Service-Differentiated Cooperative Routing in the Internet of Things

Dissertation

zur Erlangung des mathematisch-naturwissenschaftlichen Doktorgrades

"Doctor rerum naturalium"

der Georg-August-Universität Göttingen

im Promotionsprogramm Computer Science (PCS)
der Georg-August University School of Science (GAUSS)

vorgelegt von

Milad Ayoub
aus Syrien

Göttingen, 2023

Betreuungsausschuss:

Prof. Dr. Dieter Hogrefe,
Telematics Group, Institut für Informatik
Georg-August-Universität Göttingen

Prof. Dr. Xiaoming Fu,
Computer Networks Group, Institut für Informatik,
Georg-August-Universität Göttingen

Mitglieder der Prüfungskommission:

Referent: Prof. Dr. Dieter Hogrefe,
Telematics Group, Institut für Informatik,
Georg-August-Universität Göttingen

Korreferent: Prof. Dr. Xiaoming Fu,
Computer Networks Group, Institut für Informatik,
Georg-August-Universität Göttingen

Weitere Mitglieder der Prüfungskommission:

Prof. Dr.-Ing. Marcus Baum,
Data Fusion Group, Institut für Informatik
Georg-August-Universität Göttingen

Prof. Dr. Carsten Damm,
Theoretical Computer Science Group, Institut für Informatik
Georg-August-Universität Göttingen

Prof. Dr. Florin Manea,
Fundamentals of Computer Science Group, Institut für Informatik
Georg-August-Universität Göttingen

Prof. Dr. Kerstin Strecker,
Didactics of Informatics Group, Institut für Informatik
Georg-August-Universität Göttingen

Tag der mündlichen Prüfung:

03. März 2023

In memory of my loving parents

To Syria, my homeland, hoping it restores its peace and prosperity soon.

Acknowledgment

Countless people supported my efforts in this thesis. First and foremost, I would like to sincerely thank Prof. Dr. Dieter Hogrefe for supervising me and giving me this life-changing opportunity to purse a Ph.D. at the University of Göttingen. I am very grateful for his invaluable assistance and guidance. Learning from and working under his supervision was a great privilege and honor.

I am also grateful to my second supervisor Prof. Dr. Xiaoming Fu, for his comments and helpful feedback. His remarks have greatly improved this thesis.

Furthermore, I would like to thank Prof. Dr.-Ing. Marcus Baum, Prof. Dr. Carsten Damm, Prof. Dr. Florin Manea, and Prof. Dr. Kerstin Strecker for serving on my examination committee. Their comments and suggestions have improved the quality of this thesis.

I am fortunate to have been part of the Telematics Research Group. Many thanks to my friends and colleagues: Dr. Arne Bochem, Dr. Hang Zhang, Dr. Parisa Memarmoshrefi, and all previous group members, for the reviews, discussions, and lively conversations.

I am thankful to the Catholic Academic Exchange Service (KAAD) for their financial sponsorship of my Ph.D.. I am grateful for their personal and social support during my study. I would like to give my sincere thanks to the general secretary Dr. Nora Kalbarczyk and to Mrs. Santra Sontowski from the Middle East department.

My gratitude also goes to the Erasmus Mundus ASSUR scholarship program for giving me the rare opportunity of a research visit to Germany. The guidance from Dr. Vitali Altholz and Mrs. Janja Mohorko was very helpful for me in my first months in Germany.

I sincerely thank the Katholische Hochschulgemeinde (KHG) Göttingen for all the activities, projects, and events I was lucky to join there. I am thankful to Ms. Ximena Ordonez for all her compassion and heartful encouragement.

I can not express enough gratitude for my siblings, Mai and Janada, for their endless love, support, and encouragement in different stages of my life. I am lucky to have my lovely nephews in my life, Mohannad and Milad, who fill my heart with joy every time we talk.

My thankfulness goes wholeheartedly to my dearest Johanna Kückes for her unconditional love, patience, and support. Her cheerful spirit has always filled my heart with joy. I would like to express my profound thanks to Martin and Eva Kückes for their constant motivation and for being my second family in Germany. I am additionally grateful to all my friends from my homeland who kept sending me their encouragement from overseas. Finally, I am incredibly thankful for all the friends I have made in Göttingen; each of you has made my stay more fruitful, productive, and joyful. I am blessed to have you in my life.

ABSTRACT

The IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) was designed to operate with different Internet of Things (IoT) applications ranging from regular, to critical, to alarm/sporadic. That is due to its ability to support service differentiation by forwarding multiple traffic classes via different logical network subdivisions called instances. Cooperation among multiple instances running multiple applications can help in mitigating congestion, which is the main factor degrading the Quality of Service in multi-application environments such as the Smart Grid.

Available solutions for cooperation between two or more RPL instances are centralized and reactive. The problem with the centralized approach is that it requires control messages to flow in both directions in the network (from the leaf nodes to the root and back), which increases overhead and energy consumption. Reactive solutions force cooperating nodes to send cooperation requests to the root and wait for cooperation confirmation messages. In heavy traffic scenarios, which increase the probability of path congestion, the reliability of such message exchanges cannot be guaranteed. Furthermore, these proposed centralized and reactive models do not address the issue of congestion and were not evaluated under heavy traffic.

We design a novel IPv6 routing protocol, based on RPL, for low-power and lossy IoT networks that support service differentiation. Our protocol enables distributed and proactive cooperation among its instances for congestion control. In our model, congestion detection is performed using a novel routing metric that locally estimates the path's congestion level under heavy and dynamic traffic. Our protocol utilizes the path diversity offered by other instances to mitigate congestion. It also employs a novel, distributed, proactive cooperation management scheme to tackle the issue of selfishness among cooperating nodes. We evaluate our protocol in a Smart Grid system where multiple alarm and monitoring applications coexist.

We also propose a framework for cooperation among instances that belong to different authorities. Our proposed framework targets scenarios where one instance's root is located close to some leaf nodes of another instance and vice versa. By exploiting the available backbone infrastructure, we design a scheme for cooperation encouragement between Smart City subsystems based on a virtual currency exchange model.

Contents

List of Figures

List of Tables

List of Abbreviations

MTU	Maximum Transmission Unit	15
NAN	Neighborhood Area Network	23
NFC	Near Field Communication	10
OF	Objective Function	33
OF0	Objective Function Zero	33
OFQS	Objective Function for Quality of Service	62
OHCA	Optimization-based Hybrid Congestion Alleviation	54
P2MP	Point-to-Multipoint	35
P2P	Point-to-Point	35
PCM	Path Congestion Metric	112
PC-RPL	Power-Controlled RPL	53
PIO	Prefix Information Option	42
pkt	Packet	48
PLC	Power Line Communication	12
PDR	Packet Delivery Ratio	51
QoS	Quality of Service	62
QU	Queue Utilization	49
QU-RPL	Queue Utilization based RPL	53
RDC	Radio Duty Cycle	53
RE	Remaining Energy	51
RFC	Request For Comment	16
RFID	Radio-Frequency Identification	10
ROLL	Routing Over Low-power and Lossy networks	18
RPL	IPv6 Routing Protocol for Low-power and Lossy Networks	1
RSU	Road-Side Unit	21

RSSI	Received Signal Strength Indicator	53
SAA	Stateless Address Auto-configuration	15
SG	Smart Grid	1
TCP	Transmission Control Protocol	13
TSCH	Time-Slotted Channel Hopping	18
UDP	User Datagram Protocol	17
V2I	Vehicle-to-Infrastructure	21
V2N	Vehicle-to-Network	21
V2P	Vehicle-to-Pedestrian	21
V2V	Vehicle-to-Vehicle	21
V2X	Vehicle-to-Everything	21
VANET	Vehicle Ad Hoc Network	28
WAN	Wide Area Network	23
WG	Working Group	16
Wi-SUN	Wireless Smart Ubiquitous Network	105
WSN	Wireless Sensor Network	12

1. Introduction

The Internet of Things (IoT) is a trending topic due to its promising aspects that aim to improve our lives. IoT has revolutionized monitoring and control systems in numerous fields like Smart Health, Smart City, and Smart Grid (SG) [1]. The main components of IoT are smart devices that perform the tasks of sensing, actuation, processing, and communication. Enabling the Internet Protocol version 6 (IPv6) [2] in such wireless sensors and actuators allows them to be seamlessly integrated with the Internet.

However, these intelligent devices often have limited memory, processing capabilities, and battery capacity. Furthermore, they communicate over unreliable links, forming networks referred to as Low-power and Lossy Networks (LLNs) [3]. Therefore, IPv6 support in LLNs imposes multiple challenges regarding overhead and communication due to their constrained and lossy nature.

The IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) [4] was designed to meet the IPv6 routing requirements of various IoT applications while still considering the limited resources available in LLNs.

1.1. Service differentiation in LLNs

The ITU-T defines Quality of Service (QoS) as “The collective effect of service performances which determine the degree of satisfaction of a user of the service” [5]. From the service provider's perspective, QoS means that the service offered to the customer fulfills specific quality measures such as packet loss, delay, and throughput. Measuring such parameters can provide an evaluation of the current QoS in the network.

Differentiated Services (DiffServ) [6] is a QoS model designed to give some types of traffic priority over others, making it possible to provide different Quality of Service levels to different traffic classes.

In RPL, DiffServ is achieved by logically partitioning the network into multiple “RPL instances” to enable forwarding different traffic classes via different routes. RPL allows an LLN to accommodate several independent routing sub-topologies concurrently, which can be considered subnetworks. Each of these is then configured to route traffic according to a specific[1] QoS requirement (e.g., low delay, low packet loss, etc.). These routing graphs, called RPL instances, are logical subdivisions of the same physical network. This design enables such a network to incorporate several applications with distinct purposes and various resource demands. Figure 1.1 shows an LLN with two instances for two applications, where the blue nodes belong to both instances.

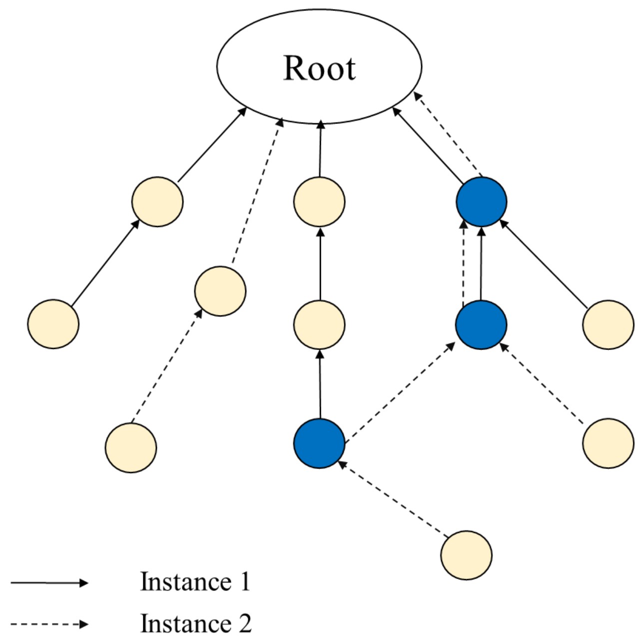


Figure 1.1: An LLN with two instances

The multi-instance feature of RPL is crucial for some IoT environments, such as the Smart Grid, where several applications with different -and sometimes conflicting- requirements coexist. An example of such a situation is displayed in Table 1.1, which lists reliability and delay constraints for Advanced Metering Infrastructure (AMI) and other SG applications. We give here a short

[1] Can also be configured according to a combination of multiple requirements.

overview of the applications included in Table 1.1, while an extended version of this table will be provided in the next Chapter:

- AMI communication networks are responsible for data collection from smart meters installed at consumers' premises. Normal AMI traffic corresponds to collecting smart meter readings of electricity consumption, which the utilities use for billing.
- Critical AMI traffic is present in the network when a critical SG application (e.g., power quality monitoring) requires additional smart meter parameters (e.g., electrical phase and frequency) to be transmitted with high reliability and low delay.
- Data used for monitoring the state of the electricity distribution network has stringent reliability and delay requirements, while network configuration traffic and firmware updates can tolerate longer delay periods.

Type of traffic	Maximum allowed delay	Reliability
Normal AMI traffic	< 5 min	> 98%
Critical AMI traffic	< 5 s	> 99.5%
Distribution network protection data	< 3 s	> 99.5%
Network configuration	hours/days	> 98%

Table 1.1: A sample of SG applications requirements [7]

1.2. Problem statement and Thesis goals

The main research problem this Thesis addresses is the absence of a solution for congestion control in multi-instance, RPL-based IoT networks that support service differentiation. Our primary research goals are to design a routing metric for congestion detection in such networks and to develop an inter-instance cooperation model for them. Our cooperation model aims to alleviate congestion by allowing nodes from a congested instance to send their packets via other, less-congested instances.

The RPL standard specification [4] neither addresses the issue of congestion nor specifies metrics for congestion control. That has motivated researchers to tackle the problem of congestion in RPL networks by designing improved RPL routing schemes that can detect and mitigate congestion. Yet, the majority of these schemes are designed for single-instance RPL networks only [8]. The few ones that target multi-instance RPL environments are based on multipath routing, where they consider other instances as alternative paths that could be used in case of congestion [9]. In this multipath approach, a node selects the best route by estimating and comparing congestion levels and path costs of all instances. To enable such calculations, it is assumed that all instances use the same configuration (routing metric, routing policy, etc.). In other words, these proposed multi-instance, multipath RPL routing models work only if all instances are identical.

We can formulate this research gap as follows:

The problem with the proposed solutions for congestion control in multi-instance RPL networks is that they require all instances to be identical. This prevents establishing service differentiation, where each instance should be configured differently to meet certain QoS requirements.

We aim to bridge this gap by:

Designing an interoperable routing metric for congestion detection in multi-instance RPL-based IoT networks, where each instance has a distinct routing configuration.

The next step after congestion detection is congestion mitigation. The solution we choose for that is utilizing path diversity; since alternative paths to the root via other RPL instances may exist. Some of the aforementioned enhanced, multi-instance, multipath RPL routing models proposed by the RPL research community already exploit routes via other instances to reduce congestion. However, they assume that a congested node from one instance can send its packets via other instances unhindered [9]. In RPL networks that support service differentiation, other instances have different settings and are not merely alternative paths to the root. This leads to the second research problem this thesis addresses:

Congestion alleviation schemes for multi-instance RPL networks do not implement any instance cooperation mechanisms. They rely on the assumption that a node can send its packets via other instances without restrictions, which raises issues of nodes' selfishness and load imbalance in the network. Furthermore, such an assumption can not be made when the instances in the network belong to different authorities.

Our approach to tackling this problem is:

Developing a cooperation procedure for RPL instances that defines how congested nodes can forward their packets to nodes from other instances. Additionally, designing a scheme to promote cooperation among instances that belong to different authorities. Moreover, defining the cooperation boundaries in these two cases.

Next, we investigate whether the two above-mentioned cooperation models should be designed by us, or if they already exist in any other RPL research domain.

Even though RPL has many advantages, it still suffers from multiple shortcomings. Mainly, RPL suffers from the problem of under-specification since many of its features were introduced only briefly. Notably, the RPL technical specification mentions the possibility of using multiple instances for traffic differentiation [4]. Still, it does not provide any mechanism for instance management or cooperation among instances.

There is a limited number of research papers exploring the area of multi-instance RPL. Studying such a scenario is challenging due to the small memory size of sensors, which restricts the options for investigating this research domain. Available solutions for cooperation among RPL instances use techniques that are both reactive and centralized [10]. Centralized schemes are questionable for two main reasons:

- IoT is envisioned to have fully autonomous devices [11]. That was the primary motivation for designing many IoT standards, including RPL. A paradigm that relies on a central authority for making decisions goes against the direction of these standards, which anticipate a distributed design of LLNs as an essential element of IoT.
- They require RPL to support two-way traffic flows between the nodes and the root to establish cooperation. RPL is designed to build routes for these two traffic directions separately; due to the high overhead incurred when using them together. Data collection applications, where data flows from the nodes to the root, form the majority of RPL applications [4, 12]. Therefore, building routes in the opposite direction from the root to the leaf nodes (called downward routing) should be avoided since it is unnecessary in most cases. Yet, these centric cooperation models require building and maintaining routes downwards despite the overhead this process causes.

The reactive aspect of these centralized RPL cooperation schemes comes from relying on the root for deciding how and when instances collaborate. When a node receives a cooperation request from a node from another instance, it forwards it to the root and waits for the root's approval of the cooperation. This request-reply process is prone to failure due to the lossy nature of the LLN links. Besides, if congestion occurs in the network, there is an increased probability that cooperation requests/replies get lost on the path to/from the root. Furthermore, in large-scale networks, this solution does not seem feasible since cooperation requests from nodes located far away from the root might not even reach it.

We can summarize the shortcomings of the research publications regarding cooperation among multiple RPL instances as:

Their reactive and centric design makes them incur noticeable overhead due to the extra RPL mechanisms (enabling downward routing) and node-root communications (cooperation requests and replies) they require. Their request-reply cooperation process is vulnerable to congestion and not scalable. Moreover, they were neither studied under congestion conditions nor designed for congestion alleviation.

The Thesis objective in this regard is:

Designing a novel inter-instance cooperation scheme for congestion mitigation in RPL networks. The design is planned to be distributed, where cooperation decisions are made locally at the nodes, to reduce overhead and enable proactive decision-making.

Essentially, we aim to design an IPv6 routing protocol with distributed mechanisms of instance cooperation for congestion control in LLNs. The design is based on RPL and intended to be compatible with its objectives defined by [13], which states that for a routing solution in LLNs to be useful, the routing protocol ought to be "energy-efficient, scalable, and autonomous.".

1.3. Research questions

RPL's multi-instance feature provides a solution for meeting different QoS requirements in LLNs. Despite the negative impacts it has on QoS, the issue of congestion has not yet been addressed in multi-instance, RPL-based IoT networks. We believe that in such networks, cooperation among instances can be utilized for congestion control. Therefore, the main question this Thesis raises is:

How to exploit the path diversity provided by RPL's multi-instance structure for congestion control in LLNs?

We believe that a distributed cooperation architecture is essential to answer this question. IoT systems are expected to be composed of a massive number of nodes deployed to cover vast geographical areas. In such environments, nodes should be as autonomous as possible, i.e., they should make routing and cooperation decisions without referring to a central gateway or authority. Such a distributed cooperation architecture has its advantages in congestion control, but it also raises multiple issues. One of which is that the root can regulate cooperation among RPL instances, as proposed by [10]. However, an LLN leaf node has limited information about its network compared to the root to perform such a task. Without a proper mechanism for path congestion estimation, an LLN leaf node accepting cooperation requests and forwarding packets from other instances may cause congestion in its own instance. Furthermore, the absence of a distributed procedure for managing inter-instance cooperation makes LLN leaf nodes from different instances face fairness challenges when they forward packets for each other.

Based on these observations, we break down the main question of this Thesis into the following Research Questions (RQs), which provide the basis for our work:

RQ1. How to locally detect the path congestion level under heavy and dynamic traffic conditions in LLNs?

RQ2. How can nodes from different RPL instances cooperate to mitigate congestion without referring to the root?

RQ3. How can RPL nodes locally manage and maintain cooperation against the issues of unfairness and selfishness?

RQ4. How can RPL networks handle asymmetric cooperation?

1.4. Motivation

Congestion leads to excessive energy consumption, packet loss, and many other factors that degrade the performance of LLNs and worsen their QoS. It is quite common that LLNs experience congestion when their nodes send data packets at rates of 30 packets per minute or higher, which is referred to as heavy traffic [9].

Dynamic traffic corresponds to alarm signals, and it also causes congestion. It appears sporadically in the network when a node, at random time slots, generates data bursts at high rates for a limited duration. An example of such a dynamic traffic situation is when nodes generate data streams at a rate of 1 packet per second for one minute, then generate another similar data stream after a random time interval, and so on [14].

In large-scale networks, congestion can take place even at low traffic rates. Kim *et al.* compared a network composed of 30 nodes that send data at a rate of 30 packets per minute with a network with 5000 nodes that send one packet every 5.5 minutes [15, 16]. The total number of packets generated per hour in both networks is the same. According to Kim *et al.*, in large-scale networks such as SG networks, the traffic is concentrated at nodes that are close to the root, which are referred to as hotspot nodes. Therefore, even in light traffic scenarios, these hotspot nodes have to deal with heavy traffic and relay packets at high rates.

In many IoT systems like the Smart Gird, a single IoT network is expected to run multiple applications simultaneously. In such environments, the support of multiple routing instances is crucial for achieving service differentiation. The majority of research papers in the domain of multi-instance RPL assume that instances in the network are isolated; thus, they do not consider cooperation among them. Therefore, we are motivated to explore the effects of such cooperation on QoS and congestion in RPL-based IoT networks.

In an IoT ecosystem such as the Smart City, multiple IoT networks from multiple authorities are expected to be co-located and overlap. Cooperation among multiple instances belonging to different authorities, where one root could be geographically closer to some leaf nodes of another root and vice versa, could provide a promising opportunity to enhance QoS for nodes located far away from their root.

In a nutshell, we believe that inter-instance cooperation could provide an opportunity to improve QoS in IoT networks in multiple scenarios, especially in large-scale LLNs, LLNs operating under heavy and dynamic traffic conditions, and overlapping LLNs that belong to different authorities.

Our research domain is an intersection between two major RPL research domains: congestion control and cooperation among multiple instances. Our work is motivated by the need to investigate this unexplored intersection domain.

1.5. Contributions and Thesis outline

This thesis is the first to explore the above-mentioned intersection research domain between congestion control and multi-instance cooperation in RPL networks. We design a novel IPv6 routing protocol for IoT networks based on RPL. Our proposed protocol introduces new mechanisms for path congestion estimation and distributed cooperation management. We also develop a virtual currency cooperation incentivization and regulation framework for LLNs with instances that belong to different authorities.

Chapter 2 gives an overview of IoT, LLNs, Smart City, and Smart Grid. An extensive review of RPL and its functionalities is also provided there. Chapter 3 comprehensively reviews the research on RPL's congestion control and RPL with multiple instances. It also provides a classification and a comparative analysis of the RPL enhancements proposed by the research community in these two domains.

RQ1, RQ2, and RQ3 are addressed in Chapter 4 through DMC-RPL, which will be evaluated in Chapter 5. Chapter 6 explores schemes for inter-domain cooperation encouragement via virtual currency systems. RQ3 and RQ4 will be tackled by a framework we introduce in Chapter 6. In Chapter 7, we give the conclusion to this thesis and discuss its potential future research directions.

2. Background

In this Chapter, we provide an introduction to the Internet of Things and an overview of its standards for integrating LLNs into the Internet. We also review the Smart City ecosystem and its applications. Next, we extensively investigate the Smart Grid system, which is a Smart City sub-system. Along with the main Smart Grid applications, we illustrate the typical paradigm of the Smart Grid communication networks and their communication prerequisites. After defining these requirements, we detail the components and mechanisms of RPL that make it suitable for such networks. Finally, we outline the scope of this thesis in light of the information introduced in this Chapter.

2.1. The Internet of Things

IoT is a vision for the future Internet where real-world objects are incorporated into the Internet by giving them unique identities and Internet connectivity. Items tagged with electronic bar codes such as Radio-Frequency Identification (RFID) or Near Field Communication (NFC) [17] can be easily scanned and identified by intelligent devices with compatible readers. Such devices play an essential role in integrating physical objects into the Internet, and in monitoring and reporting their state, thereby turning them into "Smart Objects".

Equipping elements of our surrounding environments with smart devices, such as intelligent actuators and sensors, and providing them with Internet connection and distinct Internet Protocol (IP) addresses, enables the Internet of Things to create a smooth connection between the real world and the cyberspace, thus providing a digital interface to our physical world.

The term "Things" in IoT is an umbrella for a multitude of intelligent devices that can interact with their environment. Anything that can gather and transmit information to the cloud qualifies as a *thing* in the context of IoT [18]. Thus anything, from a sensor to a sophisticated machine, can be a *thing* or an IoT device. Examples of these are personal devices (smartphones and smartwatches) and embedded ones (sensors and actuators).

In 2016, there were 6.4 million IoT devices around the world. By 2030, this number is anticipated to rise to 20–50 billion [19]. This translates to an average of 5–6 IoT devices per person. IoT is being integrated into all contemporary and rising engineering, technology, manufacturing, and development domains, with technical advances like 5G accelerating this integration [20]. We are surrounded by the IoT; it is prevalent in every aspect of our lives, including our homes, appliances, healthcare facilities, infrastructure, businesses, vehicles, and industries [21].

The following are the main components of the IoT Framework [19]:

- The Thing: The IoT device for data collection and transmission.
- The data analytics algorithm: For data processing and mapping.
- The IoT client: An application for viewing the processed information.

There are many possible ways to classify IoT systems. Figure 2.1 portrays a classification of IoT systems in various application and networking categories.

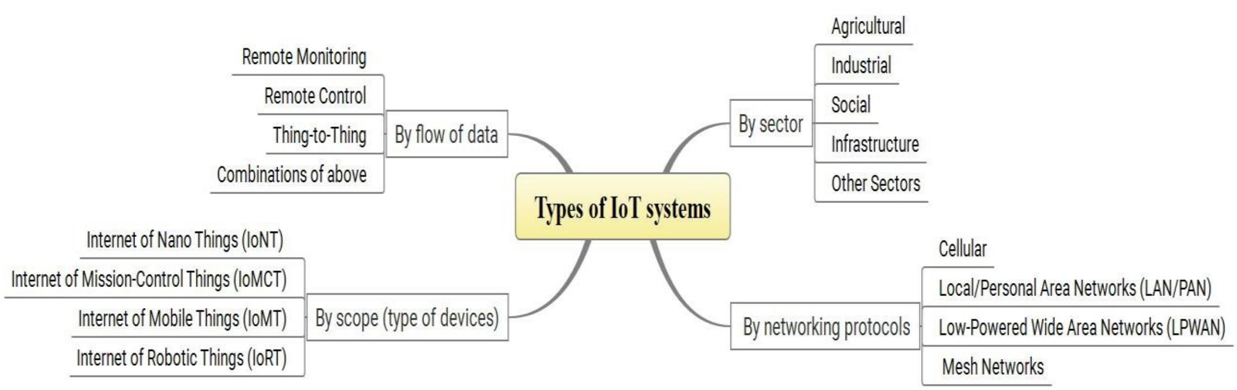


Figure 2.1: Types of IoT systems

2.2. Low-power and Lossy Networks

Constrained devices such as sensors and actuators are vital components of IoT. They are characterized by having limited resources including limited processing capabilities, small memory (RAM: 10-50 KB, ROM: 100-250 KB), and short battery lifetime [22]. These devices are typically deployed in unstable environments with components that affect their radio communication. Absorption and reflection can cause a reliable wireless link to turn unreliable for a limited period and later become usable again, hence the term "lossy" is used to label such links. Communication over wireless lossy links is prone to transmission errors and packet loss.

Since constrained smart devices typically communicate over low-bit-rate links that are lossy, they tend to form highly dynamic topologies. The IEEE 802.15.4 wireless links [23] are a well-known example of these low-rate lossy links. IEEE 802.15.4 is a standard that defines the physical layer and the data link layer for wireless personal area networks. It is characterized by low-power, short-range, and low-data-rate wireless transmissions. IEEE 802.15.4 provides rates up to 250 kbps using a 2.4 GHz band, and a communication range up to 100 m.

Many terminologies have been proposed to refer to networks of constrained Smart devices. The umbrella term used for them is “Low-power and Lossy Networks” [3]. The term LLNs is intended to be a generic term for networks of constrained nodes regardless of which link layer technology they use. LLNs communicate over a variety of fairly-unstable low-speed links such as IEEE 802.15.4 [23], low-power Wi-Fi [24], and Power Line Communication (PLC) [25]. LLNs have many subtypes. We list the most common ones here:

- **Low-power Wireless Personal Area Networks (LoWPANs)** [26]: These are LLNs that communicate over IEEE 802.15.4 wireless links. Wireless Sensor Networks (WSNs) are a well-known example of LoWPANs.
- **IP-based LoWPANs:** These are LoWPANs that support IPv6. These networks comply with the IETF's IPv6 over LoWPANs (6LoWPAN) standard [27], which enables them to support IPv6 addressing and transmit IPv6 packets over IEEE 802.15.4 links, i.e., become fully integrated into the Internet. The term 6LoWPAN denotes the standard itself, while the term 6LoWPANs refers to the networks that comply with that standard, which are also sometimes called IP-based WSNs.

LLNs are considered key components of IoT. Enabling IPv6 routing in LLNs using RPL serves as a primary step toward connecting them to the Internet.

2.3. Integrating constrained devices into the Internet

Constrained devices can perform data-gathering tasks in various environments, such as buildings, homes, and cities. The collected data is then sent in a hop-by-hop fashion to a gateway and then a central station. There are two different schemes for providing Internet connectivity to WSNs: the Proxy-based and the sensor IP-stack.

2.3.1. The proxy-based approach

This is the traditional solution for connecting WSNs to the Internet as portrayed in Figure 2.2. It was imposed by the fact that vendors had to implement their own proprietary protocols in their sensor networks; due to the lack of a standardized protocol suit for constrained devices [28]. Gateways that are vendor-specific are used to perform the protocol bridging required to connect the WSNs to the Internet. Therefore, in this paradigm, Internet users cannot directly obtain data from sensors. Instead, users have to send data acquisition requests to the gateway, which will then relay these requests after translating the standard Internet protocols into vendor-dedicated WSN protocols. In a similar way, the gateway relays the sensor replies in the other direction (to the end users) after performing the required translations.

This approach has the advantage that it does not require sensors to run the Transmission Control Protocol (TCP) / IP stack, which imposes a lot of overhead. However, it does not provide enough flexibility since the sensor-user communication is governed by the gateway and its vendor design. Furthermore, since establishing direct end-to-end connections is not possible, using real-time applications is not feasible.

Therefore, it is quite evident that this proxy-based structure restricts integrating sensors into the Internet seamlessly and, thus, was not favored as an IoT scheme [11].

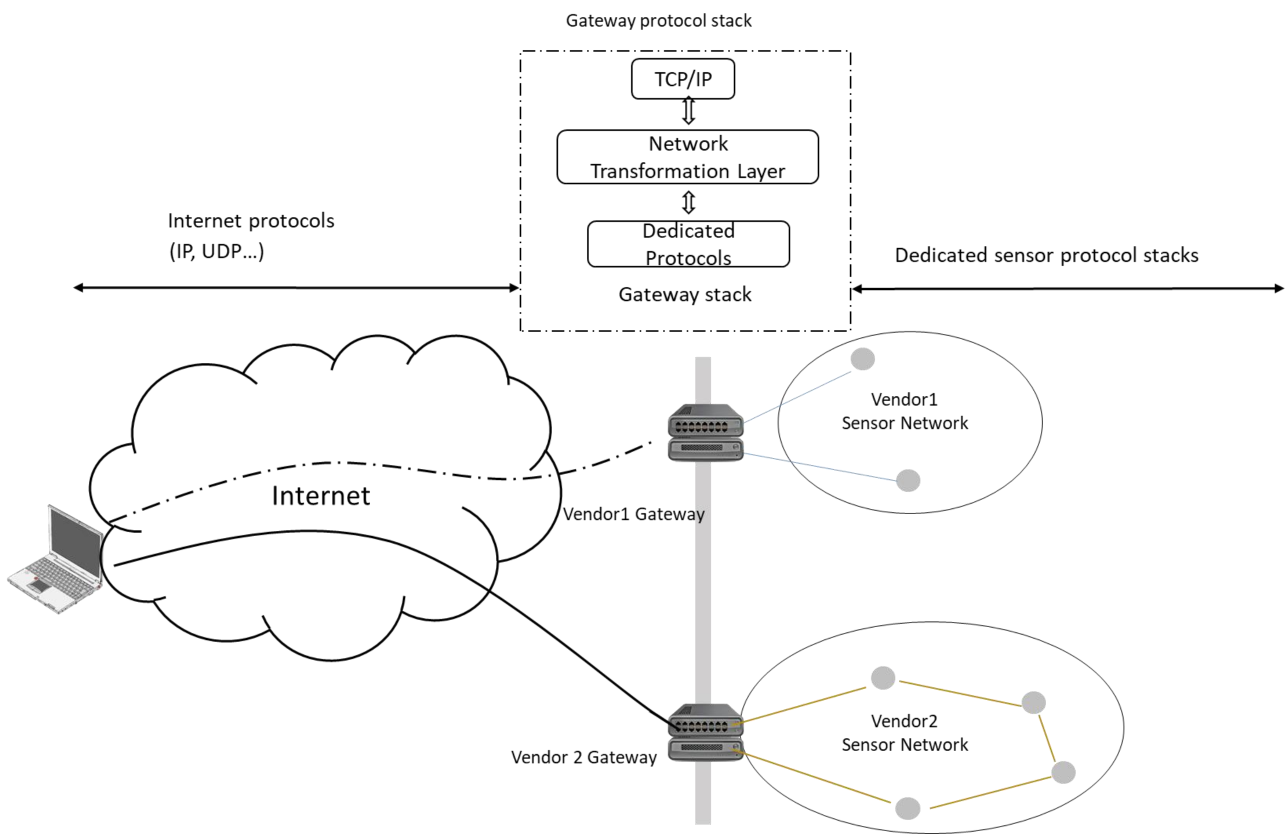


Figure 2.2: The proxy-based approach

2.3.2. The sensor IP-stack approach

In this scheme, sensors have IP addresses and can be directly accessed via the Internet over a router/sink node, which replaces the gateway in the proxy-based scheme. The sink node functions only as a router between the WSN and the Internet as shown in Figure 2.3. This implies that it operates at the network layer level and does not use upper-layer protocols to establish such connectivity.

There are many benefits to making sensors support Internet protocols [26]:

- IP-based solutions are renowned. Their efficiency and scalability have been studied for years.
- IP networks already exist and could be used. Connecting IP-based devices is straightforward and does not need any protocol translation.

- It enables connecting sensors from different manufacturers within the same WSN for better interoperability.
- It enables WSNs to use existing management and diagnostics tools developed for IP.

IPv6 [2] was chosen as the Internet protocol for constrained devices instead of Internet Protocol version 4 (IPv4) [29] because it has a much larger address space. IPv6 also has many useful mechanisms like the Stateless Address Auto-configuration (SAA) [30]. A node can use SAA to autonomously create IPv6 addresses from its local or global address prefixes and its Interface Identifier (IID) which is derived from its Media Access Control (MAC) address.

Despite its advantages, the support of IPv6 in WSNs imposes numerous challenges:

- Frame size: IPv6 requires links to support a Maximum Transmission Unit (MTU) of at least 1280 bytes [2]. This is very large for the WSN links that usually have frame sizes in the order of 40-200 bytes [27].
- Header overhead: The size of the IPv6 header is relatively large compared to the maximum packet size supported by the WSN links. For example, the largest possible frame size for IEEE 802.15.4 frames is 127 bytes, with 25 bytes of the frame reserved for the MAC header [23]. That leaves 102 Bytes for the network layer. Of the 102 bytes available packet size, the IPv6 header occupies 40 bytes [27], i.e., around 40%.
- Limited bandwidth and energy: Sensor nodes communicate over links with low data rates (250 kbps when using the IEEE 802.15.4 links) and are expected to sleep most of the time to conserve energy. While essentially, IP assumes that the nodes should always be connected [11].

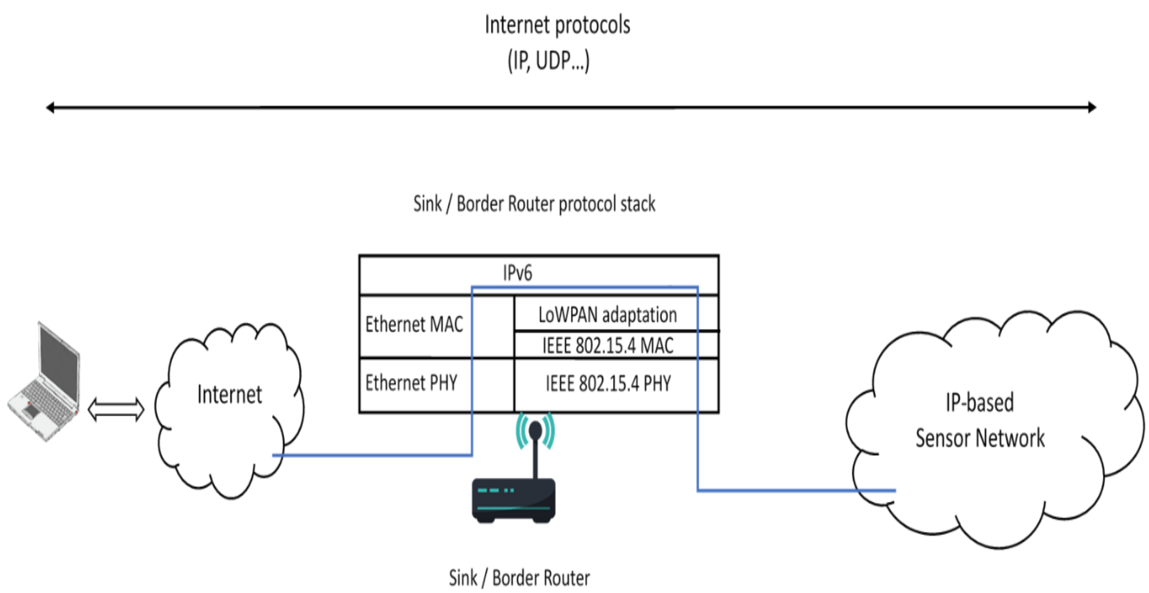


Figure 2.3: The sensor IP-stack approach

2.4. IETF Standards for IoT

The Internet Engineering Task Force (IETF) has been involved in standardizing Internet protocols such as IPv4 and IPv6. On top of that, it has defined various application, routing, and security protocols such as OSPF [31], BGP [32], HTTP [33], SMTP [34], and IPsec [35].

IETF specifications are brought forth by IETF Working Groups (WGs), which develop protocols and devise Request For Comments (RFCs). An RFC is a publication that defines methodologies, application procedures, or concepts regarding Internet protocols, applications, systems, or architecture. Creating IoT standards is performed by various IETF Working Groups within four major domains: connectivity, routing, application, and security. We focus on explaining the first two since they are relevant to this thesis.

2.4.1. IoT standards in the connectivity domain

Since several communication technologies for constrained devices exist, many RFCs were designed to meet their diverse requirements.

2.4.1.1. 6LoWPAN / 6lo

The 6LoWPAN Working Group [36] aims to enable the transmission of IPv6 packets over IEEE 802.15.4 wireless links.

To tackle the challenges mentioned in the previous Section regarding enabling IPv6 at constrained devices, 6LoWPAN introduces an adaptation layer between the network and the data link layers at these devices. This new layer performs two essential operations:

- IPv6 header compression and decompression: To reduce the overhead caused by its relatively large size.
- IPv6 packet fragmentation and reassembly: If an IPv6 packet does not fit within one frame.

The work of the 6LoWPAN group concluded in 2014 after creating several RFCs:

- RFC 4944 [27]: Specifies problems and goals of IPv6 transmission over IEEE 802.15.4 networks. It defines the 6LoWPAN adaptation layer and introduces a method for IPv6 packet fragmentation and reassembly. It also includes formats for stateless IPv6 Header Compression (HC1) and User Datagram Protocol (UDP) header compression.
- RFC 6282 [37]: Proposes the IPv6 Header Compression (IPHC), a stateful header compression scheme that uses "shared contexts" to enable compressing arbitrary prefixes. A context is an IPv6 address or prefix that is distributed by the 6LoWPAN root and shared by all the nodes. Up to 16 address contexts could exist in a 6LoWPAN network. A 4-bit field is used for context encoding, which allows the compression of a particular context from 32/64 bits to 4 bits only. Mechanisms for compressing IPv6 multicast addresses and the UDP header are also defined in this RFC.
- RFC 6568 [38]: Investigates the design space and use cases of 6LoWPANs.
- RFC 6606 [39]: Indicates the 6LoWPAN routing requirements and defines two routing schemes:
  - Route-over: IP routing at the network layer.
  - Mesh-under: Routing that takes place at the link layer using its IEEE 802.15.4 addresses.
- RFC 6775 [40]: Proposes an optimized design of the IPv6 Neighbor Discovery (IPv6 ND) protocol (RFC 4861 [41]) tailored to 6LoWPANs.

The work of the IPv6 over Networks of Resource-constrained Nodes (6lo) Working Group [42] continues where the 6LoWPAN WG has stopped. While 6LoWPAN focuses on enabling the transmission of IPv6 packets over IEEE 802.15.4 links, 6lo expands its scope by including other communication links for constrained-node networks such as:

- RFC 7668 [43]: IPv6 over Bluetooth Low Energy (BLE).
- draft-ietf-6lo-nfc-17 [44]: IPv6 over NFC.
- draft-ietf-6lo-plc-11 [45]: IPv6 over PLC.

6LoWPAN and 6lo standards substantially expand the range of IPv6-compatible technologies.

2.4.1.2. 6TiSCH

IPv6 over the Time-slotted Channel Hopping (6TiSCH) is an IETF Working Group that researches enabling IPv6 in LLNs that use the Time-Slotted Channel Hopping (TSCH) mode of IEEE 802.14.4e [46].

2.4.1.3. LPWAN

The IPv6 over Low Power Wide-Area Networks (LPWAN) group focuses on integrating LPWANs and IPv6 [47]. Devices of these networks are characterized by large coverage areas, low bandwidth, and long battery lifetime.

2.4.2. IoT standards in the routing domain

The Routing Over Low-power and Lossy networks (ROLL) Working Group [48] addresses challenges in LLN routing. Initially, ROLL studied various LLN application domains, and relevant routing requirements for Industry, Smart Home, Building Automation, and Urban scenarios were set in RFC 5673 [49], RFC 5826 [50], RFC 5867 [51], and RFC 5548 [13] respectively.

It was found that no available routing protocol could fulfill these requirements, and as a result, ROLL designed RPL. The core mechanisms of RPL were specified in RFC 6550 [4], while other RPL features were defined in separate RFCs. The reason for this separation is to give more flexibility in RPL implementations and future developments. The primary RFCs by ROLL are listed below:

- RFC 6550: Defines RPL and its main operations.
- RFC 6551 [52]: Routing metrics that can be used with RPL.
- RFC 6552 [53]: A generic method for for parent selection in RPL.
- RFC 6719 [54]: A method for parent selection in RPL that uses hysteresis.
- RFC 6206 [55]: The algorithm used to control the transmission rate of RPL control messages.
- RFC 7733 [56]: Applicability of RPL in Home and Building Automation.
- RFC 8036 [57]: Applicability of RPL in AMI networks.

These RFCs will be explained in detail later in Section 2.8.

2.5. The Smart City IoT ecosystem

Urban regions face numerous challenges that involve traffic congestion, waste management, air pollution, aging infrastructure, and many more [58]. Large groups of people living in a limited space tend to form unorganized structures with conflicting goals and values, which raises multiple social and organizational challenges in cities.

In the face of these enormous challenges, many cities explore innovative solutions to foster higher living standards and sustainability in the context of rapid urban growth. The Smart City model [59] is thus crucial for migrating into better city planning and governance, and consequently, a sustainable pattern of urban development. The term "Smart City" is being used increasingly to characterize cities pursuing these efforts.

There has been no standard definition of the term "Smart City", even though it is used with increasing frequency. There are several definitions of that term, with some using the terms "Digital City" and "Intelligent City" as synonyms of it.

We refer to the definition proposed by Harrison *et al.,* which describes a Smart City as "A city connecting the physical infrastructure, the IT infrastructure, the social infrastructure, and the business infrastructure to leverage the collective intelligence of the city" [60].

A fundamental component of the Smart City paradigm is developing efficient communication technologies across numerous urban networking services [61]. One of the most prominent solutions for achieving that is the Internet of Things. The IoT architecture incorporates sensors into everyday items. It then connects them to the Internet to interact and exchange information with people to provide them with a variety of services [62].

2.5.1. Smart City subsystems

The Smart City is an environment where many systems coexist and interact with one another, which enables designing novel inter-system applications. An example of this is intersecting the Smart Transportation system with the Smart Grid system to develop an application for collecting data from household smart meters by using public transportation buses [63].

Figure 2.4 shows a sample of Smart City systems. A more detailed list is provided in [64]. We overview two examples of these systems in this Section. Even though the Smart Grid is a Smart City subsystem, we do not list it here to extensively examine it in the next Section.

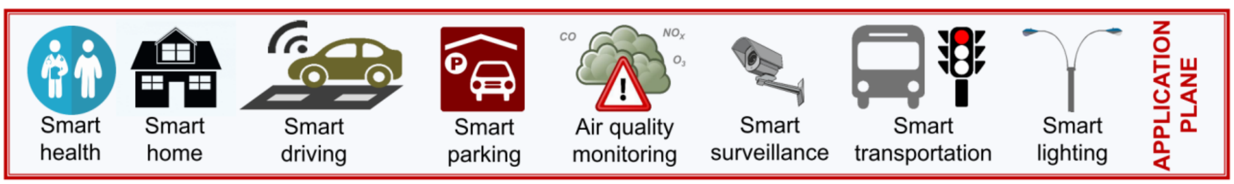


Figure 2.4: Examples of Smart City subsystems [65]

2.5.1.1. Smart Waste Management

IoT-based monitoring systems have the potential to improve solid waste management considerably. Scheduling waste collection in short intervals can cause a waste of fuel and human resources as garbage bins may not be full. On the other hand, longer intervals can lead bins to overflow and cause pollution. Resources could be used more efficiently if ultrasonic sensors were used to report the level of trash in the bins as in the systems proposed by Ramson *et al.* [66, 67]. In these solutions, sensors transmit their bin monitoring data to a central station over the Internet, where an application could optimize waste collection schedules.

2.5.1.2. Intelligent Transportation Systems

Intelligent Transportation Systems (ITS) form an important part of the Smart City infrastructure. IoT can also be used for monitoring traffic in order to minimize economic loss and environmental pollution caused by traffic congestion [68].

An ITS aims to improve logistics and commuting by utilizing four elements: Vehicles, Road Side Units (RSUs), central stations, and security subsystems [69].

The Internet of Things has enabled new domains for vehicle communications. Depending on the parties exchanging data, these types of vehicle networks can be identified [70]:

- Vehicle-to-Vehicle (V2V): Moving vehicles connecting directly to one another without relying on the support of a fixed infrastructure.
- Vehicle-to-Infrastructure (V2I) or Vehicle-to-Pedestrian (V2P): Vehicles communicating with roadside infrastructure equipped with smart devices (RSUs), or with pedestrians.
- Vehicle-to-Network (V2N): Vehicles communicating with IT networks or the Internet.

Vehicle-to-Everything (V2X) is the umbrella term that incorporates all the above-mentioned networks. IoT protocols and infrastructure are the main contributors to the vision of V2X. A network of sensors monitoring tire pressure, road conditions, and other vehicle statuses form what is known as intra-vehicle WSNs [71]. While Inter-vehicular communication could be used to avoid collisions or exchange information on lane changes [72].

2.6. The Smart Grid IoT system

Conventional electrical grids (Figure 2.5) are facing increasing electricity demand while having limited power resources as they mostly rely on non-renewable energy sources, whose greenhouse gas emissions contribute to climate change. Furthermore, their energy supply and demand are out of balance due to the overproduction of energy to prevent power outages. Lastly, traditional electrical grids lack sufficient diagnostic and monitoring tools, making it difficult for network operators to control the system remotely or monitor it in real time. The latter is primarily caused by the limited availability of suitable sensing technologies and communication networks able to transmit information from the electrical grid to the operator in a timely manner [73].

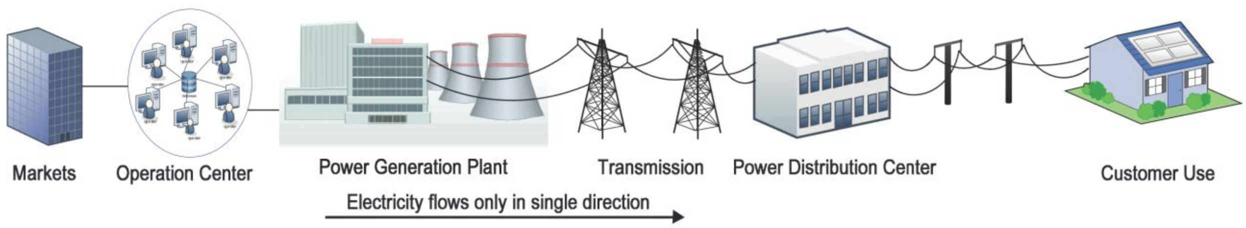


Figure 2.5: The traditional electric power system [74]

The smart electrical grid [75], also simply called "Smart Grid", is an evolution of the power grid that is going to change how electricity is managed, produced, and consumed. The Smart Grid is designed to update the current power grid by incorporating novel technologies that will enhance its reliability, security, efficiency, and scalability.

Smart Grids employ modern Information and Communication Technologies (ICTs) to administer electricity generation and distribution and manage renewable energy sources, which are often time-variable and unpredictable [76]. At the same time, Smart Grids also allow two-way communication between consumers and power providers to exchange information about energy consumption and pricing.

2.6.1. Smart Grid communication networks

The fundamental elements of the SG infrastructure are the networking technologies utilized for data transmission in the SG. Data exchange in the SG is crucial; it enables grid operators to gain more control and insight into it and consumers to benefit from many of its services. The architecture of a typical Smart Grid, including its domains and communication networks, is shown in Figure 2.6.

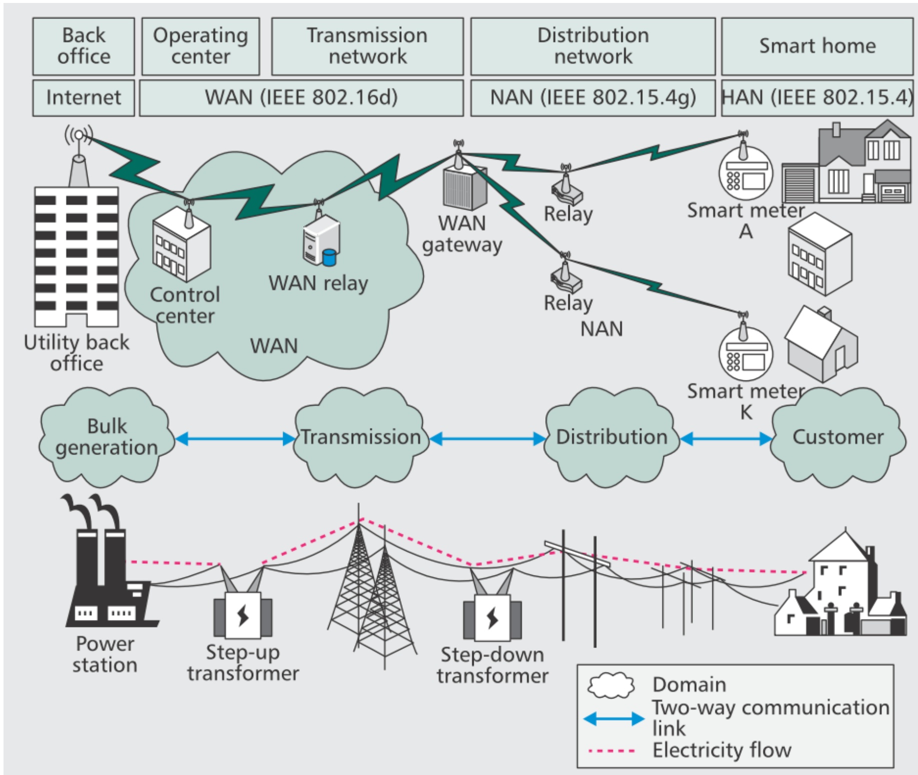


Figure 2.6: Smart Grid domains and communication architecture [77]

The structure of an SG communication network includes a Home Area Network (HAN) for data collection from various household devices, a Neighborhood Area Network (NAN) that connects HANs from the same neighborhood with a neighborhood access point, and a Wide Area Network (WAN), also known as the backhaul, which serves to connect the grid networks with the utility control center [78]. The following is an overview of these three network types:

- HAN: This is the lowest layer of the SG paradigm. Each HAN has a controller that collects electrical measurements and other data from several heterogeneous electric house appliances, such as smart meters and Home Energy Management Systems (HEMS). The collected data is then sent to the utility control center to monitor and analyze the overall energy consumption per household. Along with HANs, the customer premises domain of the SG also includes Industrial Area Networks (IANs) and Building Area Networks (BANs), which are deployed in industry and building automation, respectively. HAN/BAN/IAN components are located within the same property, and their applications do not require high data rates. That's why wireless communications are usually favored in these networks due to their simplicity of implementation and low cost. Here, a low data rate of around 100 kbps and a short communication range of up to 100 meters are sufficient. This explains why WSNs are widely used in HAN/BAN/IAN deployments [79].

- NAN: It bridges the utility backbone and the customer premises networks. For remote metering applications used by AMI, a NAN creates links between smart meters on the one hand and local access points on the other. Field Area Networks (FANs) are similar to NANs, but instead of collecting data from consumers, they are used to gather data from power lines, mobile workers, towers, and other distribution network components, all for the purpose of monitoring the power grid [75]. NANs/FANs use a variety of communication and networking technologies, including WSNs.
- WAN: Connects NANs with the core network and transfers their data to the private networks of the service providers. The technology utilized in WAN is typically optical/wired, and WAN routing is performed via a public network like the Internet.

The Smart Grid communication networks can be represented using a hierarchical multi-layer paradigm based on their communication ranges and data rates, as shown in Figure 2.7 below.

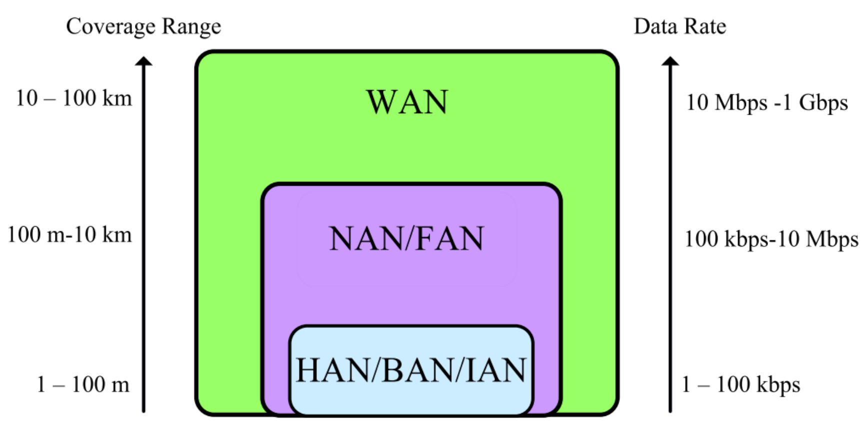


Figure 2.7: Data rates and communication ranges of the SG communication networks [79]

The SG management system can benefit from wireless sensors deployed at strategic locations for remote control and monitoring [80]. WSNs are not only more cost-efficient but can also be deployed more rapidly to cover larger areas than traditional wireless communication technologies and are therefore considered highly suitable for the facilitation of Smart Grid management [81]. Hence, WSNs are commonly deployed in the HAN and NAN segments of the Smart Grid.

2.6.2. Smart Grid applications

As an evolved vision of the power grid, the Smart Grid supports numerous enhancements, services, and applications. We investigate some of them in this Section.

2.6.2.1. Advanced Metering Infrastructure

AMI is a promising Smart Grid application designed for more efficient electricity consumption by clients [82]. AMI is essential for the operation of Smart Grids as it can provide various information on load, demand, and voltage profiles. Moreover, it can record consumed electricity data used for billing, analyzing usage durations, and power quality monitoring.

Utilizing AMI technology can be valuable for both consumers and suppliers of energy. On the one hand, utilities can reduce labor costs for metering support services like manual and on-demand meter reading, power restoration support, and field trips.

On the other hand, utilities can provide real-time information on price changes through AMI's two-way communication infrastructure, thus allowing consumers to adjust their energy usage accordingly. This communication system will therefore incentivize consumers to reduce their demand during peak times, when energy is more expensive, thus enabling utilities to implement Demand-Side Management (DSM) effectively. DSM is a set of tools to utilize energy consumption by load management and monitoring. It can regulate electricity distribution to mitigate peaks in electricity demand by consumers, thereby enhancing the grid's stability and reducing its operational costs [83].

Traditionally, households receive power, gas, and water bills per post on a regular basis. Smart metering using IoT technologies could collect, process, and share real-time information on electricity, water, and gas usage with a utility's central metering station via the Internet.

Information analyzed at these stations is then shared with the consumers. This system enables service providers to monitor resource demand at high time resolution while users can track their utility consumption daily or hourly. Internet-based services could further be deployed in the system to enable electronic payment and billing [84].

The communication of prices can be channeled through HEMS gateways or transmitted directly to appliances. These can then use the consumption data to optimize electricity consumption based on the client's wishes and needs. An effective two-way communication system deployed by utilities and consumers is crucial for realizing such complex information exchange and management processes.

Usually, an AMI network is built from a number of smaller networks connecting smart meters. As Figure 2.8 displays, house-held smart meters located in the same neighborhood transmit their data to a central unit called a data collector or concentrator. These smart meters and their concentrator compose a Neighborhood Area Network.

From the other side, each concentrator is connected to the AMI Wide Area Network, which relays the AMI network data to a central control unit.

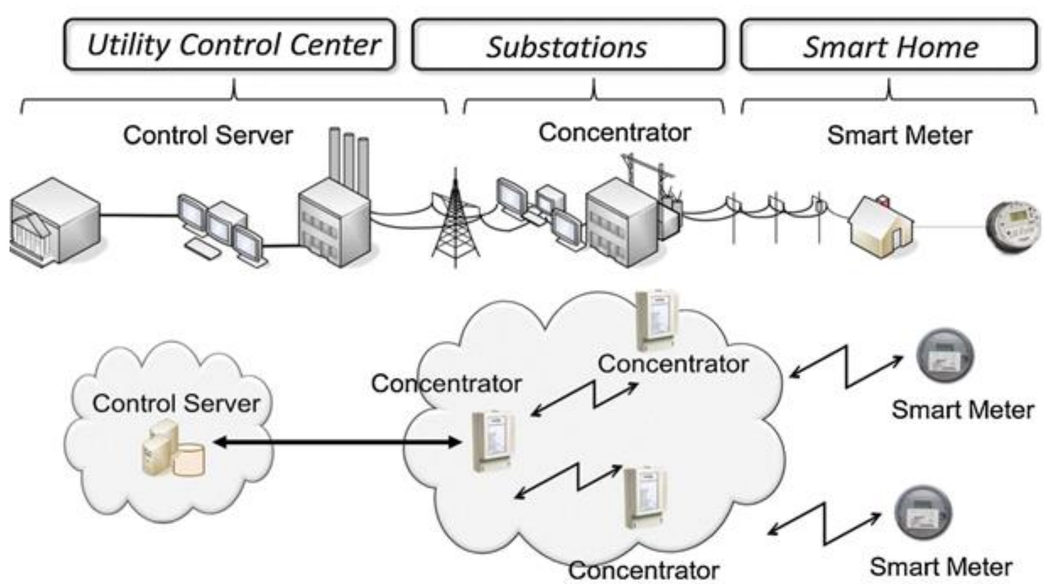


Figure 2.8: AMI in the Smart Grid [85]

2.6.2.2. Distribution Automation

Distribution Automation (DA) provides the capacity to make decisions autonomously for efficient fault management in the grid. DA uses an intelligent system for real-time grid monitoring and control to enhance the reliability of energy distribution. A DA system implements sensors at distribution components to report various aspects of their status, including voltage, current, frequency, etc., to the SG control center [86].

2.6.2.3. Distributed Energy Resources

SG introduces the concept of Distributed Energy Resources (DERs), such as solar panels and wind farms. DERs can provide electricity to their surrounding areas in case of power failures that disconnect these areas from the main grid. This distributed energy generation can save energy delivery costs since DERs are closer to clients than the Distribution Network Operators (DNOs). Smart Grid distribution systems that adopt DERs are not only cheaper and more efficient than the ones in conventional electrical grids but also more reliable and resilient [87].

Integrating DERs with the SG introduces a bidirectional electricity and data flow. Therefore, the Energy Management System (EMS) needs to adopt an active control system that can retrieve information about the status of the distribution network. As a result, large numbers of sensors are required to monitor the network's conditions, such as transformers' faults, the status of circuit breakers and switches, and the magnitude and direction of the electric flow [82]. Such data reports should be transmitted with high reliability and low delay to the controllers.

2.6.2.4. Electric Vehicles

Electric Vehicles (EVs) provide low-emission alternatives to conventional vehicles powered by fossil fuels. Moreover, recent developments in battery cell technologies have enhanced the commercial attractiveness of EVs [88].

Nevertheless, the integration of EVs into the SG remains challenging. Charging EVs during off-peak times, when power is less expensive, can lead to exceeding the feeder's thermal limits.

Likewise, periods when electricity prices are high can also be challenging. Since EVs can sell electricity back to the grid, if many EVs release electricity back to the grid system at the same time, they may harm its frequency stability.

An EV management system with an "On-the-go" communication model between the SG and the EVs plays a significant role in managing the charging and discharging of EVs. That's because it can collect and transmit their data in real time, which provides a basis for determining electricity prices [89]. Vehicle Ad Hoc Networks (VANETs) [90] can play a vital role in this communication model.

2.6.2.5. Home Energy Management Systems

HEMS can play a significant role in facilitating Dynamic Demand Control [91]. Thereby, allowing utilities to handle peak loads and enabling consumers to monitor their energy usage. A HAN serves as the network connecting a HEMS with its smart devices. In such a scheme, a reliable sensor network is required to:

- Monitor and control electricity consumption in real time.
- Interface with smart devices, meters, and plugs to facilitate electricity supervision.
- Help utilities to administer peak loads and dynamic demand response.

HEMS can rely mainly on resilient smart meters in their core functions. In more sophisticated scenarios, dedicated HEMS intelligent hardware can be used independently from smart meters.

2.6.2.6. Microgrid

Governments nowadays aim to reduce greenhouse gas emissions by enhancing energy efficiency and fostering renewable energy production. Until recently, Distributed Generation (DG) technologies, which are spread across the main electricity network and provide locally produced energy, were only regarded as secondary energy sources and have been poorly connected [92]. Nowadays, Distributed Generation via Microgrids is gaining momentum as it is increasingly being considered a primary supply of energy rather than a backup one.

A Microgrid is a distribution-level, small-scale power system that has DERs and storage components [93]. Since a Microgrid is an electricity source that can be closer to the consumer than a traditional power plant, it is more sustainable since electricity loss at its transmission power lines is less.

Microgrids are classified rather by their functionality than by their size. A microgrid typically makes up a small part of a distribution network. It can produce electricity locally and also connect to larger grids. This allows consumers to generate and use their own energy to satisfy their needs while still being able to access additional energy supplied through the main grid.

Microgrids combine renewable and non-renewable energy sources and deploy state-of-the-art energy management and storage technologies. They can work in parallel with the main grid (grid-connected mode) or disconnect from it (island mode) according to schedules, electricity needs, grid outages, or economic profitability [94].

2.6.3. Smart Grid communication requirements

The Smart Grid incorporates different types of communication networks with various coexisting applications. The heterogeneous data traffic patterns that exist in Smart Grids make QoS provisioning for SG applications a non-trivial task.

The fundamental communication requirements of Smart Grid applications can be defined as reliability, latency, scalability, and interoperability [95]. Table 2.1 lists reliability and latency constraints for some applications in the NAN segment of the SG. An extended version of these requirements can be found in [96]. To meet such diverse requirements, SG data traffic should be categorized into classes, and the SG communication protocols are expected to support service differentiation.

Type of traffic	Maximum allowed delay	Reliability
DERs data related to the protection of the distribution network	<4 s	>99.5 %
Critical traffic of: DA, DSM, AMI, DERs	<5 s	>99.5%
DA distribution network protection data	<3 s	>99.5%
Electric transport	<10 s	>98%
Non critical traffic of DSM & AMI	<15 s	>98%
Non critical traffic of DA & AMI	<30 s	>98%
Normal AMI traffic	<5 min	>98%
Network configuration traffic	hours/days	>98%

Table 2.1: Communication requirements for applications in the NAN segment of the SG [7]

2.7. Other IoT systems

Even though our work focuses on the Smart City and the Smart Grid, it can still be extended to other IoT systems. We briefly describe examples of these here, while later, in Section 2.9.3, more IoT systems and applications will be introduced.

2.7.1. Surveillance

In areas where security is crucial, IoT-based surveillance systems have the potential to provide real-time audio and video monitoring data, such as people entering and leaving a specific location, faces, or vehicle number plates [97]. They can also transmit security alerts remotely to dedicated end devices.

This way, data from locations with particular security concerns can be collected and preserved for usage as sources of evidence for potential investigation. Such surveillance systems could also be used to observe workplaces and track workers' actions for safety reasons [98].

2.7.2. Environmental monitoring

IoT-based sensor networks can be employed in several environmental monitoring scenarios, such as measuring the degree of air pollution or early warning of natural disasters like fires, earthquakes, and floods [99]. At a central authority, data collected from such sensors can be assessed to detect abnormalities and environmental phenomena [100].

2.7.3. Supply chains and logistics

RFID and sensor networks provide practical solutions that facilitate product tracking, storage monitoring, and payment processing in supply chain management systems [101]. Supply chains can benefit from IoT by acquiring detailed and up-to-date product data, which can improve their efficiency.

2.8. RPL Overview

RPL is a distance-vector proactive IPv6 routing protocol for LLNs [4]. Since it is designed to be implemented in environments where resources are highly constrained, its main characteristic is its good adaptability to the constantly changing conditions of its network.

For simplicity, we refer to LLNs that use RPL as RPL networks. Figure 2.9 displays the structure of such networks. RPL networks deal with various communication types and typically have large numbers of nodes.

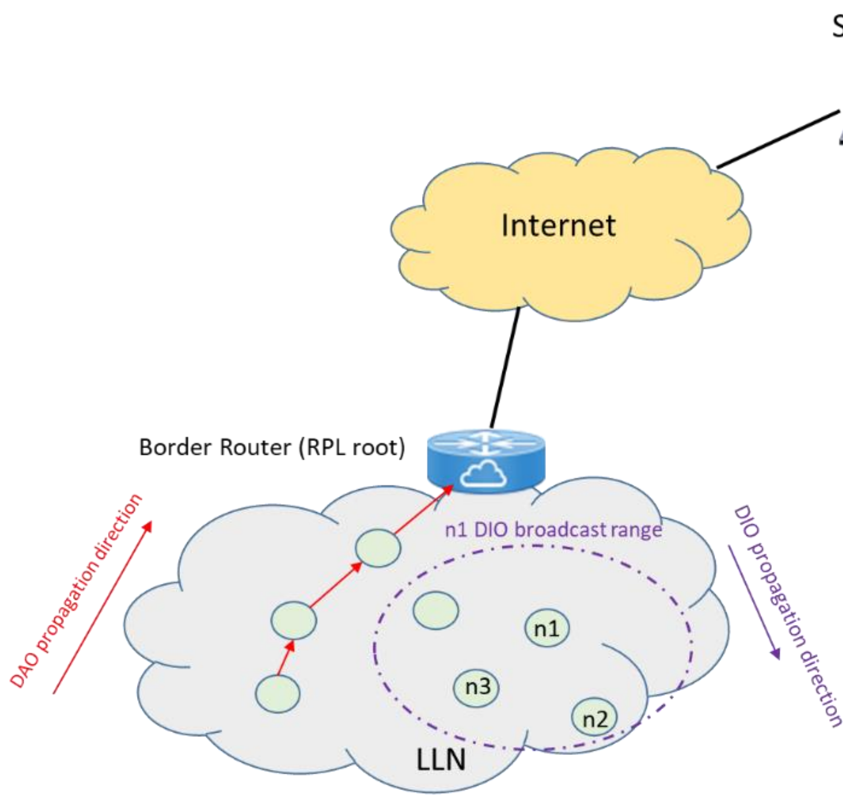


Figure 2.9: The RPL network architecture

We list in Table 2.2 the basic terms used by RPL and a brief explanation for each of them.

Term	Explanation
DAG	The network graph (topology)
DODAG	A network graph with only one root
Parent	A neighboring node that has a lower rank
Child	A neighboring node that has a higher rank
Metric	An estimation of the "cost" of a link or a path via a parent
OF	An Objective Function that evaluates metric(s) for selecting a preferred parent
Preferred parent	The parent with the lowest metric cost. A child node sends packets to the root via its preferred parent
Candidate parents	A list of parents that is monitored and updated periodically. Of which, a preferred parent is selected
Rank	A rank is a scalar value that gets lower the closer we get to the root. It represents the distance from the root
Instance	A logical topology built using specific metric(s) and an OF. Multiple OFs in the network could be used to generate multiple instances
Upward traffic	A traffic flow from the nodes to the root. This pattern exists in data collection applications
Downward traffic	A traffic flow from the root to a node within its network. This pattern exists in applications where control commands are sent to actuators
Trickle	The algorithm a node uses to control the intervals at which it broadcasts DIOs.
DIO	RPL routing control messages that are used to maintain upward routes (from the nodes to the root). Since RPL is proactive, these messages are broadcasted periodically
DAO	Control messages that are used to build downward routes (from the root to a node in the network)
ETX	A routing metric used to estimate the reliability of a link or a path

Table 2.2: The main RPL terminology

2.8.1. Routing metrics and the Objective Function

A routing metric is an estimation of the "cost" of sending a packet over a link or a path. Such cost could be a measurement of energy consumption, delay, or another QoS parameter. A routing metric is used to select the best route. RFC 6551 provides a set of routing metrics that can be used with RPL [4]. Constraint-based routing is also supported by RPL. When routing constraints are used, links and nodes that do not meet the requirements of a constraint are excluded from the best path selection. A node calculates the routing metrics of its neighbors and uses them as inputs to its Objective Function (OF). An OF is an algorithm a node uses to compute its rank and determine its preferred parent. A preferred parent is a neighbor node that provides the best route to the root, while a rank is a scalar value that corresponds to the distance from the root. There is no particular OF recommended by the RPL standard. Nevertheless, two specifications that define two OFs for RPL are proposed by IETF.

The Objective Function Zero (OF0) is described in RFC 6552 [53] as RPL's default OF. OF0 is developed to locate the closest root based on hop distance, but it doesn't ensure path optimization in terms of any particular metric. Since it uses the Hop Count (HC) as a routing metric, it enables interoperation across RPL implementations in various usage scenarios.

The Minimum Rank with Hysteresis Objective Function (MRHOF), specified in RFC 6719 [54], is the other OF standardized for RPL. MRHOF relies on the concept of hysteresis to reduce churn and to adapt to transient changes in metric values. If a node finds another path to the root with a lower cost, MRHOF's hysteresis allows choosing that path only if its cost is at least lower than the current path's cost by a specific value. MRHOF is designed to work with additive metrics.

The Expected Transmission count (ETX) routing metric is the most commonly used metric with MRHOF. ETX estimates the number of transmissions (including retransmissions) needed to deliver a packet over a link. It can be accumulated by all intermediate nodes on the path to the root to give an estimation of the path's reliability. MRHOF uses ETX to select the path with the highest reliability, which is the path that has the lowest ETX value.

2.8.2. RPL topology construction

RPL creates a Directed Acyclic Graph (DAG) topology originating from one or more root nodes. Each of these is called a DAG root and is responsible for connecting the RPL network to the Internet. In the RPL literature, other terms for a "DAG root" exist, including Broder Router (BR), RPL root, sink, and root [4]. In this Thesis, we use these terms interchangeably.

Figure 2.10 illustrates a DAG with one root, which is called a Destination-Oriented DAG (DODAG). The DODAG root initiates the process of building the network topology by broadcasting DAG Information Object (DIO) messages, which contain the information required for joining the network. The nodes nearest to the root are the first to receive these messages and determine whether to participate in the DODAG. If a node joins, it uses information in the received DIO messages as inputs to its OF to calculate its rank and metric values. Each node in the RPL network has a rank that must be less than the rank of its preferred parent. The ranking process is essential for avoiding and detecting routing loops that may occur due to topology changes.

After processing the received DIO messages, a node starts broadcasting its own DIOs, which are then received by its neighbors. These neighbors repeat the same steps by using these DIOs to join the DODAG, calculate their own DODAG parameters, and start broadcasting their DIOs. This process is repeated until all nodes join the network. The DODAG building and maintenance procedure depends on all the nodes in the network to periodically broadcast their DIOs.

Joining the network requires that each node selects one neighbor as its preferred parent. A preferred parent is a neighbor with the lowest rank. A node may receive DIO messages with different metric values from neighbors with the same rank. Such a node uses these metric values as inputs to its OF, which determines which neighbor should be selected as a preferred parent, while the others are kept in a backup list called the candidate parent list. The preferred parent serves as the primary relay node to the root, while the candidate parents are used as a backup for fault tolerance since RPL was intended for usage with lossy networks. As portrayed in Figure 2.9, node n2 is in the broadcast range of nodes n1 and n3. Based on their DIO parameters, n2 selects one of them as the preferred parent and the other as a candidate parent.

The purpose of building a DODAG is to maximize the routing performance for the Multipoint-to-Point (MP2P) traffic. This is called the upward routing direction (from the nodes to the root), and it is the dominant traffic direction in LLNs [4, 12]. Nonetheless, RPL supports routing data from the root to the nodes (the downward routing direction), which is Point-to-Multipoint (P2MP) traffic. Point-to-Point (P2P) traffic is also supported by RPL.

RPL nodes transmit Destination Advertisement Object (DAO) messages upwards to set up and maintain downward routes from a DODAG root toward leaf nodes. Figure 2.9 shows the propagation directions of DAO and DIO messages. Each node uses DAO messages to create a DAO parent set, which is a subset of its DODAG parent set (preferred and candidate parents). For managing RPL's downward routes, a DODAG operates in either the storing or the non-storing mode. In the storing mode, a node keeps a downward routing table for its sub-DODAG. On the other hand, if the non-storing mode is used, then packet routing is accomplished using source routes created by the DODAG root using the DAO messages it receives.

Point-to-point traffic in RPL relies on both downward and upward routes. P2P packets require a mutual ancestor with a valid route that allows them to reach their destination as they propagate toward a DODAG root. If RPL is operating in the non-storing mode, the DODAG root serves as the common ancestor. The primary disadvantage of P2P routing, as defined by the RPL standard [4], is that the routes from the source to the destination could be less than optimal, which means congestion might occur close to the root. Additionally, automation-based applications of IoT, such as those implemented for Smart Homes, need on-demand communication between devices. This goes in contrast to RPL's proactive approach.

A Trickle timer (RFC 6206) is used by nodes to adjust broadcasting DIO messages [55]. When there is a discrepancy among nodes, the Trickle algorithm propagates new information fast. On the other hand, when there is agreement amongst the nodes, the Trickle algorithm exponentially increases the sending interval so that the DIO broadcast is slowed down. The Trickle algorithm accomplishes energy efficiency and scalability by utilizing such adaptive transmission interval management along with its suppression mechanism.

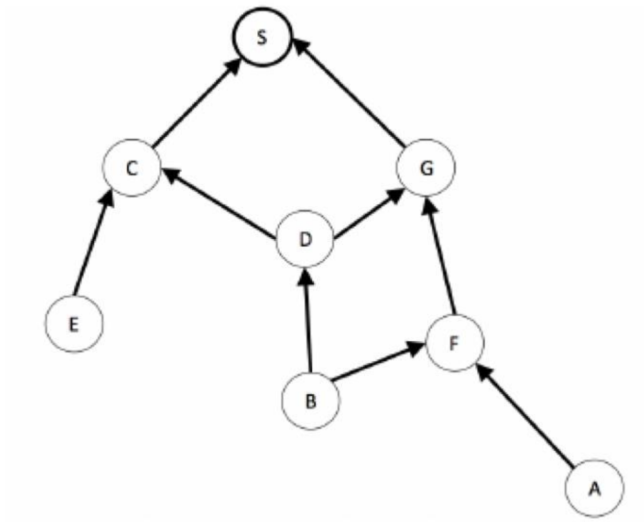


Figure 2.10: An example of a DODAG. S is the DODAG root or Border Router

2.8.3. The Trickle algorithm

Trickle is a density-aware algorithm used by RPL to adjust the rate of broadcasting its control messages, i.e., its DIO messages [55]. When a node has conflicting information with its neighbors, it sets the rate of its control message broadcast to its minimum (e.g., in order of milliseconds) to handle the inconsistency. On the other hand, when the nodes agree, that indicates that the network is stable. Therefore, the node's broadcast rate of control messages is reduced to reach as low as a few messages per hour.

Trickle handles information dissemination efficiently due to its DIO suppression mechanism and its adaptive sending rate. The main principle is that a node hears broadcasts from nearby nodes. If their data corresponds to the node's own information about the state of the network, then the node declares its own broadcast as redundant and suppresses it.

2.8.3.1. DIO inconsistency

A DIO transmission is governed by a Trickle timer. An example of consistent information is when a node receives a DIO that does not cause it to change its preferred parent or rank. However, when the DIO causes certain parameters at the node to change, such as its rank or preferred parent, then the DIO is deemed as inconsistent or incoherent.

Inconsistency can result from the node hearing outdated information or new information that makes its own state outdated. An example of the latter is receiving a DIO with a DODAG version number that is higher than the node's own. The DODAG version number is a sequence number that the DODAG root increments each time it creates a new version of the DODAG, i.e., each time it performs a global repair. This repair is applied to adapt to topology changes and could produce a different DODAG topology. Consequently, a node detecting a higher DODAG version number needs to migrate to this new version quickly. After that, it should reset its Trickle timer to propagate this information faster. Another example of incoherence is when a node discovers forwarding errors. These are spotted when the direction the packet being forwarded (up or down) does not match the value of its up/down bit "o". Inconsistency can also emerge when a routing loop is detected.

2.8.3.2. The Trickle mechanism

A Trickle interval is the length of the period during which a node listens to broadcasts from its neighbors and evaluates whether to broadcast a DIO or not. Longer intervals mean more listening periods and more energy saving.

The Trickle timer has three configurable parameters that define its interval. These parameters are learned from the DIO received from the preferred parent:

- k: An unsigned integer called the redundancy constant.
- I_{min}: The minimum interval size defined in units of time. Its default value in RPL is 8 Milliseconds.
- I_{max}: The maximum interval size. It is calculated based on I_{min} as shown in equation (2-1). Its default value is 20, which produces a maximum interval period of 2.3 hours.

$$I_{max} = I_{min} * 2^{I_{max}} \tag{2-1}$$

Each node also monitors three additional Trickle variables:

- c: A consistency counter.
- I: The current interval duration.
- t: A time within the current interval.

The steps of the Trickle algorithm can be described as seen below:

1. In the beginning, the first interval size is set: $I \in [I_{min}, I_{max}]$.
2. c is reset to 0, and t is chosen randomly in the range: $t \in [I/2, I]$. The first half of the interval ($t < I/2$) is called the listen-only period. With each consistent broadcast received in this period, the node increments the consistency counter c by one.
3. When t is reached, the node decides whether to broadcast a DIO by comparing the current value of its consistency counter c with the redundancy constant k:
   - If $c < k$ the node broadcasts a DIO.
   - If $c \geq k$ the broadcast is suppressed.
4. At the end of the interval I, the duration of the next interval I_2 is set to: $I_2 = 2 * I$.
5. Whenever an incoherent DIO is received, the Trickle timer is reset by setting I to I_{min}.

Figure 2.10 demonstrates the behavior of the Trickle algorithm at node N1 in a setup with a redundancy constant $k = 2$:

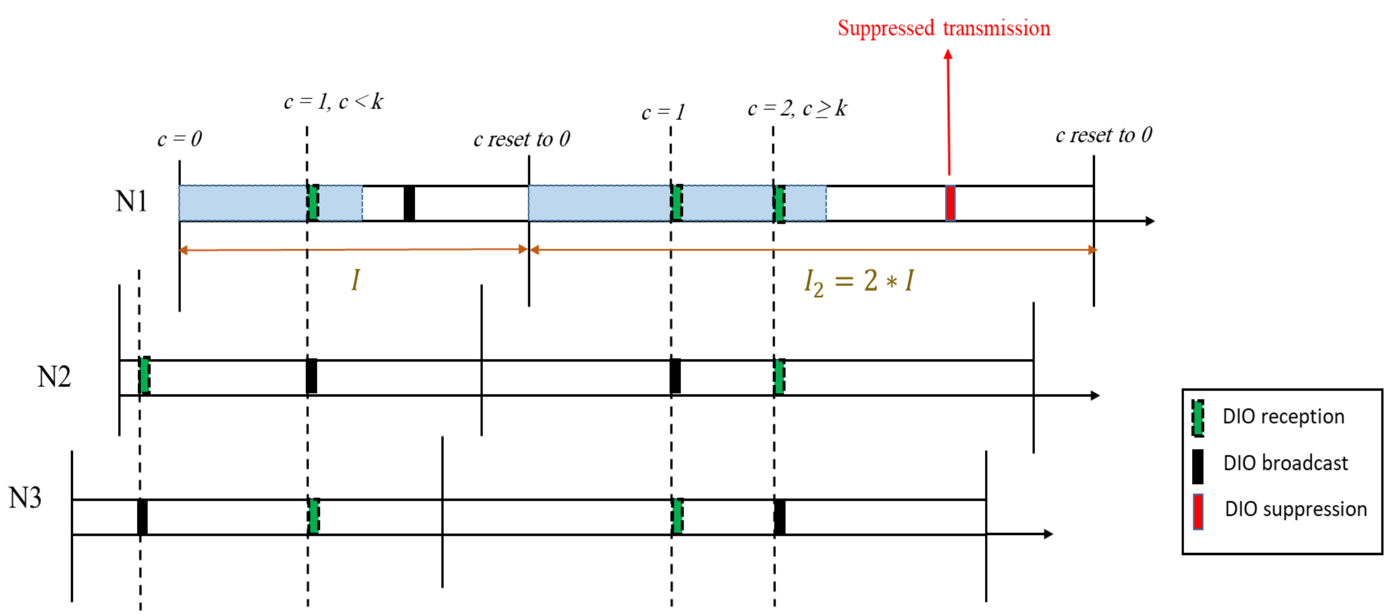


Figure 2.11: The Trickle algorithm from the perspective of node N1

2.8.4. Supporting service differentiation using RPL instances

What enables RPL to accommodate diverse routing requirements simultaneously is its structure. RPL supports traffic differentiation at the network layer through the concept of instances [4], as illustrated in Figure 1.1. An RPL instance is a logical subdivision of the network's topology, i.e., it's a virtual sub-topology of the overall topology. Multiple instances can coexist in the same network. Each instance can be set up to have its distinct metrics and Objective Function so that it supports specific QoS routing requirements.

Since multiple RPL instances may operate simultaneously in a network, and a node may be part of more than one RPL instance. Logically, however, the instances remain independent, with each instance labeled by a unique identifier called an RPL_Instance_ID.

2.8.5. RPL Control Messages

The RPL topology is defined and maintained using four types of control messages:

1. DIO: Contains information about the RPL instance, the IPv6 address of the root, the rank of the sending node, and the routing metrics/constraints.
2. DAO: These messages solely disseminate the destination information to the root. They are used only for building downward routes that enable the root to discover paths to the leaf nodes.
3. DODAG Information Solicitation (DIS): A DIS message is used for requesting information (a DIO message) from an RPL neighboring node. Those messages are sent by nodes that have not yet joined the DODAG.
4. DAO-Acknowledgement (DAO-ACK): A message that is sent as a reply to a DAO.

The scope of RPL's control messages is the link. That means their source address is a link-local address, and their destination address is either:

- The all-RPL-nodes multicast address: A new address with the value ff02::1a
- A link-local unicast address of the destination.

One exception to that is the DAO/DAO-ACK messages, which, when operating in non-storing mode, use global unicast addresses for both the source and the destination.

The generic format of an RPL control message (Figure 2.12) includes:

1. Internet Control Message Protocol version 6 (ICMPv6) header: Type, code, checksum. All RPL Control Messages have their Type header field set to 155 to distinguish them from other ICMPv6 messages.
2. Message body: The message body is comprised of:
    a. A message base.
    b. A number of options (if applicable).

The Base header field carries one of the four types of RPL control messages. The Code field identifies which RPL control message is included:

- 0x00: DIS.
- 0x01: DIO.
- 0x02: DAO.
- 0x03: DAO-ACK.

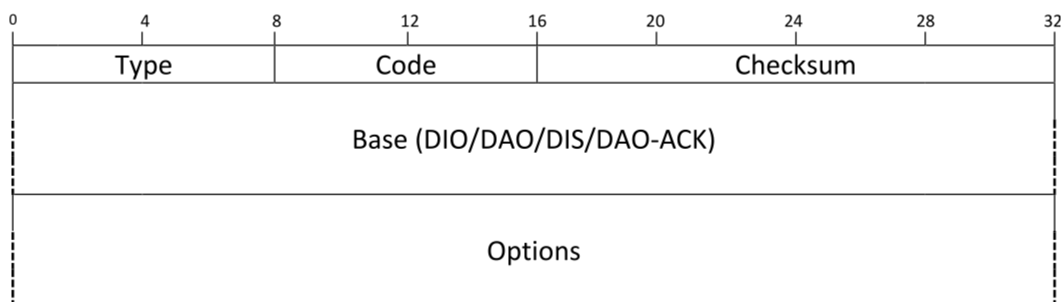


Figure 2.12: The generic format of an RPL control message

2.8.6. DIO message format

This message type, displayed in Figure 2.13, is broadcasted periodically within intervals set by the Trickle algorithm. It carries information required to join a DODAG, discover its configuration settings, select a parent set, and perform topology maintenance.

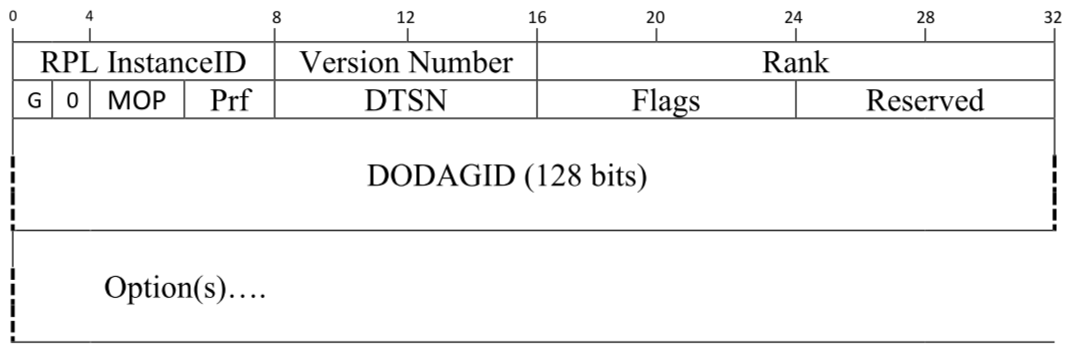


Figure 2.13: The DIO Base Object

We provide a short description of the main DIO parameters:

- RPL_Instance_ID: Each instance has a number or ID set by the root. It is used to distinguish DIO messages from different DODAGs belonging to different instances.
- Version Number: The sequence number of the current version of the DODAG.
- MOP: Mode of Operation. A value of "0" means that downward routes are disabled.
- Rank: Rank of the node sending the DIO message.
- Destination Advertisement Trigger Sequence Number (DTSN): Used for downward routes.
- Flags and Reserved: Not used and set to zero.
- DODAG_ID: A 128-bit IPv6 address set by a DODAG root.
- Options: Different configuration options may be carried within a DIO, such as routing metrics or prefix information.

The Options filed in the DIO can significantly increase its size. We will highlight the most common DIO options in the next paragraph.

2.8.6.1. DIO options

Multiple options can be carried within a DIO. The DAG Metric Container is an essential DIO option in RPL. It is used for propagating the values of the routing metrics in the network. Different applications require different routing metrics. Thus, the DAG Metric Container could carry energy, throughput, or other routing metrics or constraints. Figure 2.14 illustrates the general format of the DAG Metric Container.

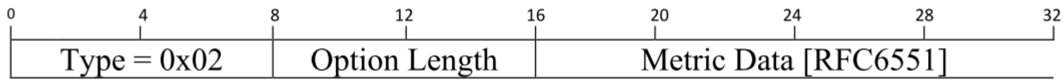


Figure 2.14: Format of the DAG Metric Container Option used for carrying metric(s) data

Other DIO options include the DODAG configuration option, which is used to disseminate the DODAG configuration information, and the Prefix Information Option (PIO), used for address autoconfiguration.

2.9. Discussion and thesis scope definition

We can now define the scope of this thesis based on the information introduced in this Chapter. In the next Chapter, we outline the thesis scope in terms of RPL's research literature.

2.9.1. Thesis scope with regard to the IoT protocol suit

We illustrate in Figure 2.15 the IoT standards and protocols this thesis is concerned with.

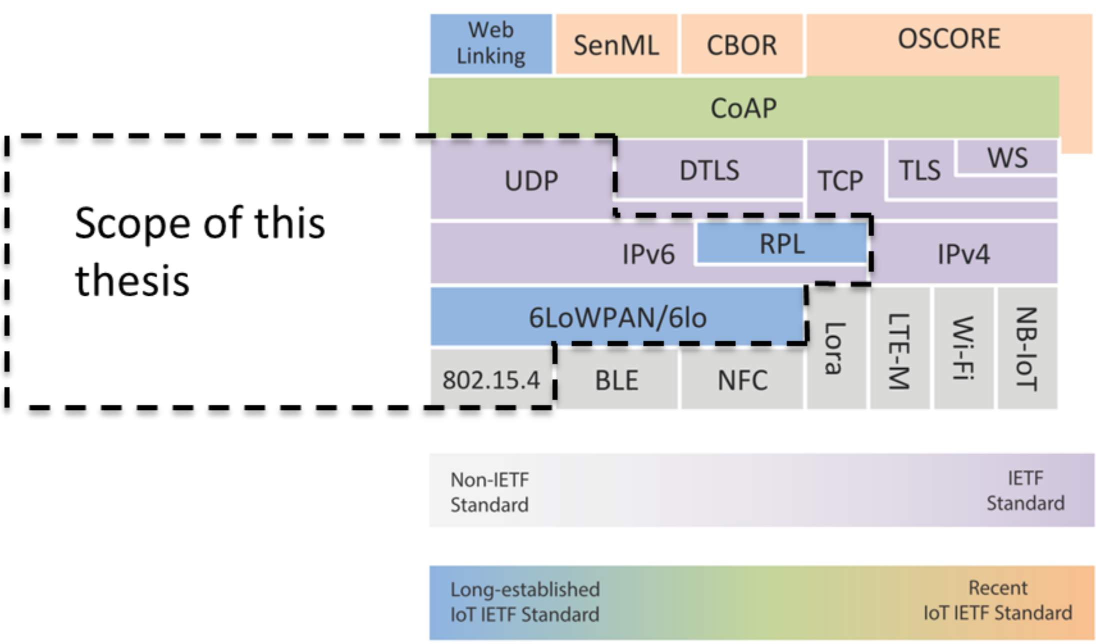


Figure 2.15: The protocol stack of IoT devices [102] with the thesis scope illustrated

2.9.2. Thesis scope regarding IETF's standards

As introduced in Section 2.4, the standardization efforts of IETF for IoT have produced multiple RFCs. This Thesis belongs to the connectivity and routing domains of these standards.

Our work focuses on ROLL RFCs [48]. They define the mechanisms required for RPL's operations, RPL's applicability in some IoT systems, and the routing requirements of LLNs in different settings. This Thesis also deals with the RFCs for transmitting IPv6 over LLNs defined by the 6LoWPAN [36] and the 6lo [42] WGs, which are closely related to RPL.

Figure 2.16 highlights the Thesis scope and its RFCs of interest. These RFCs are essential to take into consideration in any RPL implementation. In our evaluations, we referred to the RFCs regarding RPL's core mechanisms for adjusting the values of many RPL parameters. While RPL's applicability RFCs and the routing requirements RFCs were used as guidelines for setting up our simulation scenarios.

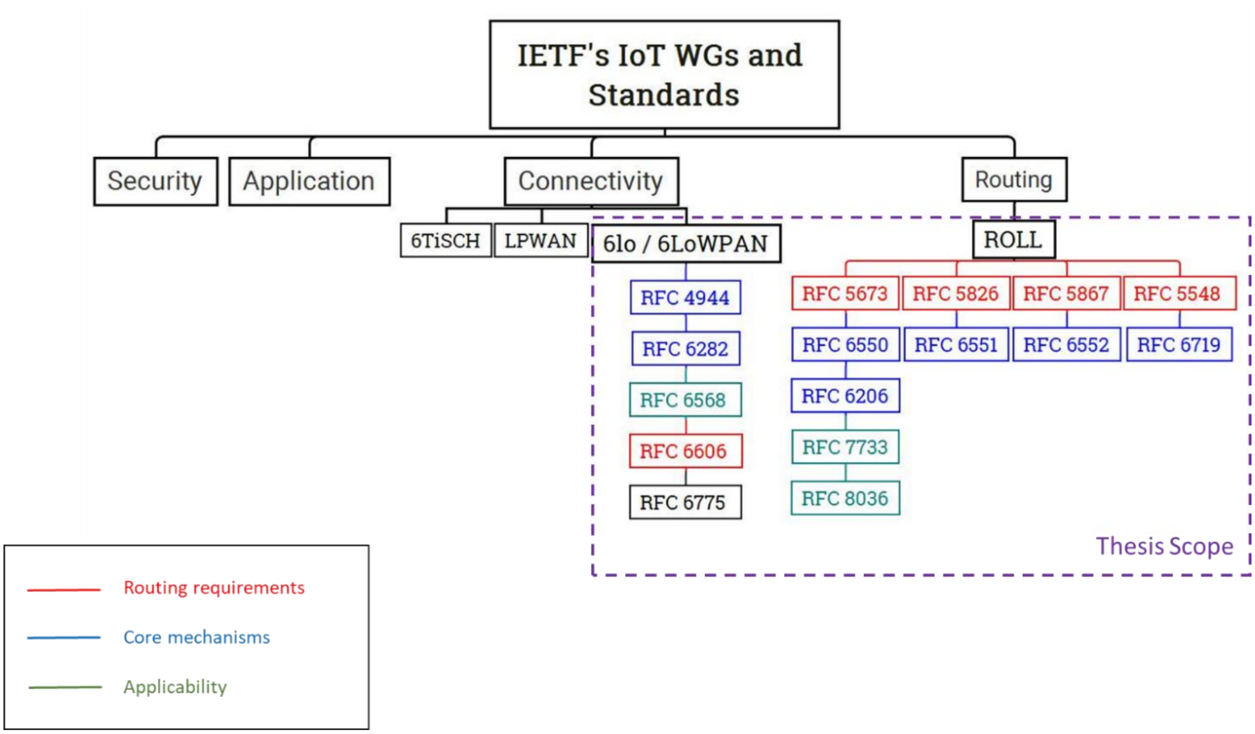


Figure 2.16: The primary RFCs relevant to this Thesis

2.9.3. Thesis scope within the context of IoT domains and applications

Figure 2.17 illustrates the fundamental IoT domains and applications based on [103-106]. Our research focuses on the Smart Grid and the Smart City as elements of the IoT Community Domain.

The Smart Grid scenario we study in this Thesis is a NAN segment of SG, where multiple applications with different QoS demands, such as AMI and DA, coexist. Our Smart City scenario of interest, which will be studied in a dedicated chapter, is a Smart City ecosystem where multiple networks from different authorities overlap. In both scenarios, the networks used are LLNs with multiple RPL instances.

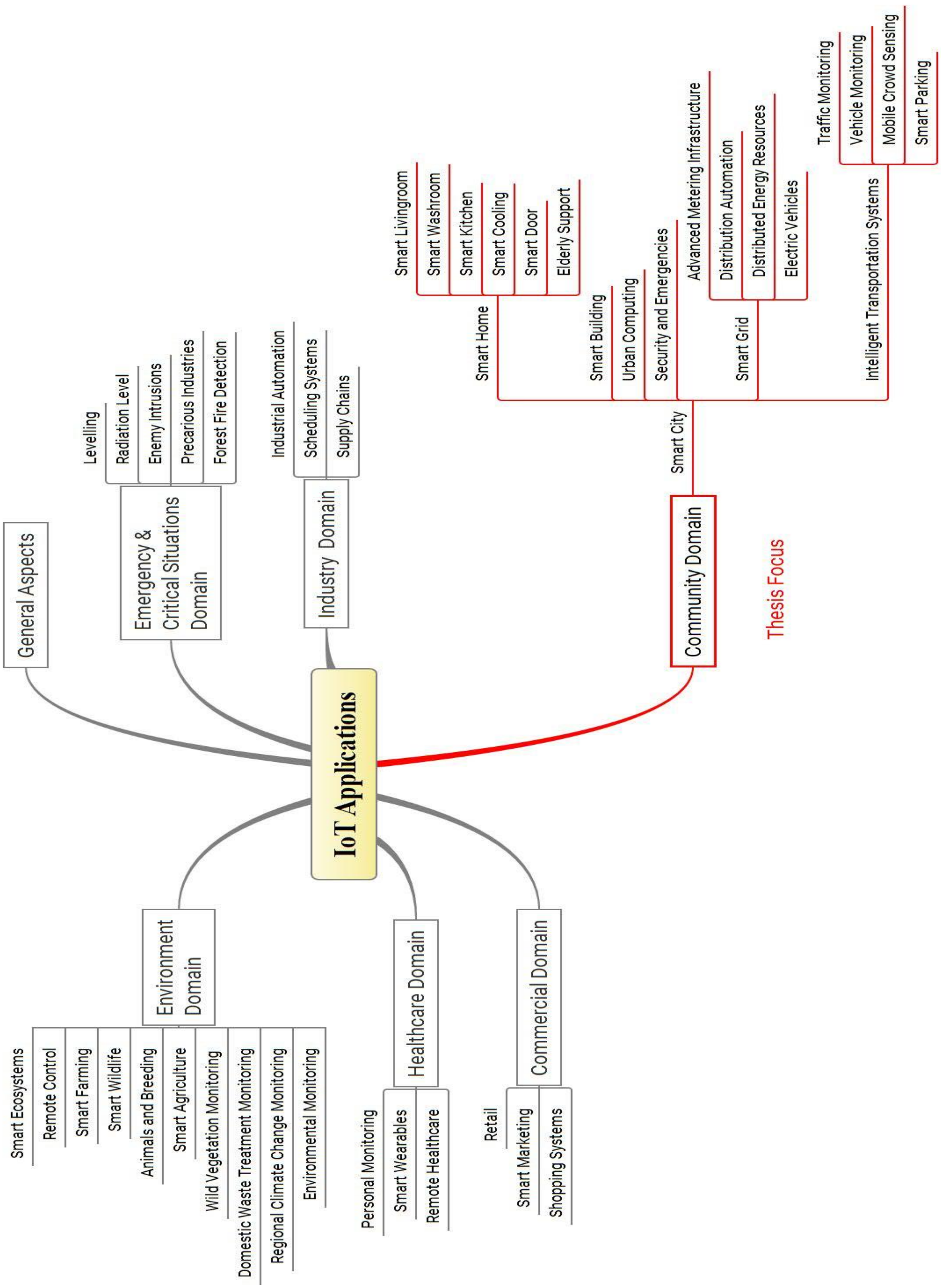


Figure 2.17: IoT domains and applications (including the Thesis scope)

2.10. Summary

In this Chapter, we have presented an overview of IoT and two of its main systems: the Smart City and the Smart Grid. We have illustrated the role of LLNs in IoT generally and in these two domains particularly.

The multi-technology networks that compose an SG, and the heterogeneous nature of its traffic, make traffic prioritization crucial in a Smart Grid since its applications have diverse requirements and priorities. RPL supports service differentiation in such scenarios by establishing multiple instances in its network.

Despite its ambitious design, RPL still suffers from shortcomings investigated in detail in numerous studies [107-109]. The next Chapter addresses RPL's challenges that are relevant to this Thesis and the research efforts to tackle them.

3. Literature review

Congestion can become a serious issue that worsens the performance of LLNs. The standard RPL does not provide any solutions for handling congestion or dealing with heavy traffic. In this Chapter, we survey multiple substantial RPL improvements that tackle the problems of congestion control and load imbalance. We analyze these research contributions in-depth and summarize their differences and limitations. Additionally, since we believe that exploiting multiple RPL instances can help in alleviating congestion, we examine the congestion control studies that go in this direction.

Next, we examine the research field of IoT environments where multiple RPL instances coexist. We investigate the trends in this domain and explore whether any previous research has targeted cooperation among multiple RPL instances.

These two RPL research domains described above form the basis for our proposed model, which we introduce in the next Chapter.

Finally, we review the schemes for promoting cooperation among communication networks using virtual currency. This review forms the background for Chapter 6, where we propose a framework for cooperation encouragement among multiple RPL instances using virtual credits.

3.1. Prerequisites

We introduce here background information that helps in understanding the upcoming Sections of this Chapter.

Many research papers that we survey in this Chapter evaluate their work using the Contiki-OS [110], which is an IoT operating system for LLNs. It implements the standard RPL as specified in RFC 6550 [4] and its related documents. This implementation is referred to as Contiki-RPL [111]. In this Thesis, we use the terms “Contiki-RPL” and “the standard RPL” interchangeably when referring to the implementation of the standard RPL in the Contiki-OS. Applications designed in the Contiki-OS are evaluated either on a real testbed or using a dedicated simulator for the Contiki-OS called the COOJA simulator [112].

An LLN node relies on an internal First In - First Out (FIFO) buffer (buf) to store its outgoing data packets (pkts) before transmitting them. This buffer is called the Queue buffer [113] and is displayed in Figure 3.1. The Contiki-OS implements the Queue buffer module at the MAC layer of the TCP/IP model [114]. It is worth noting that Figure 3.1 also displays two other buffers, namely the pkt_buf and the uIP_buf. These are single-packet buffers that are used when processing the headers of incoming and outgoing IPv6 packets. They are not relevant to this Thesis since they do not affect congestion.

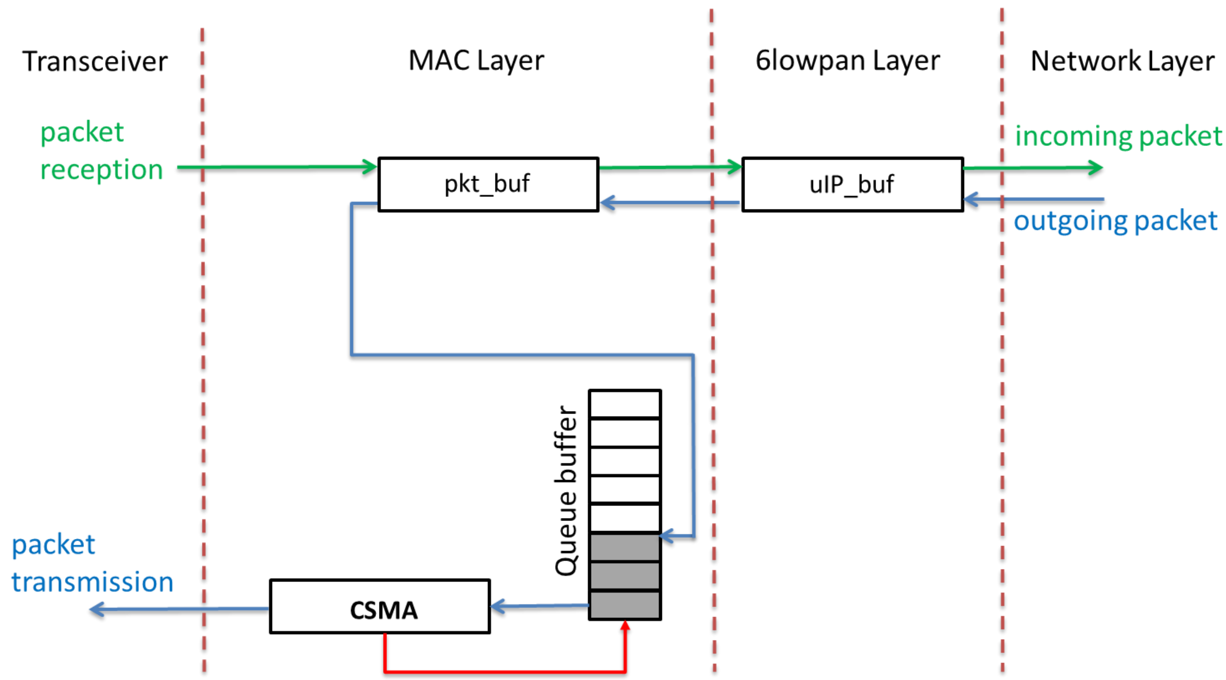


Figure 3.1: The Queue buffer module in the Contiki-OS

Carrier-Sense Multiple Access (CSMA) is a MAC protocol a node uses for channel access in LLNs [23]. It is widely used with wireless links such as the IEEE 802.15.4 links. To avoid collisions in the shared wireless medium, an LLN node that uses CSMA waits for a random period, called the backoff period, before starting a transmission. After the backoff period, the node listens to the wireless channel to check if a transmission by a neighbor node is in progress. If the node finds the channel idle, it transmits its data. If it finds the channel busy, the node waits for another random backoff period before listening to the channel again to assess it.

If the sender was able to transmit a packet, it is removed from the sender's Queue buffer only after it receives an Acknowledgment (ACK) from the receiver that the packet was delivered. If an ACK is not received, the sender retransmits the packet. If no ACK was received after a certain number of retransmissions, the packet is deleted and the transmission is deemed unsuccessful. On the other hand, If the sender is not able to transmit the packet after a certain number of CSMA backoffs, the sender cancels the transmission of the packet and deletes it from its Queue buffer.

The maximum duration of a backoff period, the maximum number of channel access retries, and the maximum number of retransmissions all depend on the CSMA configuration. These CSMA parameters control how long an outgoing packet stays stored in the Queue buffer, as Figure 3.1 shows. When the transmission of a packet is pending, other outgoing packets, which are packets that a node either generates or forwards, are stored in the FIFO Queue buffer. If the buffer is full, these packets are dropped. The longer the outgoing packets stay in the Queue buffer, the more it is prone to overflow. Congestion at a node occurs when its Queue buffer overflows or when its capacity exceeds a certain threshold.

In LLNs, a node may need a number of retransmissions to send a packet due to the lossy nature of the LLN radio links, which inflicts transmission errors and packet loss. Furthermore, under heavy traffic, the channel is busy most of the time, which forces the nodes to use multiple CSMA backoffs before sending a packet. In both cases, the packets stay longer in the sender's Queue buffer, thereby increasing the probability of congestion.

The allocated Queue buffer size in an important parameter to assess the congestion level of an LLN node. It is measured by calculating the Queue length (Q), which is the number of packets stored in a node's Queue buffer awaiting transmission [115]. The Queue length is also referred to as the Buffer Occupancy (BO) [116]. The maximum buffer capacity, which is the maximum number of packets a node can store in its Queue buffer, is referred to as Q_{max}. The occupied Queue buffer size at an LLN node can also measured as a percentage of Q_{max}. In this case, it is called the Queue Utilization (QU) and calculated as follows [15]:

$$QU = \frac{Queue\ length}{Maximum\ buffer\ capacity} = \frac{Q}{Q_{max}}$$

The maximum buffer capacity depends on the internal memory available.

3.2. Congestion control in RPL networks

In large LLNs, congestion can become an issue even if nodes generate data at low rates; since congestion can take place in network locations where traffic is concentrated [15]. Congestion increases delay and induces packet losses that degrade the reliability of applications. It also causes increased energy consumption and decreases the network's lifetime.

From a general perspective, congestion control and load balancing procedures in LLNs can be categorized into [117]:

- Congestion detection.
- Congestion notification.
- Congestion mitigation.

When it comes to RPL, such categorization must also take into account these factors:

- Routing metrics.
- Traffic pattern: P2P, MP2P, or P2MP.
- Exploiting path diversity.

Therefore, RPL-based LLNs follow certain procedures for congestion detection, notification, and mitigation.

3.2.1. Congestion detection

RPL research papers in this category focus on designing routing metrics that indicate the level of path congestion in order to forward packets via less-congested routes. Many of these protocols combine multiple routing metrics to detect congestion with a better resolution. A summary of these schemes is shown in Figure 3.2.

3.2.2. Congestion notification

In this approach, when a node detects congestion, it broadcasts DIO messages that include the congestion details. Child nodes learn about the congestion from the DIO messages they receive from their parents. In this case, the parent typically sets a Congestion Notification (CN) bit in its DIO header.

3.2.3. Congestion mitigation

The methods to mitigate congestion are divided into two groups:

- Resource control: Involves alleviating congestion by optimizing the network resources using power control, multipath routing, and alternative path selection mechanisms.
- A combination of traffic control and resource control: The traffic control method seeks to mitigate congestion by adjusting the sender's transmission rate.

RPL schemes are rarely involved with the adjustment of the sending rate because this is not a function of a network layer protocol. Therefore, the resource control approach is more common than the traffic control approach for congestion alleviation in RPL networks.

3.2.4. RPL enhancements for congestion control

We review the RPL modifications and extensions proposed by the research community for tackling the issue of congestion in LLNs.

Ullah *et al.* designed an Energy and Congestion-aware Routing Metric (ECRM) for adaptive parent selection in RPL in AMI networks [118]. ECRM is a routing metric used by nodes for rank calculation. It is a compound metric of ETX, Remaining Energy (RE), and QU. Ullah *et al.* used the COOJA simulator [112] in their evaluations and compared their work with ELPS [119], an RPL enhancement that uses an energy-aware composite routing metric. Results showed that ECRM had less average power consumption and a higher Packet Delivery Ratio (PDR) than ELPS. Nevertheless, ECRM's congestion detection is based on static thresholds.

Bhandari *et al.* proposed a Congestion-Aware RPL (CoAR) [120]. CoAR uses the same routing metrics as ECRM, but instead of combining them into a single metric, it uses them as inputs for a multi-criteria decision-making Objective Function. Its parent selection process also considers avoiding nodes with many children and few parents.

Through COOJA simulations, Bhandari *et al.* evaluated CoAR's performance against the standard RPL and ECRM [118]. Their results demonstrated that CoAR performed better in terms of PDR, energy consumption, end-to-end delay, and throughput.

To alleviate the congestion between a child node and its parent, Sheu *et al.* designed a Game Theory-based Congestion Control (GTCC) protocol [121]. In this model, each node calculates the net packet flow rate, which is the packet generation rate minus the packet service rate. If this value is positive, then congestion is detected. In this case, the node notifies its children about the congestion by setting the CN bit in its DIO packets. Each child node then reacts by starting a game-theory-based parent change.

GTCC tackles the congestion problem caused by an excessive number of nodes selecting the same parent. Sheu *et al.* introduce a parent selection potential game where the players are child nodes who receive a congestion notification from the same parent. Here, only one node is allowed to change its parent at each round. If the congestion level remains high even after the parent changes, GTCC notifies the sending nodes to reduce their packet-sending rates.

By leveraging the COOJA simulator, Sheu *et al.* evaluated GTCC in comparison to two versions of Contiki-RPL: one configured with OF-ETX and the other with OF0, which utilize ETX, and HC, respectively.

According to the simulation results, as the transmission rate increased, GTCC outperformed the other two Contiki-RPL protocols with respect to throughput and PDR. However, GTCC requires that each parent includes all its children's transmission rates in its periodic DIO packets. As a result, the size of the DIO messages increases with the number of children, which makes GTCC a resource-consuming solution in high-density networks.

A Queue Utilization RPL (QU-RPL) [15] is the solution Kim *et al.* propose for the problems of load balancing and congestion mitigation in LLNs. The authors observed that if the link capacity is higher than the buffer capacity of the nodes, then ETX may not be a good indicator of congestion under heavy traffic.

Kim *et al.* designed a QU routing metric for parent selection. This QU metric is calculated as the ratio of the number of queued packets to the maximum queue size at a node's buffer. The rank calculation in QU-RPL is based on the sum of three metrics: weighted QU, Hop Count, and ETX. Where weighted QU is the value of the QU metric multiplied by a coefficient. In QU-RPL's evaluation, the value of the coefficient was chosen as a positive integer ranging from 1 to 5.

By experimenting on a real testbed with 30 nodes without employing a Radio Duty Cycle (RDC) mechanism, Kim *et al.* found that QU-RPL enhances the PDR since it significantly reduces the average queue loss ratio compared to the standard RPL. Yet, larger weight values of the QU metric were found to cause nodes to choose longer paths. Besides, QU-RPL's power consumption was not studied.

To address the load imbalance as well as the hidden terminal problem, Kim *et al.* improved their previous work, QU-RPL [15], by designing a Power-Controlled RPL (PC-RPL) [122]. The hidden terminal problem appears when nodes can communicate wirelessly with an access point but not with one another. In this case, a node sends packets to the wireless access point without being able to detect the contenders' transmissions, which leads to packet collisions that increase packet loss and degrade throughput. PC-RPL relies on a metric composed of ETX, HC, and the Received Signal Strength Indicator (RSSI). Reference RSSI values from all neighbors of a node are obtained by making them broadcast DIO messages with maximum transmission power.

Kim *et al.* assessed the performance of PC-RPL in comparison to the standard RPL and QU-RPL using a real testbed with 50 nodes. Results showed that PC-RPL was able to tackle the hidden terminal problem and achieve better PDR than QU-RPL. However, the method for calculating reference RSSI values in PC-RPL may lead to excessive energy consumption.

Optimization-based Hybrid Congestion Alleviation (OHCA), which employs a resource control strategy and a traffic control strategy, was proposed by Al-Kashoash *et al.* in 2017 [116]. In this model, congestion is detected if the arrival rate exceeds the service rate. A congested parent notifies its children about congestion via DIO messages. Each child then starts a resource control procedure by trying to select an alternative non-congested parent. If it fails, it follows a traffic control procedure by reducing its sending rate.

The authors evaluated OHCA against the Duty Cycle-aware Congestion Control for 6LoWPAN (DCCC6) [123] as well as QU-RPL [15] using the COOJA simulator. Similar to OHCA and QU-RPL, DCCC6 uses the node's BO to detect congestion, but it employs a rate adaption strategy that considers duty cycling. Simulations showed better fairness, packet loss rate, end-to-end delay, and energy consumption results for OHCA compared to QU-RPL and DCCC6. Nevertheless, the parent selection process in OHCA is rather complex since it relies on multi-criteria optimization. The effects of this process on the network's stability and control messages overhead were not studied.

Al-Kashoash *et al.* formulated a Congestion-Aware Objective Function (CA-OF) [124], which uses BO and ETX as inputs. CA-OF assigns different weights to each input depending on the traffic level. Al-Kashoash *et al.* explain that ETX is the better indicator of packet loss at low data rates, where the wireless channel is the main reason for packet loss. While at high data rates, BO is a more significant packet loss indicator since most lost packets are dropped at the nodes due to buffer overflow.

The COOJA simulator was utilized to evaluate CA-OF's performance compared to three implementations of the standard RPL, each with a different Objective Function: OF0, OF-ETX, and Energy-OF. CA-OF was found to outperform the standard RPL implementations when it comes to PDR and energy consumption.

Tang *et al.* propose a multipath improvement of RPL called Congestion Avoidance RPL (CA-RPL) [125]. This protocol was designed for emergency scenarios, which generate high traffic volumes, and require alarm data to be delivered with high reliability and low delay. For this purpose, a new routing metric called DELAY_ROOT is introduced in CA-RPL to minimize the delay toward the sink based on the wake-up intervals of the candidate parents.

DELAY_ROOT is combined with three other metrics: ETX, the number of received packets during a time interval, and the parent's rank. After that, the node splits its traffic among multiple parents depending on the values of their composite metrics. The performance evaluation of CA-RPL against the standard RPL was performed using the COOJA simulator. CA-RPL outperformed the standard RPL in PDR, latency, throughput, and the number of packets received by the root per second. Despite that, when the nodes generated traffic at lower rates (about one packet every 8 seconds), CA-RPL had a higher packet loss ratio and a lower throughput than the standard RPL.

To offer more precise estimations of network reliability, Marco *et al.* offer two new metrics that exploit the MAC layer's information [126]. The first is the R-metric, which represents the end-to-end reliability between two nodes. The R-metric is similar to the ETX metric but takes into account the effects of the contention at the MAC layer on reliability. The second metric Marco *et al.* introduce is the Q-metric, which is designed for load balancing by avoiding overloaded parents.

The COOJA simulator was used to evaluate the proposed metrics relative to the standard RPL and the Backpressure Collection Protocol [127]. Both the R-metric and the Q-metric achieved higher overall end-to-end reliability, with the Q-metric achieving the best energy consumption balancing. Still, these simulations were performed with less than 20 nodes, which is a small number to judge the performance of CA-RPL properly.

In LLNs with heavy and time-varying traffic that incorporate mobile nodes, RPL needs to adapt to network dynamics. For this reason, Tahir *et al.* combined RPL with backpressure routing [128] to create Backpressure RPL (BRPL) [9]. Unlike standard routing algorithms, backpressure routing does not perform source-to-destination path calculations. Instead, it makes per-packet forwarding decisions by computing a link weight that is a function of a local queue and link-state information. In BRPL, the link weight is computed using queue size, rank and two algorithms designed to adapt to dynamic traffic and mobility.

BRPL was implemented in a real testbed with 100 nodes without employing an RDC mechanism. In the experiments with dynamic traffic, BRPL showed 50% less packet loss than the standard RPL. Nevertheless, the results also showed that BRPL had higher delay and communication overhead. This drawback was observed at high traffic rates (2 packets/second or higher), where BRPL used suboptimal paths to achieve better throughput.

RPL's increased packet loss and power consumption under heavy and dynamic traffic conditions have motivated Taghizadeh *et al.* to design Context-aware and Load-balancing RPL (CLRPL) [14]. The authors introduce a Context-Aware Objective Function (CAOF) that uses two path metrics: ETX for path reliability estimation and a novel metric for assessing the remaining energy of the chain of ancestor nodes on the path to the root. A node calculates the residual energy value used in CAOF based on its own energy level and its parent's. This calculation is recursively repeated while building and maintaining the DODAG. A new routing metric called the Context-Aware Routing Metric (CARF) was also designed for CLRPL to estimate QU and the traffic load dynamicity. CARF is calculated using the BO values of the ancestor nodes and a traffic dynamicity index Taghizadeh *et al.* proposed.

CLRPL was evaluated against the standard RPL using the COOJA simulator in two settings. The first had high traffic rates (15-100 packets/minute). The second one was performed under dynamic traffic, where each node sent one stream of packets for 100 seconds at rates of 15 or 30 packets/minute. CLRPL performed better regarding PDR, queue loss rate, and remaining energy in both scenarios. Yet, CLRPL had higher control messages overhead than the standard RPL.

Many more RPL research papers tackled the congestion control problem using composite routing metrics. CQARPL [129] follows the same steps as CLRPL [14] and uses similar metric combinations. Iova *et al.* designed a composite metric for multipath routing, called the Expected Lifetime metric, which is composed of RE, ETX, and the traffic rate [130]. The most common RPL metric combinations are shown in Figure 3.2, where the primary metrics that are used are: ETX, RE, QU, HC, and RSSI. The (&) sign before a metric's name in Figure 3.2 means it is combined with the metric in its parent branch.

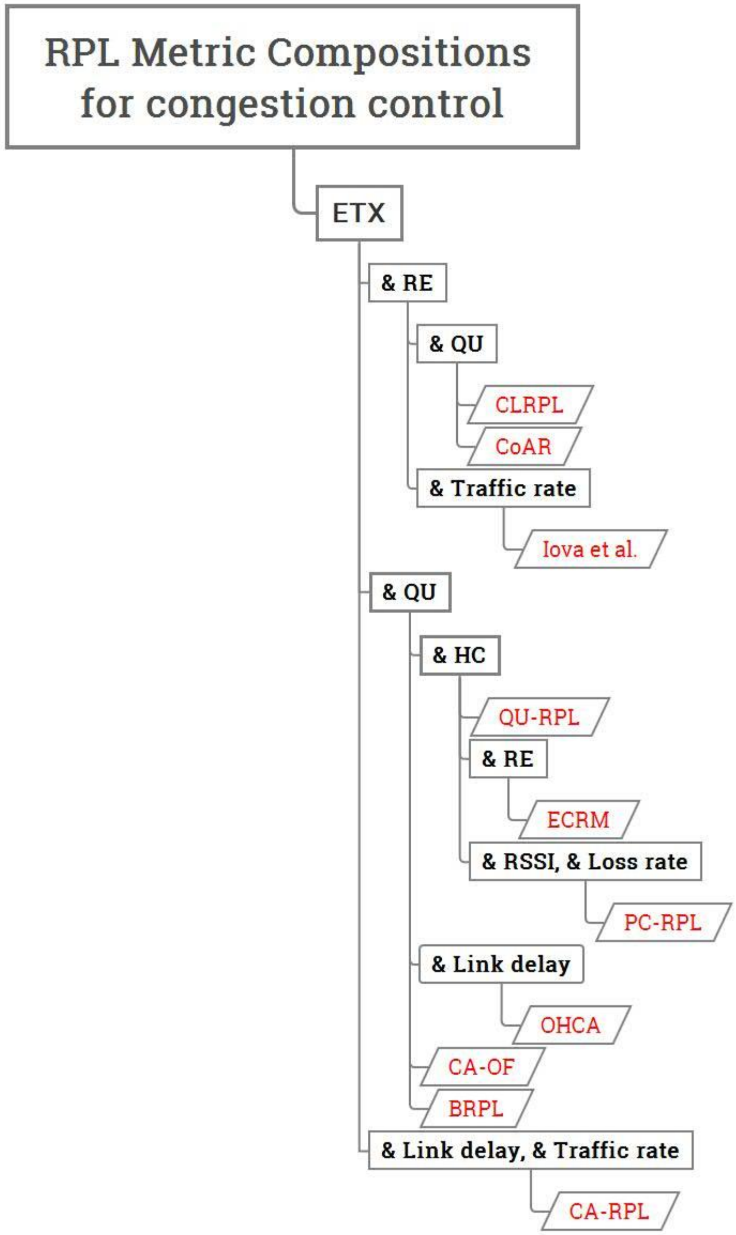


Figure 3.2: RPL congestion control schemes that use composite metrics

Table 3.1. provides an outline of the research papers surveyed in this Section.

Scheme	Contribution	Congest-ion detection metrics	Improvements	Drawbacks	Evaluation	Number of Nodes	Exploits multiple instances?
CA-OF [124]	A novel routing metric: weighted sum of BO and ETX Uses BO as an indicator of traffic rate Assigns different weight values depending on the buffer occupancy	BO and ETX	Energy consumption, PDR, throughput	Dynamic changes in the BO make the routing metric unstable	COOJA simulator	19, 35	No
CoAR [120]	Multi-criteria decision-making OF	QU and recent traffic rates	PDR, end-to-end delay, throughput, energy consumption	Complex parent selection mechanism Network stability and control messages overhead were not studied	COOJA simulator	16-41	No
GTCC [121]	A game theory-based mechanism for selecting a less-congested parent Congestion notification via DIO messages	Data arrival rate and data service rate	Throughput, packet loss ratio	DIO overhead	COOJA simulator	22, 26	No
PC-RPL [122]	Addresses the hidden terminal problem	Packet loss	PDR, parent change frequency	Calculating the reference RSSI is energy consuming	Real testbed	50	No

OHCA [116]	Hybrid mechanism of resource control and traffic control Multi-criteria OF for a less-congested parent selection	Data arrival rate, data service rate	Throughput, fairness, end-to-end delay, energy consumption, packet loss rate	Complex parent selection mechanism Network stability and control messages overhead were not studied Simulation of a small number of nodes	COOJA simulator	10, 25	No
Marco *et al.* [126]	R-metric: a reliability metric Q-metric: a load balancing metric	Data arrival rate, data service rate	Reliability, energy consumption	Evaluation with a small number of nodes	Real testbed	20	No
ECRM [118]	A composite metric of ETX, RE, and QU	QU	Average power consumption, PDR	Static congestion thresholds Evaluation with a small number of nodes	COOJA Sim.	16	No
CA-RPL [125]	A mechanism for Congestion avoidance in emergency applications Designed the DELAY_ROOT metric to minimize the delay toward the root The sender splits its traffic and sends it via the two best parents	Forwarding delay	the number of packets received by the root per second, throughput, PDR, average latency	Has a higher packet loss rate at light traffic scenarios compared to the standard RPL DIO overhead	COOJA simulator	20	No
QU-RPL [15]	Rank calculation using HC, ETX, and QU Introduced stability bound to reduce unnecessary parent changes	A static threshold	Average queue loss ratio, PDR	Chooses longer paths when the QU metric has larger weight values. Power consumption was not studied Higher overhead and higher parent changes	Real testbed	30	No

BRPL [9]	Combined RPL and backpressure routing Designed a forwarding model that is adaptable to dynamic traffic and mobility	A composite metric	PDR, throughput	Uses suboptimal and longer paths Higher delay and overhead Energy consumption and network stability were not studied	Real testbed	100	Yes
CLRPL [14]	A novel Context-aware OF A novel Context-aware composite routing metric Includes QU, energy, and ETX in rank calculations and parent selection	A composite metric	PDR, RE, queue loss rate	High overhead	COOJA simulator	50-100	No

Table 3.1: Summary of the main RPL improvements for congestion control

3.3. Multi-instance RPL

Even though we are essentially interested in schemes for cooperation among RPL instances, we'll examine all research efforts to enhance RPL's performance when multiple instances coexist, even the ones that do not rely on inter-instance cooperation.

Long *et al.* [131] aimed to enable efficient delivery of critical data packets in multi-instance RPL networks by introducing a QoS-aware solution based on a cross-layer procedure. They study a network model with two classes of nodes: regular nodes that generate periodic traffic at low rates, and alarm nodes that generate sporadic traffic at high rates. In their design, alarm nodes can forward their traffic via regular nodes, but regular nodes cannot send packets to alarm nodes.

Their MAC-layer traffic differentiation process gives a different buffer priority to each of the two traffic types at the forwarding nodes. Periodic packets are given lower buffer priority than alarm packets and are dropped in their favor if congestion occurs. Simulations with COOJA showed that the proposed model improved PDR and end-to-end delay for alarm packets, but reduced PDR slightly for regular packets compared to the standard RPL.

The RPL improvement proposed by Long *et al.* is based on a network with one root and two instances, where each instance is formed of two subgraphs (DODAGs). Both instances in their paper are identical and have the same OF and routing metrics. Thus, Long *et al.* did not actually exploit the multi-instance feature of RPL since their traffic prioritization takes place at the DODAG level rather than the instance level, i.e., within the same instance and not between two different instances.

Rajalingham *et al.* [132] analyzed RPL in the NAN segment of the Smart Grid, where multiple applications coexist, each with its own QoS constraints. They studied an RPL network with two traffic classes for two AMI applications:

- Collecting Smart Meter readings: This application is used for billing. It has low-priority periodic traffic, requires medium reliability, and can tolerate delays of up to 15 seconds.
- Reporting alert messages: These messages report the grid operating conditions. They are classified as critical and high-priority AMI data that require high reliability and low delay of about 200-300 milliseconds (ms).

Rajalingham *et al.* established MAC-layer traffic differentiation by changing the CSMA backoff values according to the traffic class. Shorter backoff periods were given to alarm messages compared to the periodic ones, which means that alarm traffic had to wait less to be retransmitted in case of a collision.

As to the traffic prioritization at the network layer, a two-instance RPL scheme was suggested where each instance supports a different traffic class. An alarm instance that employed a reliability metric (ETX) was used for routing alarm traffic, while Hop Count was used in the second instance that is dedicated to periodic traffic.

OMNET++ simulator was used by Rajalingham *et al.* to evaluate their proposal against a standard RPL implementation with ETX metric. Regarding delay and PDR, results showed that the proposed scheme did not perform much better overall at low data rates. At high data rates of 0.1 packets/second, the proposed model without the backoff mechanism performed the best. However, integrating the shorter backoff periods for alarm traffic in this scenario led to higher delays and less PDR for the proposed model. This is understandable since granting nodes faster channel access would increase the chance of collisions in heavy traffic conditions. The problem with all these evaluations is that they used IEEE 802.11b links with transmission rates up to 11 Mbps, which are incompatible with LLNs.

Aiming to meet different routing requirements for heterogeneous applications in SG, Nassar *et al.* [133] introduced a new RPL OF called Objective Function for Quality of Service (OFQS). It evaluates a multi-purpose routing metric (mOFQS) that weighs delay, link quality, and battery state for parent selection. Nassar *et al.* evaluated their proposal using the COOJA simulator. Their results showed that their model performed slightly better than Contiki-RPL regarding PDR but showed noticeable delay and energy consumption improvements. The problem with OFQS is that it may select longer and less reliable paths when performing load balancing. Later, Nassar *et al.* extended their work by using a real testbed for evaluating it [7].

The authors of [134] use RPL instances to include QoS differentiation in the RPL routing strategy. They use three QoS classes to support the requirements of different application classes, each with its own OF. With this, the paper provides a scheme to create a QoS-aware LLN for IoT applications with conflicting requirements. Nevertheless, they introduce a rather complex parent selection mechanism.

Mardini *et al.* [135] evaluated implementing multiple RPL instances in a healthcare system where the sensors generate multiple traffic types ranging from periodic to highly critical. They did not introduce a new model but just performed a direct implementation of RPL with four instances and compared it with a single-instance RPL.

Considering industrial monitoring applications, the work of [136] designs four RPL instances to support four traffic classes. They introduce four routing metrics and use combinations of 3 or 4 of them to create composite metrics for three instances, while the fourth instance relies on remaining energy as a metric. Their results showed lower delay and higher PDR for critical traffic.

Barcelo *et al.* introduced C-RPL [10], which enables RPL instances to cooperate by forwarding traffic for each other. In this model, instances share their nodes using coalition game theory to create a "grand coalition" that forwards their traffic to the root. However, the game was designed to take place at the root, i.e., each node must send information about its nearby nodes and all possible cooperation outcomes to the root. After receiving every possible cooperation outcome from every node in the network, the root decides how coalitions are formed. The evaluation results for C-RPL showed enhancements of PDR and delay compared to the standard RPL. However, these evaluations cannot be properly assessed since they used MAC and physical layers incompatible with LLNs.

Junior Seidni *et al.* designed DYNASTI [137], a multi-instance scheduling mechanism for regular and sporadic Smart Grid applications. The primary assumption in DYNASTI is that all nodes belong to all instances. Depending on the type of application running on the network, the root can schedule suitable instances. When a monitoring application is running, the root schedules a "normal instance" relevant to routing regular traffic. When a node has critical data or an alarm to report, it notifies the root, which schedules a "sporadic instance" that is more suitable for routing critical traffic. Junior Seidni *et al.* analyzed the performance of DYNASTI using the COOJA simulator. Their simulations used low-rate traffic and showed that their model performed better than the standard RPL regarding energy consumption and delay. However, DYNASTI was not evaluated under heavy traffic, where requests from nodes to change the instance might not reach the root. Another drawback in the design of DYNASTI emerges when the root changes the instances in the network. At that moment, all the traffic belonging to the instances that are not scheduled anymore has to be dropped. The effects of such a packet-dropping process were not studied. Finally, DYNASTI was evaluated in a setting where all instances use the same metric.

Table 3.2 outlines the multi-instance RPL research papers discussed in this Section.

Scheme	Metric(s)	Sending rate	Improved	Drawbacks	evaluation	number of instances	Number of nodes	Cooperation among instances?
DYNASTI [137]	ETX	1 packet every 3/5/15 minutes	Consumed energy, delay	Did not investigate the effects of dropping instances Heavy traffic was not considered All instances used the same metric	COOJA	3	Not provided	No
Nassar *et al.* [7]	ETX, RE, delay	1 packet every 1-60 seconds	RE, delay	Path stretch	Testbed	3	67	No
Nassar *et al.* [133]	ETX, RE, delay	1 packet every 3-4 minutes	RE, delay	Chooses less reliable paths	COOJA	3	35	No
C-RPL [10]	Rank and RE	1 packet every: 5, 10, 100 seconds 1-5 packets/sec	RE, PDR, fairness	Uses MAC and physical layers not supported by LLNs	Matlab	2	120	Yes
Bhandari *et al.* [134]	Delay, ETX, QU, RE	15-150 packets/min	Delay, PDR	Routing overhead Was not compared to the standard RPL	COOJA	2	300	No

Long *et al.* [131]	Buffer priority	Periodic: 1 packet/min Alert: 1.5 packets/sec	PDR and delay for alarm data	Uses identical instances	COOJA	2	35 (4 alarm nodes and 31 regular nodes)	No
Rajalingham *et al.* [132]	At the MAC layer: backoff periods At the network layer: ETX, HC	Various rates up to 0.1 packets/sec	Delay and PDR (at high data rates)	Evaluated using links and transmission rates not compatible with LLNs	OMNET++	2	1000	No

Table 3.2: Summary of the research on RPL with multiple instances

3.4. Cooperation promotion using virtual credits

In the past, various approaches have attempted to use virtual currencies in the context of promoting cooperation among communicating nodes.

Carels *et al.* [138] explored the idea of using multiple sinks (roots) with RPL in a way compatible with its standard protocol description. Following a suggestion in RFC 6550 [4], which specifies RPL, the authors investigate the usage of virtual DODAG roots in disparate locations within the network. While these nodes act as sink nodes, towards the rest of the network, they pretend to be regular nodes of the network, that are one hop away from a virtual root node. It is found that adding additional sink nodes to the network can decrease the average energy consumption by 30-50%.

One of the foundational works in the field of using virtual currencies to promote cooperation in self-organized networks was published by Buttyan *et al.* in 2001 [139]. The authors propose a virtual currency called Nuglets for usage in Mobile Ad hoc Networks (MANETs). Any packet forwarding is paid for by using these Nuglets, thereby incentivizing nodes to cooperate with other nodes to earn Nuglets, which they can then use to send packets. Two main modes of payment are discussed: one in which packet sources are charged and another in which destinations are charged. The possibility of a hybrid payment model is also discussed. However, due to being originally designed for MANETs, this scheme is not well optimized for LLN-IoT scenarios using RPL, where traffic mainly flows toward sink nodes.

Building upon their previous work, Buttyan *et al.* revisited the topic in 2003 [140]. A Nuglet counter is added to every packet. It is protected from manipulation through the use of a tamper-resistant security module built into the nodes. Through the addition of this module, the overall scheme becomes more resilient against attacks. Still, it also requires additional hardware, which is unlikely to be present on most, especially low-cost, IoT devices.

Zhong *et al.* [141] introduced Sprite. This scheme is designed for MANETs and aims to stimulate cooperation among mobile nodes through a credit-based system that does not require tamper-proof hardware for any node. To facilitate this, nodes with fast connections coordinate with a centralized Credit Clearance Service (CCS), which handles charging and crediting nodes for transmissions in the network, depending on how their receipt was reported.

A similar approach to solving this issue using virtual currencies was proposed by P. Das *et al.* [142] in 2012. The authors introduce the Credit Based Routing (CBR) algorithm, which ensures high message passing rates while minimizing message replications in Delay-Tolerant Networks (DTNs). The credit used here is a value assigned to a node once it has a connection to another node. However, CBR is unsuitable for IoT applications that are sensitive to packet delays.

R. Aslani *et al.* [143] introduced a dynamic Token-Based Incentive Mechanism, called TOBIM, for P2P networks in 2017. In this system, each peer needs to spend tokens for sending its own packets, and it can also earn tokens by forwarding other peers' packets. According to the dynamics of the request arrival, demand submission, and bandwidth availability processes, TOBIM adapts the admission control policy of peers based on a Constrained Markov Decision Process (CMDP). In general, TOBIM is designed to tackle the free-riding problem in Peer-to-Peer video streaming networks, and the required, relatively heavy computational tasks at each peer make it not applicable in many IoT scenarios.

In 2018, M. M. Umar *et al.* [144] introduced a game theoretic reward-based system to stimulate or/and punish selfish nodes in static WSNs. In this system, a Rubinstein-Steel bargaining game is applied, and virtual money (called score) is used by nodes during their cooperation. Since it is a centralized design, the base station needs to monitor all nodes in the network and update the appropriate parameters for each node regularly. In RPL-based IoT systems, additional communication overhead between the root and its nodes should be avoided for efficiency reasons.

3.5. Discussion

Our research domain is an intersection between two RPL research domains: congestion control and cooperation among different RPL instances. Our literature review shows that there is no paper that explores this domain. However, we found that C-RPL [10] and BRPL [9] are the closest to our work, so we will discuss them in detail after discussing the approaches provided in Sections 3.2 and 3.3. We will also cover the cooperation paradigms that use virtual coins.

3.5.1. Limitations of the congestion control schemes

The problem with RPL's models that use composite routing metrics to reduce congestion (listed in Section 3.2.4) lies within the composite metric itself. Applying such a solution in a network with multiple RPL instances requires that all instances use the same proposed composite metric for routing. This prevents establishing differentiated services and goes against the purpose of using multiple instances to begin with. The reason for using a multi-instance architecture in RPL networks is to have the freedom to use different routing metrics in each instance.

The composite metrics suggested by the papers summarized in Figure 3.2 can indeed reduce congestion. However, in an IoT system such as an SG, which incorporates multiple applications with different QoS demands, a single composite metric cannot meet all their requirements. While our approach to fulfilling such requirements relies on establishing multiple instances, each with its dedicated routing metric(s).

Interoperability should be considered when designing a composite routing metric for RPL so that it is compatible with other RPL metrics; otherwise, it prevents RPL instances from cooperating with one another. Interoperability was not considered in any of the papers we surveyed in Table 3.1., which means cooperation among RPL instances is not possible in these proposed models.

The composite metric approach also introduces instability to the network. A compound metric can become unstable if a slight change in one of its inputs causes a change in its value. If such small changes persist, they can cause an increased frequency of parent changes and trigger what is known as the Thundering Herd Phenomenon [14]. Table 3.1. demonstrates the RPL schemes that suffer from instability.

Another issue with combining routing metrics is that it leads to relatively large DIO sizes. As explained in Section 2.3.2, the maximum IEEE 802.15.4 frame size is 127 Bytes. 25 and 40 Bytes of it are used by the MAC and IPv6 headers, respectively. As illustrated in Section 2.8.6.1, assuming a DIO message with a PIO and a DODAG Configuration Option, its corresponding size would be:

- DIO header: 4 Bytes.
- DIO base: 24 Bytes (including a 16 Byte DODAG_ID).
- DODAG Configuration Option: 16 Bytes.
- PIO: 24 Bytes.

Moreover, assuming we use a composite routing metric of throughput, latency, and ETX, the size of the DIO metric container would be:

- Metric container for throughput: 6 Bytes.
- Metric container for latency: 6 Bytes.
- Metric container for ETX: 4 Bytes.

All these aforementioned values add up to 84 Bytes. This shows that the composite metric approach increases overhead by increasing the sizes of the DIO messages. It can also cause more overhead if a DIO message can not fit within one link-layer frame and has to be fragmented and reassembled at each node.

3.5.1.1. Limitations of BRPL

Of all the surveyed papers on congestion control in RPL networks, only BRPL [9] employs a multi-path approach that exploits multiple instances for congestion alleviation. BRPL assumes that alternative paths via other instances exist and can be used in case of congestion. One problem of BRPL is that it is designed for networks where all instances are similar and have the same configuration. BRPL treats other instances merely as alternative routes, not considering that they may have different configurations and routing policies. Therefore, service differentiation is not possible in BRPL since it has identical instances.

Additionally, in BRPL, any node can forward its packets via other instances without restrictions. It neither implements a cooperation mechanism among instances nor considers the node's selfishness in this case.

3.5.2. Limitations of the centralized and reactive inter-instance cooperation schemes

We have surveyed the research papers where RPL networks with multiple instances are studied. The majority of those papers do not rely on cooperation among instances, but rather they aim for an overall performance improvement by improving the performance of each instance individually. For that purpose, they implement different routing metrics at different instances or establish service differentiation policies at the network and link layers. C-RPL [10] is the only paper we found where cooperation among RPL instances was performed. C-RPL used a novel centralized and reactive mechanism for that.

As we have explained before, the majority of RPL's traffic is in the upward direction. In centralized cooperation schemes like C-RPL, leaf nodes send cooperation requests to the root (upward direction) and wait for its reply (downward direction). For the root to be able to send replies to the leaf nodes, a downward routing mechanism needs to be employed. As explained in Section 2.8.2, this mechanism can be performed in one of two ways:

- Storing mode: Each intermediate node keeps a routing table in the downward direction to be able to relay packets from the root to its children. The more children a node has, the more routes it needs to save, and the more memory it uses. Due to the limited memory capacity of the sensor nodes, this method is resource-consuming and is not scalable.

- Non-storing mode: The leaf node attaches a source routing header to its request. The root uses this header to route its reply in the downward direction. The issue with this method is the incurred overhead from the size of these source routing messages, which get bigger the further the leaf node is from the root. Furthermore, according to Aishwarya *et al.* [145], in this mode, the destination can not be more than eight hops away from the source when IPv6 header compression is not used, and 64 hops away from the source when IPv6 header compression is used to compress IPv6 addresses.

In both cases, downward routing in RPL is not scalable and causes additional overhead. Besides, it is not guaranteed that cooperation requests or replies could reach their destinations in case of heavy traffic and path congestion. Moreover, the reactive process of sending cooperation requests and waiting for cooperation confirmation from the root is not suitable for alarm and time-sensitive applications.

3.5.2.1. Limitations of C-RPL

We focus on C-RPL [10] since it is the only paper from Table 3.2 that uses inter-instance cooperation. C-RPL suffers from the abovementioned drawbacks of using a centralized and reactive inter-instance cooperation mechanism. In addition, C-RPL neither considers congestion nor aims to mitigate it. Furthermore, it used Matlab for its evaluations and employed a link layer not compatible with LLNs. It is not clear if some core RPL functionalities like Trickle are supported in Matlab. Contiki-OS is the most used software for evaluations in RPL literature because it supports most RPL components. Dubrulle *et al.* describe ContikiRPL as "the most mature and complete RPL implementation to date." [146]. Therefore, it is not clear which RPL mechanisms and link layer were implemented in C-RPL, which makes judging its results difficult.

3.5.3. Limitations of the virtual credit systems

In principle, all approaches mentioned in Section 3.4 could be applied in RPL-based IoT systems. However, the requirements of such IoT systems are different from what these approaches were originally designed to fulfill. For example, in RPL-based IoT networks, most communications take place in a one-directional way, with nodes sending sensed data toward the root (upward direction) [12]. This makes credit-based approaches that require additional communication in the opposite direction inefficient. Moreover, some of these approaches require P2P communications. Building and maintaining P2P routes in RPL is complex and resource consuming.

Additionally, there may be multiple instances with different QoS requirements that basically act as separate networks, among which, it would be beneficial to allow cooperation. We follow an approach that specifically addresses the need for LLNs employing the RPL protocol to facilitate communications in situations with low-powered nodes and IoT scenarios like, for example, Smart Grids in Smart City environments. None of these credit-based, previous works address this use case of growing importance.

3.6. Summary

Since the RPL standard does not address congestion control, we have explored the research that tackles this issue. We concluded that the research community did not design any RPL congestion control scheme that utilizes cooperation among multiple instances.

We highlighted the major limitations of the composite routing metric approach in mitigating congestion, which are: overhead, instability, and lack of interoperability.

We have also surveyed the research papers regarding cooperation among multiple RPL instances. We found only one paper that directly addresses this issue, namely C-RPL [10]. We have listed the drawbacks of using a reactive and centric approach for cooperation management in C-RPL, which are: the overhead caused by downward routing and the susceptibility of cooperation requests and replies to high packet loss rates in case of congestion.

In the next Chapter, we introduce our novel protocol design that aims to tackle these challenges.

We also reviewed the incentivizing mechanisms to encourage cooperation and listed their application domains. In Chapter 6, we design a framework that uses the concept of virtual currency to promote cooperation in RPL-based IoT networks.

4. DMC-RPL

In this Chapter, we design a Distributed Multi-instance Cooperative RPL (DMC-RPL), which is a novel IoT routing protocol based on RPL. Our proposed routing protocol introduces a congestion control mechanism for IoT networks that use multiple routing instances for service differentiation. DMC-RPL follows the resource control approach for congestion mitigation by utilizing the presence of other instances to exploit path diversity. This enables forwarding the traffic of a congested instance via other instances to reduce congestion. DMC-RPL also employs a novel distributed cooperation scheme among its instances. The design of DMC-RPL aims to answer the research questions the first three research questions presented in Section 1.3 (RQ1, RQ2, RQ3) and tackle the shortcomings of the RPL-based routing protocols listed in Section 3.5.

Table 4.1 outlines the primary symbols used in this Chapter.

Symbol	**Definition**
DMC-RPL	Distributed Multi-instance Cooperative RPL
Q_t	The node's queue length (the number of packets stored in its Queue buffer waiting to be transmitted) at a time t
Q	The queue length at a neighbor node. A node learns the Q values of its neighbors from their DIO messages
BO	Buffer Occupancy. Another term for queue length. Both can be used interchangeably
Q_{p_est}	The parent's estimated queue length
Q_{pth}	The estimated congestion level of the nodes on the path to the root
Q_{ev}	A congestion estimation metric used by a node for creating a congestion evaluation of its path to the root based on its Q_t and Q_{pth}
GP	Grandparent
IQR	Inter-Quartile Range
Q_1 , Q_3	The first quartile, the third quartile
IQR_{sc}	*IQR* scaled to Q_{max}
Q_{max}	The maximum buffer capacity, which is also known as the maximum queue length. It is the maximum number of packets a node can store in its Queue buffer

Med	The median
Med_{est}	The estimated median value
Med_{est_gp}	The estimated median queue length of the Grandparent chain
DI	Dynamic-traffic Index

Table 4.1: The main symbols used by DMC-RPL

4.1. Design principles

Based on our analysis in the previous Chapter, we define the following principles that we rely on for designing DMC-RPL's mechanisms.

Design principle #1: Interoperability

For nodes from different instances to cooperate, each node should be able to evaluate other paths via other instances. This assessment is essential for a leaf node to decide if a path via another instance is more optimal than its own. Such evaluation is not possible if instances use different routing metrics, which is the case in networks that support service differentiation, where instances have different routing metrics, OFs, and routing configurations.

To solve this problem, we need to rely on a routing metric that can be understood and assessed by LLN nodes regardless of their instances or routing setup. For this, we use the queue length metric (Q), assuming that all leaf nodes in an LLN have the same maximum buffer capacity (Q_{max}). This assumption is made in many RPL papers [9, 14, 116]. DMC-RPL relies on the (Q) metric to calculate a path-congestion estimation, which is used by a leaf node to evaluate paths via other instances.

Design principle #2: path-congestion estimation at the leaf nodes

To enable autonomous decision-making, a leaf node from one instance should be able to decide on its own whether to accept cooperation and forward packets from leaf nodes of other instances. The challenge in this distributed approach is that a leaf node has limited information about its network, whereas centralized cooperation schemes rely on the root, which has a view of the entire network.

A problem can emerge if a leaf node accepts forwarding packets from other instances and causes congestion in its own instance in the process. This limited view of the network by the leaf nodes is enhanced in DMC-RPL using a novel metric for estimating the congestion level of the path to the root. This metric is used by the sending and forwarding node as follows:

- A sending leaf node estimates the path congestion in its instance and decides whether to initiate a cooperation process to forward its packets via another instance.
- A forwarding leaf node receives cooperation requests from leaf nodes of other instances. Using our proposed metric, it estimates the congestion level of its path to the root and decides if it can forward packets from other instances without incurring congestion on its own.

Design principle #3: Network stability

As discussed in Section 3.5.1, using multiple metrics in an RPL network can cause topology instability since it increases the probability that the nodes change their preferred parents more frequently.

In DMC-RPL, each instance implements its local routing metric(s), which is used to select the best parent and, thereby, find the optimal intra-instance path to the root. In addition to the local metric(s), the queue length metric (Q) is used at every instance as a global metric for evaluating paths via other instances. To avoid topology instability issues in DMC-RPL, the Q metric is not included in the process of selecting the best intra-instance path. In other words, each node uses the local metric(s) at its instance to calculate its rank, select its best parent, and select its best path to the root. While the Q metric is used for congestion detection within the instance, and congestion estimation of inter-instance routes.

4.2. Congestion estimation in DMC-RPL

LLNs are highly dynamic networks; it is not always possible for their leaf nodes to accurately estimate the conditions of long paths. Some routing metrics can be used in LLNs to evaluate the conditions of a path to the root, such as ETX [52], which is used to estimate a path's reliability. Still, they can be inaccurate as the state of the nodes and links in LLNs can vary with time.

We follow such a path estimation approach using the (Q) metric. However, due to the dynamicity of LLNs, we divide the path estimation into two parts:

- Estimating the parent's congestion: The parent is the first node on the path to the root. Its updates are received more frequently than the subsequent nodes on that path, and therefore, its (Q) values are more up-to-date.
- Estimating the congestion of the Grandparents (GPs): A node's Grandparents are all the nodes on its path to the root, excluding its parent. We use the (Q) values of the GPs to provide a rough estimate of their path congestion.

We use this two-part assessment method for congestion detection in DMC-RPL. We use the term "ancestors" to refer to all the subsequent nodes on the path to the root, i.e., the parent and the Grandparents. Figure 4.1 illustrates a leaf node, n_1 of instance 1, and the classification of the nodes on its path to the root.

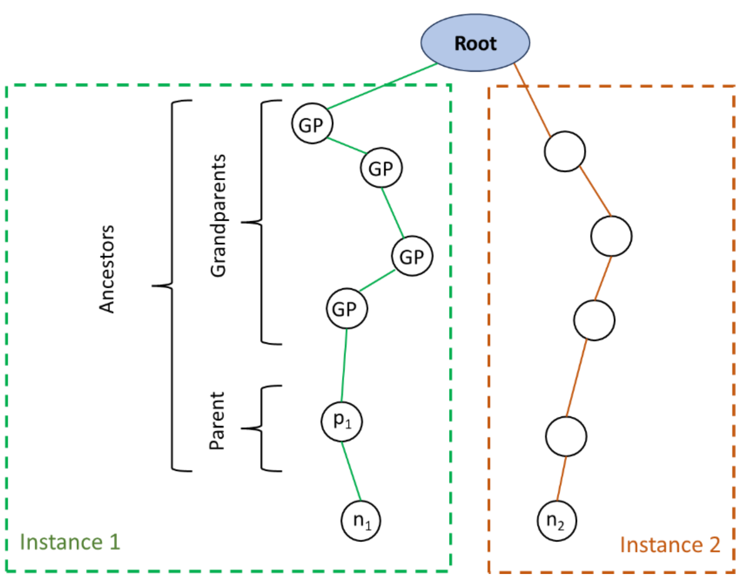


Figure 4.1: Representation of a node's ancestors in DMC-RPL

4.2.1. Grandparent-chain buffer occupancy estimation

A single-hop metric is an estimation of a certain value of a neighboring node, e.g., link quality, delay, etc. On the other hand, a path metric is calculated using its values that are reported by all the nodes on that path. A path metric is updated at each hop and can be either aggregated or recorded [52]. A recorded metric is represented by a series of values (DIO sub-objects) that are the metric's measurements at the ancestor nodes. When a recorded metric is used in a DODAG, each node adds its local estimation of the metric as a new element to the series and sends it in its DIO messages.

To facilitate distributed collaboration amongst instances, the leaf nodes must overcome a critical challenge: Unlike the root, the leaf nodes have a limited view of the network and are unable to assess locally whether cooperating and forwarding traffic from another instance could cause more congestion in their own instance.

To solve this issue, we utilize the DIO messages. Since DIO messages can carry path metrics, nodes can use them to pass information about their state to their descendants. This information from ancestors can then be used by a leaf node to evaluate its path to the root and decide, on its own, whether to cooperate with nodes from another instance.

DMC-RPL uses the Q metric as a recorded metric. This means that each DIO message a node receives from its parent has all the queue length values of its ancestor nodes. Since we separate the congestion estimation of a node's parent from its Grandparents, the Q value of the parent will be used separately as we will explain later. The remaining Q values a node receives in a DIO message, i.e., the Q values of the Grandparents, are stored locally in an array called the Q values array. DMC-RPL views the Grandparents as a chain of nodes on the path to the root, and relies on their Q-array for estimating the buffer occupancy of the chain.

DMC-RPL's estimation relies on various tools and methods from statistics, which are referred to as methods of descriptive statistics [147]. In DMC-RPL, the median value of the Q-array is used to estimate a Q value of the Grandparent chain. We prefer to use the median instead of the mean since it is more resilient to outliers, as this example demonstrates:

Example 1: Suppose we have the following Q values: 2, 4, 3, 1, 5.

Both the mean and median of these values are equal to 3. However, if we change the second Q value from 4 to 7, the mean will become 4, but the median will remain the same. This shows that a small change in one of the values did not affect their median. This property of the median is valuable for our estimation since LLNs are dynamic, and the Q values of their nodes fluctuate constantly. We aim for a Q estimation of the Grandparent chain that does not get easily affected by temporary changes of the Q values of few nodes in the Q-array.

In some cases, multiple paths may have equal Q medians, such as in this example:

Path	**Q values**	**Ordered Q values**
P_1	0, 20, 5, 6, 3, 8, 1	0, 1, 3, 5, 6, 8, 20
P_2	16,1, 3, 20, 15, 0, 5	0, 1, 3, 5, 15, 16, 20

Table 4.2: Example 2

After ordering the values in Example 2, we can see that the paths P_1 and P_2 have the same median, although P_2 is more congested than P_1. This proves that the median alone is not sufficient to evaluate different Q-arrays with acceptable accuracy. To enhance the accuracy of our median-based Q estimation, we integrate the Inter-Quartile Range (IQR) in our calculations. The IQR is a measure of the spread of 50% of the data in a set with regard to the median. It is measured as the distance between the third quartile (Q_3) and the first quartile (Q_1), as seen in Figure 4.2. In such representation, 25% of the data is below Q_1, 25% of the data is above Q_3, and 50% of the data is between Q_1 and Q_3. The IQR shows how this 50% of the data is spread above and below the median. The smaller the IQR, the higher the median's accuracy in representing the data, since a small value of IQR suggests that the data values in the range ($Q_3 - Q_1$) are close to the median.

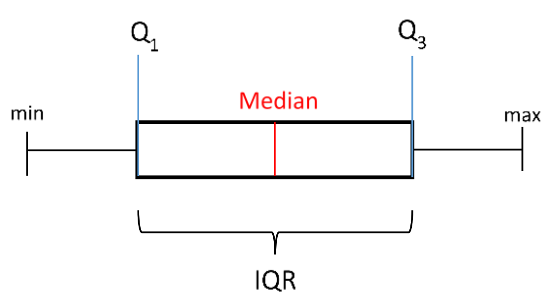


Figure 4.2: Boxplot representation of the median and the IQR

In DMC-RPL, We take the Q-array of the Grandparents and calculate the *IQR* of its values (IQR_Q).To make the value of IQR_Q more relevant, we use it with regard to the maximum buffer size. Since we assume all nodes in the network have the same buffer capacity, we scale IQR_Q to the maximum buffer size Q_{max} as follows:

$$IQR_{sc} = \frac{IQR_Q}{Q_{max}} \tag{4-1}$$

Where IQR_{sc} is the scaled *IQR* of the Q-array values.

Instead of relying solely on the median for evaluating the Q values of a GP chain, we include the *IQR* in our estimation, as we can see in equation (4-2):

$$Med_{est\_GP} = Med_{GP\_Q} + (Med_{GP\_Q} * IQR_{sc}) \tag{4-2}$$

Where: Med_{GP_Q} is the median of the Grandparents' Q-array values; Med_{est_GP} is the estimated median value of that Q-array.

Returning to Example 2 in Table 4.2, we can compare P_1 and P_2 by computing their Med_{est_GP} values, which we call Med_{est_P1} and Med_{est_P2} respectively. Assuming $Q_{max} = 20$, our evaluation of these two paths is as follows:

P_1: 0, 1, 3, 5, 6, 8, 20	P_2: 0, 1, 3, 5, 15, 16, 20
$Med_{P_1\_Q} = 5$	$Med_{P_2\_Q} = 5$
$Q_1 = Med\ (0,1,3) = 1$	$Q_1 = Med\ (0,1,3) = 1$
$Q_3 = Med\ (6,8,20) = 8$	$Q_3 = Med\ (15,16,20) = 16$
$IQR = Q_3 - Q_1 = 7$	$IQR = Q_3 - Q_1 = 15$
$IQR_{sc1} = \frac{IQR}{Q_{max}} = \frac{7}{20} = 0.35$	$IQR_{sc2} = \frac{IQR}{Q_{max}} = \frac{15}{20} = 0.75$
$Med_{est\_P_1} = Med_{P_1\_Q} + (Med_{P_1\_Q} * IQR_{sc1}) = 5 + (5 * 0.35) = 6.75$	$Med_{est\_P2} = Med_{P_2\_Q} + (Med_{P_2\_Q} * IQR_{sc2}) = 5 + (5 * 0.75) = 8.75$

We can see above that $Med_{est\_p1} < Med_{est\_p2}$ .Therefore, P_1 is evaluated as a less congested path than P_2.

The estimated median ($Med_{est_{GP}}$), computed using equation (4-2), is the metric a node uses in DMC-RPL to evaluate congestion in its Grandparent chain. It is used for assessing the path's congestion together with another metric that estimates the parent's congestion, which we introduce next.

4.2.2. Estimating the parent's buffer occupancy under heavy and dynamic traffic

As mentioned in Section 1.4, congestion in LLNs can take place when nodes generate heavy or dynamic traffic. That is why we focus on designing a metric for estimating the parent's congestion level under heavy and dynamic traffic.

A node can capture the effects of dynamic traffic on its parent more feasibly than its other ancestors; since a parent is a neighbor node that is one hop away, where updates carried within its DIO messages are received more often.

It is crucial to consider the effect of dynamic traffic on buffer occupancy since it can cause it to fluctuate rapidly when a node is generating or forwarding dynamic traffic. For this reason, we consider the dynamicity of the parent's buffer occupancy in DMC-RPL. We design a Dynamic-traffic Index (DI) to estimate the level of congestion caused by dynamic traffic. DI is based on statistical variance, where a high variance in the queue length at a node suggests that it is either generating or forwarding non-periodic traffic.

Let Q_t be the node's queue length at a time t. The variance of this value ($var\ Q_t$) is given as:

$$var\ Q_t = \frac{1}{m}\sum_{i=1}^{m}(Q_i - \bar{Q})^2 \qquad (4\text{-}3)$$

Where: Q_i is the value of Q_t at a previous measurement (Q_{t-1} , Q_{t-2} , $Q_{t-3}\ldots$), and $\bar{Q}$ is the mean value of these measurements. In other words, $\bar{Q}$ is the mean of the previous (m) measurements of a node's queue length, as equation (4-4) shows:

$$\bar{Q} = \frac{1}{m}\sum_{i=1}^{m}(Q_i) \qquad (4\text{-}4)$$

In this Thesis, and due to the limited memory capacity of the sensor platform we use, we set $m = 4$. This means we evaluate the variance of the buffer length using its past four values. In DMC-RPL, the buffer length values are measured every 10 seconds. We chose this interval to be compatible with heavy traffic since a heavy traffic rate is defined as a rate of one packet every 2 seconds or higher, according to CLRPL [14].

Our proposed DI is calculated by scaling $(var\ Q_t)$ to (Q_{max}), which limits its value to the range [0,1]:

$$DI = \frac{var\ Q_t}{Q_{max}} \qquad (4\text{-}5)$$

When LLN nodes transmit data at high rates, it was observed that the state of a node's Queue buffer was a significant indicator of its packet loss rate, since most of the lost packets, in this case, were packets that were dropped due to buffer overflows [124]. Since heavy traffic is an important factor in congestion, we rely on the parent's queue length (Q_p) for estimating the parent's congestion level.

The congestion caused by heavy and dynamic traffic is estimated using Q_p and DI. A child learns these two values from its parent's DIO messages, which are broadcast periodically. The parent's estimated queue length (Q_{p_est}) is calculated as follows:

$$Q_{p\_est} = Q_p + \left(DI * Q_p\right) \qquad (4\text{-}6)$$

It is worth mentioning that CLRPL [14] uses an index close to our proposed DI, but there are two main differences between them:

- CLRPL scales the variance of its index to $(Q_{max})^2$, which produces very low DI values that are not very sensitive to changes in the buffer size. Compared to that, DMC-RPL scales its DI to a much smaller value, which is Q_{max}.

- In CLRPL, the DI is calculated by the child nodes, where each node records the previous broadcasted Q values of its parent, then calculates their variance and DI. Due to the lossy nature of LLNs, it is not guaranteed that the periodic Q values of a parent will always be delivered. If some Q values are not received by a child, then the calculation of their variance and DI in CLRPL will neither be accurate nor up-to-date. This is crucial in dynamic traffic scenarios, where precise, up-to-date DI values are essential. In DMC-RPL, the parent's DI is calculated by the parent itself, not its children. By constantly monitoring the queue length at its buffer, each parent node periodically calculates its DI and broadcasts it to its children. This is a better approach than CLRPL because even if a child disconnects for a while and does not receive the recent Q values of its parent, it can still receive accurate and up-to-date DI values from its parent as soon as it reconnects again.

Our proposed metric for estimating the parent's queue length accounts for the congestion caused by heavy and dynamic traffic and is used in the calculations of the path's congestion level.

4.2.3. Path-congestion estimation

Using the metrics given in (4-2) and (4-6), a DMC-RPL node can estimate its ancestors' congestion level as follows:

$$Q_{pth} = \alpha \left(Q_{p\_est}\right) + (1-\alpha)\,(Med_{est\_GP}) \tag{4-7}$$

Where: Q_{pth} is the estimated congestion level of the path to the root; $\alpha \in [0,1]$ is a weighting factor.

The final step of congestion estimation in DMC-RPL includes evaluating the node's buffer occupancy (Q_t) and its path's congestion level (Q_{pth}). Such an estimation is performed as follows:

$$Q_{ev} = \beta\,(Q_t) + (1-\beta)\,(Q_{pth}) \tag{4-8}$$

Where: $\beta \in [0,1]$: is a weighting factor; Q_{ev} is the overall congestion level of a node, including the congestion levels of its buffer and its path to the root.

In many congestion detection models, congestion is detected if the queue length at a node exceeds a certain threshold [15]. We follow this approach by using Q_{th} as the threshold queue length. Q_{th} is a fixed value that we set it in our implementation to 80% of Q_{max}. Using equation (4-8), a node calculates Q_{ev} to evaluate its buffer state and the congestion level of its path to detect congestion. If $Q_{ev} > \; Q_{th}$ then congestion is detected.

Q_{ev} is the distributed congestion estimation metric we propose for congestion detection. If congestion is detected by a node, it initiates a cooperation process for congestion alleviation by asking its neighbor nodes from other instances to forward its packets. Those nodes follow that same evaluation process by calculating their own Q_{ev} values to decide whether they can accept cooperating with the requesting node without causing congestion in their instances.

In DMC-RPL, a certain free space of the node's buffer is considered occupied according to the congestion level of its path. Figure 4.3 demonstrates the concept of using Q_{ev} for congestion detection, where the weights of Q_{ev} are not included to simplify the illustration. Figure 4.4 displays the whole congestion estimation process using Q_{ev}.

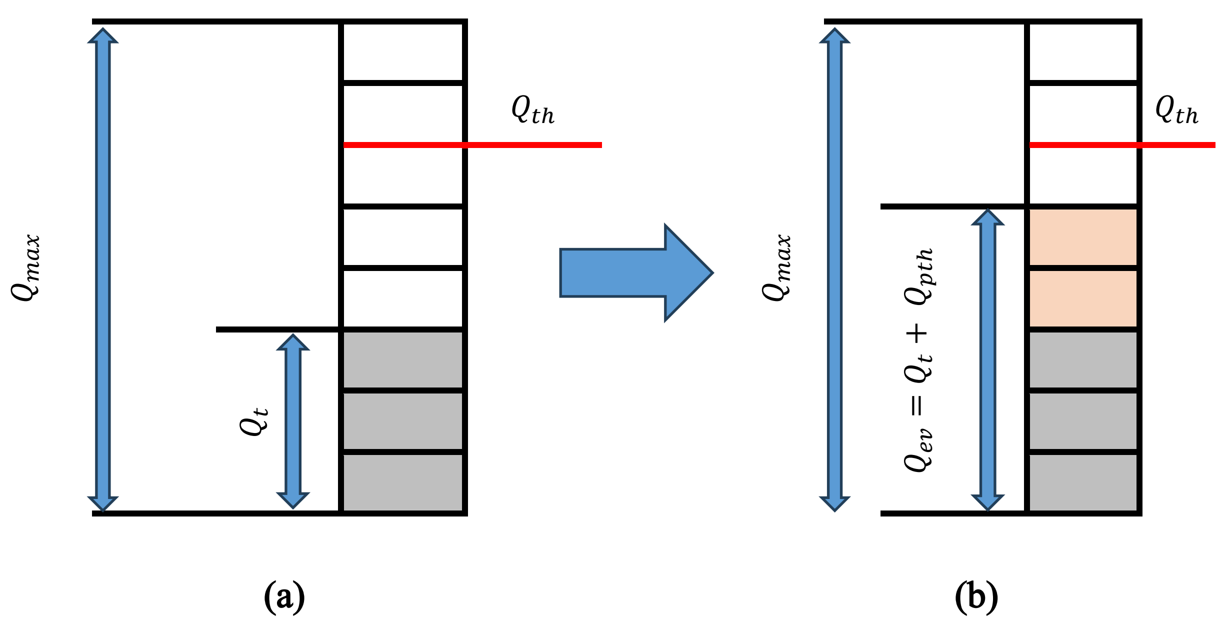


Figure 4.3: Congestion detection at a DMC-RPL node
(a) The traditional method: When $Q_t > \; Q_{th}$ (b) In DMC-RPL: When $Q_{ev} > \; Q_{th}$

Finally, it is important to modify the Trickle timer to propagate the buffer-length information faster in case of congestion. While some researchers use a timer that sends DIOs with Q values every second [9], DMC-RPL sets the limit of the Trickle timer to 5 seconds. If a DIO is not sent when the Trickle timer is running, a DIO is sent when it expires.

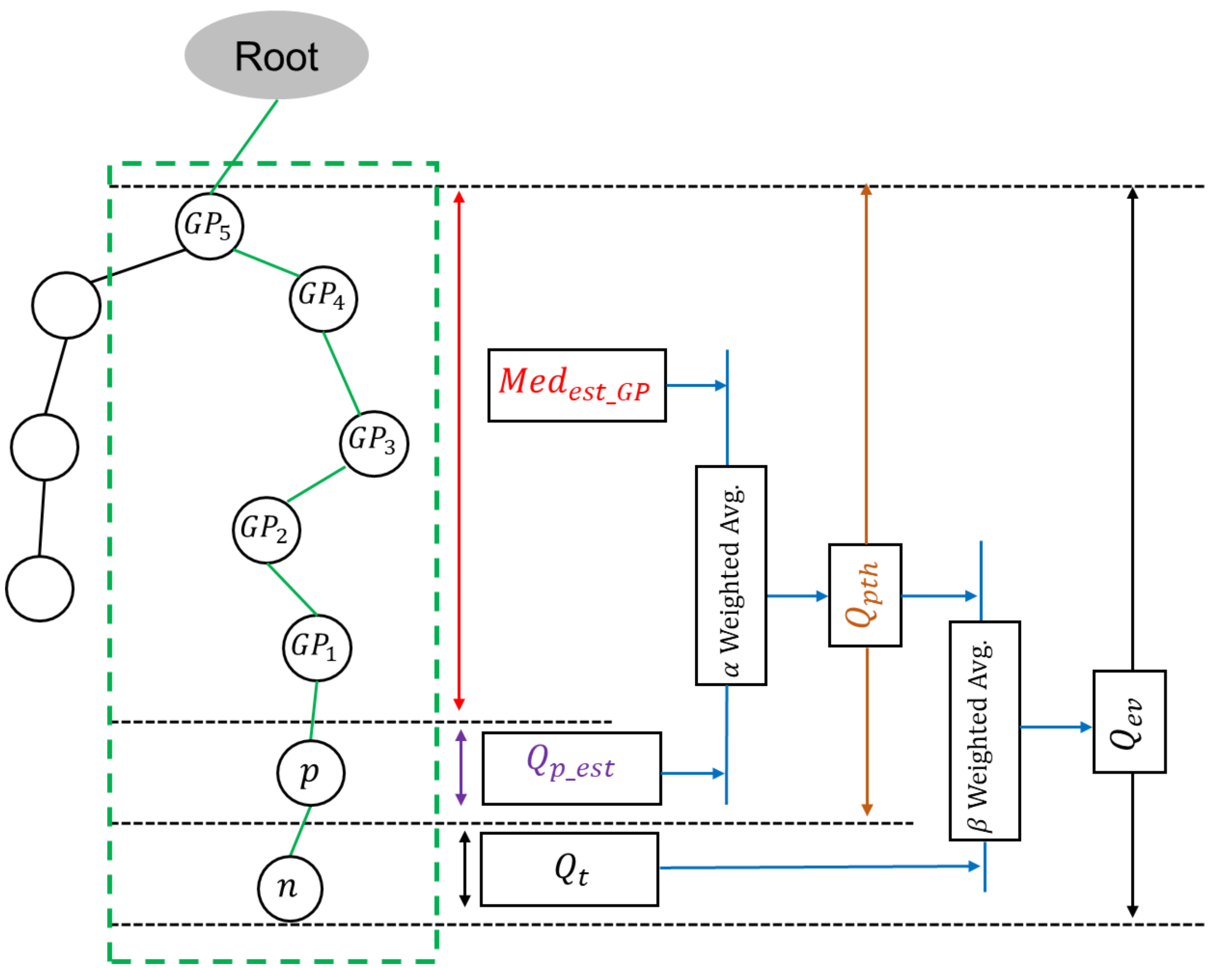


Figure 4.4: Visual representation of equations used to calculate Q_{ev} at the leaf node n

4.3. DMC-RPL's path diversity and distributed cooperation control

DMC-RPL supports multi-path routing for inter-instance packets. In RPL, a node receives DIO messages from all its neighboring nodes, which can categorized into two groups:

1- Nodes from its instance: These are normally candidate parents or the preferred parent. We refer to them simply as parents.
2- Nodes from other instances within the vicinity of the node: We call them inter-instance nodes. In DMC-RPL, each of these nodes has the potential to forward its packets via another instance. They aim to mitigate congestion by cooperating with each other.

An RPL node maintains a table that stores the parameters of all its parents (the preferred parent and the candidate parents). This table has the address, rank, metric, and other values of a parent. These values are updated whenever a parent's DIO message is received. In DMC-RPL, the routing table is extended to store and update the DIO values of neighboring inter-instance nodes as well. Therefore, in DMC-RPL, the DIO values are used for path-congestion estimation of the path via the preferred parent, as well as the paths via the candidate parents and the inter-instance paths via inter-instance nodes.

Each node in RPL has one path to the root, which passes through the preferred parent. In DMC-RPL, the presence of neighboring nodes provides multiple alternative paths to the root. A DMC-RPL node can accept forwarding packets from other instances even if its main path to the root (via the preferred parent) is congested, given that an alternative, less-congest path via a candidate parent can be used. Such a path could be optimal congestion-wise but less optimal from the view of the local routing metric. For example, a node can forward packets from another instance via a candidate parent, where the path via that parent is less congested but also less reliable or has more delay compared to the path via the preferred parent. Figure 4.5 shows an example of these paths, where in the path (n → o → p →q →r) the inter-instance node (q) forwards inter-instance traffic via a candidate parent (r).

Moreover, inter-instance nodes may have alternative inter-instance paths via nodes from other instances, which they can use to forward their traffic when they are congested. The paths (p →q) and (c → d) in Figure 4.5 are examples of these paths.

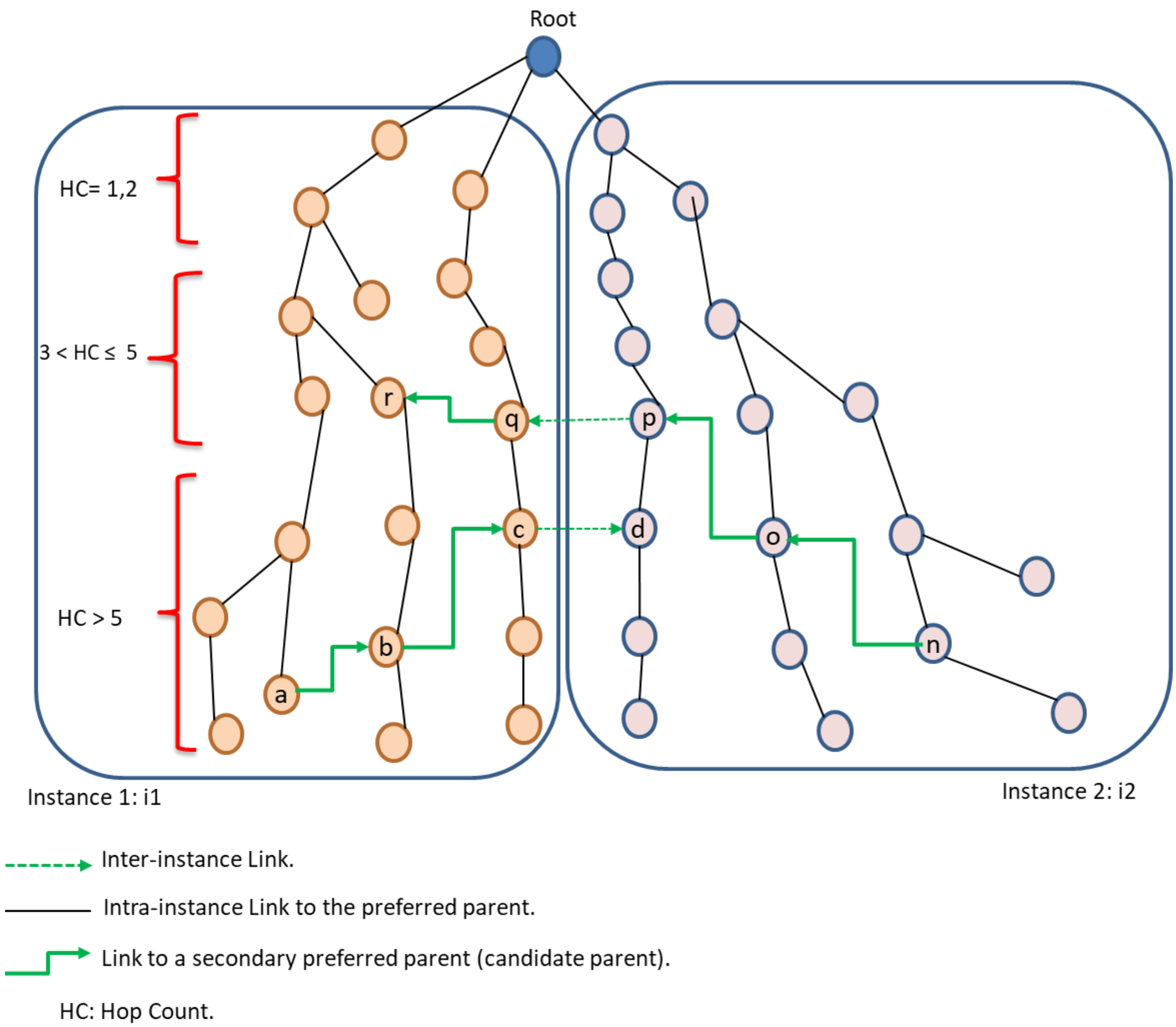


Figure 4.5: Multipath routing in DMC-RPL with HC ranges

To enable the cooperating inter-instance nodes to manage their cooperation autonomously, and to to tackle to problem of selfishness among them, we design a distributed Cooperation Control (Co-Co) mechanism for DMC-RPL.

To illustrate how Co-Co works, we refer to each time a node forwards an inter-instance packet as a cooperation round. We assume that the cooperating nodes are N_1 from instance i_1 (N_1^{i1}) and N_2 from instance i_2 (N_2^{i2}), where the number of packets they send to each other is n_1 and n_2, respectively. The steps of the Co-Co algorithm can be explained as follows:

1. N_1^{i1} forwards a certain number of packets ($n_2 \leq n_{max}$) from node N_2^{i2}, without requesting N_2^{i2} to forward N_1^{i1} packets ($n_1 = 0$), i.e., N_1^{i1} Cooperates (C) and N_2^{i2} Defects (D) for cooperation rounds that are up to (n_{max}).
2. After forwarding ($n_2 = n_{max}$) packets, N_1^{i1} stops forwarding N_2^{i2} packets (Defects), until N_2^{i2} forwards a number of packets ($n_1 \geq n_{min}$) from N_1^{i1}.
3. After N_2^{i2} has forwarded (n_{min}) packets from N_1^{i1}, the cooperation can resume.

Throughout the entire interaction period between N_1^{i1} and N_2^{i2}, each of them tracks its (C, D) moves in a score-like fashion. The (C, D) values along with threshold values (n_{max} , n_{min}) are the input parameters of the DMC-RPL Cooperation Control algorithm.

Figure 4.6 illustrates an example of the Co-Co algorithm, where $n_{max} = 5$ and $n_{min} = 3$. The Figure displays when cooperation ends (co-op ends) and when it resumes (co-op resumes) between N_1^{i1} and N_2^{i2}. The R-axis in Figure 4.6 denotes the cooperation rounds.

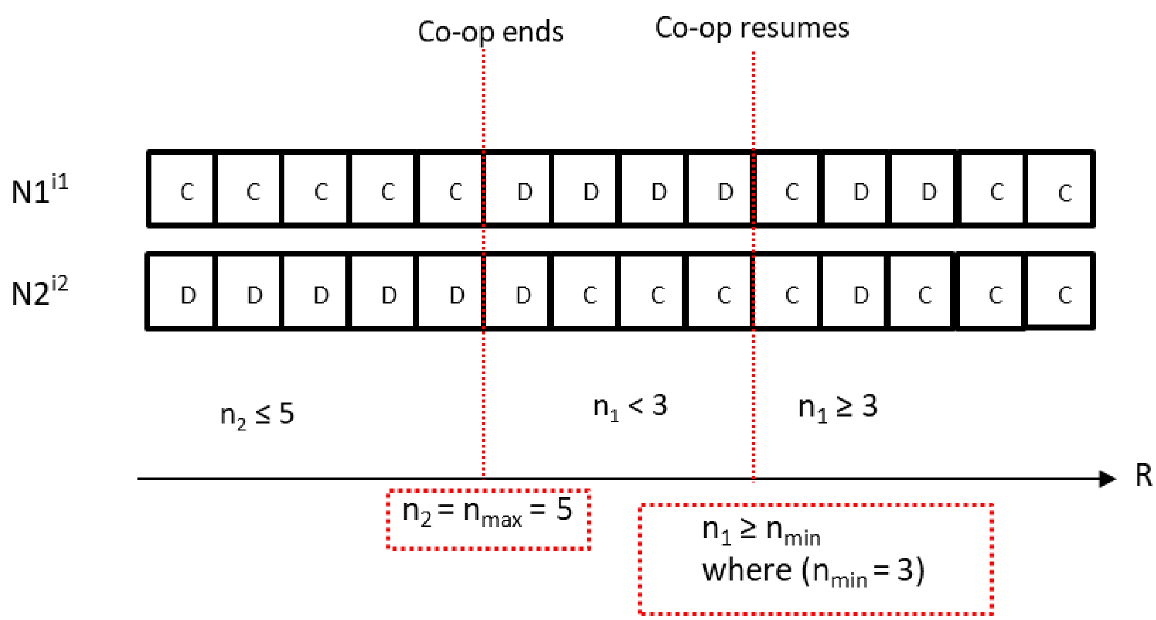


Figure 4.6: Cooperation Control in DMC-RPL

4.4. DMC-RPL weights

The weights α and β from equations (4-7) and (4-8) are set dynamically by each node based on its Hop Count. We rely on three main HC ranges as shown in Figure 4.5.

Nodes with HC = 1 have the root as their parent. They do not have any parent or Grandparent leaf nodes to be used in congestion estimation, so they always set β to 1 in equation (4-8). Similarly, HC = 2 means that the node has a parent leaf node and the root as a Grandparent. Since there is no Grandparent chain for such nodes, they set α in equation (4-7) to 1.

At higher HC values, we choose weight values that increase the weight of Med_{est_GP} the further a node is from the root. This is performed by decreasing the values of α and β with the increase of the HC. The reason for that is that the further we get from the root, the less the congestion. Thus, we place more focus on the congestion estimation of the path than the node's buffer, and specifically, more importance is given here to the Grandparent chain than the parent. Likewise, the closer a node is to the root, the more importance is given to its own Q value and its parent's, which is reflected in the higher values of α and β.

Another motivation for selecting these values in such a way is that the estimation of the median and the *IQR* calculation is challenging when the data set has a few values only. With HC = 5, a node has three Grandparents, which is the minimum number of samples required to calculate *IQR*. We set the minimum values for α and β as: $\alpha_{min} = \beta_{min} = 0.2$, as Table 4.3 shows.

HC	$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$
1	-	1
2	1	0.75
3	0.65	0.65
4	0.5	0.5
5	0.4	0.4
6	0.35	0.35
7	0.3	0.3
8	0.25	0.25
≥ 9	α_{min}	β_{min}

Table 4.3: α and β values according to the Hop Count

4.5. Summary

In this Chapter, we introduced DMC-RPL, a novel routing protocol that uses distributed, inter-instance cooperation for congestion control in IoT networks. The design of DMC-RPL is intended to be lightweight, interoperable, and autonomous. We designed a metric for evaluating the path congestion level locally and proposed Q_{ev}, a distributed congestion estimation routing metric for congestion detection in LLNs.

As explained in Section 3.5, relying on downward routing is a major shortcoming of RPL's inter-instance cooperation models. DMC-RPL does not rely on downward routing and thus spares itself from the overhead it causes.

DMC-RPL employs a cooperation control mechanism that enables autonomous and proactive cooperation among inter-instance nodes.

In the next Chapter, we evaluate the performance of DMC-RPL in several settings.

DMC-RPL does not rely on downward routing; thus, it spares itself from the overhead this routing causes, especially since it is a major shortcoming of the proposed inter-insurance cooperation solutions designed for RPL as explained in Section 3.5.

In the next Chapter, we evaluate the performance of DMC-RPL in several scenarios.

5. Evaluation

In this Chapter, we use multiple simulation scenarios to evaluate DMC-RPL and assess its performance.

5.1. Configuration

We evaluate DMC-RPL using the Contiki-OS [110] and the COOJA simulator [112] using 60 Wismote leaf nodes [148] (30 per instance) and one Wismote node as a root. The nodes were distributed randomly in an area of 200*200 m^2, as shown in Figure 5.1. We disabled the radio duty cycling to enable sending packets at high rates, as performed in many RPL evaluations under heavy traffic [9, 14]. We compared our implementation with Contiki-RPL [111], which is the default RPL implementation in the Contiki-OS. Table 5.1 displays the simulation setup summary.

The Queue buffer used in our simulations is the Contiki Queuebuf [114], which is the FIFO buffer demonstrated in Figure 3.1. We set the maximum Queuebuf capacity (Q_{max}) to 15, since it usually ranges from 8 to 20 in RPL's previous studies [14, 116, 124].

Our evaluations were performed under heavy and dynamic traffic. When LLN nodes generate heavy and/or dynamic traffic, their networks become prone to congestion. Kim *et al.* studied congestion in LLNs with 30 nodes and a heavy traffic rate of 30 packets/minute (pkts/minute) [15]. Those values guided the selection of our simulation parameters, where each instance in our simulations was composed of 30 nodes with traffic rates ranging from 4 to 60 pkts/minute.

As mentioned in Section 1.4, congestion can take place in large-scale networks even with light traffic. According to Kim *et al.*, studying a network of 30 nodes with a traffic rate of 30 pkts/minute is equivalent to studying a large-scale network of 5000 nodes with a traffic rate of 11 pkts/minute [15]. We followed the approach of Kim *et al.* in our evaluations by simulating a network of 30 nodes per instance with traffic rates up to 30 and 60 pkts/minute. Therefore, based on the conclusion of Kim *et al.*, our results can be extended to larger networks with thousands of nodes and lower traffic rates.

Taghizadeh *et al.* investigated congestion under dynamic traffic in LLNs, in which a node sends one stream of packets for 100 seconds at rates of 15 or 30 pkts/minute [14]. We followed a similar approach for our dynamic traffic setup, where each node sent sporadic traffic for 60 seconds at a rate of 30 pkts/minute.

Parameter	**Value**
Number of instances	2
Number of nodes per instance	30
Periodic traffic rate	4, 6, 10, 30,60 pkts/minute
Sporadic traffic rate	30 pkts/minute
Sporadic traffic duration	1 minute per node
OS	Contiki 3.x
Mote type	Wismote
Buffer size	15 pkts
Mac protocol	CSMA
Routing metrics	ETX, Energy
OF	MRHOF
Transmission range	50 m
Interference range	100 m
distance loss	90%
Duration	15 min
Traffic type	Constant bitrate
Transceiver	CC2520
Queue type	FIFO

Table 5.1: The primary simulation parameters

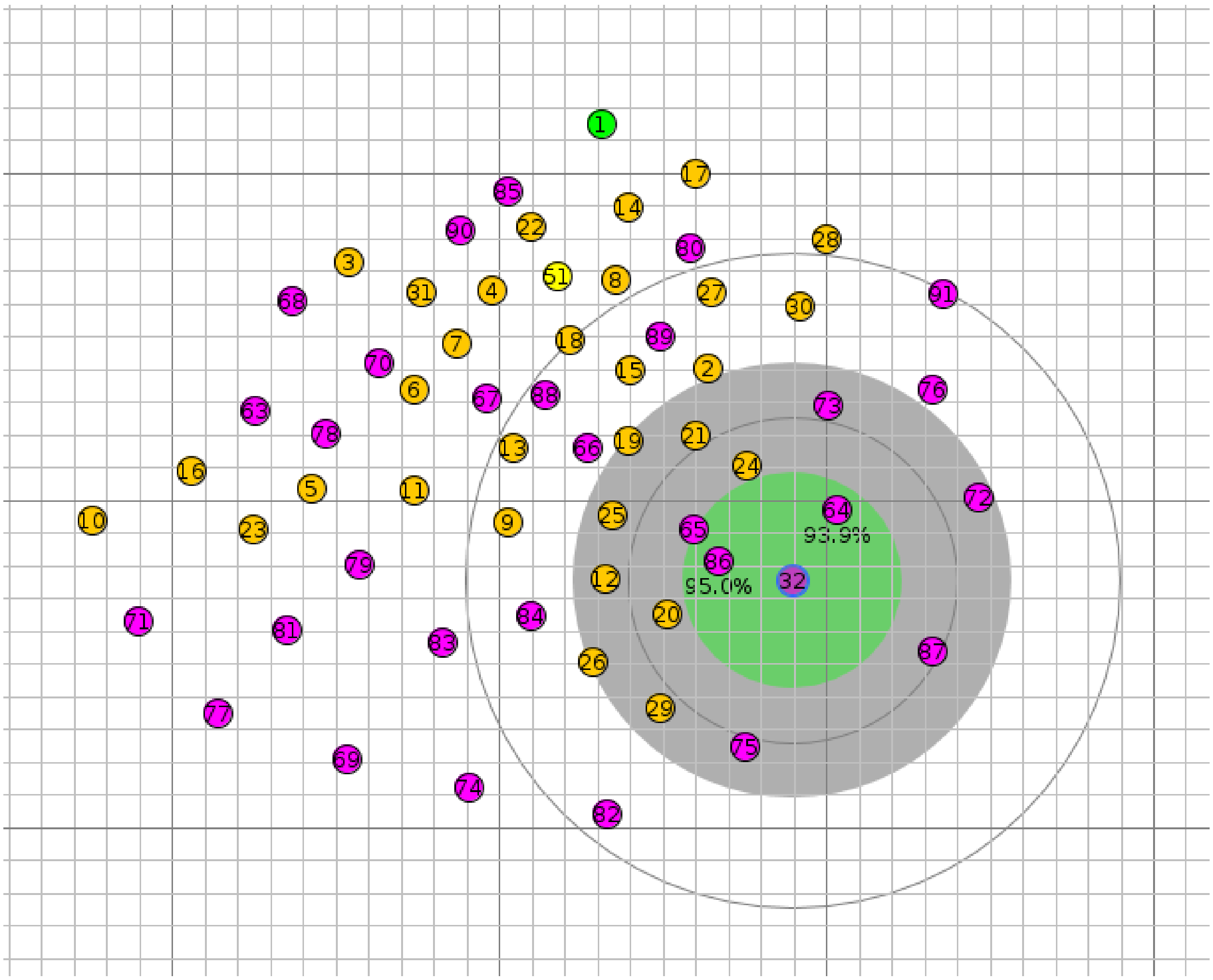


Figure 5.1: The simulation topology in COOJA

5.2. Application Scenario

The applications we chose to study are SG applications that are detailed in Section 2.6.2. Our simulation environment is a NAN segment of the Smart Grid, where we study two AMI applications:

1. Critical_AMI_Data: Its traffic requires a high level of reliability. An example application for that is electricity quality monitoring. Instances running this application use ETX as a routing metric to achieve high reliability.

2. Non-critical_AMI_Data: Traffic that can tolerate errors such as collecting Smart Meters' data of energy consumption for billing. Instances supporting this traffic use energy consumption as a metric to increase the network's lifetime.

Both applications above generate periodic traffic at high rates of up to 1 pkt/sec to produce heavy traffic.

The dynamic traffic application we studied is an alarm application for Distributed Energy Resources monitoring. Its traffic is characterized to be sporadic, i.e., each node generated this type of traffic as a single stream of packets for a limited time during each simulation run. In our implementation, the sporadic traffic was triggered by a sporadic event detector, which we implemented using a timer. During each simulation run, a node that is configured to generate sporadic traffic would have its sporadic event detector launched only once at a random time. This caused the node to generate sporadic traffic characterized by a data stream of 30 pkts/min for one minute.

During an experiment, each instance ran this sporadic application concurrently with its periodic AMI application. That means that the sporadic application does not have its own routing metric and was routed based on the metric used for the periodic application. We here provide a description of this sporadic application:

- DER_Alarm_Data: This data is generated from sensors that monitor the Distributed Energy Resources and pertains to the protection of the electricity distribution network. The traffic of this type is sporadic and appears randomly. It requires high reliability and low delay. Data belonging to this application include transformers' faults, the status of circuit breakers and switches, and the magnitude and direction of the electric flow.

Since we compared the standard RPL (Contiki-RPL) with our proposed protocol (DMC-RPL) when both are running two periodic and one sporadic application, we use the following notion to distinguish them:

- DMC-RPL-Pr: refers to a periodic application (either Critical_AMI_Data or Non-critical_AMI_Data) running with DMC-RPL.
- DMC-RPL-Sp: denotes the sporadic application (DER_Alarm_Data) when running with DMC-RPL.

- Contiki-RPL-Pr: refers to a periodic application (either Critical_AMI_Data or Non-critical_AMI_Data) running with Contiki-RPL.
- Contiki-RPL-Sp: denotes the sporadic application (DER_Alarm_Data) when running with Contiki-RPL.

5.3. Scenario 1: Both instances use the same routing metric

In this scenario, both instances ran the Critical_AMI_Data periodic application. That means; they both employed the ETX metric to achieve better reliability in data forwarding. In addition, both instances supported the DER_Alarm_Data application, which generates sporadic traffic. The Non-critical_AMI_Data application does not exist in this scenario; hence, the energy metric was not used. The reason for this choice is to compare the performance of Contiki-RPL and DMC-RPL when both instances run the same metric (ETX). Figure 5.2 shows the setup of this scenario.

It is worth noting that in the results we show in this section, the X-axis is the packet interval (pkt interval). A packet interval of 1,2,6,10,15 seconds means a data rate of 60,30,10,6,4 pkts/minute respectively.

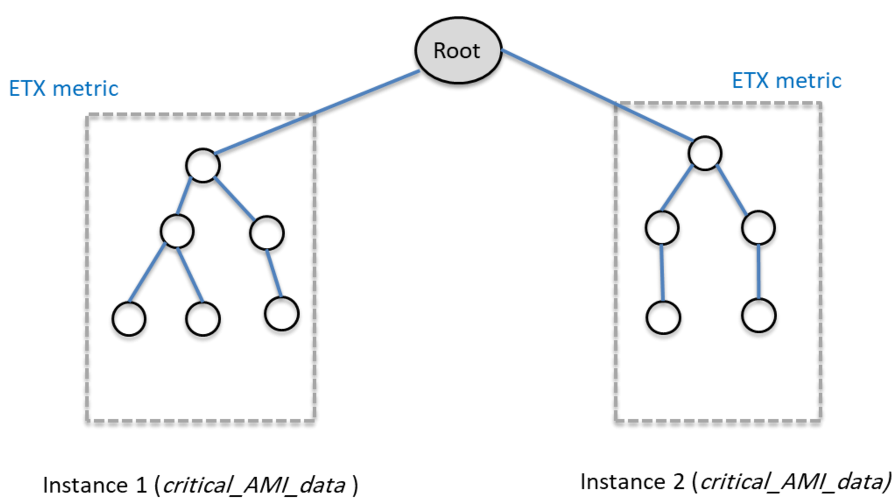


Figure 5.2: The network topology in scenario 1

5.3.1. Packet Delivery Ratio

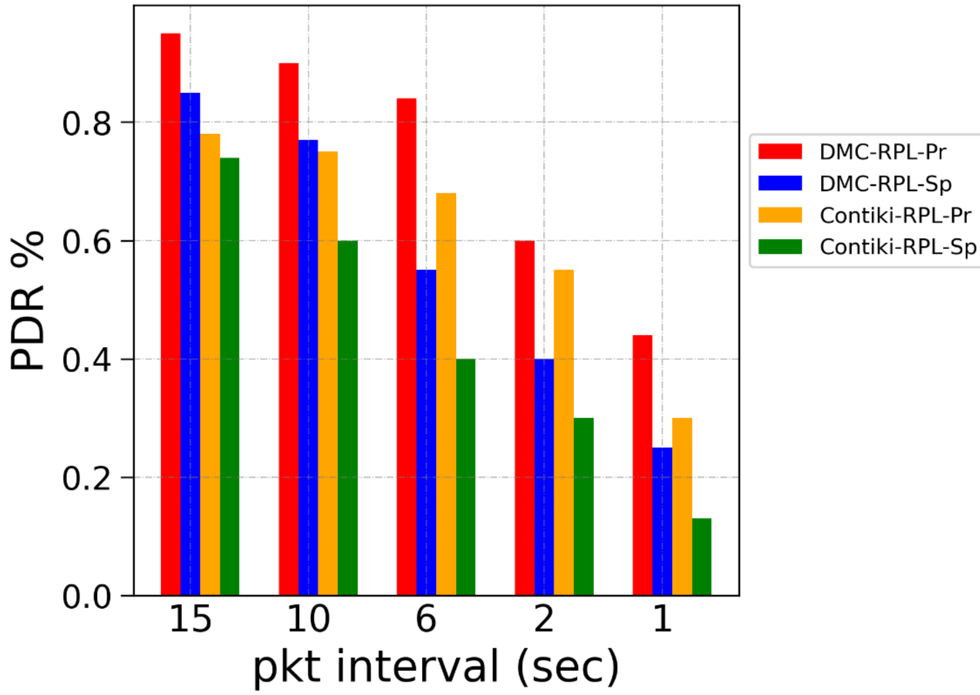


Figure 5.3: PDR at various data rates

Figure 5.3 shows the packet delivery rate at different data rates for the periodic and sporadic applications. The effect of DMC-RPL finding less congested paths is noticeable in this Figure. This effect is smaller at low data rates of 4 pkts/minute, which corresponds to a packet interval of 15 seconds. At these low data rates, the network can still handle congestion. Therefore, Contiki-RPL-Pr has an acceptable PDR at traffic rates up to 10 pkts /minute. Yet, it is clear from the low PDR of Contiki-RPL-Pr at packet intervals of 1 to 2 seconds that congestion degrades its performance at heavy traffic rates of 30 pkts/minute and higher. Whereas DMC-RPL-Pr still keeps a higher PDR under heavy traffic due to its congestion detection and alleviation mechanism.

The sporadic application Contiki-RPL-Sp packet delivery rate drops quickly with the increase in the data rate. As bottleneck nodes get congested, the delivery of packets belonging to nodes situated far away from the root becomes difficult. Traffic streams from such nodes may not be able to travel far in the network, and thus, sporadic traffic from these far away leaf nodes suffers from high packet losses.

When it comes to DMC-RPL-Sp and DMC-RPL-Pr, alternative paths to the root are exploited to handle data delivery at high rates. However, with heavy traffic accompanied by sporadic data streams in the network, and due to the limited buffer size (Q_{max}) available, it gets difficult for DMC-RPL-Sp to find paths that are not congested, thus its PDR drops significantly with heavy traffic.

5.3.2. Buffer loss ratio

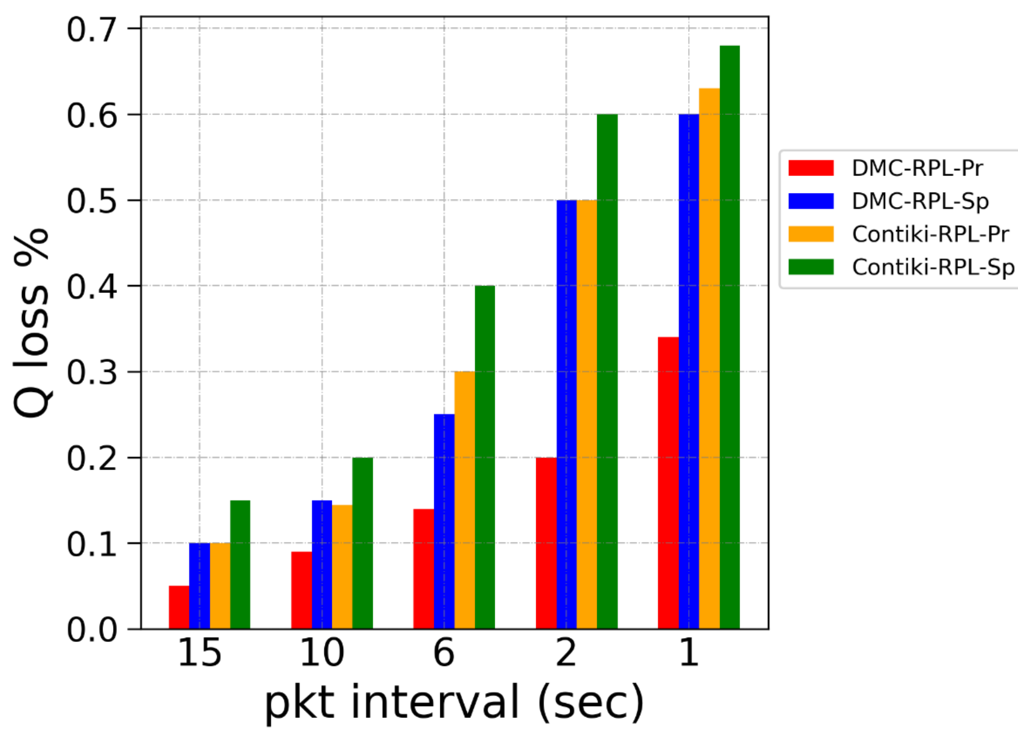


Figure 5.4: Packet loss at nodes' buffers at different traffic loads

The buffer loss ratio, or Q loss, is the average packet loss caused by nodes dropping packets when their buffers overflow. It is illustrated in Figure 5.4.

Again, when sending 4 to 6 packets per minute, the network can still handle such rates even when sporadic traffic is present. With the increase in the traffic rate, we can observe that DMC-RPL-Pr has the best performance. That is because DMC-RPL anticipates congestion using its Q_{ev} metric and searches for alternative paths before the buffer becomes full.

Both simulated sporadic applications suffer from high queue losses. The reason for that is when a node sends a sporadic traffic stream, it can quickly fill up the buffers of the relay nodes on its path to the root. DMC-RPL-Sp still tries to find less congested paths, but the options are limited since the number of nodes in the vicinity of each other in our simulation is not big.

5.3.3. Energy consumption

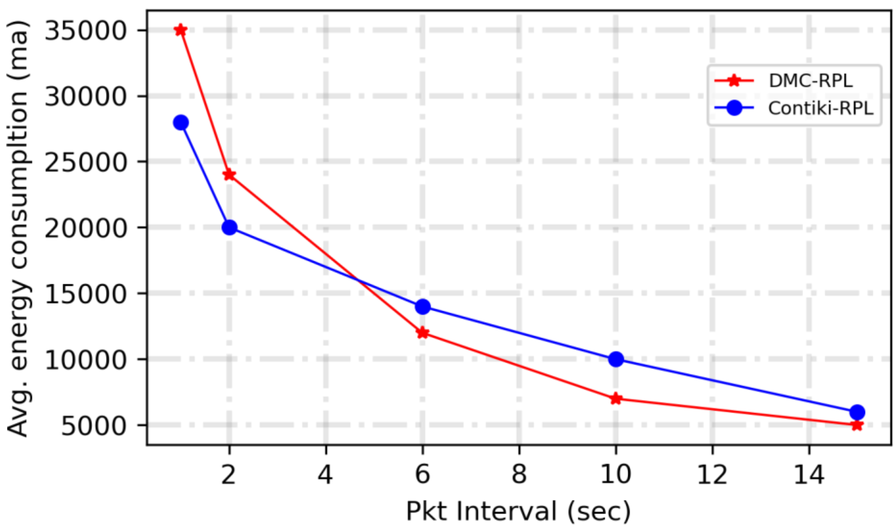


Figure 5.5: Consumed energy at various data rates

The average energy consumption is plotted in Figure 5.5. The average energy consumed by DMC-RPL was calculated by averaging the energy consumption of DMC-RPL sporadic and periodic (DMC-RPL-Sp and DMC-RPL-Pr). The average energy consumption of Contiki-RPL was calculated in a similar way using Contiki-RPL-Pr and Contiki-RPL-Sp.

For low-to-medium traffic loads, DMC-RPL consumes less energy because finding a less-congested path means also finding a path where channel loss due to collision is lower. That is why DMC-RPL outperforms Contiki-RPL when packet intervals of 6 seconds or more are used.

However, with higher traffic rates, Contiki-RPL consumes less energy. The reason for that is that Contiki-RPL drops more packets due to buffer overflow. The more packets it drops, the less energy it spends on trying to deliver data to the root. In contrast, DMC-RPL consumes more energy as it tries sending packets via less-congested paths instead of dropping them. That is reflected in Figure 5.3, which shows that DMC-RPL achieves a better PDR than Contiki-RPL.

5.4. Scenario 2: Each instance uses a different routing metric

In this scenario, each instance ran a different periodic application. Instance 1 ran the Critical_AMI_Data application and used ETX as a routing metric, while Instance 2 ran the Non-critical_AMI_Data application and used the energy routing metric, as shown in Figure 5.6.

We studied each instance separately. We show the results of Instance 1 when it used DMC-RPL compared to when it used Contiki-RPL. Then we do the same for Instance 2.

As a reminder, in the results in this section, the X-axis is the packet interval (pkt interval). A packet interval of 1,2,6,10,15 seconds corresponds to a data rate of 60,30,10,6,4 pkts/minute respectively.

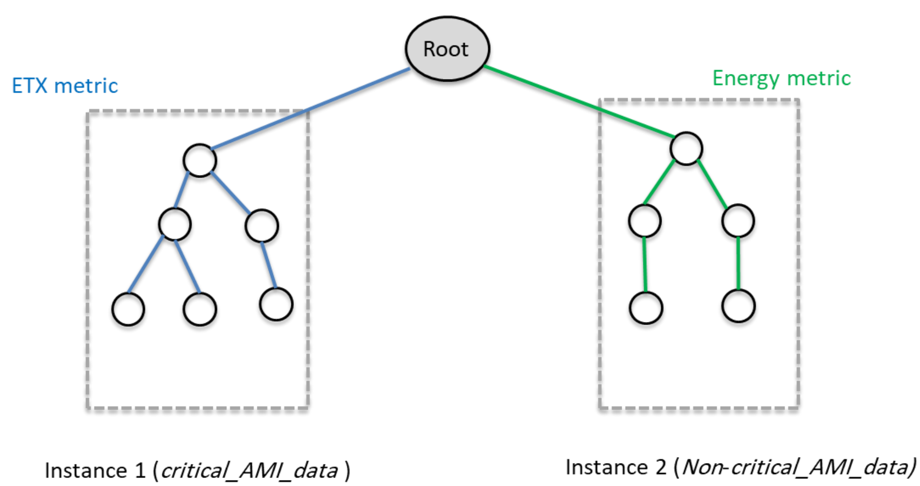


Figure 5.6: The network topology in scenario 2

5.4.1. Results for Instance 1 (the instance with the ETX metric)

Since the metric used in this instance is ETX, we refer to the periodic and sporadic applications of DMC-RPL as DMC-RPL-Pr-etx and DMC-RPL-Sp-etx, respectively. DMC-RPL-etx is the term used when plotting the average energy consumption of DMC-RPL in Instance 1, which is the average of the energy consumption of DMC-RPL-Pr-etx and DMC-RPL-Sp-etx. Similarly, we use the following terms with Contiki-RPL in Instance 1: Contiki-RPL-Pr-etx, Contiki-RPL-Sp-etx, and Contiki-RPL-etx.

In Figure 5.7 we can see the average energy consumed in Instance 1 when ETX was used as a metric, i.e., the routing was based on the link reliability and not the consumed energy. We notice that the energy consumption for Contiki-RPL-etx increases with the increase in the traffic rate. This is expected since the energy consumption is not considered in Instance 1. As to DMC-RPL-etx, it performed better than Contiki-RPL-etx in terms of energy consumption under light traffic.

Nevertheless, at heavy traffic loads of 1-2 pkts/sec, we can see that the energy consumption of DMC-RPL-etx increased noticeably and exceeded the energy consumption of Contiki-RPL-etx. At these rates, the nodes get congested more frequently. In DMC-RPL-etx, the Trickle timers were resetting more often, and more DIO messages were propagated to spread information about congestion across the network. In addition, DMC-RPL-etx tried to find alternative paths to forward packets instead of dropping them, which made it consume more energy compared to Contiki-RPL-etx, which dropped more packets at these rates, thus saving energy in this process. We can see the effect of this packet dropping under heavy traffic from Contiki-RPL-etx on PDR in Figure 5.8.

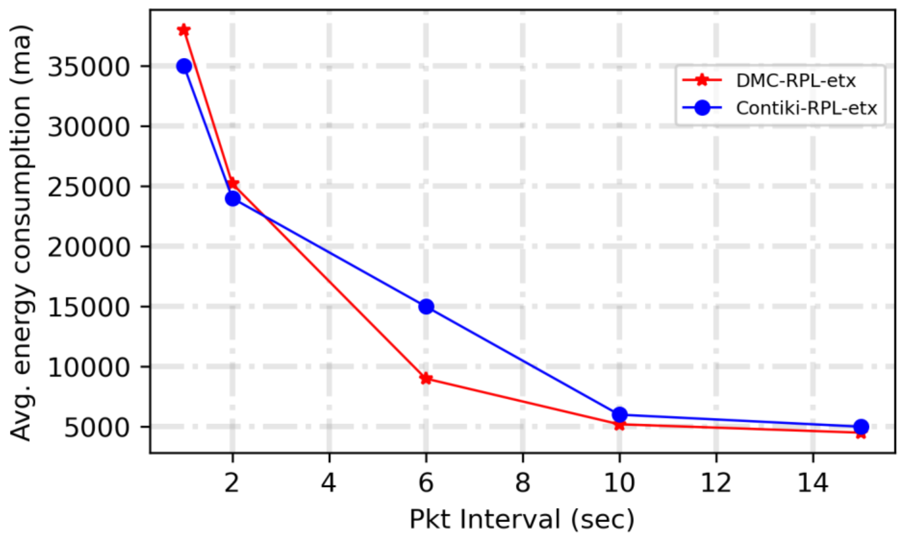


Figure 5.7: Consumed energy for Instance 1 at various data rates

Figure 5.8 shows PDR values for Instance 1 for the simulations we performed with DMC-RPL and Contiki-RPL. We observe that at lower data rates, both protocols perform relatively well. So for packet intervals ranging from 6 to 15 seconds, the periodic applications Contiki-RPL-Pr-etx and DMC-RPL-Pr-etx keep a PDR over 40% and 55%, respectively.

At higher data rates, Contiki-RPL failed to cope with heavy traffic and started dropping packets more frequently. That is because ETX cannot detect congestion at the nodes' buffers, which happens rapidly under dynamic traffic. Therefore, Contiki-RPL-Pr-etx and Contiki-RPL-Sp-etx had low PDR values at packet intervals of 1-2 seconds. At these intervals, DMC-RPL performed better than Contiki-RPL, since it is more dynamic with the DIO updates. When congestion is detected, the nodes in DMC-RPL send DIO messages more frequently, which increases the rate of ETX updates in the network, thereby enhancing reliability. In addition, forwarding packets via the other instance also helps to relieve congestion in DMC-RPL.

Figure 5.8 displays that DMC-RPL-Pr-etx had the best overall performance in this scenario. As to sporadic applications, DMC-RPL-Sp-etx performed better than Contiki-RPL-Sp-etx, even under heavy traffic. That is due to the ability of DMC-RPL to detect congestion and react to it by finding alternative paths.

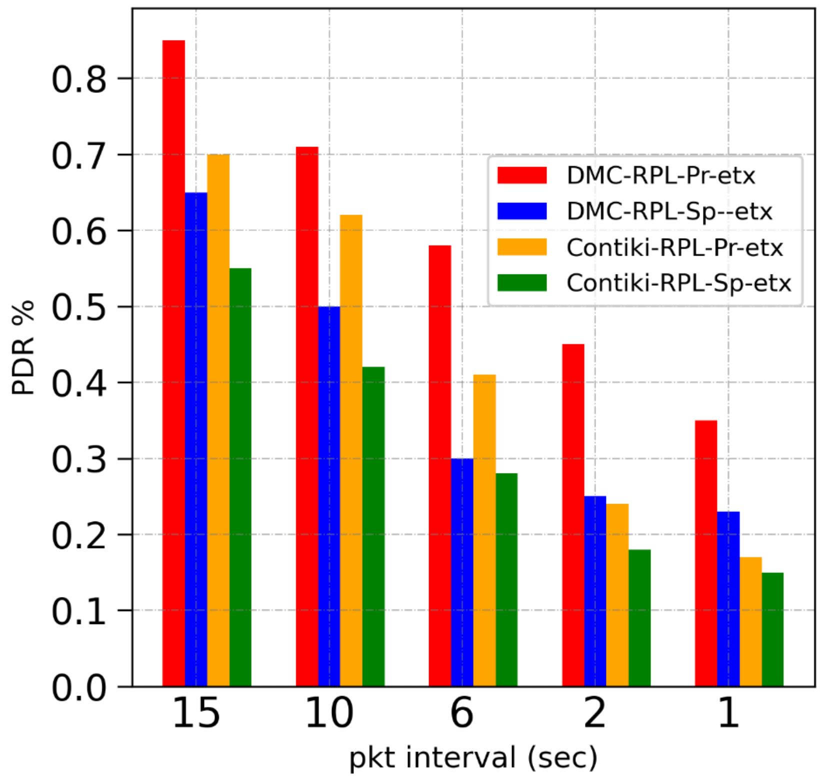


Figure 5.8: PDR at various data rates for Instance 1

5.4.2. Results for Instance 2 (the instance with the energy metric)

Figure 5.9 shows the energy consumption in Instance 2, which used an energy-consumption metric to achieve energy-efficient routing. In this instance, we refer to the periodic and sporadic applications of DMC-RPL as DMC-RPL-Pr-energy and DMC-RPL-Sp-energy, respectively. DMC-RPL-energy used in Figure 5.9 is the averaged energy consumption of DMC-RPL-Pr-energy and DMC-RPL-Sp-energy. Similarly, the corresponding terms for Contiki-RPL in Instance 2 are: Contiki-RPL-Pr-energy, Contiki-RPL-Sp-energy, and Contiki-RPL-energy.

We can see in Figure 5.9 that the energy consumption of Contiki-RPL-energy is less than DMC-RPL-energy, which is because it only forwards traffic generated by its own instance, i.e., intra-instance packets. While in DMC-RPL-energy, at high traffic rates, Instance 2 may forward inter-instance packets coming from Instance 1 in addition to its own. It is true that Instance 2 also uses Instance 1 for packet forwarding when congestion is detected, but the presence of sporadic traffic is what makes this cooperation one-sided at certain times in the simulation. The sporadic traffic appearing at Instance 1 could make it congested faster and forward its traffic via Instance 2. However, Instance 2 (energy instance) can handle congestion better than Instance 1 (ETX instance). That is because the ETX metric is agnostic to the buffer congestion, which makes the parent selection process unaware of the congestion at the parent. On the other hand, the energy metric can detect the effects of congestion. When the congested nodes turn into bottlenecks, their energy consumption increases, and thus, they will be less likely to be chosen as parents in Instance 2. The combination of our Q_{ev} metric and the energy metric in DMC-RPL-energy made it perform better regarding energy consumption compared to DMC-RPL-etx in Figure 5.7. The Cooperation Control mechanism of DMC-RPL limits the number of inter-instance packets that can be sent unilaterally from one instance to another; thus, the energy consumption of DMC-RPL-energy was still not far higher than Contiki-RPL-energy.

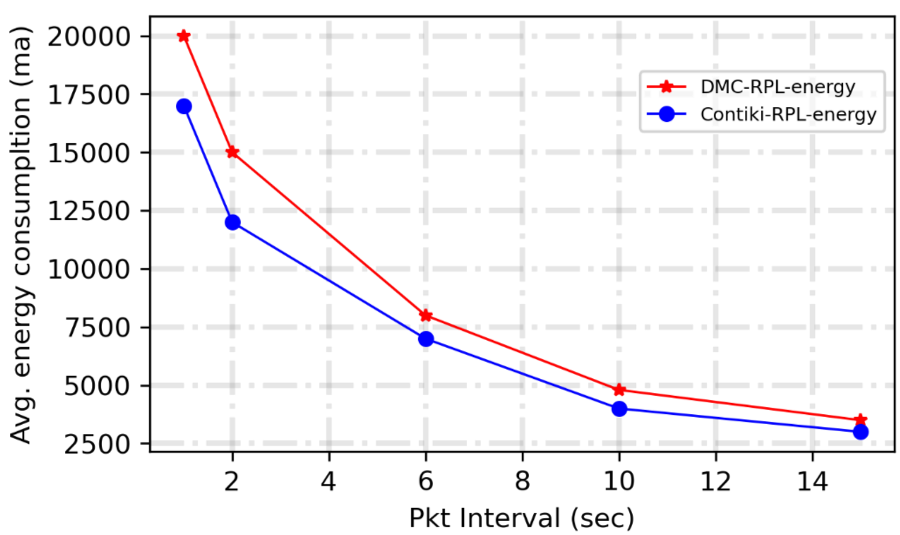


Figure 5.9: Instance 2 consumed energy at various data rates

Figure 5.10 illustrates the PDR values for Instance 2. DMC-RPL performs better than Contiki-RPL in this case as well. At packet intervals between 10-15 seconds, the sporadic application of

DMC-RPL (DMC-RPL-Sp-energy) had better results than the sporadic application of Contiki-RPL (Contiki-RPL-Sp-energy). Nevertheless, both applications suffered from high packet losses under heavy traffic. At packet intervals of 1-2 seconds, the network already had periodic traffic at rates of 60-30 pkts/minute. At the same time, sporadic traffic with the same rate appeared randomly in the network. The PDR of such traffic is better for DMC-RPL-Sp-energy compared to Contiki-RPL-Sp-energy, but was still low. This can be justified given that under light traffic, the congestion level in the network was low and DMC-RPL-Sp-energy was able to react to congestion, whereas under heavy traffic, finding alternative paths for sporadic data streams became difficult.

It is worth mentioning that compared to Figure 5.8, the Contiki-RPL performed better when it comes to routing periodic traffic at high data rates (packet intervals of 1-2 seconds). In other words, Contiki-RPL-energy had a better PDR than Contiki-RPL-etx under heavy traffic. That is because the energy metric can indirectly indicate congestion since bottleneck nodes become detectable by their energy levels. We see in Figure 5.10 that periodic DMC-RPL actually performed well in terms of PDR, given that the routing metric used in this instance is an energy, not a reliability metric. This shows the effect of DMC-RPL's congestion mitigation. Still, at high data rates, the PDR of DMC-RPL-Pr-energy drops to 30-40%. Under heavy traffic, it becomes difficult to find non-congested paths, if any exist. Forwarding via a less congested parent does not mean that the path is still optimal.

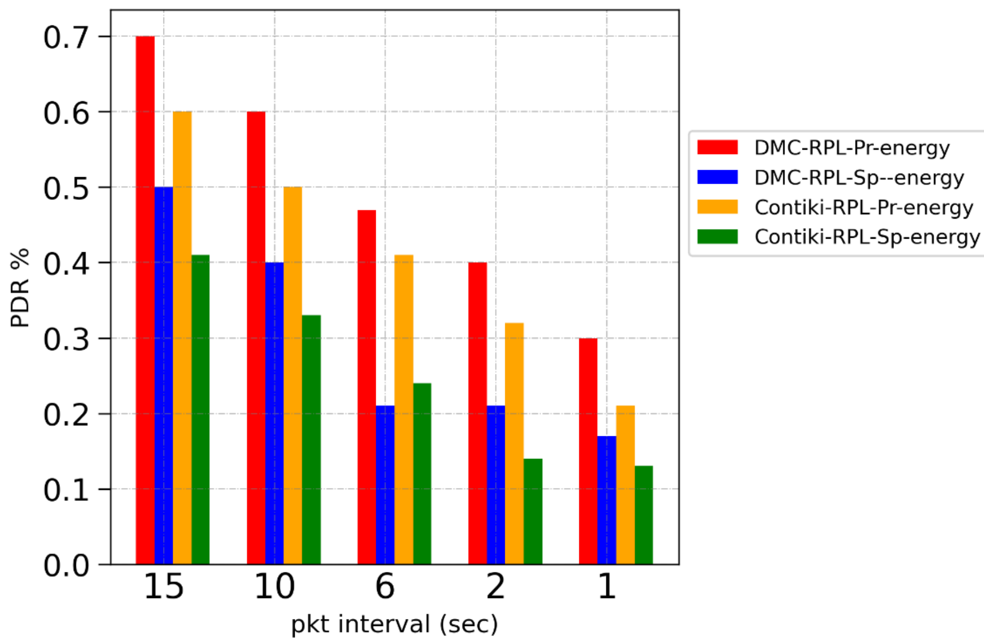


Figure 5.10: PDR at various data rates for Instance 1

5.5. Summary

Using different simulation scenarios, we evaluated DMC-RPL and compared its performance to Contiki-RPL. Our results show that DMC-RPL performed better regarding PDR, especially when routing periodic traffic.

The distributed and proactive approach of DMC-RPL to mitigate congestion has shown improvements in reliability and congestion mitigation. At the same time, DMC-RPL had an acceptable increase in energy consumption compared to Contiki-RPL under heavy traffic, which is caused by the higher rate of DIO messages in this case, and routing packets via alternative paths instead of dropping them.

6. A framework for cooperation among RPL instances using virtual credits

In this Chapter, we introduce a novel scheme to promote cooperation among nodes from different RPL instances by using virtual credits gained and spent through forwarding inter-instance packets. Our proposed cooperation scheme is asymmetric, which implies that the requesting nodes have an interest in cooperation, but the forwarding nodes do not. However, it is assumed that somewhere else in the network, the same situation exists but in reverse. Thus, cooperation on the instance level, not the node level, is envisioned in our proposed framework. Our framework targets the Smart City ecosystem, where many of its subsystems are co-located, making instances that belong to different authorities interwind.

6.1. Motivation and application scenarios

We first examine selected related works in multi-gateway Smart City systems, where different networks or different network instances coexist.

6.1.1. Smart City systems that share their communication networks

The implementation of IoT is expected to be gradual, with several governments and organizations deploying different components for different objectives [149]. Greater advantages of IoT can be achieved when these systems cooperate adequately. In large-scale IoT environments like Smart Cities, interoperability on the device and the network levels is essential. In other words, interworking different platforms from different systems is expected [149].

Such an approach enables many novel Smart City applications. Some of these are based on merging Smart Grid and Intelligent Transportation Systems to introduce a scheme for collecting data from smart meters installed at houses using public transportation buses [63] or public taxis [150].

When implementing an IoT system, the infrastructure of another already-established IoT system can be exploited. This is beneficial in developing countries or rural areas where establishing a dedicated communication infrastructure for each IoT system may not be feasible.

Heck *et al.* performed a case study of the Smart Grid in the city of Ipiranga, Brazil, where consumer density is low (5.66 consumers / km^2) and 60% of them live in rural areas without cell phone network coverage, which is available only in the urban part of the city [151]. Since implementing a communication infrastructure for Smart City applications in such an environment would be challenging, the paper presents GRID-CITY, a framework to enable the use of the SG communication infrastructure by Smart City applications. The motivation for such a framework is the fact that sharing the SG network with Smart City systems would enable faster application implementation in developing countries and areas with limited communication infrastructure.

In the GRID-CITY framework, a solution is presented for designing three Smart City applications in the city of Ipirange: weather monitoring, water metering, and street light control. The framework suggests connecting the devices of these applications wirelessly to the SG mesh network at the NAN level. The Ipiranga Smart Grid mesh will then connect devices from these three applications with their Smart City application servers via the backhaul of the Smart Grid network.

To achieve interoperability between the Smart City and the Smart Grid devices, GRID-CITY suggests using the Wireless Smart Ubiquitous Networks (Wi-SUN) specification at the communication layer of the NANs.

The main issue with GRID-CITY is that it does not provide a clear network topology in its framework and does not address how cooperation among Smart City networks can be established and maintained.

6.1.2. Real-world Smart City testbeds

Multiple IoT Smart City implementations for research purposes already exist. Their infrastructures and designs have many similarities, such as nodes supporting various radios.

City of Things is a real-world Smart City testbed implemented in the city of Antwerp, Belgium [152]. Its goal is to provide a living lab design to perform experiments and application development in realistic city environments.

The network layer of the City of Things testbed is an IoT infrastructure composed of sensors and gateways installed throughout the city.

The City of Things gateways can be used in two modes simultaneously:

- Infrastructure mode: The gateway functions as an access point that connects various types of sensors to the internet wirelessly. Each gateway supports multiple wireless technologies, such as IEEE 802.15.4, IEEE 802.15.4g, Bluetooth, WiFi, and others. Which allows connecting both long-range and close-range sensors.
- Ad-hoc peer-to-peer mode: City of Things gateways are interconnected via the high-speed fiber network of the city, which creates a heterogeneous mesh network covering the whole city.

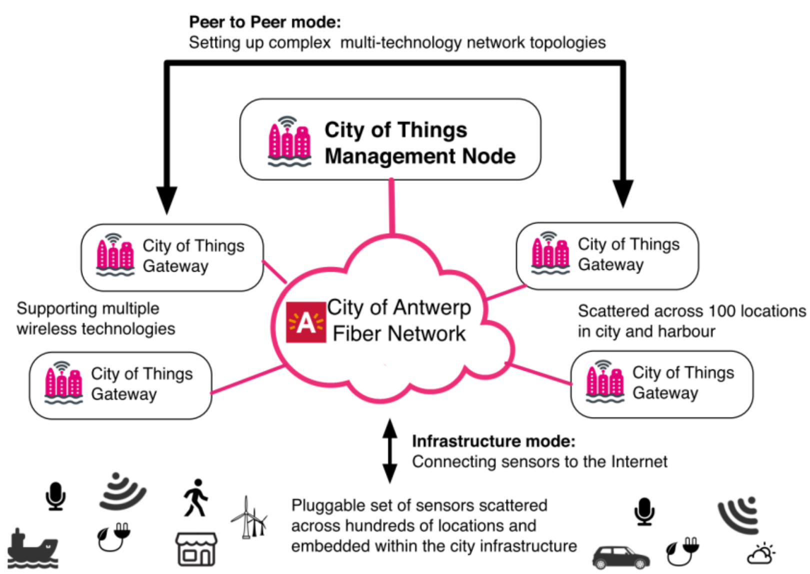


Figure 6.1: City of Things gateways with their two modes of operation: Peer-to-Peer and infrastructure [152]

City of Things is a practical IoT implementation that focuses on enabling multi-application support in its nodes rather than the interworking of its networks and instances.

6.1.3. Multi-gateway IoT Environments

In networks where nodes generate application data at high rates, forwarding packets within the same instance toward a single gateway could increase congestion and degrade the instance's performance.

To tackle this problem, Taghizadeh *et al.* introduced Enhanced RPL for Multi-gateway IoT environments (EM-RPL) [153]. EM-RPL is based on RPL with an Anycast routing extension, where a source node sends data to anyone from a set of destinations rather than just a specific one.

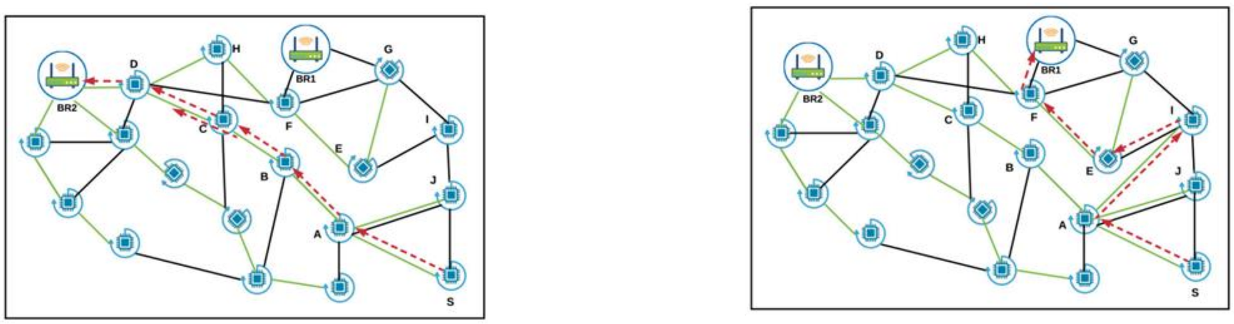
In the standard RPL, each instance acts as an independent network with its own gateways and routing mechanisms, where a node can only communicate with nodes and gateways that belong to the same instance. As a result, gateways from other instances are neither reachable nor considered as destinations by the sender. Taghizadeh *et al.* assume an inter-instance routing scheme where the set of destinations in EM-RPL includes gateways from the same instance of the sender, as well as gateways from other instances reachable by the sender.

The motivation for such a design is that in the IoT paradigm, the role of a gateway is not to process application data received from the nodes but rather to route it to the system user/admin via the internet. That is the reason for calling RPL gateways Broder Routers. EM-RPL was designed for applications where data should reach the gateway with the most optimal route rather than a pre-defined one. An example of such is a firefighting application where the sender needs the alarm data to be delivered to any firefighter, not a specific one.

Figure 6.2 demonstrates the difference between the standard RPL and EM-RPL, where Instance-1 routes packets for application-1, has the gateway BR1, and is represented by black lines. Similarly, Instance-2 routes packets for application-2, has the gateway BR2, and is represented by green lines. Some nodes belong to both instances.

In a standard RPL network displayed in Figure 6.2 (a), the source node (S) sends application-2 packets to BR2. Since BR2 is only reachable via Instance-2 connections, the only possible path from S to BR2 is:

S → A → B → C → D → BR2



(a) (b)

Figure 6.2: Different routing methods. (a) The standard RPL. (b) EM-RPL [153].

The EM-RPL scheme is portrayed in Figure 6.2 (b), where the Anycast support gives node S the option of sending Application-2 data to either BR1 or BR2. Using a reactive routing process, node S sends routing requests to all available gateways and decides, based on their routing replies, which path to which gateway is the least congested. The optimal path in this example is:

S → A → J → I → E → F → BR1

In this path, packets are routed via nodes that belong to both instances all the way to BR1.

A vital shortcoming of EM-RPL is that it does not consider cooperation among its instances. It just assumes that all instances and gateways are accessible by all nodes without restrictions. In situations where each instance belongs to a different authority, expecting that nodes of one instance can send their packets via another instance without restrictions is not reasonable. EM-RPL needs a mechanism for cooperation management among its instances.

6.2. Problem statement and contributions

IoT networks are expected to cover vast geographical regions while comprising a huge number of nodes and numerous gateways (BRs).

Typically, RPL nodes send their gathered data to a single destination. However, sometimes the path towards that destination may not be in an optimal condition. Notably, under heavy traffic, RPL networks become prone to excessive energy consumption and data loss. As a result, sending packets to a specific destination could cause more congestion and packet drops. The situation becomes even worse when the sender is located very far from the gateway, considering that the packets have to traverse many hops and could be dropped on the way.

In locations where another gateway from another instance is closer to the sender than its own, forwarding packets via that closer gateway would be beneficial, especially if the sender is reporting sensitive or alarm data. In this case, the challenge is how to manage inter-instance cooperation in a distributed, proactive manner.

Our proposed framework targets environments where gateways do not exist in the same plane or in the vicinity of each other. Compared to all the topologies assumed in research on multi-instance RPL, such as the one shown in Figure 6.2, we study topologies where each gateway lays by the end (furthest hop) of the other's network, as shown in Figure 6.3. Compared to the related work discussed in Section 6.1, we indicate these common points with our proposed framework:

- Our design focuses on scenarios indicated by the GRID-CITY framework, where multiple Smart City systems coexist and can share their networks with each other. Based on the examples presented by GRID-CITY, interconnecting devices from one Smart City application to another is encouraged if these devices are located further from their gateway and closer to a gateway of another Smart City network. That case, according to GRID-CITY, is quite common in Smart City implementations in rural areas and developing countries.
- Our suggested framework has a similar infrastructure as the City of Things testbed. In which, nodes can communicate with other nodes from other networks, and the gateways are interconnected by high-speed backbone links with much higher data rates than the IEEE 802.15.4 links (tens of Mbps and higher vs. 250 kbps).

- We share the same design motivation as EM-RPL. Forwarding packets to gateways from other instances can be favorable, often when congestion occurs in the network.

Our framework, however, addresses the following open issues in the research on multi-instance RPL:

- The majority of multi-instance RPL literature assumes the existence of only one gateway, while we target multi-gateway environments.
- Cooperation among instances is not considered.
- Topologies where gateways are not located in the same region are not studied.

The contributions of our proposed framework can be summarized as follows:

- Designing an inter-instance cooperation mechanism using virtual credits. The proposed scheme is proactive and distributed, which is a requirement of many IoT applications.
- The design targets RPL topologies not considered before. Particularly where asymmetric cooperation is needed.

Figure 6.3 shows an example of the multi-instance RPL topologies our framework studies. In this Figure, n_1^* is a node from Instance1 that receives cooperation requests from node n2, which is in its range and belongs to Instance2. Similarly, n_2^* from Instance2 receives cooperation requests from n1 which belongs to Instance1.

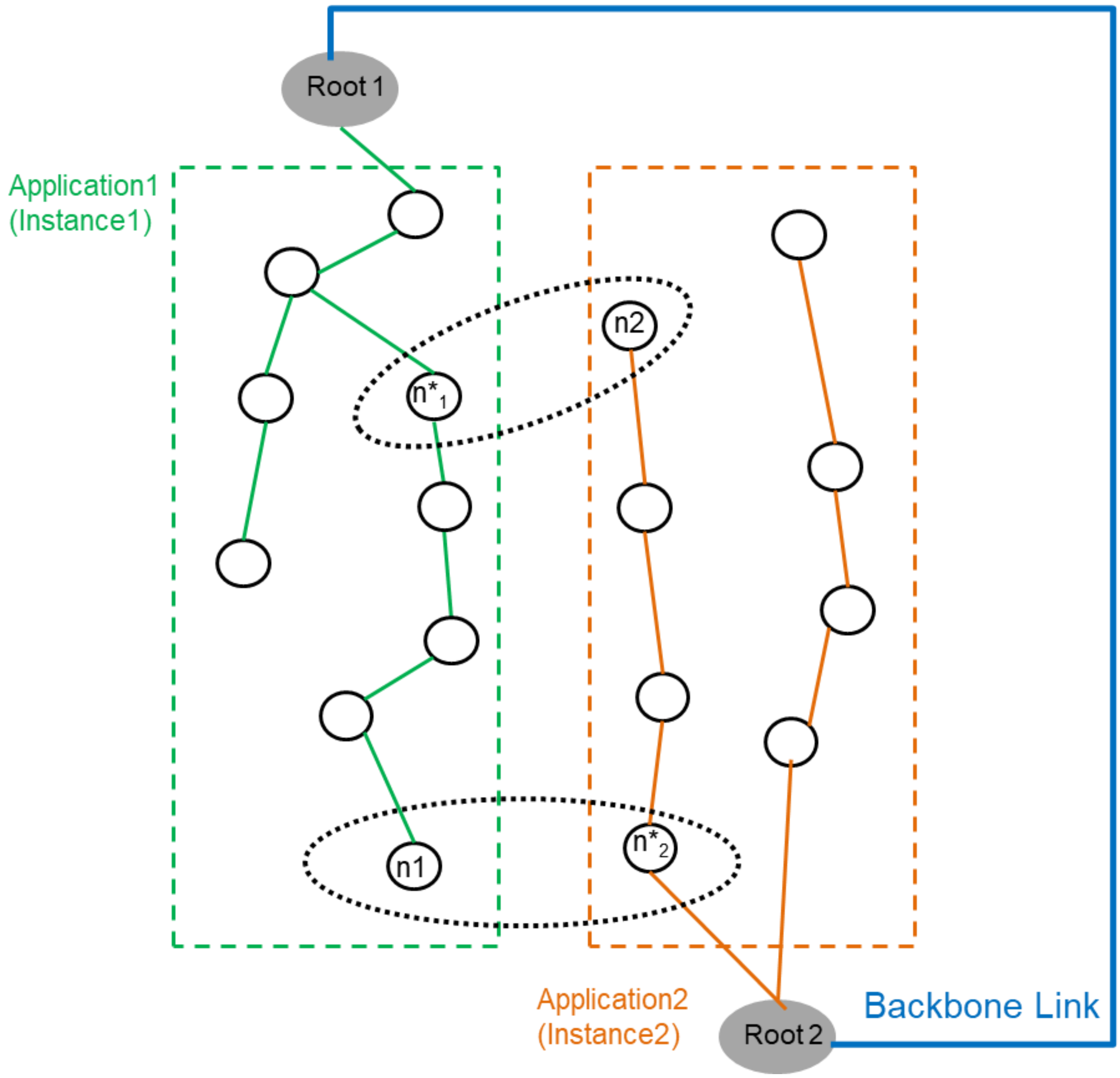


Figure 6.3: A multi-instance RPL topology used by our proposed framework

6.3. Framework design

Our proposed framework design consists of four steps: congestion estimation, using a virtual currency for sending inter-instance packets, gateway-credit update via backbone interconnections, and instance-credit update.

6.3.1. Step 1: path-congestion estimation

We introduce our novel Path Congestion Metric (PCM), which is the sum of the queue lengths at the Queue buffers of all the intermediate (n) nodes on the path to a root divided by the Hop Count. PCM represents the average per-hop congestion and is given by equation (6-1):

$$PCM = \frac{\sum_{j=1}^{n} Q_j}{HC} \tag{6-1}$$

A node that belongs to Instance1 (i_1) decides that it is better to send its packets via Instance2 (i_2) if the path toward the root of Instance2 is less congested than the path to the root of instance1:

$$PCM_{i2} < PCM_{i1} \tag{6-2}$$

In this case, if the node has enough credit to send a packet via the other instance, it can initiate sending the packet, as shown in Figure 6.4, where the node n1 wants to send a packet via the node n2 of Instance2 all the way to BR2.

6.3.2. Step 2: sending a packet via another instance

We use a virtual currency system to control the number of inter-instance packets a node can send. Each node has two balances:

- Cr: Credit for the packets sent via another instance.
- Cf: Credit for the packets received from another instance and forwarded by the node within its own instance.

The previous values of a node's Cr and Cf are referred to as Cr´ and Cf ´, respectively.

In our framework, nodes within an instance keep track of the number of inter-instance packets they forward for each of their children. We call this number the Child's Forwarded Packets (CFPs). In other words, the CFPs number enables a parent to count the number of inter-instance packets it forwards and also record which child they came from.

It can be observed in Figure 6.4 that when node n1 sends a packet, its credit for sending packets (Cr) is decreased by one, while its forwarding credit (Cf) remains unchanged since n1 is not forwarding inter-instance packets at this phase:

$$Cr_{n1} = Cr'_{n1} - 1$$

$$Cf_{n1} = Cf'_{n1}$$

The opposite happens at n2, with Cf increasing by one and Cr not changing its value:

$$Cr_{n2} = Cr'_{n2}$$

$$Cf_{n2} = Cf'_{n2} + 1$$

As the inter-instance packet propagates through Instance2, each parent node that forwards it increases the CFPs score of the child node it came from. As can be seen in Figure 6.4, n3 forwards the inter-instant packet from its child, n2, and increases n2's CFPs by one. Similarly, n4 also increases n2's CFPs by one after forwarding its packet.

It is worth noting that in Figure 6.4 the number CFPs´ is the previous value of CFPs.

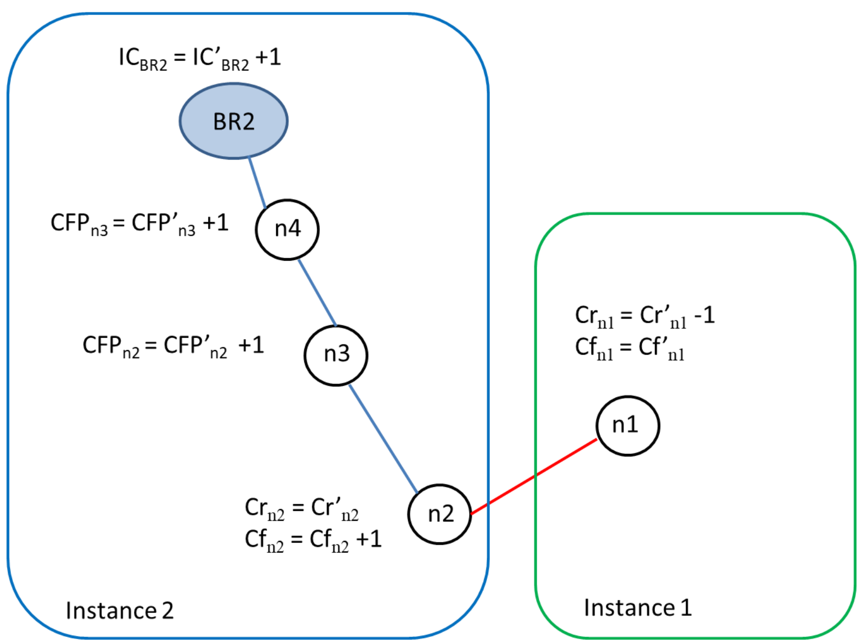


Figure 6.4: Earnings and costs of inter-instance packet forwarding

6.3.3. Step 3: Credit update between gateways

Each gateway counts all the inter-instance packets it receives from the nodes of its network. We call this number the Instance Credit (IC). As shown in Figure 6.4, IC is increased by one with each reception of an inter-instance packet. In this Figure, we can observe the IC update for BR2:

$$IC_{BR2} = IC'_{BR2} + 1$$

Where IC_{BR2} is the Instance Credit of BR2, and IC'_{BR2} is the previous value of IC_{BR2}.

As Figure 6.5 indicates, both gateways periodically exchange their Instance Credits via a backbone link. Just like the City of Things Smart City testbed [152], we assume that the backbone link is a high-speed fiber interconnection or something similar, which makes exchanging ICs a rapid process. We call each exchange of the IC values between two BRs an IC exchange round (r).

After each round, each gateway subtracts the IC of the other gateway from its own to get the value of its IC for the current round:

$$IC^{ri}_{BR2} = IC_{BR2} - IC_{BR1}$$

Where IC^{ri}_{BR2} is the IC value of BR2 at the i^{th} exchange round.

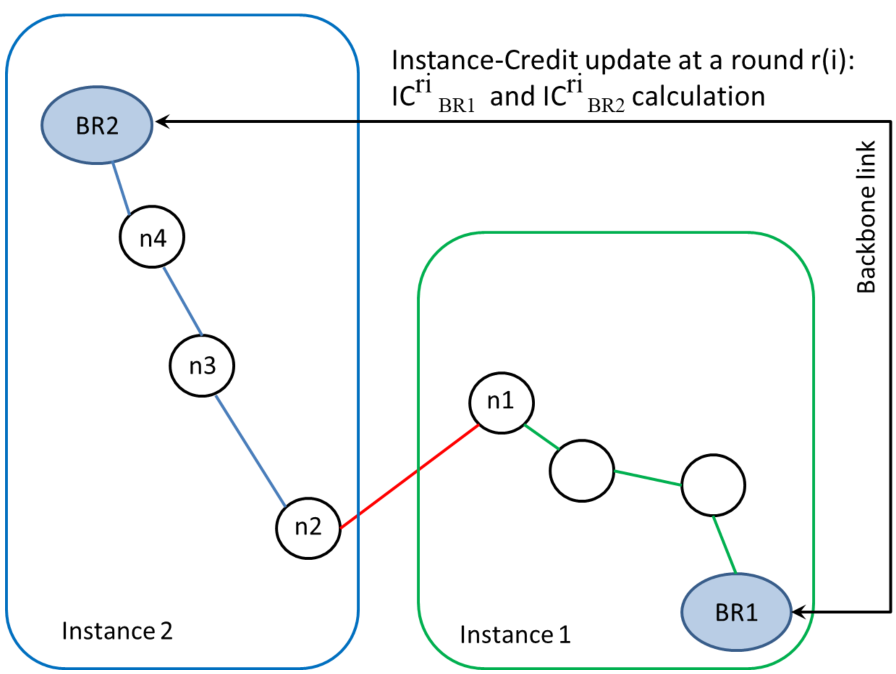


Figure 6.5: Instance-Credit update between gateways

6.3.4. Step 4: Propagating credit updates downwards

After each cooperation round, each gateway disseminates the Instance Credit (IC^{ri}) value of the other gateway throughout its network. A parent distributes the received Instance Credit among its child nodes proportionally based on the number of inter-instance packets they have forwarded, i.e., based on their CFPs. This way, we guarantee that nodes that do not forward inter-instance packets do not receive credits, which is the case for nodes that are not located in the range of nodes from other instances. The tuple (Node_id, credit) is used to indicate which node gets how many credits. These Instance Credit values of the other instance keep propagating downwards until they reach the nodes of the other instance. The IC updates are carried within DIO messages, i.e., they are broadcasted with the routing metric(s) periodically.

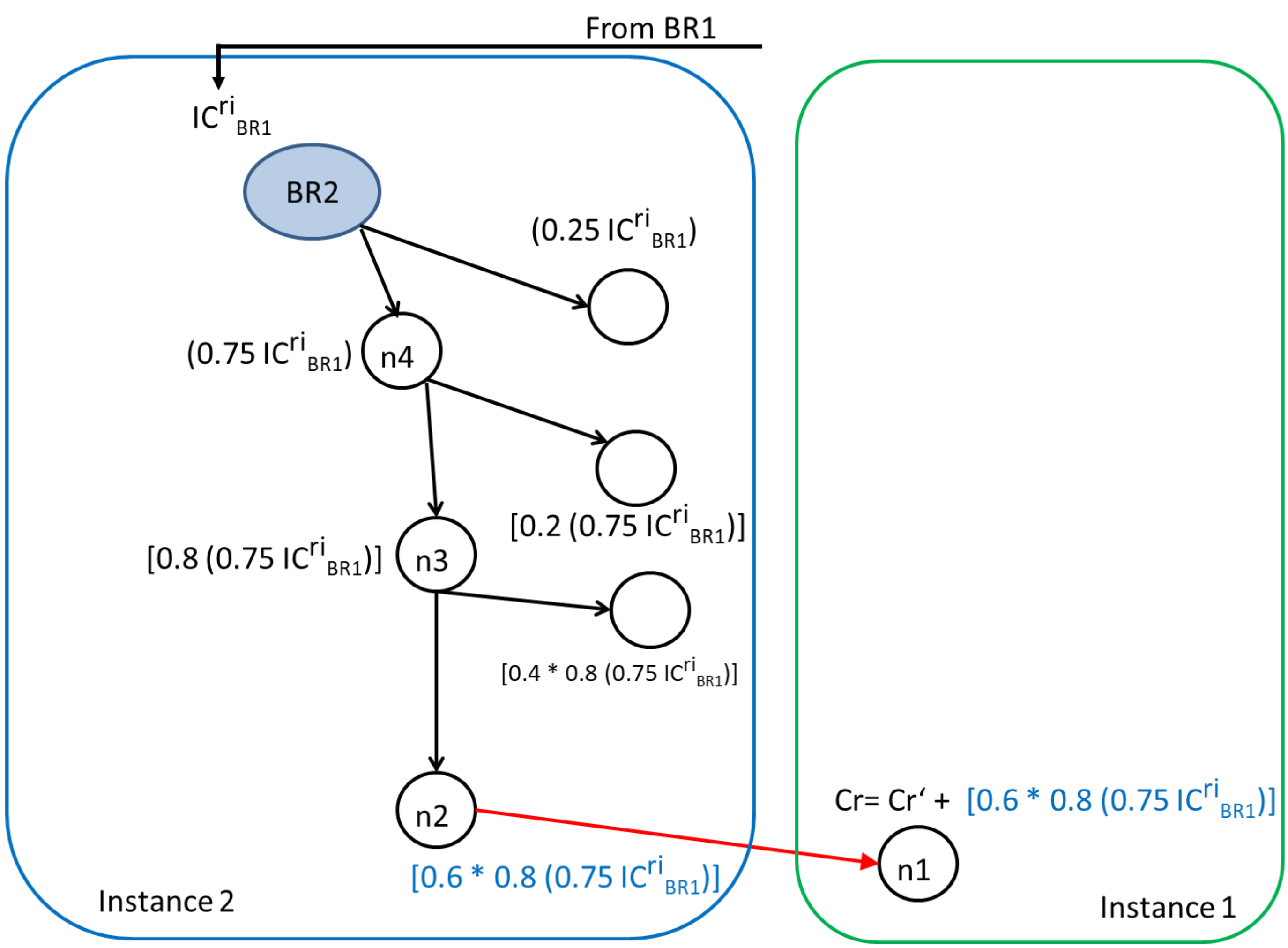


Figure 6.6: An example of disseminating credit updates

Figure 6.6 portrays an example of how the IC can be distributed. We assume that all Instance2 nodes in this Figure are within the vicinity of some nodes from Instance1, and therefore IC^{ri}_{BR1} is divided among these Instance2 nodes and sent from them to nodes in Instance1. Figure 6.6 illustrates how the node n1 gains new credit from BR1 via BR2 after a cooperation round (r^i). Assuming $IC^{ri}_{BR1} > 0$, this credit propagates from BR1 to BR2 via the backbone link, and from BR2 downwards through Instance2 via DIO messages to n2. Each receiving node on the path downward distributes the credits based on the CFPs of its children. Node n3, for example, gives 60% of the credit it receives to n2 assuming that n3 receives 60% of the inter-instance forwarding requests from n2. Finally, n2 sends the final credit value to n1, which adds it to its Cr balance to use for requesting to send inter-instance packets via n2.

6.4. Summary

In this Chapter, we have examined several designs for a Smart City composed of multiple subsystems, each having different RPL instances and belonging to a distinct authority. Some researchers propose that networks of these systems should share their infrastructures to improve their scalability. Others suggest that it would be optimal in some cases that leaf nodes from one instance send their packets to the root of another.

We have presented our framework that relies on a virtual currency to facilitate cooperation among RPL instances that belong to different authorities. Our design establishes a distributed mechanism for nodes located further from their gateway to cooperate with nodes closer to a gateway of another instance. This mechanism enables such nodes to use a virtual currency they obtain from their instance to forward their packets via another instance.

7. Conclusion and future work

This Chapter concludes the Thesis and outlines its contributions. In addition, it suggests potential directions for future research.

7.1. Conclusion

In this Thesis, we have reviewed LLNs and examined their properties. Due to their limited resources, integrating LLNs into IoT faces numerous challenges. RPL was standardized to facilitate such integration by enabling IPv6 routing in LLNs while still considering their constrained nature.

IoT environments typically incorporate applications with different QoS requirements. We have illustrated an example of such a case in Table 2.1, where we showed that the LLNs that are deployed in the NAN segment of the SG have to deal with routing heterogeneous traffic with different priorities. RPL offers a solution for service differentiation in IoT networks by allocating different instances to route different traffic classes.

Our research focuses on multi-instance RPL for service differentiation in IoT networks. We investigated congestion mitigation via cooperation among RPL instances. Our work was motivated by the idea that, under heavy and dynamic traffic, it would be beneficial to exploit the path diversity offered by RPL's multiple instances for congestion control. To facilitate that, a mechanism for cooperation among RPL instances is needed.

Our research domain overlaps the domains of congestion control and cooperation among multiple RPL instances. The standard RPL offers no solutions in either of these two domains; it does not address congestion and suffers from the problem of under-specification regarding instance cooperation. We have also explored the research efforts for improving RPL in these two domains.

We have provided an extensive study of the proposed congestion control schemes for RPL. We outlined the main drawbacks of using a composite-metric approach for congestion control: lack of interoperability, complexity, overhead, and instability. We designed DMC-RPL to tackle these issues.

In DMC-RPL, we have introduced a path-congestion estimation metric. The simple calculations of our proposed metric make it less complex compared to other composite metrics. At the same time, it was designed to be uninvolved in the parent-selection process to avoid introducing instability to the network. The proposed metric was also designed to be interoperable.

We explored the multi-instance research domain of RPL and found that only one paper proposed a model for cooperation between RPL instances, which is reactive and centralized. We have listed the problems with centralized approaches, namely the problems with storing mode (memory overhead) and non-storing mode (source routing header overhead). We designed DMC-RPL to be distributed in order to avoid these problems. The proposed Q path metric is used in DMC-RPL to detect congestion and initiate cooperation with nodes from other instances locally.

The reactive cooperation management models proposed for RPL are not suitable for alarm applications. DMC-RPL tackles the problem of selfishness among cooperating nodes by using a novel cooperation control procedure. The distributed and reactive design of this procedure makes it suitable for handling sporadic traffic.

Based on the features of DMC-RPL listed above, we can say that its design is the answer to our first three research questions listed in Section 1.3: it is distributed (RQ1), aims to alleviate congestion (RQ2), and has a distributed mechanism to deal with selfish nodes (RQ3).

We evaluated DMC-RPL under heavy and dynamic traffic for two types of AMI applications that generate periodic and sporadic traffic.

Simulation results for the first scenarios with two identical instances have shown good improvement for DMC-RPL in terms of PDR, especially under light to moderate traffic of 4-6 pkts/minute. With the increase of the traffic rate, the performance of DMC-RPL was still better than the standard RPL. The packet drops at the buffer level were less, due to the queue length being the main component of DMC-RPL's congestion metric. However, at high data rates, the power consumption of DMC-RPL was higher compared to the standard RPL due to the excessive control-messages exchange needed to notify about congestion, and due to packets being forwarded by DMC-RPL more than the standard RPL, which dropped more packets, spending less energy on forwarding them compared to DMC-RPL.

Simulations performed using two different instances have shown similar trends. At high traffic rates, DMC-RPL could still perform decently in terms of PDR but at the expense of extra power consumption.

Finally, we have examined environments where multiple IoT systems coexist, as in the Smart City ecosystem. When Smart City subsystems are co-located, RPL instances that belong to different authorities may interwind. We laid out a network architecture for such a scenario and proposed a proactive, distributed framework for using virtual credits to send packets via other instances.

7.2. Future work

In the future, we believe some DMC-RPL issues need to be addressed. First, a dynamic Trickle algorithm mechanism should be integrated with DMC-RPL. When a node detects congestion in DMC-RPL, it resets its Trickle timer to spread the congestion information quickly through the network. A strategy for such information dissemination should be designed to avoid excessive DIO broadcasts. Moreover, when the network is highly congested, DMC-RPL still broadcasts DIOs at high rates, even when no alternative paths are available. A mechanism that links the availability of non-congested paths with the broadcast rate of the congestion notifications should be added to DMC-RPL in the future.

We have noticed that DMC-RPL still tries to forward as many packets as possible even if the network is heavily congested. For some congested paths, it would be more helpful to drop the packets earlier rather than propagate them. Forwarding a packet through multiple hops consumes the network's resources. If such a packet is going to be dropped eventually before it even reaches the root, it would be better to drop it earlier if the application can tolerate errors. Therefore, we believe that a forwarding-cost mechanism can be helpful with DMC-RPL since it is a distributed protocol, where nodes far away from the root have to make forwarding choices locally. The feasibility of such a forwarding process should also be studied. An alternative approach would be to design a traffic control mechanism that seeks to mitigate congestion in DMC-RPL by adjusting the sender's transmission rate based on the congestion level.

DMC-RPL propagates the Q values of the Grandparent chain to be used for path-congestion estimation. In large-scale networks, the number of the Q-array's elements becomes bigger as the chain becomes longer. That's why it would be helpful to employ a compression mechanism with DMC-RPL to reduce the size of its Q-arrays. A possible solution would be to employ a context-based compression similar to IPHC [37].

Finally, our proposed framework for cooperation encouragement using virtual credits should be implemented and evaluated to judge its efficiency.

Bibliography

[1] R. Hassan, F. Qamar, M. K. Hasan, A. H. M. Aman, and A. S. Ahmed, "Internet of Things and its applications: A comprehensive survey," *Symmetry,* vol. 12, p. 1674, 2020.

[2] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *IETF RFC,* vol. 2460, p. 39, 1998.

[3] J. Vasseur, "Terms Used in Routing for Low-Power and Lossy Networks," *IETF RFC,* vol. 7102, p. 8, 2014.

[4] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis*, et al.*, "RPL: IPv6 routing protocol for low-power and lossy networks," *IETF RFC,* vol. 6550, p. 157, 2012.

[5] ITU-T, "Recommendation E. 800: Quality of Service and Dependability Vocabulary," ed: Nov, 1988.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," *IETF RFC,* vol. 2475, 1998.

[7] J. Nassar, M. Berthomé, J. Dubrulle, N. Gouvy, N. Mitton, and B. Quoitin, "Multiple instances QoS routing in RPL: Application to smart grids," *Sensors,* vol. 18, p. 2472, 2018.

[8] C. Lim, "A survey on congestion control for RPL-based wireless sensor networks," *Sensors,* vol. 19, p. 2567, 2019.

[9] Y. Tahir, S. Yang, and J. McCann, "BRPL: Backpressure RPL for high-throughput and mobile IoTs," *IEEE Transactions on Mobile Computing,* vol. 17, pp. 29-43, 2017.

[10] M. Barcelo, A. Correa, J. L. Vicario, and A. Morell, "Cooperative interaction among multiple RPL instances in wireless sensor networks," *Computer Communications,* vol. 81, pp. 61-71, 2016.

[11] Z. Shelby and C. Bormann, *6LoWPAN: the wireless embedded internet* vol. 43. U.K.: John Wiley & Sons, 2011.

[12] O. Iova, P. Picco, T. Istomin, and C. Kiraly, "Rpl: The routing standard for the internet of things... or is it?," *IEEE Communications Magazine,* vol. 54, pp. 16-22, 2016.

[13] M. Dohler, D. Barthel, T. Watteyne, and T. Winter, "Routing requirements for urban low-power and lossy networks," *IETF RFC,* vol. 5548, p. 21, 2009.

[14] S. Taghizadeh, H. Bobarshad, and H. Elbiaze, "CLRPL: context-aware and load balancing RPL for Iot networks under heavy and highly dynamic load," *IEEE Access,* vol. 6, pp. 23277-23291, 2018.

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

[16] H.-S. Kim, H. Kim, J. Paek, and S. Bahk, "Load balancing under heavy traffic in RPL routing protocol for low power and lossy networks," *IEEE Transactions on Mobile Computing,* vol. 16, pp. 964-979, 2016.

[17] A. M. Zungeru, J. M. Chuma, C. K. Lebekwe, P. Phalaagae, and J. Gaboitaolelwe, *Green Internet of Things Sensor Networks: Applications, Communication Technologies, and Security Challenges*: Springer, 2020.

[18] S. Kraijak and P. Tuwanut, "A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends," in *11th international conference on wireless communications, networking and mobile computing (WiCOM 2015)*, 2015, pp. 1-6.

[19] S. S. H. Hajjaj and K. R. Gsangaya, *The Internet of Mechanical Things: The IoT Framework for Mechanical Engineers*: CRC Press, 2022.

[20] J. Wang, M. K. Lim, C. Wang, and M.-L. Tseng, "The evolution of the Internet of Things (IoT) over the past 20 years," *Computers & Industrial Engineering,* vol. 155, p. 107174, 2021.

[21] A. Rayes and S. Salam, "Internet of things from hype to reality," *The road to Digitization,* vol. 2, 2017.

[22] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," 2070-1721, 2014.

[23] IEEE, *IEEE Standard for Local and metropolitan area networks-Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*: IEEE Std 802.15.4™, 2011.

[24] E. Lopez-Aguilera, I. Demirkol, E. Garcia-Villegas, and J. Paradells, "IEEE 802.11-enabled wake-up radio: Use cases and applications," *Sensors,* vol. 20, p. 66, 2019.

[25] L. Lampe, A. M. Tonello, and T. G. Swart, *Power Line Communications: Principles, Standards and Applications from multimedia to smart grid*: John Wiley & Sons, 2016.

[26] N. Kushalnagar, G. Montenegro, and C. Schumacher, "IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals," *IETF RFC,* vol. 4919, p. 12, 2007.

[27] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15. 4 networks," *IETF RFC,* vol. 4944, p. 30, 2007.

[28] J. J. Rodrigues and P. A. Neves, "A survey on IP-based wireless sensor network solutions," *International Journal of Communication Systems,* vol. 23, pp. 963-981, 2010.

[29] J. Postel, "Internet protocol," vol. 0791, 1981.

[30] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," *IETF RFC,* vol. 4862, p. 30, 2007.

[31] J. Moy, "OSPF Version 2," *IETF RFC,* vol. 2328, 1998.

[32] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," *IETF RFC,* vol. 4271, 2006.

[33] R. Fielding and J. Reschke, "Hypertext transfer protocol (HTTP/1.1): Semantics and content," *IETF RFC,* vol. 7231, 2014.

[34] J. Klensin, "Simple mail transport protocol," *IETF RFC,* vol. 5321, 2008.

[35] S. Frankel and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap," *IETF RFC,* vol. 6071, 2011.

[36] IPv6 over Low power WPAN (6lowpan). Available: https://datatracker.ietf.org/wg/6lowpan/about (accessed Dec. 2022).

[37] J. Hui and P. Thubert, "Compression format for IPv6 datagrams over IEEE 802.15. 4-based networks," *IETF RFC,* vol. 6282, p. 24, 2011.

[38] E. Kim and D. Kaspar, "Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)," *IETF RFC,* vol. 6568, p. 28, 2012.

[39] C. Gomez, E. Kim, D. Kaspar, and C. Bormann, "Problem statement and requirements for IPv6 over low-power wireless personal area network (6LoWPAN) routing," *IETF RFC,* vol. 6606, p. 32, 2012.

[40] Z. Shelby, S. Chakrabarti, E. Nordmark, and C. Bormann, "Neighbor discovery optimization for IPv6 over low-power wireless personal area networks (6LoWPANs)," *IETF RFC,* vol. 6775, p. 55, 2012.

[41] T. Narten, E. Nordmark, W. A. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," *IETF RFC* vol. 4861, p. 97, 2007.

[42] IPv6 over Networks of Resource-constrained Nodes (6lo). Available: https://datatracker .ietf.org/wg/6lo/about/ (accessed Dec. 2022)

[43] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, "IPv6 over Bluetooth Low Energy," *IETF RFC,* vol. 7668, 2015.

[44] Y. Hong, Y. Choi, J. Youn, D. Kim, and J. Choi, "Transmission of IPv6 packets over near field communication " *IETF, draft-ietf-6lo-nfc-17,* 2020, Internet Draft (Work in Progress, [Online], Available: https://datatracker.ietf.org/doc/html/draft-ietf-6lo-nfc-20. (accessed Dec. 2022).

[45] J. Hou, B. Liu, X. Tang, Y. Hong, and C. Perkins, "Transmission of IPv6 packets over PLC networks," *IETF, draft-ietf-6lo-plc-11,* 2022, Internet Draft (Work in Progress), [Online], Available: https://datatracker.ietf.org/doc/html/draft-ietf-6lo-plc-11. (accessed Dec. 2022).

[46] IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH). Available: https://datatracker .ietf.org/wg/6tisch/about/ (accessed Dec. 2022)

[47] IPv6 over Low Power Wide-Area Networks (LPWAN). Available: https://datatracker .ietf.org/wg/lpwan/about/ (accessed Dec. 2022)

[48] Routing Over Low power and Lossy networks (ROLL). Available: https://datatracker .ietf.org/wg/roll/about/ (accessed Dec. 2022)

[49] K. Pister, P. Thubert, S. Dwars, and T. Phinney, "Industrial Routing Requirements in Low-Power and Lossy Networks," *IETF RFC,* vol. 5673, p. 27, 2009.

[50] A. Brandt and J. Buron, "Home automation routing requirements in low-power and lossy networks," *IETF RFC,* vol. 5826, p. 17, 2010.

[51] J. Martocci, P. Mil, N. Riou, and W. Vermeylen, "Building automation routing requirements in low-power and lossy networks," *IETF RFC,* vol. 5867, p. 26, 2010.

[52] J. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel, "Routing metrics used for path calculation in low power and lossy networks," *IETF RFC,* vol. 6551, p. 30, 2012.

[53] P. Thubert, "Objective function zero for the routing protocol for low-power and lossy networks (RPL)," *IETF RFC,* vol. 6552, p. 14, 2012.

[54] O. Gnawali and P. Levis, "The minimum rank with hysteresis objective function," *IETF RFC,* vol. 6719, p. 13, 2012.

[55] P. Levis, T. H. Clausen, J. Hui, O. Gnawali, and J. Ko, "The trickle algorithm," *IETF RFC,* vol. 6206, p. 13, 2011.

[56] 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," *IETF RFC,* vol. 7733, 2016.

[57] J. Hui, D. Popa, R. Salazar, N. Dejean, and J. Jetcheva, " Applicability Statement for the Routing Protocol for Low-Power and Lossy Networks (RPL) in Advanced Metering Infrastructure (AMI) Networks," *IETF RFC,* vol. 8036, 2017.

[58] A. Monzon, "Smart cities concept and challenges: Bases for the assessment of smart city projects," in *2015 international conference on smart cities and green ICT systems (SMARTGREENS)*, 2015, pp. 1-11.

[59] L. Anthopoulos, M. Janssen, and V. Weerakkody, "A Unified Smart City Model (USCM) for smart city conceptualization and benchmarking," *Smart cities and smart spaces: Concepts, methodologies, tools, and applications,* pp. 247-264, 2019.

[60] C. Harrison, B. Eckman, R. Hamilton, P. Hartswick, J. Kalagnanam, J. Paraszczak*, et al.*, "Foundations for smarter cities," *IBM Journal of research and development,* vol. 54, pp. 1-16, 2010.

[61] S. Zeebaree, S. Ameen, and M. Sadeeq, "Social media networks security threats, risks and recommendation: A case study in the kurdistan region," *International Journal of Innovation, Creativity and Change,* vol. 13, pp. 349-365, 2020.

[62] M. Sadeeq, A. I. Abdulla, A. S. Abdulraheem, and Z. S. Ageed, "Impact of electronic commerce on enterprise business," *Technol. Rep. Kansai Univ,* vol. 62, pp. 2365-2378, 2020.

[63] B. E. Bilgin, S. Baktir, and V. C. Gungor, "Collecting smart meter data via public transportation buses," *IET Intelligent Transport Systems,* vol. 10, pp. 515-523, 2016.

[64] A. B. Haque, B. Bhushan, and G. Dhiman, "Conceptualizing smart city applications: Requirements, architecture, security issues, and emerging trends," *Expert Systems,* vol. 39, p. e12753, 2022.

[65] H. Habibzadeh, T. Soyata, B. Kantarci, A. Boukerche, and C. Kaptan, "Sensing, communication and security planes: A new challenge for a smart city system design," *Computer Networks,* vol. 144, pp. 163-200, 2018.

[66] S. Jino Ramson, D. Jackuline Moni, A. Alfred Kirubaraj, and S. Senith, "Self-powered wireless sensor network framework to monitor bin level," *The Journal of Solid waste technology and management,* vol. 43, pp. 295-304, 2017.

[67] S. J. Ramson and D. J. Moni, "Wireless sensor networks based smart bin," *Computers & Electrical Engineering,* vol. 64, pp. 337-353, 2017.

[68] D. Singh, G. Tripathi, and A. J. Jara, "A survey of Internet-of-Things: Future vision, architecture, challenges and services," in *2014 IEEE world forum on Internet of Things (WF-IoT)*, 2014, pp. 287-292.

[69] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials,* vol. 17, pp. 2347-2376, 2015.

[70] K. Kiela, V. Barzdenas, M. Jurgo, V. Macaitis, J. Rafanavicius, A. Vasjanov*, et al.*, "Review of V2X–IoT standards and frameworks for ITS applications," *Applied Sciences,* vol. 10, p. 4314, 2020.

[71] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE internet of things journal,* vol. 1, pp. 289-299, 2014.

[72] F. Bai and B. Krishnamachari, "Exploiting the wisdom of the crowd: localized, distributed information-centric VANETs [Topics in Automotive Networking]," *IEEE communications magazine,* vol. 48, pp. 138-146, 2010.

[73] S. Rekik, N. Baccour, M. Jmaiel, and K. Drira, "Wireless sensor network based smart grid communications: Challenges, protocol optimizations, and validation platforms," *Wireless Personal Communications,* vol. 95, pp. 4025-4047, 2017.

[74] Y. Cunjiang, Z. Huaxun, and Z. Lei, "Architecture design for smart grid," *Energy Procedia,* vol. 17, pp. 1524-1528, 2012.

[75] G. Dileep, "A survey on smart grid technologies and applications," *Renewable energy,* vol. 146, pp. 2589-2625, 2020.

[76] M. Erol-Kantarci and H. T. Mouftah, "Smart grid forensic science: applications, challenges, and open issues," *IEEE Communications Magazine,* vol. 51, pp. 68-74, 2013.

[77] W. Meng, R. Ma, and H.-H. Chen, "Smart grid neighborhood area networks: a survey," *IEEE Network,* vol. 28, pp. 24-32, 2014.

[78] E. Kabalci and Y. Kabalci, *Smart grids and their communication systems*: Springer, 2019.

[79] M. Kuzlu, M. Pipattanasomporn, and S. Rahman, "Communication network requirements for major smart grid applications in HAN, NAN and WAN," *Computer Networks,* vol. 67, pp. 74-88, 2014.

[80] V. C. Gungor, B. Lu, and G. P. Hancke, "Opportunities and challenges of wireless sensor networks in smart grid," *IEEE transactions on industrial electronics,* vol. 57, pp. 3557-3564, 2010.

[81] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati*, et al.*, "Smart grid technologies: Communication technologies and standards," *IEEE transactions on Industrial informatics,* vol. 7, pp. 529-539, 2011.

[82] F. Bouhafs, M. Mackay, and M. Merabti, "Links to the future: Communication requirements and challenges in the smart grid," *IEEE Power and Energy Magazine,* vol. 10, pp. 24-32, 2011.

[83] E. Sarker, P. Halder, M. Seyedmahmoudian, E. Jamei, B. Horan, S. Mekhilef*, et al.*, "Progress on the demand side management in smart grid and optimization approaches," *International Journal of Energy Research,* vol. 45, pp. 36-64, 2021.

[84] V. Preethi and G. Harish, "Design and implementation of smart energy meter," in *2016 International Conference on Inventive Computation Technologies (ICICT)*, 2016, pp. 1-5.

[85] D. Li, Z. Aung, J. R. Williams, and A. Sanchez, "Efficient and fault-diagnosable authentication architecture for AMI in smart grid," *Security and Communication Networks,* vol. 8, pp. 598-616, 2015.

[86] A. Mahmood, N. Javaid, and S. Razzaq, "A review of wireless communications for smart grid," *Renewable and sustainable energy reviews,* vol. 41, pp. 248-260, 2015.

[87] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE communications surveys & tutorials,* vol. 15, pp. 5-20, 2012.

[88] H. Su, M. Qiu, and H. Wang, "Secure wireless communication system for smart grid with rechargeable electric vehicles," *IEEE Communications Magazine,* vol. 50, pp. 62-68, 2012.

[89] X. S. Shen, "Empowering the smart grid with wireless technologies [editor's note]," *IEEE Network,* vol. 26, pp. 2-3, 2012.

[90] F. Cunha, L. Villas, A. Boukerche, G. Maia, A. Viana, R. A. Mini*, et al.*, "Data communication in VANETs: Protocols, applications and challenges," *Ad Hoc Networks,* vol. 44, pp. 90-103, 2016.

[91] U. Zafar, S. Bayhan, and A. Sanfilippo, "Home energy management system concepts, configurations, and technologies for the smart grid," *IEEE access,* vol. 8, pp. 119271-119286, 2020.

[92] M. Soshinskaya, W. H. Crijns-Graus, J. M. Guerrero, and J. C. Vasquez, "Microgrids: Experiences, barriers and success factors," *Renewable and sustainable energy reviews,* vol. 40, pp. 659-672, 2014.

[93] S. P. Bihari, P. K. Sadhu, K. Sarita, B. Khan, L. Arya, R. Saket*, et al.*, "A comprehensive review of microgrid control mechanism and impact assessment for hybrid renewable energy integration," *IEEE Access,* 2021.

[94] M. Y. Worku, M. A. Hassan, and M. A. Abido, "Real time energy management and control of renewable energy based microgrid in grid connected and island modes," *Energies,* vol. 12, p. 276, 2019.

[95] S. Rekik, N. Baccour, M. Jmaiel, and K. Drira, "Wireless sensor network based smart grid communications: Challenges, protocol optimizations, and validation platforms," *Wireless Personal Communications,* vol. 95, pp. 4025-4047, 2017.

[96] OpenSG, "Smart Grid Networks System Requirements Specification Release Version 5 Final," *SG-Network task force* 2012.

[97] S. Kumar and S. Solanki, "Remote home surveillance system," in *2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA)(Spring)*, 2016, pp. 1-4.

[98] M. G. Gnoni, P. A. Bragatto, M. F. Milazzo, and R. Setola, "Integrating IoT technologies for an “intelligent” safety management in the process industry," *Procedia manufacturing,* vol. 42, pp. 511-515, 2020.

[99] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: the internet of things architecture, possible applications and key challenges," in *2012 10th international conference on frontiers of information technology*, 2012, pp. 257-260.

[100] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad hoc networks,* vol. 10, pp. 1497-1516, 2012.

[101] A. Whitmore, A. Agarwal, and L. Da Xu, "The Internet of Things—A survey of topics and trends," *Information systems frontiers,* vol. 17, pp. 261-274, 2015.

[102] R. Morabito and J. Jiménez, "IETF protocol suite for the Internet of Things: Overview and Recent Advancements," *IEEE Communications Standards Magazine,* vol. 4, pp. 41-49, 2020.

[103] S. J. Ramson, S. Vishnu, and M. Shanmugam, "Applications of internet of things (iot)–an overview," in *2020 5th international conference on devices, circuits and systems (ICDCS)*, 2020, pp. 92-95.

[104] E. A. Shammar and A. T. Zahary, "The Internet of Things (IoT): a survey of techniques, operating systems, and trends," *Library Hi Tech,* 2019.

[105] P. Matta and B. Pant, "Internet of things: Genesis, challenges and applications," *Journal of Engineering Science and Technology,* vol. 14, pp. 1717-1750, 2019.

[106] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," *Computer Networks,* vol. 148, pp. 241-261, 2019.

[107] 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," *IEEE Communications Surveys & Tutorials,* vol. 19, pp. 2502-2525, 2017.

[108] A. Parasuram, "An analysis of the RPL routing standard for low power and lossy networks," M.S. thesis, EECS Dept., University of California at Berkeley, Berkeley, CA, USA, 2016.

[109] B. Ghaleb, A. Y. Al-Dubai, E. Ekonomou, A. Alsarhan, Y. Nasser, L. M. Mackenzie*, et al.*, "A survey of limitations and enhancements of the ipv6 routing protocol for low-power and lossy networks: A focus on core operations," *IEEE Communications Surveys & Tutorials,* vol. 21, pp. 1607-1635, 2018.

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

[111] K. D. Korte, A. Sehgal, and J. Schönwälder, "A study of the RPL repair process using ContikiRPL," in *the 6th IFIP International Conference on Autonomous Infrastructure, Management, and Security*, 2012, pp. 50-61.

[112] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Proceedings of the 31st IEEE Conference on Local Computer Networks*, 2006, pp. 641-648.

[113] M. Shirbeigi, B. Safaei, A. Mohammadsalehi, A. M. H. Monazzah, J. Henkel, and A. Ejlali, "A cluster-based and drop-aware extension of RPL to provide reliability in IoT applications," in *2021 IEEE International Systems Conference (SysCon)*, 2021, pp. 1-7.

[114] The Contiki packet buffers. Available: https://docs.contiki-ng.org/en/develop/doc/programming/Packet-buffers.html (accessed Dec. 2022)

[115] Y. Cao and M. Wu, "A novel RPL algorithm based on chaotic genetic algorithm," *Sensors,* vol. 18, p. 3647, 2018.

[116] H. A. Al-Kashoash, H. M. Amer, L. Mihaylova, and A. H. Kemp, "Optimization-based hybrid congestion alleviation for 6LoWPAN networks," *IEEE Internet of Things Journal,* vol. 4, pp. 2070-2081, 2017.

[117] A. Ghaffari, "Congestion control mechanisms in wireless sensor networks: A survey," *Journal of network and computer applications,* vol. 52, pp. 101-115, 2015.

[118] R. Ullah, Y. Faheem, and B.-S. Kim, "Energy and congestion-aware routing metric for smart grid AMI networks in smart city," *IEEE access,* vol. 5, pp. 13799-13810, 2017.

[119] M. Nassiri, M. Boujari, and S. V. Azhari, "Energy-aware and load-balanced parent selection in RPL routing for wireless sensor networks," *Int. J. Wirel. Mob. Comput.,* vol. 9, pp. 231-239, 2015.

[120] K. S. Bhandari, A. S. Hosen, and G. H. Cho, "CoAR: Congestion-aware routing protocol for low power and lossy networks for IoT applications," *Sensors,* vol. 18, p. 3838, 2018.

[121] J.-P. Sheu, C.-X. Hsu, and C. Ma, "A game theory based congestion control protocol for wireless personal area networks," in *2015 IEEE 39th annual computer software and applications conference*, 2015, pp. 659-664.

[122] H.-S. Kim, J. Paek, D. E. Culler, and S. Bahk, "Do not lose bandwidth: Adaptive transmission power and multihop topology control," in *2017 13th international conference on distributed computing in sensor systems (DCOSS)*, 2017, pp. 99-108.

[123] V. Michopoulos, L. Guan, G. Oikonomou, and I. Phillips, "DCCC6: Duty Cycle-aware congestion control for 6LoWPAN networks," in *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*, 2012, pp. 278-283.

[124] H. A. Al-Kashoash, Y. Al-Nidawi, and A. H. Kemp, "Congestion-aware RPL for 6L0WPAN networks," in *2016 Wireless Telecommunications Symposium (WTS)*, 2016, pp. 1-6.

[125] 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," *Journal of Sensors,* vol. 2016, 2016.

[126] P. Di Marco, G. Athanasiou, P.-V. Mekikis, and C. Fischione, "MAC-aware routing metrics for the internet of things," *Computer Communications,* vol. 74, pp. 77-86, 2016.

[127] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing without routes: The backpressure collection protocol," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010, pp. 279-290.

[128] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," in *29th IEEE Conference on Decision and Control*, 1990, pp. 2130-2132.

[129] F. Kaviani and M. Soltanaghaei, "CQARPL: Congestion and QoS-aware RPL for IoT applications under heavy traffic," *The Journal of Supercomputing,* vol. 78, pp. 16136-16166, 2022.

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

[131] N. T. Long, M.-P. Uwase, J. Tiberghien, and K. Steenhaut, "QoS-aware cross-layer mechanism for multiple instances RPL," in *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*, 2013, pp. 44-49.

[132] G. Rajalingham, Y. Gao, Q.-D. Ho, and T. Le-Ngoc, "Quality of service differentiation for smart grid neighbor area networks through multiple RPL instances," in *Proceedings of the 10th ACM symposium on QoS and security for wireless and mobile networks*, 2014, pp. 17-24.

[133] J. Nassar, N. Gouvy, and N. Mitton, "Towards multi-instances QoS efficient RPL for smart grids," in *Proceedings of the 14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks*, 2017, pp. 85-92.

[134] K. S. Bhandari, I.-H. Ra, and G. Cho, "Multi-topology based QoS-differentiation in RPL for internet of things applications," *IEEE Access,* vol. 8, pp. 96686-96705, 2020.

[135] W. Mardini, S. Aljawarneh, and A. Al-Abdi, "Using Multiple RPL Instances to Enhance the Performance of New 6G and Internet of Everything (6G/IoE)-Based Healthcare Monitoring Systems," *Mobile Networks and Applications,* pp. 1-17, 2020.

[136] M. M. Monowar and M. Basheri, "On Providing Differentiated Service Exploiting Multi-Instance RPL for Industrial Low-Power and Lossy Networks," *Wireless Communications and Mobile Computing,* vol. 2020, 2020.

[137] S. Junior, A. Riker, B. Silvestre, W. Moreira, A. Oliveira-Jr, and V. Borges, "DYNASTI—Dynamic Multiple RPL Instances for Multiple IoT Applications in Smart City," *Sensors,* vol. 20, p. 3130, 2020.

[138] D. Carels, N. Derdaele, E. D. Poorter, W. Vandenberghe, I. Moerman, and P. Demeester, "Support of multiple sinks via a virtual root for the RPL routing protocol," *EURASIP Journal on Wireless Communications and Networking,* vol. 2014, pp. 1-23, 2014.

[139] L. Buttyan and J.-P. Hubaux, "Nuglets: a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks," 2001.

[140] L. Buttyán and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Mobile Networks and Applications,* vol. 8, pp. 579-592, 2003.

[141] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, 2003, pp. 1987-1997.

[142] P. Das, K. Dubey, and T. De, "Credit based routing in delay tolerant networks," in *2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing*, 2012, pp. 158-163.

[143] R. Aslani, V. Hakami, and M. Dehghan, "A token-based incentive mechanism for video streaming applications in peer-to-peer networks," *Multimedia Tools and Applications,* vol. 77, pp. 14625-14653, 2018.

[144] M. M. Umar, S. Khan, R. Ahmad, and D. Singh, "Game theoretic reward based adaptive data communication in wireless sensor networks," *IEEE Access,* vol. 6, pp. 28073-28084, 2018.

[145] A. Parasuram, D. Culler, and R. Katz, "An analysis of the RPL routing standard for low power and lossy networks," *Electrical Engineering and Computer Sciences University of California at Berkeley,* 2016.

[146] J. Dubrulle, B. Quoitin, and A. Gallais, "Opportunities and limitations of multi-instance RPL," in *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2021, pp. 10-17.

[147] M. F. Triola and L. Iossi, *Essentials of statistics*: Pearson Addison Wesley Boston, MA, USA:, 2008.

[148] Difference between Skymote and Z1mote and Wismote in Contiki Cooja Simulator. Available: https://slogix.in/source-code/contiki-cooja-samples-for-IoT/difference-between-skymote-and-z1mote-and-wismote-in-contiki-cooja-simulation (accessed Dec. 2022)

[149] J. An, F. Le Gall, J. Kim, J. Yun, J. Hwang, M. Bauer*, et al.*, "Toward global IoT-enabled smart cities interworking using adaptive semantic adapter," *IEEE Internet of Things Journal,* vol. 6, pp. 5753-5765, 2019.

[150] K. G. Ngandu, K. Ouahada, and S. Rimer, "Smart meter data collection using public taxis," *Sensors,* vol. 18, p. 2304, 2018.

[151] G. C. Heck, R. Hexsel, V. B. Gomes, L. Iantorno, L. L. Junior, and T. Santana, "GRID-CITY: A Framework to Share Smart Grids Communication with Smart City Applications," in *2021 IEEE International Smart Cities Conference (ISC2)*, 2021, pp. 1-4.

[152] S. Latre, P. Leroux, T. Coenen, B. Braem, P. Ballon, and P. Demeester, "City of things: An integrated and multi-technology testbed for IoT smart city experiments," in *2016 IEEE international smart cities conference (ISC2)*, 2016, pp. 1-8.

[153] S. Taghizadeh, H. Elbiaze, and H. Bobarshad, "EM-RPL: Enhanced RPL for multigateway Internet-of-Things environments," *IEEE Internet of Things Journal,* vol. 8, pp. 8474-8487, 2020.