Mobility Aware RPL (MARPL): Mobility to RPL on Neighbor Variability

Vinícius de Figueiredo Marques(✉) and Janine Kniess

Programa de Pós-Graduação em Computação Aplicada,
Universidade do Estado de Santa Catarina, Joinville, SC, Brazil
vinicius.marques@edu.udesc.br, janine@udesc.br

Abstract. Low Power and Lossy Network (LLN) is a common type of wireless network in IoT applications. LLN communication patterns usually require an efficient routing protocol. The IPv6 Routing Protocol for Low-Power and Lossy Network (RPL) is considered to be a possible standard routing protocol for LLNs. However, RPL was developed for static networks and node mobility decreases RPL overall performance. These are the aims of the Mobility Aware RPL (MARPL), presented in this paper. MARPL provides a mobility detection mechanism based on neighbor variability. Performance evaluation results obtained by the Cooja Simulator confirm the effectiveness of MARPL regarding DODAG Disconnection prevention, Packet Delivery Rate, Packet Delivery Delay and Overhead when compared to other protocols.

Keywords: Low Power and Lossy Networks · Wireless sensor network · Routing protocol · RPL · Mobility support

1 Introduction

Internet of Things (IoT) is a concept that aims to include wireless connectivity for day-to-day devices in order to enable many forms of smart applications. The motivation for such applications is the analysis of a huge amount of data collect by devices with internet connection. So information could be extracted from these data to enhance decision making and planning [13]. An IoT device is commonly composed by a sensing and wireless communication component. The sensing component is responsible for data collection, while the communication component might differs in terms of radio range and transmission power depending on the IoT application requirements [18]. The main characteristics of radio technologies used by IoT devices are short transmission range and low power consumption, since many IoT devices may be battery powered [18].

Low Power and Lossy Network (LLN) is a common type of network formed by IoT devices [15]. These devices also operates with low range radio technology, such as the IEEE 802.15.4, in order to be energy efficient [7]. The usage of low range radio technology demands a hop-by-hop communication model, which in

R. Miani et al. (Eds.): GPC 2019, LNCS 11484, pp. 59–73, 2019.
https://doi.org/10.1007/978-3-030-19223-5_5

turn demands an efficient routing protocol. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) is a routing protocol for LLNs designed by the IETF group [17]. The motivation for the RPL design was the lack of a proper routing protocol for LLNs. RPL is compatible with IPv6 through the 6LoWPAN adaptation layer [17]. RPL is intended to become a standard routing protocol for data collection LLN applications.

Many IoT application domains, such as smart health and smart city, may have both static and mobile devices in the network [9]. Accordingly, the complexity of the network management is increased. Mobility demands a resilient routing protocol to handle frequent topology changes [19]. A routing protocol for LLN should have efficient mechanisms for rapid mobility detection, so it may diminish packet loss caused by the mobility of devices and minimize disconnection effects. The RPL routing protocol was initially designed for static LLN topologies. Therefore it faces some issues when used in mobile topologies, such as low packet delivery rate. Nevertheless, RPL can be enhanced to become well suited for mobile scenarios as well [6]. Some of the RPL issues are related to the lack of a mobility detection mechanism and efficient preferred parent selection in which mobility is taken into account.

Many routing protocols have been proposed to cope with absence of a mobility support for RPL [1,2,4,11] and [10]. Nevertheless, the majority of them differentiate the nodes as mobile or static only. MARPL can be used in scenarios where all nodes can move around, remaining static for some periods. Also, a mobility monitoring mechanism based on nodes neighbor variability is not exploited by the analysed related work. We defend the idea that neighborhood monitoring may provide an efficient way for the node detect its mobility and consequently, enhance RPL performance in mobile LLN. Thus, in order to increase Packet Delivery Rate while maintaining low overhead, we propose the Mobility Aware RPL (MARPL), a mobility support for the RPL routing protocol based on neighbor variability. MARPL brings the following main contributions:

- A metric related to node mobility called Neighbor Variability;
- A mechanism for mobility detection through the proposed metric, Neighbor Variability;
- A preferred parent unavailability prevention mechanism;
- A Trickle adjustment mechanism to increase the transmission of control messages only when it's necessary to minimize malicious disconnection effects.
- A route selection mechanism that takes node mobility into account.

The road map of this paper is organized as follows. Section 2 shows details about RPL and the issues faced by it when applied to mobile network topologies. Section 3 presents RPL mobility support proposals found in the literature. Section 4 depicts a mobility support for RPL, MARPL, proposed by the authors of this paper. The results of MARPL performance analysis are shown at Sect. 5. Finally, Sect. 6 presents the conclusions and future work.

2 RPL Mobility Issues

RPL was originally developed for static networks. However, mobility support is a requirement for a plenty of IoT application domains such as smart health and smart cities [9]. RPL faces a series of performance challenges when there are mobile nodes in the topology such as: packet loss and frequent disconnections. Nevertheless, RPL can be adapted for a better mobility support [6].

Devices executing RPL can perform three types of roles in a LLN: a root, a router or a host node [17]. A root node receives all data collected inside the LLN. A router and host node are responsible for data collection, but only a router node can forward packets towards the root node (i.e. a root node in the LLN). The RPL topology is based on a Directed Acyclic Graph (DAG). DAG is a graph with no cycles and all its edges are oriented toward one or more root nodes [17].

A Destination-Oriented DAG (DODAG) is a DAG rooted at a single root node. RPL has three control packet: DODAG Information Object (DIO), Destination Advertisement Object (DAO) and a DODAG Information Solicitation (DIS). DIO is used by RPL to construct and maintain a DODAG topology. When a node joins a DODAG, it does so by selecting a neighbor node with the best route towards the root node. In RPL, route quality is assess based on a rank value every node has when joined to a DODAG. This rank is a distance metric that indicates how far a node is from the root node. Therefore, the best route is the route with smaller rank [17]. After joining a DODAG, a node sends a DAO message to the selected neighbor with best rank. This neighbor with the best route towards the root node is called preferred parent in the RPL terminology. DIS is utilized for a node to request DIO messages from its neighbors to assess the possible routes towards the root node.

RPL has a proactive route discovery and topology construction approach through a periodic DIO transmission. The construction of a RPL DODAG is initiated by DODAG root nodes [17]. DIO control packet dissemination in RPL is controlled by the Trickle algorithm [16].

Trickle is an algorithm for DIO dissemination in RPL in a simple, robust and scalable manner [16]. Trickle has two mechanisms to achieve it: (i) when an inconsistency in the network is detected (e.g. loop), Trickle increases the signaling rate of messages as a way of solving the inconsistency. By contrast, Trickle exponentially decreases control message transmissions when the network is stable in order to save node energy; (ii) Trickle has a suppression mechanism in which DIO transmission is suppressed when its content is considered trivial [16].

In RPL, DIO propagation by the Trickle algorithm is configured through three parameters: I_{min}, I_{max} and k. I_{min} specifies the minimum period of time the suppression of DIO transmission can last. I_{max} regulates the maximum period of DIO suppression. k is the redundancy factor that is used to verify if a message can be transmitted at a specific time [16]. Trickle's transmission suppression is adjusted by the variable I. I value is selected randomly in the closed set $[I_{min}, I_{max}]$ and grows exponentially until it reaches I_{max} [16]. In MARPL,

the Trickle adjustment is applied in the variable I, reducing it by half at any moment a mobile node is identified in the neighbor.

The suppression period regulated by the Trickle algorithm influences the performance of the RPL routing protocol in mobile topologies. When the suppression period is too long, nodes might not be able to detect a preferred parent disconnection efficiently since there are less control messages being propagated in the network [6]. Another issue with long periods of DIO suppression is that the DODAG will take longer to update its topology and, therefore, DODAG disconnection effects are aggravated.

It was found in the analysis of related works that most of the issues the RPL routing protocol faces when dealing with mobility are: the lack of efficient mechanisms for DIO and DIS control messages transmission for DODAG disconnection detection; a mechanism for mobility monitoring; and preferred parent selection in which it takes node mobility into consideration.

2.1 Preferred Parent Unavailability Detection

Figure 1a illustrates the RPL default operation mode when dealing with mobile nodes in the network topology. There are three nodes: one mobile and its two neighbor nodes, **A** and **B**. Assuming that mobile node has node **A** as its preferred parent, the mobile node still inside the radio range of node **A** while sending the first data packet. As depicted at Fig. 1a, the mobile node moves outside the range of node **A**'s communication radio. From then on, all data packet sent from the mobile node to its preferred parent, **A**, is lost. This is because RPL does not uses a mechanism for disconnection detection. The mobile node will keep to try

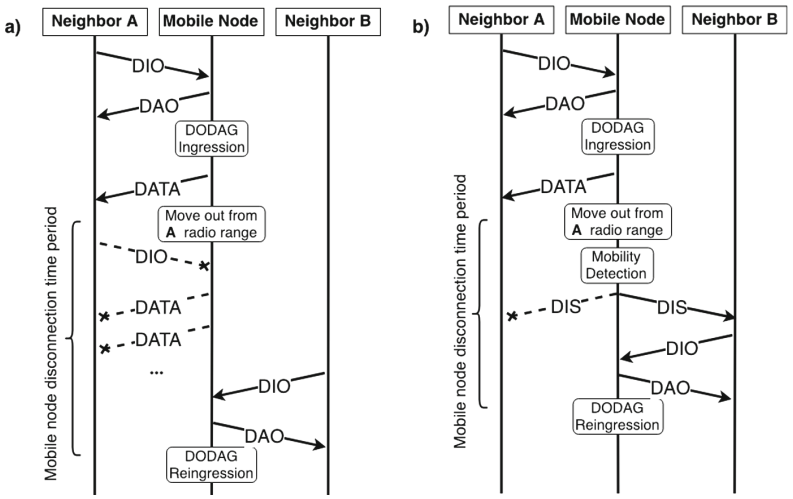


Fig. 1. (a) RPL message exchange without mobility support. (b) MARPL mobility support for RPL.

to send packets to node **A** until it receives a DIO messages from another node in the topology and the rank of this node is lesser than **A**'s rank. Otherwise, the mobile node will still consider node **A** as its preferred parent. As Fig. 1a depicts, the mobile node re-enters the DODAG only when it receives a DIO message from another node, **B**. By the means of simplicity, in this example, it is assumed that the rank of node **B** is lesser then node **A**.

The behavior of the Mobility Aware RPL (MARPL) is illustrated in Fig. 1b. When the mobile node moves out of node **A**'s radio range, MARPL's mobility management of the mobile node detects its movement. The canonical RPL specification does not specify how DIS control messages can be used to detect or avoid preferred parent unavailability. MARPL uses DIS messages when it detects the unavailability of the preferred parent due to mobility. This is performed through sending DIS messages to all its neighbors. The RPL canonical specification states that it might reset the Trickle's timer. Therefore, the mobile node could receive DIO messages from the potential new parents in the neighborhood of the node, see Fig. 1b. Therefore, the mobile node could re-enter the DODAG. It's expected that such mechanism could improve the RPL performance in terms of packet delivery rate. For example, compare the mobile node disconnection time period in Fig. 1b in contrast with Fig. 1a.

The usage of DIS messages to detect or avoid preferred parent unavailability may improve RPL performance in terms of reconnection delay or packet delivery rate. Nevertheless, Trickle adjustment might be necessary to improve RPL performance even further in mobile topologies. As mentioned before, RPL utilizes the Trickle timer for the dissemination of DIO messages. DIO is a RPL control message responsible for the DODAG construction and maintenance. In MARPL, Trickle is adjusted by dividing the suppression time by half when a mobile node is detected in the neighborhood. Consequently, it is expected that it could improve the responsiveness of MARPL in a mobile LLN. MARPL is detailed in Sect. 4.

3 Related Work

As stated at Sect. 2, the main issues regarding RPL when dealing with mobility are the lack of a mobility detection mechanism and link disconnection detection or prevention.

In [2], the authors argued that when a node disconnects from its preferred parent because of mobility, it might wait for too long to receive a DIO message. Therefore, increasing disconnection time and packet loss. In RPL, a node can update its preferred parent by receiving a DIO message from another candidate parent with lower rank [17]. [2] proposes a reverse Trickle algorithm that the DIO suppression time starts short and increases over time. The main rationale behind this idea is that mobile node connects to a preferred parent and it remains connected for a considerable amount of time. A drawback of [2] proposal is that it depends on the existence of static nodes. In contrast, MARPL makes no distinction between mobiles and static nodes, since mobiles nodes can stay static for a period of time.

In [4], the authors proposed a RPL extension named Mobile Compliant RPL (mRPL). mRPL utilizes a handoff mechanism called SmartHop [5]. The authors showed that mRPL enables the exchange of preferred parent efficiently with low overhead and power consumption. mRPL is a proactive preferred parent unavailability prediction mechanism. Therefore, it enables frequently control message exchange in order to assess if the preferred parent still connected. Nevertheless, it's expected that mRPL increases the network overhead in order to perform a fast disconnection identification.

In [11], the authors proposes MoRoRo, a mobility support mechanism for RPL. With MoRoRo, the node mobility can be detected based on packet loss rate by the increase of control message to assess link quality. [6] argues that packet loss is usual in LLN for its lossy links. Therefore, MoRoRo approach increases packet loss and leads to greater overhead. Nevertheless, [11] argues that the utilization of proactive handoff mechanisms in LLN (e.g. such as mRPL) is too aggressive because it generally performs detailed link analysis to detect preferred parent unavailability.

In [1], the authors proposed an enhancement for mRPL through a mobility prediction mechanism. Such mechanism seeks to solve two issues: the high RSSI interference from the environment and the costs of the increased overhead caused by proactive preferred parent unavailability proposal, such as mRPL. [1] mechanism is based on the following assumption: static nodes are required in the topology and their positions are known by the mobile nodes before the network starts to execute. This assumption may not be realistic for every LLN application scenario. Besides that, the authors' proposal has another drawback related to the processing power required by the static nodes in order to process the mobility predicting model.

In [10], the authors proposed a reverse-Trickle timer based on the Received Signal Strength Indication (RSSI) called Dynamic RPL (D-RPL). Every node executing D-RPL keeps track of the RSSI from the last two packet message for every single local-link neighbor node. It could be a control or data packet. Upon reception of a new packet, a node measures the RSSI and compare it to the last measurement from the same neighbor node. If the new RSSI plus a redundancy constant K_{RSSI} is lesser then the last RSSI, the reverse-Trickle timer is executed. The node also sends a local-link multicast DIS to all its neighbors. Otherwise, the default Trickle is executed.

Section 4 presents more details about MARPL, a proposal of mobility support for RPL. D-RPL and mRPL were implemented and compared against MARPL. These protocols were chosen for comparison because it is the most similar to the approach proposed in this paper. Details about the obtained results are show at Sect. 5.

4 Mobility Aware RPL

This section details the Mobility Aware RPL (MARPL) protocol. The design of MARPL encompass a mobility detection and a preferred parent unavailability detection mechanisms. Also, an enhancement to the RPL trickle timer.

As mentioned before, many IoT application domains requires a LLN with both static and mobile nodes. Thus, the routing protocol should be resilient enough to handle constant topology changes caused by node mobility. MARPL is compatible with the canonical RPL. Therefore, both MARPL and RPL can coexist in the same LLN. MARPL is composed by three mechanisms: (i) **mobility detection through the metric Neighbor Variability (γ)**; (ii) **preferred parent unavailability detection** and; (iii) **Trickle adjustment**. Algorithm 1 demonstrates the MARPL mechanisms.

Algorithm 1. MARPL Protocol

```
1:  procedure MOBILITY_MONITORING
2:      start T_monitoring
3:      if received a packet then                          ▷ data or control packet
4:          update sender IP, γ and RSSI in the neighbor table   ▷ if control packet
5:          if packet is a DIS or DAO from child node then
6:              TRICKLE_ADJUSTMENT
7:          if packet is DIO then
8:              neighbor_new_rank ← α * neighbor_old_rank + β * γ
9:      if T_monitoring expires then
10:         if max{var{Δp_i}_{i=1}^y} > K_γ then   ▷ if there's a greater variance then K_γ
11:             K_γ = max{var{Δp_i}_{i=1}^y}          ▷ update K_γ with the greatest var
12:         γ ← var{Δp_i}_{i=1}^y / K_γ
13:         if received no packet from the preferred parent then   ▷ data or control
14:             if γ > 0 then
15:                 send DIS to neighbors
16:         restart T_monitoring
17: procedure TRICKLE_ADJUSTMENT
18:     γ_packet ← γ from control packet                   ▷ DIO, DIS or DAO
19:     if γ_packet > γ then
20:         I ← I/2
21:         if I < I_min then
22:             I ← I_min
```

As depicted in Algorithm 1, when a node first enters a DODAG, MARPL starts the $T_{monitoring}$ timer (line 2) in order to monitor the node mobility. $T_{monitoring}$ operates based on the rate of sensor data generation (i.e. the frequency of measurements made by the sensor). The frequency of sensor data generation may vary depending on the application characteristics.

At every data or control packet reception, MARPL updates its neighbor table with the packet sender IP, the proposed metric Neighbor Variability (γ) and the Received Signal Strength Indication (RSSI) (line 4). If the received packet is a DIS or DAO control message, MARPL analyses if it's necessary to adjust the Trickle timer (line 6). If the received packet is a DIO message, MARPL updates the neighbor's rank with its γ (line 8) by the Eq. 1.

$$neighbor_{new\_rank} = \alpha * neighbor_{old\_rank} + \beta * \gamma \quad (1)$$

In RPL, a preferred parent is selected by its rank value. The rank value is related by the distance of the preferred parent candidate to the root node. A rank value is calculated by a RPL objective function. MARPL updates the rank of a candidate parent with its γ value since γ is a metric related to the node mobility. The metric γ is updated by Eq. 2. Thus, it's expected that by using γ in the preferred parent selection, static nodes will have greater probability to be selected. Equation 1 shows how the rank value is updated. MARPL utilizes two weight parameters: α for the rank calculated by the RPL objective function and β for the Neighbor Variability metric.

The metric γ is derived by the variance of all the positive RSSI variations (i.e. $\Delta p_i \mid \Delta p_i > 0$) from every neighbor node $i \mid i \subset [1, y]$, y being all neighbors with two consecutive RSSI measurements, over a threshold K_γ. The RSSI (p) variation for every neighbor (i) is calculated by Eq. 3. K_γ is the maximum variance ever calculated by the node during its execution as depicted in Eq. 4 (Algorithm 1, lines 10 and 11).

The first step for Trickle's adjustment (Algorithm 1 line 17) is to read metric γ from the control packet (i.e. identified by γ_{packet} at line 18). If γ_{packet} is greater than the node's γ (line 19), the Trickle's variable I is reduced by half (line 20). It's important to make sure that $I >= I_{min}$ (line 21 and 22), since the following is a requirement for Trickle to work: $I \subset [I_{min}, I_{max}]$.

The MARPL Trickle adjustment mechanism is inspired by [10] and is executed in order to temporary increase DIO transmissions. It's expected that such increase might improve MARPL overall performance in terms of packet delivery rate through the prevention of further disconnections.

At every $T_{monitoring}$ expiration (Algorithm 1, line 9), MARPL updates the proposed metric Neighbor Variability (γ) (line 12). Using γ, a sensor node can identify its mobility. $T_{monitoring}$ execution time is adjusted by the frequency of sensor data generation. Consequently, it's expected that $T_{monitoring}$ execution time is sufficient to enable a node to received packets from all of its neighbors.

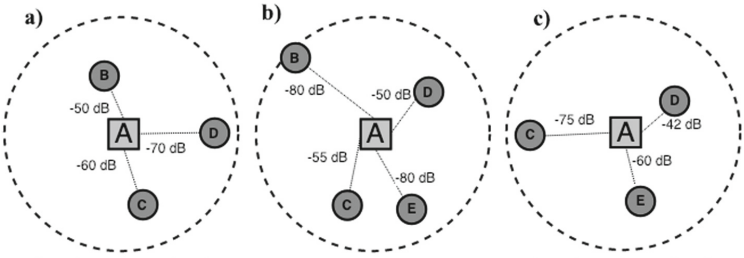
$$\gamma = \frac{var\{\Delta p_i\}_{i=1}^{y}}{K_\gamma} \mid y > 0, \Delta p_i > 0, K_\gamma > 0 \quad (2)$$

$$\Delta p_i = ||p_{i-1}|| - ||p_i|| \quad (3)$$

$$K_\gamma = max\{var\{\Delta p_i\}_{i=1}^{y}\} \quad (4)$$

For a better understanding of the rationale behind metric γ calculation, consider Fig. 2, an example of γ calculation. There are three different $T_{monitoring}$ periods: Fig. 2a, b and c. As can be seen in Fig. 2a, the mobile node **A** initially has three neighbors. **A**'s γ is set to 0 since there's no entries in its neighbor table with two consecutive RSSI measurements (p_{i-1} and p_i). Since there hasn't been calculated any variance yet, K_γ has no value. In Fig. 2b, node **A** moved and there are RSSI variations of neighbor **B**, **C** and **D**, besides the new neighbor **E**. The

variations of neighbors **C** and **D** are positive (5 and 20 respectively). Therefore, γ can be calculated. Since it's the first time the variance (var) is calculated, K_γ will be set with the initial variance value. In Fig. 2b, the variance and K_γ have the same, 56, since it's the first time a RSSI variance is calculated. Thus, γ is updated to 1, $\frac{var}{K_\gamma} = 1$. In Fig. 2c, node **A** moved again. This time, it has three neighbors with RSSI variation. Nevertheless, only two of them (**D** and **E**) has positive variation (8 and 20 respectively). At Fig. 2c the variance is 36. Since $36 < 56$, K_γ is not updated and γ is updated to 0.64. It's possible to assess that at Fig. 2c, **A** neighbor varied less then at Fig. 2b. Thus, MARPL assumes that the mobile node **A** moved less at 2c.



Node	pi-1	pi	Δp
B	-	-50 dB	-
C	-	-60 dB	-
D	-	-70 dB	-
var			-
γ			0
K_γ			-

Node	pi-1	pi	Δp
B	-50 dB	-80 dB	-30
C	-60 dB	-55 dB	5
D	-70 dB	-50 dB	20
E	-	-80 dB	-
var			56
γ			1
K_γ			56

Node	pi-1	pi	Δp
C	-55 dB	-75 dB	-20
D	-50 dB	-42 dB	8
E	-80 dB	-60 dB	20
var			36
γ			0.64
K_γ			56

Fig. 2. MARPL γ calculation example.

Link disconnection prevention is critical in topologies with mobile nodes since disconnections will be frequent. RPL does not specify any preferred parent unavailability mechanism [17]. The RPL specification suggests the use of an external mechanism for this task [17]. Hence, if no packet from the preferred parent was received after $T_{monitoring}$ expiration (Algorithm 1, line 13) and node's $\gamma > 0$ (line 14), MARPL sends a DIS message to all the neighbor sensor nodes (line 15) to assess information about the available candidate parents. After that, $T_{monitoring}$ is started again (line 16).

Section 5 presents a performance analysis of MARPL against the canonical RPL specification [17], mRPL [4] and D-RPL [10].

5 Simulation Results and Analysis

This section presents a performance analysis of MARPL compared to the protocols, RPL [17], mRPL [4] and D-RPL [10]. mRPL and D-RPL where chosen

by the following reasons: mRPL [4] is a proactive RPL mobility support proposal. In other words, it tries to identify as fast as possible the preferred parent unavailability to re-establish DODAG connection. mRPL aims to achieve this by making the sensor node to send periodic DIS messages to its preferred parent, while monitoring the reception of DIO messages in return. In contrast, MARPL doesn't try to monitor link disconnections since it utilizes Trickle's adjustments based on node mobility to diminish disconnections. D-RPL, proposed by [10], utilizes a reverse Trickle adjustment so that each time a sensor node identifies a RSSI variation, the Trickle suppression time is reduced by half. Differently, MARPL's reverse Trickle adjustment stands on the node's neighbor variability monitoring.

A total of 20 simulations were executed using the Cooja simulator [14] for each routing protocol. Cooja is a simulation tool for the Contiki Operational System [3]. Contiki was designed to execute in low powered devices commonly utilized in LLNs. The Contiki LLN networking stack is compatible to 6LoWPAN and the IEEE 802.15.4 radio technology.

Table 1. Cooja simulation parameters.

Parameters	Values
Number of mobile nodes	50
Number of root nodes	1, 2 and 3
Radio	CC2420 [8]
Simulation time	10 min
Node placement	Random
Mobility model	Steady-State Random Waypoint
Maximum node velocity	3 m/s
Maximum pause time	40 and 20 s
Data generation rate	8 s
Transmission medium	*Unit Disk Graph Medium* (UDGM)
Radio transmission range	50 m
Simulation area	300 m x 300 m
$T_{monitoring}$	8 s
α	1
β	1

Table 1 presents the parameters used in the simulations. The Steady-State Random Waypoint [12] mobility model was used to simulate node mobility. The Steady-State Random Waypoint model extends the Random Waypoint model to enable a time period of pause for the node [12]. All sensor nodes are mobile, but they can remain static for a period of time. Every node is randomly distributed in the simulation area and a new simulation seed is generated at every execution.

We simulated a LLN with a total of 50 sensor nodes within an area of 300 m of width and 300 of height. The number of root nodes varies from 1 up to 3 the time of pause was set in 40 s. We also simulated a scenario with 3 root nodes and 20 s of pause time to asses how the protocols would perform when dealing with more mobility in the topology.

The simulation analysis was performed in terms of: (i) **Packet Delivery Rate (PDR):** that means the rate of received data packets over sent data packets; (ii) **Packet Delivery Delay (PDD):** the time needed for a data packet to travel from the router to the root node; (iii) **DODAG Disconnections:** the number of DODAG disconnections caused by the node mobility. This metric enables to evaluate how good a protocol could prevent disconnections; (iv) **DODAG Reconnection Delay**: the time needed for a DODAG disconnection to be solved; (v) **Overhead:** the rate of control packets of a routing protocol over the total of control packets transmitted to the network.

We present simulation results varying the number of root nodes for every evaluated RPL based routing protocol (1 *Root*, 2 *Roots* and 3 *Roots* in the plots). Such simulations were executed with maximum node velocity of 3 m/s and pause time of 40 s. We also simulate LLN scenarios with a maximum pause time of 20 s and 3 Roots. These cenarios are presented in figures as 3 *Roots* (∗). Reducing the time of pause increases the mobility in the LLN.

Figure 3 depicts the simulations results in terms of Packet Delivery Rate in milliseconds. In all simulated scenarios, MARPL presented the best performance in terms of PDR, 5.98% with 1 root node, 11.24% with 2 root nodes, 16.63% with 3 root nodes and 18.03% with 3 root nodes and maximum pause time of 20 s. As can be seen in Fig. 3, the PDR for all the protocol increases as the number of root nodes in the topology also increases when the time of pause is 40 s. When time of pause is 20 s, only MARPL presented a increase in PDR (3 *Roots* (∗) in the plots). RPL, mRPL and D-RPL had a decrease in PDR when there's a increase of node mobility. This result is explained by the MARPL usage of the metric Neighbor Variability for Trickle adjustments and route selection. By doing so, MARPL enables the selection of routes with less mobility, therefore, increasing PDR even in more mobile LLNs. MARPL had also better performance in terms of delay with the smallest delays in almost all the evaluated scenarios depicted in Fig. 4. (36045 ms, 29057 ms, 26759 ms and 26912). It's also noticeable that packet delay decreases as the number of root node increases for all the evaluated routing protocols. Except for mRPL, since there's no statistical difference between the results of 1 Root, 2 Roots and 3 Roots.

Figure 5 depicts the results in terms of overhead. mRPL, D-RPL and MARPL presented greater overhead when compared with RPL. Nevertheless, no significant statistical difference was found in terms of overhead between the usage of 1, 2 or 3 root nodes in the LLN topology for any of the evaluated routing protocol. Among the RPL mobility support proposals, MARPL had the smallest average overhead (19.95% for 1 Root, 18.89% for 2 Roots, 18.81% for 3 Roots and 18.51% for 3 Roots and time of pause as 20 s). This result is justified because MARPL's mechanism of Trickle adjustment based on node mobility. Therefore, MARPL only increases control packet transmission when it's necessary to prevent link disconnections.

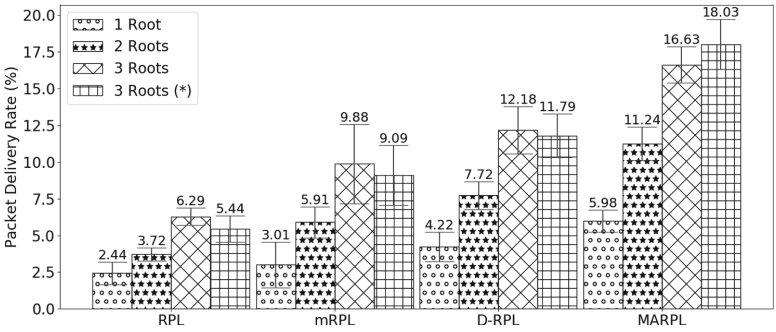


Fig. 3. Packet delivery rate.

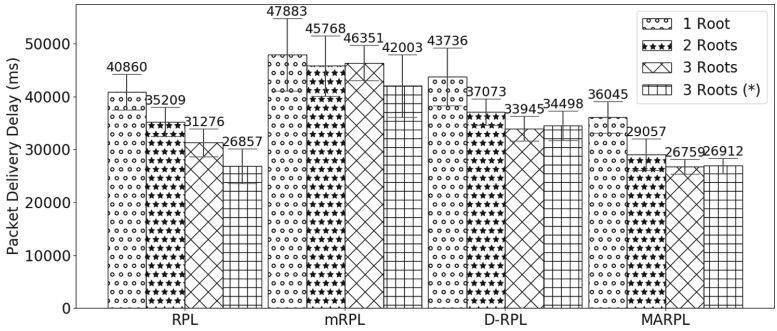


Fig. 4. Packet delivery delay.

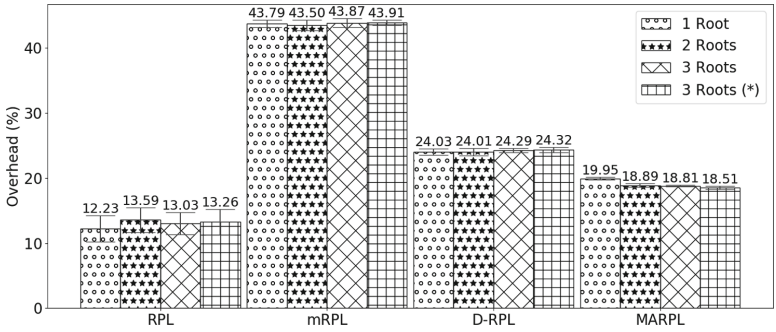


Fig. 5. Overhead.

By analyzing Figs. 6 and 7, we conclude that there is a relationship of DODAG Disconnections and DODAG Reconnection Delays. It's noticeable that there's a inverse correlation between the number of DODAG Disconnections and DODAG Reconnection Delays. It's possible to analyze the capacity of a routing protocol to prevent link disconnections by the number of

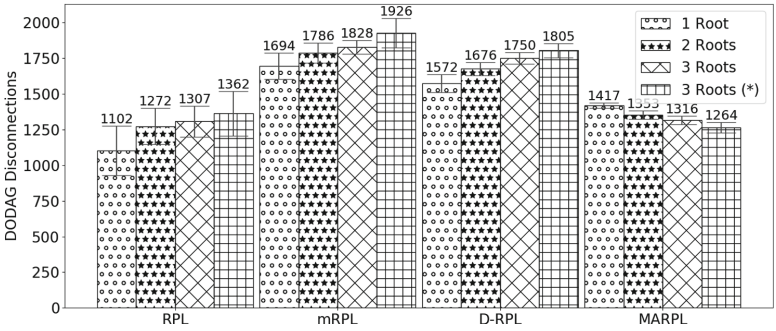


Fig. 6. DODAG disconnections.

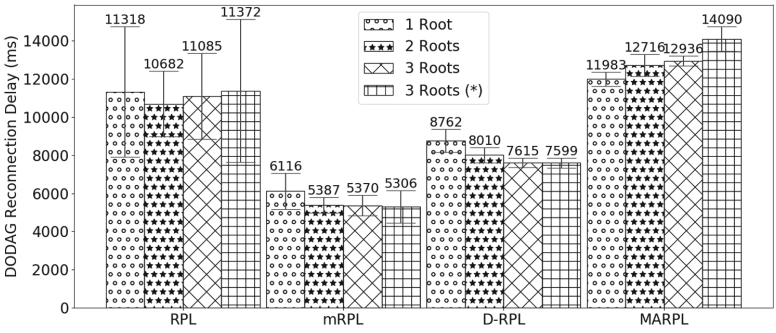


Fig. 7. DODAG reconnection delay.

DODAG Disconnections. It's also possible to analyze how fast a routing protocol is capable to detect a link disconnection by DODAG Reconnection Delays. Therefore, we can compare two different approaches to solve link disconnections: either to prevent link disconnections or to detect it as fast as possible. mRPL and D-RPL tries to detect link disconnection as soon as it happens. Therefore, both had the smallest DODAG Reconnection Delay, Fig. 7. In contrast, they had the greatest number DODAG Disconnections. An important result is the fact that while RPL, mRPL and D-RPL increases the disconnections when the number of root nodes also increases. On the contrary, MARPL decreases it, see Fig. 6. Another relevant result is related to the simulation with 3 *Roots* $(*)$ and 20 s of pause. Since, when there's more mobility in the topology, RPL, mRPL and D-RPL increase the number of DODAG Disconnections, MARPL presented lesser disconnections when dealing with a LLN with increased mobility. Therefore, we conclude that MARPL is better suited to mobile LLNs. MARPL's link disconnection prevention approach might be a better solution since it had better PDR and delay while maintaining lower overhead when comparing it against other evaluated proposals.

6 Conclusion

RPL is a routing protocol for Low Power and Lossy Networks (LLNs), a common type of network formed by Internet of Things (IoT) applications. Mobility support is a requirement for a wide range of IoT applications. Regardless the fact that RPL was initially intended for static LLNs (i.e. LLNs composed only by static devices) it can be enhanced to include mobility support capabilities. Node mobility increases link disconnections and consequently increases packet loss. RPL faces a series of issues when dealing with mobile nodes. Natively, RPL doesn't have a way of detecting when a node is moving, nor a way of identifying when a link with its preferred parent is unavailable. This paper discussed the issues RPL faces when it deals with mobile nodes and approaches to solve them.

This paper presented the Mobility Aware RPL (MARPL). MARPL intends to add mobility support to the RPL routing protocol. MARPL is composed by two mechanisms: (i) mobility detection and (ii) control packet transmission adjustment. This paper presents the results obtained by simulations executed in the Cooja simulator. This paper also presents a performance analysis of MARPL proposal, the canonical RPL [17] and more two proposals found in the literature: mRPL [4] and D-RPL [10].

The results indicates that MARPL has better performance in relation to Overhead, DODAG Disconnection prevention, Packet Delivery Rate and Packet Delivery Delay. MARPL prevents more disconnections when comparing it against RPL, mRPL and D-RPL. By the simulation analysis, we concluded that the prevention of DODAG disconnection might be inversely proportional to the delay of DODAG reconnection. It means that, efforts to prevent link disconnections may increase reconnection delay. Since MARPL presented better PDR in comparison with other proposals, we believe that disconnection prevention is more important than disconnection detection to face frequent link disconnections caused by node mobility.

It was concluded that further studies can be done to improve MARPL transmission delay while preserving good PDR and Overhead results. Other further improvements are in terms of diminish the DODAG Reconnection Delay while keeping a small number of disconnections. MARPL outperforms all the three proposals in terms of Packet Delivery Delay. Nevertheless, an overhead increase is expected because of the node mobility management requires a high number of control messages since the topology will need to be updated more frequently. We believe that it's necessary for the RPL mobility management to only increase such control message transmissions in specific moments. MARPL only increases control message transmission when a node has mobile nodes connected to it in its neighborhood. It causes the DODAG topology to be updated when needed only. Thereby, preventing disconnections from happening.

References

1. Bouaziz, M., Rachedi, A., Belghith, A.: EKF-MRPL advanced mobility support routing protocol for internet of mobile things: movement prediction approach. Future Gener. Comput. Syst. **93**, 19–24 (2017)
2. Cobarzan, C., Montavont, J., Noel, T.: Analysis and performance evaluation of RPL under mobility. In: Computers and Communication (ISCC), pp. 1–6 (2014)
3. Dunkels, A., et al.: The Contiki OS: the operating system for the Internet of Things (2011). http://www.contikios.org. Accessed Dec 2018
4. Fotouhi, H., Moreira, D., Alves, M.: mRPL: boosting mobility in the Internet of Things. Ad Hoc Netw. **26**, 17–35 (2015)
5. Fotouhi, Hossein, Zuniga, Marco, Alves, Mário, Koubaa, Anis, Marrón, Pedro: Smart-HOP: a reliable handoff mechanism for mobile wireless sensor networks. In: Picco, Gian Pietro, Heinzelman, Wendi (eds.) EWSN 2012. LNCS, vol. 7158, pp. 131–146. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28169-3_9
6. Gara, F., Saad, L.B., Hamida, E.B., Tourancheau, B., Ayed, R.B.: An adaptive timer for RPL to handle mobility in wireless sensor networks. In: 2016 International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 678–683, Paphos, Cyprus (2016)
7. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
8. Instruments, T.: CC2420 datasheet. Reference SWRS041B, pp. 1–93, December 2007. http://www.ti.com/product/CC2420
9. Iova, O., Picco, P., Istomin, T., Kiraly, C.: RPL: the routing standard for the Internet of Things... or is it? IEEE Commun. Mag. **54**(12), 16–22 (2016)
10. Kharrufa, H., Al-Kashoash, H., Al-Nidawi, Y., Mosquera, M.Q., Kemp, A.H.: Dynamic RPL for multi-hop routing in IoT applications. In: Wireless On-demand Network Systems and Services (WONS), pp. 100–103. IEEE, Jackson (2017)
11. Ko, J., Chang, M.: MoMoRo providing mobility support for low-power wireless applications. IEEE Syst. J. **9**(2), 585–594 (2015). https://doi.org/10.1109/JSYST.2014.2299592
12. Navidi, W., Camp, T.: Stationary distributions for the random waypoint mobility model. IEEE Trans. Mobile Comput. **1**, 99–108 (2004)
13. Oppitz, M., Tomsu, P.: Internet of Things. Inventing the Cloud Century, pp. 435–469. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-61161-7_16
14. Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., Voigt, T.: Cross-level sensor network simulation with COOJA. In: 2006 31st IEEE Conference on Local Computer Networks, pp. 641–648. IEEE, Tampa (2006)
15. Paul, P.V., Saraswathi, R.: The Internet of Things–a comprehensive survey. In: 2017 International Conference on Computation of Power, Energy Information and Communication (ICCPEIC), pp. 421–426. IEEE, Melmaruvathur, India (2017)
16. The trickle algorithm: RFC 6206. No. RFC 6206, pp. 1–13 (2011)
17. RFC 6550: RPL: IPv6 routing protocol for low-power and lossy networks. No. RFC 6550, pp. 1–157 (2012)
18. Sethi, P., Sarangi, S.R.: Internet of Things: architectures, protocols, and applications. J. Electr. Comput. Eng. **2017**, 1–26 (2017)
19. Zhao, M., Kumar, A., Chong, P.H.J., Lu, R.: A comprehensive study of RPL and P2P-RPL routing protocols: implementation, challenges and opportunities. Peer Peer Netw. Appl. **10**(5), 1232–1256 (2017)