Efficient Routing Primitives for Low-power and Lossy Networks in Internet of Things

Baraq Ghaleb

A Thesis submitted in partial fulfilment of the requirements of Edinburgh Napier University, for the award of Doctor of Philosophy

June 2019

ABSTRACT

At the heart of the Internet of Things (IoTs) are the Low-power and Lossy networks (LLNs), a collection of interconnected battery-operated and resource-constrained tiny devices that enable the realization of a wide range of applications in multiple domains. For an efficient operation, such networks require the design of efficient protocols especially at the network layer of their communication stack. In this regards, the Routing Protocol for LLNs (RPL) has been developed and standardised by the IETF to fulfil the routing requirements in such networks. Proven efficient in tackling some major issues, RPL is still far from being optimal in addressing several other routing gaps in the context of LLNs. For instance, the RPL standard lacks in a scalable routing mechanism in the applications that require bidirectional communication. In addition, its routing maintenance mechanism suffers from relatively slow convergence time, limiting the applicability of the protocol in time-critical applications, and a high risk of incorrect configurations of its parameters, risking the creation of sub-optimal routes. Furthermore, RPL lacks in a fair load-distribution mechanism which may harm both energy and reliability of its networks. Motivated by the above-mentioned issues, this thesis aimed at overcoming the RPL's weaknesses by developing more efficient routing solutions, paving the way towards successful deployments and operations of the LLNs at different scales. Hence, to tackle the inefficiency of RPL's routing maintenance operations, a new routing maintenance algorithm, namely, Drizzle, has been developed characterized by an adaptive, robust and configurable nature that boosts the applicability of RPL in several applications. To address the scalability problem, a new downward routing solution has been developed rendering RPL more efficient in large-scale networks. Finally, a load-balancing objective function for RPL has been proposed that enhances both the energy efficiency and reliability of LLNs. The efficiency of the proposed solutions has been validated through extensive simulation experiments under different scenarios and operation conditions demonstrating significant performance enhancements in terms of convergence time, scalability, reliability, and power consumption.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
6TiSCH	IPv6 over the TSCH mode of IEEE 802.15.4e
6top	6TiSCH Operation sublayer
AODV	Ad hoc On-Demand Distance Vector (AODV)
ARSSI	Average Received Signal Strength Indicator
BDI	Battery Discharge Index
BLE	Bluetooth Low Energy
CH_{IP}	Child IP Address
CH_{List}	Children List
CSMA	Carrier-Sense Multiple Access
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
CTP	Collection Tree Protocol
DAG	Directed Acyclic Graph
DAO	Destination Advertisement Object
DAO-ACK	Destination Advertisement Object Acknowledgment
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination Oriented Directed Acyclic Graph
DSR	Dynamic Source Routing
DualMOP-RPL	Dual Mode Of Operation RPL
ELT	Expected Lifetime
ETX	Expected Transmission Count
HC	Hop Count
HBH	IPv6 Hop-by-Hob Extension Header
Hydro	Hybrid Routing Protocol for LLNs

ICMPv6	Internet Control Message Protocol version 6
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IPv6	Internet Protocol version 6
L2AM	Lifetime and Latency Aggregateable Metric
LBPLAIN	Load-Balancing Plain
LBR	LLNs Border Router
LBRPL.	Leaf-Based RPL
LBS	Load-Balancing with Scheduling
LBSR	Load-Balancing with Scheduling and Resetting
LC	Link Congestion
LLNs	Low-power and Lossy Networks
LOADng	Lightweight On-demand Ad hoc Distance vector routing protocol – Next Generation
LoWPAN	Low Power Wireless Personal Area Network
MAC	Media Access Control
MERPL	Memory-Efficient RPL
MP2P	MultiPoint-to-Point
MPL	Multicast Protocol for Low-power and Lossy Networks
MRHOF	Minimum Rank with Hysteresis Objective Function
MTU	Maximum Transmission Unit
NC	Node Congestion
NoCH	Number of Children
OF	Objective Function
OF0	Objective Function Zero
OF-FL	Fuzzy-Logic Objective Function
OLSR	Optimized Link State Routing Protocol
OMC-RPL	Optimized Multi-Class RPL
Opt-Trickle	Optimized-Trickle
OSPF	Open Shortest Path First

PP	Preferred Parent
Px	Candidate Parent x
P2P	Point-to-Point
P2MP	Point-to-MultiPoint
PD	Propagation Delay
PFI	Packet Forwarding Indication
PLC	Power-line Communication
PRR	Packet Reception Ratio
PSO	Particle Swarm Optimization
QoS	Quality of Service
RDC	Radio Duty Cycling
RE	Residual Energy
RER	Residual Energy Ratio
ROLL	Routing Over Low-power and Lossy networks
RPL	IPv6 Routing Protocol for Low-power and Lossy networks
RTT	Round-Trip Time
SCAOF	Scalable Context-Aware Objective Function
SI	Stability Index
SPRR	Smoothed Packet Reception Ratio
SRNP	Smoothed Required Number of Packet retransmissions
TDMA	Time Division Multiple Access
TSCH	Time-slotted Channel Hopping
UDGM	Unit Disk Graph Medium
U-LLNs	Urban Low-power and Lossy Networks

ACKNOWLEDGMENT

First and foremost, I would like to dedicate this thesis to the memory of my beloved father, Ahmed Alabarah, who couldn't wait to see his dream doctor son. Your wisdom, principles and thoughts were the light for me whenever I ran into a trouble spot. You are gone, but your continuous guidance, encouragement and inspiration have made this journey possible and pushed me to achieve such a great project in the academic arena.

I would like also to thank and dedicate this work to my mother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without her support.

Furthermore, I would like to thank and acknowledge my director of studies Professor Ahmed Al-Dubai, who was always available whenever I had a question about my research or writing. I thank you for your understanding, wisdom, patience, enthusiasm, and encouragement and for pushing me further than I thought I could go.

I am also thankful to my supervisor Dr. Elias Ekonomou for his very valuable comments and consistent guidance throughout the journey of this work.

I offer my regards and blessings to my beloved wife Amatalwahab Alabarah whose unconditional encouragement and support made it possible for me to finish this project successfully.

Finally, I wish to express my heartfelt thanks and love to my children: Ahmed, Nezar and Basheer for coping with the undue paternal deprivation during the course of my study.

DECLARATION

I, Baraq Ghaleb, confirm that this thesis submitted for assessment is my own work and is expressed in my own words. Any uses made within it of the works of other authors in any form e.g., ideas, equations, figures, text, tables, programs, etc. are properly acknowledged and referenced.

Name: Baraq Ghaleb

Matriculation Number: [redacted]

Signature: Date: 10/January/2019

LIST OF PUBLICATIONS

The following is a list of publications made while working on this thesis.

Patents Publications

1- **B. Ghaleb** and A. Al-Dubai, "A leaf-based Routing Mechanisms for Low-Power and Lossy Networks", Provisional Patent, 2017.

Peer-reviewed Journal Publications

1. B. Ghaleb et al., "A Survey of Limitations and Enhancements of the IPv6 Routing Protocol for Low-Power and Lossy Networks: A Focus on Core Operations," in IEEE Communications Surveys & Tutorials, vol. 21, no. 2, pp. 1607-1635, Second quarter 2019.
2. B. Ghaleb, A. Y. Al-Dubai, E. Ekonomou, I. Romdhani, Y. Nasser and A. Boukerche, "A Novel Adaptive and Efficient Routing Update Scheme for Low-Power Lossy Networks in IoT," in IEEE Internet of Things Journal, vol. 5, no. 6, pp. 5177-5189, Dec. 2018.
3. B. Ghaleb, A. Al-Dubai, E. Ekonomou, M. Qasem, I. Romdhani and L. Mackenzie, "Addressing the DAO Insider Attack in RPL's Internet of Things Networks," in IEEE Communications Letters, vol. 23, no. 1, pp. 68-71, Jan. 2019

Peer-reviewed Conference, and Workshop Publications:

1. **B. Ghaleb,** A. Al-Dubai, E. Ekonomou, W. Gharibi, L. Mackenzie, and M. Bani Khalaf, "A New Load-Balancing Aware Objective Function for RPL's IoT Networks", In the proceedings of the 20th IEEE International Conferences on High Performance Computing and Communications (IEEE HPCC), June 2018.
2. **B. Ghaleb,** A. Al-Dubai, I. Romdha, Y. Nasser and A. Boukerche, "Drizzle: Adaptive and fair route maintenance algorithm for Low-power and Lossy Networks in IoT," 2017 IEEE International Conference on Communications (IEEE ICC), Paris, May 2017, pp. 1-6**.**
3. **B. Ghaleb,** A. Al-Dubai, E. Ekonomou and I. Wadhaj, "A new enhanced RPL based routing for Internet of Things," 2017 IEEE International Conference on Communications Workshops (IEEE ICC Workshops), Paris, May 2017, pp. 595-600.
4. **B. Ghaleb**, A. Al-Dubai, E. Ekonomou, B. Paechter and M. Qasem, "Trickle-Plus: Elastic Trickle algorithm for Low-power networks and Internet of Things," 2016 IEEE Wireless Communications and Networking Conference Workshops (IEEE WCNC), Doha, 2016, pp. 103-108.
5. **B. Ghaleb**, A. Al-Dubai and E. Ekonomou, "E-Trickle: Enhanced Trickle Algorithm for Low-Power and Lossy Networks," 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, 2015, pp. 1123-112.

CHAPTER 1: INTRODUCTION

The Internet of Things (IoT) is an ever-growing communication paradigm where billions of "things" like sensors, actuators, home appliances, smart-phones, vehicles, wearable devices and people are connected together via the Internet enabling them to interact with each other and exchange data [1][2]. It is predicted that there will be billions of IoT smart objects connected to the Internet in the near future generating more than 45% of the entire internet traffic [2][3]. Hence, IoT would have a crucial role on the quality of our life by opening the door for a plethora of new applications on different scales, from smart home to smart cities including, but are not limited to, home networks, building automation, industrial automation, environmental monitoring, healthcare reporting, smart metering and logistics [2] [4]. For instance, smart-homes will enable their residents to automatically open their garage when reaching home, prepare their coffee, control climate control systems, TVs, and other appliances.

One of the main pillars of the IoT is the *Low-power and Lossy Networks* (LLNs), a collection of interconnected embedded devices, such as sensor nodes, typically characterized by constraints on both node resources and underlying communication technologies [5][6][7][8][9]. In fact, due to their limited characteristics, LLNs were considered unsuitable for IP connectivity and some sort of translation gateway solutions were developed to connect the LLNs to the IP world. However, the lack of scalability, and flexibility in addition to the inherent complexity of gateway solutions necessitated the migration of the LLNs to IP [4]. One of the prominent challenges that appears as a result of LLNs' movement from being an isolated system to being an essential part of the IoT is the need to smoothly integrate these networks to the IP-based IoT world [4]. In particular, this requires the support of IP-based networking over constrained communication technologies such as the IEEE 802.15.4 standard [4][5][6][7]. The

concern here is that IP was not developed with such LLN limitations in mind. For instance, while IEEE 802.15.4 can support a maximum transmission unit (MTU) of 127 bytes, the recent version of IP standard (IPv6) requires a minimum datagram size of 1280 bytes, approximately ten times more than what IEEE 802.15.4 can support [8]. To address this gap, the Internet Engineering Task Force (IETF) (the Internet protocols standardization body) commissioned a working group named the IPv6 over Low Power Wireless Personal Area Network (LoWPAN). The LoWPAN working group achieved its task in 2007 by introducing the IPv6 over Low-power Wireless Personal Area Network (6LoWPAN) standard [6].

Another challenge faced by such networks was the lack of a standardized and efficient routing protocol [10]. Hence, soon after the introduction of 6LoWPAN, the IETF chartered the Routing Over Low power and Lossy networks (ROLL) working group to design such a solution. The ROLL working group recognized the fact that LLNs are constrained in several aspects, have special characteristics and will run on a range of applications with conflicting routing requirements [9][10][11]. For instance, LLN routers typically operate with constraints on processing power, memory, and energy (battery power) while its communication system is subjected to high packet loss, frame size limitations, instability, low data rates, short communication ranges and dynamically changing network topologies [9][10][11]. In addition, they are interconnected by a variety of links, such as Low Power Wi-Fi, Bluetooth, or wired Power-line Communication (PLC) links [10]. Furthermore, the traffic patterns and reporting models vary widely in such networks. Typical traffic patterns are not simply unicast flows (e.g., in some cases most if not all traffic can be point-to-multipoint) [12][13][14][15]. Such unique characteristics and limitations make the development of efficient routing solutions for LLNs difficult, a task made still more arduous by the potential large-scale deployments of such networks, anticipated to comprise thousands of nodes [14][15]. Taking that into perspective, the ROLL working group started this process by firstly specifying various routing requirements

for four envisioned application categories: home-automation networks [12][16], building-automation networks[13][16], industrial environment [14], and Urban LLNs (U-LLNs) [15][17]. Secondly, the group moved into evaluating whether the conventional ad-hoc routing protocols, such as the Dynamic Source Routing (DSR)[18][19] , the Ad hoc On-Demand Distance Vector (AODV) [20], the Optimized Link-state (OLSR) [21][22], and the Open Shortest Path First (OSPF) [23] can satisfy the specified requirements. However, they found that these protocols are too inefficient for satisfying requirements of overhead, power, reliability and latency [10], thus the group moved on to develop new routing protocols for LLNs. In fact, various efforts have been made by the IETF Routing Over LLNs (ROLL) working group to address such an issue. The Collection Tree Protocol (CTP) [24], the Hybrid Routing Protocol for LLNs (Hydro) [25] and the Lightweight On-demand Ad hoc Distance vector routing protocol – Next generation (LOADng) [26] were some of the research community attempts to devise an efficient routing protocol for LLNs. The final result of the ROLL efforts is the 2012 specification of the distance-vector Routing Protocol for Low Power and Lossy networks (RPL) [27][28][29][30] (a detailed overview of RPL is provided in Chapter 3). In addition, the ROLL working group has specified a routing primitive within RPL, namely, the Trickle algorithm [31][32] to regulate the emission of control traffic messages among neighboring nodes.

1.1 MOTIVATION

As mentioned earlier, the LLNs play a crucial role in the realization of the IoT paradigm opening the door for a wide spectrum of services and applications that would make our life easier and more flexible. However, the successful deployment of LLNs requires addressing several issues and challenges that may degrade their perceived performance. On top of these challenges is devising efficient routing mechanisms that consider the unique characteristic and routing requirements of such kind of constrained networks [10]. Hence, many efforts have

been made by standardization bodies to specify efficient routing protocols for the LLNs resulting in specifying the IPv6 Routing Protocol for LLNs (RPL) as the de-facto standard. However, since the introduction of RPL, several studies have reported that it suffers from various limitations and weaknesses including slow convergence, unfairness of load distribution, and inefficiency of bidirectional communication among many others [33][34][35][36][37][38][39]. Thus, the aim of this thesis is to address some of the key gaps of the RPL standard considering the specific requirements of LLNs, and ultimately pave the way for widespread deployments of LLN applications and services in many different fields.

1.2 Problem Statement and Research Questions

Efficient routing strategies are essential for efficient LLNs operation, and the poor performance of such networks severely affects the services that can be offered and supported hindering LLNs deployment in the future. Despite the advantages brought out by the standardized routing protocol RPL, research studies have reported that the standard still suffers from several gaps that may harm its efficiency [40][41][42][43][44][45][46][47]. These gaps can be identified as follows:

First, RPL adopts an algorithm called Trickle [32] to propagate and maintain routing information. Among reported issues pertaining to adopting Trickle for routing maintenance in LLNs are: slow convergence time limiting the applicability of the protocol to be used in time-critical applications and its response to failures, uneven load-distribution that may lead to constructing sub-optimal routes, high risk of incorrect configurations of Trickle parameters which may result in creating sub-optimal routes especially in heterogeneous topologies composed of regions of different densities [48][49][52][53][54][55].

Second, RPL suffers from a significant scalability problem in applications that require bidirectional communications as it was designed with the vision that the multipoint-to-point (MP2P) communication pattern is the predominant pattern in the context of the IoT[8][40]. For

instance, the table-driven mode of RPL requires that every node must maintain the routing state for all nodes in its sub-network (which is too demanding for memory-constrained nodes). In addition, the source-routing mode is restricted by number of hops that can be attached to source header which is far away from satisfying the scalability requirement of a wide spectrum of applications [8][40].

Third, RPL routing selection and optimization mechanism suffers also from a load balancing issue that may harm the protocol performance in terms of energy efficiency and reliability. In fact, once a RPL node's has selected its preferred parent (next-hop), all traffic will be forwarded through this preferred parent, as long as it is reachable, without any attempt to perform load balancing among other available parental candidates [8][27][57][58]. This behavior may drain the power of overloaded parents leading to network disconnections and reliability issues, as it is likely that overloaded nodes will die earlier [57][59]. This issue is more serious in the context of RPL compared to general routing protocols as RPL's environments (i.e., LLNs) are highly restricted in terms of their resources.

Based on the above observations, the following research questions are to be addressed:

- How can routing maintenance operations in LLNs be achieved while featuring fast convergence time, low overhead and fair load distribution?
- How can routing in LLNs be designed to be scalable and reliable under various traffic patterns whilst taking into account the limitations of such networks?
- Is it possible to design a load-balancing routing primitive that preserves the stability of the network and enhances its performance?

1.3 AIMS AND OBJECTIVES

The main aim of this thesis is to address the key gaps in the RPL standard and its routing maintenance algorithm, Trickle, in terms of scalability, reliability, convergence efficiency, and

load distribution. Our intended research work contains a group of objectives that will pave the way towards achieving our overall aim listed as follows:

- **Objective 1:** Gain an in-depth knowledge and master the state-of-the-art of IoT and LLN concepts, their potential applications, and the new standard protocols stack of the IoT with a special focus on routing.
- **Objective 2:** Scrutinize and analyze the major concerns and key design issues of the standardized IoT routing primitives (i.e., RPL and Trickle).
- **Objective 3:** Develop a new routing maintenance solution for LLNs that enhances the efficiency in terms of overhead, power consumption, and convergence time, and evaluate its performance to prove the feasibility of such a solution.
- **Objective 4:** Develop a new route selection and optimization objective function that strives to fairly distribute the traffic among LLN nodes while maintaining stability and evaluate its performance to prove the validity of such a solution.
- **Objective 5:** Develop a new downward routing solution for LLNs that widens RPL applicability in bidirectional large-scale network and evaluate its validity.

1.4 CONTRIBUTIONS

In this thesis, the aim is to push the boundaries of routing in LLNs beyond the state-of-the-art standardized solutions to further enable widespread deployments of scalable, reliable, and energy-efficient LLNs in the context of the IoT. To this end, and in addition to the literature review, the key contributions of this thesis can be summarized as follows:

- **An Efficient Routing Maintenance Algorithm**

The first major contribution of this thesis is the proposal of an efficient routing maintenance algorithm named, Drizzle algorithm which addresses the shortcomings of RPL's routing maintenance primitive (Trickle) related to its convergence time and the inefficiency of

its suppression mechanism. A distinguishing feature of this solution is the introduction of an adaptive suppression mechanism that permits the nodes to have different transmission probabilities consistent with their transmission history. Another distinctive feature of Drizzle is its fast convergence time thanks to the removal of the listen-only period from Drizzle's intervals whilst proposing a new policy for setting the redundancy counter that mitigates the side-effect of short-listen problems resulting from the removal of listen-only period.

- **A New Load-Balancing Aware Objective Function**

The second major contribution of this thesis is the development of a new load-balancing objective function for LLNs, named, LBSR. The main goal of this new objective function was to incorporate a load-balancing scheme into RPL while preserving the network stability which might be harmed as the network strives to load-balance the traffic. Hence, the instability problem is mitigated by advising a new mechanism for path selection and optimization that perform the parent selection according to a pre-specified regular scheduling interval rather than the way used by RPL.

- **A Leaf-Based Downward Routing Mechanism**

The third major contribution of this thesis is the introduction of a new routing mechanism that lowers significantly the number of routing entries that need to be maintained at the routing tables of RPL's nodes. In this novel approach, a node only needs to maintain the routing state of leaf nodes in its sub-DODAG rather than the whole group of nodes which enhances the scalability of RPL standard and reliability.

1.5 CONTRIBUTIONS COMPLEMENTARITY

While each contribution made in this thesis can stand alone to strengthen the RPL protocol applicability under a related category, they can be also be enabled all at once to achieve more than one objective simultaneously. For instance, our proposed maintenance algorithm can be

integrated into RPL where applications require stringent convergence requirements only whereas the three contributions can be enabled together where applications require stringent convergence requirements, efficient memory utilization and fair load distribution. Indeed, I have found that three key gaps hindering the applicability of the protocol in a wide array of applications are: (1) slow convergence time as it limits the applicability of protocol in real-time applications, (2) the an inefficiency of the standard in terms of downward routing as it limits the applicability in large-scale networks dominated by traffic going from the sink towards its associated nodes, and the absence of fair load distribution among nodes as it may affect the protocol applicability in applications with strict reliability and power consumption requirements.

1.6 THESIS STRUCTURE

The remainder of thesis is organized into six chapters as follows:

- **Chapter 2: Low-power and Lossy Networks (LLNs)**

This chapter presents a thorough background on LLNs. It specifically provides an overview of LLN environments, characteristics, limitations, their unique routing challenges and routing requirements defined by the standardization bodies. It also elaborates on the relevant standards and radio communication technologies that underpin the transition of such networks into the Internet of Things (IoT) world.

- **Chapter 3: Literature Review of The IPv6 Routing Protocol for LLNs (RPL): Operations, Limitations and Enhancements**

In this chapter, a comprehensive overview of the RPL standard is presented including its topology and technical operations, as well as its limitations and drawbacks reported in the literature that are related to its core operations (i.e., routing selection and optimization, routing maintenance operations and downward routing). This chapter enables the reader to gain the

necessary background for understanding the contributions made in this thesis. An extensive survey and an in-depth analysis of research efforts made to address the limitations of RPL have been provided assessing where they fail in overcoming RPL's limitations.

- **Chapter 4: Drizzle: Fair Route Maintenance Algorithm for LLNs (Contribution 1)**

This chapter presents the thesis' first contribution which proposes a new routing maintenance algorithm for LLNs, named, the Drizzle algorithm. The new algorithm addresses the two main limitations of RPL's standardized routing maintenance scheme (i.e., the slow convergence and the inefficiency of the suppression mechanism of the Trickle algorithm). Drizzle employs an adaptive suppression mechanism to boost the fairness in RPL networks. To enhance the convergence time, Drizzle removes Trickle's listen-only period and introduces a new scheme for setting the redundancy counter in order to eliminate the short-listen problem resulting from removing the listen-only period.

- **Chapter 5: A New Load-Balancing-Aware Objective Function for RPL IoT Networks (Contribution 2)**

Chapter five describes the second contribution of this thesis which addresses the load balancing issue in the RPL standard. It mainly proposes a new load-balancing-aware Objective Function that ensures a fair distribution of data traffic among nodes while minimizing overhead as well as preserving network stability. The new OF consists of four modules: i) a new less-overhead scheme for calculating the number of children (the load-balancing metric); ii) a new propagation mechanism to disseminate the routing information more efficiently; iii) a new composite metric that lexically combines the number of children and the ETX metric with the goal of building a balanced and reliable topology; and v) a new policy for switching to the preferred parent while preserving network stability.

- **Chapter 6: A Leaf-Based Downward Routing Mechanism (Contribution 3)**

This chapter presents the thesis' third contribution which addresses the limitations of RPL's storing mode with the goal of enhancing its scalability. The mechanism aims at significantly lowering the number of routing entries need to be maintained at the routing tables of RPL nodes. In this novel approach, a node needs only to maintain the routing state of leaf nodes in its sub-DODAG rather than the whole group of nodes which enhances the scalability of the RPL standard and reliability.

- **Chapter 7: Conclusion**

This chapter concludes the thesis by summarising the main gaps addressed in this research and elaborating on the main methods used to tackle such gaps. It also elaborates on the main findings obtained and sheds light on the key lessons learned highlighting future research prospects and directions.

CHAPTER 2: LOW-POWER AND LOSSY NETWORKS

In this chapter, a thorough background on LLNs characteristics, environments, communication technologies, as well as their unique routing requirements are presented within the context of IoT applications.

2.1 LLN CHARACTERISTICS

The term Low-power and Lossy Networks (LLNs) was introduced by the IETF standardization body to refer to a class of wired and wireless networks where the hosts are tightly constrained in their resources and communication technologies [8]. While the resources limitations include restricted power reserves and restricted processing and storage capacities, the underlying communication technologies may exhibit low data rates, highly asymmetric link characteristics, high data loss and high variability of data loss, and short communication ranges [8]. A typical LLN may consist of anything from a few routers to thousands of resource-constrained actuators and sensors with some routing capabilities connected to the external world (e.g., Internet) through a special LLN Border Router (LBR) that has no restrictions itself [16][17]. The architecture of a typical LLN is shown in **Figure 2-1**.

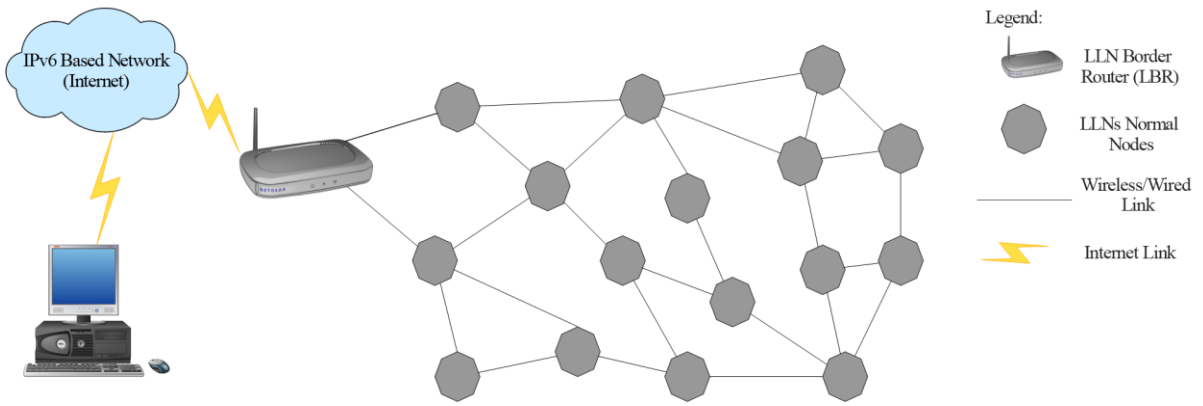


Figure 2-1. The architcure of LLNs including the LBR [4]

The LLN hosts generally exhibit similar characteristics, however, differences may exist in computing and storage capabilities. In this regard, the IETF has classified sensor nodes, based on their capabilities, into three classes: 0, 1 and 2 [11]. Class 0 devices are severely constrained in terms of memory and processing with no more than 10 KiB of memory: they are incapable of carrying out communications without the help of a gateway node [11]. Class 1 devices are less constrained in terms of memory and processing capabilities, have the capacity to run a lightweight protocol stack and carry out communications with other hosts without requiring a gateway node. Finally, Class 2 devices are the least constrained in terms of memory or processing capabilities and have the capacity to support a protocol stack similar to that employed in traditional computers. However, even Class 2 devices can gain benefit from running a lightweight stack since more application resources will be available if fewer resources are used for networking [11]. This also has benefits in reducing development cost and supporting the interoperability between the three classes [11].

2.2 LLN Standards and Radio Technologies

In order to facilitate the efficient deployment of LLNs in the context of IoT, several standards and radio technologies have been developed by different standardization bodies and research communities. In the following subsections, the three main standards underpinning the LLNs are outlined, namely, the IEEE 802.15.4, the 6LowPAN, and the IETF 6TiSCH. In addition, an overview of other radio technologies within LLN environments is provided.

2.2.1 IEEE 802.15.4

In order to satisfy the special requirements of the Low Rate Wireless Personal Area Networks (WPANs), an initial version of the IEEE 802.15.4 standard was introduced in 2003 by the IEEE 802.15 WPAN™ Task Group 4 (TG4). This version [60] defines the operations of two optional PHYs in different frequency bands with a very simple MAC layer. The standard

was then revised and amended several times specifically in 2006, 2007 and 2009. All these amendments were finally rolled out in a new version into a single document in 2011[61]. Pertaining to the technical features, the IEEE 802.15.4 has a maximum transmission rate of 250 Kb/s and a *maximum* transmission unit (MTU) of 127 bytes; however, only up to 116 bytes are available for an upper layer protocol [4]. At the MAC layer, the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) scheme is used to govern access to the wireless medium. Thanks to the efficiency of the standard, many of the recently specified upper layer networking stacks including 6LowPAN, ZigBee, and WirelessHART are built on top of IEEE 802.15.4 [4].

2.2.2 6LoWPAN

Due to the restrictions imposed by LLN devices and their underlying technologies, initially the Internet Protocol version 6 (IPv6) [62] was considered to be too resource intensive for such constrained devices [63]. Alternative proprietary solutions tended to implement complex application gateways to translate the non-IP format understood by those networks to the IP world [63][64]; however, various issues limited the adoption of this technology and led to a re-evaluation of the suitability of IPv6 [65]. In this new vision, the LLNs are no longer seen as isolated systems, i.e., proprietary solutions, rather they are seen as a key enabling technology for the ever-growing Internet of Things (IoT) paradigm where myriads of identifiable smart objects, including smartphones, computers, laptops, actuators and sensors, are connected on the Internet [66][67]. However, the eventual LLN transition to the IPv6 world did not automatically resolve the old concerns about the demand on device resources and underlying communication technologies. For instance, while the key IEEE 802.15.4 medium access standard can only support a Maximum Transmission Unit (MTU) of 127 bytes, the IPv6 protocol requires a minimum datagram size of 1280 bytes, approximately ten times greater [8]. In order to address such obstacles, the IETF commissioned the “IPv6 over Low Power Wireless

Personal Area Network (LoWPAN) working group" to generate protocols that ensure smooth integration between LLNs and other networks running the IPv6 [4][10]. These efforts culminated in specifying a new standard that allows IPv6 packets to be carried within the IEEE 802.15.4 MAC layer named 6LoWPAN [4]. This has been enabled by identifying an adaptation layer between the data-link layer and the IPv6 network layer as illustrated in **Figure 2-2a.** In particular, the 6LoWPAN adaptation layer defines mechanisms for IPv6 header compression, IPv6 packet fragmentations and reassembly so that the IP datagram can be carried within the IEEE 802.15.4 frames.

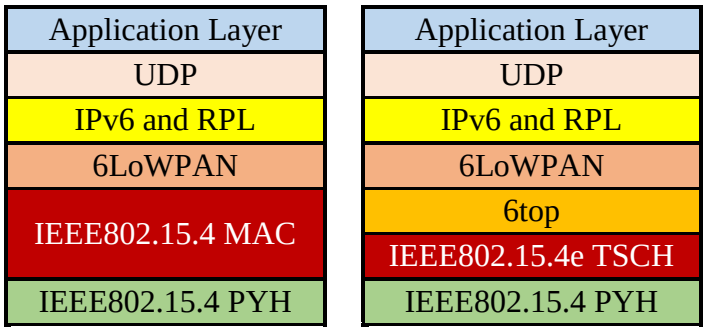


Figure 2-2. 6LoWPAN stack (a) and 6TiSCH stack (b) [68]

2.2.3 IEEE 802.15.4E TSCH AND IETF 6TISCH

Due to the single-channel related unpredictability of the IEEE 802.15.4 CSMA/CA in multi-hop networks, and to cope with the resource-constrained nature of LLNs, the IEEE introduced the Time-slotted Channel Hopping (TSCH) mode as an amendment to the MAC part of the IEEE802.15.4 standard in 2012 [68]. This new mode combines the TDMA (Time Division Multiple Access) with the channel hopping with the goal to improve both the energy efficiency and reliability [68][69]. While the TDMA scheduling minimizes the contention, and thus providing more efficient energy consumption, the channel hopping enhances the network reliability and mitigates the effect of channel fading [70][71]. In order to integrate the TSCH MAC protocol with IPv6 LLNs especially for industrial applications, the IETF chartered the "IPv6 over the TSCH mode of IEEE 802.15.4e" (6TiSCH) working group to enable IPv6 on

top of TSCH mode [71]. The 6TiSCH defines the 6TiSCH Operation Sublayer (6top) that specifies how nodes can communicate to add or delete cells and when such an addition and deletion can occur. The 6TiSCH also defines a set of distributed scheduling protocols that manage the allocation of resources. At the time of writing, the 6TiSCH is still active with two RFCs and five Internet-Drafts. These IEEE and the IETF joint standardization efforts have given birth to a modified 6LoWPAN stack named the 6TiSCH stack shown in **Figure** 2-2**b**.

2.2.4 OTHER COMMUNICATION TECHNOLOGIES

2.2.4.1 Bluetooth Low Energy (BLE)

BLE, marketed as Bluetooth Smart, is a WPAN technology designed for very low power operation, and is optimized for data transfer solutions as opposed to media transfer [72]. The recent specification of BLE (Bluetooth 5.0) [72] provides support for a data rate of up to 2 Mb/s within a short range (up to 200 meters) with multiple network topologies, including peer-to-peer, star, and mesh [73][74]. Recently, the IETF has developed an adaptation layer for supporting IPv6 over BLE, thus facilitating the integration between IoT and BLE devices. The low-power consumption profile, the AES-based security, the support of channel hopping, the speed of connection and the compatibility of BLE devices are among the key advantages for this promising technology. BLE do have some limitations that may hinder its applicability under specific scenarios such as its limited coverage range, and low data rate restricting its applicability in applications that require a data rate of more than 2 Mb/s or deployed in a large geographical area.

2.2.4.2 Power-line Communication (PLC)

The IEEE 1901.2 standard or PLC specifies communications for low-frequency narrowband power line devices. It relies on re-using the existing electrical wires to provide communication capabilities, thus eliminating the need to run Ethernet cabling throughout the premise which is considered to be one of its key advantages [75]. It also has the advantage of

being a wired medium eliminating the risk of interference associated with wireless connections and also making it the most suitable technology for smart grid environments. Compared to its wireless counterparts, the PLC has the longest communication range, which is only limited by the length of the underline electrical cables [4]. Furthermore, PLC exhibits low latency profile, and high data transmission rates rendering it an ideal option in multimedia applications (e.g. audio, video streaming and gaming) [75][76]. Although PLC has generally more stable connections in comparison to wireless communication technologies, its communication channel can still exhibit some sort of unpredictable and variable quality. In addition, the network can suffer from attenuation due to the presence of other high-speed PLC technologies [4]. Finally, the speed of transmission can be affected by the noise created by electrical devices such as kitchen appliances and vacuum cleaners.

2.2.4.3 Wi-Fi HaLow:

To support the emerging concept of IoT networks, the IEEE 802.11ah Task Group introduced a new communication technology in 2016 named Wi-Fi HaLow based on the IEEE 802.11ah standard [77][78][79]. This new technology operates in frequency bands below 1GHz, thus providing support for the connectivity of low power resource-constrained devices such as small wearable devices or sensors. The perceived range of Wi-Fi HaLow's is nearly double that of mainstream Wi-Fi, with the capacity to penetrate walls or other obstacles while offering very low power consumption. In addition, the technology supports a star topology with thousands of nodes per access point (AP), exhibiting low latency and congestion profiles making it an optimal choice in challenging industrial IoT environments. As Wi-Fi HaLow has adopted most of the mainstream Wi-Fi protocols, it has the advantages of multi-vendor interoperability, easy internet connectivity and robust security. The Wi-Fi HaLow supports data rates of at least 100 Kb/s with a communication range of around a kilometer [79]. The key

limitation of Wi-Fi HaLow is the issue of interference with other devices as its band is shared with such devices.

2.3 UNIQUE ROUTING CHALLENGES IN LLNS

The design of efficient routing protocols for LLNs is driven by the unique characteristics of such networks. The limited memory and processing power, low data rates and limited power supply in the majority of devices along with the lossy nature of interconnects (links), all need to be addressed. In the following, we shed light on some of the routing process design issues arising in the context of LLNs.

2.3.1 DIVERSITY OF APPLICATIONS

Several applications are envisaged to run under the umbrella of LLNs including home/building automation, industrial applications, environmental monitoring, military applications, etc. These diverse applications exhibit characteristics and, consequently, different requirements in terms of power consumption, convergence time, traffic overhead, reliability, latency or other performance metrics. Hence, a big challenge for a LLN routing protocol is to accommodate all of these diverse and conflicting requirements within the application's resource budget [31][80].

2.3.2 COMMUNICATION PATTERNS

The dominant communication pattern in LLN applications is the MultiPoint-to-Point (MP2P) [27] , in which data is gathered by a group of sensors and reported to a common destination called the LBR or the sink. Other communication patterns also exist, including the Point-to-MultiPoint (P2MP), where the sink sends data to the associated sensor nodes and the Point-to-Point (P2P) in which a sensor node communicates with one other in the network [12][13]. This diversity in communication patterns represents another challenge when designing LLNs routing protocols. For instance, while the MP2P pattern will require a minimal

routing state to be maintained, the P2MP and P2P patterns will be memory-demanding as a significant routing state needs to be maintained by the routers in the network.

2.3.3 REPORTING MODELS

The data communication models in LLNs vary widely, but are roughly classified into three categories, query-based, event-based, and time-based [80]. In the query-based model, data is only reported upon the receipt of an explicit query. In the time-based model, sensing devices report their data of interest periodically at a pre-specified time interval. In the event-based model, sensing nodes only report their readings upon detecting abrupt and significant changes in the value of data of interest. Hybrid models combining two or more of these are also encountered [80]. Hence, a periodic reporting model would require the routing protocol to be of a proactive nature, whereas it might be more efficient for the protocol to be reactive with query and event based reporting models as such models will usually require on-demand construction of routing paths. Thus, a routing protocol should have the capacity to tweak its state according the model it handles.

2.3.4 SCALABILITY

It is envisioned that LLNs will operate in deployments of many different densities, ranging from a few neighbors per node to hundreds [12][13][14][15]. The density of a specific deployment may or may not be fixed in advance. In fact, a protocol that operates well in a low-density network might not run efficiently in large-scale deployments, as they would generate relatively a larger volume of traffic causing congestion problem that need to be considered. A key issue that might emerge due to the congestion in routing protocols is the loss of exchanged control messages that carry routing information. This may result in taking routing decisions based on obsolete or incorrect routing information leading to performance problems. Thus, a routing protocol should be able to handle such cases within the viable range and its parameters

should be dynamically tuned according to what it encounters in practice [31]. In other words, scalability is a design issue that should be satisfied by a LLN routing protocol.

2.3.5 SCARCITY OF RESOURCES

The resource-constrained nature of LLNs imposes a new set of restrictions on developing efficient routing protocols and primitives. Generally speaking, the small-battery capacity of a sensor node is the most restrictive factor and must be carefully considered [81]. Thus, a routing protocol should opt to send just enough updates to ensure the freshness of the constructed routes while maintaining low-power profile. 'Just-enough' updates can vary from transmitting one update every second to a bulk transmission every few minutes depending on the current conditions of the network and to ensure that the application energy budget is met [31]. The first might be less energy efficient, however, it might be more efficient in terms of other performance factors such reliability or latency and vice versa.

2.3.6 LINKS UNRELIABILITY

LLNs are characterized by lossy and unreliable links, and an update is not guaranteed to reach its destination from its first transmission [82][83]. In some cases, the link loss rate in a network cannot be predicted beforehand and, even worse, the same link may exhibit different loss rates over time due to factors such as collisions at the receiver, the hidden terminal problem and interference with the radio transmitters of neighboring nodes [82] [83]. However, there are still cases where an a priori loss rate can be roughly predicted, for example, based on the statistics of previous deployments or based on machine-learning algorithms. In fact, the lossy nature of LLNs may result in intermittent changes in link qualities that will influence the network formation by forcing a node to change its next-hop frequently, thus harming the network stability. Hence, a routing protocol should have the capacity to operate efficiently under such unreliable conditions

2.3.7 MOBILITY AND NETWORK DYNAMICS

The sensor nodes in LLNs are conceived to be stationary in the vast majority of scenarios, however, there are still cases in which there are a considerable number of mobile nodes [12][13][14][15]. For instance, in health monitoring applications, the usual mode of deployment is mobile because sensor nodes are attached to the human body in order to monitor conditions remotely while the subjects go about their business [84]. Therefore, general routing strategies must take account of possible node mobility.

2.4 LLNS ROUTING REQUIREMENTS

The introduction of 6LoWPAN emphasized the need for additional IPv6-based routing solutions for LLNs and, soon afterwards, the IETF commissioned the Routing over Low power and Lossy networks (ROLL) working group to design IPv6 routing solutions for LLNs [12]. The ROLL working group recognized that a wide range of applications exists in LLNs each with its own routing requirements. Its first objective, therefore, was to define the routing requirements for four anticipated application areas, namely, Home Automation [12], Building Automation [13] Industrial LLNs [14], and Urban LLNs [15]. A discussion of these areas is now presented. **Table 2-I** summarizes the routing requirements of the four specified categories of LLN applications.

Table 2-I. LLN routing requirements

Requirements	Home Automation	Building Automation	Industrial	Urban
Latency	Real-time, alarm and light control applications:<250 ms Other Applications: tens of seconds	Real-time, alarm and light control applications:<250 ms Other Applications: tens of seconds	Tens of milliseconds to seconds based on the type of the application	Variable based on the type of application
Convergence	Mobile: few seconds Fixed : less than 500 ms Subjected to: up to 250 nodes and four hops.	Fixed : less than 5 seconds Mobile: less than 10 seconds	Newly added device: within tens of seconds or several minutes Subjected to: tens of devices	Reporting Applications: lower than the smallest reporting interval
Network Scale	Typical: 10- 100 nodes Max: 250 nodes	Typical : 100- 1000 nodes Max: 10000 nodes Subnetworks: Up to 255 nodes	Typical: 10- 200 nodes	Max: 10^7 Subnetworks: 10^2- 10^4
Hops	Typical: 5 hops Max: 10 hops	Typical: 5 hops Max: 10 hops	Max: 20 hops	Several hops to several tens of hops
Mobility	Needs to be supported	Needs to be supported	Needs to be supported	Generally fixed locations
Traffic Pattern	P2P (prevalent), P2MP, MP2P	P2P (30%), MP2P and P2MP (70%)	MP2P (prevalent) P2MP (rare) P2P (rare)	MP2P (prevalent) P2MP and P2P (moderate)
Communication Model	Query-based (prevalent) Regular-based Event-based	Regular Query-based Event-based	Periodic , Query-based and Event-based	Regular (prevalent), Query-based (occasionally) Alarm-based(rare)

2.4.1 HOME AUTOMATION

Recently, the usage of sensing devices and actuators has increased in smart home applications. The modern home automation applications typically encompass both sensors such as gas detectors, and actuators such as heating valves [12][16]. These applications are designed to allow for the electrical devices at home to be connected to an IP-based system that controls these devices based on some input values. Typical use-cases include: at-home health reporting and monitoring; lighting, central heating and air conditioning remote control; alarm systems for various hazards (e.g. carbon-monoxide, smoke, fire detection, panic button, etc.) [12]. The majority of devices (sensors and actuators) in a home-operated network are stationary; however, there are scenarios where mobile devices are present, such as the wearable healthcare devices used to collect bio-medical signals remotely and home applications controlled using a remote controller that moves from one location to another at random [12]. Supporting mobility is, therefore, a necessary requirement for the successful deployment of home automation networks. The traffic patterns within this category vary widely [12][16]. For example, the

MP2P communication model is used for communicating health conditions (e.g. blood pressure, temperature, insulin level, weight), while the P2MP model is more appropriate for a lighting control system, where a central device sends control commands to associated devices [16]. However, the P2P traffic pattern has dominance here as most of the traffic in home-automation applications is generated by wall controllers and remote controls to their associated light or heat sources. It is envisioned that a typical home automation network will be composed of tens of nodes with a maximum hop separation of a few nodes, and typically network diameter of five hops [16]. Many devices will be battery-powered so power consumption should be kept minimal to prolong network lifetime [12]. The majority of devices in home automation networks are likely to be Class 0 nodes (e.g., wall switches) with the rest of typically Class 1. The routing protocol for stationary devices has convergence requirements of no more than half a second, relaxed to four seconds in the presence of mobility. For instance, a remote control appears unresponsive if it takes more than a second to pause the music [12][16].

2.4.2 BUILDING AUTOMATION

These systems are deployed in a large set of commercial buildings such as hospitals, colleges, universities, high schools, governmental and manufacturing facilities [13]. They typically enable automatic control of a commercial building's lighting, air conditioning, ventilation, elevator, fire-response and physical security, among other systems [13]. The main purposes behind building automation are: reducing operating costs and energy consumption; enhancing occupant comfort; improving building service quality and the utilities life cycle [13]. As with home automation, the majority of nodes in building-automation networks are stationary devices with a small proportion of mobile nodes [13] and the majority of nodes are Class 0 and Class 1 devices. It is expected that 30% of the traffic in building networks will be P2P with a typical frequency of one packet per minute. For example, in a temperature-controlling application, a sensor will unicast periodically (e.g., each minute) temperature

readings to its associated controller and expect an acknowledgment unicast from that controller. The MP2P and P2MP will account for 70% of traffic in this domain [13][16]. This is due to that most messages in building-automation applications are directed toward an aggregation point and then routed off the LLN for further processing (MP2P). In addition, an acknowledgment is unicast from the destination to the respective sender (P2MP). The number of nodes in such a network is likely to be of larger size than in the domestic equivalent. However, a large building network would typically be divided into subnetworks of no more than 255 nodes to ensure that critical systems such as air conditioning and light systems are not vulnerable to global failures [16]. The latency requirements in building automation systems is somehow similar to the latency requirements in home-automation applications. However, many of the applications in this category are mission-critical (e.g., security fire) that are very sensitive to delay and require in-time delivery of messages. Network devices (sensors and actuators) might be mains-powered, battery-less, or battery-powered [13].

2.4.3 INDUSTRIAL LLNS

Industrial applications of LLNs enable plant and factory workers to manage remotely multiple control units at the site as well collect large amounts of information. Many application scenarios fall under this category and they can be roughly classified into two different segments known as Factory Automation and Process Control [14]. Process Control applications target fluid products such as liquid chemicals, gas and oil, whereas Factory Automation is concerned with individual products such as cars, toys and screws [14]. All three communications patterns (P2MP, P2P and MP2P) will usually be present; however, the predominant traffic pattern is expected to be MP2P. The majority of applications will comprise tens of field devices and forwarders with a few hops to reach a backbone router [14]. LLN devices in industrial networks may use a variety of sources to provide power: while some will be line-powered, the majority will be battery-operated with lifetime requirements of at least five years [14]. The issue of

mobility in this category is more complicated than in the home and building scenarios and velocities of up to 35kph are possible [14]. For instance, some field devices may be located on moving objects such as cranes. Since critical classes within this category are not expected to be handled by LLN routing protocols, the requirement of rapid convergence is somewhat relaxed. It is stated in [14] that a routing protocol should converge within a few minutes of adding a new node with a latency of no more than ten seconds when delivering packets via established routes.

2.4.4 URBAN LLNS (U-LLNS)

These networks are dedicated to measuring and reporting a wide gamut of data in outdoor urban environments with the primary goal to improve inhabitants' living conditions and monitoring compliance with environmental laws [15]. Typical applications include the monitoring of meteorological conditions or pollution and allergen concentration in specific regions. The dominant communication paradigm is the MP2P, as most of the traffic in this category will be generated by the sensor nodes and directed to the LBRs [15]. For example, the sensing nodes that gather temperature readings could communicate data every hour or every day. The P2MP model is also present: for instance, a query statement can be launched by a central unit to request pollution level readings from a group of sensors in a specific region. Although most sensing devices in this category are expected to be stationary, the dynamicity of the network is not negligible, due to node disappearance, disassociation and association, in addition to perturbations of node interconnects [17]. Scalability represents the biggest concern in this category as the extensive measurement spaces in urban environments can result in very large networks. As currently imagined, an urban network will comprise more than a hundred nodes but sizes of tens of thousands, perhaps even millions of nodes, may be reached in the future [15][17]. Although urban network node cardinality is expected to be of the order of 5 to 10 nodes, examples of nodes with hundreds of neighbors may be encountered [15][17]. In

addition, the physical distance between network devices can span from hundreds of meters to as much as a kilometer. Thus, it is unlikely that any field device will be able to reach its border router in a single hop, and multi-hop distributions composed of as many as tens of hops may be unavoidable [15]. A mix of sparse and dense deployments in urban networks is expected: for instance, hundreds of devices may be presented in close proximity within one building in an urban area, whereas sparse node distributions, with low cardinality, would be the norm in sparsely built-up areas [17]. Devices may be powered using a variety of mechanisms, including: non-rechargeable batteries; rechargeable batteries with irregular or regular recharging; inductive/capacitive energy provision; or always-on (e.g., a powered electricity meter) [15]. It is likely, however, that the majority of nodes will use non-rechargeable batteries with lifetime requirements of 10-15 years [15]. Latency requirements in urban applications vary widely. For instance, for periodic traffic, latencies of up to a fraction of the reporting interval may be acceptable, while query-based applications will have somewhat more stringent requirements. Alert traffic is highly sensitive to delay and cannot tolerate a wait of more than a few seconds in the vast majority of cases [15].

2.5 SUMMARY

This chapter has outlined the main concepts related to the LLNs highlighting their characteristics, and the main standardization efforts that paved the way for the transition of such networks from being isolated networks to become an integral part of the IoT paradigm. A discussion of the key radio and communication technologies facilitating the efficient deployment of LLNs in the context of IoT including the IEEE 802.15.4, BLE, PLC, TSCH and Wi-Fi HaLow is also provided. Finally, an elaboration on the routing process design issues arising in the context of LLNs is presented highlighting the routing requirements that are perceived under four key IoT application categories including home, building, industrial and urban applications.

CHAPTER 3: THE IPV6 ROUTING PROTOCOL FOR LLNS (RPL): OVERVIEW OF OPERATIONS, LIMITATIONS AND ENHANCMENTS

RPL is an IPv6-based proactive routing protocol designed by the IETF community to fulfil the routing requirements of a wide gamut of LLN applications [14]. RPL is optimized particularly for data gathering applications (i.e., MP2P traffic pattern), and it also provides reasonable support for the P2MP traffic pattern, while providing indirect support for the P2P pattern [14][15]. In this chapter, an overview of RPL's operations, its routing selection and optimization mechanisms (i.e., objective functions), and its routing maintenance mechanism (i.e., the Trickle algorithm) is provided. In addition, a thorough analysis of RPL's limitations and enhancements proposed to overcome such limitations is presented.

3.1 RPL TOPOLOGY AND OPERATIONS

RPL organizes its physical network into a form of *Directed Acyclic Graphs* (DAGs) where each DAG is rooted at a single destination referred to as a *Destination-Oriented DAG* (DODAG) [27][28][29]. The DODAG represents the final destination of the traffic within the network domain bridging the topology with other IPv6 domains such as the Internet [27][29]. The abbreviation LBR, LLN Border Router, is the terminology used to refer to the DODAG entity in the context of LLNs. In addition, RPL uses the term *upward* routes to refer to routes that carry traffic from normal nodes to the LBR (i.e. MP2P) whereas routes that carry the traffic from the DODAG root to other nodes (i.e. P2MP) are called the *downward* routes [27]. To build the *upward* routes, each node within the network should select the best neighbor in terms of the *rank*, a value represents the node's relative distance from the root, as its preferred parent. Similarly, each node willing to participate in the *downward* routing must announce itself to one of its parents, preferably the preferred parent. The details of building the *upward* and *downward* routes are given in the following subsections. RPL uses the term *instance* to refer to multiple

DODAGs that share the same objective function, a set of routing policies and mechanisms for routes selection and optimization. Multiple RPL instances may coexist concurrently in a specific physical topology, and a node may join more than one instance at time. However, within each instance, a node is allowed to associate with only one root (DODAG) [27][29]. An illustration of a RPL topology with two instances is depicted in **Figure** 3-1. As shown in the figure, Instance 1 uses the latency metric to build its own DODAG topology whereas Instance 2 uses the ETX metric. Hence, the numbers on the links represent the cost of latency on that link in Instance1 whereas they represent the ETX cost in Instance 2. A node with more than one candidate parent will select the parent with the least total cost as its preferred parent. The total cost or the node's own rank is calculated by adding the rank of its preferred parent and the cost of link to that parent.

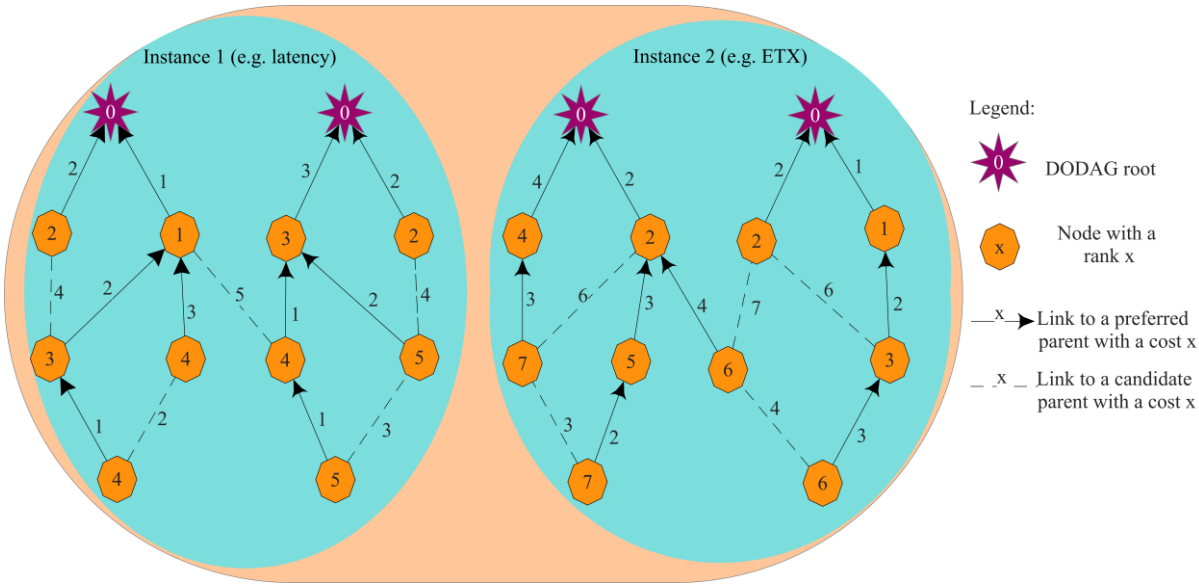


Figure 3-1. Two instances on the same physical topology with each instance incorporates two DODAGs

3.1.1 RPL Control Messages

To exchange routing information needed to construct the network topology and routing paths, RPL introduces four types of ICMPv6 control messages (excluding the security messages) as detailed below.

DODAG Information Object (DIO): the DIOs are used to carry the relevant information and configuration parameters that enable a node to discover RPL instance, join a specific DODAG, select a set of candidate parents, and maintain the DODAG [27].

Destination Advertisement Object (DAO): this control message allows a node to propagate its destination information upward along the DODAG to the DODAG root so that the downward routes from the DODAG root to its associated nodes can be constructed [27].

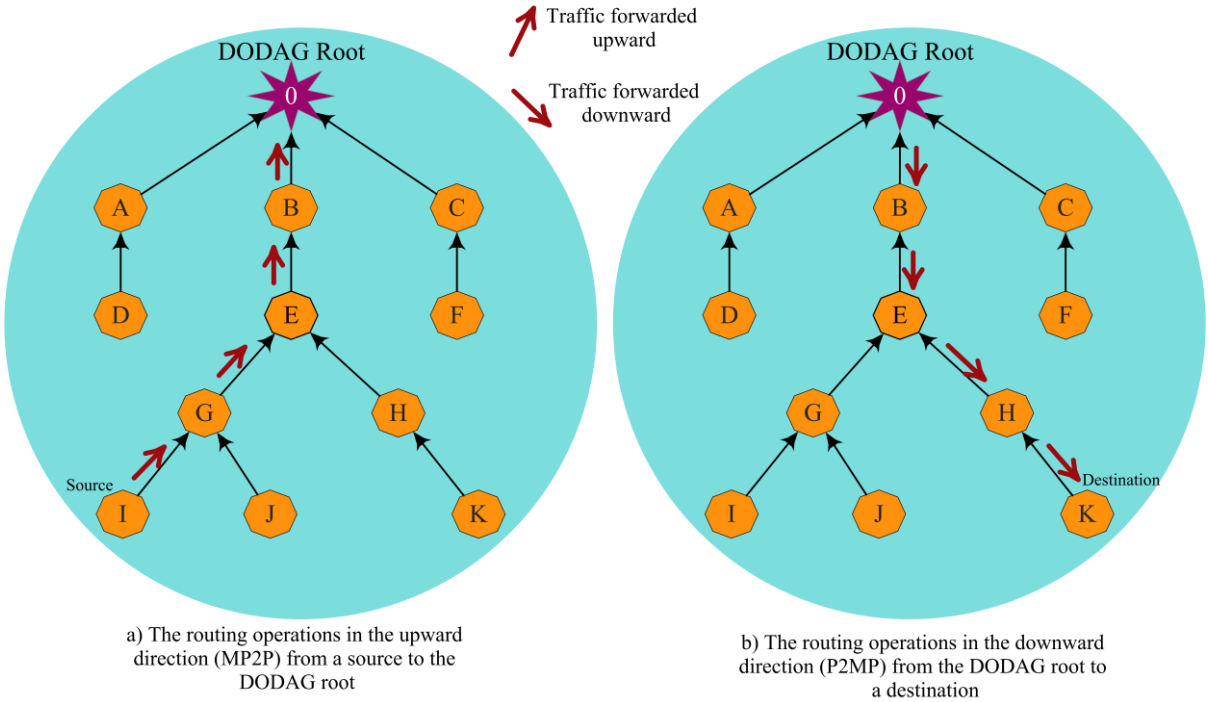
DODAG Information Solicitation (DIS): this message is used by a RPL node to solicit a DIO from neighboring nodes in order to join the DODAG[27].

Destination Advertisement Object Acknowledgement (DAO-ACK): the DAO-ACK may be unicast by a DAO recipient to the DAO sender to acknowledge the reception of that DAO [27].

3.1.2 RPL UPWARD ROUTES (BUILDING THE DODAG TOPOLOGY)

The process of building the DODAG and upward routes is controlled by DIO messages referred to previously [27][29]. In addition to other routing information, the DIOs carry the rank, the relative position of a RPL node with respect to the DODAG root, and a routing policy called the Objective Function (OF) that specifies how a RPL node computes its rank and selects its preferred parent (the details of rank and OF are presented in Section 3.2). Specifically, the construction of the DODAG is initiated by having the DODAG root multicasts DIO messages to its neighboring nodes announcing its rank and the OF that should be used [27][30]. When receiving a DIO, a RPL node (a) adds the sender address to its candidate parents set, (b) calculates its own rank, (c) selects its preferred parent from the candidate parents, and finally, (d) updates the received DIO with its own rank and then multicasts the calculated rank to other neighboring nodes [27][29][30]. The node may also silently discard the received DIO based on some criteria defined in the RPL specification. This process lasts until all nodes have setup their routes in the upward direction towards the DODAG root. Hence, the forwarding of packets

from nodes to the DODAG root proceeds normally by the source forwarding its packet to its parent that in turn forwards such a packet to its own parent until it reaches the root as depicted in **Figure** 3-2**a** . In **Figure** 3-2**a**, a packet is sent from node I to the DODAG root, hence, node I will simply forward the packet to its own parent node G which in turns will forward it upward until it reaches the DODAG root.



a) The routing operations in the upward direction (MP2P) from a source to the DODAG root

b) The routing operations in the downward direction (P2MP) from the DODAG root to a destination

Figure 3-2. The propagation of data packets in the upward and downward directions of RPL.

3.1.3 RPL DOWNWARD ROUTES

In order to facilitate P2MP and P2P communication patterns, downward routes must also be established and maintained. RPL uses a type of ICMPv6 control messages called Destination Advertisement Object (DAO) for this purpose. A RPL node willing to announce itself as a reachable destination from the root point of view, unicasts a DAO to its preferred parent advertising its own destination prefix [27][29]. The processing of the received DAO by the parent relies on the current mode of operation advertised in the DIO messages. To this end, RPL has specified two modes for creating and maintaining downward routes, namely, storing (table-driven) and non-storing (source routing)[27][29].

In the storing mode, when a parent receives a DAO from one of its children, it: (a) stores the announced destination prefix locally in its routing table along with the DAO sender address, as the next hop to reach that destination; and (b) forwards the received DAO, in turn, to its own preferred parent to ensure the propagation of the advertised destination upward to the DODAG root [27]. For data-plane operations, classical hop by hop IPv6 routing is used by RPL nodes to communicate with destinations learned from DAOs.

In the non-storing mode, the advertised DAO carries also the address of the destination's parent in addition to the advertised destination prefix. Here, however, a parent receiving a DAO just forwards it to its own preferred parent without maintaining any routing state, until it is finally received by the DODAG root. Once the DODAG root receives the transmitted DAO, it maintains the received information in its routing table in the form of a child-parent relationships, used later by the data-plane to construct a source route for the intended destination [27][29]. Hence, when the root needs to communicate with a specific destination, it attaches the source route of that destination to the packet header and forwards the packet to the next hop. A forwarding node receiving that packet will simply inspect the source routing header to determine on which interface it should send the packet next [27][29]. The operation of forwarding the packet from the DODAG root to a specific destination within its domain is illustrated in **Figure** 3-2**b**.

RPL also provides a support for the P2P traffic pattern in which a node communicates with another node in the network. Hence, when a node needs to send a packet to another node within the DODAG, the packet is first forwarded upward the DODAG until it arrives at an ancestor that has a known path to the destination node. Then, the packet is forwarded downward the DODAG by that ancestor via the intermediate routers and finally to the destination node. A high-level illustration of these operations is depicted in **Figure** 3-3**a,** and **Figure** 3-3**b**. **Figure** 3-3**a shows** the propagation of a data packet from the source I to the destination K in the non-

storing mode. Hence, the packet will travel upward until it reaches the root as the intermediate nodes did not create any routing tables and only the root can construct the source route that needs be inserted to the packet's header for correctly forwarding the packet to its destination K. Once the source route is calculated, the root will simply forward the packet to the first hop in the source route that in turns will forward it downward until it reaches the destination K. **Figure** 3-3**b** shows the propagation of a data packet from the source I to the destination K in the storing mode of RPL. In this mode, the packet will travel upward until it reaches a common ancestor, which is node E in this case, then, node E will forward it downward to node K. This is possible as each node in the storing mode will have created a routing table with routing entries for all of its children. For instance, a snapshot of the routing table of node E that will be built under this scenario is shown in Table 3-I.

Compared to the OSPF protocol, RPL is optimized for the upward traffic and the routes used for forwarding that traffic are used to forward the downward traffic even though they may not be efficient for that traffic pattern. In addition, OSPF uses only static link metrics and a proactive routing maintenance approach whereas RPL uses both static and dynamic metrics and reactively maintains its topology [27].

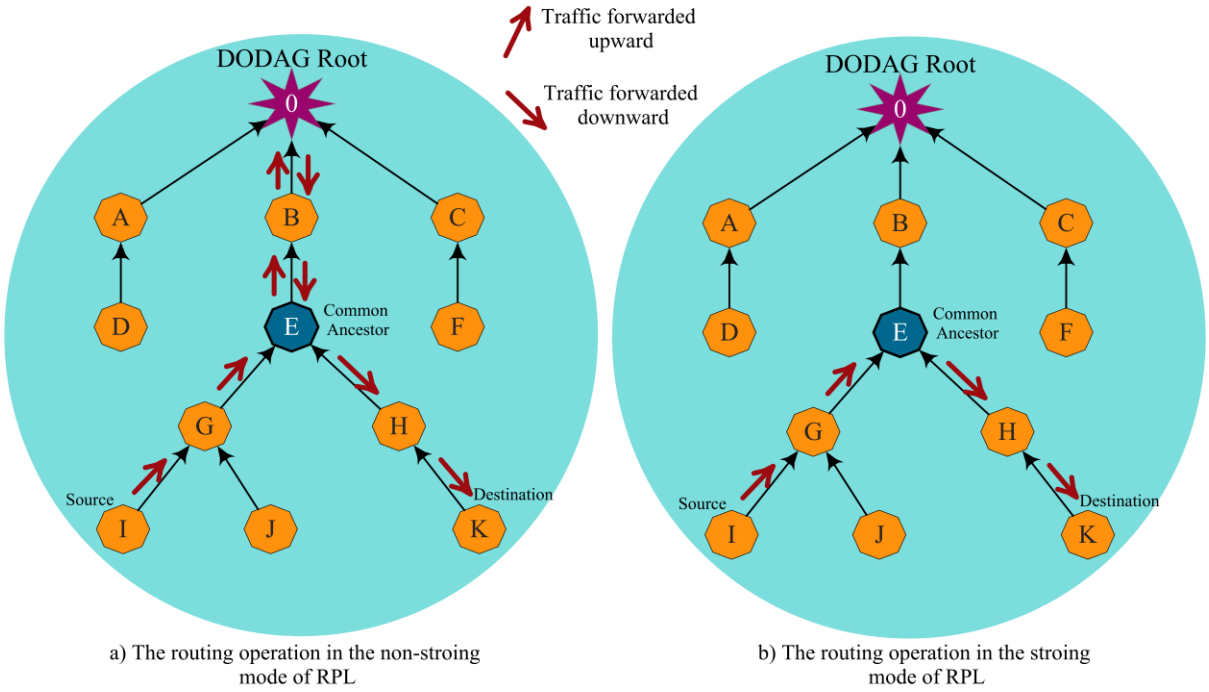


Figure 3-3. a) The propagation of data packets from the source I to the destination K in the storing and non-storing modes.

Table 3-I. The routing table of Node E

Entry	Next Hop
G	G
H	H
K	H
I	G
J	G

3.2 Objective Functions (OFs)

In order to meet the conflicting requirements of different LLN applications, RPL decouples the route selection and optimization mechanisms from the core protocol operations such as packet processing and forwarding [27]. Hence, the core of the protocol is centered on the intersection of these conflicting requirements, whereas additional modules are designed to address application-specific objectives such as minimizing the energy consumption or

maximizing the reliability [85][86]. The term Objective Function (OF) is used to describe the set of rules and policies that governs the process of route selection and optimization, in a way that meets the different requirements of the various applications. In technical terms, the OF is used for two primary goals: first, it specifies how the Rank can be derived from one or a set of routing metrics [87](e.g., energy, hop count, latency, throughput, link reliability and link color); and second, it defines how the rank should be used for selecting the preferred parent. Currently, two OFs have been standardized for RPL namely, the Objective Function Zero (OF0) [85] and the Minimum Rank with Hysteresis Objective Function (MRHOF) [86].

3.2.1 THE OBJECTIVE FUNCTION ZERO (OF0)

The OF0 is designed to select the nearest node to the DODAG root as the preferred parent with no attempt to perform load balancing [85]. The *rank* of a node (R_n) is calculated by adding a strictly positive scalar value ($rank\_increase$) to the *rank* of a selected preferred parent (R_p) according to Eq. 1 and Eq. 2 as follows:

$$R_n = R_p + rank\_increase \quad (1)$$

$$rank\_increase = (R_f * S_p + S_r) * MinHopRankIncrease \quad (2)$$

where the *step-of-rank* (S_p) represents a value related to the parent link metric and properties such as the hop-count or the Expected Transmission Count (ETX), while the *rank* factor (R_f) and *stretch_of_rank* (S_r) are normalization factors. The default values of R_f, S_p, S_r, and $MinHopRankIncrease$ are 1, 3, 0 and 256 respectively [85]. For example, if a node X received a DIO from a node Y with a rank of 256. According to OF0, the calculated rank of node X will be 1024 based on the default values. The OF0 does not specify which metric/metrics should be involved in the calculation of rank increase. For parent selection, a node running OF0 considers always the parent with least possible rank as its preferred parent.

OF0 considers also selecting another parent as a backup in case the connectivity with its preferred parent is lost [85].

3.2.2 MINIMUM RANK WITH HYSTERESIS OBJECTIVE FUNCTION (MRHOF)

The MRHOF [86] is designed with the goal to prevent excessive churn in the network topology (i.e., frequent change of the preferred parent). In the MRHOF, a node calculates the path cost through each neighbor by adding up two components; the value of the candidate neighbor node's or link's metric and the value of the selected metric advertised in the Metric Container. After calculating the path costs of all candidate parents, a node selects the parent with lowest path cost as its preferred parent. However, unlike OF0, MRHOF switches to a new parent only if the new minimum calculated path cost is smaller than the preferred parent's path cost by at least PARENT_SWITCH_THRESHOLD, which is the hysteresis part of MRHOF [28]. For examples, if a node X has a preferred parent Y with a rank of 256 and the link cost to that parent is 128, hence the total cost to parent Y will be 384. Assuming that the PARENT_SWITCH_THRESHOLD is set to 100 and a new candidate parent N becomes available with a rank of 256 and link cost of 68 (total cost of 324), node X will stick to its preferred parent Y although parent N has less total cost. This is because the difference between the total costs of both parents is only 80 which is less than the threshold of 100.

3.3 ROUTING MAINTENANCE (TRICKLE TIMER)

One of the key design principles of RPL is minimizing the routing control overhead and signaling data in order to reduce energy consumption and enhance reliability. In this regard, RPL employs the Trickle algorithm [31][32] to govern transmission of the signaling traffic used to construct and maintain the DODAG. The basic idea behind Trickle is to adjust the frequency of message transmission based on network conditions. Trickle relies on two simple mechanisms to disseminate routing information efficiently. The first is to change adaptively

the signaling rate according to conditions currently present in the network [31]. Specifically, Trickle increases the transmission rate when a change in routing information is discovered (i.e., an inconsistency is detected) as a means to populate the network rapidly with up-to-date information[31] [32]. As the network approaches its steady phase, Trickle exponentially reduces the transmission rate to limit the number of transmissions when there is no update to propagate. The second mechanism used by Trickle is the suppression mechanism in which a node suppresses the transmission of its control packet if it detects that enough of its neighbors have transmitted the same piece of information, thus limiting redundant transmissions. The time in Trickle is divided into intervals of a variable size. A node running Trickle schedule a message to be sent at randomly selected time in each interval. The transmission of a scheduled message is governed by Trickle parameters, variables, and steps. As specified in [32], Trickle uses three maintaining-state variables, three configuration parameters and six steps to operate. The following are the three parameters used by Trickle to configure its timeline.

- The minimum interval length (I_{min}),
- The maximum interval length (I_{max}), and
- The redundancy constant (k).

In addition, Trickle uses the following three variables for maintaining its current state:

- I, the length of the current interval,
- t, A randomly selected time within I to transmit at, and
- c, message counter to keep a track of number of received consistent messages within the current interval.

The following six steps recap the operation of the Trickle algorithm:

1. Trickle starts its first interval by setting I to a value from the range [I_{min}, I_{max}], usually it sets the first interval to the length of I_{min}.

2. When an interval starts, Trickle resets the counter *c* to 0, and randomly selects *t*, from the range [*I*/2, *I*].
3. Upon receiving a consistent message, Trickle increments its counter by a value of 1.
4. At the randomly chosen time *t*, if the counter *c* is greater than or equal to the redundancy constant *k* (There are no rules provided on how to set *k* itself), Trickle suppresses its scheduled message. Otherwise, the message is transmitted.
5. When the interval *I* expires, Trickle doubles the size of the interval. If the size of the new interval would exceed the maximum interval length I_{max}. Trickle sets the interval size *I* to I_{max} and re-executes the steps from step2.
6. If Trickle detected an inconsistent message, it resets *I* to I_{min}, if it was not already set to I_{min} and starts a new interval as in step 2.

An illustration of Trickle's operations is shown in **Figure** 3-4 under a network of three nodes: N1, N2, and N3 with a redundancy constant *k* of 2. In **Figure** 3-4, you will notice that it happens that N1 and N2 select randomly to submit their DIOs earlier than N3 after entering a listen period of the half of the interval. Hence, they will proceed with submitting their DIOs as they have received less than *k* messages till their time of submission (i.e. their redundancy counter is less than 2). N3 will suppress its own transmissions in the first interval as it receives 2 messages (from N1 and N2) until that point of time so its redundancy counter is 2 in this case which is not less than *k*. At the start of the second interval, the three nodes will reset their counter *c* and re-run the same logic as in the first interval.

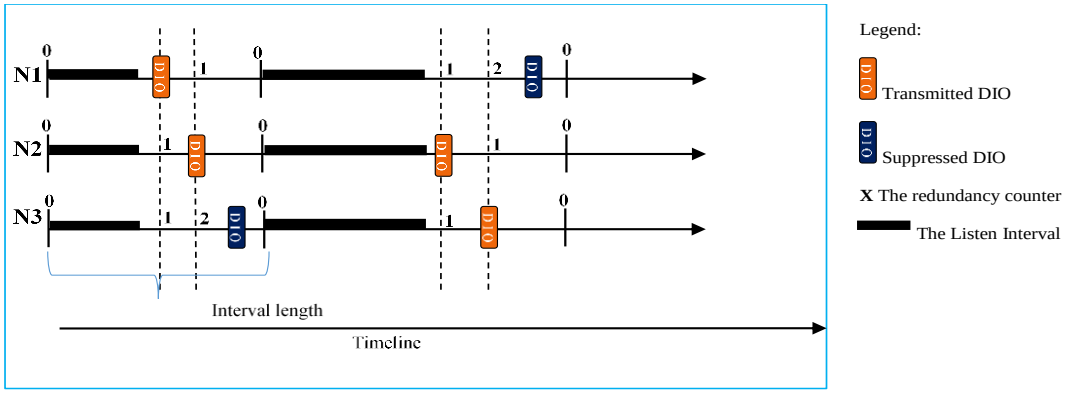


Figure 3-4. Trickle operations with 3 nodes, k=2.

3.4 RPL LIMITATIONS AND DRAWBACKS

As the de-facto standard for routing in IoT networks, a plethora of recent studies have evaluated RPL performance reporting several limitations that need to be addressed. In the next subsections, an elaboration on the key limitations reported in the literature related to RPL's OFs, downward routing and routing maintenance is given. A summary of these limitations is presented in **Table 3-II.**

3.4.1 LIMITATIONS RELATED TO STANDARDIZED OBJECTIVE FUNCTIONS

In this section, the issues related to RPL OFs are discussed including the single-path routing, the under-specification of metric composition, and the implicit hop-count impact.

3.4.1.1 Single-path routing

In RPL, once a preferred parent has been selected, all traffic will be forwarded through this preferred parent, as long as it is reachable, without any attempt to perform load balancing among other available parental candidates [27][29][30]. This behavior may drain the power of overloaded parents leading to network disconnections and unreliability problems, as it is likely that overloaded nodes will die earlier or drop packets due to the overflow of their buffers [56][57][59]. For instance, the authors in [56] reported that RPL relies solely on a single quality metric such as the ETX in path construction which may result in a single node with a good transmission quality being selected by large number of neighbors as their preferred parent. This

may lead to congestion at the preferred parent and buffer overflow enforcing it to drop a large number of packets. The study in [57] reported simulation experiments that show how the reliability of heavy-traffic networks can be affected negatively under the single path nature of RPL. The study found that the main reason behind the reliability degradation is that some nodes are highly overloaded leading to packets drop at these overloaded nodes. The authors in [59] demonstrated how the single-path routing of RPL may lead to building an unbalanced routing tree creating hot spots (i.e., overloaded nodes) in the network. Such hot spots will deplete their energy resources much faster than other nodes, an issue that can be mitigated by introducing the load-balancing techniques as they reported.

Table 3-II. The summary of major RPL's limitations

The problem	The module	Brief description	Side effects/ pitfalls
Incompatible modes for *downward* routing	Downward routing	The *downward* MOPs are not specified to understand each other.	Forwarding failure and network partitions
Memory limitations	Downward routing Storing mode	Each node must maintain the routing entries of all nodes in its sub-DODAG which might not be possible for memory-constrained nodes	Memory overflow jeopardizing reliability and scalability
Long source headers in the non-storing mode	Downward routing Non-storing mode	Transmitted packet must carry the addresses of all nodes to destinations	Higher overhead jeopardizing reliability and scalability
Under specification of DAOs emission	Downward routing	when a node should transmit its DAOs is unspecified	May lead to inefficient implementations
Listen-only period	Routing maintenance timer	A node must wait for the half of the interval before transmitting a routing update	Slow convergence and load-balancing problems
Suppression mechanism Inefficiency	Routing maintenance timer	Node must suppress a specific routing update should it hear that a certain number of the neighbors have transmitted the same routing update	If not configured correctly, forming sub-optimal routes
Single-path routing	Objective Function	A node keep forwarding traffic to its preferred parent with no attempt of load balancing	No load balancing affecting negatively both reliability and energy efficiency.
Under-specification of metrics composition	Objective Function	No guidelines are specified on how to combine several metrics	Jeopardizing the capacity of the protocol to get the benefit of combining several metrics
Implicit hop-count impact	Objective Function	A path with better global quality (due to its less number of hops) may contains one or more links with critically low-quality links that undermine its apparent quality	May impact negatively any performance aspect

3.4.1.2 Under-specification of metrics composition

RPL supports the use of multiple metrics for routing with the possibility of optimizing the routes based on combining several metrics, however, no guidelines are provided on how such combination should be achieved [88][89]. Hence, relying on a single routing metric in the OF may satisfy one application requirement, yet also violate another [89][90][91][92]. For example, while the ETX routing metric allows the protocol to choose the most reliable path, it may result in early network partitioning due to the absence of a load-balancing mechanism that might protect vulnerable nodes from exhausting their battery power. The problem of unbalanced traffic is exaggerated by the fact that standard RPL permits forwarding of traffic through the preferred parent only, even in the case when several candidate parents are available [88].

3.4.1.3 Implicit hop-count impact

In RPL OFs, the routing cost of a specific path (i.e., rank of a node) is calculated by adding up the cost of its constituent links as explained previously under OF0 and MRHOF. However, the number of hops can have a misleading effect on the final cumulative cost/rank in that longer paths can appear more costly, even though quality of the constituent links is quite good [93]. A path with small number of hops has a higher probability of being selected than another path with a larger number of hops (lower global quality) even though the first path might have one or more very low-quality individual links [93].

3.4.2 LIMITATIONS RELATED TO RPL DOWNWARD ROUTES

According to the specification of the RPL standard in [27], it is expected that MP2P traffic pattern will be the dominant pattern in the context of LLNs while other traffic patterns (i.e., P2MP and P2P) are expected to be less common. Adhering to these expectations, RPL optimizes it routes for the upward traffic in a way that requires less overhead and minimized routing state. However, this has been achieved at the cost of somewhat inefficient downward

route construction in terms of control overhead, routing state and path stretch [8][39][94][95][96][97], resulting in some issues as follows:

3.4.2.1 Incompatible modes for downward routing

Although RPL supports two different modes for downward traffic (i.e., storing and non-storing), the standard specifies that RPL-compliant deployments should use either the non-storing mode or the storing mode within the same instance [27][94]. Hence, when nodes belonging to different instances running different modes of operation meet in the same RPL network, RPL permits nodes from one instance to join the other instance only as leaf node which gives the rise for several interoperability problems. For instance, when a node from one instance located in the middle of a forwarding path joins another incompatible one as a leaf, but is the only available next-hop to the DODAG root [94][95]. Nodes downstream of the new node cannot now communicate with the root through it, since the leaf is not allowed to operate as a router and the network is thus partitioned in both the upward and downward directions. One solution is to relax the restriction and allow nodes with different modes of operation to join incompatible instances as routers [94][95]. However, a forwarding failure may still occur in downward traffic as a router operating in storing mode will have no capacity to understand the source header of a packet sent by a non-storing peer [94][95].

3.4.2.2 Memory limitations in the storing mode

RPL requires that every node running in storing mode must maintain the routing state for all nodes in its sub-DODAG. Although RPL is designed specifically for small and limited-memory sensor nodes, the protocol has the ambition to handle dense networks comprising up to thousands of nodes. In such high density networks, it is highly likely that the routing state that needs to be maintained will overflow the storage capacity of constrained nodes [96][97]. An overloaded node will be unable to accommodate all the routing entries required in its

routing table, rendering several destinations in its sub-DODAG unreachable from the root point of view enforcing it to drop the packets destined to unreachable destinations [96][97].

3.4.2.3 Long source headers in the non-storing mode

In the non-storing mode of RPL, the root is required to attach a source route header for each transmitted datagram in the downward direction. However, RPL is designed to operate on link layers with a Maximum Transmission Unit (MTU) of 127 bytes. Out of the 127 bytes available for the physical layer frame, a maximum of 46 bytes are reserved for the L2 header, a minimum of 2 bytes for the compressed IPv6 fixed header, and a fixed header size of 8 bytes for the attached source route. Taking this into account, only 71 bytes remain for the L3 datagram payload. Thus, a maximum of four hops in the source route header are possible as each IPv6 address has a fixed length of 16 bytes without compression. The compression techniques mentioned in [6][7] can allow for up to 70 hops in the source header; however, as LLNs require IPv6 auto-configuration, a maximum of 8 bytes can be taken out of any compressed IPv6 address allowing for a path length of maximum of eight hops from the source to the destination. This imposes a tight constraint on multi-hop transmission.

3.4.2.4 Under specification of DAO emission

A key issue in constructing RPL downward routes is that the timing of DAO transmission is not explicitly specified. This under-specification of DAO timing may lead to conflict and inefficient implementations of the protocol, consequently harming its performance [8]. For instance, the study in [35] has opted to transmit DAO messages periodically every 5 seconds, significantly increasing the control overhead compared to the ContikiRPL implementation [127] which transmits DAO messages based on the Trickle timers of DIOs. A conservative timing approach may lead to DAOs not being transmitted before old routes expire, impacting negatively the data-plane reliability [8]. Hence, an implementation that does not guarantee the receipt of all DAOs from intermediate routers along a path would render the root unable to

calculate the source route for that destination [8]. This is because the accurate calculation of a source route relies on all route segments advertised in the DAOs of its ancestors, up to the DODAG root. Here, the root would again have no option but to drop all packets for the affected unreachable destination.

3.4.3 LIMITATIONS RELATED TO THE ROUTE MAINTENANCE (TRICKLE TIMER)

As discussed above, the RPL standard specifies that Trickle must be used for routing information exchange and maintenance. Relying on Trickle has given rise to some issues as presented next.

3.4.3.1 Listen-only period

A key issue in Trickle is the introduction of listen-only period in the first half of each Trickle interval, *I* [48][49]. The goal behind the listen period is to solve the so-called short-listen problem in asynchronous networks [31]. In a network with no listen-only period, a node may start sending its current DIO very soon after starting a new interval, a behavior that may result in turning down the suppression mechanism in the current and subsequent intervals, leading to significant redundant transmissions and limiting the algorithm scalability [31]. However, the listen-only period comes with its own shortcomings. Firstly, the period imposes a delay of at least *I/2* before trying to propagate the new information. In an *m-hop* network, an inherited delay will progressively accumulate at each hop resulting in an overall delay proportional to the number of hops [49]. Secondly, the listen-only period may result in uneven load distribution among network nodes with some nodes transmitting less than others do during the operational time [48] [49]. In the worst-case scenario, the transmitting period of a node may substantially overlap with the listen-only period of a neighboring node, preventing the former from sending for a long time. A key issue here is that the blocked node may be the one whose transmission is vital for resolving network inconsistences [48]. Furthermore, the absence of

load balancing among Trickle nodes may render some routes undiscoverable, even though the undiscovered routes might be more efficient than those already active in the network [48].

3.4.3.2 Supression Mechanism Inefficiency

Another issue with the Trickle algorithm is related to its suppression mechanism. In order to lessen the control overhead in the network, Trickle suppresses the transmissions of control messages that seem to be redundant. It does so by counting the number of consistent messages that are received within a specific window and, then, when this number surpasses a pre-configured redundancy constant *(k)*, it suppresses any further propagation of such messages. However, studies have reported that the optimal setting of the redundancy constant is not a trivial task and relies greatly on the application scenario, in addition to some issues that may emerge if configured incorrectly [48][53]. For instance, it was shown in [48] that, if the redundancy constant is not configured correctly, the suppression mechanism might result in sub-optimal routes negatively affecting the reliability of the network, especially in heterogeneous topologies with regions of different densities. This is attributed to the fact that Trickle is originally designed to disseminate code updates, which are quite similar in the context of reprograming protocols. However, this is not the case in the context of routing as two routing update messages originated from different sources may carry different routing information and thus "suppressing one transmission or another is not always equivalent" [48].

3.5 RPL'S ENHANCEMENTS: PROSPECTS AND PITFALLS

In this section, a survey of the enhancements and extensions made to RPL since its introduction in relation to its OFs, downward routing, and routing maintenance is presented providing an in-depth analysis of such extensions highlighting their key weaknesses and pitfalls. In particular, a survey of the extensions of RPL's OFs (Section 3.5.1), the extensions targeting RPL's downward routing (Section 3.5.2) and, finally, the extensions targeting RPL's routing maintenance (Section 3.5.43) is provided. For a quick reference, Table 3-III illustrates

OF extensions and their weaknesses whereas Table 3-IV shows RPL's downward and routing maintenance extensions with their key limitations. A classification of various enhancements is illustrated in **Figure** 3-5. As observed in **Figure** 3-5**,** the enhancements reported in the literature have targeted mainly three modules of the RPL standard, namely, routing maintenance, the objective function, and the downward traffic operations. Relating to the routing maintenance, the studies strived to address the inefficiency of suppression mechanism used by RPL and the side-effect of its listen-only period. Pertaining to the downward enhancements, the research efforts based their enhancements on either using multicast techniques or combining the two modes of RPL. Combining more one than a metric or employing multipath routing were the main techniques used by the research community to address issues of RPL's objective functions.

Table 3-III. The RPL's OF enhancements and their weaknesses and pitfalls

Ref.	Metrics	Multi path	Type of Metric composition	Brief description	Limitation Addressed	Drawbacks	Minimum DIO Size increase (in bytes)
[88]	HC and PFI or HC and RE	NO	Lexical and additive	Combines HC and PFI for better detection of malicious nodes. Also combines HC and RE for load-balancing	Under-specification of metrics composition	No real testbed experiments Very low-quality paths still can be selected	+13 or +14
[89]	RE and ETX	NO	additive	Combines RE and ETX for load-balancing	Under-specification of metrics composition	Only up to 6 nodes for evaluation. Very low-quality paths still can be selected	+14
[90]	RE and ETX	NO	Lexical	Combines RE and ETX for building reliable and energy-efficient topology simultaneously.	Under-specification of metrics composition	No real testbed experiments Very low-quality paths still can be selected.	+14
[91]	Transmit power, Energy and ETX	NO	additive	Combines RE and ETX for enhancing reliability and energy-efficiency with a mechanism to lessen the impact of highly depleted nodes.	Under-specification of metrics composition	Claimed reliability not reported nor justified No clarification on how DIO intervals selected.	+19
[92]	HC, energy, ETX and delay	NO	Fuzzy-based	Combines hop count, energy, link ETX and delay to satisfy the most important requirements.	Under-specification of metrics composition	Higher risk of fragmentation. Very low-quality paths still can be selected.	+28

[93]	HC and ETX	NO	average	Combine the hop count and the ETX by taking the average of ETX to avoid long single-hop problem.	Implicit hop-count impact	The monotonicity property is not satisfied. Suffer from excessive churn.	+14
[98]	RE, ETX , Link color and other context-aware metrics	NO	Lexical and additive	Combines RE, ETX, link color and other metrics to boost reliability while avoiding nodes that have depleted their energy.	Under-specification of metrics composition	Higher risk of fragmentation. Only up to 11 nodes for evaluation. Very low-quality paths still can be selected.	+21
[99]	HC, Number of children and distance to parent	NO	additive	Combine the distance, number of children nodes and the HC.	Implicit hop-count impact	High risk of fragmentation. No indication of the used simulation tool.	+20
[100]	SI and ETX	NO	additive	Introducing new stability metric and combines it with ETX to build more stable and reliable topology.	Under-specification of metrics composition	Less control messages not only indicate stability, it may also indicates unreliable links. Unclear how the SI and EXT are combined.	+13
[104]	Delay, ETX and energy	NO	Fuzzy-based	Combines the delay, ETX and energy to boost stability, reliability and energy-efficiency.	Under-specification of metrics composition	The enhanced stability and the slightly improved delay are not justified. Very low-quality paths still can be selected.	+20
[105] [106]	ARSSI, SPRR and SRNP	NO	Fuzzy-based	Combines ARSSI, SPRR and SRNP to improve reliability with a mechanism to balance between the global quality of a path and the individual quality of its constituent links.	Under-specification of metrics composition	The claim that the proposed metric allows avoiding paths having low-quality links is not fully supported. It is unclear how DIOs have been incorporated into the link estimation calculation. A small number of nodes (10 nodes).	+17
[108] [109]	Traffic, ETX, Data-rate, Transmit power and RE	YES	additive	Designing a new metric called ELT and using multipath forwarding for the aim of balancing the energy consumption.	Single-path routing	Higher risk of fragmentation. The monotonicity property is not satisfied.	+29
[110]	N/A	YES	N/A	Uses multiple paths during congestion as a way of overcoming such a congestion.	Single-path routing	More overhead due to the new control messages. It is unclear how the congestion threshold is set.	
[111]	DELAY ROOT, Received packet number and ETX	YES	additive	Designing a composite multipath routing metric to mitigate congestion resulting from the sudden events in the emergency scenarios.	Single-path routing	Higher risk of fragmentation. No real testbed experiments.	+23

[112]	ETX and RE	YES	Lexical	Design a new ETX-based and then combine it RE to improve reliability and load balancing.	Single-path routing	No reliability metric is used for comparison purposes. The monotonicity property is violated. The simulation tool used for evaluation is undisclosed.	+14
[113]	Remaining battery voltage	NO	N/A	Introducing the remaining battery voltage as a new metric with a hysteresis of 5% to prevent excessive churn	Under-specification of metrics composition	Only up to 7 nodes are used for evaluation. No justification of the higher churn experienced by OF0.	+7
[113]	PD, NC, (LC) and energy	Yes	additive class-based	Combining four weighted metrics and using virtualization and SDN to supports multiple classes of traffic.	Single-path routing	One-hop communication is supposed which is unrealistic. No clarification on how DIOs are communicated in the NONSDN-based OMC-RPL. The reporting interval of the SDN-based OMC-RPL is not given.	+23
[114]	BDI, PER and ETX	NO	additive	Combine BDI, PER and ETX with the focus on excluding highly depleted nodes in terms of energy.	Under-specification of metrics composition	The superiority of proposed OF over ETX-based OF in terms of PDR seems unjustifiable. No real testbed experiments.	+19

Table 3-IV. The RPL's core operations enhancements and weaknesses

	The name	The module	Brief description	Limitations Addressed	Drawbacks
[94] [95]	DualMOP-RPL	Downward routes	Allows nodes operating different MOP within one physical network to understand each other and cooperate as single connected network.	Incompatible modes, memory limitation and long source headers (Section IV-B-1, 2, and 3)	Inherits the limitations of the non-storing mode in terms of higher fragmentation risk and the storing mode memory overflow. Only up to 25 nodes are used for evaluation, not an example for a large scale-network.
[96]	Memory-efficient RPL (MERPL)	Downward routes	Combining the non-storing and storing modes of operation to carry out the forwarding decisions in the *downward* direction.	Memory limitation and long source headers (Section IV-B-2, and 3)	Unclear how to set the value of the pre-specified factor N. Unpopular simulation tool is used for evaluation.
[97]	D-RPL	Downward routes	Using the multicast to overcome the memory limitations in the storing mode of RPL, when the node's memory overflows.	Memory limitation (Section IV-B-2)	Multicast added more complexity and sometimes it might be counter-productive.
[48]	Trickle-F	Routing maintenance	Gives the node a priority to send its scheduled DIO based on its recent history of transmission.	Suppression Mechanism Inefficiency (Section IV-C-2)	Slow convergence time due to the listen-only period.
[49]	Optimized-Trickle (Opt-Trickle)	Routing maintenance	Allows nodes to pick the random time, *t,* from the range *[0, Imin]* in the first interval.	Listen-only period (Section IV-C-1)	Unrealistic MAC protocol with 100% duty-cycle is used for simulation experiments. Fast convergence time, however, moderate in lossy networks as there is listen-only period in the subsequent intervals.
[53]	adaptive-k	Routing maintenance	Allows each node to tune its redundancy factor dynamically based on the number of its neighbors	Suppression Mechanism Inefficiency (Section IV-C-2)	The number of DIOs may not reflect correctly the number of neighbors. Slow convergence time due to the listen-only period.
[55]	Trickle-offset	Routing maintenance	Calculates the redundancy factor as a function of node degree.	Suppression Mechanism Inefficiency (Section IV-C-2)	Adding more a complexity by introducing two new configuration parameters. Slow convergence time due to the listen-only period.

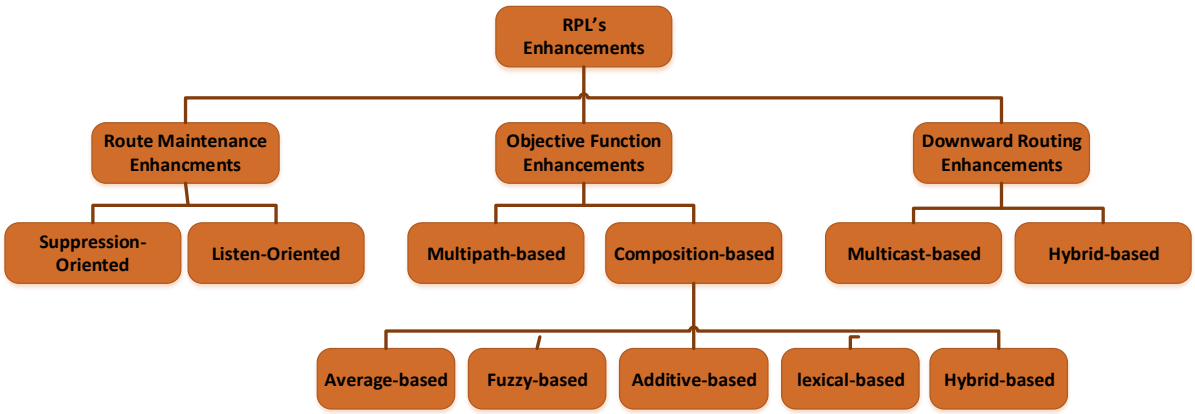


Figure 3-5. RPL's enhancements classification

3.5.1 OBJECTIVE FUNCTIONS ENHANCMENTS

Several efforts have been made to fill in the gaps presented in RPL's objective functions. Most of these efforts have focused on designing OFs with a composite routing metric to fulfill conflicting routing requirements in the same application domain. Introducing multipath routing as a means of enhancing the efficiency of OFs is the focus of another class of studies. The next section will discuss these proposals.

3.5.1.1 OF Enhancements Based on Metric Composition

Several research studies have been carried out into overcoming the problem of the under-specification of metrics composition of the RPL standard. Multiple mechanisms are proposed to combine the respected metrics including lexical, additive, hybrid, and fuzzy based composition. In the lexical composition, the selection of the parent is done based on the first composition metric and if two parents have equal values for the first composition metric, the second composition metric is used to break the tie [88]. In the additive composition, the weighted values of participating metrics are added to produce one composite value. Then, the selection of the preferred parent is performed based comparing the parents' composite metric values. In the hybrid composition, both the lexical and the additive techniques are used to combine between two or more metrics. The fuzzy based-composition is based on the concepts and principles of fuzzy logic. In the following, a discussion of these extensions is given.

3.5.1.1.1 HYBRID COMPOSITION ENHANCEMENTS

The authors in [88] propose lexical and additive composition techniques that combine two routing metrics to optimize multiple performance aspects. They point out that the monotonicity property of the combined metric must hold to ensure a loop-free routing protocol. When using additive composition, the two component metrics must hold the same order relation to ensure validity of the composite metric. However, this restriction is not necessary when using lexical composition. The work proposes a combined Hop Count (HC) and Packet Forwarding Indication (PFI) metric, to construct shorter paths that avoid nodes acting maliciously or selfishly. Simulation results have shown that lexical combination of these two metrics gives better detection of misbehaving nodes and selection of reliable paths while showing comparable latency in comparison with the hop count metric only. The authors also show that combining Residual Energy (RE) and hop count metrics either in an additive or lexical manner results in better energy load distribution among nodes in comparison with hop count only.

The Scalable Context-Aware Objective Function (SCAOF) for agriculture low power and lossy networks is proposed in [98]. SCAOF combines the metrics of remaining energy, ETX, availability information, and hardware robustness (number of restarts) and affordable workload (the tendency of node to consume energy), in a way that guarantees the selection of a reliable path while avoiding nodes that have depleted their power reserves. This study also introduces the notion of ETX_Threshold and RE_Threshold in order to allow for a configuration that is consistent with specific applications [98]. The proposed objective function is evaluated by means of simulations and testbed experiments, and compared to RPL-ETX (the exact used OF is unclear) in terms of packet loss rate, routing table size, Round-Trip Time (RTT), overheads, path hop distance, packet delays, network churn, and network lifetime. It is shown that the

developed protocol can reduce network churn, prolong network lifetime, and enhance the quality of service of A-LLNs applications.

3.5.1.1.2 ADDITIVE BASED COMPOSITION ENHANCEMENTS

The study in [89] again addresses the issue of RPL relying only on a single metric: energy or reliability. The authors highlight the problems of unbalanced traffic and consequent uneven energy consumption distribution among network nodes in RPL. The study articulates that using ETX as a single metric in a RPL network would result in excessive use of some paths, especially those with high delivery rates. This excessive use of good-quality paths will result eventually in network partition and reduce the overall lifetime of the network [89]. If energy is selected as the sole routing metric, on the other hand, the reliability of the path might be impacted negatively. To balance energy consumption of nodes while providing highly reliable paths, the study proposes a weighted energy-oriented composite metric that takes into consideration a node's residual energy in addition to ETX. The study results show that energy consumption is balanced to some extent by the proposed technique, which enhances network lifetime by up to 12%.

An Energy Efficient and Reliable Composite Metric for RPL Networks is proposed in [91]. This composite metric takes into consideration both the reliability, represented by the ETX metric, and energy efficiency to balance energy consumption among nodes and enhance the network lifetime. The proposed metric is called the Lifetime and Latency Aggregateable Metric (L2AM). In particular, a node running L2AM, first combines the transmission power of the link and a node's residual energy using an exponential function to produce what is called the primary metric. The ETX metric is then multiplied by the primary metric to get the composite metric overall cost: this is what must be minimized when selecting the preferred parent. For evaluation purposes, the proposed metric is compared to ETX RPL in terms of

network lifetime and remaining energy. The results have shown that L2AM outperforms ETX RPL by up to 56% in terms of network lifetime.

In [99], the authors highlighted the fact that relying on hop-count only in calculating the ranks of nodes may result in constructing paths characterized by long physical distances. As the transmitter energy consumption is directly proportional to the square of the distance between communicating nodes, it may lead to routes that suffer from higher power consumption rates. The authors propose a new composite metric based on the distance between a new node joining the DODAG and its potential parent; the number of children that the potential parent has, and the hop count metric. The new framework is compared to OF0 and to the Karkazis [88] composition metric in terms of device longevity and power consumption. It is shown that the proposed framework manages to reduce power consumption significantly and enhance the longevity of the DODAG.

The instability and unreliability issues of RPL are considered in [100]. The authors report that RPL may suffer from frequent route changes that may affect network performance negatively. They assert that even though several metrics are defined for RPL, there is not a metric that represents the stability of nodes. Thus, a new stability metric, referred to as Stability Index (SI), is proposed, to overcome this issue. The new metric relies on the transmission rate of control messages to estimate the stability of links. The SI is measured at each node by adding up the weighted number of DIO, DIS and DAO control messages transmitted during a specified interval (the Hearing Window) and dividing the sum by the size of the interval [100]. The weighting is used to give each type of control message a different importance. The study suggests combining the new metric with ETX to boost protocol reliability further. The proposed and combined metrics are evaluated using NS2 simulations and compared to RPL with hop-count and ETX metrics, in terms of control message overhead, latency and packet delivery ratio (PDR). It is shown that RPL with the proposed metric reduces significantly the

control plane overhead by up to 90% and the average number of transmissions by up to 50% compared to RPL hop count and ETX. In addition, the simulation results indicate that SI-RPL and SI-ETX-RPL outperform both ETX-RPL and HC-RPL in terms of PDR and that the amount of enhancement depends on the size of the hearing window. On the other hand, SI-RPL and SI-ETX-RPL have slightly longer latency compared to HC-RPL as they prefer more stable and reliable paths even at the cost of more hops.

A new RPL objective function, named Improved RPL, has been proposed in [101] with the aim to boost survival time of the network and reduce energy consumption. Improved RPL introduced a new composite metric called the life cycle index (LCI) which takes into consideration a node metric such node energy and a link metric such as link quality. The index also incorporated a congestion detection factor into its logic to detect and disseminate congestion information. The node and link metrics were then combined with the congestion factor to calculate the ranks of nodes. The node with the largest LCI is selected as the next-hop towards the DODAG root. In order to mitigate the congestion, another node is selected as a backup parent and traffic is forwarded via both parents under congestion [101]. It was shown [101] by means of simulation experiments that the proposed OF has outperformed RPL in terms of energy consumption, delay, PDR, churn, load-balancing and network lifetime. As several parameters is used to calculate the ranks of nodes, this OF is susceptible to the risk of DIO fragmentations. In addition, the X-MAC duty-cycling protocol, which is an obsolete protocol, was used at the MAC layer casting doubts on the efficiency of the proposed scheme under state-of-the-art MAC protocols.

3.5.1.1.3 LEXICAL-BASED COMPOSITION ENHANCEMENTS

An energy-aware objective function for the RPL protocol is introduced in [90], referred to as EAOF. In this study, the authors highlight the issue that current RPL objective functions do

not use energy-based metrics. They proposed combining the ETX metric with the residual energy of nodes in order to build a topology that is energy-efficient and reliable. A node running EAOF must first select a subset of nodes with the lowest ranks, in terms of ETX, from its neighbors. Then, the node with the maximum residual energy is selected from this subset as a preferred parent. The parameter MAX-ETX is introduced to limit the size of the ETX-based subset from which the preferred parent is selected, according to the application requirements. In addition, the parameter MIN_ENER is proposed to introduce a hysteresis value when switching parents based on energy, in order to ensure network stability. The study uses the popular Cooja [125] simulator with Contiki OS [126] to validate the proposed objective function and compare it with the ContikiRPL implementation of the ETX-based MRHOF, in terms of Packet Reception Ratio (PRR), energy efficiency and network lifetime. It is shown that EAOF significantly improves the network lifetime and balances energy consumption compared to RPL MRHOF, with a negligible impact on reliability. A slight degradation in PRR is attributed to EAOF sometimes favoring balanced paths over high quality paths to enhance network lifetime.

3.5.1.1.4 CROSS-LAYER BASED COMPOSITION ENHANCEMENTS

A cross-layer based composition is proposed in [102] named RPL-SCSP, which combines the ETX and Queue Load metrics aiming at providing the network with QoS support. The RPL-SCSP proposes that the selection of parent is firstly done based on the number of packets in the queue (*nqpacket*). The parent who has *nqpacket* between one and *s*, a pre-specified threshold, should be selected as the preferred parent. When multiple parents have *nqpacket* between one and *s*, then the selection of preferred parent is done based on the ETX values. The selection of preferred parent based on ETX values is also applied when all parents have *nqpacket* less than one or greater than *s*. The speed at which the values of ETX and *nqpacket* change depends

largely on the network topology and its operating conditions. For instance, a network characterized by a high density may have such values change abruptly, especially when experiencing interference from other resources and vice versa. It was shown by means of simulation experiments that RPL-SCSP has reduced the end-to-end delay and enhanced the network lifetime.

Another cross-layer and fuzzy-logic OF for LLNs is proposed in [103] with the aim to satisfy the reliability and latency requirements by combining ETX and latency metrics. The proposed OF was implemented within Contiki OS and compared to the MRHOF of RPL in terms of PDR, latency and stability of the topology using Cooja simulator. It was shown [103] that the new OF has outperformed the MRHOF in terms of stability latency and PDR. An issue with the simulation setup adopted in this study is the use of the null radio duty cycle driver as the duty-cycling protocol which is unrealistic in the context of LLNs. In addition, just up to 12 nodes were used to conduct the simulation experiments which does not reflect the real-life deployments even in small home-automation networks.

3.5.1.1.5 AVERAGE-BASED COMPOSITION ENHANCEMENTS

The study in [93] addresses the long single-hop problem introduced when RPL relies on a single metric such as hop count or expected transmission cost, in large networks. The authors report that, since the ETX metric adds up the ETX values of the nodes along a routing path, the number of hops (rather than the quality of transmission) tends to have more impact on the calculated rank. Therefore, a node will tend to select the path with a small number of hops because this passes through fewer nodes and accumulates a relatively smaller total ETX [93]. Hence, the calculated ETX rank for a path with fewer hops tends to be smaller, even when such a path has constituent links with quite poor transmission quality. In a large network, a long single-hop path with bad transmission quality can restrict the whole network affecting

negatively its reliability. To overcome this problem, the study proposes combining the hop count and the ETX metrics to produce a composite metric called PER-HOP ETX. The rank is calculated based on the cumulative value of ETX along a path divided by the number of hops on that path. The new metric is evaluated using Cooja [125] and compared with both the MRHOF and the OF0 objective functions. The results indicate that PER-HOP-ETX improves PDR in dense networks while reducing power consumption and latency.

3.5.1.1.6 FUZZY-BASED COMPOSITION ENHANCEMENTS

In [92], the authors also highlighted the problem of relying on a single-metric objective function. They further pointed out that even combining two routing metrics might be insufficient to address the requirements of multiple applications as the performance objectives may vary so widely. In addition, combining just two routing metrics may enhance the network performance of the parameters associated with these, but at the expense of negatively affecting other parameters. For example, considering the ETX and latency metrics may help the RPL network to discover more reliable paths with low delay, but may lead to battery depletion due to the overuse of some routers. Thus, they assert that there is a need to design a holistic objective function that combines multiple routing metrics to optimize all significant parameters simultaneously which seems a far reachable goal or even impossible. Hence, they propose a fuzzy logic approach, the Fuzzy-Logic OF (called thereafter OF-FL), that combines four representative routing metrics. OF-FL combines hop count, node energy, link quality and end-to-end delay to satisfy the most important requirements of an LLN application. It is shown [92] that the proposed OF-FL has a tendency to reduce average hop count in comparison with the MRHOF in dense networks. In addition, OF-FL has a much better performance in terms of PDR than OF0, and almost the same PDR as MRHOF with ETX. Furthermore, the results indicated that OF-FL has a better load distribution among nodes leading to a more balanced

energy consumption than F0 or MRHOF with ETX. Finally, OF-FL demonstrates the lowest average end-to-end delay for nodes on the edge of network while the delay is comparable with the standardized OFs in other cases. However, OF-FL experiences higher churn compared to MRHOF with ETX.

Another fuzzy-based approach to combining routing metrics is introduced in [104]. The authors used a two-stage fuzzy process to combine three linguistic variables (routing metrics), namely, delay, ETX and energy. In the first stage, the delay and ETX are combined to compute what they call Quality of Service (QoS). In the second stage, the energy is combined with the QoS metric. The proposed fuzzy-based approach is then evaluated against ETX-RPL using a real testbed network of twenty-eight sensor nodes. The two protocols are compared in terms of packet loss ratio, energy consumption and routing stability (number of preferred parent changes). It is reported [104] that the fuzzy-based approach outperforms ETX-RPL in terms of packet loss ratio by up to 20% and slightly enhances end-to-end delay. In addition, the proposed approach is shown to build a topology of more stable routes with an average of 6.63 parents change per hour compared to ETX-RPL with an average of 43.52.

A third fuzzy-based routing metric is proposed in [105][106] referred to as Opt-FLQERM. This metric considers three link estimation metrics: Average Received Signal Strength Indicator (ARSSI); Smoothed Packet Reception Ratio (SPRR); and Smoothed Required Number of Packet retransmissions (SRNP). These three routing metrics are combined using a fuzzy approach that produces a score from the range [0...100], where 100 is the best quality and 0 is the worst. To select the optimal path, the inverses of the individual link qualities are added and then the path with minimum value is selected. The authors claim that relying on the inverse when selecting the optimal path allows the metric to avoid low-quality links while favoring paths with fewer hops. For evaluation purposes, the proposed routing metric is compared to RPL, ETX-RPL and the four-bit CTP [107] metrics using the well-known Cooja

[125] simulator, in terms of average packet loss, average end-to-end delay, average hop-count and average power consumption. The authors show that their proposal produces the lowest packet loss, the lowest end-to-end delay and the lowest churn (number of parent changes) among the compared metrics. The superiority of Opt-FLQERM over ETX is attributed to the conservative approach used by ETX to estimate link quality: this is based on data traffic, which is only obtained after topology establishment. In contrast, Opt-FLQERM bases its calculation of link qualities on both control and data traffic resulting in an accurate estimation of link quality at the time of topology construction and yielding more stable paths.

3.5.1.2 OF Enhancements Based on Multi-path Routing

In order to overcome some performance issues resulting from single-path based routing in RPL, several multipath forwarding optimizations have been proposed and still other studies have proposed multi-path forwarding approaches that use composite metrics. The authors in [56] propose a probability-based load-balancing multi-path solution for RPL referred to as LB-RPL. LB-RPL achieves load balancing by having each node distributes traffic among its top *k* parents, in terms of rank, based on their traffic load. A parent experiencing heavy load may signal its status by delaying the broadcasting of its scheduled DIO message. This enables child nodes to remove that parent from their top *k* and hence, exclude it from further data forwarding. It is shown [56] by means of simulations that LB-RPL outperforms RPL in terms of PDR, delay, and workload distribution.

The work in [108][109] highlights the advantages of incorporating multipath forwarding schemes into the RPL protocol. Intuitively, the multipath mechanisms have been proven to have a wide spectrum of benefits such as improving fault-tolerance, enhancing reliability, minimizing congestion and improving QoS. The authors propose a multi-path routing mechanism based on RPL in order to allow the protocol to forward traffic to multiple preferred parents. The study asserts that a routing metric must: (1) capture the variations in link quality;

(2) use energy-efficient paths to maximize the end-to-end reliability; and (3) minimize the energy expenditure for those nodes consuming the most energy (the bottleneck nodes). In this regard, a new metric is proposed, referred to as the Expected Lifetime metric (ELT) that aims to balance energy consumption among network nodes and maximize the lifetime of the bottleneck nodes. The network lifetime is defined as the time before the first node dies (runs out of energy). The ELT of a specific node is calculated by: (1) computing the throughput of that node based on its own traffic and also the traffic of its children; (2) multiplying the average number of retransmissions by the calculated traffic; (3) computing the time ratio required for transmission based on the sending data rate; (4) computing the energy consumption based on the transmission power of the radio only; and, finally, (5) calculating the ELT as the ratio between the node's remaining energy and the energy calculated in the previous step. Based on the ELT calculated value, the bottleneck nodes are first identified and advertised along the topology, then a multiple-parents, energy-balanced topology is constructed, in which the traffic is balanced among parents with careful consideration of bottleneck nodes. The proposed protocol is evaluated using WSNet [138] and compared to RPL considering the metrics of: residual energy; the ETX-using-hysteresis objective function; and a linearly combined metric of ETX and residual energy. The experimental results indicate that the proposed multipath ELT has almost the same reliability as ETX although selecting the paths with maximum residual energy. Multipath ELT was also found to enhance routing stability through preventing sudden changes in the parent weight in comparison with standard RPL [109].

In [110], the authors again highlighted the issue of RPL being a single path routing protocol and the inability of standard objective functions to provide multipath routing between source and destination. The ultimate goal of the study is to provide RPL with multipath routing capabilities that will enable the protocol to react efficiently to congestion. The authors propose an extension referred to as a multi-path RPL (M-RPL)[110] that provides temporary multiple

paths during congestion. In M-RPL, the PDR is used by the forwarding nodes to detect congestion. If a forwarding node on a routing path detects that the PDR has decreased below a specific threshold, the node sends a notification to its children, by means of DIO messages, informing them of congestion. Each child node that hears the congestion advertisement message starts multipath routing by splitting its forwarding rate in half. Thereafter, only every second packet is sent to its original congested parent while the others are forwarded to any other parent from its parent list (PT). The proposed protocol is evaluated using Cooja [125] and compared to RPL with MRHOF in terms of energy consumption, latency, and throughput. Their simulation results show that M-RPL has better throughput and lower per-bit energy consumption than RPL, due its splitting mechanism. The results also indicated that, while the delay of M-RPL is initially comparable to RPL, this changes when congestion begins. Initially, M-RPL experiences greater delay as multiple paths are introduced but, when the network stabilizes, delay becomes better than that of RPL [110].

The work in [111] proposes a multi-path forwarding approach based on a composite metric. The authors point out that the two objective functions specified for the RPL protocol, each based on just a single metric, are especially vulnerable to scenarios where a sudden increase in traffic volume introduces congestion, resulting in significant delay and packet loss. The authors propose a congestion avoidance multipath routing protocol based on RPL, referred to as CA-RPL, whose primary goal is to enable the network to react quickly and reliably to sudden events. They have designed a composite routing metric for RPL based on the ContikiMac duty cycle protocol with the aim of minimizing the average delay towards the DODAG root, referred to as DELAY ROOT. Under this metric, a node saves time by first learning the wakeup phase of its candidate parents and then sending the packets to the first awake parent [111]. CA-RPL is a composite multi-path routing metric that combines the new proposed DELAY ROOT with the number of received packets and ETX to calculate the path

weights. Cooja [125] with the Contiki operating system [127] are used to compare the proposed protocol with standard RPL in terms of latency, packet loss ratio, throughput and the packet reception number (PRN) of the DODAG root per unit time. The use of Contiki and Cooja rather the conventional simulators such as NS-2 for conducting the experiments has the special characteristic of experimenting with the code that will run on real IoT devices. The experimental results illustrate that the proposed protocol relieves network congestion resulting in PRN values up to 50% higher, throughput 34% higher, packet loss reduced by up to 25%, and average delay by 30% compared to RPL.

The authors in [112] reported that the ETX metric used in RPL is inefficient in quantifying the quality of links as it only "reflects the quality of a single link". To overcome this issue, the study proposes a link-quality-aware routing protocol for LLNs referred to as LQA-RPL. LQA-RPL calculates the rank of a node based on the quality of links to all its neighbors, which is derived from the ETX and defined as the expected probability of unsuccessful transmissions. If a node has more than one parent in its parent set, the node uses multi-path routing by selecting the parent with the maximum residual energy to act as the next-hop relay node to the DODAG root. LQA-RPL is evaluated and compared to RPL with hop count in terms of PDR, energy consumption, and network lifetime. The reported results indicated that LQA-RPL outperforms RPL in terms of PDR, which is attributed to the higher number of candidate parents. It is also shown [112] that LQA-RPL can balance energy consumption and prolong network lifetime due to its capacity to distribute traffic among multiple candidate parents based on residual energy, prolonging the network lifetime.

The work in [113] has reported a new energy-based OF. The authors propose selecting the preferred parent based on the remaining energy with a hysteresis value of 5% to reduce network frequent changes. The remaining energy is obtained by polling each node to check its battery voltage, which is claimed to be a good indicator as it is "the electric potential energy per unit

charge" [113]. The proposed OF is evaluated using two (basic and extended) testbed deployments, and compared to RPL objective functions (OF0, MRHOF) in terms of packet loss, delay, energy consumption and network churn. In both testbeds, a sink and seven sensor nodes are deployed in a building area differentiated by node position, distance between nodes, RF interference and noise. Based on the obtained results, it is shown that the proposed OF has improved network lifetime by up to 40% compared to RPL OFs. The authors also claim that their OF lowered the delay in the Basic Deployment compared to MRHOF, which was expected, and to OF0, which was not. In explaining the latter surprise, they observe that OF0 suffers from excessive churn and frequent changes in the network topology resulting in higher delays and more energy expenditure. Pertaining to packet loss, the reported results show that the proposed energy-based OF is superior to RPL OF0 but it is outperformed by RPL MRHOF.

The authors in [114] propose an optimization for RPL referred to as Optimized Multi-Class RPL (OMC-RPL) based on virtualization and software-defined networking techniques. The study asserts that standard RPL faces two significant issues when offering QoS. The first is the absence of a holistic and comprehensive objective function. For example, an objective function may enhance delay but at the cost of higher energy consumption as all packets overuse the same paths with the minimum delay. The second issue is that RPL does not support a mechanism for data classification which is a critical component in ensuring the QoS. Thus, a holistic objective function that supports multiple data classes is needed. The steps of OMC-RPL are as follows: first, the nodes send the information required to construct the virtual DODAG to the SDN controller, using one-hop communication; then the SDN controller calculates the ranks of nodes in the network for each traffic class using a custom weighted-metric objective function [114]. The main parameters of the proposed objective function are the Propagation Delay (PD), Node Congestion (NC) and Link Congestion (LC). Energy is considered as a secondary parameter and is thus incorporated into the objective function in a

way that it can be removed or considered as desired. The weight values of the objective function parameters were found using the Particle Swarm Optimization (PSO) algorithm. OMC-RPL is simulated with four different classes of traffic and compared to standard ETX-RPL in terms of end-to-end delay, packet loss, network lifetime and traffic overhead. OMC-RPL then outperforms RPL in terms of end-to-end delay for the class of traffic that requires minimum delay and likewise performs better than RPL in terms of PDR with the class of traffic that requires reliability. It is also found that OMC-RPL reacts better to network failures since it can use a backup parent to replace a failed one. OMC-RPL outperforms RPL in terms of network lifetime by up to 41% and shows better fairness in energy distribution by about 18%. The study also reports the impact of incorporating a SDN controller with OMC-RPL. This reduces the number of exchanged control packets compared to both OMC-RPL and standard RPL by approximately 62% and minimizes the energy consumption by more than 50% compared to standard RPL [114].

In [115], the authors propose a new composite energy-aware routing metric, RERBDI, which aims at enhancing the energy consumption of LLN nodes. The study considers the Battery Discharge Index (BDI) and the Residual Energy Ratio (RER) of nodes for taking the routing decision. The study also defines a new objective function, referred to as OFRBE, which combines the new proposed metric with ETX for calculating the rank and selecting the preferred parent. The study mentioned that using RER as a primary routing metric favors paths with higher average residual energy. BDI was introduced as an additional cost function to favor paths that do not include nodes whose battery energies have been depleted or overburdened nodes [115]. Hence, the protocol avoids selecting paths in which some nodes have low residual energy even though the average residual energy is high. The Cooja [125] simulator is used to compare the proposed scheme to standard RPL in terms of PDR, network lifetime, and energy consumption. It is found that the RER metric outperforms ETX in terms of energy consumption

by avoiding paths with lower average residual energy. However, there is still a chance that nodes with a low power profile could be selected as preferred parents which may reduce network lifetime. This situation is improved in RERBDI, which favors paths with higher average residual energy, while avoiding ones that include nodes with very low energy [115]. The new metric enhances network lifetime compared to RPL with hop-count, ETX and RER, but is slightly outperformed in terms of PDR, as it does not consider link quality. Finally, it was shown that the proposed objective function has exceled in both network lifetime and the PDR compared to RPL with ETX and hop count.

3.5.1.3 Challenges and Pitfalls of OFs Enhancements

Despite the advantages brought about by the proposed objective functions, they do exhibit some limitations and pitfalls that need to be considered in future studies. For instance, although the study in [88] presents a good proposal to distribute energy load among nodes by combining the RE and the hop-count, it does not elaborate on the effect of this combination on network reliability, a critical performance criterion. It is also unclear whether the study uses the aggregated value of the RE metric or a local optimum value. A major issue with the study in [89] is that only up to six nodes are used for the simulation experiments, a number that does not reflect even the setting in a small home-automation network. In addition, the authors did not elaborate on how the composite metric may affect the reliability of the network. The shortcomings of the articles in [88][89] are addressed in [90]. First, the author introduces the parameter MIN_ENER to limit the churn in the network due to energy-related parent switches. Second, the study introduces a reliability-related performance evaluation of the composite metric. However, only 25 nodes were used in the simulation experiments, which means that conclusions reached cannot be generalized to urban or industrial networks which comprise hundreds of nodes.

Although the study in [91] claims that the gain in network lifetime is obtained without affecting network reliability, the study does not report any results regarding the reliability nor does it justify how the authors reach this conclusion. In addition, the authors used their own bespoke simulator for evaluation purposes, which may lack in features compared to well-known simulators such as Cooja [125]. The study reports setting the Trickle timer interval for emitting DIOs to 1 hour. It seems the authors have configured only one interval in their simulations, which is a confusing deviation from the normal operation of the Trickle protocol.

In [98], there is a higher risk of layer 2 fragmentation as DIOs transmitted by nodes running SCAOF need to carry a relatively large poll of parameters in their headers. This represents a serious problem in the LLNs as it increases the probability of errors and packet loss, especially in multipath routing.

A major issue with the metric of PER-HOP ETX proposed in [93] is that the monotonicity property of the combined metric is not satisfied, hence the network might be at risk of forming loops. The work in [99] suffers from the problem that the estimation of a node's positions in real testbed deployments is not a straightforward process and hence live physical distance estimations are likely to be either imprecise (e.g. RSSI) or power-hungry (e.g. GPS) [116][117]. The frequency of control messages (DIOs, DAOs, and DISs) is used in [100] to measure stability of the node and the routing topology but, in some cases, the higher frequency of control messages does not imply higher instability. For instance, a node with a higher number of children will have to transmit a higher number of DAOs than a node with a fewer number. In this scenario, it is clear that the number of children has caused the higher control overhead, and not the instability problem. A more elegant solution is to base the measuring of the instability index on the DIO messages alone.

The fuzzy-based approaches are known to incur greater complexity compared to other approaches, especially when multiple instances exist under the same RPL topology [41]. For

instance, in [92], more than four parameters need to be transmitted within the DIO metric container. Thus, there is a higher risk of fragmentation, which incurs more overhead due to the larger size of DIOs [41]. In [104], the stability of routes is claimed to be the reason of the superiority of the proposed approach; however, no justification is given to explain why the fuzzy-based approach is more stable. The lack of justification also applies to the slightly improved delay. The work in [105] does not clarify how the control traffic messages (DIOs) have been incorporated into the link estimation calculation. Finally, Opt-FLQERM tends to favor shorter paths in terms of hop count which may result in selecting paths containing low-quality, single-hop links.

The implicit signaling through delayed DIO proposed in [56] has no extra overhead, but a lost DIO might easily be misinterpreted as delayed, giving a false indication of higher workload at some nodes. In addition, the long transmission periods of Trickle's DIOs cause slow recovery. Moreover, the protocol may suffer from the herding effect problem by always changing parent set members [57].

In [108], because several parameters must be exchanged (i.e., data rate, retransmission count, throughput, transmission power, and residual energy) to calculate the *rank*, this approach requires high overhead and bigger DIOs, increasing the risk of fragmentation. This represents a problem in LLNs when multipath routing is used as two fragments belonging to the same packet may take different paths, increasing the probability of errors and packet loss. In addition, the monotonicity property does not hold for the ELT metric; hence, the study proposes to use ETX to build the DODAG and the ELT to calculate the *rank* of nodes. This would introduce an extra complexity to an already complex protocol [41] and an extra overhead of at least 29 bytes.

The study in [110] suggests that each child node must report its current forwarding rate to its parent node by means of DAO messages to calculate the PDR. Apart from being an optional

feature in RPL, only used when *downward* paths are needed, DAO messages are costly in terms of overhead and energy consumption as they are transmitted in an end-to-end manner. The proposed protocol in [111] is based on ContikiMac that assumes that all nodes have similar wakeup intervals, which may not hold in all LLN scenarios. In addition, many additional fields are carried in the DIO message which increases the risk of fragmentation.

An obvious issue with the proposed protocol in [112] is that it is compared to RPL with hop count, even though the problem statement focuses on explaining the unsuitability of the ETX metric to quantify the reliability of links. Thus, it would seem more logical to compare the proposed combined metric with ETX as both metrics quantify link reliability. Another issue is that the reported metric, if implemented according to the algorithm shown in the study, would violate the monotonicity property of *rank*, potentially resulting in a loop-prone DODAG topology.

A noticeable issue related to the study in [113] is that only seven nodes are used for the evaluation which again does not even reflect the setting in even a small home-automation network with tens of nodes. In addition, while the increased delay in OF0 was attributed to high churn, no justification was offered as to why OF0 experiences more frequent changes in the topology given the perceived stability of the hop-count metric.

In [114], it is assumed that all nodes are within the range of the SDN controller so that the messages can be communicated by one-hop, but this is unrealistic in the majority of cases. In addition, while RPL uses a Trickle timer for communicating control packets, it is unclear what mechanism is used by the NONSDN-based OMC-RPL (OMC-RPL without SDN controller) for communicating such messages. Furthermore, even for SDN-based OMC-RPL, the reporting interval to the SDN is not quoted, although it could have a big effect on the control plane overhead.

The decadic logarithm (i.e., log with base 10) is proposed in [115] to calculate the BDI (an additional cost inversely proportional to the node's residual energy). The calculated additional cost, based on the node's initial and residual energies, will be very small compared to the node's residual energy due to the use of the decadic logarithm so will have no significant effect on the composite metric's final cost. According to the paper, the additional cost (i.e. BDI) will be calculated by adding up the node's initial energy and current remaining energy, and dividing the sum by the node's current remaining energy that will results in a value of 2 at most. Hence, the logarithmic value of such a number will be very small. Although, the study suggests using different weights to adjust the influence of metrics involved, restricting the weights to be within the interval [0, 1] limits the extent to which the influence of BDI can be adjusted.

In general, although combining two or more metrics may give an application the ability to optimize more than one aspect at a time, it may lead to undesirable consequences if not designed efficiently as indicated below.

Firstly, using multiple composite metrics means that a higher load of information needs to be carried in DIO control messages which in turns increases the risk of layer 2 fragmentation. Apart from consuming network resources such as energy and bandwidth, fragmentation represents a serious problem in the LLNs especially when multipath routing is used, as two fragments belonging to the same packet may take different paths, increasing the probability of errors and packet loss [41]. The risk of fragmentation is more evident in fuzzy-based approaches as they feature greater complexity, especially when multiple instances exist in the same RPL topology. Secondly, in weighted composition, it is usually hard to decide on what weights to assign to the component metrics and whether the assigned values should be static or dynamic according to the context (e.g., time, position). Thirdly, some suggested metrics, such as node position, cannot be easily estimated in real environments [116][117]. Fourthly, the composite metric may fall into the trap of giving one metric such a high priority that behaves

effectively as if it was itself a single metric. Finally, designing a composite metric that violates the monotonicity property should be avoided as it can lead to loops that harm reliability and waste resources [88]. In fact, the monotonicity property should be preserved even with single-based metric OFs.

Unrealistic assumptions related to the operations of the protocol itself or its perceived environment is another pitfall that may lead to false conclusions. Such assumptions include one-hop communication range, building network-wide decisions based on optional features in the standard, and non-duty-cycle and synchronous MAC protocols. For instance, in [100], the frequency of control messages was used to measure the stability of the node and the routing topology but the higher frequency of control messages does not necessarily imply higher instability, and this may mislead the routing decision.

The RPL standard is intended to run on LLNs encompassing thousands of sensor nodes. However, a number of the surveyed enhancements were evaluated on networks comprising less than 10 nodes. The small scale of the test network is inadequate to reach strong conclusions reported or display the advantages of proposed enhancements as results cannot be generalized to large-scale deployments.

Multi-path routing techniques are highly desirable in LLN environments as they have been proven to provide a wide spectrum of benefits such as improving fault-tolerance, enhancing reliability, minimizing congestion, increasing network capacity (bandwidth aggregation), and improving QoS [118][119]. However, multi-path routing techniques do have their own disadvantages that should be considered carefully when designing routing primitives for LLNs. One of the primary concerns is that multi-path approaches introduce greater complexity and overhead. In such techniques, the intermediate nodes are required to maintain the state of multiple routes to a destination; this might be infeasible for memory-constrained LLN devices especially in the case of downward traffic, where a node must store routing entries for all

destinations in its sub-DODAG [119]. The way the data packets are allocated to multiple paths represents another challenge [119]. When fragmentation occurs, fragments of the same packet might be transmitted on different routes raising the need for packet re-ordering. This risk is high if a round-robin traffic allocation is used to distribute traffic among multiple paths based on per-packet granularity [119].

Ensuring full efficiency of multipath routing requires the discovery and the maintenance of network-wide node-disjoint paths, which creates extra overhead and may be infeasible in resource-constrained networks with highly dynamic links and scarce energy resources [120]. Moreover, the broadcast nature of the wireless medium may impede goals of reducing congestion or load balancing due to the route-coupling effect, a phenomenon in the wireless medium that takes place when several paths are located in close proximity causing communication interference and increasing the risk of collisions [120]. Although location-aware routing can be used to mitigate the effect of route-coupling problem by constructing non-interfering routes, the high overhead incurred by such techniques in terms of computational and communication complexity makes them unsuitable for the resource-constrained LLN devices [119].

3.5.2 ROUTING MAINTENANCE ENHANCEMENTS

Several extensions have been proposed to overcome the problems associated with introducing the listen-only period and the suppression mechanism in RPL's routing maintenance primitive as detailed below.

3.5.2.1 Suppression-Oriented Enhancements

The first trial to solve the Trickle algorithm issues in LLN routing is the study in [48]. This reports that suppressing RPL control messages by means of the Trickle algorithm may result in sub-optimal path creation, worsening as the number of suppressed DIOs increases. This behavior is explained by the fact that Trickle is originally designed for propagating the same

piece of information with the least number of messages across a network [48]. However, the DIO messages in RPL are not necessarily identical as the information carried strictly depends on the source of the message. Hence, suppressing one or another is not always the same. To address this issue, an enhanced version of Trickle referred to as Trickle-F, which strives to guarantee a fair multicast suppression among RPL nodes is proposed. Trickle-F gives each node a priority to send a scheduled DIO, depending on how many DIOs it has suppressed recently. The more the node suppresses DIOs, the higher the chance it will transmit in the next interval frame. The proposed enhancement is compared to the original Trickle under RPL by means of simulation, in terms of network stretch, average energy consumption and the distribution of suppressed messages. The evaluation results have shown that Trickle-F reduces the number of nodes with sub-optimal routes compared to Trickle while displaying the same energy consumption profile. This superiority is attributed to the spatial fairness achieved by Trickle-F among nodes.

The work in [53] highlighted the ambiguity associated with configuring the redundancy parameter, *k*, in RPL networks. For instance, the Trickle RFC [32] states that typical values for *k* are 1-5, while the RPL RFC [27] sets the value 10 as default. However, the best value for the redundancy constant is claimed to be between 3 and 5 in the last IETF draft titled "Recommendations for Efficient Implementation of RPL" [54]. Finally, it is recommended in the MPL RFC [121] to set the default value of *k* to one. This shows that the optimal setting of *k* is not a trivial task and relies greatly on the application scenario as well as the network topology at hand [53]. The authors here suggest setting *k* for each node individually based on that node's degree, a mechanism they call *adaptive-k*. They use the number of Trickle messages received during a specific interval as an implicit indication of node degree. By means of simulations and testbed experiments, it is shown that the proposal improves the performance

of RPL through lowering the control-plane overhead while enabling the discovery of more optimal routes.

The authors in [51] have proposed a new route maintenance algorithm, named, I-Trickle as an extension of Trickle-F to improve the load distribution of control traffic in LLN and reduce power consumption [51]. The new proposed algorithm resets the redundancy counter value to zero at the time of transmission or suppression DIO messages rather than at the beginning of each interval as it is currently performed by Trickle. The efficiency of I-Trickle in terms power consumption and PDR was then compared to Trickle-F by means of simulation experiments using Cooja simulator and Contiki operating system. The simulation results demonstrated that I-Trickle outperformed Trickle-F in terms of power consumption while showing comparable PDR profiles [51]. However, I-Trickle did not remove the listen-only period from its operations, thus it is still susceptible to slow convergence time. In addition, no details of the simulation parameters have been provided in the study making it difficult to judge the applicability of the algorithm in a specific environment.

In [52], it has been shown by means of mathematical analysis, that the single redundancy constant adopted by Trickle may result in higher transmission load and consequently higher power consumption rates for those nodes having fewer neighbors. To alleviate this issue, the study proposes an enhancement of Trickle in which each node calculates its own version of the redundancy constant as a function of its degree. Each node having a number of neighbors less than a pre-specified threshold, called the offset, will set its redundancy constant to one. The redundancy constant of other nodes should be set by subtracting the number of neighbors from the offset and taking the ceiling of dividing the result by another predetermined value called the step. It is shown, by simulations, that the proposed algorithm balances the transmission distribution among network nodes in comparison with standard Trickle.

3.5.2.2 Listen Interval Oriented Enhancements

In [49], the authors highlighted the problem of increased latency resulting from introducing the listen-only period in the Trickle algorithm. To address this problem, an Optimized-Trickle, (Opt-Trickle) is proposed. The authors observe that nodes receiving inconsistent transmissions simultaneously will reset their timers (returning to Imin) immediately, thus exhibiting a form of implicit synchronization. Such synchronization in fact eliminates the need for the fixed listen-only period in the first interval and allows the affected nodes to pick a random time *t* from the range [0, I_{min}]. This is the only modification in Opt-Trickle.

The authors in [50] have introduced a new routing maintenance approach as an extension of Trickle named RPL-FL. To reduce the network convergence time, RPL-FL assigns the variable *t* the value *I/2* in the first or the second intervals rather than selecting it randomly from the range [I, I/2]. In addition, RPL-FL starts usually with a higher value of I_{min} to minimize the overhead [50]. To reduce the energy consumption, RPL-FL proposes to skip several intervals at once governed by introducing a special parameter named the *skipped interval*. The Cooja simulator with Contiki OS were used to simulate RPL-FL and evaluating its performance in comparison to the RPL standard. The simulation results have shown [50] that RPL-FL outperformed RPL in terms of several metrics including the PDR, power consumption, overhead, convergence time and network lifetime. However, RPL-FL did not remove the listen-only period rendering the protocol susceptible to slow convergence time in comparison to Trickle extension that removed that period. The slow convergence of RPL-FL can also be worsened by selecting a higher value for the minimum interval I_{min}, especially at the stage of constructing the DODAG.

3.5.2.3 Routing Maintenance Main Challenges and Enhacments Pifalls

In general, there are two routing discovery maintenance schemes in the context of LLNs; proactive and reactive [122][123]. In proactive routing, the process of establishing routes is

carried out in advance and maintained periodically, which induces a large amount of overhead, albeit with minimal forwarding delay [122][123]. On the other hand, in reactive routing, the routes are only discovered when needed, thus suffering from higher delay compared to proactive schemes [123]. Reactive routing schemes have been the preferred option when the network features a high number of mobile nodes whereas proactive schemes are preferred in stationary networks (e.g. LLNs). However, the resource-constrained nature of these networks imposes several challenges on proactive route maintenance. Despite the stationary nature of the majority of scenarios, LLNs do exhibit some dynamicity that may render the network unstable, dictating the need for a rapid and reactive response. Choosing small update intervals has the advantage of faster propagation but with a high communication overhead. Long update intervals, on the other hand, have lower communication overhead but disseminate routing information slowly [31]. To address these problems, Trickle [31] has introduced the notion of adaptive and dynamic interval size. The idea is to start propagating routing information at a high transmission rate and then gradually reduce when the network reaches its steady phase, ensuring rapid propagation and low overhead. Trickle uses the term inconsistency to describe the point in time at which the network must start transmitting at its fastest rate. Although Trickle and its extensions have defined how information exchange should proceed in the consistent and inconsistent states of the network, they fail to define clearly what constitutes inconsistent or consistent transmission in the context of routing.

In addition, although Trickle-F [48] has succeeded to some extent in solving the sub-optimality of constructed routes; the algorithm still suffers from slow convergence time due to the listen-only period. The study pertaining to Opt-Trickle [49] assumes a MAC protocol with 100% duty-cycle, which is neither reasonable nor realistic. Furthermore, Opt-Trickle still has a listen-only period in subsequent intervals, which would contribute to the increased latency, especially in a lossy network where it is not guaranteed that a transmitted multicast message

will reach all of its destinations in its first transmission of the first interval. In [53], it is unclear why the study resorts to the number of messages received at specific node and not the number of actual neighbors to estimate indirectly the network density at that node. Although this method might give an approximately accurate estimation for node degree when the network is characterized by synchronized intervals among its nodes, it may suffer from inaccurate estimation in non-synchronized networks. For instance, in a non-synchronized network, the frequency of transmission may differ significantly from a node currently in its minimal interval to another node currently in its maximum interval. The node in its minimum interval will transmit more frequently, giving the receiver node an impression that it has more neighbors than it actually has, thus affecting the accuracy of the network density estimation. The work in [55] did not demonstrate the impact of the proposed enhancement either on the quality of constructed routes or on network power consumption. In addition, introducing two new parameters, the step and the offset, adds complexity which is better be avoided.

3.5.3 RPL Downward Routes Enhancements

Some effort has been directed at increasing the efficiency of constructing downward routes based on combining both modes of operation (hybrid mode), or using multicast techniques.

3.5.3.1 Hybrid Based Enhancements

For instance, the issue of interoperability between RPL's non-storing and storing modes of operation has been highlighted in [94][95]. To solve the interoperability problem, the authors propose DualMOP-RPL [95] which allows nodes operating in different modes to understand each other and cooperate as a single connected network. In this regard, two major enhancements are suggested on top of RPL: firstly, nodes operating in storing mode should attach source routing headers to transmitted messages so that nodes configured in non-storing mode can understand them. Secondly, all nodes operating in non-storing mode should advertise their destination prefixes in a hop-by-hop manner rather than the end-to-end approach currently

specified. Hence, a router configured in storing mode is able to store the routing information of all other nodes in its sub-DODAG, even when some of its children are configured in non-storing mode. DualMOP-RPL is evaluated using both simulations and testbed experiments and compared to RPL in terms of end-to-end packet reception ratio (PRR). It is shown that DualMOP-RPL outperforms RPL in terms of PRR when the two modes of operation are mixed together in a single network consisting of 25 nodes. This is due to mixed-mode network partitioning in RPL and resultant selection of non-optimal paths.

The authors in [96] aim to mitigating the issue of storage limitation in storing mode. They note that RPL storing mode requires every node to maintain the routing state of all other nodes in its sub-DODAG, and many nodes, especially those close to the root, may not have adequate resources for this. To overcome this issue, the authors propose memory-efficient RPL (MERPL) [96]. The primary idea here is that a node, whose routing entries reach a pre-specified threshold N, should delegate a child in its sub-DODAG to act as its store. The overloaded node should then remove from its routing table all routing entries whose next hop is the delegated child. Next, all those destinations reachable through the delegated child should be advertised to the DODAG root in a separate DAO. A hybrid approach of non-storing and storing modes of operation is employed by the network nodes to carry out the forwarding decisions in the downward direction. To validate MERPL, it is compared to standard RPL in terms of the average number of routing table entries, average path length, and the number of items in the source root. A Python language simulator is used with network sizes of 576 and 1204 nodes. The results show that MERPL does indeed reduce the routing entry storage requirements especially at nodes near the root. The average number of items in a source route is reduced by 61.5% compared to RPL when N is set to 10. MERPL average path length is also shown to be shorter than that of RPL in non-storing mode, but slightly longer than that of RPL in storing mode.

3.5.3.2 Multicast Based Enhancements

A different approach to overcoming storage limitations in RPL storing mode is reported in [97]. Here, it is noted that when a node fails to store a new destination routing entry, it should not propagate the information further, as it will not be able to forward to that destination. A negative effect of this behavior is that a path is partially built but is useless since the destination is unreachable by routers higher in the DODAG, including the root [97]. To address this problem, the authors suggest D-RPL [97], which integrates multicast dissemination into RPL storing mode. Here, any node that fails to announce a destination, either for one of its children or for itself, should first register itself with a special multicast group. Then, the multicast address of this special group can be used by the DODAG root to communicate with such destinations unreachable through normal operation. The multicast can be implemented by any suitable protocol such as MPL (Multicast Protocol for Low power and Lossy Networks) [121] or via the multicast mechanism in the RPL protocol itself. D-RPL is evaluated by Cooja [125] with Contiki [127] and compared to the standard RPL in terms of PDR, radio duty cycle and the end-to-end delay. The simulation results show that D-RPL yields significantly better performance in terms of PDR with a 6-fold improvement compared to ContikiRPL. Both protocols have comparable performance in terms of average duty cycle when the number of nodes is less than 60 but above this size, D-RPL has a higher average duty-cycle due to its higher delivery rates. The average end-to-end delay also increases in D-RPL compared to RPL, but this is attributed to the forwarding mechanism in SMFR that opts to delay packet forwarding at each hop for a specific time to avoid collisions. Finally, it is concluded that there is a higher cost in terms of delay and average duty cycle to deliver packets using D-RPL, "but this cost is only paid for packets that would otherwise not be delivered at all" [97].

3.5.3.3 Downward Routes Challenges and Enhacments Pifalls

The development of efficient downward routing schemes for RPL faces two primary challenges: firstly, memory limitations constrain the number of entries that can be stored in a node's routing table. Secondly, packet size is limited by the underlying communication technology. The first restricts the number of nodes that can be accessed by the root (table-driven) while the second limits the number of hops that can be inserted into the IP packet header (source routing) [95][96][97]. This is exacerbated by the fact that RPL is a single-path routing protocol which prevents nodes from benefiting from the combined capabilities of multiple parents.

Although [95] propose combining the table-driven approach (storing mode) and the source routing approach (non-storing mode), both are subject to the storage and hop constraints. As discussed above, the use of multicasting proposed in [97] may only be beneficial in limited cases. In other cases, using multicast will be just counterproductive as it will harm the normal traffic efficiency of those un-overflowed nodes. In addition, all suffer from some weaknesses as explained below.

For instance, the authors in [95] use only 25 nodes for evaluation purposes, which is insufficient to prove the superior performance of DualMOP-RPL in large-scale industrial or urban networks that comprise hundreds of nodes. In addition, although enabling interoperability between RPL modes enhances performance, problems still occur due to the limitations of the two modes themselves. For instance, enabling interoperability does not solve the issue of long source headers in the non-storing mode, nor the issue of memory-overflow in storing mode. In [96], other than the number of nodes, no other simulation parameters are reported and, in particular, there is no clear specification of how the value of N should be set.

In [97], although it is claimed that the additional cost "is only paid for packets that would otherwise not be delivered at all", this may only hold true if we assume that all node routing

tables overflow at the same time. This is not the case if the nodes experience overflow at different times because such nodes then flood the network with multicast packets, negatively affecting the flow of data from all nodes, including those not currently experiencing overflow. The negative affect is, therefore, not limited to packets that would otherwise not be delivered at all.

3.6 RPL'S IMPLEMENTATIONS AND RESEARCH TOOLS

Having discussed and analyzed the literature review related to the LLNs and RPL's concepts and limitations, it is important to shed light into the different implementations of RPL and tools that can be employed to develop and evaluate new contributions. Hence, several vendor-specific and open-source RPL tools and implementations exist in the literature as follows:

3.6.1 OPEN-SOURCE TOOLS

3.6.1.1 ContikiOS

Contiki [126][127] is a lightweight, open-source operating system designed specifically for the low-power resource-constrained IoT devices [128]. Contiki features a highly optimized networking stack including several IoT standards such as 6LoWPAN and IPv6. It also features an implementation for the RPL standard fundamental mechanisms within a library called ContikiRPL. Both the OF0 and the MRHOF are implemented within the library with OF0 using the hop-count as its routing metric and the MRHOF using the ETX. In addition, the latest version of ContikiRPL includes both the storing and the non-storing modes of RPL.

In 2017, the authors of Contiki started a new fork of the operating system named Contiki-NG [129] which features two different implementations of RPL: RPL-classic, and RPL-light. RPL-classic has a code size of 227 KB whereas RPL-light has a relatively smaller code footprint of 204 KB. The main difference between the two implementations is that RPL-light does not implement some features that seem unnecessary such as the storing mode and the

multiple instances (e.g., only one instance has been supported that uses the MRHOF and ETX metric). However, all Contiki-based implementations of RPL do not include any of its security features.

3.6.1.2 TinyOS

TinyOS [130] is another open source, component-based operating system designed for low-power wireless motes. TinyOS has its own implementation of the RPL standard named TinyRPL, which is designed to be used with BLIP (the Berkeley Low-power IPv6 Stack). The last implementation of TinyRPL supports both the storing and non-storing modes of RPL with the default upward routes. It also supports the two standardized OFs (i.e., OF0 and MRHOF). However, TinyRPL only supports a single instance with multiple DODAGs whereas it lacks any support for RPL security features. The code size of Tiny RPL is smaller than that of ContikiRPL with only 113 KB.

3.6.1.3 RIOT-RPL

RIOT [131] is an operating system for memory-constrained low-power wireless Internet of Things (IoT) devices that has also its own implementation of the RPL standard named RIOTRPL [132]. RIOTRPL supports the two downward RPL's modes of operations; however, it only implements the OF0 with hop-count routing metric. It has a code size of more than 105 KB, and it does not provide any support for the security modes of RPL.

3.6.1.4 Unstrung

Unstrung is a user-space Linux-based implementation of the RPL protocol intended for wired/Ethernet backhaul networks and gateway systems [133][134]. It can run on laptops, multipurpose IoT nodes, access points and diagnostic devices [134]. The implementation is mostly written in C++ with a code size of 1 MB. While Unstrung supports the storing mode of RPL, it does not provide support for the non-storing mode.

3.6.1.5 SimpleRPL

SimpleRPL is another user-space implementation of RPL for Linux-based systems. It is written in Python and has a code size of 228 KB. Pertaining to downward routing, SimpleRPL supports only the storing mode without multicast [135]. In addition, only the OF0 with hop-count metric is supported by SimpleRPL with the capability to form only one DODAG. Like other implementations, SimpleRPL does not provide any support for the security features of RPL as it is expected to be run on a secure environment [135][41].

3.6.2 RPL VENDOR IMPLEMENTATIONS

According to [41], several vendors have implemented their own versions of RPL including Samsung, Huawei, and Cisco. However, the available information about these implementations is very scarce as they are confidential. Only Cisco has revealed some of the implemented features in a form of configuration guide available online in [136]. Several features have not been implemented by Cisco including the secure mode of RPL and the non-storing mode. In order to cover a wide spectrum of uses in smart cities, Cisco implementation of RPL includes support for three OFs, namely, OF0, OF1 (latency) and OF1 (ETX) [136].

3.7 SUMMARY

In this chapter, we have outlined the main concepts related to RPL operations (i.e., RPL's topology routes construction, RPL's objective functions and RPL's routing maintenance mechanism). In addition, we elaborated on the key limitations of the protocol presenting how the research community has responded to such limitations and highlighting the major pitfalls of RPL's extensions proposed to overcome its limitations. Hence, the chapter concludes that RPL suffers from three major gaps and novel solutions need to be developed in order to address such gaps.

The first major identified gap in this context is the lack of the standard for an efficient routing maintenance primitive that opt for a rapid convergence while maintaining very low

overhead and power consumption profiles. In response, a Novel Adaptive and Efficient Routing Update Scheme for LLNs has been introduced as demonstrated in **Chapter 4.**

The second major gap being identified concerns the lack of the standard for an efficient load-balancing objective function that ensures a fair distribution of data traffic among respective nodes while minimizing overhead and maintaining network stability. In response, a New Load-Balancing Aware Objective Function has been designed as presented in Chapter 5.

The third major identified gap concerns the lack of the standard for an efficient routing primitive that addresses the memory limitations in IoT's networks. In response, a Leaf-Based Downward Routing Mechanism for RPL Protocol in Internet of Things has been proposed as illustrated in **Chapter 6**.

CHAPTER 4: DRIZZLE: FAIR ROUTE MAINTENANCE ALGORITHM FOR LLNS

In this chapter, the Drizzle, a new algorithm for maintaining routing information in the Low-power and Lossy Networks (LLNs) is introduced. The aim is to address the limitations of the currently standardized routing maintenance (i.e., the Trickle algorithm) in such networks. The chapter starts with revisiting the issues related to adopting Trickle for routing maintenance in RPL's networks that motivated us to develop the Drizzle algorithm. Next, an overview of Drizzle highlighting its main features and characteristics is presented. The chapter, then, ends with an analysis and evaluation of Drizzle performance in LLNs compared to Trickle and its extensions in the literature.

4.1 BACKGROUNAD AND PROBLEM STATEMENT

As mentioned previously, RPL has adopted the Trickle algorithm [31][32] for exchanging routing information and maintaining the topology with the aim of minimizing the control overhead and energy consumption while persevering network reliability. Indeed, Trickle relies on two primary mechanisms to disseminate efficiently the routing information. The first mechanism is to change adaptively the signaling rate according to the conditions currently present in the network. The second is the suppression mechanism in which a node blocks the transmission of its control packet if it detects that it is redundant (i.e., enough number of its neighbors has transmitted the same piece of information). Hence, the adaptive signaling rate in addition to suppressing redundant information enables the network to use its available resources efficiently, and consequently save energy and bandwidth. However, several research studies have recently reported some issues that limit the efficiency of the Trickle algorithm in LLNs as follows.

4.1.1 INTRODUCING THE LISTEN-ONLY PERIOD

A noticeable issue in Trickle is the introduction of the listen-only period in the first half of each Trickle's interval as discussed in Chapter 3 (Section 3.3). In fact, the listen-only period was proposed to solve the so-called short-listen problem arising when running Trickle in asynchronous networks [31]. In an asynchronous network with no listen-only period, a node may start sending its current DIO very soon after starting a new interval, a behavior that may result in turning down the suppression mechanism in the current and subsequent intervals, leading to significant redundant transmissions and limiting the algorithm scalability [31]. However, introducing the listen-only period has its own shortcomings. Firstly, this period will impose a delay of at least half the interval before trying to propagate an update. In an *m-hop* network, the inherited delay will be progressively accumulated at each hop resulting in an overall delay proportional to the number of hops [48][49]. Secondly, the listen-only period may result in an uneven load distribution among network nodes with some nodes transmitting less than others do during the operational time [48]. In the worst-case scenario, the transmitting period of a node may substantially overlap with the listen-only period of a neighboring node, preventing the former from sending for a long time. A key issue here is that the blocked node may be the one whose transmission is vital for resolving network inconsistences, consequently, negatively affecting the network performance [49]. In addition, the absence of a load balancing scheme may render some routes undiscoverable even though they might be more efficient than the active paths which may affect the network [48].

4.1.2 SUPRESSION MECHANISM INEFFICIENCY

In order to minimize the overhead in the network, Trickle employs a counter-based suppression approach which suppresses the transmissions of control packets that seem to be redundant. It does so by counting the number of consistent messages that are received within a specific window and, then, when such a number surpasses a pre-configured redundancy

constant *(k)*, it suppresses any further propagation of such received messages. However, studies have reported that the optimal setting of the redundancy constant is not a trivial task and relies greatly on the application scenario hence, if configured incorrectly, it may give rise to some issues such as the creation of sub-optimal paths [48][53]. Indeed, the work in [48] highlighted such a difficulty associated with configuring the redundancy parameter *k* in RPL-based networks. It reported, as an example, that the Trickle RFC restricts the typical values of *k* to be between 1 and 5, while RPL RFC [27] has set 10 as the default for *k*. However, the adequate value for the redundancy constant is claimed to be between 3 and 5 in the last IETF draft titled "Recommendations for Efficient Implementation of RPL" [54]. Finally, it is recommended in MPL RFC to set the default value of *k* to one. The risk of harming the network performance due to the inefficient suppression mechanism of Trickle was reported in [48]. In particular, it was shown that the inefficient suppression of DIOs by means of Trickle might result in sub-optimal routes, especially in heterogeneous topologies with regions of different densities. This is attributed to the fact that Trickle was originally designed to disseminate code updates which are quite similar in the context of reprograming protocols. However, this is not the case in the context of routing as two routing update messages originated from different sources may carry different routing information and thus *"suppressing one transmission or another is not always equivalent"* [48]. Hence, depriving a node from sending its DIO for long period may be problematic as it might be located on the most effect path to the root.

4.2 THE PROPOSED SOLUTION (DRIZZLE ALGORITHM)

To address the above-mentioned issues of the Trickle algorithm, a new routing maintenance primitive for RPL-based networks is proposed, named the Drizzle algorithm. Compared to Trickle, Drizzle has two key distinguishing features and different policies that endorse its superiority as a promising solution for routing maintenance in LLNs. The first distinguishing feature is that Drizzle eliminates the listen-only period presented in Trickle

intervals so that each node can schedule its transmission at any point throughout the interval rather than the second half only. This will enable the nodes to contend in a wider window and rapidly disseminate the routing updates leading to faster convergence time. In order to mitigate the negative side effect of the short-listen problem presented when removing the listen-only period, a new policy for setting the redundancy counter has been devised. The second distinguishing feature of Drizzle is the introduction of an adaptive suppression mechanism that different transmission probabilities to the nodes based on their transmission history. This, on one hand, relieves the network administrator from the task of configuring the redundancy coefficient. On the other hand, it will ensure the fairness of the algorithm, as the nodes that have transmitted more in the previous intervals would have less probability to send in the current interval. The fairness of the algorithm has been further supported by assigning each node a transmission slot within each interval based also on their transmission history. In this regards, Drizzle uses the same number of parameters used by Trickle and seven state-maintaining variables. **Table 4-I** outlines the seven variables used by Drizzle to maintain its current state.

Table 4-I. Drizzle's state-maintaining variables

Variable	Meaning
s	Number of sent messages until the algorithm reset its interval to the minimum interval
n	Number of intervals between two resets
rFlag	0 or 1 according to the case that produced inconsistency state
ck	Current value of the redundancy coefficient
I	Length of the current interval
t	A randomly chosen time within the current interval to transmit at
c	Message counter to keep a track of number of received consistent messages within the current interval

In what follows, the parameters used by Drizzle to configure its timeline are defined.

Definition 1: The minimum interval length (I_{min}): This is the fastest transmission rate in time units when a significant change in the network has been discovered (inconsistency).

Definition 2: The maximum interval length (I_{max}): This is the slowest transmission rate in time units of a node in the steady state.

Definition 3: The redundancy factor (k): represents the number for received consistent messages that a node should receive during a specific period before suppressing its own transmission.

The following steps illustrate in detail the operations of the Drizzle algorithm whereas the Drizzle pseudo-code is presented in **Algorithm 4-1.**

1. Drizzle starts its operation by setting its first interval to I_{min} and the redundancy value, *ck*, to the initial value of the redundancy coefficient, *k*. It also sets the broadcasted messages number, *s*, and the consistency counter, *c*, to zero. Finally, it sets the *rFlag* and the number of intervals, *n*, to one.
2. At the beginning of each interval, Drizzle assigns a randomly selected value in the interval to the variable, *t*, taken from Eq. 3. The rationale behind this is explained later.

$$[\, s * \frac{I}{n}, (s+1) * \frac{I}{n} \,] \qquad (3)$$

3. Upon receiving a consistent message, Drizzle increments its consistency counter by one.
4. When a node running Drizzle detects an inconsistent state, Drizzle resets its timer by setting I to $I_{\min}$, if it was not already set, and resets the interval counter, and the message counter to zero. It also resets the value of interval counter to one. Finally, it sets the value of the *rFlag* to either one or zero according to the case that produced the inconsistency. I limit the cases in which the *rFlag* is set to one to only three: (a) when the root establishes the construction of the DODAG, (b) when the root initiates a global repair, and (c) when a node firstly joins the DODAG.
5. At randomly selected time, if the consistency counter is less than the redundancy coefficient, Drizzle transmits its scheduled message; otherwise, the message is

suppressed. At this time, Drizzle also resets the consistency counter to zero.

6. If the scheduled message has been transmitted, Drizzle increases the broadcasted messages number by one. It also decrements the redundancy coefficient current value by one. If the value of the redundancy coefficient would be less than zero, Drizzle sets it to zero.
7. If the scheduled message has been suppressed, Drizzle increments the redundancy coefficient current value by one. If the new value of the redundancy coefficient *k* would exceed its initial value, Drizzle resets *k* to its initial value.
8. Once the interval *I* expires, Drizzle decreases its transmission rate through doubling the length of the interval providing that the *rFlag* value is one. If the value of the *rFlag* is zero, Drizzle decreases its transmission rate through entering directly the slowest transmission rate. In all cases, if the size of the new interval would exceed the I_{max}. Drizzle sets the interval size *I* to I_{max} and re-executes the steps from step 2. The interval counter, then, is increased by one.

In Eq. 3, the selection of random *t* is governed by the parameters *n* and *s* and the length of interval *I*. This will allow nodes to consider the history of their transmission when selecting their transmission slots randomly, thus a node that has transmitted less messages will opt to select an earlier *t* for the current interval so to have a better priority to transmit. In addition, increasing the value of redundancy coefficient when a node suppresses its DIO will further enhance its chance to transmit in the next interval and vice versa as in steps 6 and 7. This fulfils the first primary goal of Drizzle of being able to provide the network with a fair suppression mechanism, an issue that was not addressed by other Trickle variances. The reset of redundancy counter after the expiration of the timer *t* as in step 5, will allow a node to consider all DIOs received in taking the suppression decision. In other Trickle's extensions, the reset is only done at the start of each interval. Hence, DIOs received from the start of each interval until the

expiration of *t* timer will be ignored, thus taking the suppression decision will be based on an incomplete view of the state of network. Combining this procedure with the removing the listen-only period fulfils the second primary goal of Drizzle of being able to converge quickly while not experiencing the short-listen problem.

Algorithm 4-1 presents a description for the operation of Drizzle and its flowchart is depicted in Figure 4-1. The *Initialization* procedure will initialize the parameters of Drizzle to their default values. *The New-Interval* procedure will be executed every time the algorithm starts a new interval, increase the redundancy counter when receiving a consistent DIO , and select the value of the random timer *t*. On the other hand, the procedure will reset Drizzle parameters to their default values upon receiving an inconsistent DIO. The logic of *t* timer expiration is captured in the procedure *t_Timer Expired* in which the steps from 5-6 will be executed. Finally, the procedure Interval-*Expired* realizes the step 8 of Drizzle.

4.3 PERFORMANCE ANALYSIS AND EVALUATION

In this section, an analysis of the Drizzle algorithm highlighting its main advantages for LLNs is presented. The performance of the proposed algorithm is also validated through extensive simulation experiments under different scenarios and operation conditions.

4.3.1 PERFORMANCE ANALYSIS

4.3.1.1 Rapid Propagation

One of the observable issues presented in the standardized algorithm (i.e., Trickle) for route maintenance in LLNs is the introduction of the listen-only period in the first half of each interval with the goal to solve the so-called short-listen problem in asynchronous networks. The short-listen problem may turn down the suppression mechanism of Trickle resulting in significant redundant transmissions and, thus limiting the algorithm scalability.

This short-listen problem is illustrated in **Figure** 4-2 with three nodes (N1, N2, N3) running Trickle without the listen-only interval and *k=2*. One can observe that none of the three

Algorithm 4-1 : Drizzle Algorithm

```
1:  procedure Initialization
2:      I ← I_min , ck ← k
3:      s ← 0, c ← 0
4:      n ← 1, rFlag ← 1
5:  end procedure
6:  Procedure New Interval
7:      Start t_Timer as in (1)
8:      if ConsistentTransmissionReceived then
9:          c ← c + 1
10:     end if
11:     if InconsistencyDetected then
12:         I ← I_min , c ← 0
13:         n ← 1, s ← 0
14:         if InitDODAG , JoinDODAG , or GRepair then
15:             rFlag ← 1
16:         else
17:             rFlag ← 0
18:         end if
19:     end if
20: end procedure
21: Procedure t_Timer Expired
22:     if c < ck then
23:         Transmit Scheduled Message
24:         s ← s + 1
25:         ck ← ck – 1
26:         if ck < 0 then
27:             ck = 0
28:         end if
29:     else
30:         ck ← ck +1
31:         if ck > k then
32:             ck ← k
33:         end if
34:     end if
35:     c = 0
36: end procedure
37: procedure Interval_Expired
38:     if rFlag = 1 then
39:         I ← 2 * I
40:         if I > I_max then
41:             I ← I_max
42:         end if
43:     else
44:         I ← I_max
45:     end if
46:     n ← n +1
47: end procedure
```

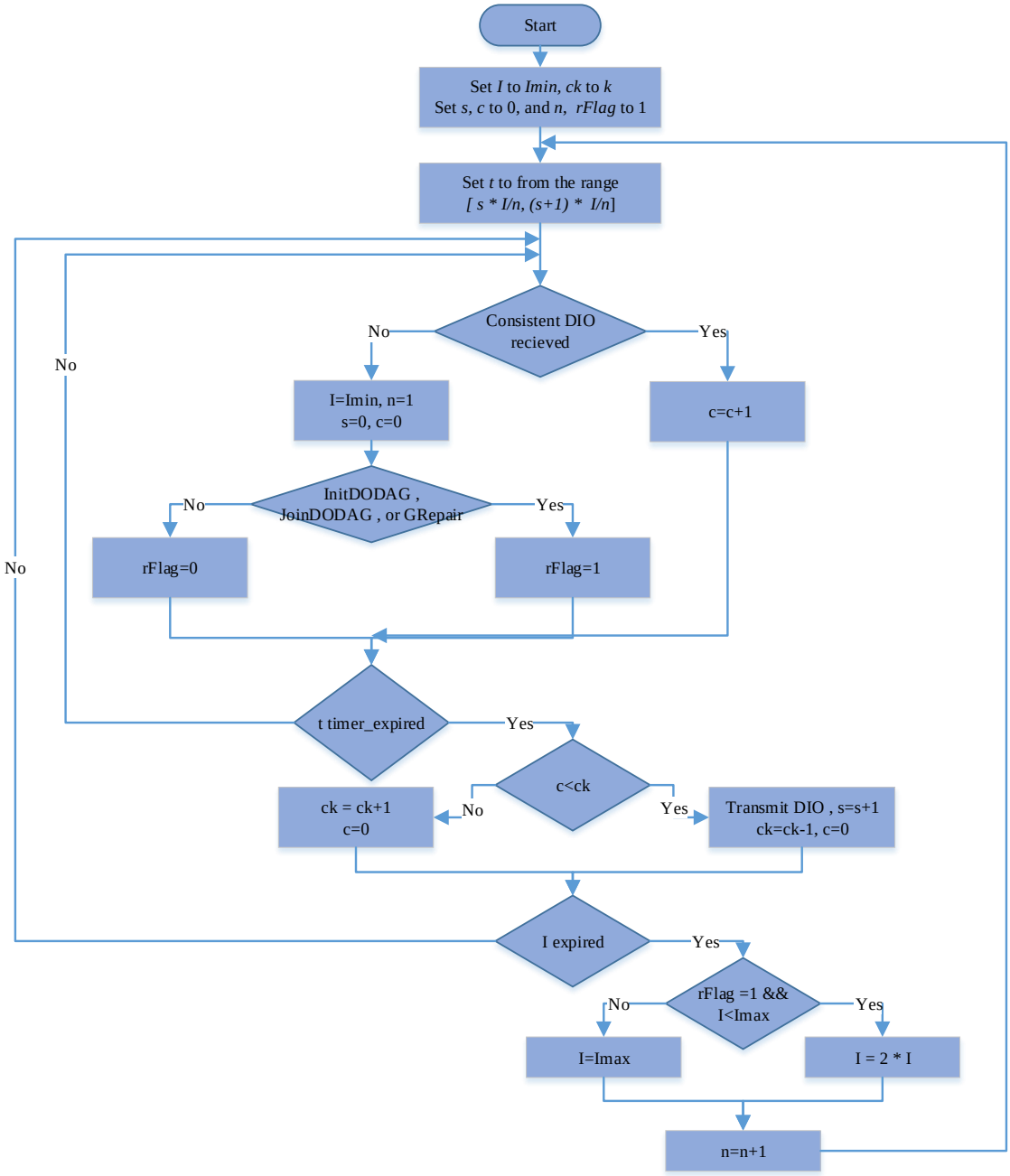


Figure 4-1. Drizzle algorithm flowchart

nodes have managed to suppress any DIO due to the short-listen problem as each node begins transmitting directly after starting its new interval and resetting its redundancy counter to zero. The figure shows that N1, N2 and N3 has started their first interval at different time setting their redundancy counter c to zero. N1 has randomly selected to transmit its DIO first; however, its transmission has been missed by N2 and N3 as they have not started yet their intervals at that point of time. N2 will start its interval next and will randomly select its slot early before

N3 starts its first interval. Hence, the transmitted DIO by N2 will be only received by N1 that will increase its redundancy counter to 1. Later, N3 will start its first interval and transmit its DIO at *t* that will be received by both N1 and N2 increasing their redundancy counters by 2 and 1 respectively. Note that no node managed to suppress its DIO due to the short-listen period. In a synchronized environment, one of the three nodes should be able to suppress its DIO under the same conditions. Note also that N1 will reset its counter *c* at the beginning of the second interval missing the history of past transmissions from N2 and N3, thus the chance to suppress its DIO in the second interval will be missed. This issue has been addressed in Drizzle by allowing the node to reset its counter *c* only after the expiration of timer *t* rather than the start of each interval.

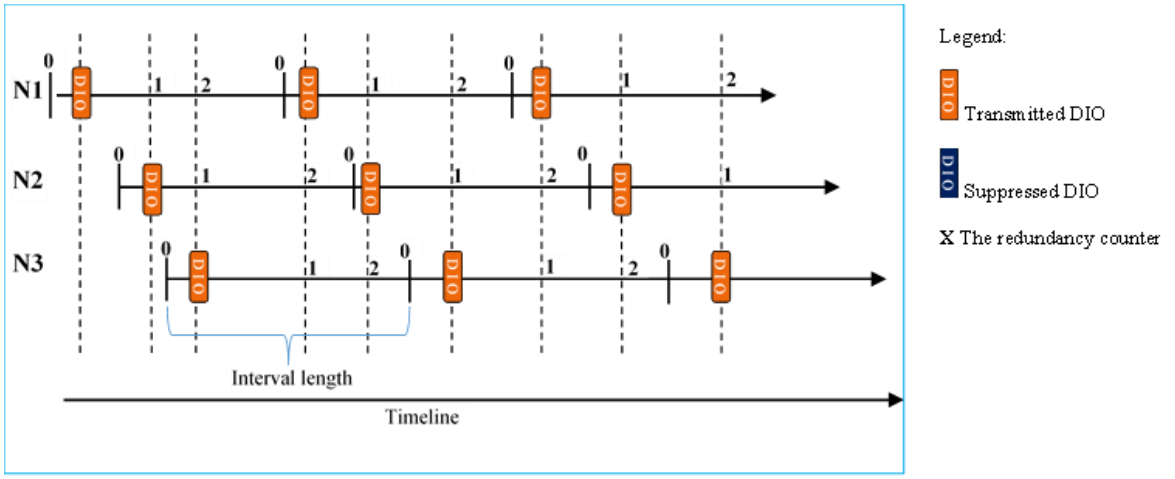


Figure 4-2. The short-listen problem in three asynchronous nodes; no suppressed transmissions at the absence of listen-only period

In Drizzle, the listen-only period is removed in order to facilitate faster propagation of the new information as each node would schedule its transmission from the range in Eq. 3 rather than [*I/2, I*]. In order to mitigate the effect of the short-listen problem, Drizzle keeps track of all messages received until the next scheduled time slot rather than the beginning of the next interval. Hence, instead of resetting the redundancy coefficient at the beginning of each interval, Drizzle resets this coefficient only at the beginning of the minimum interval and at the randomly selected time *t*. This behavior is illustrated in **Figure** 4-3 showing that the three nodes

have started their first interval at different times (i.e., they are not synchronized). Hence, all nodes initiated their redundancy factor to two and redundancy counter to zero at the beginning of the first interval. Then, the nodes randomly selected different transmission slots based on Eq. 3. Noticing the behavior of N1, this node has transmitted its scheduled DIO in the first interval as it has not received any DIOs from N1 and N3, hence, its redundancy counter was still zero which is less than the redundancy factor at the time of transmission. As N1 has transmitted its scheduled DIO in the first interval, it will decrease the value of its redundancy factor by one to increase the probability of suppressing DIOs in its second interval. Entering the second interval, N1 would have received two DIOs from N1 and N3, hence, its redundancy counter will be set to two. At time *t* in the second interval, N1 will suppress its scheduled transmission as its redundancy counter, which is two, is not less than its redundancy factor at this point of time. Also, at this time, N1 will reset its redundancy counter to zero performing the same logic in the subsequent intervals. Looking again at the figure and in a way similar to N1, N2 did not reset its redundancy counter *c* at the end of the first interval, rather, N2 waited until after its scheduled transmission slot to reset that counter. Hence, N2 has suppressed its transmission in the second interval, as the value of the redundancy counter is still greater than the redundancy factor *k* at the time of taking the transmission decision. This is not possible with Trickle as at the time of taking the transmission decision, the redundancy counter would have been reset to zero. Thanks to these new policies, Drizzle is able to resolve inconsistencies and propagate the new information much faster than other algorithms without even suffering from the short listen problem, except during the first interval, endorsing its energy efficiency and scalability.

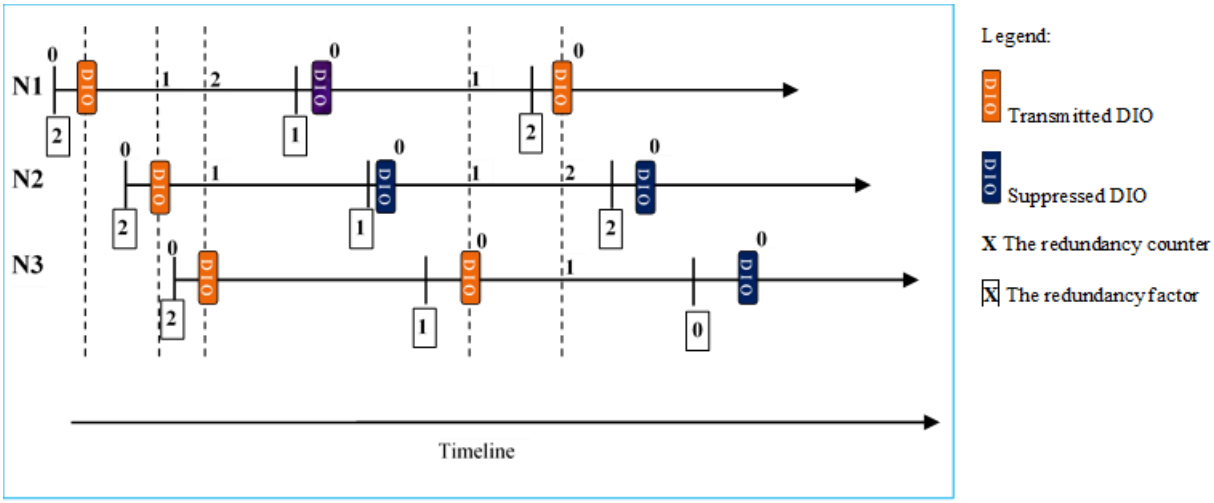


Figure 4-3. Drizzle operations in three asynchronous nodes; reinforcing suppression mechanism even at the absence of listen-only period

Note here that Trickle introduces the idea of the listen-only period in which a node must select *t* from the second half of the interval to avoid the short-listen problem. However, introducing the listen-only period comes with its own shortcomings. First, the listen-only period will impose a delay of at least *I/2* (i.e., half of the interval) before trying to propagate the new information. In a *m-hop* network, the inherited delay will be progressively accumulated at each hop resulting in an overall delay proportional to the number of hops. Indeed, I found that turning down of the suppression mechanism is not mainly caused by the absence of a listen-only period especially in the subsequent intervals. Instead, this problem mainly occurs due to the node ignoring all the received control messages from the randomly selected time in the previous interval to the end of that interval.

4.3.1.2 Load-Balancing

The distribution of the overhead evenly among nodes is one of the primary goals of any routing primitive primarily for the sake of avoiding disconnected regions in the network, which may lead to some kind of service disruption. In fact, the uneven-load distribution among nodes may lead to having some nodes drain their power faster than others and consequently shortening their lifetime. For instance, 100 messages evenly disseminated by 100 nodes, does

not incur a high cost. However, 100 messages disseminated only by one node does incur a high cost [31] and might lead to an earlier death of this over-burdened node. This may have a serious impact on the connectivity of the network as a whole especially if the nodes that drain their power faster are those representing the only-route to the base station (bottleneck nodes). The death of a bottleneck node means disconnecting that part of the network that forwards its data through that node which affects, in turn, the reliability of running applications and even denying some of the network services. In this regard, Drizzle introduces two mechanisms in order to guarantee efficient load distribution among network nodes. First, on the interval-level, a node is given a broadcast transmission probability according to how many transmissions it has sent. In other words, the higher the number of broadcasted transmissions, the lower the probability that a node would transmit in the current interval. This has been realized by introducing Eq. 3 in which the selection of random *t* is governed by the parameters *n* and *s* and the length of interval *I*. Hence, *s* is the number of sent messages until the algorithm reset its interval to the minimum interval whereas *n* represent the number of intervals. This equation will allow nodes to consider the history of their transmission when selecting their random slots, thus a node transmitted less messages will opt to select an earlier *t* for the current interval so to have more priority to transmit. For example, if the length of the current interval *I* is 100s, assuming that the current interval is the 4th interval, and that three nodes A, B, and C have 0, 1, and 2 transmissions respectively in the three previous intervals, (i.e., A has never transmitted any DIO during the three intervals, B has only transmitted once, and C has transmitted two DIOs). According to our algorithm, the three nodes should select their transmission slots according to Eq. 3 as follows:

$$A_t = [0 * 100/4, 1 * 100/4] = [0, 25].$$

$$B_t = [1 * 100/4, 2 * 100/4] = [25, 50].$$

$$C_t = [2 * 100/4, 3 * 100/4] = [50, 75].$$

You can observe from the above ranges that A will have a better chance to transmit in the current interval (i.e., 4th interval) by selecting *t* from the range [0, 25].

Second, Drizzle allows each node to have its own value for the Suppression Coefficient *k* referred to as *ck*. Each node changes the value of its initial *k* autonomously according to how many transmissions have been suppressed or sent during the previous intervals. This is different from that of the standard Trickle algorithm where a node is given the same broadcast probability every interval, even though it might never have had a chance to transmit. The unequal broadcast probability gives the opportunity for each node to broadcast its routing information as soon as possible enabling more efficient discovering of all possible paths and distributing load evenly among respective nodes.

4.3.2 SIMULATION EXPERIMENTS

In this subsection, I compare the proposed scheme with the standardized Trickle algorithm as well as three Trickle variances in the literature, namely, opt-Trickle [49], Trickle-F [48], and the adaptive-k (Trickle-Ad) [53] in terms of control-plane overhead (i.e., the number of the DIOs transmitted), convergence time, power consumption, and PDR. The compared algorithms have been implemented in Contiki [126], a lightweight and open-source operating system designed specifically for the low-power resource-constrained IoT device. Contiki features a highly optimized networking stack including several IoT standards such as CoAP, UDP, 6LoWPAN and IPv6. It also features implementations for the RPL standard fundamental mechanisms including the routing maintenance mechanism (Trickle) within a library called ContikiRPL [127] which is used as a ground for our implementation. Cooja [125], a Java-based, cross-level simulator for the Contiki operating system is used to carry out the simulation experiments. One advantage of using Cooja [125] with Contiki is that it allows us to emulate the exact binary code that runs on a real mote hardware. Cooja incorporates an internal hardware emulator called MSPsim [137] which is used in our simulations to emulate accurately

(i.e., impose hardware constraints) the Tmote Sky platform, an MSP430-based board with an ultra-low power IEEE 802.15.4 compliant CC2420 radio chip. The Unit Disk Graph Radio Medium (UDGM) with different loss rates was used in order to simulate the radio propagation in lossless and lossy networks. The CSMA/CA protocol is used as at the MAC layer, while the ContikiMac is used at the radio duty cycling (RDC) layer. The Minimum Rank with Hysteresis Objective Function (MRHOF) with ETX metric is selected for calculating the ranks of nodes and building the DODAG due to its efficiency in characterizing the quality of links. At the application layer, a periodic data collection application where each node sends to the sink one packet every 60 seconds (the time of packet transmission is randomly chosen within the 60 seconds period) is simulated. Both uniform and random topologies where nodes are spread in a square area of 200 x 200m dimensions are considered in the simulations experiments. The border router (sink) is placed in the middle of the network. For each scenario, ten simulation experiments with different seeds are run in order to get statistically solid results. The graphs show the average (mean) values of the results and the error bars at the 95% confidence interval of the mean. The simulation time is selected to be 20 virtual minutes for each experiment. For clarity, other simulation parameters are provided in **Table 4-II** which covers a wide range of scenarios that could be deployed in real home-automation environments.

Table 4-II. Simulation parameters

Parameter Name	**Values**
Number of nodes	100
Redundancy Factor (*k*)	1,3,5,7,10
I_{min} (ms) / I_{max} (ms)	$2^{10}/2^{20}$
Simulation time	20 minutes
Data Packet Rate	60 s
Mac/Adaptation Layer	ContikiMac/6LoWPAN
Radio Medium	Unit Disk Graph Medium (UDGM)
Loss model	Distance loss
Loss Ratio	0,10,30,50,70,90
Range	30 m
Interference Range	35 m

In the first set of experiments, the five algorithms are compared in lossy networks under the distance loss model varying the physical link loss rate between 0% and 50%. The 0% loss rate means that the network is lossless and as a result it does not experience any loss due to signal fading. However, the loss may still occur due to other factors such as hidden terminals and collisions. **Figures 4-4, 4-5, and 4-6** show the compared protocols performance in terms of control-plane overhead, which is defined here as the number of control messages, power consumption, and PDR, respectively. As it can be observed from **Figure** 4-4**,** the compared algorithms increase their control traffic overhead when the loss rate increases. However, Trickle's variances suffer heavily in terms of scalability in comparison with Drizzle, especially when the network is characterized by higher loss rates. In the worst-case scenario (50% loss rate), Drizzle registers an overhead rate of approximately five times less than that of Trickle while it registers also an overhead of approximately three times less than that of Trickle-adaptive. In fact, Trickle-adaptive uses a density-based mechanism to control the value of the redundancy factor. Although Trickle-adaptive has managed to reduce the control-plane overhead compared to other Trickle variances, it is not as efficient as Drizzle. Trickle-adaptive uses the number of DIO messages received by a specific node to estimate indirectly the network density at that node. Although this method might give approximately accurate estimation for the node degree when the network is characterized by synchronized intervals among its nodes, it may suffer from an inaccurate estimation in asynchronized networks. For instance, in an asynchronous network, the frequency of transmission may differ significantly from a node currently in its minimal interval to another node currently in its maximum interval. Hence, the node in its minimum interval would transmit more frequently giving the receiver node an impression that it has more neighbors than it actually does, hence affecting negatively the accuracy of the network density estimation at that node. On the other hand, the superiority of

Drizzle can be attributed to its adaptive suppression mechanism that allows a node to autonomously decrease its own transmission probability in the current interval according to how many control messages it has sent previously. In other words, the higher the control messages a node has sent, the lower its probability to transmit in the current interval and, therefore, bringing down the number of redundant control messages. Another reason behind the lower control-plane overhead of Drizzle is that it does not gradually double the current interval each time it receives an inconsistent control message. In several cases, according to the value of *rFlag*, Drizzle moves directly and not gradually to the lowest transmission rate, skipping the intermediate intervals and by that suppressing many redundant transmissions.

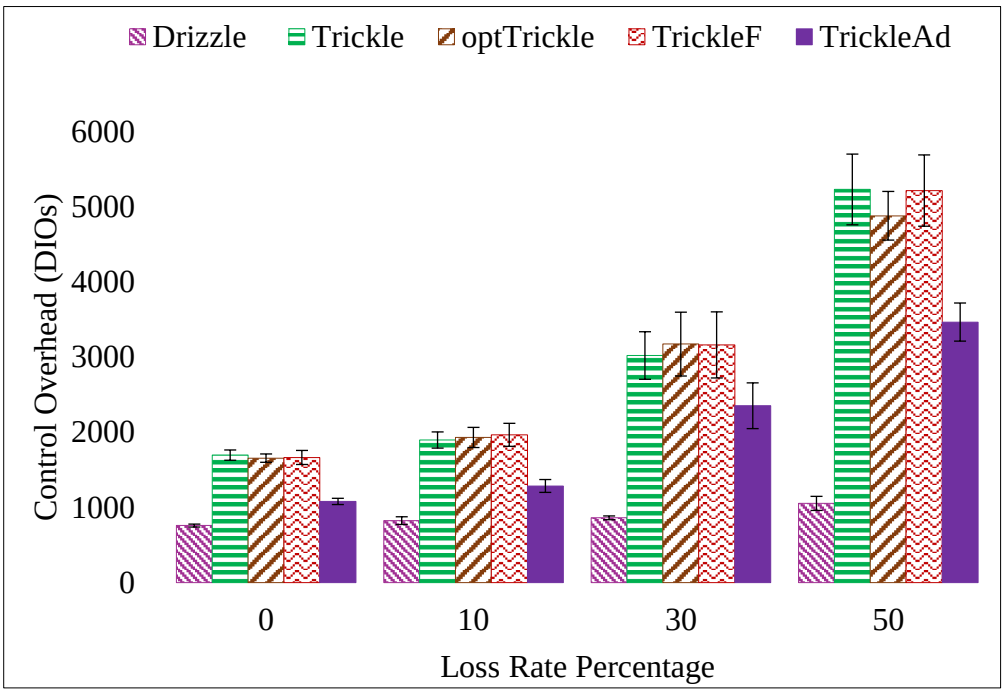


Figure 4-4. Control overhead under different loss rates (uniform)

It can be noticed also that the performance of Drizzle in terms of control overhead is almost independent of the loss rate which can be attributed in the first place to the capacity of Drizzle in minimizing the number of transmitted DIOs in general due to the mechanisms of adaptive *k* and history-based random-timer selection. Both the fair distribution and the minimized congestion will allow messages to reach their destinations under drizzle more often, thus the

suppression mechanism will work efficiently under drizzle compared to other algorithms. The decline in the number of transmitted control messages has resulted in lower power consumption for Drizzle when compared to the other algorithms except Trickle-Ad under loss rate of 0%, as depicted in **Figure** 4-5**.** However, it is not with the same rate of that of the control-plane overhead. This is because there are other factors contributing to energy consumption alongside with the number of control packets exchanged in the network such as its duty-cycle protocols.

With respect to the PDR, Drizzle performance here is comparable to Trickle variances as shown in **Figure** 4-6. However, it is very important to point that this PDR rate of Trickle's variances is obtained by generating more control packets compared to Drizzle and consuming more power. This indicates that Drizzle is able to discover comparable optimal paths to Trickle variances, however, with much less control overhead. In other words, the saving in control overhead and power consumption of Drizzle was not at the expense of the PDR.

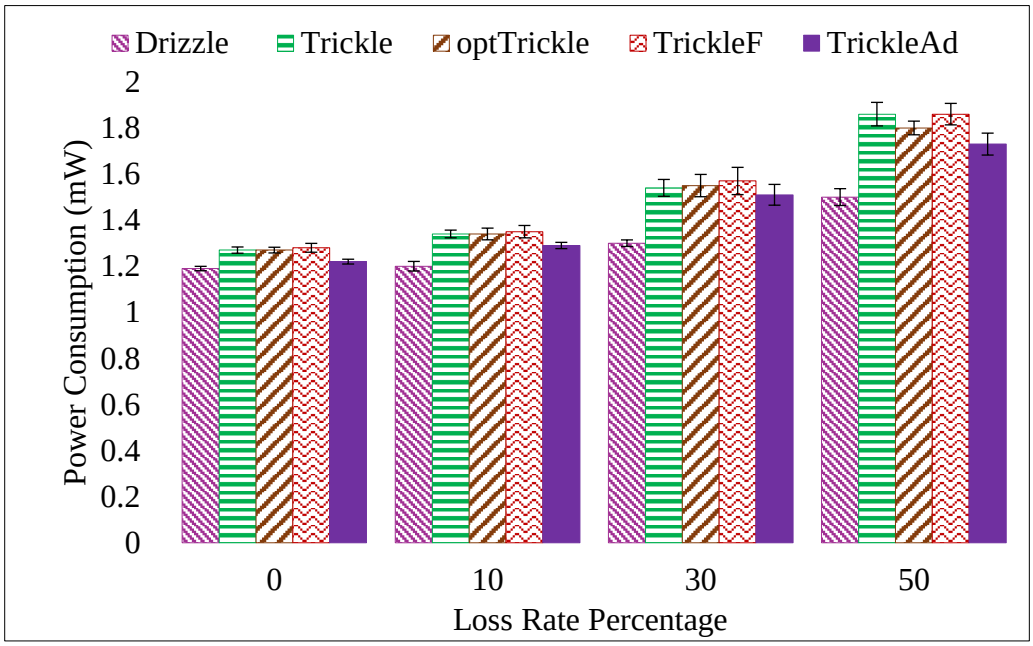


Figure 4-5. Average power consumption with various loss rates (uniform)

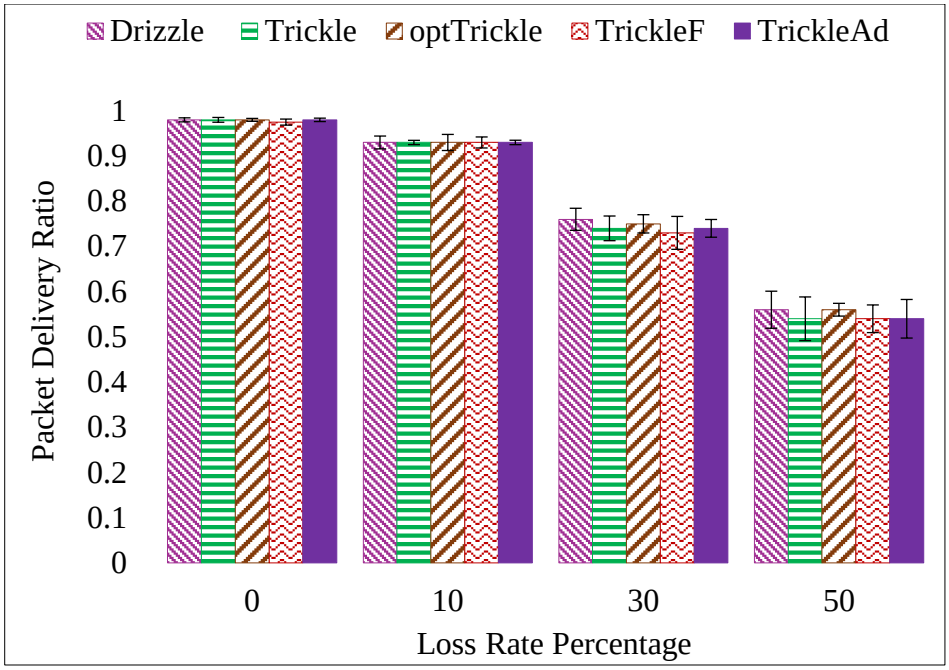


Figure 4-6. PDR under different loss rates (uniform)

Figure 4-7 compares the five algorithms in terms of average convergence time under various loss rate. The convergence time of a node in this study refers to the time at which the node has joined the network. Hence, the average convergence time is the convergence time of all nodes divided by the number of the nodes in the network.

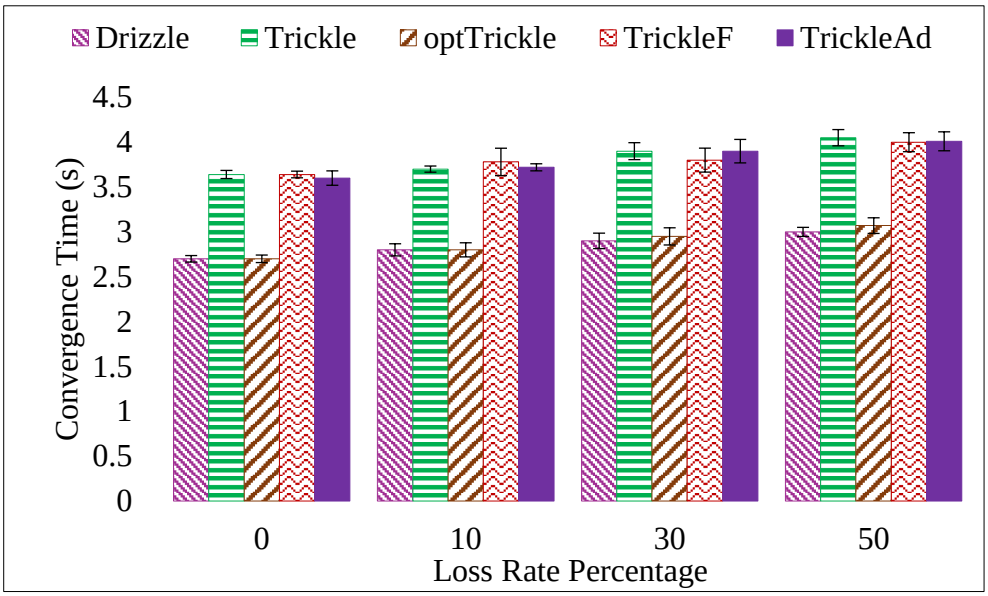


Figure 4-7. Average convergence time under various loss rates (uniform)

As can be observed from **Figure** 4-7**,** a network running Drizzle has the fastest convergence time compared to Trickle, Trickle-F and Trickle-adaptive even when the network suffers from higher loss rates. The case is somewhat different when considering opt-Trickle. Drizzle performance here is comparable to that of opt-Trickle in terms of convergence time. The superiority of Drizzle in terms of convergence time stems mainly from eliminating the listen-only period that allows the node to schedule its transmission as early as possible without even experiencing short-listen problem. Both Drizzle and opt-Trickle permit removing the listen-only from the first interval, hence, the comparable performance in terms of convergence time. The fact that Drizzle does not experience the short-listen problem can be confirmed by observing that Drizzle achieved faster convergence time, however, with generating much less control messages as illustrated in **Figure** 4-4. It can be also observed from the results that the higher the value of loss rate, the slower the convergence time in all algorithms. This can be explained by the fact that the higher the loss rate, the higher the probability that a control packet would be lost delaying the joining process until the next successfully received packet.

Figures 4-8, 4-9, 4-10, and 4-11 present comparisons among the five algorithms in a random topology with various loss rates in terms of control overhead, power consumption, PDR and convergence time, respectively. Similarly, the results illustrate that Drizzle generally performs better in terms of control overhead and power consumption than other algorithms except TrickleAd, especially under high rates of loss which can be again attributed to the adaptive suppression mechanism of Drizzle. Drizzle and TrickleAd seem to have a comparable performance in terms of power consumption and control overhead under lower loss rates whereas Drizzle has a better performance in this context under higher loss rates. This behavior can be attributed to the ability of TrickleAd to predict accurately the density of the network when it is characterized by lower loss rates and vice versa. The results also show that Drizzle has the fastest convergence time along with opt-Trickle while featuring a relatively comparable

PDR which also demonstrates that the optimal behavior of Drizzle in terms of control overhead and power consumption was not at the cost of the PDR.

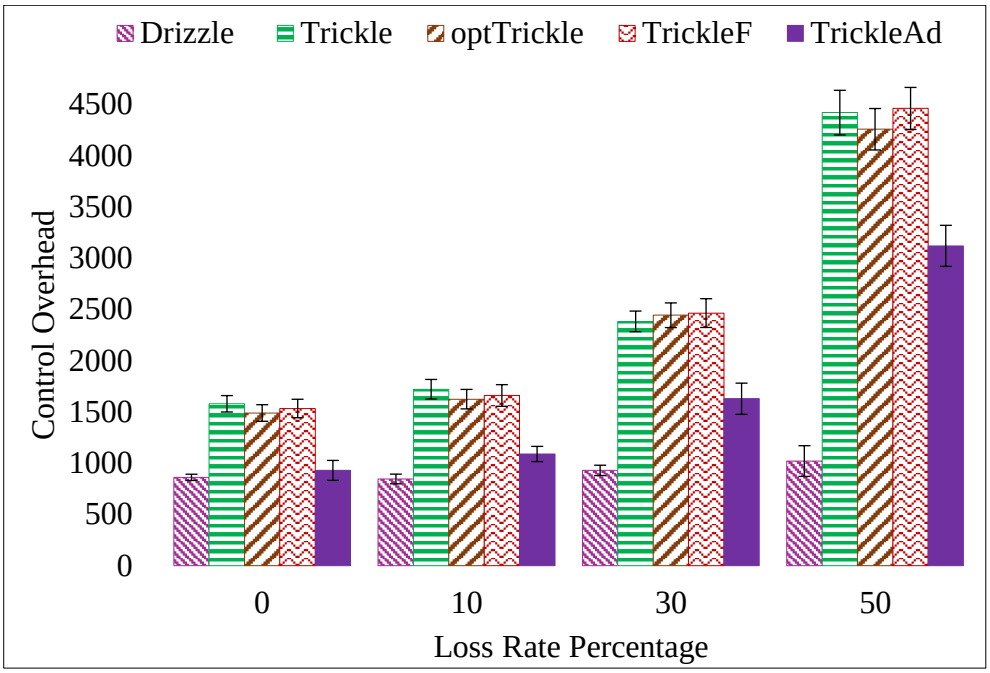


Figure 4-8. Control overhead under different loss rates (random)

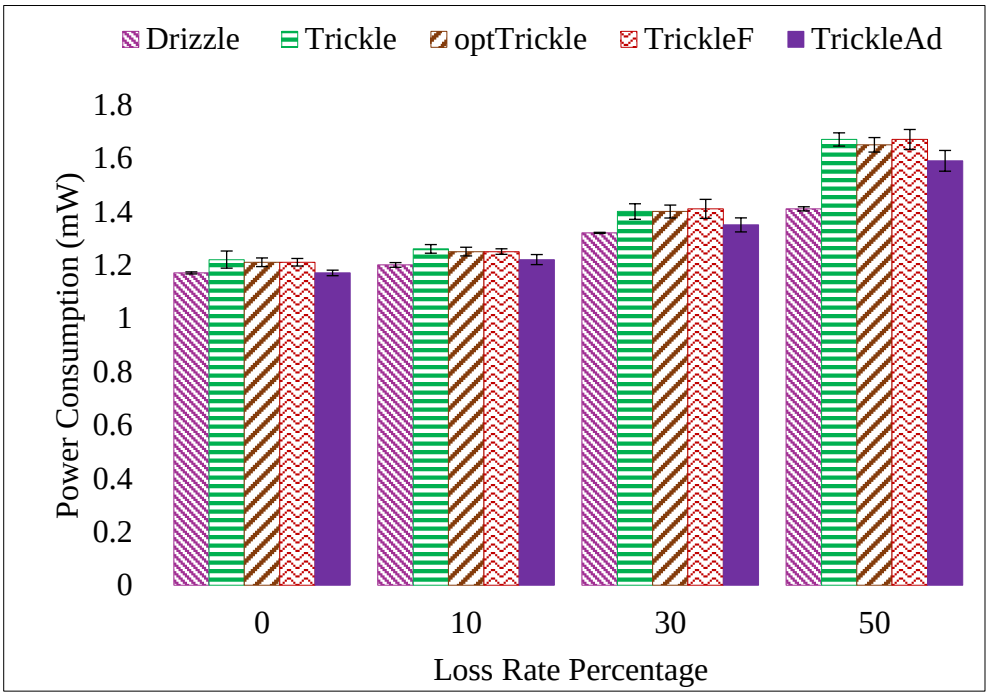


Figure 4-9. Average power consumption with various loss rates (random)

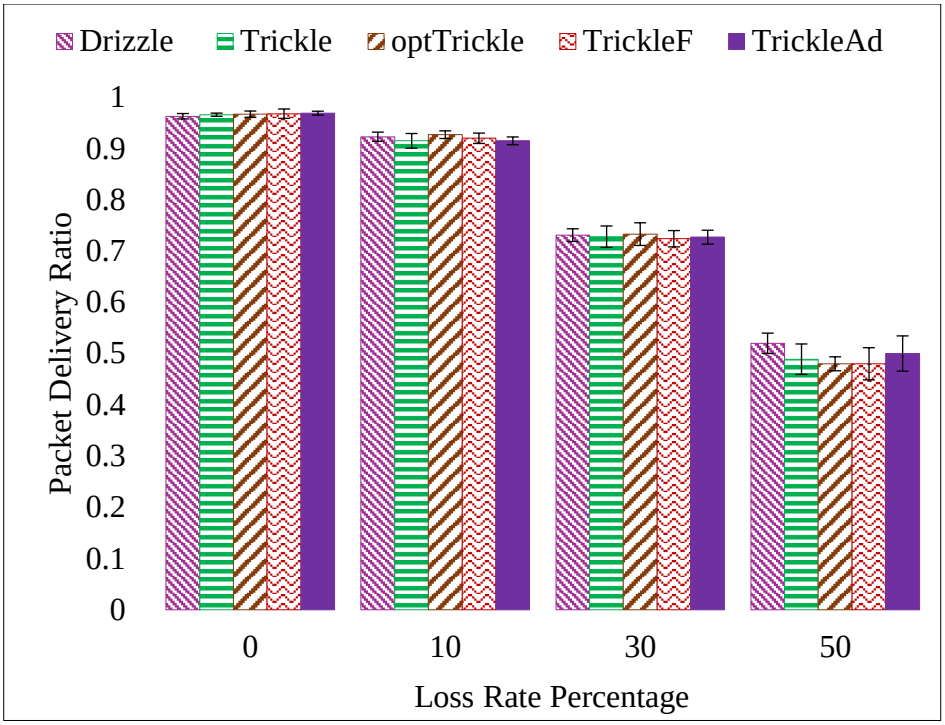


Figure 4-10. PDR under different loss rates (random)

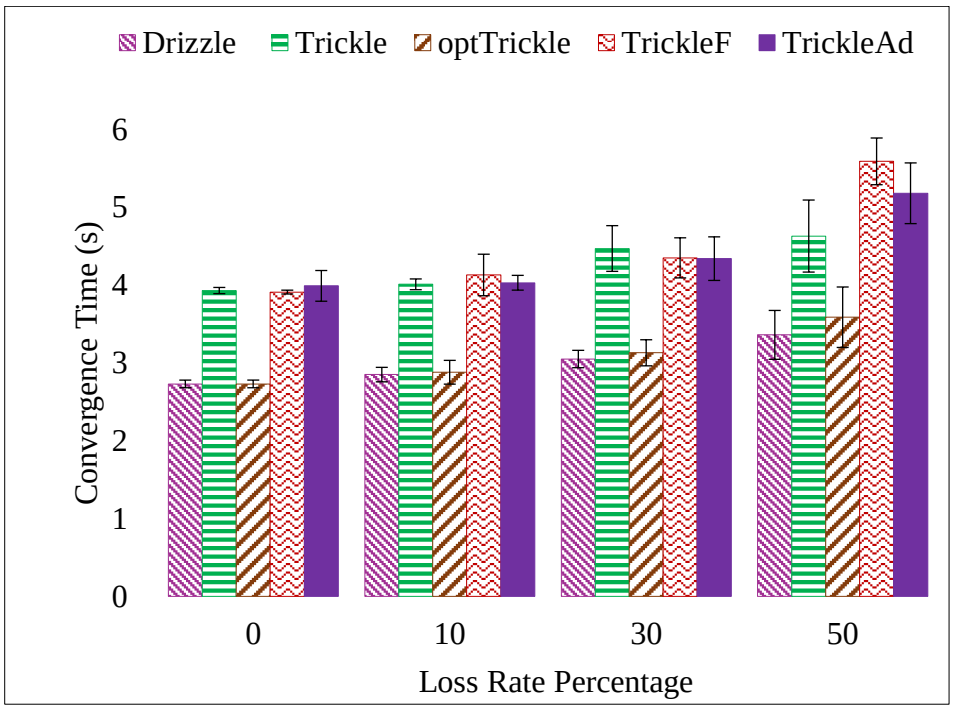


Figure 4-11. Average convergence time under various loss rates (random)

In the second set of experiments, the impact of the redundancy coefficient on network performance is evaluated in two variants of LLNs (lossless and lossy with 50% loss rate) under both random and uniform nodes distributions. As observed from **Figures 4-12 and 4-13,** it is clear that increasing the redundancy factor results in higher traffic overhead for Drizzle, Trickle, opt-Trickle and Trickle-F in both kinds of networks (lossy and lossless). A noticeable point here is the behaviour of Trickle-Ad under various redundancy values. It seems that there is no correlation between the initial value of the redundancy factor and control plane overhead. This is attributed to the fact that the value of *k* is dynamically changed based on the node degree so, whatever is the initial value; it will be decreased or increased to the extent that reflects the network density at that node. However, Drizzle still shows the best results in terms of traffic overhead in comparison to Trickle's variances including Trickle-Ad in all cases. The positive correlation between the value of *k* and traffic overhead in the compared algorithms (except Trickle-Ad) can be explained easily by the fact that the nodes tend to suppress less messages as *k* increases. On the other hand, the superiority of Drizzle in terms of traffic overhead again can be attributed to the adaptivity of Drizzle's suppression mechanism which allows the nodes to change dynamically their suppression coefficient according to their transmission history. Regardless of the initial value of the redundancy coefficient, a node running Drizzle is able to decrease its version of *k* each time it sends a message reducing its priority to transmit in the next interval, thus bringing down the number of unnecessary transmissions.

.

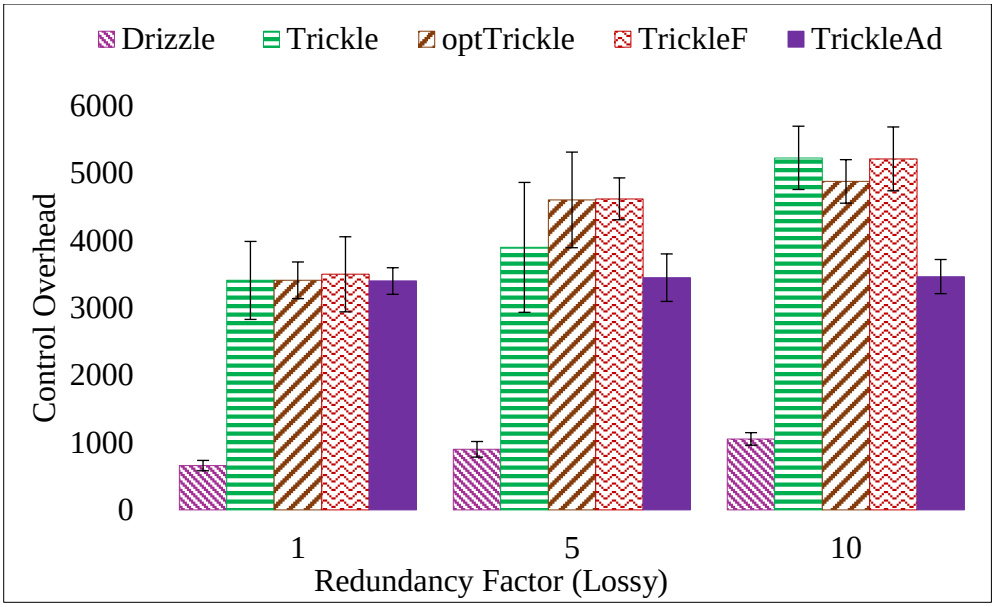


Figure 4-12. Control overhead under various values of *k* (lossy)

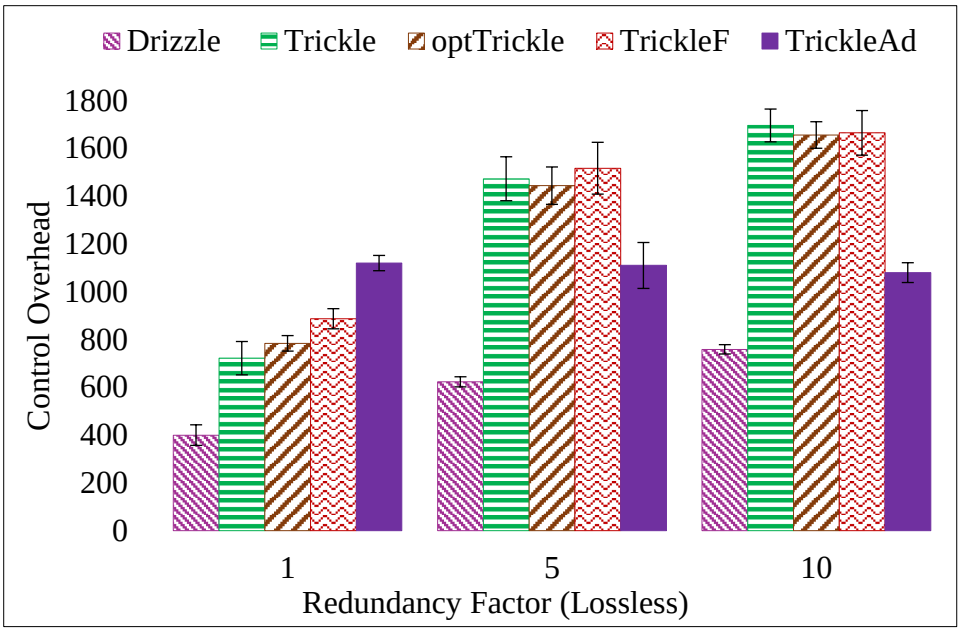


Figure 4-13. Control overhead under various values of *k* (lossless)

It is also clear from **Figures 4-14 and 4-15** that Drizzle outperforms Trickle's variances in terms power consumption rates in both networks types (i.e., lossy and lossless) regardless of the value of the redundancy factor. This can also be attributed to the capacity of Drizzle to minimize the overhead through its adaptive suppression mechanism which positively affect the power consumption. The power consumption of a specific node in Contiki is calculated by

tracking the fraction of time that a node remains in a particular power mode (i.e., Idle, listen, transmit and CPU) and then multiplying the time spent in each mode with its respective current consumption which is hardware-dependent. The total current of the four modes is then added up, multiplied by the voltage of the system and finally divided by the total running time to find the power consumption in mW.

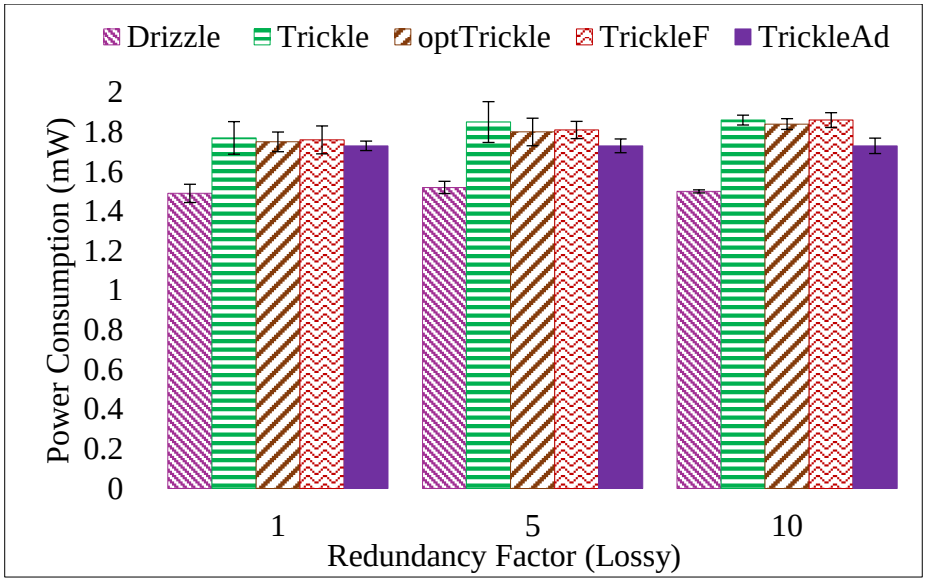


Figure 4-14. Average power consumption with various *k* (lossy)

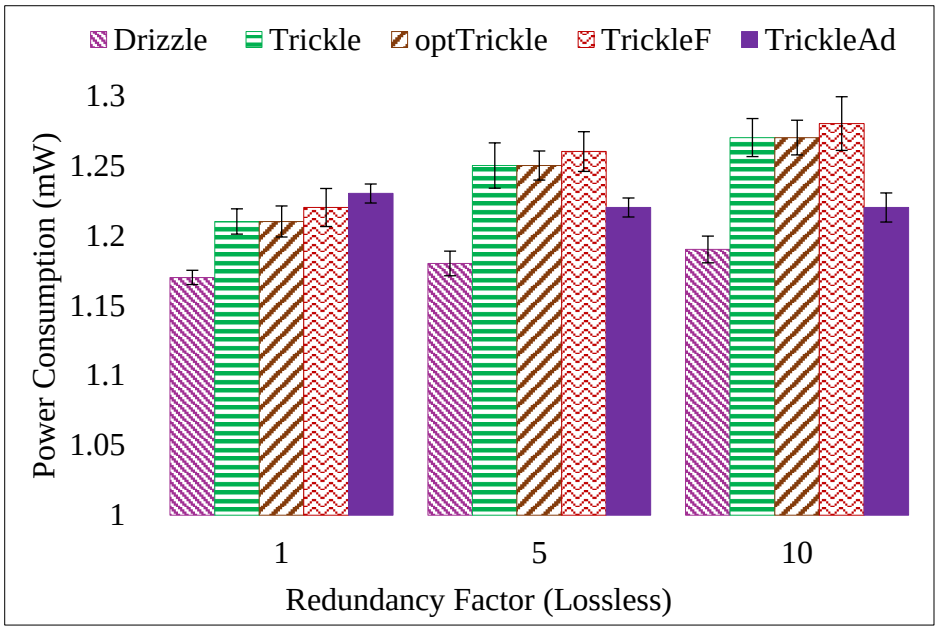


Figure 4-15. Average power consumption with various *k* (lossless)

Pertaining to convergence time, Drizzle also converges faster than Trickle's variances except optTrickle under different values of *k*, and whether the network is lossless or lossy as illustrated in **Figures 4-16 and 4-17,** respectively. This is also attributed to the facts explained previously regarding removing the listen-only period which contributes to enhancing convergence time.

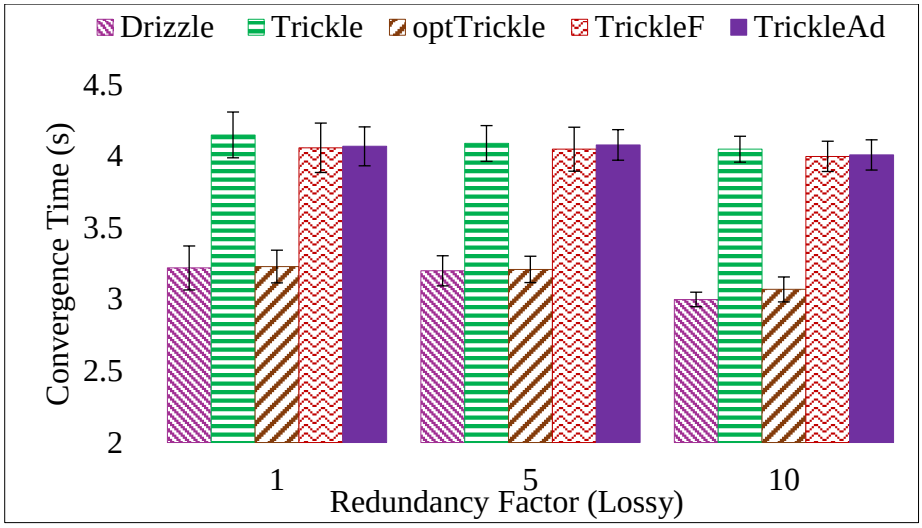


Figure 4-16. Convergence time under various values of *k* (lossy)

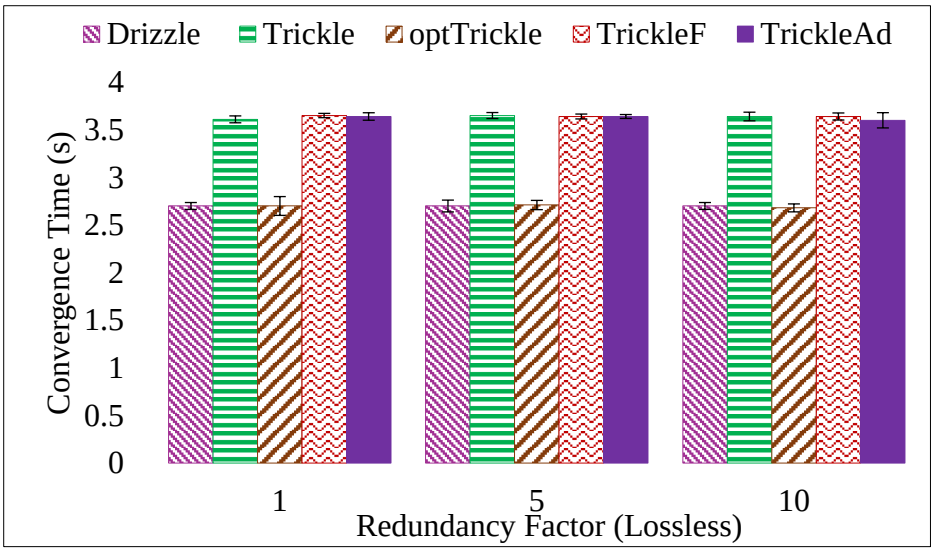


Figure 4-17. Convergence time under various values of *k* (lossless)

4.4 SUMMARY

In this chapter, a new routing primitive for route maintenance named Drizzle has been proposed for LLNs. Drizzle relies on the transmission history of nodes to configure their

suppression mechanism. In addition, Drizzle introduces a new policy for mitigating the negative effect of the so-called short-listen problem with the goal to limit transmission redundancy while providing faster convergence time. A performance evaluation of the proposed algorithm in comparison with the state-of-the-art routing maintenance algorithms including Trickle-F, opt-Trickle, Trickle-adaptive and Trickle itself has been conducted. Trickle-F strives to guarantee a fair multicast suppression among RPL nodes by giving each node a priority to send a scheduled DIO depending on how many DIOs it has suppressed recently. Opt-Trickle has the notion of equipping Trickle with a faster convergence time by eliminating the listen-only period from the first interval of Trickle whereas Trickle-adaptive aims to set the redundancy factor as a function of density of nodes seeking to ensure efficient suppression of DIO messages. The results highlighted the efficiency of Drizzle algorithm and improvements of up to 80%, 20% and 26% in terms of control overhead, power consumption and convergence time respectively have been achieved while maintaining comparable PDR rates. In addition, a demonstration of how Drizzle exhibits better load distribution and scalability in comparison with the standard IETF Trickle algorithm and its variances is given.

CHAPTER 5: A NEW LOAD-BALANCING AWARE OBJECTIVE FUNCTION FOR RPL'S IOT NETWORKS

5.1 BACKGROUND AND PROBLEM STATEMENT

As for load distribution, RPL lacks an efficient routing primitive that ensures a fair distribution of traffic among nodes while minimizing overhead. The absence of such mechanism may prevent the distribution of traffic among multiple nodes, potentially increasing data loss caused by the node packet buffer overflow or leading to a faster depletion of the energy of overloaded nodes which in turn may result in service disruption [56][57][59]. However, poorly implemented load balancing causes problems too. An example is the herding-effect [57], in which the network suffers topological instability caused by nodes repeatedly switching preferred parents in a futile attempt to achieve load balancing [57]. For instance**,** in **Figure** 5-1**a**, you can see that six nodes have selected the lightly loaded parent, *A*, upon receiving its DIO. However, in **Figure** 5-1**b**, all six nodes simultaneously changed their preferred parent to, *B*, in an attempt to load-balance the traffic upon receiving a new DIO from that node announcing a fewer number of children than node *A*. However, when receiving a new DIO from node A, the migration reverses, resulting in load oscillation with no balancing.

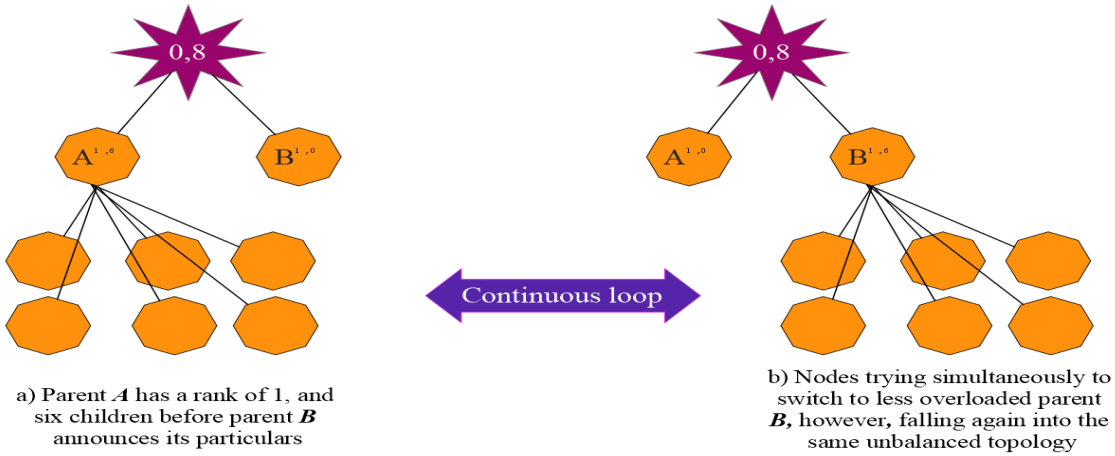


Figure 5-1. Herding-effect, the numbers besides name of nodes represent their ranks and children#

5.2 LOAD-BALANCING AND OBJECTIVE FUNCTIONS

The term load-balancing in the context of RPL's objective functions has been used by several studies [57][58] to refer to the ability of the objective function to devise a routing metric that will ensure the building of a balanced topology. Hence, no node will get overloaded in a way that will consume its energy, create congestion or overflow its buffers. For instance, the authors in [57] proposed a load balancing OF based on queue utilization for large scale industrial LLN named QU-RPL. In their proposal, the selection of parent is performed based on the parent's queue utilization and its hop distance to the LBR. Hence, the node does not perform load-balancing by distributing its traffic over equal-cost paths, it does so by changing its preferred parent to the less-overloaded parent, thus creating a balanced topology. The term load-balancing has been also used by several other studies to refer to the ability of the objective function to load balance the traffic over equal-cost paths [59]. For instance, the study in [59] has proposed an approach that selects multiple preferred parents relatively with equal costs as potential forwarders, and then forwards the data via one of them in a probabilistic manner based on their workload. In the thesis, I used the term load-balancing to refer to the ability of the objective function to build a balanced topology that ensures a fair distribution of traffic among RPL nodes of the same rank. However, our proposed load-balancing OF does not distribute the traffic of a specific node among its equal-rank paths in a probabilistic or deterministic manner, it simply ensures that a node will select the least-overloaded parent among its equal-rank parents as its preferred parent.

5.3 CRITICAL REVIEW OF RELATED WORK

The load-distribution problem of the RPL standard in LLNs has been the subject of several studies. For instance, in [56], authors propose a probability-based load-balancing solution for RPL referred to as LB-RPL. LB-RPL achieves load balancing by having each node distributing the traffic among its top *k* parents (in terms of Rank) based on their traffic load. A parent

experiencing heavy load may signal its status by delaying the broadcasting of its scheduled DIO message. This enables the child nodes to remove that parent from their top *k* and hence, exclude it from data forwarding. The implicit signaling through delayed DIO has no extra overhead, but a lost DIO might easily be misinterpreted as delayed, giving a false alarm of higher workload at some nodes [57].

The load-distribution in LLNs was also addressed in [92], which proposed a QoS-aware fuzzy logic OF referred to as *Fuzzy-Logic Objective Function* (OF-FL). OF-FL combines hop count, node energy, link quality and end-to-end delay in what was called *holistic* routing. The drawback of this OF is that several parameters must be transmitted to calculate the fuzzy values, thus requiring larger DIO messages, at increased risk of fragmentation and consequent additional overhead. In addition, fuzzy-based approaches are known to incur greater complexity, especially when multiple instances exist under the same RPL topology.

The authors in [108][109] propose a multi-path routing mechanism based on RPL allowing the protocol to forward traffic to multiple preferred parents. In this proposal, a new metric, the *Expected Lifetime metric* (ELT) is introduced, which aims to balance energy consumption among network nodes and maximize the lifetime of the bottleneck nodes. However, because several parameters must be exchanged (i.e., data rate, retransmission count, throughput, transmission power, and residual energy) to calculate the rank, this approach, like OF-FL, requires higher overhead and larger DIOs. Apart from these shortcomings, all the aforementioned mechanisms lack an efficient routing primitive that handles the "herding-effect" problem. They also either overreact to changes in load-balancing routing information or respond too slowly.

5.4 PROPOSED SOLUTION

In order to address the load-balancing problem of RPL, a new load balancing OF is proposed in this study and discussed in the following subsections. I emphasize here that the

main goal of the proposed solution is to introduce a load-balancing mechanism into RPL while maintaining stability with minimal overhead. In order to achieve this goal, I articulate that a stable and efficient load-balancing mechanism must go through several steps. The first step is to determine the most suitable metric for load-balancing and how to measure it. The second step is to ascertain how best to propagate load balancing information in a timely and efficient manner before it becomes obsolete. The third step how best to combine such a load-balancing metric with other metrics to preserve other performance aspects such as reliability. Finally, how the new proposed mechanism can mitigate the instability problem that may arise from introducing load-balancing into the network.

5.4.1 DECIDING ON THE BEST LOAD-BALANCING METRIC

Several routing metrics for achieving load balancing in RPL networks have been proposed in the literature including number of children, throughput, and queue utilization factor with each having its own shortcomings. In this study, I opt to use the number of children metric for the purpose of load-balancing for two primary reasons. First, it can be measured easily based on the data-plane traffic without incurring any extra overhead in the control-plane, especially in periodic applications as discussed next. Second, in the vast majority of applications, it reflects the actual network load.

5.4.1.1 Measuring the Number of Children Metric

In this study, I introduce for the first time the notion of calculating the number of children based on the data-plane messages rather than relying on RPL's DAO messages. In fact, number of children can be calculated based on RPL's DAO messages, however, as these messages are an optional feature of RPL, they may not available in all scenarios [57]. In addition, [57] points out that calculating the number of children based on DAOs may not reflect the actual load of the network promptly as it is only updated by the reception of DAO messages and timeouts of routing table entries (deletion).

To calculate the number of children based on data-plane traffic, I exploit the fact that IPv6 data packets carry the source address of the sender in the header of the packet, in addition to the RPL Hop-by-Hop (HBH) header option. Hence, when an IPv6 packet is received, the protocol first determines if it is a control or a data packet by inspecting if the RPL HBH Option is presented or not. The presence of such an option indicates that a data packet is being received at the network layer. If the received packet is data, it inspects its direction to decide if the sender is a child or not (direction is specified by the Down flag in the RPL HBH Option). If the packet is heading upward, the sender is a child so the protocol adds the sender IP Address (CH_{IP}) to the list of children (CH_{List}) of the receiver. If no data packets are received from an existing child during a pre-specified interval, it is removed. This interval is set proportional to the traffic rate. The pseudocode for this process is illustrated in Algorithm 5-1.

Another approach is to resort to the solution introduced in [124] by adding the parent address of each node to the DIO messages as an option, so as to enable the calculation of child numbers in the absence of DAO messages. However, an extra 16-byte option field is required to carry the parent address. In addition, due to the irregularity of DIOs, they may not reflect timely the number of children in the network (i.e., it is not guaranteed that routing information would be timely updated and there might be a problematic gap between the time the node has changed its preferred parent and the scheduled time for the DIO to be sent at).

Note that a combination of DIO-based calculation and data-plane based calculation of the number of children may be used in non-periodic applications.

Algorithm 5-1: Load-Balancing Objective Function

```
1:  procedure Initialization
2:      Set FastPropagation Timer
3:      Set Scheduling Timer
4:      Init CH_List // Children list
5:  end procedure

6::  procedure Data Packet Received
7:      if RPL HBH Option is set to 1 Then
8:          if CH_IP is not in CH_List Then
9:              add CH_IP to CH_List
10:             Set CH_IP _lifetimeTimer
11:         else
12:             Reset CH_IP _lifetimeTimer
13:         end if
14:     end if
15: end procedure

16: procedure CH_IP_lifetimeTimer Expired
17:     Remove CH_IP from CH_List
18: end procedure

19: procedure FastPropagation Timer Expired
20:     if CH_List has changed by a specific threshold Then
21:         Reset Trickle Timer
22:     end if
23: end procedure

24: procedure Scheduling Timer Expired
25:     Execute Parent Selection Algorithm
26: end procedure
```

5.4.2 ENSURING TIMELY PROPAGATION OF LOAD-BALANCING INFORMATION

The second step towards realizing an efficient load-balancing objective function is the timely propagation of the routing information, especially those related to the load-balancing metric (i.e., number of children). In fact, the propagation of routing information carried in RPL's DIO messages is regulated by means of Trickle in which the DIO transmission rate is increased upon detecting inconsistencies in the network. The current implementation of Trickle in Contiki OS has restricted the number of times the network is declared inconsistent

to a few cases to minimize the control-plane overhead. These cases include global repair, local repair, and parent change. Here, it is clear that a change in a node's rank does not trigger a resetting of the Trickle timer, so some routing decisions might be taken based on obsolete routing information due to the long DIO transmission period in the consistent state. Hence, a problematic gap may arise between the time the node has changed its preferred parent and the scheduled time for the DIO to be sent at.

To overcome this concern, I have opted to permit the node to declare an inconsistency upon detecting that its number of children has been increased or decreased by a pre-specified threshold. This will allow neighboring nodes to receive the load information in a timely manner, at the cost of some increased overhead. To reduce the overhead resulting from resetting the Trickle timer, the protocol does not do this every time the node's balancing information changes (in terms of number of children). Instead, I opt to use a *FastPropagation Timer (Reset Timer)*: when this expires a node checks whether it should reset its Trickle timer or not, as shown in **Algorithm 5-1**. The propagation of rank information is still governed by the Trickle algorithm itself.

5.4.3 Introducing the Composite Metric and Parent Selection

The third step towards realizing an efficient load-balancing objective function is answering the question of how to combine the load-balancing metric with another metric without negatively effecting other performance metrics, and how to select the preferred parent based on such a composite metric. Hence, I proposed that the new OF should lexically combine (the lexical composition is defined in Chapter 3) a primary metric (e.g., hop count) with the number of children load-balancing metric of a specific parent (NoCH) with the goal of building a balanced topology. To select the preferred parent based on such metric, the primary metric is used by a node to calculate its rank and select a set of candidate parents toward the DODAG root. Once the candidate parent set has been selected, the load-balancing metric is used to break

ties among selected parents if there is more than one in the parent set. The details of the parent selection process are illustrated in **Algorithm 5-2**.

Algorithm 5-2: Parent Selection Algorithm

```
1:  Function getPreferredParent(P1,P2)
    Input : P1 , P2 from the parent set
    Output : Preferred Parent (PP)
2:      if P1= PP or P2 = PP Then
3:          if P1. Rank = P2.Rank Then
4:              if P1. NoCH < P2. NoCH - α Then
5:                  return P1
6:              else if P2. NoCH < P1. NoCH - α Then
7:                  return P1
8:              else
9:                  return PP
10:             end if
11:         else if P1. Rank < P2.Rank – β Then
12:             return P1
13:         else if P2. Rank < P1.Rank – β Then

14:             return P2
15:         else
16:             return PP
17:         end if
18:     else
19:         if P1. Rank = P2.Rank Then
20:             if P1. NoCH < P2. NoCH Then
21:                 return P1
22:             else
23:                 return P2
24:             end if
25:         else if P1. Rank < P2.Rank Then
26:             return P1
27:         else
28:             return P2
29:         end if
30:     end if
31: end function
```

5.4.4 AVOIDING THE HERDING PROBLEM

One of the obstacles toward achieving load balancing in RPL is the way the protocol switches preferred parents. The common strategy adopted by RPL is to allow a specific node

to switch immediately to a better parent upon detecting such a parent by means of DIOs. This may have the advantage of timely switching; however, in the context of load balancing, changing preferred parent immediately on discovery may give rise to the herding effect explained earlier. To address this issue, I have introduced the idea of the *Balancing Timer (or Scheduling Timer).* This timer regulates the timing of parent selection process. Instead of having a node performing this process immediately upon receiving a new DIO, the parent selection in our proposed mechanism is performed regularly at pre-determined interval. However, I exclude the first received DIO from this policy to allow faster convergence time at the stage of DODAG construction. In other words, when a node receives a DIO for the first time, it will immediately choose the sender as its preferred parent. Then, the node must wait until the expiration of the *Balancing Timer* in order to check whether a better parent is available or not so it can change to a new preferred parent. The *Balancing Timer* is reset every time the node performs the parent selection process. The value of the *Balancing Timer* should be set in a way that allow for several DIOs to be received at the node so it can have several candidate parents which is better to be set empirically. The details of this process is also illustrated in **Algorithm 5-1.**

5.5 PERFORMANCE EVALUATION

In this subsection, the proposed scheme is compared to the RPL with OF0 in terms of power consumption and PDR. In particular, I have compared three versions of our proposed schemes with RPL. The compared versions are:

Load-Balancing Plain (LBPLAIN): In this version, I implement the load balancing part of our proposed scheme without employing *Scheduling* or *FastPropagation* timers. This is to show how introducing the load-balancing may affect performance.

Load-Balancing with Scheduling (LBS): In this version, I employ the *Scheduling Timer*, but not the *FastPropagation Timer*.

Load-Balancing with Scheduling and Resetting (LBSR): This is the complete version of our proposed load-balancing objective function which employs both *FastPropagation* (i.e., the Reset Timer) and *scheduling* timers.

The popular Cooja [125] simulator of the Contiki operating system [127] is used to carry out the simulation experiments. For each scenario, five simulation experiments with different seeds are run to get statistically valid results. The simulation time is 60 virtual minutes for each experiment.

In the first set of experiments, an evaluation of the performance of RPL, LBPLAIN, LBS and LBSR in a network of 50 nodes spread randomly over an area of 50 x 50 meters is conducted. **Figure** 5-2 shows the PDR under various traffic rates, specifically 30, 12 and 6 packets per minutes.

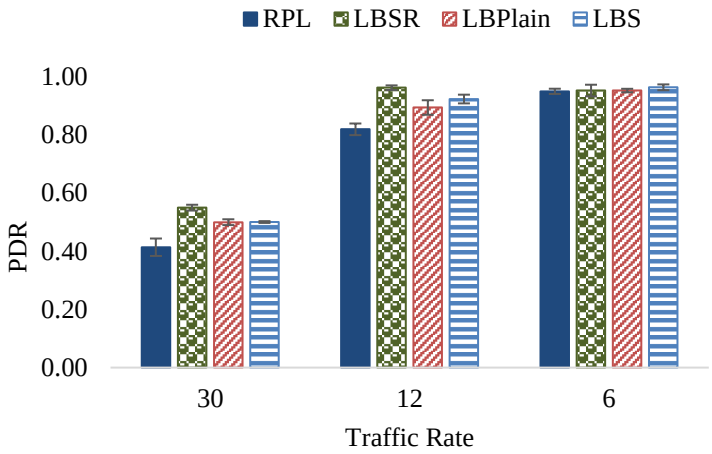


Figure 5-2. Packet delivery ratio under various traffic rates

As can be observed, the RPL protocol has the lowest PDR under traffic loads of 30 and 12 packet per minute, whereas the LBSR has the highest PDR (33% improvement over RPL and 10% over both LBS and LBPLAIN in the case of 30 packets per minute). The superiority of LBSR over RPL is attributed to the load-balancing primitive that strives to distribute load among respective parents by constructing a balanced tree in terms of number of children. The absence of load balancing in RPL leads to some nodes being highly overloaded, risking buffer

overflows. While both LBS and LBPLAIN also try to build a balanced topology, they fail to achieve the same delivery rates of LBSR. In the case of LBPLAIN, this can be attributed to the herding-effect, which prevents it from achieving a balanced topology giving rise to several overloaded nodes, though fewer than in RPL. LBS is somewhat different and the main reason for the low PDRs compared to LBSR is the inefficient propagation of the load-balancing routing information, with outdated information leaving the topology partially balanced. This is addressed in the LBSR protocol by resetting the Trickle Timer periodically upon detecting that the load-balancing information (number of children) has changed significantly.

The protocols were also evaluated in terms of average power consumption per packet as shown in **Figure** 5-3. It is also clear from the figure that the LBSR protocol is more efficient in terms of average power consumption per packet. In RPL, when a node becomes overloaded with children, there is a higher probability that packets and acknowledgments will be lost, leading to more retransmissions at the MAC layer and, hence, increased power consumption compared to LBSR. LBPLAIN registers the worst power consumption rates among the new OF versions. Apart from being incapable of fully balancing the topology, resulting in higher energy consumption rates than LBSR, it suffers from the herding problem causing churn in the network (i.e., frequent change of the preferred parent), as depicted in **Figure** 5-4, leading to higher energy consumption rates. This is addressed in LBS and LBSR by limiting the number of times a node is allowed to change preferred parent through introducing the notion of the *Scheduling* timer.

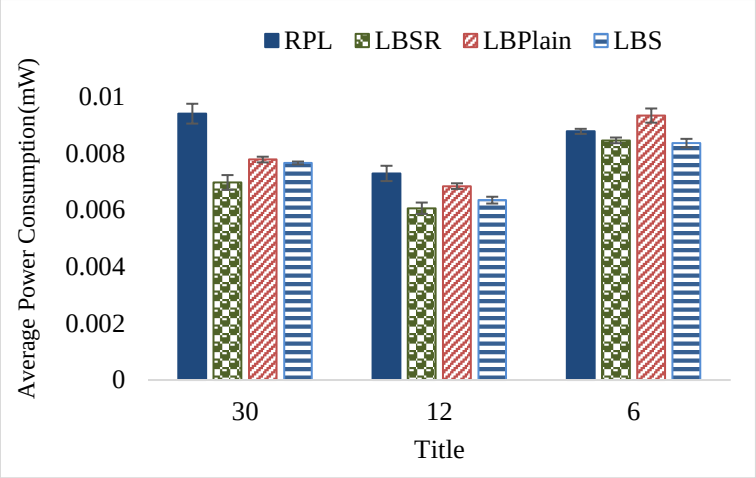


Figure 5-3. Average power consumption under various traffic loads

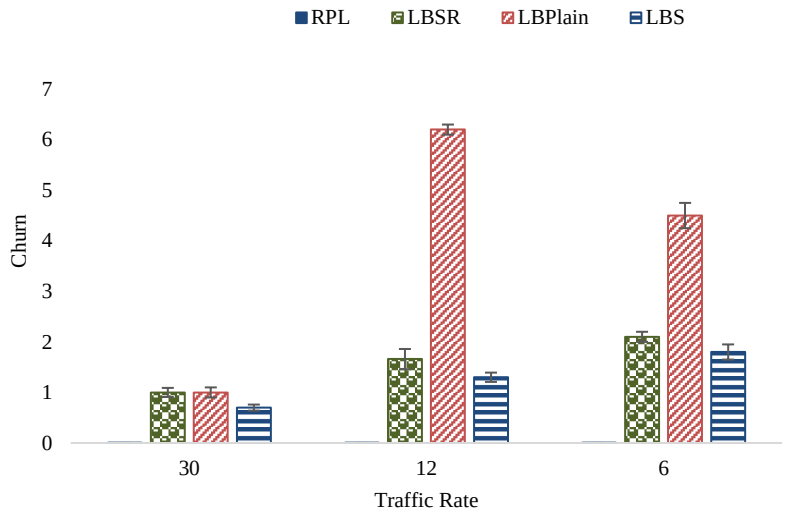


Figure 5-4. Churn under various traffic loads

I have also investigated how well the network is load-balanced in terms of energy expenditure under the different schemes. One way to assess this is via the Coefficient of Variance (CV) of power consumption metric, the ratio of the standard deviation to the mean. The lower the value of the CV, the better balanced the network is and vice versa. **Figure** 5-5 depicts the CV for varying traffic rates for the protocols being compared. It is clear from the figure that the RPL protocol has the highest CV values with approximately 90%, 40% and 27% under the different traffic rates while the LBSR has the lowest CV values (i.e., 10%, 15% and

18%), indicating a significant enhancement over RPL. This illustrates that the proposed LBSR protocol succeeds in building a balanced topology evident from the fair distribution of energy expenditure among RPL's nodes as the CV of power indicates.

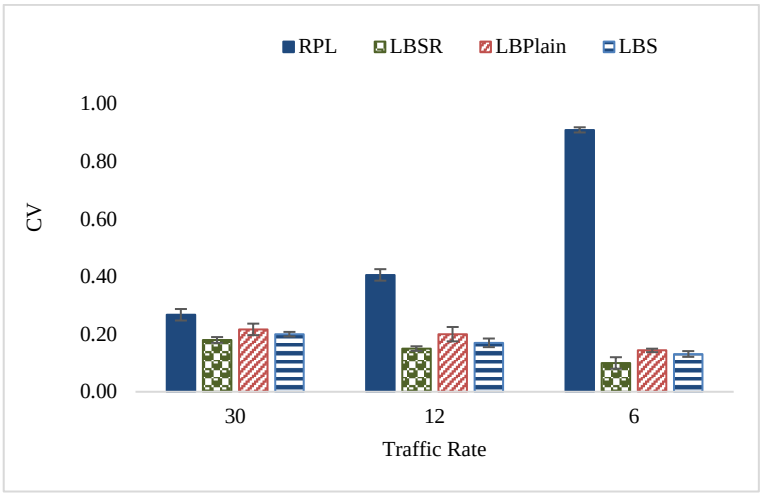


Figure 5-5. Coefficient of variance under different traffic loads

To get more insight into these facts, I have plotted a 3D mesh of power consumption for both RPL and LBSR in **Figure** 5-6**.** The power consumption of a specific node in Contiki is calculated by tracking the fraction of time that a node remains in a particular power mode (i.e., Idle, listen, transmit and CPU) and then multiplying the time spent in each mode with its respective current consumption which is hardware-dependent. The total current of the four modes is then added up, multiplied by the voltage of the system and finally divided by the total running time to find the power consumption in mW. The figures indicate that there are a few nodes that significantly consume power under standard RPL. It also illustrates how LBSR manages to load balance energy consumption with all nodes having power consumption rates between 4 and 8mW. In the cases of RPL, some nodes register average power consumption of 14mW, which is double that of the most overloaded node in LSBR.

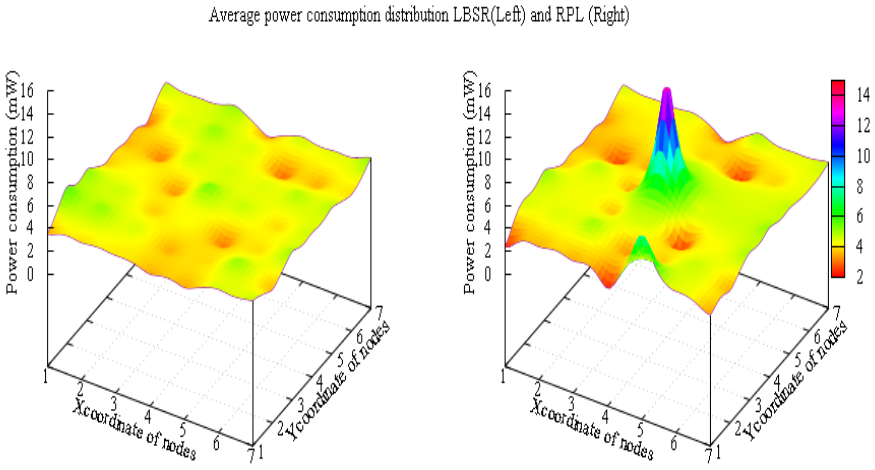


Figure 5-6. Power consumption distribution, LBSR (left), RPL (Right)

5.6 SUMMARY

In this chapter, a new load balancing Objective Function named the LBSR is proposed as an extension for the RPL protocol. In this objective function, a new routing primitive is devised to calculate the number of children based on the data-plane packets. In addition, a parent selection and optimization primitive is introduced based on the lexical combination of the number of children and another primary metric such as hop count. Furthermore, a new primitive for scheduling the parent selection process in order to mitigate the herding-effect problem is introduced. Finally, the proposed objective function is augmented with the notion of a *FastPropagation* timer that facilitates the timely and efficient propagation of routing information. A performance evaluation of the proposed protocol in comparison with RPL has been carried out demonstrating improvements of up to 33% and 88% in terms of PDR and load distribution, respectively, while maintaining comparable power consumption rates.

CHAPTER 6: LEAF-BASED DOWNWARD ROUTING MECHANISM FOR RPL PROTOCOL

6.1 BACKGROUND AND PROBLEM STATEMENT

As mentioned earlier, RPL has two modes of operations, namely, the storing and the non-storing modes. Pertaining to the storing mode of operation, each RPL router must maintain a routing entry for each destination in its own sub-DODAG in order to build the downward routes that carry the P2MP and P2P traffic. For instance, assuming that a DODAG topology has been constructed as depicted in **Figure** 6-1 and in order to enable communication in the downward direction from the LBR to other nodes in the network, each node should maintain a routing table entry as depicted in **Table 6-I**. Indeed, this poses a major challenge to memory-constrained LLNs as a router may run out of memory easily rendering it unable to accommodate new entries in its routing table. The incapacity of a router to add new routes will render several destinations in its sub-DODAG unreachable from the DODAG's root point of view. This in turn will affect negatively the application reliability as the root would drop all the packets destined to unreachable destinations [94][97]. Although many research studies have tried to mitigate this issue, they all have concentrated on combining both storing and non-storing modes of RPL or mixing the storing mode of RPL with multicast forwarding [95][96][97]. Hence, none of these studies has targeted reducing the number of entries in a node's routing table which is the major problem of RPL storing mode. In this chapter, a new routing primitive that aims at reducing the number of routing entries in the nodes routing tables is proposed.

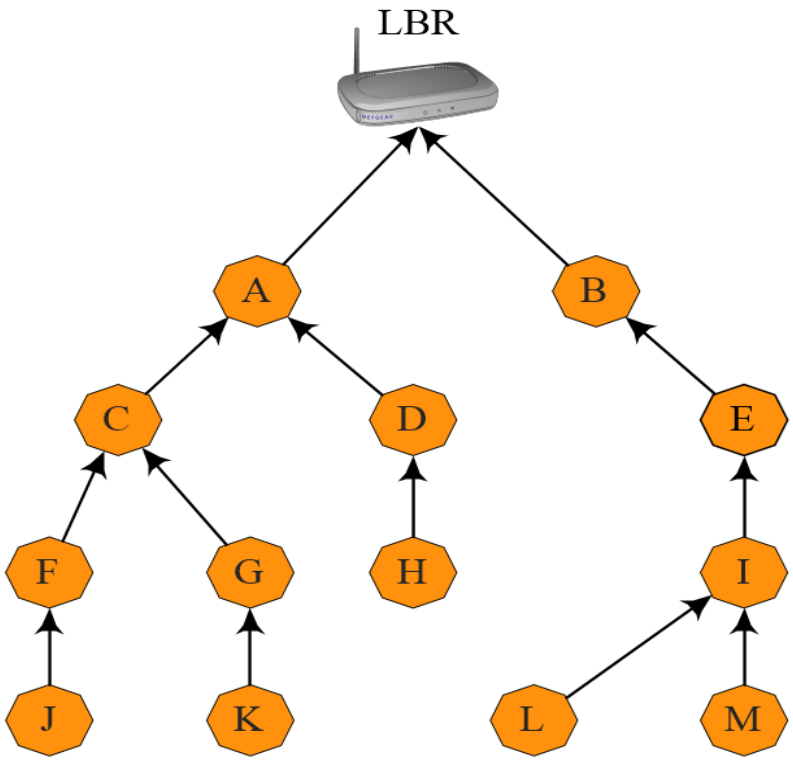


Figure 6-1. A DODAG topology, an arrow represents a relation between a child and a parent

Table 6-I. Nodes routing table's entries in RPL's storing mode

Nodes	Routing tables entries
LBR	(A,A),(B,B), (C,A), (D,A), (F,A), (J,A), (H,A), (G,A), (K,A), (E,B), (I,B), (L,B), (M,B)
A	(C,C), (D,D), (F,C), (J,C),(H,D), (G,C), (K,C)
B	(E,E), (I,E), (L,E), (M,E)
C	(F,F), (J,F), (G,G), (K,G)
D	(H,H)
E	(I,I), (L,I), (M,I)
F	(J,J)
G	(K,K)
I	(L,L), (M,M)

6.2 THE PROPOSED LEAF-BASED ROUTING

In this section, the novel proposed mode of operation for downward routing in RPL is presented highlighting the differences between this new mode and the storing mode of RPL. Thereafter, I would call this mode LBRPL.

6.2.1 THE MAIN IDEA

In the LBRPL, I argue that a router needs only to maintain the routing state of the leaf children in its sub-DODAG rather than the whole set of children so to enable the flow of data in the downward direction. This argument has been perceived based on the fact that a packet destined to a node must pass by its parent first. Thus, if a source node has a packet destined to a parent, the source would be able to forward that packet to that destination parent if the source node has a routing entry for one of that parent's children. In other words, there is no need to maintain a routing entry for a node that has at least one child. In formal terms, in the DODAG topology, if a node X has learnt a path to destination Y, then node X is able to communicate with any ancestor of that destination Y, providing that X knows that Y is located on that ancestor's sub-DODAG (i.e., Y is a child of X). This knowledge (i.e., X knows that Y is a child of it) can be provided by having the LBR (the DODAG root) maintaining the relationship information among the non-root routers and injecting this information later in the headers of transmitted packets. Indeed, to enable this mode, several small but critical changes to the format of DAO messages, their processing, and also routing tables needs to be involved as detailed next.

6.2.2 ROUTING TABLES STRUCTURE

In this section, an elaboration on the routing table structures that need to be maintained at the non-root routers (i.e., normal nodes) and the DODAG root to enable the LBRPL mode is presented. Hence, two variants of routing table structures are needed to be maintained by the normal nodes and the DODAG root. The first structure is for the non-root routers in which a

router needs only to store a two-part entry for each leaf child in its sub-DODAG in a form of destination-next hop relations. The two-part entry represents the child IP address and the IP address of the next hop to that child. The second table structure is for the DODAG root in which a three-part entry for each node in the network is maintained. The first part of the entry is the node's IP address (i.e., destination IP), the second part is the last advertised IP address of a leaf child node located in the sub-DODAG of that destination (if the destination is a leaf, the second part will be its own IP address), and the third part is the next-hop IP address to that leaf child. For the sake of simplicity, the second part of the entry is named thereafter the Branch Address. **Table 6-II** depicts an instance of the routing table that might be constructed for the topology shown in **Figure** 6-1 assuming that the nodes advertise their destinations according to their alphabetical order. In this table, you can observe that the root has a three-part entry for each node in the DODAG. For example, the entry (C, K, A) means that the root has a path for node C via node A and the Branch address for that node (i.e. node C) is K. On the other hand, the table shows that each non-root router has a two-part entry for each leaf child in that router sub-DODAG. For instance, the node C (a non-root router) has two entries for the leaf nodes J and K which are reachable via nodes F and G respectively. Referring back to the table, you also can observe that the root has learnt a path a path to destination J. Also, the root knows that nodes A, C, and F are ancestors of node J. Here, if the root has a packet destined to node F (it knows that F is an ancestor of J) and in order to correctly send the message to F, the root needs to include two addresses rather than the destination address only, namely, the destination address of node F itself as well as its Branch address (the address of node J) in the transmitted message. Here, any forwarder node on the path to the destination F should inspect both addresses to determine where the packet should go next. As per our example, the node A will forward the packet to C which is the next hop for destination J. The forwarding decision here is taken based on the Branch address and not based on the destination address. With the same

logic, node C will forward the received packet to node F. When node F receives the packet, it can infer from inspecting the destination address that it is the intended destination, consequently forwarding the packet to its upper layers.

Table 6-II. Routing tables in the proposed mode (LBRPL)

Nodes	Routing tables entries
LBR	(A,K,A), (B,M,B), (C,K,A), (D,H,A), (E,M,B), (F,J,A), (G,K,A), (H,H,A), (I,M, B), (J,J,A), (K,K,A), (L,L,B), (M,M,B)
A	(J,C),(K,C), (H,D)
C	(J,F),(K,G)
D	(H,H)
F	(J,J)
G	(K,K)
B	(L,E), (M, E)
E	(L,I), (M, I)
I	(L,L), (M, M)

6.2.3 AMENDED DAO FORMAT

As mentioned previously, each node willing to advertise itself as a destination should transmit a DAO to its preferred parent towards the DODAG root. This DAO would carry the necessary information that enable routers upward to build downward routes toward that destination. The format of the DAO base object including the Target option as it has been specified by the RPL standard is depicted in **Figure** 6-2.

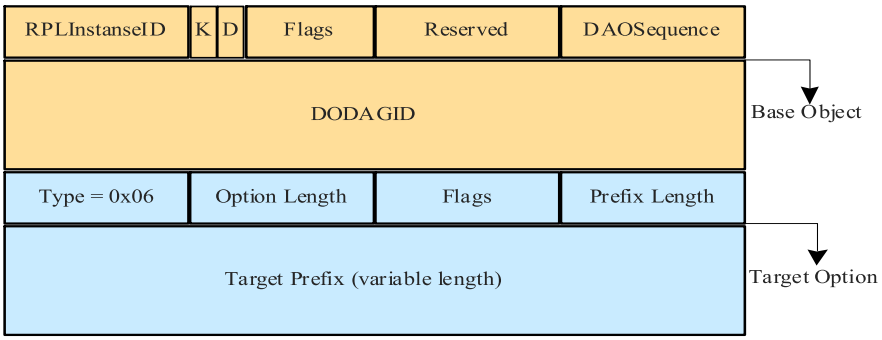


Figure 6-2. DAO format in the storing mode of RPL

In the storing mode, the RPL Target option is used by the DAO initiator to advertise its destination IPv6 address in the field of the Target Prefix. Thus, when a node running the storing mode receives a DAO, it adds the target prefix to its routing table along with the IPv6 address of the DAO sender as the next hop to the advertised target (destination). In the non-storing mode, the DAO must also contain information regarding the advertised target's parent IPv6 address in addition to the destination address. This information is carried in the RPL Transit option in the Parent Address field and it is meant to help the DODAG root in constructing the source routing headers based on "per hop" routing segments received from all the nodes in the network.

In our proposed approach, the root node needs to learn the last advertised leaf node in that destination sub-DODAG (i.e., the Branch address). This information can be deduced also based on child-parent relationship segments received from all network nodes. Thus, like the non-storing mode, the Parent Address field within the Transit Option must be presented in the DAOs transmitted using our proposed mode of operation. The proposed DAO format of our mode of operation is depicted in **Figure** 6-3. In fact, the Parent Address field serves two purposes in the proposed mode. First, it enables the DODAG root to calculate the branch addresses as mentioned earlier. Second, it enables the non-root routers to remove the routing entry of this parent that has been previously stored in the router tables. For example, if node F in **Figure** 6-1 has advertised its destination before node J, then nodes A and C would store a routing entry

for F as it is the last advertised leaf node at this moment in time. Later on, J would advertise its destination up to the DODAG root with setting F as the parent address. Here, A and C should remove F from their routing tables as it is no longer a leaf node. This is done by inspecting their routing tables for an entry similar to the parent of the advertised destination and then removing that entry.

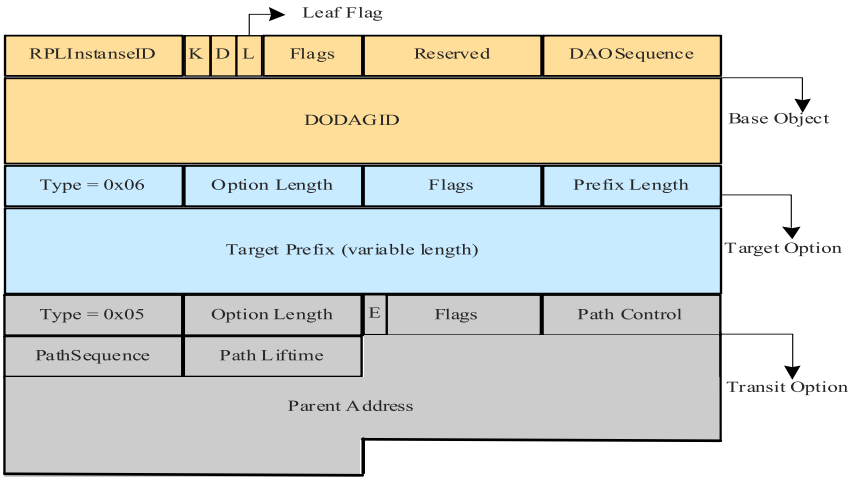


Figure 6-3. DAO format in the proposed mode

Removing a parent from a router's routing table by inspecting the destination's parent address in the Transit Information Option will solve the problem of the previously unknown parent that has been added to the routing table (i.e., it has been added before it becomes a parent). However, what would be the case upon receiving a DAO from a parent node (the node sent a DAO after it receives a DAO from one of its children). In the current format, the routers do not have the capacity to distinguish between a leaf and a non-leaf (parent) node advertised in DAO messages. As a result, the router will always add the destination advertised in the DAO message whether it is a leaf node or a parent, and it should wait until it receives a DAO from one of the added parent's children in order to be able to remove that parent. To resolve this issue and to allow for a router to prevent adding an already known parent to its routing table, one of the bits in the Flags field within the DAO base object is used to represent the current status of the DAO sender. This bit is named as the Leaf Flag (abbreviated as L flag) as depicted

in **Figure** 6-3. When this flag is set to zero, the DAO sender is considered a leaf node and a router must add the entry advertised in the DAO to its routing table. On the other hand, when the L flag is set to one, the DAO sender is already a parent and the DAO should be forwarded without adding that entry to routing table.

6.2.4 AMENDED RPL HOP-BY-HOP OPTION (HBH)

In an IPv6 network, a node willing to send an IPv6 packet to a specific destination should attach that destination address in the field of destination address in the IPv6 fixed header which is depicted in **Figure** 6-4. In our proposed MOP, the packet must carry two addresses in its header, the destination address and also the Branch address so that a non-root router can correctly forward that packet. In fact, the IPv6 fixed header contains only the information that is necessary for a router to perform forwarding decisions. All additional information, which is not always used, is carried in the form of Extension Headers placed between the Fixed Header and the Upper layer header. Several extension headers have been defined such as the HBH Option header and the Routing header. Among all defined extension headers, only the HBH header is to be processed by each non-root router along a packet's delivery path. All other headers are allowed to be processed only at the packet's destination so that to increase the speed of header processing and also to enhance the forwarding process performance. To maintain compatibility with IPv6, I choose to carry the branch address within a HBH Option as this address needs to be examined by every router along the destination's path. In this context, I found that the IETF has defined a new HBH Option called RPL Option for the purpose of data-path verification. In the data-path verification, some routing information is carried within the RPL option to help in detecting routing inconsistencies such as loops. The format of the HBH Option header including the RPL Option is depicted in **Figure** 6-5**a**. Exploiting this fact, I choose to carry the Branch IPv6 address within the already specified Hob-by-Hob RPL option by defining a new field, referred to as Branch Address in order to carry the branch IPv6 address.

This is depicted in **Figure** 6-5**b**. A forwarder node on the path to the destination should then inspect the Branch Address and also the Destination Address presented in the IPv6 fixed header for correctly forwarding the packet. The operation of how these two addresses should be used is explained in the Data-Plane Operation subsection.

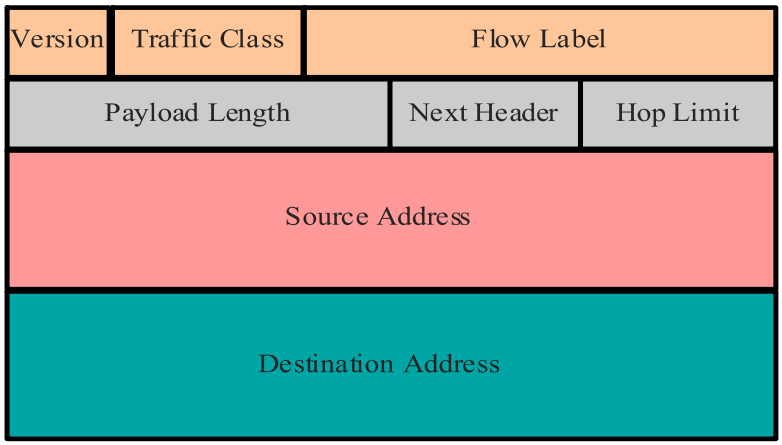


Figure 6-4. IPv6 header including the Destination Address

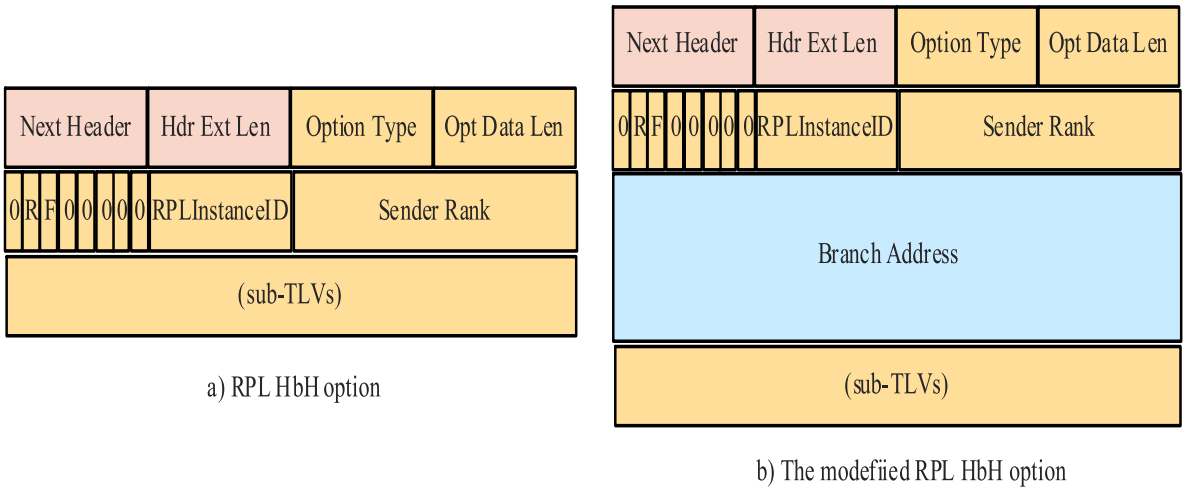


Figure 6-5. RPL Hop-by-Hop Option before and after modification

6.2.5 CONTROL PLANE OPERATION

In our proposed mode, each node wishing to participate in the downward routing from the sink to the normal nodes must unicast a DAO to its preferred parent advertising its destination prefix in addition to its parent IP address. A non-root node receiving that DAO should perform

the following operations: (a) if the L flag bit is set to zero, the node must add a routing entry for the advertised DAO to its routing table along with the DAO sender as the next-hop to the destination, otherwise, the routing entry must not be added, (b) remove the routing entry associated with parent address advertised in the DAO Transit information option if it exists, (c) set its own L flag bit to 1 in the next advertised DAO, and (d) forward the DAO further up to the DODAG root. The root is in the charge of calculating the Branch address for each destination in the network and attaching it to that destination information.

6.2.6 DATA PLANE OPERATION

Root Operation: In the RPL storing mode, when the root sends a packet to a destination in its sub-DODAG, it attaches the destination IP address in the Destination address field of the IPv6 header. Then, it forwards that packet to the next hop associated with that destination. A receiving router on the path to the destination simply inspects its routing table for an entry associated with that destination to look for which interface it should be forwarded next. This process is repeated by all the routers along the path to the destination until the packet finally reaches the intended destination. In our proposed mode of operation, inspecting the destination address will not be enough to correctly forward the packet to the next hop. This is because only the root stores the routing information of the entire topology while a non-root router stores only the routing information of the leaf nodes in its sub-DODAG. Thus, in addition to the destination address, the root needs to attach the branch address associated with that destination in the Branch Address field within the RPL HBH Option and then forward the packet to the next-hop.

Non-root router operation: As explained earlier, the branch address represents the last leaf address advertised in that destination sub-DODAG. Here, once the packet is received by a non-root router, the router first inspects the destination address within the IPv6 fixed header. If that destination address is not found, instead of dropping the packet, the router assumes that

the packet is forwarded to a leaf node. Hence, the router inspects the Branch Address within the RPL HBH Option to look up the leaf node associated with that address so that to forward the packet on that leaf next-hop interface. This process must be repeated until the packet reaches its intended destination.

Point-to-Point communication: In LBRPL, the point-to-point communication is applicable, however, there are slight differences compared to the storing mode of RPL. In storing mode of RPL a node wishing to send a message to another node in the DODAG should send the message up the DODAG until it finds a common ancestor for both communicating nodes. The transmitted message should then be forwarded down by the common ancestor until it finally reaches the destination. In LBRPL, there are two different cases based on whether the destination node is a leaf or a router. For leaf nodes, the message should be forwarded up the DODAG until an ancestor of that leaf is found, then it should be forwarded by that ancestor to the respective leaf node. On the other hand, in case the destination node is a router, only the root knows all the necessary information for correct forwarding. Thus, the message should be first forwarded to the root which then would attach the Branch address of the destination router in the transmitted message. This represents an additional overhead compared to the storing mode of RPL, however, it will be restricted to the case when a non-root router needs to communicate with another non-root router. All intermediate routers should finally inspect both the branch and the destination address of the message to correctly forward that message to its final destination.

6.3 PERFORMANCE EVALUATION AND DISCUSSION

6.3.1 SIMULATION

In this section, the proposed *downward* mode of operation is compared with the RPL storing mode in terms of average routing entries and the average PDR. The evaluation has been carried out by means of the Cooja [125] network simulator for the Contiki operating system

[127] which is the de-facto simulator of IoT constrained devices. This allows us to evaluate exactly the executable code that runs on real sensor motes such as Sky motes. The ContikiRPL library is used as a ground for implementing the proposed LBRPL. To build a network of stable routes, Objective Function Zero (OF0) is used for constructing the network paths. The MAC and underlying duty-cycling layers are set to the CSMA and ContikiMac protocols, respectively. For data model, every node is setup to send an application data packet every minute to the root at a random time with a maximum data length of 30 bytes. For each message received at the root, the root should instantly send an acknowledgment reply to the source node. For each setup, five experiments with different seeds are run in order to get statistically solid results with a simulation time of 20 virtual minutes for each experiment. The log files of all experiments are fed into a python script developed by us to extract the statistical results from these logs.

In the first set of experiments, an evaluation of both protocols in a grid topology comprising 50 nodes where nodes are uniformly spread over 100x100 m area and also where the border router (root) is placed in the middle of the network is carried out. **Figures 6-6 and 6-7** demonstrate a comparison between both RPL and LBRPL in terms of PDR and routing entries per node respectively as a function of routing capacity in a network with 99 nodes. Here, the routing capacity refers to the maximum number of routing entries that a node can hold in its routing table. As shown in **Figure** 6-6, both protocols achieve approximately similar PDR rates when the routing capacity is large enough to hold all entries in a parent sub-DODAG. However, RPL starts to suffer from degradation in the PDR at a routing capacity of 40 entries while LBRPL starts to show degradation at a routing capacity of 15 entries.

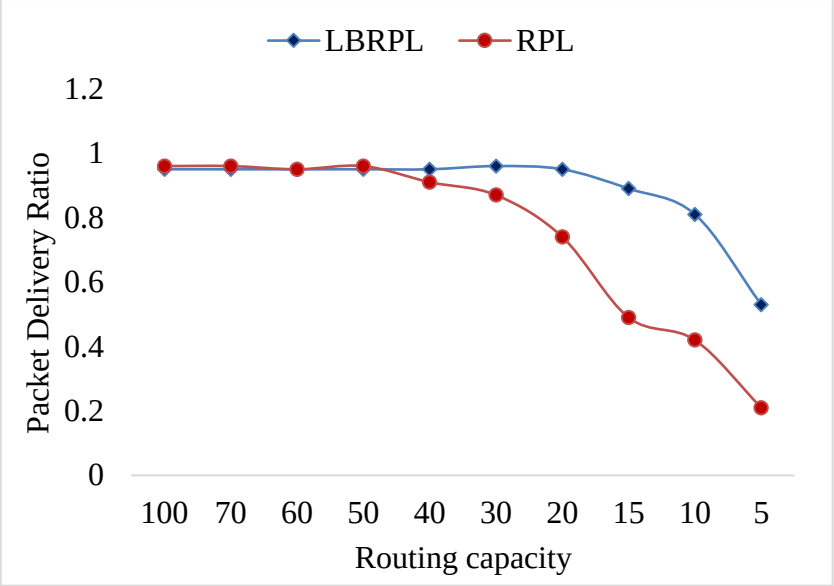


Figure 6-6. The PDR in RPL and LBRPL under different routing capacities of non-root routers

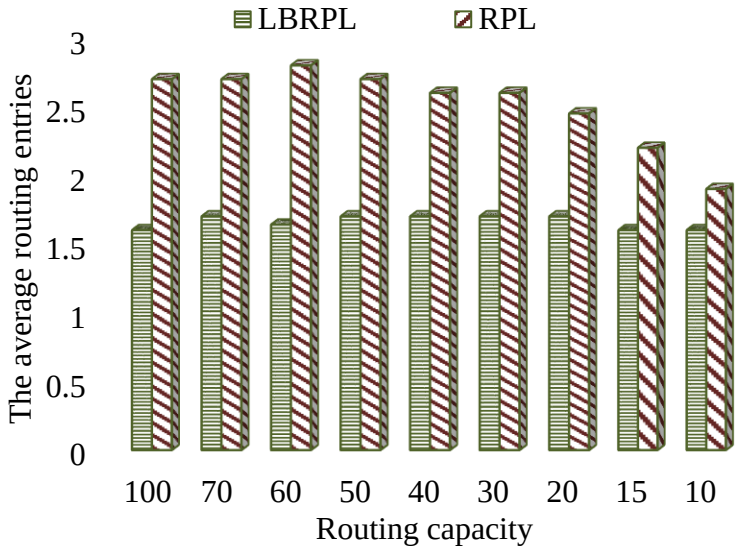


Figure 6-7. Average routing entries under different routing capacities

The early degradation of RPL in comparison to LBRPL in terms of PDR is attributed to the fact that every node running RPL should maintain a routing entry for all nodes in its own sub-DODAG while a node running LBRPL should only store information of leaf nodes. Hence, when the available routing capacity is not enough to accommodate all entries in the node's sub-DODAG, that node will deny storing any new destinations, leaving them unreachable for all

nodes upward to the DODAG root. **Figure** 6-7 shows a comparison between LBRPL and RPL in terms of the average routing entries as a function of routing capacity. As observed from the figure, LBRPL always achieved more efficient utilization of storage resources regardless of the permitted routing capacity. For instance, LBRPL has an average of 1.7 routing entries per node whereas RPL has an average of 2.6 routing entries per node at a routing capacity of 50 entries. This is also justified by the previous mentioned fact that LBRPL requires to maintain the routing states only for the leaf nodes in its routing tables rather than the whole group of destinations.

Indeed, the optimality of LBRPL depends largely on the number of leaf nodes in a given topology. The less the leaf nodes in a given deployment, the less the routing state needs to be maintained at the intermediate routers. Hence, in chain-like topologies, which is the case in many deployments, each intermediate router needs only to store the IPv6 address of the last node in the chain rather than the IPv6 addresses of all nodes in its sub-chain. To validate this case, an evaluation for the performance of both protocols in terms of PDR and average routing entries in a chain-like topology consisting of 40 nodes is carried out. **Figures 6-8** and **6-9** show a comparison between LBRPL and RPL in terms of number of routing entries and PDR respectively as a function of routing capacity. It is clear from the figures that LBRPL significantly outperforms RPL in terms of number of routing entries required to be maintained in each node's routing table without experiencing any degradation in the PDR. Regardless of the available routing capacity, LBRPL only needs to maintain an average of 0.974 entries per node. In contrast, RPL shows always degradation in its PDR soon the available routing capacity becomes less than the number of nodes in the chain. This is because RPL requires that each router on the chain should maintain a routing entry for each node in its sub-chain. Thus, when the number of the nodes in the chain exceeds the available routing capacity, the router is going to deny the routing entries beyond its routing capacity limit, leaving them unreachable from its

point of view and also from the point of view of all routers upward to the chain root. Hence, the root would have no option but to drop all packets destined to the destinations associated with the denied routing entries, negatively affecting application reliability.

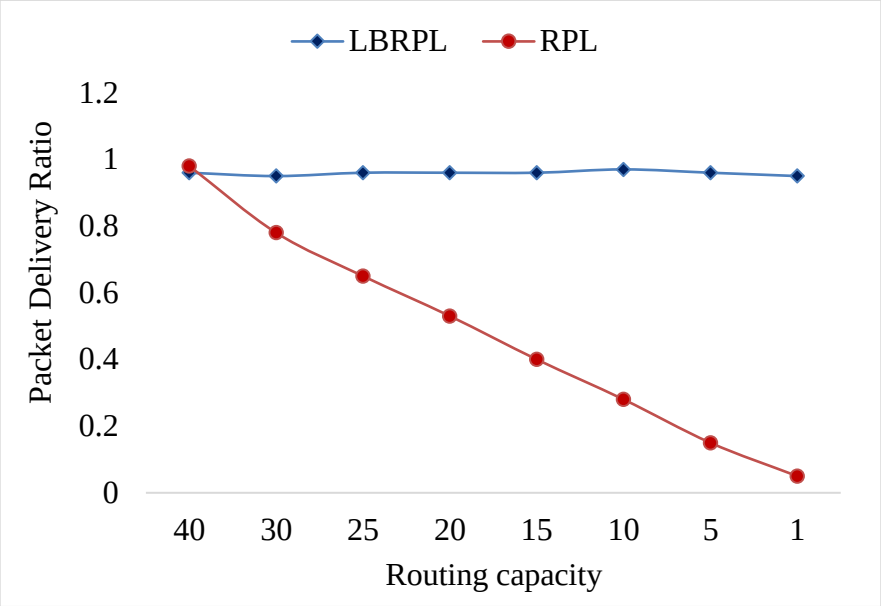


Figure 6-8. The PDR in RPL and LBRPL under different routing capacities in a chain-like topology

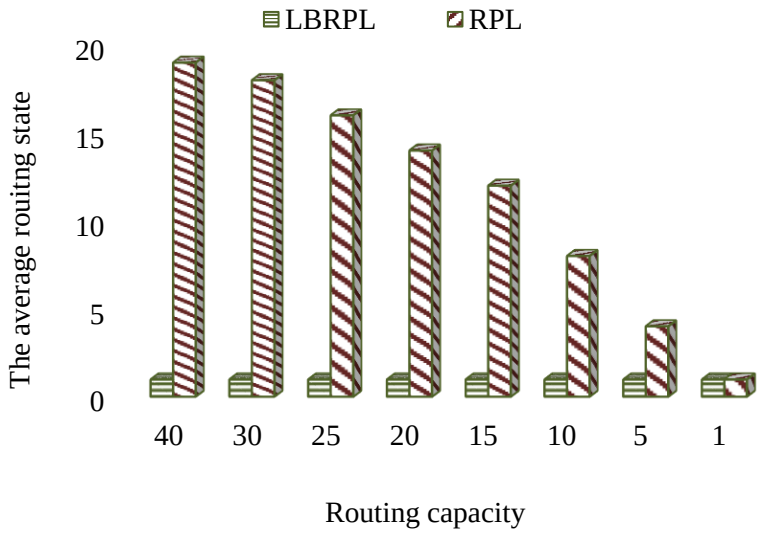


Figure 6-9. Average routing entries under different routing capacities in a chain-like topology

6.3.2 CONTROL AND DATA PLANES OVERHEAD ANALYSIS

As has been mentioned previously, LBRPL mandates the present of the optional Transit Option including the parent address in all transmitted DAOs to ensure the propagation of node

relations to the DODAG root so that the root can infer the Branch addresses of all nodes in the DODAG. Thus, an additional overhead of approximately 16 bytes is introduced in each transmitted DAO in comparing with RPL storing mode in which this option is not mandatory. Furthermore, in order for each data packet to be forwarded properly, LBRPL mandates the attaching of the Branch Address in each transmitted data packet going from the root to the nodes. This represents an extra overhead of a maximum size of 16 bytes (the actual amount of overhead ranges between 1 byte and 16 bytes depending on whether this address is compressed or not, our implementation uses uncompressed IPv6 Branch Addresses). Despite this additional overhead, the overhead impact on energy expenditure seems to be insignificant as depicted in **Figure** 6-10. The figure shows that there are minor differences between LBRPL and RPL in terms of energy consumption which confirms that benefits gained from the proposed protocol largely outweigh the shortcomings that the protocol may incur.

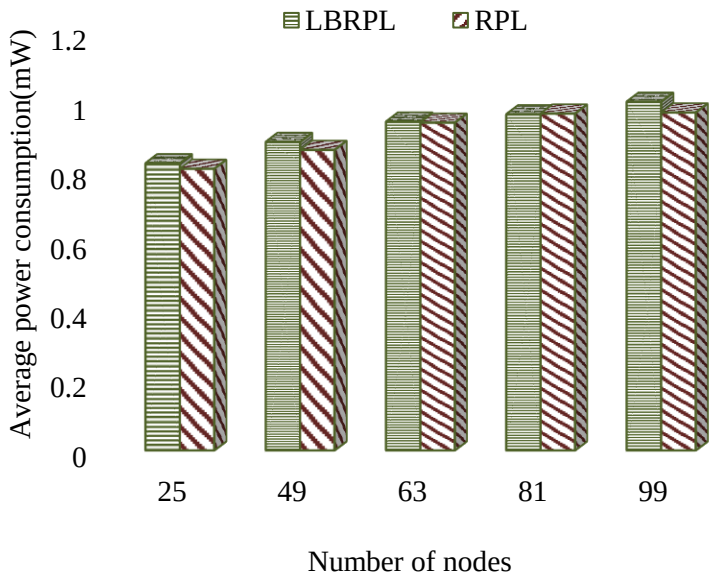


Figure 6-10. Power Consumption in LBRPL and RPL vs number of nodes

6.4 SUMMARY

In this chapter, a new routing mode is proposed for the RPL protocol that facilitates the construction of downward routes with minimal routing state. In this mode, a node needs only

to maintain the routing state of the leaf children, rather than the whole set of children, in its sub-DODAG. To enable such a mode, several critical changes to the format of the DAO messages, their processing, the structure of routing tables, and the operations of control and data planes have been proposed. A performance evaluation of the proposed protocol in comparison to the RPL standard has been carried out that highlights the efficiency of our proposed protocol in terms of routing state, PDR, and energy consumption.

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

This chapter summarises the problems addressed in this thesis, briefly discussing the contributions achieved and their limitations, as well as highlighting the potential avenues for future research.

7.1 THESIS SUMMARY AND OBJECTIVES REVIEW

The ultimate aim of this thesis has been to contribute to the advancement of LLN technology, a key enabling factor for the IoT, by addressing some of the key gaps of the RPL protocol, the LLNs routing standard, ultimately paving the way for widespread deployments of applications and services for such networks. To realize such an aim, several objectives have be defined as follows:

The first objective was to gain an in-depth knowledge and master the state-of-the-art of IoT and LLN concepts, their potential applications, and the new standard protocols stack of the IoT with a special focus on routing, which is achieved, as documented in Chapters 2. In This chapter, a thorough background on LLNs is presented providing an overview of LLN environments, characteristics, limitations, their unique routing challenges and routing requirements defined by the standardization bodies. It also elaborates on the relevant standards and radio communication technologies that underpin the transition of such networks into the Internet of Things (IoT) world.

The second objective of this thesis was to scrutinize and analyze the major concerns and key design issues of the standardized IoT routing primitives including the RPL standard and its routing maintenance algorithm Trickle. This goal is achieved and documented in Chapter 3 providing a comprehensive overview of RPL and Trickle, and highlighting their key limitations.

Guided by the literature review of Chapter 2 and Chapter 3, three major gaps related to RPL and Trickle have been identified as follows:

1) The standard lacks an efficient routing maintenance primitive that would offer a rapid convergence while maintaining very low overhead and power consumption profiles.
2) The standard lacks an efficient load-balancing objective function that would ensure a fair distribution of traffic between respective nodes while minimizing overhead and maintaining network stability.
3) The third identified limitation concerns the lack of the standard for an efficient routing primitive that addresses the memory limitations in IoT networks.

Hence, the third objective was to develop, implement and evaluate a new routing maintenance solution for LLNs that enhances the efficiency in terms of overhead, power consumption, and convergence time. This objective was achieved and documented in Chapter 4 that presents our major first contribution in which a new routing maintenance algorithm, named Drizzle, has been advised for LLNs. In general, Drizzle employs an adaptive approach in setting the value of the redundancy coefficients of nodes based on their transmission history so to boost network fairness and enhance the quality of discovered routes. In order to enable RPL networks to converge quickly, Drizzle removes the listen-only interval from the original RPL routing maintenance algorithm (i.e., Trickle). However, the removal of such interval introduces the so-called short-listen problem that may harm the scalability of the protocol at edge conditions and, hence, Drizzle introduced a new policy for setting the redundancy counter that governs the suppression mechanism. This new policy enabled Drizzle to mitigate the negative effect of the short-listen problem and, hence enhancing the protocol scalability while maintaining rapid convergence and low power consumption profiles. The performance evaluation of Drizzle in comparison to state-of-the-art routing maintenance algorithms has been presented in the same chapter. The results demonstrated that important performance

achievements have been realized with improvements of up to 80%, 26% and 20%, concerning control overhead convergence time, and power consumption, respectively, while maintaining comparable PDR.

The fourth objective was to develop a new route selection and optimization objective function that strives to fairly distribute the traffic among LLN nodes while maintaining stability. This objective was achieved and documented in Chapter 4 that presents our major second contribution. In particular, the chapter introduced a new load balancing OF for the RPL protocol named LBSR. The salient feature of the new proposed OF is its capacity to load balance traffic while ensuring that such a process will not significantly harm network stability. The first step towards realizing such a proposal was to decide on the optimal load-balancing metric and how efficiently such a metric can be estimated. Figuring out that the number of children metric should be used, a new routing primitive was devised to calculate that metric. Having decided on the optimal load balancing metric and the calculation procedure, the second step was to propose a load-balancing parent selection and optimization primitive based on the selected metric. To realize such a primitive, I propose that the parent selection should be done based on the lexical combination of number of children and a primary metric such as the hop count. However, due to the proactive nature of RPL in propagating routing information, I noticed that the routing decisions might be taken based on obsolete routing information especially the information related to the load-balancing metric. Thus, the notion of the *FastPropagation Timer* that ensures the timely propagation of routing information is proposed. The introduction of such a timer gave rise to the so-called herding-effect problem in which nodes keep changing their preferred parents to achieve load balancing, a behavior that harms the stability of the network. To address this problem, a new routing policy for switching the preferred parent has been advised. In this new policy and instead of having a node performing the parent selection process immediately after receiving a new DIO, the process of parent

selection is conducted according to a regular pre-specified scheduling interval. However, the proposed protocol excludes the first received DIO from this policy to allow faster convergence time at the stage of DODAG construction. In the same chapter, a performance evaluation of the proposed protocol in comparison to RPL is presented demonstrating improvements of up to 33% and 88% in terms of PDR and load distribution, respectively, while maintaining comparable power consumption rates.

The final objective was then to develop a new downward routing solution for LLNs that widens RPL applicability in bidirectional large-scale network and evaluate its validity, which has been addressed and documented in Chapter 6 that presents our major third contribution. In this chapter, the problem of memory limitations in RPL's networks is mitigated by proposing a Leaf-Based Downward Routing Mechanism for the RPL Protocol. In this mechanism, I have argued that a router needs only to maintain the routing state of the leaf children in its sub-DODAG rather than the whole set of children in order to enable the flow of data in the downward direction. This argument has been perceived based on the fact that a packet destined to a node must pass by its parent first, thus I have proposed a mechanism in which the routers only store the addresses of leaf nodes in their sub-DODAGs and through this saving precious memory resources and indirectly enhancing the reliability of the network. A performance evaluation of the proposed mechanism in comparison with the RPL standard has been also reported demonstrating its efficiency concerning average routing entries, PDR, and energy consumption with the percentage of improvements depending largely on the scale and topology of the network.

7.2 FUTURE DIRECTIONS

Three objectives have been the primary focus of this thesis including devising an efficient route maintenance algorithm, a load-balancing objective function, and a memory-efficient

downward routing mechanism for the RPL standard. While this thesis has made good progress towards realizing such objectives, there are still a myriad of challenges and research directions that need to be addressed which is outlined in the following subsections.

7.2.1 Downward Traffic Patterns

As discussed in the literature review, RPL did not pay much attention to the optimization of downward traffic (P2MP) compared to that of upward traffic (MP2P), giving rise to several limitations and drawbacks. While I have addressed one such limitation by proposing a leaf-based routing mechanism for RPL's network, several limitations in this context are still to be investigated as future work. These include, for instance, the issues of long source headers in the non-storing mode of RPL, and the under-specification of mechanisms that define the timing of DAO and DAO-ACK control messages.

7.2.2 Real Testbed Experimentations

Using simulations to validate the efficiency of network protocols has several advantages over testbed experiments. For instance, simulations can be easily controlled and configured making it easier to conduct several experiments in a shorter window of time [139]. In addition, simulations enable the modeling of large-scale networks, a task that is very expensive, if not impossible, using real testbed experimentation [139]. However, simulations do have their own limitations with the main limiting factor being their inability to reflect all aspects of real world scenarios casting some doubts on the trustworthiness of the results obtained. Hence, as a direction for future work, I aim to validate the efficiency of our proposed routing primitives in this thesis under real testbeds, such as the FIT IoT-LAB testbed [140], with different densities and under a wide range of operating conditions.

7.2.3 SINGLE-INSTANCE VS MULTI-INSTANCE OPTIMIZATION

One of the design considerations of RPL is that multiple applications with conflict routing requirements may run concurrently in a single physical topology. This can lead to multiple instances, where each has its own OF (i.e., one OF per instance), featuring one or more routing metrics, different mode of operations and routing policies. However, the simultaneous operation of multiple instances will increase the implementation and configuration complexity of the protocol, and RPL's specification does not provide a guidance on how this should be done [41] [43]. Some recent studies [41][43] propose removing this feature from the protocol but, while this may overcome the problems of interoperability and implementation complexity, it may hinder the capacity of the protocol to accommodate antagonistic requirements within a single network. Hence, I believe that it is too early to judge whether the multi-instance feature of RPL should be removed or not, especially in the absence of research studies that evaluate and compare both scenarios, an issue that represents another avenue for future research.

7.3 CONCLUDING REMARKS

Providing IoT networks with efficient communication protocols is a major step towards realizing such a promising paradigm. Indeed, if such networks are to be widely deployed, they should be able to satisfy the user requirements in terms of being reliable, scalable and energy-efficient even under harsh conditions. The results of this research set foundations for such a direction by developing several routing primitives that strive to satisfy the above-mentioned requirements. However, I assert that the achievements made in this research should be only seen as one-step towards achieving the vision of a truly felt Internet of Things, and there is still a need for more research efforts in order to fully realize such a vision.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” in *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," in *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347-2376, Fourth quarter 2015.

[3] J. Gantz and D. Reinsel, “The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east,” IDC iView: IDC Anal. Future, vol. 2007, pp. 1–16, December, 2012.

[4] J. Vasseur and A. Dunkels, “Interconnecting Smart Objects with IP: The Next Internet”, 1st ed., Burlington, MA, Morgan Kaufmann Publishers/Elsevier, Jun. 2010, pp. 1-432.

[5] J. Hui and D. Culler, "Extending IP to Low-Power, Wireless Personal Area Networks," in *IEEE Internet Computing*, vol. 12, no. 4, pp. 37-45, July-Aug. 2008.

[6] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, Sep. 2007.

[7] J. Hui, and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, Sep. 2011.

[8] T. Clausen, U. Herberg and M. Philipp, "A critical evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)," *The IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Wuhan, 2011, pp. 365-372.

[9] J. Hui, JP. Vasseur, D. Culler, and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, Mar. 2012.

[10] IETF (2009), Charter for Working Group [Online]. Available: http://www.ietf.org/dyn/wg/charter/roll-charter.html.

[11] C. Bormann, M. Ersue, and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.

[12] A. Brandt, J. Buron, and G. Porcu, "Home Automation Routing Requirements in Low-Power and Lossy Networks", RFC 5826, Apr. 2010.

[13] J. Martocci, P. De Mil, N. Riou, and W. Vermeylen, "Building Automation Routing Requirements in Low-Power and Lossy Networks", RFC 5867, June 2010.

[14] K. Pister, P. Thubert, S. Dwars, and T. Phinney, "Industrial Routing Requirements in Low-Power and Lossy Networks", RFC 5673, October 2009.

[15] M. Dohler, T. Watteyne, T. Winter, and D. Barthel, "Routing Requirements for Urban Low-Power and Lossy Networks", RFC 5548, May 2009.

[16] A. Brandt, E. Baccelli, R. Cragie, and P. van der Stok, "Applicability Statement: The Use of the Routing Protocol for Low-Power and Lossy Networks (RPL) Protocol Suite in Home Automation and Building Control", RFC 7733, Feb. 2016.

[17] N. Cam-Winget, J. Hui, and D. Popa, "Applicability Statement for the Routing Protocol for Low-Power and Lossy Networks (RPL) in Advanced Metering Infrastructure (AMI) Networks", RFC 7733, Jan. 2017.

[18] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, Mobile Computing, vol 353, pp. 153–181. Kluwer Academic Publishers, 1996.

[19] D. A. Maltz and D. B. Johnson and Y. Hu. "The dynamic source routing protoco (DSR) for mobile ad hoc networks for IPv4", IETF RFC 4728, February, 2007.

[20] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," *Mobile Computing Systems and Applications, Proceedings. WMCSA '99. Second IEEE Workshop on*, New Orleans, LA, 1999, pp. 90-100.

[21] T. Clausen (ed) and P. Jacquet (ed), "Optimized link state routing protocol (OLSR), "IETF RFC 3626, Experimental, October 2003.

[22] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum and L. Viennot, "Optimized link state routing protocol for ad hoc networks," Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century., 2001, pp. 62-68.

[23] J. Moy, “Open Shortest Path First Routing Protocol (OSPF version 2),” RFC 2328, Standards Track, April 1998.

[24] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, “Collection tree protocol,” *in Proc. 7th ACM Conf. Embedded Networked Sensor Syst.*, 2009, pp. 1–14.

[25] S. Dawson-Haggerty, A. Tavakoli and D. Culler, "Hydro: A Hybrid Routing Protocol for Low-Power and Lossy Networks," *2010 First IEEE International Conference on Smart Grid Communications*, Gaithersburg, MD, 2010, pp. 268-273.

[26] T. Clausen, A. C. de Verdiere, J. Yi, A. Niktash, Y. Igarashi, and U. Herberg, “The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng),” IETF, Draft, Oct 2012.

[27] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, Mar. 2012.

[28] J. Tripathi, J. C. de Oliveira and J. P. Vasseur, "A performance evaluation study of RPL: Routing Protocol for Low power and Lossy Networks," *the 44th Annual Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, 2010, pp. 1-6.

[29] J. Vasseur et al., “RPL: The IP routing protocol designed for low power and lossy networks,” *Internet Protocol for Smart Objects (IPSO) Alliance*, San Jose, CA, USA, 2011.

[30] O. Gaddour, A. Koubâa, “RPL in a nutshell: A survey”, in *Computer Networks*, Vol. 56, no. 14, pp. 3163-3178, 2012.

[31] P. Levis, N. Patel, D. Culler, S. Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks", *Proc. 1st USENIX/ACM Symp. Networked Systems Design and Implementation (NSDI’04)*, pp. 15-28, March, 2004.

[32] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The Trickle Algorithm", RFC 6206, March, 2011.

[33] J. Tripathi, J. de Oliveira and JP. Vasseur, Ed., "Performance Evaluation of the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6687, Oct. 2012.

[34] E. Ancillotti, R. Bruno and M. Conti, "RPL routing protocol in advanced metering infrastructures: An analysis of the unreliability problems," in *Sustainable Internet and ICT for Sustainability (SustainIT)*, Pisa, 2012, pp. 1-10.

[35] U. Herberg and T. Clausen, "A comparative performance study of the routing protocols LOAD and RPL with bi-directional traffic in low-power and lossy networks (LLN)," in *Proc*. of *the 8th ACM Symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, Oct-Nov 2011, pp. 73–80.

[36] J. Yi, T. Clausen and Y. Igarashi, "Evaluation of routing protocol for low power and Lossy Networks: LOADng and RPL," in *the IEEE Conference on Wireless Sensor (ICWISE)*, Kuching, 2013, pp. 19-24.

[37] C. Cobârzan, J. Montavont and T. Noël, "Analysis and performance evaluation of RPL under mobility," in *the IEEE Symposium on Computers and Communications (ISCC)*, Funchal, June 2014, pp. 1-6.

[38] S. Elyengui, R. Bouhouchi and T. Ezzedine, "A comparative performance study of the routing protocols RPL, LOADng and LOADng-CTP with bidirectional traffic for AMI scenario," in *the IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, Sept. 2015, pp. 561-568.

[39] E. Ancillotti, R. Bruno and M. Conti, "The role of the RPL routing protocol for smart grid communications," in *the IEEE Communications Magazine*, vol. 51, no. 1, pp. 75-83, Jan. 2013.

[40] T. Clausen and U. Herberg, "Some considerations on routing in particular and lossy environments," *in 1st IAB Interconnecting Smart Objects with the Internet Workshop*, Mar. 2011.

[41] A. Parasuram, "An Analysis of the RPL Routing Standard for Low Power and Lossy Networks," Master's thesis, EECS Department, University of California, Berkeley, May 2016.

[42] O. Afonso, and T. VazÃ£o. "Low-Power and Lossy Networks under Mobility: A Survey." in *Computer Networks*, vol. 107, no. 2, pp. 339–352, Oct. 2016.

[43] H. S. Kim, J. Ko, D. E. Culler and J. Paek, "Challenging the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL): A Survey," in *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2502-2525, Fourthquarter 2017.

[44] X. Liu, Z. Sheng, C. Yin, F. Ali and D. Roggen, "Performance Analysis of Routing Protocol for Low Power and Lossy Networks (RPL) in Large Scale Networks," in *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2172-2185, Dec. 2017.

[45] O. Iova, P. Picco, T. Istomin and C. Kiraly, "RPL: The Routing Standard for the Internet of Things... Or Is It?" in *IEEE Communications Magazine*, vol. 54, no. 12, pp. 16-22, Dec. 2016.

[46] D. Airehrour, J. Gutierrez, and S. K. Ray, "Secure routing for Internet of Things: A survey," in *Journal of Network and Computer Applications*, vol. 66, pp. 198–213, May 2016.

[47] N. Saputro, K. Akkaya, and S. Uludag., "A survey of routing protocols for smart grid communications", in *Computer Networks*, vol. 56, no. 11, pp. 2742–2771, Jul. 2012.

[48] C. Vallati and E. Mingozzi, "Trickle-F: Fair broadcast suppression to improve energy-efficient route formation with the RPL routing protocol," in *the Sustainable Internet and ICT for Sustainability (SustainIT)*, Palermo, Oct. 2013, pp. 1-9.

[49] B. Djamaa and M. Richardson, "Optimizing the Trickle Algorithm," in *IEEE Communications Letters*, vol. 19, no. 5, pp. 819-822, May 2015.

[50] H. Lamaazi and N. Benamar, "A Novel Approach for RPL Assessment Based on the Objective Function and Trickle Optimizations," Wireless Communications and Mobile Computing, vol. 2019, Article ID 4605095, 9 pages, 2019.

[51] S. Goyal and T. Chand, "Improved Trickle Algorithm for Routing Protocol for Low Power and Lossy Networks," in *IEEE Sensors Journal*, vol. 18, no. 5, pp. 2178-2183, 1 March1, 2018.

[52] T. M. M. Meyfroyt, "An analytic evaluation of the Trickle algorithm: Towards efficient, fair, fast and reliable data dissemination," in *the IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Boston, MA, Jun. 2015, pp. 1-2.

[53] T. M. M. Meyfroyt, M. Stolikj and J. J. Lukkien, "Adaptive broadcast suppression for Trickle-based protocols," *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Boston, MA, Jun. 2015, pp. 1-9.

[54] O. Gnawali and P. Levis, "Recommendations for Efficient Implementation of RPL," Internet Draft, March, 2013.

[55] T. Coladon, M. Vučinić and B. Tourancheau, "Multiple redundancy constants with trickle," in *the 26th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC*), Hong Kong, Aug-Sep 2015, pp. 1951-1956.

[56] X. Liu, J. Guo, G. Bhatti, P. Orlik and K. Parsons, "Load balanced routing for low power and lossy networks," in the *IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2013, Shanghai, pp. 2238-2243.

[57] H. S. Kim, J. Paek and S. Bahk, "QU-RPL: Queue utilization based RPL for load balancing in large scale industrial applications," in *the 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON),* Seattle, WA, Jun. 2015, pp. 265-273.

[58] H. Kim, H. Kim, J. Paek and S. Bahk, "Load Balancing Under Heavy Traffic in RPL Routing Protocol for Low Power and Lossy Networks," in *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 964-979, 1 April 2017.

[59] J. Tripathi and J. C. De Oliveira, "Quantifying load imbalance: A practical implementation for data collection in low power lossy networks," *47th Annual Conference on Information Sciences and Systems (CISS)*, Baltimore, MD, 2013, pp. 1-6.

[60] J. A. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter and B. Heile, "IEEE 802.15.4: A Developing Standard for Low-Power, Low-Cost Wireless Personal Area Networks," IEEE Network, vol. 15, no. 5, pp. 12-19, Sept./Oct. 2001.

[61] IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)," in *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)*, pp.1-314, Sept. 2011.

[62] S. Deering, and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December, 1998.

[63] S. Gopinath Rao, Z. Suryady, U. Sarwar and M. Abbas, "A gateway solution for IPv6 wireless sensor networks," *International Conference on Ultra Modern Telecommunications & Workshops*, St. Petersburg, Russia, 2009, pp. 1-6.

[64] T. Teubler, M. A. Hail and H. Hellbruck, "Transparent Integration of Non-IP WSN into IP Based Networks," *IEEE 8th International Conference on Distributed Computing in Sensor Systems*, Hangzhou, 2012, pp. 353-358.

[65] A. G. F. Elias, J. J. P. C. Rodrigues, L. M. L. Oliveira and Liang Zhou, "IPv4/IPv6 transition mechanisms for ubiquitous wireless sensor networks monitoring," *Fifth International Conference on Ubiquitous and Future Networks (ICUFN)*, Da Nang, 2013, pp. 192-196.

[66] L. Mainetti, L. Patrono and A. Vilei, "Evolution of wireless sensor networks towards the Internet of Things: A survey," *19th International Conference on Software, Telecommunications and Computer Networks*, Split, 2011, pp. 1-6.

[67] N. Khalil, M. R. Abid, D. Benhaddou and M. Gerndt, "Wireless sensors networks for Internet of Things," *IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, Singapore, 2014, pp. 1-6.

[68] N. Accettura and G. Piro, "Optimal and secure protocols in the IETF 6TiSCH communication stack," *IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, Istanbul, 2014, pp. 1469-1474.

[69] Y. Al-Nidawi and A. H. Kemp, "Mobility Aware Framework for Timeslotted Channel Hopping IEEE 802.15.4e Sensor Networks," in *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7112-7125, Dec. 2015.

[70] M. R. Palattella, N. Accettura, L. A. Grieco, G. Boggia, M. Dohler and T. Engel, "On Optimal Scheduling in Duty-Cycled Industrial IoT Applications Using IEEE802.15.4e TSCH," in *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3655-3666, Oct. 2013.

[71] D. Dujovne, T. Watteyne, X. Vilajosana and P. Thubert, "6TiSCH: deterministic IP-enabled industrial internet (of things)," in *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36-41, Dec. 2014.

[72] Bluetooth Special Interest Group (SIG) (Dec. 2016), Bluetooth Core Specification Version 5 [Online]. Available: https://www.bluetooth.com/ specifications/Bluetooth-core-specification.

[73] Bluetooth Special Interest Group (SIG), Radio Versions [Online], available: https://www.bluetooth.com/bluetooth-technology/radio-versions.

[74] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of Bluetooth low energy: An emerging low-power wireless technology," in *Sensors*, vol. 12, no. 9, pp. 11 734–11 753, 2012.

[75] IEEE Standard for Broadband over Power Line Networks: Medium Access Control and Physical Layer Specifications," in *IEEE Std 1901-2010* , pp.1-1586, Dec. 30 2010.

[76] C. Cano, A. Pittolo, D. Malone, L. Lampe, A. M. Tonello and A. G. Dabak, "State of the Art in Power Line Communications: From the Applications to the Medium," in *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 7, pp. 1935-1952, Jul. 2016.

[77] IEEE Draft Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment 2: Sub 1 GHz License Exempt Operation," in *IEEE P802.11ah/D6.0, February 2016 (Amendment to IEEE Std 802.11REVmc/D5.0)* , vol., no., pp.1-645, Jan. 2016.

[78] D. Bankov, E. Khorov, A. Lyakhov and E. Stepanova, "Fast centralized authentication in Wi-Fi HaLow networks," *IEEE International Conference on Communications (ICC)*, Paris, May 2017, pp. 1-6.

[79] Y. Wang, K. K. Chai, Y. Chen, J. Schormans and J. Loo, "Energy-aware Restricted Access Window control with retransmission scheme for IEEE 802.11ah (Wi-Fi HaLow) based networks," *13th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, Jackson, WY, 2017, pp. 69-76.

[80] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, “Wireless sensor networks: a survey”, in *Computer Networks*, vol. 38, no. 4, pp. 393-422, ISSN 1389-1286, Mar. 2002.

[81] Q. Tang, L. Yang, G. B. Giannakis and T. Qin, "Battery Power Efficiency of PPM and FSK in Wireless Sensor Networks," in *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1308-1319, Apr. 2007.

[82] A. Meier, T. Rein, J. Beutel and L. Thiele, "Coping with unreliable channels: Efficient link estimation for low-power wireless sensor networks," *5th International Conference on Networked Sensing Systems*, Kanazawa, 2008, pp. 19-26.

[83] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks,” in *Proc. 1st international conference on Embedded networked sensor systems*, 2003, pp. 1–13.

[84] S. Movassaghi, M. Abolhasan, J. Lipman, D. Smith and A. Jamalipour, "Wireless Body Area Networks: A Survey," in *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1658-1686, Third Quarter 2014.

[85] P. Thubert, “Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL),” IETF RFC 6552, Mar. 2012.

[86] O. Gnawali and P. Levis, “The Minimum Rank with Hysteresis Objective Function”, IETF RFC 6719, Sep. 2012.

[87] JP. Vasseur, M. Kim, K. Pister, N. Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, Mar. 2012.

[88] P. Karkazis *et al*., "Design of primary and composite routing metrics for RPL-compliant Wireless Sensor Networks," in *the International Conference on Telecommunications and Multimedia (TEMU)*, Chania, Jul-Aug. 2012, pp. 13-18.

[89] L. H. Chang, T. H. Lee, S. J. Chen and C. Y. Liao, "Energy-Efficient Oriented Routing Algorithm in Wireless Sensor Networks," in *the IEEE International Conference on Systems, Man, and Cybernetics*, Manchester, 2013, pp. 3813-3818.

[90] C. Abreu, M. Ricardo, and P. M. Mendes, “Energy-aware routing for biomedical wireless sensor networks,” in *the Journal of Networks and Computer Applications*, vol. 40, pp. 270–278, Apr. 2014.

[91] S. Capone, R. Brama, N. Accettura, D. Striccoli and G. Boggia, "An Energy Efficient and Reliable Composite Metric for RPL Organized Networks," in *the 12th IEEE International Conference on Embedded and Ubiquitous Computing*, Milano, Aug. 2014, pp. 178-184.

[92] O. Gaddour, A. Koubâa, N. Baccour and M. Abid, "OF-FL: QoS-aware fuzzy logic objective function for the RPL routing protocol," in the 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), Hammamet, May 2014, pp. 365-372.

[93] W. Xiao, J. Liu, N. Jiang and H. Shi, "An optimization of the object function for routing protocol of low-power and Lossy networks," in *the 2nd International Conference on Systems and Informatics*, Shanghai, Nov. 2014, pp. 515-519.

[94] J. Ko, J. Jeong, J. Park, J. A. Jun, and N. Kim, “Towards full RPL interoperability: addressing the case with downwards routing interoperability,” in *Proc. of the 10th ACM Conference on Embedded Network Sensor Systems*, Nov. 2012, pp. 353–354.

[95] J. Ko, J. Jeong, J. Park, J. A. Jun, O. Gnawali, and J. Paek, “DualMOP-RPL: Supporting Multiple Modes of Downward Routing in a Single RPL Networks,” in *ACM Transactions on Sensor Networks (TOSN)*, vol. 11, no. 2, pp. 1–20, Feb. 2015.

[96] W. Gan, Z. Shi, C. Zhang, L. Sun and D. Ionescu, "MERPL: A more memory-efficient storing mode in RPL," in *the 19th IEEE International Conference on Networks (ICON)*, Singapore, Dec. 2013, pp. 1-5.

[97] C. Kiraly, T. Istomin, O. Iova and G. P. Picco, "D-RPL: Overcoming memory limitations in RPL point-to-multipoint routing," in *the IEEE 40th Conference on Local Computer Networks (LCN)*, Clearwater Beach, FL, Oct. 2015, pp. 157-160.

[98] Y. Chen, J.-P. Chanet, K.-M. Hou, H. Shi, and G. de Sousa, “A Scalable Context-Aware Objective Function (SCAOF) of Routing Protocol for Agricultural Low-Power and Lossy Networks (RPAL),” in *Sensors*, vol. 15, no. 8, pp. 19507–19540, Aug. 2015.

[99] H. Matsuura, "New routing framework for RPL: Constructing power-efficient wireless sensor network," in *the IEEE Network Operations and Management Symposium (NOMS)*, Krakow, May 2014, pp. 1-9.

[100] X. Yang, J. Guo, P. Orlik, K. Parsons and K. Ishibashi, "Stability metric based routing protocol for low-power and lossy networks," in *the IEEE International Conference on Communications (ICC)*, Sydney, NSW, June 2014, pp. 3688-3693.

[101] Wang, Z., Zhang, L., Zheng, Z. et al. Peer-to-Peer Netw. Appl. (2018) 11.

[102] R. Ben Abdessalem, N. Tabbane N, "RPL-SCSP: A Network-MAC Cross-Layer Design for Wireless Sensor Networks", in *Proc. of Ninth International Conference on Wireless Communication and Sensor Networks*, New Delhi,, Apr. 2014, pp 27-35.

[103] T. G. Harshavardhana, B. S. Vineeth, S. V. R. Anand and M. Hegde, "Power control and cross-layer design of RPL objective function for low power and lossy networks," the 10th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, 2018, pp. 214-219.

[104] P. O. Kamgueu, E. Nataf and T. Ndie Djotio, "On design and deployment of fuzzy-based metric for routing in low-power and lossy networks," in *the IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops)*, Clearwater Beach, FL, Oct. 2015, pp. 789-795.

[105] S. Rekik, N. Baccour, M. Jmaiel and K. Drira, "Low-Power link quality estimation in smart grid environments," in *the International Wireless Communications and Mobile Computing Conference (IWCMC)*, Dubrovnik, Aug. 2015, pp. 1211-1216.

[106] S. Rekik, N. Baccour, M. Jmaiel and K. Drira, "Holistic link quality estimation-based routing metric for RPL networks in smart grids," in *the IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Valencia, Sept. 2016, pp. 1-6.

[107] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four Bit Wireless Link EstimationFour-bit wireless link estimation. in *Proc. of the Sixth Workshop on Hot Topics in Networks*, Atlanta, GA, Nov. 2007.

[108] O. Iova, F. Theoleyre and T. Noel, "Exploiting multiple parents in RPL to improve both the network lifetime and its stability," 2015 IEEE International Conference on Communications (ICC), London, Jun. 2015, pp. 610-616.

[109] O. Iova, F. Theoleyre, and T. Noel, "Using multiparent routing in RPL to increase the stability and the lifetime of the network," in *Ad Hoc Networks*, vol. 29, pp. 45–62, Jun. 2015.

[110] M. A. Lodhi, A. Rehman, M. M. Khan and F. B. Hussain, "Multiple path RPL for low power lossy networks," in *the IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*, Bandung, Aug. 2015, pp. 279-284.

[111] W. Tang, X. Ma, J. Huang, and J. Wei, "Toward Improved RPL: A Congestion Avoidance Multipath Routing Protocol with Time Factor for Wireless Sensor Networks," in *Journal of Sensors*, vol. 2016, pp. 1–11, 2016.

[112] S. S. Wang, H. T. Liu and Y. D. Chen, "Link quality aware routing protocol for low-power and lossy networks," in *the IEEE Wireless Communications and Networking Conference (WCNC)*, Istanbul, May 2014, pp. 2588-2593.

[113] D. Todolí-Ferrandis, S. Santonja-Climent, V. Sempere-Payá and J. Silvestre-Blanes, "RPL routing in a real life scenario with an energy efficient objective function," in *the 23rd Telecommunications Forum Telfor (TELFOR)*, Belgrade, Nov. 2015, pp. 285-288.

[114] M. Alishahi, M. H. Yaghmaee Moghaddam, and H. R. Pourreza, "Multi-class routing protocol using virtualization and SDN-enabled architecture for smart grid," *in Peer-to-Peer Networking and Applications*, vol. 11, no. 3, pp. 380-396, Dec. 2016.

[115] A. Hassan, S. Alshomrani, A. Altalhi and S. Ahsan, "Improved routing metrics for energy constrained interconnected devices in low-power and lossy networks," in *Journal of Communications and Networks*, vol. 18, no. 3, pp. 327-332, Jun. 2016.

[116] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," in *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54-69, Jul. 2005.

[117] S. Adler, S. Pfeiffer, H. Will, T. Hillebrandt and J. Schiller, "Measuring the distance between wireless sensor nodes with standard hardware," in *the 9th Workshop on Positioning, Navigation and Communication*, Dresden, Mar. 2012, pp. 114-119.

[118] M. Radi, B. Dezfouli, K. A. Bakar, and M. Lee, "Multipath routing in wireless sensor networks: Survey and research challenges," in *Sensors*, vol. 12, no. 1, pp. 650–685, Jan. 2012.

[119] S. Mueller, R. Tsang, and D. Ghosal, Multipath routing in mobile ad hoc networks: Issues and challenges, in *Performance Tools and Applications to Networked Systems*, vol. 2965 of Lecture Notes in Computer Science. Springer-Verlag, 2004.

[120] S. Roy, S. Bandyopadhyay, T. Ueda, and K. Hasuike, "Multipath Routing in Ad Hoc Wireless Networks with Omni Directional and Directional Antenna: A Comparative Study," *Proc. of the 4th International Workshop on Distributed Computing, Mobile and Wireless Computing*, Dec. 2002, pp. 184–191.

[121] J. Hui and R. Kelsey, "Multicast Protocol for Low-Power and Lossy Networks (MPL)," RFC 7731, Feb. 2016.

[122] N. A. Pantazis, S. A. Nikolidakis and D. D. Vergados, "Energy-Efficient Routing Protocols in Wireless Sensor Networks: A Survey," in *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 551-591, Second Quarter 2013.

[123] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," in *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6-28, Dec. 2004.

[124] M. Qasem, A. Al-Dubai, I. Romdhani, B. Ghaleb and W. Gharibi, "A new efficient objective function for routing in Internet of Things paradigm," 2016 IEEE Conference on Standards for Communications and Networking (CSCN), Berlin, pp. 1-6, 2016.

[125] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne and T. Voigt, "Cross-Level Sensor Network Simulation with COOJA," *Proc. of the 31st IEEE Conference on Local Computer Networks*, Tampa, FL, Nov. 2006, pp. 641-648.

[126] A. Dunkels, B. Gronvall and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *the 29th Annual IEEE International Conference on Local Computer Networks*, Nov. 2004, pp. 455-462.

[127] A. Dunkels et al (2006), Contiki-os [Online], Available: https://github.com/ contiki-os/ contiki/tree/master/core/net/rpl.

[128] N. Tsiftes, J. Eriksson, and A. Dunkels, "Low-Power Wireless IPv6 Routing with ContikiRPL," in *9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, Apr. 2010, pp. 406-407.

[129] G. Oikonomou et al (2006), contiki-ng [Online], Available: https://github.com/contiki-ng/contiki-ng/tree/develop/os/net/routing.

[130] E. Decker et al (2004), tinyos [Online], Avialable: https://github.com/ tinyos/tinyos-main/tree/master/tos/lib/net/rpl

[131] H. Petersen et al (2010), RIOT-OS [Online], Available: https://github.com/ RIOT-OS/RIOT/tree/master/sys/net/gnrc/routing/rpl.

[132] E. Baccelli, O. Hahm, M. Gunes, M. Wahlisch and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Turin, 2013, pp. 79-80.

[133] M. Richardson et al, Unstrung Features [Online], Available: http://unstrung.sandelman.ca/

[134] M. Richardson et al (2009), Unstrung [Online], Available: https://github.com/AnimaGUS-minerva/unstrung

[135] T. Cheneau (2013), SimpleRPL [Online], Available: https://github.com/ tcheneau/simpleRPL

[136] Cisco, (2015), Routing Protocol for LLN (RPL) Configuration Guide, Cisco IOS Release 15M&T [Online] Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/rpl/configuration/15-mt/rpl-15-mt-book.html.

[137] J. Eriksson, A. Dunkels, N. Finne, F. ¨ Osterlind, and T. Voigt, “Mspsim an extensible simulator for msp430-equipped sensor boards,” in Proceedings of the European Conference on Wireless Sensor Networks (EWSN ’07), Delft, The Netherlands, 2007.

[138] E. Ben Hamida (2009), WSNet/Worldsens simulator [Online], Available: http://wsnet.gforge.inria.fr/.

[139] K. Tan, D. Wu, A. Chan and P. Mohapatra, "Comparing simulation tools and experimental testbeds for wireless mesh networks," 2010 IEEE International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), Montrreal, QC, 2010, pp. 1-9.

[140] C. Adjih *et al*., "FIT IoT-LAB: A large scale open experimental IoT testbed," *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, Milan, Dec. 2015, pp. 459-464.