real deployment, the number of packets received at the sink was increased by more than 33%.

1 | INTRODUCTION AND MOTIVATION

The main objective of a routing protocol in multihop wireless networks is to create an optimal path that connects the source and destination nodes and minimizes the use of resources. It plays a special role in low-power and lossy networks (LLNs), as they are mainly composed of unreliable links that suffer from external interference and have wide fluctuations in quality over time. This kind of a dynamic scenario is challenging for sensor nodes since collecting topology information requires communication bandwidth and processing power, which must be reduced as much as possible. The *Routing Protocol for LLNs* (RPL) is the protocol standardized by IETF for the routing layer of LLNs. Even though RPL was designed for dynamic networks, several studies have shown that it underperforms in dynamic and mobile environments.[1] The main reason is the slow responsiveness to variations in the link qualities.

RPL has to constantly calculate the path cost to build the best forwarding trees; the cost used by RPL implementations is based on the estimated transmission count (ETX) of links.[2] Estimating the ETX is a challenging task because it usually requires the exchange of unicast packets. If a node needs to quickly react to changes in the link quality, it has to keep an ETX estimation not only of its current parent but also of other neighbors to select a better path proactively. Hence, there is a trade-off between the exploitation of the currently selected path with a known minimum cost, and the exploration of alternative paths to ensure an accurate estimation of their costs and also to respond to network changes. Moreover, RPL was not designed to work on multichannel networks.[1] Usually, when RPL is used in multichannel networks, it ends up operating with aggregated statistical data from all the channels, which may mislead the routing decisions. An ideal RPL implementation for multichannel networks should optimize the paths by taking into account the frequencies used at each hop.

Another challenge in RPL implementations is how to leverage broadcast packets to improve the link quality estimation. Usually, RPL only estimates the ETX using unicast packets, when the data is followed by ACK packets, and it is easy to measure the packet losses. Broadcast packets are not followed by ACK packets and, hence, losses can only be predicted by other means, eg, based on the interval of broadcast transmissions. In some scenarios, where (unicast) data packets are not often transmitted, broadcast packets are one way of obtaining a better ETX estimation of many links. Besides, each broadcast packet can be received by all the nodes that are listening to the packets, which increases the number of nodes that can keep track of the neighbors' statistical link qualities and may cause the RPL algorithms to react more quickly to network changes or mobility.

In this paper, we introduce the Thompson sampling–based multichannel RPL (TAMU-RPL) protocol. In TAMU-RPL, the selection of the preferred parent in the RPL protocol is modeled as a multiarmed bandit (MAB) problem, and the use of the Thompson sampling heuristic is investigated to improve the reactiveness of RPL. TAMU-RPL has a dynamic link quality estimation algorithm that keeps track of the link quality of a larger number of neighbors and can make RPL react more quickly to link quality changes. It regards unicast packet transmissions as the main source for ETX estimation but also leverages the broadcast nature of wireless communication and uses information from other packets that are overheard. Finally, an improved version of the *DAGrank* calculation algorithm is designed, which carries out the ETX estimation for individual channels and, hence, can perform better in multichannel networks.

This work considers the use of time-slotted channel hopping (TSCH) protocol for an enhanced LLN solution. TSCH slices time into slots, each with sufficient duration (typically 10 ms) to accommodate a data packet of maximum size and an acknowledgement (ACK) packet, as well as all the required guard times. It employs time division multiple access together with multiple channels for communication. More than one transmission can be scheduled to occur at the same time in a TSCH network through a well-designed scheduling algorithm. As long as simultaneous transmissions are made through different frequencies or between noninterfering nodes, a collision-free operation can be guaranteed.

The main contributions of this paper are as follows:

1. An optimized RPL based on the MAB problem that improves the agility to react to link quality changes. The main algorithm is designed to carry out a new link quality estimation based on Thompson sampling that keeps track of the ETX of multiple neighbors.
2. A modified RPL objective function that makes use of statistics from different channels for the *DAGrank* calculation of neighbors and leverages RSSI statistics from broadcast packets to improve ETX estimation.

This paper is organized as follows. Section 2 lists the related work. Section 3 briefly introduces RPL and shows how most implementations execute the preferred parent selection. Section 4 introduces TAMU-RPL, the protocol proposed in this paper. Section 5 shows the evaluation of the proposal through simulations. Section 6 shows the evaluation through experiments in a small real network. Section 7 summarizes the lessons learned from the experiments. Finally, Section 8 concludes this paper.

2 | RELATED WORK

The MAB problem has been widely studied and applied to practical problems in the realm of wireless communication, such as opportunistic spectrum access[3] and reconfigurable antennas.[4] Gai et al[5] introduced efficient policies for the problem of selecting multiple random variables (each one associated with an arm) and noting the reward obtained by each random variable. In the context of routing, this framework corresponds to the execution of source routing, where a source node selects a predetermined path at each time. Liu and Zhao[6] examined the same source routing scenario, but only the cumulative end-to-end cost is observed. Since paths share links, the arms that are played (the chosen paths) are not independent, and the policy takes advantage of that to improve the complexity of the solution. Both works[5,6] considered the question of source routing and required each node to have the global knowledge of all possible paths. In LLNs, source routing is restricted to downward traffic, since only the sink node has such global knowledge (usually not completely up to date) and have capabilities for heavy routing calculations.

The work by Zou et al[7] is the first, to the best of our knowledge, to tackle the problem of optimizing hop-by-hop decisions. Zou et al[7] showed that the regret lower bound for hop-by-hop policies is the same as the source routing, which means that there is no advantage in using source routing, even if the reward is observed for each link. The hop-by-hop decision-making algorithm employed is based on Kullback-Leibler upper confidence bound (KL-UCB). When a packet needs to be forwarded, an index is calculated for each link using the Kullback-Leibler divergence number between two Bernoulli distributions. The minimum cumulative index from the sensor node to the sink must also be calculated, for instance by using the Bellman-Ford algorithm, as suggested by Zou et al.[7] The packet is forwarded to the link that minimizes the summation of the cumulative indexes. Although the forwarding decision is executed at each hop, all the nodes still need to partially know the topology so that they can calculate the shortest path from them to the sink. Besides, the heavy computation of all proposals makes them impractical for implementing them in LLNs. Moreover, none of the works can be suitably integrated into generic routing protocols or have been evaluated in real scenarios, where there are constant topology changes and the rewards cannot be observed right after each action.

The RPL protocol has been extensively analyzed and tested in several related works,[1,8] many of them concerned with RPL's capabilities for convergecast routing and adaptability to interference-prone and mobile scenarios.[9]

A large number of new objective functions have been proposed for improving RPL performance in different scenarios where default ETX-based metrics are combined with more specialized ones that allow the routing tree to be optimized for particular applications. One particularly interesting problem that is addressed by Djedjig et al[10] and Karkazis et al[11] is the trustworthiness of the forwarding nodes. In addition to the link quality, the routing metric should also account the probability of (intentional or nonintentional) packet drops from malicious or selfish nodes. Other objective functions aim to extend the lifetime of the network with routes that optimally distribute the traffic among the forwarding nodes and avoid the energy bottlenecks.[12,13] Regardless of the specialization of the routing objective function, one metric that is always present is the ETX of links. This metric is the basis for any RPL routing rule, and it requires precise estimation of the link quality to a set of neighbors of each node. TAMU-RPL focuses on the use of ETX as a basis for the routing metric, but it can be combined with more complex metrics that account other factors, such as trustworthiness or energy balance.

Ancillotti et al[14] improved the RPL link quality estimation algorithm by using a hybrid link-monitoring framework. It adaptively selects one of three schemes for the measurement of link quality: (1) a regular passive estimation based on unicast data and ACK packets, (2) an overhearing mechanism where each of the nodes listens to packets transmitted to other nodes and counts the number of retransmissions to estimate the packet loss, and (3) active probing using bursts of ten DIS packets sent to a specific target for faster and more precise estimation. These three mechanisms are combined by a controller that is based on the current state of the RPL state machine and events from the neighbor's management module determine the best way to measure the link quality of neighbors. These solutions completely rely on events such as new destination-oriented directed acyclic graph (DODAG) information object (DIO) messages being heard from nodes that are joining the network and DIS packets being sent on-demand to specific nodes. It does not proactively monitor the links to all the neighbors. Besides, it is designed for single-channel networks and does not carry out link estimation across different channels.

The combination of different methods to estimate the link quality more effectively has been investigated in different wireless technologies by several studies. Kim and Shin[15] proposed an algorithm to switch between passive, active and cooperative link quality estimators in IEEE 802.11 networks. Passive estimation involves regular unicast data packets for ETX calculation, while active estimation requires probing for estimating idle links. The cooperative scheme can leverage the overhearing characteristic of wireless links. Node A exchanges data to node B and asks node C to overhear the traffic and reports back the link estimation between A and C. The cooperative scheme introduced by Kim and Shin[15] necessarily requires coordination and may impose extra overhead to the routing protocol.

Hermeto et al[16] argued that good routes can be found simply by ranking the neighbors according to their rate of broadcast packets received. They empirically determine that the packet delivery ratio (PDR) to neighbors is correlated with the rate of broadcast packets received, such as enhanced beacons and DIO packets. Hence, an accurate link quality estimation to each neighbor is not necessary, since a simple ranking that accounts for the broadcast packets received is sufficient. The authors examined an RPL implementation with Trickle disabled so that the control packets could be broadcast within a fixed period. They also disregarded the external interference that could affect both the unicast and broadcast packets. The scheme was implemented on OpenWSN.

Finally, Ancillotti et al[9] took the first steps towards using a reinforcement learning–based scheme to derive an efficient link quality estimation algorithm that leverages both synchronous and asynchronous measurements. The proposal, called RL-Probe, carries out asynchronous probing to all the neighbors as soon as certain patterns in the RSSI and ETX measurements are detected. These patterns indicate that the network was disrupted and/or significant events occurred in the network topology and may need an agile reaction. A MAB-based algorithm was created for the synchronous probing, where the neighbor nodes are split into three groups based on their path cost toward the sink node. Each group of neighbors to be probed is an arm in the MAB problem, and the "exploration versus exploitation" problem is solved employing a simple ϵ-greedy algorithm. Within each group of neighbors, the node that will be probed is also chosen with the aid of an ϵ-greedy algorithm. The ϵ value chosen in the evaluation was 3%. The experiments by Ancillotti et al[9] did not provide a comprehensive evaluation of the best ϵ value. Moreover, the patterns that are used to start asynchronous probing are statically set for all nodes, which may not be the optimal solution. As pointed out by Ancillotti et al,[9] the solution is not tailored to multichannel networks based on RPL, and this is still an open research question.

It is clear from the state-of-the-art that there are still three open problems:

1. How to implement a dynamic online learning–based link quality estimator that is simple to run in a constrained device and can improve the RPL protocol to explore links to neighbors while exploiting the current best parent selection.
2. How to improve ETX estimation through traffic overhearing that does not require any kind of coordination and leverage information systems such as RSSI and previous PDR measurements.
3. How to use statistics from individual channels in multichannel networks and improve the estimation of next-hop link quality with the knowledge of aggregated and channel-specific statistics.

The goal of our TAMU-RPL proposal is to solve these three open problems.

3 | ROUTING PROTOCOL FOR LOW-POWER AND LOSSY NETWORKS (RPL)

The RPL is a distance-vector routing protocol designed for 6LoWPAN networks; it creates *destination-oriented directed acyclic graphs* (DODAGs) towards the sink node.[17] RPL separates the packet processing and forwarding from the routing optimization via its *objective function* (OF). An OF describes how nodes should convert one or more metrics and/or constraints into a *DAGrank*, which represents the node's distance to the DODAG root. The *DAGrank* strictly increases in the downward direction. It is used by the RPL to detect and avoid loops and allows the nodes to distinguish between their candidate parents and children or sibling nodes.

The DODAG formation starts at the sink, which generates periodic *DIO packets*. The nonsink nodes listen to the DIOs and join the DODAG. Every node that joins the routing tree starts to periodically broadcast the DIOs so that the graph can be extended towards the leaf nodes. Each DIO packet contains the node's most recent *DAGrank* and the nodes store the *DAGrank* from all the neighbors. During the topology construction, a subset of stable nodes is chosen from the list of neighbors, and a preferred parent is selected based on the OF. Different OFs can be designed to comply with specific optimization criteria and satisfy the different requirements of the applications, such as minimum energy consumption or minimum end-to-end latency.

RFC[18] defines the *minimum rank with hysteresis objective function* (MRHOF), which has been adopted as the default implementation of RPL in most LLNs. MRHOF uses a hysteresis mechanism to prevent unnecessary parent switches being caused by small metric changes. It supports different types of metrics, such as hop count, latency, etc. The most commonly used metric, however, is the ETX. In summary, MRHOF is a greedy objective function that minimizes the end-to-end ETX from the nodes to the sink.

One issue of RPL is the slow convergence and route fix in face of lossy links, which makes it unsuitable for critical networks. Since nodes only exchange data with the neighbor with the lowest *DAGrank*, they do not have much knowledge of the link quality of other neighbors and may take a long time to find a better route when interference levels change.

3.1 | RPL's *DAGrank* calculation and preferred parent selection

In RPL with MRHOF the *DAGrank* of a given node $R(N)$ is calculated as the *DAGrank* of its preferred parent $R(P)$ plus a rank increase, as in Equation (1).

$$R(N) = R(P) + R_{\text{inc}}, \tag{1}$$

where R_{inc} is the rank increase, as in Equation (2).

$$R_{\text{inc}} = (Rf \times Sp + Sr) \times m_{\text{inc}}, \tag{2}$$

where m_{inc} is the minimum rank increase, which by default has a value of 256; Rf is the rank factor; Sp is the rank step; and Sr is the rank stretch. Rf is a factor that is usually multiplied by a link property, Sp is a link property of a given neighbor, and Sr is an optional value that is used to allow the selection of a feasible additional parent.

The values used for *Rf*, *Sr*, and *Sp* are implementation independent. The 6TiSCH protocol (RFC 8180) specifies a minimal configuration that set by default *Rf* equal to 1, *Sr* equal to 0, and *MinHopRankIncrease* equal to 256.[19] The parameter *Sp* is computed as a normalized ETX, as in Equation (3).

$$Sp = (3 \times ETX) - 2. \tag{3}$$

Hence, the *DAGrank* for each neighbor is periodically calculated following Equation (4). The preferred parent is selected as the neighbor with a minimum $R(N)$. Hysteresis is calculated, as recommended in RFC 6552, to avoid instability.

$$R(N) = R(P) + ((3 \times ETX) - 2) \times 256 \tag{4}$$

The calculation of ETX is the most important task for assigning an accurate *DAGrank* to every node and, hence, select the best path to forward the packets. The default ETX estimation specified in the 6TiSCH minimal configuration and implemented on OpenWSN * simply measures the long-term ratio of the data packets transmitted to the neighbor (variable named *numTx*) and the number of ACK packets received from the same neighbor (variable named *numTxAck*).

Since the data packets are only exchanged with the currently preferred parent, the ETX estimation concerning all other neighbors is not updated until the link to the current parent becomes too weak, which may take a long time and incur the loss of many packets.

4 | TAMU-RPL

The TAMU-RPL protocol consists of three key algorithms.

The first algorithm (see Section 4.1) uses a Thompson sampling heuristic to keep a better estimate of ETX to a subset of neighbors. It explores the $K-1$ neighbors with lowest *DAGrank* while exploiting the current best parent. The algorithm estimates the probability distribution of ETX for all K neighbors, using the number of unicast packets transmitted and acknowledgements received.

The second algorithm (see Section 4.2) is added on top of the first. The nodes take note of the channel used for each unicast packet that is transmitted and thus can measure the ETX per channel. The next hop is now calculated employing the *DAGrank* of the neighbors (as in Equation (4)), but the ETX in question is the one calculated for the channel being used for the current time slot.

The third algorithm (Section 4.3) offers a way to improve the ETX measurement by making use of broadcast packets. It uses the "PDR versus RSSI" profile of channels[20] and improves the ETX measurements based on RSSI values from unicast as well as broadcast packets.

Figure 1 shows the relationship between the algorithms and the nodes in the network.

4.1 | Thompson-based ETX estimation

Thompson sampling is a heuristic that addresses MAB problem by taking actions that maximize the expected reward based on a randomly drawn belief. It considers a set of contexts χ, a set of actions A, and rewards in R. In each round, an action a is chosen and a reward r is observed. Reward $r \in R$ follows a distribution that depends on a and on the current context x. The distribution of r is parametrized by $\theta \in \Theta$. The *prior* distribution $P(\theta)$ represents the learner's prior belief in the parameter for the rewards distribution. After observing the triplets $D = \{(x, a, r)\}$, the learner obtains a *posterior* distribution $P(\theta \mid D) \propto P(D \mid \theta) \times P(\theta)$. The parametric likelihood probability $P(D \mid \theta)$ is the probability of observing D from the rewards parametrized by θ.

In TAMU-RPL, each action $a \in A$ corresponds to the selection of a particular neighbor as the preferred parent. The underlying MAB model has one arm linked to each of the K neighbors with lowest *DAGrank*. Each node has to decide which arm (neighbor) it should play to optimize the reward. We only applied Thompson sampling to improve the ETX estimation in the *DAGrank* calculation. The equation used to calculate the *DAGrank* is the same as in Equation (4).

*www.openwsn.org

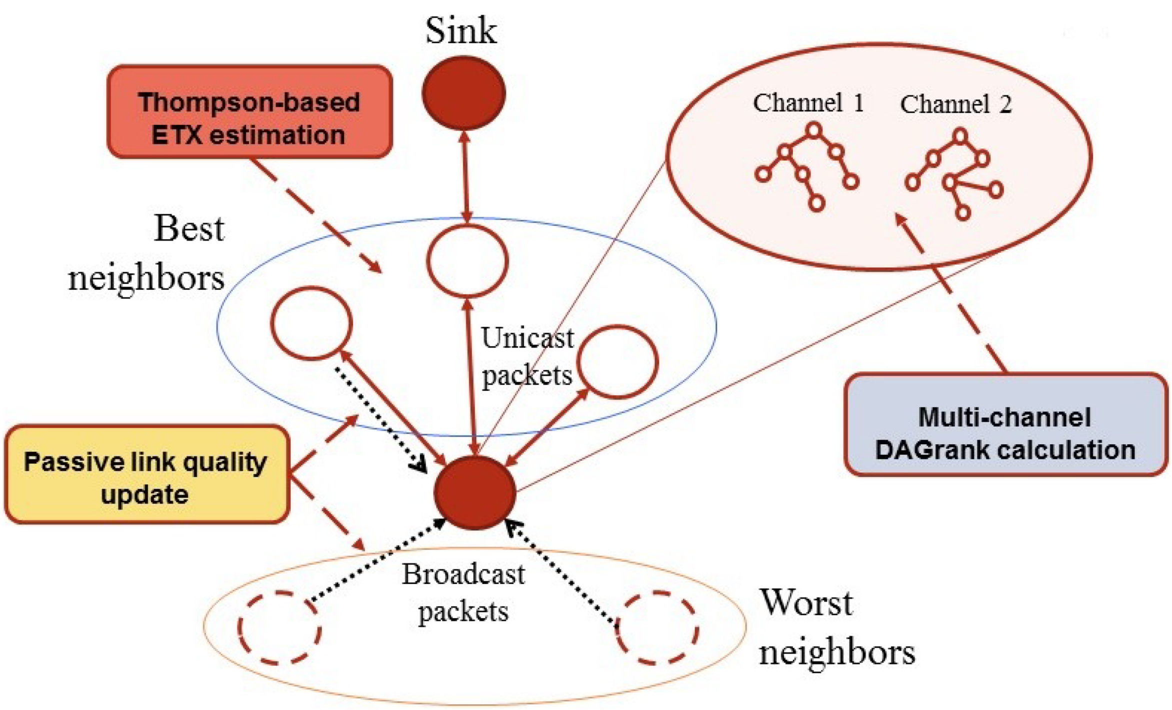

FIGURE 1 TAMU-RPL algorithms

The distribution of the number of acknowledged packets can be modeled as a binomial random variable with parameter p, where p is the PDR link. The prior distribution is *Beta*. The reward is the number of successful transmissions (k) in the last N trials.

The algorithm starts with prior distribution $Beta(1,1)$ (uniform over [0,1]) for all the neighbors. The posterior distribution is as follows[21]:

$$\text{Beta}\left(1+\sum_{i=1}^{n}x_i, 1+\sum_{i=1}^{n}N_i-\sum_{i=1}^{n}x_i\right)=\text{Beta}(1+S_n, 1+F_n), \tag{5}$$

where x_i is the number of successful transmissions up to iteration i, and N_i is the number of trials (successful and failed transmissions) up to iteration i. The posterior can be simply calculated by using parameters S_n and F_n, which are the number of successful and failed transmissions, respectively.

The algorithm for ETX estimation in TAMU-RPL is shown in Algoritm 1.

Algorithm 1 Thompson sampling with binomial observations and *Beta* prior

for each t = 1,2,..., **do**
  L = list of K neighbors with lowest *DAGrank*
  for each neighbor i in L **do**
    Independently sample $\theta_{i,t} \sim Beta(1+S_{i,t-1}, 1+F_{i,t-1})$
    Set $E\hat{T}X_i = 1/\theta_{i,t}$
  end for
  Select the preferred parent as the neighbor with minimum cost using Equation (4), where ETX is replaced by $E\hat{T}X_i$
  Update number of successful transmissions ($S_{N_t,t}$)
  Update number of failed transmissions ($F_{N_t,t}$)
end for

TAMU-RPL does not require any parameter to be set, except K, which is the number of neighbors that will be included in the sampling.

Finally, it should be noted that although the estimated ETX ($E\hat{T}X_n$) was used to calculate the cost for each neighbor, the *DAGrank* that is broadcast by each node should use the measured ETX, as defined below.

$$ETX_n = S_n/(S_n + F_n), \tag{6}$$

where ETX_n is the measured ETX to neighbor n, and S_n and F_n are the number of unicast data packets received successfully and failed, respectively.

As time goes by, it is expected that both $E\hat{T}X_n$ and ETX_n will converge to the same value for all neighbors.

4.2 | Multichannel *DAGrank* calculation

In any multichannel LLN, the link qualities vary considerably in different frequencies. Both the estimated and measured ETX used for the algorithm in Section 4.1, make use of the aggregated statistics over all 16 channels. We have improved the performance of TAMU-RPL by keeping track of the number of successful and failed transmissions per channel (S_n and F_n, respectively) in Equation (6). Hence, each node stores the aggregated measured ETX (see Equation (6)) as well as the measured ETX per channel, as defined below.

$$ETX_{n,c} = S_{n,c}/(S_{n,c} + F_{n,c}) \tag{7}$$

When a node is transmitting a data packet, it calculates Equation (4) for each neighbor by measuring the current channel statistics ($S_{n,c}$ and $F_{n,c}$). If the rank of a given neighbor is smaller than the rank of the current preferred parent by a certain threshold, the data packet is forwarded to the other neighbor instead of to the preferred parent. In this way, whenever the preferred parent has a bad link quality in the current channel, an alternative neighbor is used as a relay for data packets.

This stepwise improvement has in practice (see Section 5 and Section 6) proved to be important to avoid using channels with bad quality, especially when the preferred parent is located near a Wi-Fi access point that causes interference in a particular portion of the 2.4-GHz band.

4.3 | Passive link quality update

The last algorithm that makes up TAMU-RPL leverages the broadcast feature of wireless communications and seeks to keep the (multichannel) link quality estimation updated even when no unicast packet is exchanged with the neighbors. This is achieved using RSSI information from broadcast and unicast packets that are overheard, ie, the packets that have a different destination address but are still received by nodes in listening mode. The passive link quality update algorithm is useful since (1) all the nodes in TSCH periodically transmit (broadcast) beacons from which the RSSI information can be extracted and used by all the neighbors, and (2) the nodes may stop exchanging unicast packets either because the routing table points to a different node, or because the application does not generate sufficient data.

For every (unicast or broadcast) packet that a node successfully overhears, the RSSI value is stored and used to update the current estimated ETX to the transmitter node. The RSSI from the received (overheard) packets indicates the link quality in the reverse direction from what we want to estimate. We assume that links are symmetrical and, hence, use the RSSI from received packets to estimate the PDR of the outgoing link (ie, from the node that overheard the packet to the transmitter node). Although link quality is asymmetrical in many indoor deployments,[22] some other researchers[23] have also shown that symmetry is a valid assumption, especially for links with high or low PDR.

It has been shown that wireless channels can be well characterized by the "PDR versus RSSI" curve.[20] Given this, the current estimated PDR can be derived from RSSI measurements by applying regression to a "PDR versus RSSI" curve that is built during the network operation. In our passive link quality update algorithm, all the RSSI are taking into account, both from unicast and broadcast packets, from overheard packets as well as from packets directed towards the node estimating the link quality.

The RSSI measurements vary largely due to external interference and internal noise. Kalman filters can be used to make a more accurate estimate of unknown variables based on a series of measurements observed over some time with statistical noise. They have been used for different applications, including wireless link estimation.[24,25] Our implementation of Kalman filter was based on the work of Senel et al[24] but required a different filter for each of the 16 channels. Besides, we did not consider that a node has a precalibrated "PDR versus RSSI" curve but used the last E PDR-RSSI pairs of values

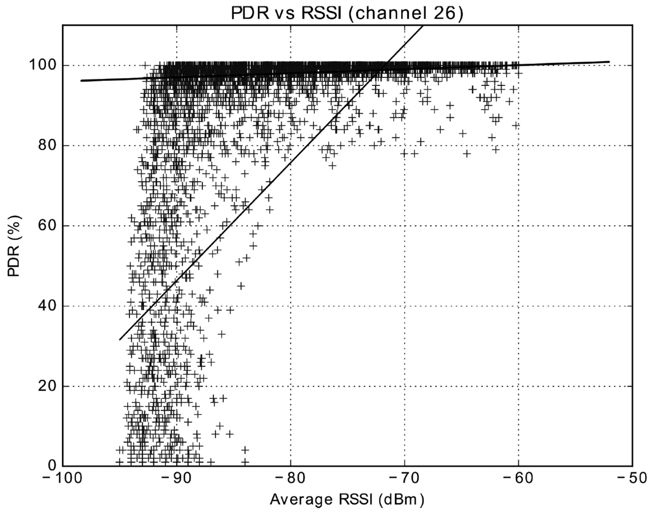

FIGURE 2 Example of linear regression with two linear functions and threshold r equal to 90%

to form our "PDR versus RSSI" curve. In this way, the "PDR versus RSSI" curve changes over time and captures the link quality dynamics.

After calculating the current estimated RSSI, we converted it to map an estimated PDR. This was carried out through linear regression. We approximated the "PDR versus RSSI" curve by two linear functions. The first linear function contains all the data points with PDR below a threshold t, while the second function contains all the data points above the same PDR threshold. In our simulations and testbed evaluation, the threshold t was equal to 90%. Figure 2 shows an example of a PDR versus RSSI plot with the suggested approximation by two linear functions.

The data points (PDR-RSSI pairs) needed for the linear regression calculation are only unicast packets exchanged between the node calculating the linear regression and the neighbors, since these are the only packets that give us more precise PDR statistics because of the acknowledgement packet that follow them. Hence, every node keeps track of the last p unicast packets exchanged with each neighbor. These packets can be data or any other unicast. The PDR associated with each of the p data points is calculated by following a moving average, as follows:

$$PDR_{\text{current}} = \begin{cases} \alpha \times PDR_{\text{last}} + (1-\alpha) \times 100, & \text{if ACK was received} \\ \alpha \times PDR_{\text{last}}, & \text{otherwise} \end{cases} \tag{8}$$

In the case of all the packets that have an ACK successfully received, the RSSI from the ACK is associated with the PDR, as calculated in Equation (8). Since there is no RSSI information for packets that did not have an ACK received, only successful transmissions are included in the data points. However, for every packet lost (with no ACK received), the PDR estimation in Equation (8) is updated to keep track of the most up-to-date link quality.

5 | SIMULATION RESULTS

We first evaluated the performance of TAMU-RPL employing simulations to assess its effectiveness and compare its performance with an optimal solution. We used a custom-made simulator written in C language.[†] The simulator receives a set of connectivity traces and, based on the PDR statistics, calculates the routing paths with different types of RPL implementations. The simulator uses a TSCH slotframe with 101 time slots (all of them of shared type), where control (DIO, KA, etc) or data packets are exchanged.

The DIO packets are broadcast by each node to announce its *DAGrank* to the neighbors. It starts with a fixed interval (by default 2 seconds) and the interval is doubled until a network change resets the interval to the minimum value. The default interval of KA packets is 10 seconds. The data packets are transmitted by default with an interval of 30 seconds.

[†]The source code of the simulator tool can be found at https://github.com/pedrohenriquegomes/tsch-scheduler-and-simulator

We used connectivity traces obtained from the Tutornet[‡] test bed.[26] The data sets represent the connectivity of 40 nodes. All the data sets of connectivity traces are publicly available.[§]

5.1 | Different RPL implementations

We implemented two variants of RPL and two types of TAMU-RPL protocol so that the results could be compared.

We first implemented *RPL with MHROF*; this uses Equation (4) for the *DAGrank* calculation by taking ETX as the ratio between the data packets transmitted and ACK packets received. In *RPL with MHROF*, the nodes have to determine an initial ETX for the links to neighbors to which, until then, no unicast packet has been exchanged. By default, this ETX value is equal to 4, which makes the algorithm conservative. In the experiments, we evaluated how this parameter affects the performance of *RPL with MHROF*. One could alternatively derive the initial ETX value based on the RSSI of the first packets received from a node. Such improvement can reduce the time required for precisely estimating the ETX, but due to noise, it is not guaranteed that a few RSSI measurements would lead to an accurate estimation, even if a precalibrated curve is estimated with measurements from the environment.

We also implemented *Dijkstra with MHROF*, where the shortest-path tree is calculated with the Dijkstra algorithm. It is an ideal protocol where the weights of each edge of the routing tree are calculated with complete knowledge of the PDR of all links. The *DAGrank* calculation of *RPL with MHROF* is kept, which means that the weights in the shortest-path are calculated as in Equation (4), but using the actual ETX values from the connectivity traces. This is the best solution that we want to compare TAMU-RPL with.

Finally, we implemented two variants of TAMU-RPL. *Single-channel TAMU-RPL* uses the aggregated statistics over all 16 channels. Hence, the algorithms shown in Section 4.2 and Section 4.1 are not used. Multichannel TAMU-RPL uses all the algorithms shown in Section 4, which means that it considers statistics for each channel when estimating ETX, and uses information from unicast and broadcast packets. *Multichannel TAMU-RPL* implements the entire TAMU-RPL proposal.

5.2 | Evaluating *DAGrank* variation over time

We carried out a set of 1-hour simulations to understand how nodes learn the best neighbors. These experiments also aimed at demonstrating the differences between algorithms *RPL with MHROF* and *single-channel TAMU-RPL*. The end-to-end ETX is defined as the sum of the ETX of links in the routing tree from a sensor node to the sink. We first evaluated over a short period which nodes would have the highest end-to-end ETX, ie, the nodes more distant from the sink. We plotted the variation of *DAGrank* of the four nodes with the highest end-to-end ETX. The *DAGrank* is the cost of a path from a given node to the sink; hence, the smaller the *DAGrank* the better. The interval of 1 hour was chosen because this is a period long enough to show the convergence of the algorithms and longer periods are harder to be shown in a plot due to the number of data points in the graphs.

Figure 3 shows the variation over time of the *DAGrank* for all the four nodes selected. It corroborates the fact that all the nodes have a smaller *DAGrank* when they use *single-channel TAMU-RPL*, which means that neighbors with better links are selected and it is possible to lower the cost to the sink. The variation of *DAGrank* is much higher for the *single-channel TAMU-RPL* since the neighbors are more widely explored. The variations in the *DAGrank* can be attributed to the changes of parent and the dynamics of the links, which causes the ETX to change over time. It should also be noted that the exploration stages of *single-channel TAMU-RPL* may give rise to a large increase in the *DAGrank* and even sometimes make *single-channel TAMU-RPL* perform worse than *RPL with MHROF*.

5.3 | Evaluating end-to-end ETX over time

We used more connectivity traces from the testbed and carried out 8-hour simulations to determine how the end-to-end ETX of the network varies over time. Every 15 minutes a new set of PDR matrices is loaded for each link in this new simulation. We added up the end-to-end ETX of all the 40 nodes in the network and plotted the variations of this value over time in Figure 4.

The comparison was made between *single-channel TAMU-RPL*, *RPL with MHROF*, and *Dijkstra with MHROF*. We took advantage of this set of simulations and assessed the effect that the initial ETX value for new links had on the *RPL with*

---

‡ https://anrg.usc.edu/www/tutornet/

§ https://github.com/pedrohenriquegomes/phd-datasets

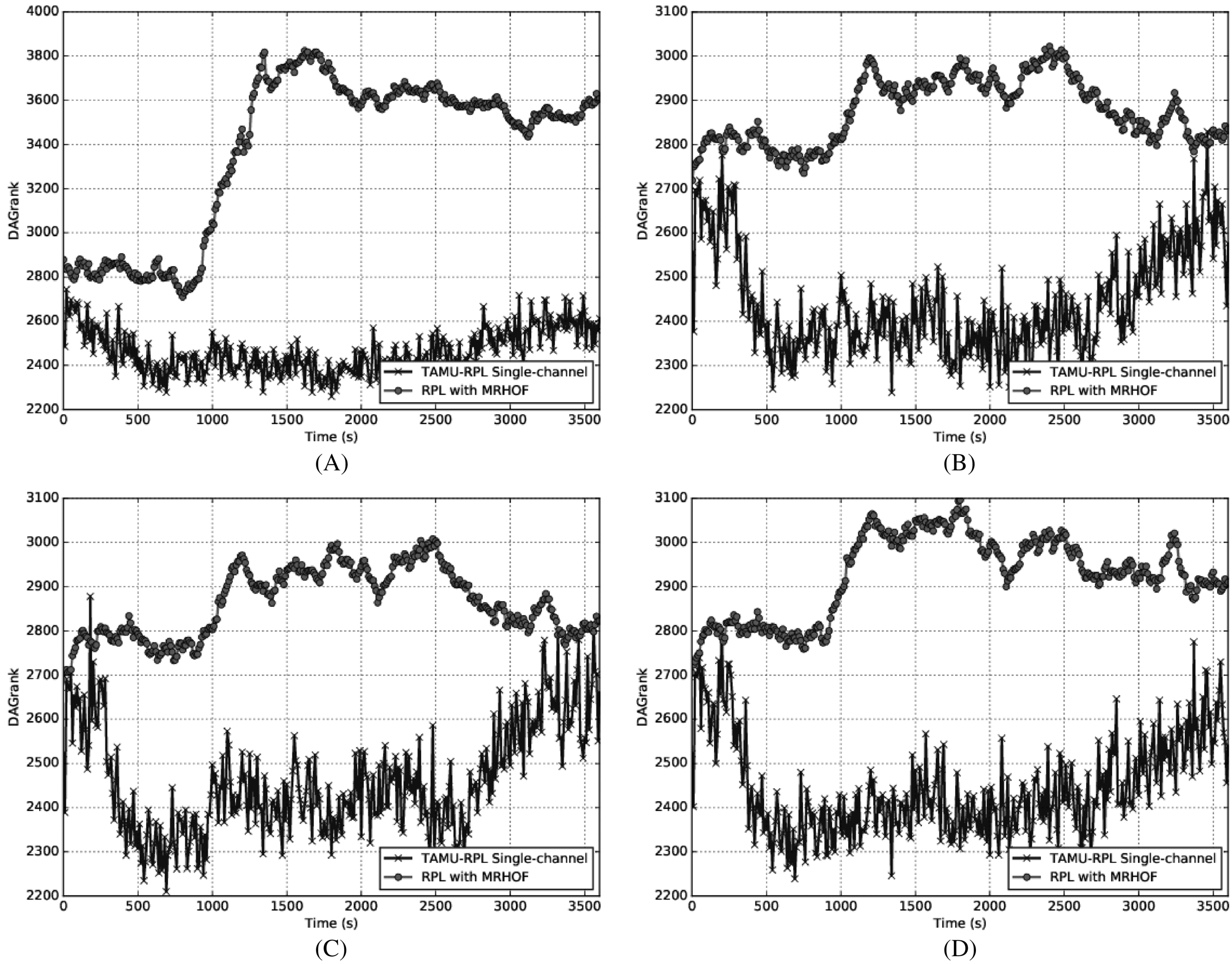

FIGURE 3 *DAGrank* variation over time. A, Node 23; B, Node 27; C, Node 31; D, Node 35

MHROF. In *RPL with MHROF*, whenever a new neighbor is discovered, a default value for the link to that neighbor has to be set before the unicast data packets are exchanged. By default, this value is equal to 4.0 in the OpenWSN implementation. We assessed to what extent the performance of *RPL with MHROF* changes if this value is reduced from 4.0 to 1.0.

Figure 4 shows the end-to-end ETX sum for all the nodes in the Tutornet test bed. The simulation of *RPL with MHROF* was carried out by assuming that the initial ETX of the new links was equal to 1.0, 2.0, 3.0, and 4.0, and plotted four different graphs to help the visualization.

Figure 4 has a stepwise behavior. This is because every 15 minutes a new set of network connectivity matrices are loaded into the simulator. The *Single-channel TAMU-RPL* obtained a sum of end-to-end ETX that is smaller than *RPL with MHROF* and much closer to the *Dijkstra with MHROF*. *RPL with MHROF* has slightly better results when the default ETX is equal to 1.0, but this incurs more variability of the ETX, which makes *RPL with MHROF* to perform much worse at certain times. Since we want to make a fair comparison, we use the default ETX value of *RPL with MHROF* equal to 1.0 for the rest of the simulations.

5.4 | Evaluating loop formation

As mentioned in Section 4, the *DAGrank* announced by the nodes in both *Multichannel TAMU-RPL* and *Single-channel TAMU-RPL* uses the measured ETX values. Since the node that is selected as preferred parent changes frequently before a good estimate of ETX to the neighbors is achieved (or whenever the network statistics change rapidly), the number of loops formed in TAMU-RPL is likely to be higher than in *RPL with MHROF*. RPL has mechanisms for loop detection and in regular network operations, this problem would be resolved, with some overhead and energy waste.

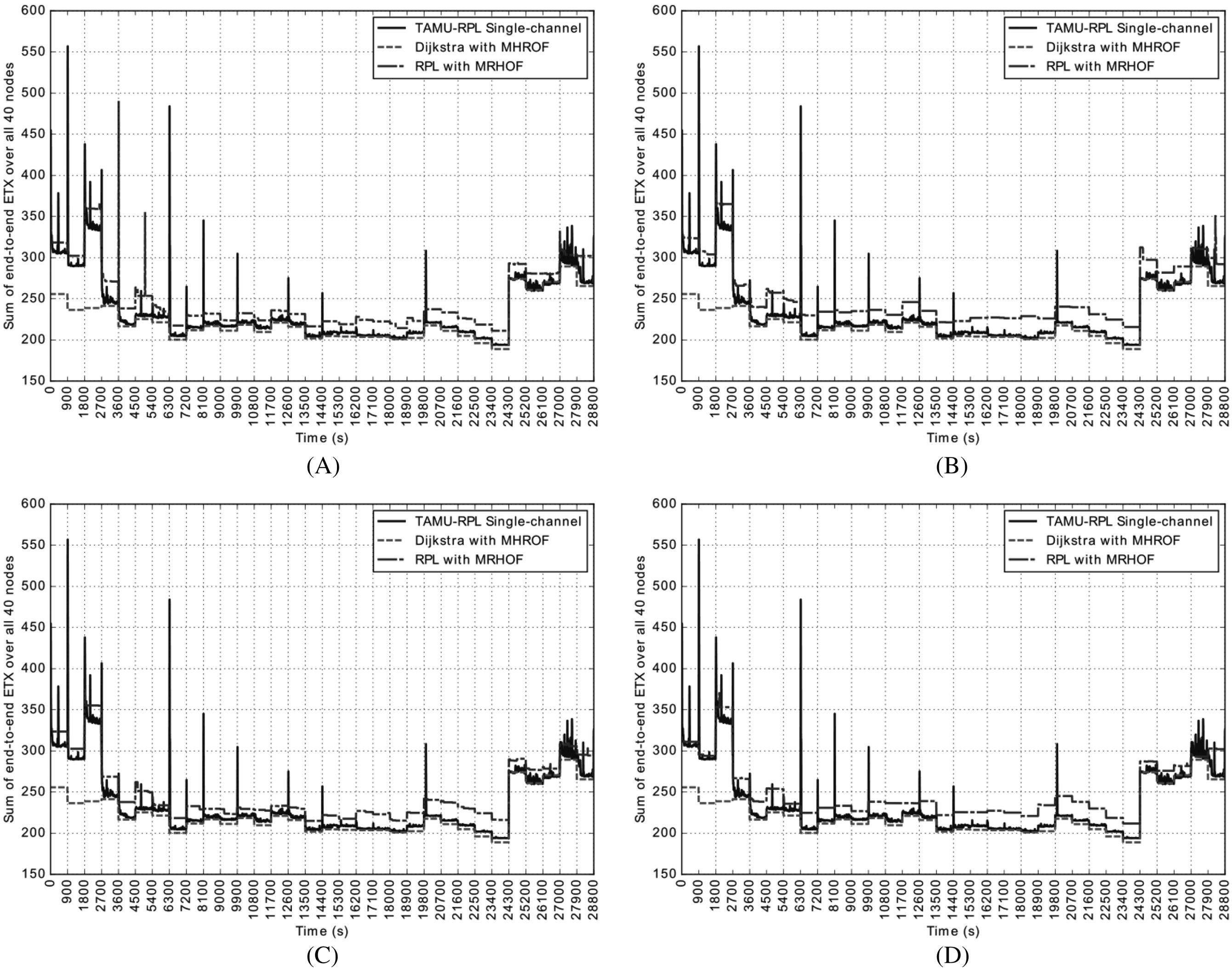

FIGURE 4 End-to-end ETX sum for all the nodes. A, RPL with MHROF default ETX = 1.0; B, RPL with MHROF default ETX = 2.0; c, RPL with MHROF default ETX = 3.0; D, RPL with MHROF default ETX = 4.0

The loop detection mechanism that runs during network operations was not implemented in the simulations. Instead, we were able to check (and avoid) the formation of loops whenever a new routing tree was formed. If a loop is detected, the change of preferred parent is avoided, and the current preferred parent is kept.

Figure 5 shows the number of loops formed (average and standard deviation) for *single-channel TAMU-RPL* and *RPL with MHROF*. We also examined two cases for *RPL with MHROF*, when the default ETX is equal to 4.0, and when it is equal to 1.0. Figure 5 shows the number of loops detected for each node.

It can be seen that only a few nodes formed loops in the routing tree if *RPL with MHROF* with a default ETX equal to 4.0 was used. In this case, the nodes spend less time exploring the possible neighbors and stick to the first nodes from which they listened a DIO. On the other hand, more loops are formed when more changes are made in the routing tree. Besides, more loops are formed when *single-channel TAMU-RPL* and *RPL with MHROF* are employed with default ETX equal to 1.0. It can be seen that *single-channel TAMU-RPL* forms loops, but the number of occurrences is less than *RPL with MHROF* with default ETX equal to 1.0. This shows that the use of *TAMU-RPL* does not imply much more overhead than *RPL with MHROF* with more exploration.

5.5 | Evaluating the number of packets received at the sink

The nodes transmit unicast data packets in addition to the DIOs and KAs in the next two analyses. The data packets are forwarded towards the sink node by employing the current routing tree that is built by the different RPL implementations. The four different RPL implementations described in Section 5.1 were compared. New data packets were generated by

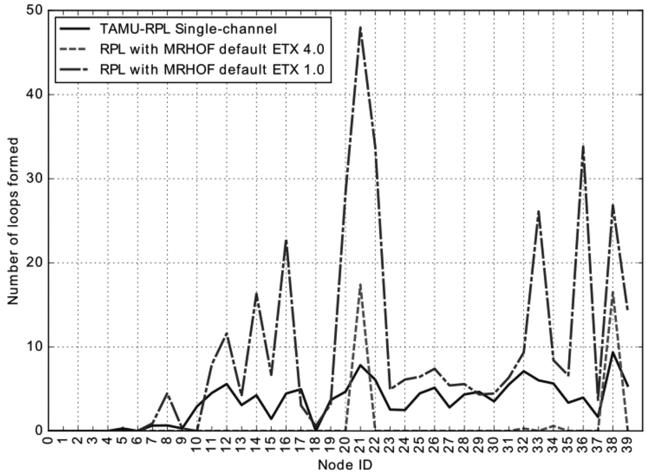

FIGURE 5 Number of loops formed in the simulation

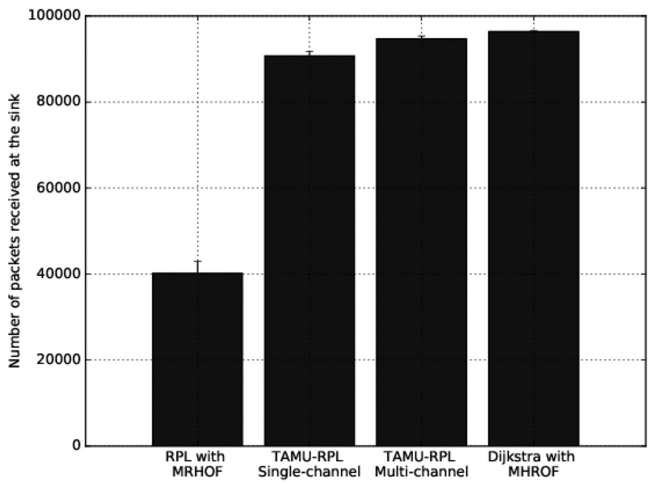

FIGURE 6 Number of packets received at the sink

each nonsink node periodically every 30 seconds. The number of link-layer retransmissions was set to 3. The total number of data packets transmitted in the network was 112 320.

Figure 6 shows the number of data packets received at the sink.

It can be seen that the number of packets received when both variants of *TAMU-RPL* were employed is more than double the number of packets received with *RPL with MHROF*. Moreover, *TAMU-RPL* was able to achieve results much closer to the results obtained by *Dijkstra with MHROF*. Out of the 112 320 packets transmitted, more than 95 000 were received with *TAMU-RPL*. This was possible because the ETX of links was reduced and link-layer retransmission was able to recover packet losses.

5.6 | Evaluating the delay of packets received at the sink

The packets are timestamped when they are generated and when they reach the sink node, and we analyzed how long they take to traverse the routing tree. Whenever a packet is relayed or retransmitted due to a packet loss, a delay is randomly chosen with a uniform distribution between 5 and 10 time slots.

Figure 7 shows the number of time slots needed for the data packets to be received at the sink.

It can be seen that the delay of the packets received at the sink was reduced in the case of both variants of *TAMU-RPL*. This means that the packets required fewer retransmissions along the routing tree. The large standard variation is due to the different number of hops from the nodes to the sink, hence the different delays affecting each packet. *TAMU-RPL* can reduce the delay (in timeslots) by more than 10% when compared to *RPL with MHROF*. This results, again, from the reduction of ETX of the links employed in *TAMU-RPL*, which was possible due to the quick adaptation of the routing tree to link changes in the environment.

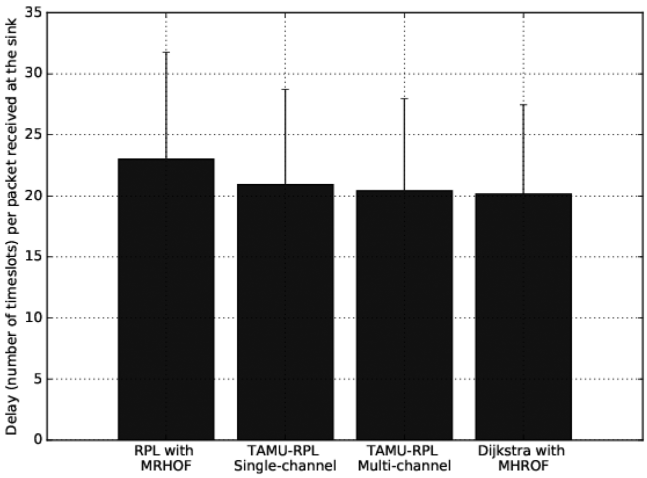

FIGURE 7 Delay (number of time slots) per packet received at the sink

5.7 | Evaluating the energy consumption

We evaluated the energy consumption of different solutions. The only difference between *TAMU-RPL* and the other two solutions (*RPL with MHROF* and *Dijkstra with MHROF*) is the number of control packets transmitted. The RPL protocol broadcasts DIO messages that include the current node's DAGrank, among other information. The Trickle algorithm is used to reduce the number of packets transmitted. The initial interval is 2 seconds, and this value is doubled for every DIO that is transmitted until it reaches a maximum value (in our case 60 seconds) or a networking event occurs, eg, preferred parent is changed. Since in TAMU-RPL the preferred parent is changed much more frequently, the number of DIOs transmitted is expected to be increased.

Figure 8 shows the number of DIO packets transmitted in the experiment.

It can be seen that the number of DIO packets transmitted in TAMU-RPL solutions is much higher than in *RPL with MHROF* and *Dijkstra with MHROF*. The large overhead is caused by the frequent reset on the Trickle timer due to changes in the preferred parent choices. This overhead could be reduced drastically with the adaptation of Trickle algorithm and its parameters. It is clear that for TAMU-RPL since the changes in the routing topology is very frequent; therefore, the interval of DIO broadcast could increase faster, eg, multiplying by 3 or 4 the interval for every DIO transmitted. Besides that, DIO packets are much shorter than Data packets and, hence, the energy increase is not as much as one can expect just looking at Figure 8. We decided to display the number of packets instead of energy consumption because the latter depends on different factors, eg, transmit power, etc.

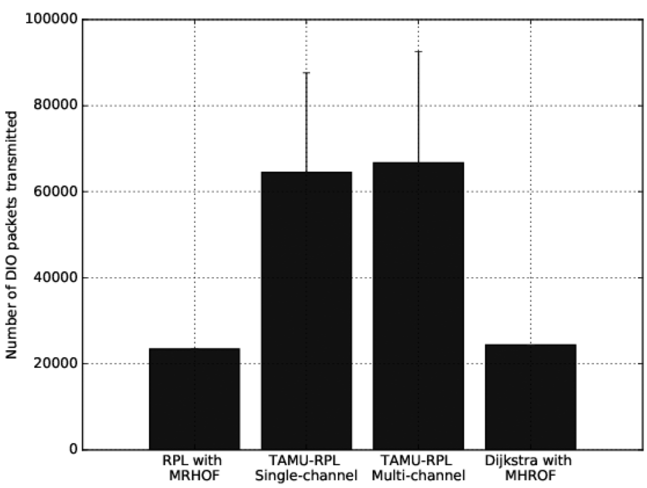

FIGURE 8 Number of DIO packets transmitted by all nodes

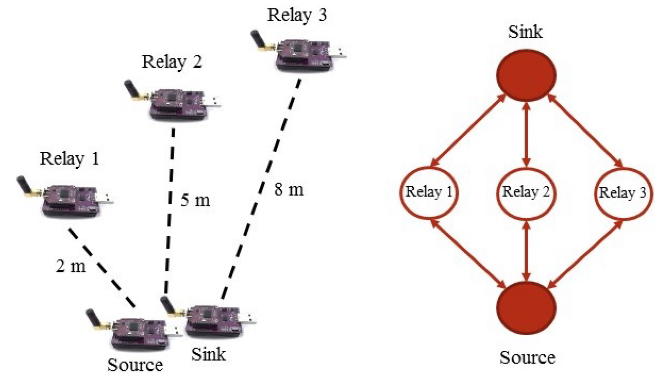

FIGURE 9 Physical placement of five OpenMote nodes (left) and logic topology (right)

6 | EXPERIMENTAL RESULTS

TAMU-RPL was also evaluated experimentally. We implemented *single-channel TAMU-RPL* and *multichannel TAMU-RPL* on OpenWSN 1.10[¶] and OpenMote as the hardware platform. Both the TAMU-RPL variants were compared with *RPL with MHROF*.

For Algorithm 1, we implemented the widely used Cheng's BB algorithm[27] to generate *Beta* distribution. Within the slotframe that contains 101 time slots, we reserved one time slot where each node runs the Thompson-sampling ETX estimation. For the passive link quality estimation (Section 4.3) we considered the recommended values for the parameters Q (the estimation variance) and R (the measurement variance) of the Kalman filters.[24] The last 10 RSSI measurements and 20 noise measurements were stored and every 10 time slots (ie, approximately every second) Q and R were updated. For the linear regression responsible for mapping the estimated RSSI to an estimated PDR value (Section 4.3), all the nodes store the current estimated RSSI of the last 10 unicast packets (ie, data, ACK, or KA packets) that were successfully received and associate that value with the current estimated PDR calculated with Equation (8). This table with the last ten pairs of "PDR versus RSSI" is stored per channel. Every time a new broadcast packet is received or a unicast packet is overheard from a given neighbor, the RSSI measurement is mapped to an estimated PDR. An estimated ETX is computed as the inverse of the estimated PDR.

The ROM consumption of TAMU-RPL is equal to 10 401 bytes. Almost 2 KB was only consumed by the *Beta* random variable generation code that was used in Algorithm 1. The RAM consumption was approximately 1760 bytes; the code for Section 4.1 required 480 bytes, the code for Section 4.2 required 960 bytes, and the code for Section 4.3 required 320 bytes. Some extra functions also required RAM space, but they accounted for less than 300 bytes.

6.1 | Evaluating TAMU-RPL in a real environment

The initial ETX value for *RPL with MHROF* was set to 1.0. As in the simulation, the slot frames were set up with 101 time slots of 10 ms of duration each. Out of the 101 time slots, 5 were used for serial communication between the computer and the sensor nodes, 1 was used for running TAMU-RPL calculations, and the rest (95 time slots) were of shared type. The nodes exchange DIO packets, which are broadcast, and KA and data packets, which are unicast. DIO packets are transmitted with an interval of 30 seconds, KA with an interval of 10 seconds, and data packets with an interval of 1 second.

We started the evaluation process considering a small setup with 5 OpenMote nodes. The physical placement consists of one sink node close to a source node. There are also three relay nodes placed at different distances from the sink and source nodes, at approximately 2 m, 5 m, and 8 m. The environment is a working office with average levels of external interference from Wi-Fi networks. Figure 9 illustrates the physical placement and the logical topology.

We filter out packets in OpenWSN so that, although the sink and the source nodes are close to each other, they do not process packets from each other. Both nodes (sink and source) only process to packets from the relay nodes. This setup creates three possible paths from the source to the destination, each one with two hops that have the same link quality. The main purpose of such a scenario is to verify how the source node selects the relay nodes as its best parent towards the sink node. All experiments lasted 1 hour and were repeated 10 times during different times of the day. Only the sensor node transmits data packets; hence, there are a total of 3600 packets transmitted in each experiment. All nodes report statistics to a laptop connected through USB cable once every slotframe.

[¶]The source code of TAMU-RPL can be found at https://github.com/pedrohenriquegomes/tamu-rpl.

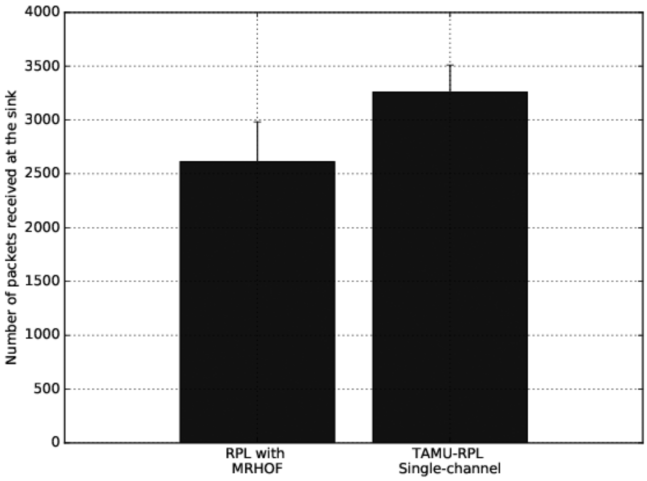

FIGURE 10 Number of packets received at the sink node for the *Experiment 1*. Initially all three relays are on; at 15 minutes *Relay 1* is turned off, at 30 minutes *Relay 2* is also turned off, and at 45 minutes *Relay 1* is turned back on

In *Experiment 1*, we analyzed the reactiveness of TAMU-RPL. We start unicast data transmission from source to the sink with all three relays on. After 15 minutes, *Relay 1* is turned off, and only *Relay 2* and *Relay 3* remain on. After 15 minutes more, *Relay 2* is also turned off and only *Relay 3* remains on. After 15 minutes more, *Relay 1* is turned back on and *Relay 1* and *Relay 3* remain on. Figure 10 shows the number of received packets at the sink node during the *Experiment 1*.

Single-channel TAMU-RPL can deliver approximately 90% of the packets transmitted in the experiment. It outperforms *RPL with MHROF* by approximately 20%. This performance is much lower than the result obtained in the simulations, probably due to the higher levels of external interference in the *Tutornet* testbed when compared to the environment used in the experimental evaluation. Due to the limited levels of external interference, the difference of packets received should be mainly caused by packets losses while relay nodes are turned off.

We analyzed which preferred parent the sensor node has selected over time in *Experiment 1*. From the log files, we noticed that, as expected, *single-channel TAMU-RPL* keeps changing the selected preferred parent over time, while *RPL with MHROF* usually sticks to the first parent that it selects. In general *RPL with MHROF* is much slower reacting to nodes being turned off and drops many packets before switching to an available neighbor. Besides, in some cases, the preferred parent selected by *RPL with MHROF* does not have the best link, which incurs more packet loss due to low PDR in the link. Figure 11 shows the preferred parent selection in two situations; in Figure 11A, we can see a common case where both protocols select the best parent, but *RPL with MHROF* reacts more slowly to the changes in the topology, and in Figure 11B we can see a case where *RPL with MHROF* did not select the best parent at time 15 minutes and remained using *Relay 3*. In none of the experiments, we noticed that *Single-channel TAMU-RPL* made the wrong decision and got stuck with a parent that did not have the best link.

We analyzed the log files from all 10 experiments and calculated the average time that *single-channel TAMU-RPL* and *RPL with MHROF* take to switch to the parent with the best link. We disregarded the cases where *RPL with MHROF* did not switch to the parent with the best link. On average, *single-channel TAMU-RPL* took 26.4 seconds with a standard deviation of 19.7 seconds. Moreover, *RPL with MHROF* took 144.9 seconds with a standard deviation of 96.2 seconds. We can conclude that *RPL with MHROF* is not able to quickly detect when nodes in the network are turned on and off, and this incurs a large number of packet losses.

In the *Experiment 2*, we analyzed how TAMU-RPL can detect external interference in a specific channel and avoid neighbors located close to the sources of interference.

We changed the FHSS used and only utilized three channels in the hopping sequence. The IEEE 802.15.4 channels used were channel 12, 17, and 22, which overlaps with the three most commonly used Wi-Fi channels, ie, channels 1, 6, and 11. Close to *Relay 1*, we placed a Wi-Fi access point working on channel 1, which interferes mostly to IEEE 802.15.4 channel 12. The experiment started with the Wi-Fi network off, and after 15 minutes, it is turned on and two laptops start to exchange traffic using software iPerf3.[#] The Wi-Fi traffic lasts 30 minutes and is turned off at 45 minutes. Since the Wi-Fi network interferes mostly to one particular IEEE 802.15.4 channel (channel 12), we expect that the source node would avoid relaying packets to *Relay 1* in time slots that use this particular channel. We also set up a variable that keeps track of the channel utilized and makes sure that all three IEEE 802.15.4 channels are used uniformly in a round-robin fashion.

[#] https://iperf.fr/.

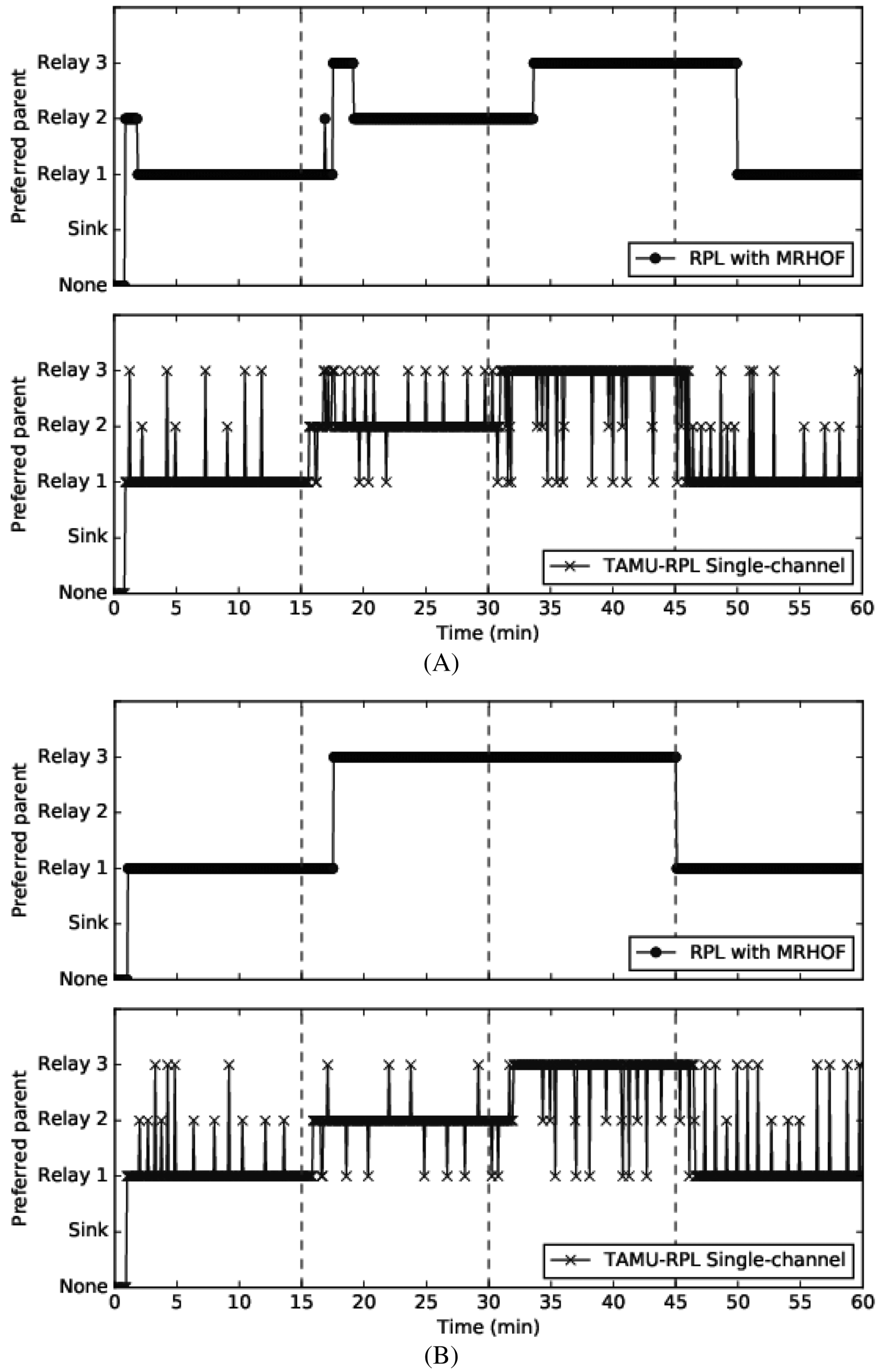

FIGURE 11 Preferred parent selection over time for the *Experiment 1*. A, Example of situation where *RPL with MHROF* makes right decisions, but delayed; B, Example of situation where *RPL with MHROF* makes wrong decisions (chooses *Relay 3* instead of *Relay 2* at time 15 minutes)

Figure 12 shows the number of received packets at the sink node during the *Experiment 2*.

The difference in performance increases to about 33% when we compare *Single-channel TAMU-RPL* with *RPL with MHROF*. This difference is mainly caused by the fact that *RPL with MHROF* often is not able to detect the degradation of the link quality between source and *Relay 1* and does not switch the preferred parent. Moreover, Figure 13 shows an example of preferred parent selection made by the source node over time. On the other hand *single-channel TAMU-RPL* switches the preferred parent to *Relay 2* and can avoid the path that suffers more interference.

The difference in performance when we compare *single-channel TAMU-RPL* and *multichannel TAMU-RPL* is very small but not negligible. Although *Relay 2* is far from the source of interference, it also suffers from link degradation. When we analyzed the log files we could notice that *multichannel TAMU-RPL* protocol utilized approximately 5% fewer times channel 12, which explains the slight improvement in the number of packets received at the sink.

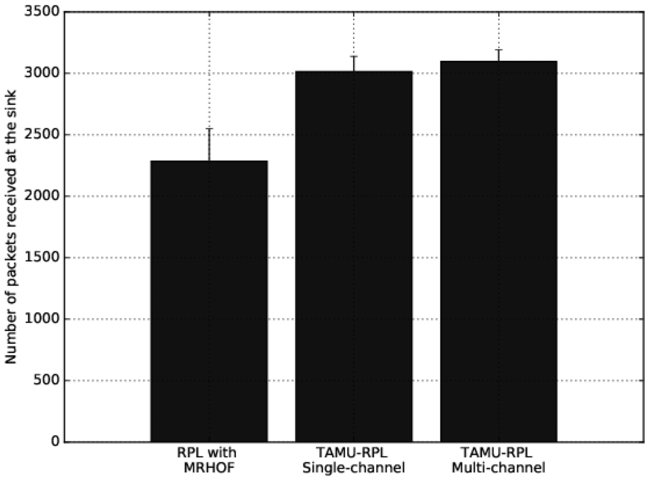

FIGURE 12 Number of packets received at the sink node for the *Experiment 2*. A Wi-Fi network that is placed close to *Relay 1* starts traffic at 15 minutes and stops traffic at 45 minutes

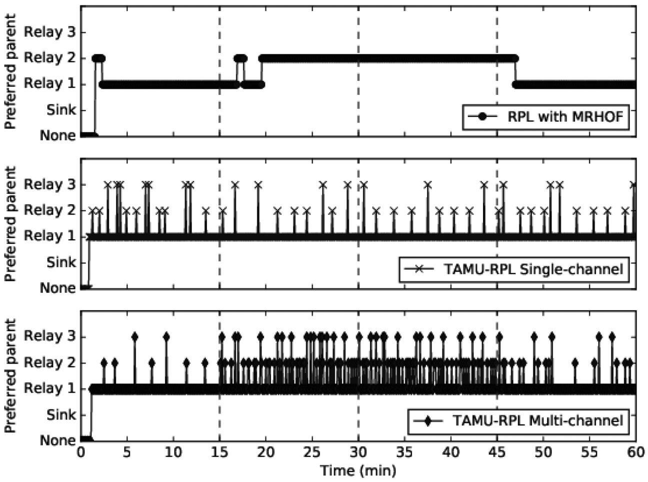

FIGURE 13 Preferred parent selection over time for the *Experiment 2*

Finally, in the *Experiment 3*, we analyzed if TAMU-RPL can keep track of the link qualities even in the absence of unicast data traffic being exchanged with neighbors. The experiment setup is similar to the *Experiment 2*. The difference is that when the Wi-Fi traffic is started, the unicast data traffic from source to the sink stops. The unicast data traffic is resumed 30 minutes after the beginning of the experiment, and the Wi-Fi traffic continues until the end of the experiment. Since there is no unicast data traffic when the external interference is large, the only way of keeping track of the link quality is by using information from broadcast packets (DIOs and KAs). We expect that TAMU-RPL can keep the link quality up-to-date and when the unicast data traffic resumes, the best parent selection is changed from *Relay 1* to another neighbor.

Figure 14 shows the number of received packets at the sink node during the *Experiment 3*.

We can check that the improvement of *Single-channel TAMU-RPL*, when compared to *Single-channel TAMU-RPL*, is of approximately 25%. On the other hand, the improvement of *multichannel TAMU-RPL* when compared to *Single-channel TAMU-RPL* is negligible (approximately 2%). Verifying the log files we could notice that *Multichannel TAMU-RPL* that the routing tree formed by *multichannel TAMU-RPL* and *Single-channel TAMU-RPL* were the same in most of the experiments. Although *multichannel TAMU-RPL* analyses the broadcast packets and estimates the PDR of links even in the absence of unicast traffic, we conjecture that the RSSI levels of broadcast packet did not change significantly in the environment, and/or the number of broadcast packets received is not sufficient to change the PDR estimation that was calculated with unicast packets. The small improvement of *multichannel TAMU-RPL* shows that more sophisticated link quality estimation based on a mix of unicast and broadcast packets may not be so efficient in environments with mild levels of external interference. However, in simulations especially using traces from testbed with high levels of interference show that *multichannel TAMU-RPL* causes significant improvements in certain environments.

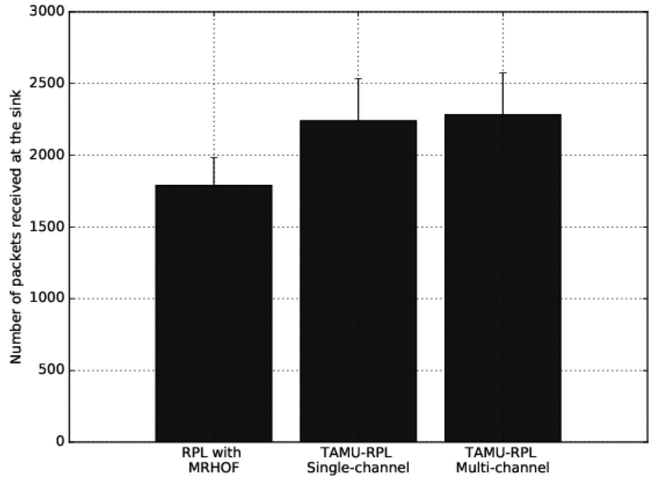

FIGURE 14 Number of packets received at the sink node for the *Experiment 3*. A Wi-Fi network that is placed close to *Relay 1*; Wi-Fi traffic starts and LLN unicast traffic stops together at 15 minutes. The LLN unicast data traffic resumes at 30 minutes

7 | LESSONS LEARNED

Based on the evaluation results, TAMU-RPL showed that it is possible to improve the reactiveness of RPL using Thompson-sampling heuristic to keep track of more accurate ETX estimation to more neighbors.

Some lessons have been learned from the evaluation process:

1. TAMU-RPL increases the reactiveness of RPL to quick changes in the link quality of neighbors.
2. When TAMU-RPL performs link quality estimation per channel, it can opportunistically avoid neighbors with bad quality in specific channels.
3. TAMU-RPL is also able to leverage the broadcast nature of transmissions and keep the link quality estimation even in the absence of unicast data packets sent towards the sink node.
4. In the simulation, TAMU-RPL outperformed the default RPL implementation and was able to achieve performance close to the one obtained by Dijkstra-based routing.
5. In real experiments, TAMU-RPL outperformed the default RPL implementation and increased the number of packets received at the sink by up to 33%.
6. In real experiments, however, the evaluation shows that the improvements of *multichannel TAMU-RPL* may be dependent on the levels of external interference in the environment.

8 | CONCLUSION

TAMU-RPL is a solution for improving the reactiveness of RPL to quick changes in the link quality from nodes to their neighbors. It is aimed at improving the performance of a network that suffers from large external interference levels. The solution involves the use of the three key algorithms. The first performs an ETX estimation based on Thompson sampling. It keeps a more accurate link quality estimation to a subset of neighbors and allows the nodes to quickly change the preferred parent when the link to it is degraded. Thompson sampling is the heuristic chosen to solve the exploration versus exploitation problem that needs to be tackled. The second algorithm implements a multichannel *DAGrank* calculation and makes the nodes to be aware of link quality to neighbors at individual channels. Whenever a node predicts that the link to the current preferred parent faces degradation at the channel being used, it forwards the packet to a neighbor with better link quality. The third algorithm leverages the broadcast nature of wireless transmissions and uses physical layer information extracted from overheard packets to improve the ETX estimation when unicast packets are not exchanged for long periods. The TAMU-RPL was able to improve the performance of RPL and achieve results closer to shortest-path Dijkstra algorithm. In the evaluation based on simulations, TAMU-RPL doubled the number of packets received at the sink when compared to RPL with MRHOF. Moreover, it was able to reduce by more than 10% the average delay (in timeslots) of packets that reached the sink node. In the evaluation based on real deployment, TAMU-RPL could increase the number of received packets at the sink by about 33%. These improvements were the results of TAMU-RPL being able to create routing trees with links with smaller ETX values, which reduced the number of packet losses. The main take away from TAMU-RPL implementation is the fact that performing online learning to better estimate the ETX to different neighbors and therefore improving the routing paths is a practical way of improving the performance of the networks.

Future work is on the study of limitations of the current approach in real scenarios where hardware constraints and higher dynamics of the environment may require simplifications that would limit the performance improvements.

ORCID

Pedro Henrique Gomes https://orcid.org/0000-0003-1425-2615

REFERENCES

1. Kim H-S, Ko J, Culler DE, Paek J. Challenging the ipv6 routing protocol for low-power and lossy networks (RPL): a survey. *IEEE Commun Surv Tutor*. 2017;19(4):2502-2525.
2. De Couto DSJ, Aguayo D, Bicket J, Morris R. A high-throughput path metric for multi-hop wireless routing. *Wireless Networks*. 2005;11(4):419-434.
3. Gai Y, Krishnamachari B. Distributed stochastic online learning policies for opportunistic spectrum access. *IEEE Trans Signal Process*. 2014;62(23):6184-6193.
4. Gulati N, Dandekar KR. Learning state selection for reconfigurable antennas: a multi-armed bandit approach. *IEEE Trans Antennas Propag*. 2014;62(3):1027-1038.
5. Gai Y, Krishnamachari B, Jain R. Combinatorial network optimization with unknown variables: multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Trans Netw*. 2012;20(5):1466-1478.
6. Liu K, Zhao Q. Adaptive shortest-path routing under unknown and stochastically varying link states. Paper presented at: 2012 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt); 2012; Paderborn, Germany.
7. Zou Z, Proutiere A, Johansson M. Online shortest path routing: the value of information. Paper presented at: 2014 American Control Conference; 2014; Portland, OR.
8. Clausen T, Herberg U, Philipp M. A critical evaluation of the IPv6 routing protocol for low power and lossy networks (RPL). Paper presented at: 2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob); 2011; Wuhan, China.
9. Ancillotti E, Vallati C, Bruno R, Mingozzi E. A reinforcement learning-based link quality estimation strategy for RPL and its impact on topology management. *Computer Communications*. 2017;112:1-13.
10. Djedjig N, Tandjaoui D, Medjek F, Romdhani I. New trust metric for the RPL routing protocol. Paper presented at: 2017 8th International Conference on Information and Communication Systems (ICICS); 2017; Irbid, Jordan.
11. Karkazis P, Papaefstathiou I, Sarakis L, Zahariadis T, Velivassaki T-H, Bargiotas D. Evaluation of RPL with a transmission count-efficient and trust-aware routing metric. Paper presented at: 2014 IEEE International Conference on Communications (ICC); 2014; Sydney, Australia.
12. Iova O, Theoleyre F, Noel T. Using multiparent routing in RPL to increase the stability and the lifetime of the network. *Ad Hoc Netw*. 2015;29:45-62.
13. Oliveira TB, Gomes PH, Gomes DG, Krishnamachari B. ALABAMO: a load balancing model for RPL. Paper presented at: Brazilian Symposium on Computer Networks and Distributed Systems (SBRC); 2016; Belem, Brazil.
14. Ancillotti E, Bruno R, Conti M. Reliable data delivery with the IETF routing protocol for low-power and lossy networks. *IEEE Trans Ind Inform*. 2014;10(3):1864-1877.
15. Kim K-H, Shin KG. On accurate and asymmetry-aware measurement of link quality in wireless mesh networks. *IEEE/ACM Trans Netw*. 2009;17(4):1172-1185.
16. Hermeto RT, Gallais A, Laerhoven KV, Theoleyre F. Passive link quality estimation for accurate and stable parent selection in dense 6TiSCH networks. In: Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks; 2018; Madrid, Spain.
17. Winter T, Thubert P, Brandt A, et al. Rpl: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC Editor. Internet Requests for Comments. 2012. http://www.rfc-editor.org/rfc/rfc6550.txt
18. Gnawali O, Levis P. The Minimum Rank with Hysteresis Objective Function. RFC Editor. Internet Requests for Comments. 2012. http://www.rfc-editor.org/rfc/rfc6719.txt
19. Pister K, Watteyne T. Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration. RFC Editor. Internet Requests for Comments. 2017. http://www.rfc-editor.org/rfc/rfc8180.txt
20. Brun-Laguna K, Minet P, Watteyne T, Gomes PH. Moving beyond testbeds? lessons (we) learned about connectivity. *IEEE Pervasive Comput*. 2018;17(4):15-27.
21. Gelman A, Carlin JB, Stern HS, Rubin DB. *Bayesian Data Analysis*. Vol. 2. Boca Raton, FL: Chapman & Hall/CRC; 2014.
22. Zamalloa MZ, Krishnamachari B. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans Sen Netw*. 2007;3(2):1-34.
23. Cerpa A, Wong JL, Kuang L, Potkonjak M, Estrin D. Statistical model of lossy links in wireless sensor networks. Paper presented at: IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks; 2005; Boise, ID.
24. Senel M, Chintalapudi K, Lal D, Keshavarzian A, Coyle EJ. A Kalman filter based link quality estimation scheme for wireless sensor networks. Paper presented at: IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference; 2007; Washington, DC.

25. Qin F, Zhang Q, Zhang W, Yang Y, Ding J, Dai X. Link quality estimation in industrial temporal fading channel with augmented Kalman filter. *IEEE Trans Ind Inform*. 2018:1936-1946.
26. Gomes PH, Chen Y, Watteyne T, Krishnamachari B. Insights into frequency diversity from measurements on an indoor low power wireless network testbed. Paper presented at: 2016 IEEE GLOBECOM Workshops (GC Wkshps); 2016; Washington, DC.
27. Cheng RC. Generating beta variates with nonintegral shape parameters. *Commun ACM*. 1978;21(4):317-322.

How to cite this article: Gomes PH, Krishnamachari B. TAMU-RPL: Thompson sampling-based multichannel RPL. *Trans Emerging Tel Tech*. 2019;e3806. https://doi.org/10.1002/ett.3806