KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second Cycle 30 credits

Performance Evaluation of Different RPL Formation Strategies

ZIYI CHANG

Stockholm, Sweden 2023

Performance Evaluation of Different RPL Formation Strategies

ZIYI CHANG

Master's Programme, Communication Systems, 120 credits
Date: June 7, 2023

Supervisor: Voravit Tanyingyong
Examiner: Markus Hidell
School of Electrical Engineering and Computer Science
Swedish title: Prestationsutvärdering av olika RPL-bildningsstrategier

Abstract

The size of the IoT network is expanding due to advancements in the IoT field, leading to increased interest in the multi-sink mechanism. The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) is a representative IoT protocol that focuses on the Low-Power and Lossy Networks. However, research on comparing multi-sink strategies within the RPL network is limited. Therefore, this project aims to compare three common strategies: multiple-DODAG in one instance, virtual root, and multiple-instance. Using these strategies, we design and implement RPL networks and conduct simulations in various scenarios. Five different topologies are utilized in the experiments, considering different packet loss rates. Performance evaluation of each strategy is conducted using the Cooja simulator and Contiki-NG system, with a focus on the number of RPL control packets, Packet Delivery Ratio (PDR), and energy consumption. The results indicate that both the virtual root and multiple-DODAG strategies perform well with low packet loss, while the virtual root strategy outperforms the multiple-DODAG strategy with high packet loss. Additionally, the virtual root strategy incurs slightly higher energy costs than the multiple-DODAG strategy. Furthermore, the multiple-instance strategy demonstrates poor performance in most scenarios, except for the packet delivery ratio under high packet loss conditions. Besides the analysis, potential areas for future research on the RPL's multi-sink mechanism are finally identified.

Keywords

IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL), Multiple Sinks, Packet Delivery Ratio (PDR), Internet of Things

Sammanfattning

Storleken på IoT-nätverket expanderar på grund av framsteg inom IoT-området, vilket leder till ökat intresse för multi-sink-mekanismen. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) är ett representativt IoT-protokoll som fokuserar på Nät med låg effekt och förluster. Forskningen om jämförelse av multi-sink-strategier inom RPL-nätverket är dock begränsad. Därför syftar detta projekt till att jämföra tre vanliga strategier: multiple - DODAG i en instans, virtuell rot och multi-instans. Med hjälp av dessa strategier designar och implementerar vi RPL-nätverk och genomför simuleringar i olika scenarier. Fem olika topologier används i experimenten, med olika packet loss rate. Prestationsutvärdering av varje strategi utförs med hjälp av Cooja-simulatorn och Contiki-NG-systemet, med fokus på antalet RPL control packets, Packet Delivery Ratio (PDR) och energiförbrukning. Resultaten indikerar att både virtuell rot och multiple-DODAG strategier fungerar bra vid låg datapaketförlust, medan den virtuella rotstrategin överträffar multiple- DODAG strategin vid hög datapaketförlust. Dessutom medför den virtuella rotstrategin något högre energikostnader än flera DODAG-strategin. Dessutom visar multi-instans-strategin dålig prestanda i de flesta scenarier, förutom när det gäller datapaketleveransförhållandet under höga datapaketförlustförhållanden. Utöver analysen identifieras slutligen potentiella områden för framtida forskning om RPL-protokollets multi-sink-mekanism.

Nyckelord

IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL), Multi-Sink, Packet Delivery Ratio (PDR), Sakernas Internet

Acknowledgments

I would like to express my sincere gratitude and appreciation to my examiner, Markus Hidell, and my supervisor, Voravit Tanyingyong, for providing me with the opportunity to work on the Master's Degree Project when I was faced with the challenge of finding a project in the company. I am deeply thankful to Markus and Voravit for their immense help and guidance throughout this project.

During my study at KTH, I would like to extend my appreciation to all the teachers who have assisted me, including Markus Hidell, Voravit Tanyingyong, Marco Chiesa, Slimane Ben Slimane, Peter Sjödin, Dejan Manojlo Kostic, and Marina Petrova. They have all shown great responsibility in their teaching.

I would also like to sincerely thank my friends whom I met at KTH, including Tongxin Wang, Shengyao Shangguan, Gengwu Du, Huanyu Wang, Ruiqi Zhang, Jinfeng Xiong, Tingrui Zhang, Xingyu Zhao, Yue Wang, Li Zha, Rongfei Pan, and others who have supported me or worked alongside me. I am grateful for all the wonderful times we shared in Stockholm.

Lastly, I would like to express my gratitude to Sweden and Stockholm. These two years have been incredibly interesting and meaningful for me.

I am deeply grateful to all of you for your support and assistance during my Master's study period.

Thank you all.

Stockholm, June 2023
Ziyi Chang

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
CoAP	The Constrained Application Protocol
CSMA	Carrier-Sense Multiple Access
DAG	Directed Acyclic Graph
DAO	Destination Advertisement Object
DAO-ACK	Destination Advertisement Object Acknowledgement
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination-Oriented Directed Acyclic Graph
IoT	Internet of Things
LLN	Low-Power and Lossy Networks
MOP	Mode of Operation
MRHOF	Minimum Rank with Hysteresis Objective Function
OF	Objective Function
OF0	Objective Function Zero
PDR	Packet Delivery Ratio
RFID	Radio-Frequency Identification
RPL	IPv6 Routing Protocol for Low-Power and Lossy Networks
UDGM	Unit Disk Graph Model
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Background

Since the Internet was invented, routing protocols are always crucial as the essential component because networks use the protocols to communicate with each other [1]. As the routing protocol is essential, there are many protocols and much research related to the protocol. For example, comparing different TCP versions is a popular research area [2].

Nowadays, besides the traditional network, Internet of Things (IoT) devices and sensor networks are widely used in multiple fields, including healthcare, industrial field [3], transportation, automotive industries [4] and smart home [5]. With the development of IoT, there are many IoT protocols which become famous, including Bluetooth Low Energy, ZigBee, Thread, and RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks). Different protocols are usually applied in multiple areas. For example, RPL is suitable for low-power and lossy networks as the name indicates [6]. With the increasing application of IoT protocols, there are many kinds of research covering multiple aspects, including the architectures [7], security [8], and congestion problems [9]. As for RPL, there are still many types of research related to several aspects, including the challenge of RPL [10], the survey concentrating on security, mobility, and topology [5], and the performance evaluation for specific RPL network strategy [11]. However, there is little research related to the different network strategies of RPL. In this report, there is a comparison among three different strategies of RPL. All three approaches also include multiple root nodes (sinks).

In RPL, network topology forms the Destination-Oriented Directed Acyclic Graph (DODAG), including a single root node that generally serves

as a gateway of the Low-Power and Lossy Networks (LLN). An LLN may run multiple RPL Instances to serve different constraints of performance criteria. Each RPL Instance may contain multiple DODAGs, and each DODAG includes one root separately [12]. Besides the independent root nodes, multiple root nodes can also be coordinated to synchronize the DODAG state to make them act as a single DODAG root like a virtual root [13]. In addition, a network can include multiple RPL Instances with only one DODAG in each RPL Instance. However, several common nodes belong to various DODAGs and RPL Instances [14]. Overall, the above methods are three popular RPL strategies.

Many experts have worked on the protocol and achieved many results. Some researches focus on the function of a RPL network, such as security, optimization, or limitation [5]. Some experts focus on the performance of a certain RPL strategy [11]. However, it is still being determined what the advantages and disadvantages are of using these strategies and in which situations one strategy would be preferred over the others. Consequently, we need a comparative study to gain more insights into different RPL network formation strategies. The performance of the three strategies in different scenarios will be compared in this paper. There is also a conclusion that a particular method is more suitable for a specific scenario.

1.2 Problem

Although many researchers have studied the RPL, only a few studies focus on the performance comparison of different RPL network topologies. Therefore, there is a need to evaluate the network performance of different RPL network topologies and identify suitable scenarios for each network topology.

The main research question is, “What are the advantages and disadvantages of using different RPL network formation strategies when deploying multiple sinks?”. In this context, three strategies need to be analyzed:

1. A single RPL Instance with multiple DODAGs using one sink per DODAG, called multiple-DODAG strategy.

2. A single RPL Instance with one DODAG using a virtual root node with multiple sinks, called virtual root strategy.

3. Multiple RPL Instances strategy.

1.3 Purpose

The main purpose is to identify the advantages and disadvantages of the three RPL network formation strategies, which helps determine which strategy is suitable for a certain scenario. The main benefit is the sustainable development of a RPL network, as this study's results can help reduce the cost of deployment and operation.

1.4 Goals

This project aims to evaluate the performance of different RPL network formation strategies. A series of experiments need to be designed and implemented to figure out the advantages and disadvantages of different strategies.

The goal can be divided into the following three sub-goals:

1. Design and develop RPL network topologies based on the three RPL network formation strategies. In this project, Contiki-NG is used as the experiment environment, which is an open-source operating system for IoT devices. In addition, Cooja is used as the simulation tool, which is integrated into Contiki-NG [15].

2. Identify relevant parameters and proper methods to collect data, such as energy consumption and communication overhead (in terms of the number of generated messages) at each device.

3. Evaluate and analyze the data from experiments to determine the advantages and disadvantages of the three RPL network formation strategies.

1.5 Research Methodology

The quantitative research method is applied in this project, which usually focuses on analyzing data collected through experiments [16]. The most crucial characteristic of quantitative research is collecting and generalizing data to explain a specific phenomenon or evaluate performance. This project aims to compare the performance in different scenarios by analyzing data collected from the experiment. This project includes several groups of comparison experiments with different scenarios. The experiment scenario

is identical in each group, including an identical physical topology, the same duration of each experiment, and the same kind of data. According to the experiment's data, the result is to detect the different performances of different RPL topologies in the same scenario. Furthermore, the conclusion also identifies which strategy is better suited for specific scenarios.

1.6 Delimitations

The first delimination is that the project focuses on comparing different RPL network formation strategies rather than a specific RPL strategy parameter. For example, RPL networks can apply several configurations, including different Objective Functions (OFs) and Mode of Operations (MOPs). The OF is applied to calculate the distance between nodes and sinks. Additionally, the MOP includes two information-storing modes of the RPL. However, all experiments determine the same OF and MOP configuration rather than comparing the performance of different deployments. Therefore, the comparison between different RPL parameters is not the main point of this project. Furthermore, the second delimitation is the lack of practical experiments since the project is based on the simulation experiment rather than the actual devices. However, the project is also valuable since the result can also indicate how different strategies affect the RPL performance.

1.7 Structure of the thesis

Chapter 1 presents a brief overview of the project, including its background and problem statement.

Chapter 2 extends the brief introduction by introducing the development of Internet of Things (IoT) and explaining the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) protocol. This chapter also presents the experiment environment and related work, including the limitation of the RPL, the virtual root strategy, and the multiple-instance strategy.

Chapter 3 presents the methodology of this project, including the introduction of the multi-sink mechanism and a detailed introduction of the multiple-DODAG, virtual root, and multiple-instance strategies.

Chapter 4 dedicates the five network topologies and three experiment metrics, which only includes the theoretical introduction in this chapter.

Chapter 5 presents the detailed implementation of the experiment, including five topologies, the node implementation, and an introduction to the

process of collecting data of the three evaluation metrics.

Chapter 6 presents the result of each experiment of each topology, along with data analysis. Additionally, this chapter includes a discussion of all the results.

Chapter 7 provides a conclusion of the project, including limitations and potential future work.

Chapter 2

Background

2.1 Internet of Things

2.1.1 IoT Introduction

Allegedly the Internet of Things (IoT) term was proposed as early as the 1980s [17]. However, IoT technology has become popular since the broad application of Radio-Frequency Identification (RFID) technology in the late 1990s [18]. With the rapid development of IoT technology, IoT devices have become popular in our modern life [19]. According to the forecast institution, the number of IoT-connected devices will reach 29.4 billion in 2030 [20]. In addition, the potential economic value of IoT is significant. According to McKinsey's research, IoT's estimated value may reach 12.6 trillion dollars in 2030 at most [21].

The IoT technology is widely used in both industrial areas and household areas [22]. For example, traffic flow monitoring system is widely deployed in some cities. Suppose there is heavy traffic in one direction at a crossroad. In that case, the duration of the green light will be extended to allow more cars to pass in that direction and prevent traffic congestion at the intersection. In contrast, the duration of the traffic light can return a small value without the heavy traffic, which allows cars and people to wait for a short time in both directions. The self-adaptive traffic light is a simple but typical application of IoT technology [23]. Besides the smart city, the smart home also applies IoT technology. Nowadays, more people use smartphones to control things in their rooms. For example, people can now use their smartphone to turn on or turn off the light, change the color of the light, reserve the start of a rice cooker, and even turn on the air conditioner before arriving home [24].

2.1.2 IoT Communication Models

The IoT usually includes four basic models, device-to-device communication, device-to-cloud communication, device-to-gateway model, and back-end data-sharing model [19] [25].

2.1.2.1 Device-to-Device Communications Model

The device-to-device model is the simplest, representing two or more devices connected directly rather than through an intermediary dedicated forwarding server. The devices communicate with each other over many networks or protocols like Bluetooth, ZigBee, or RPL, as shown in figure 2.1.

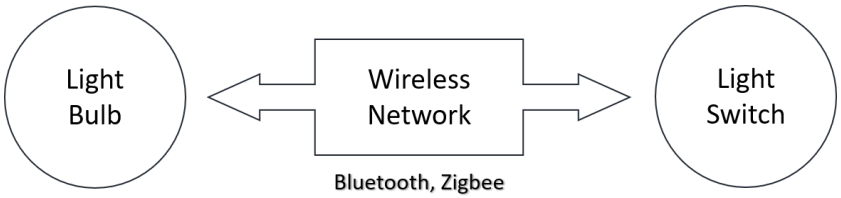

Figure 2.1: Device-to-Device Communications Model [26].

This model is widely used in small application scenarios, like the smart home. In this situation, devices usually send a small number of packets and require a low data rate relatively. However, all devices need to choose the same protocol to construct the IoT network in the device-to-device model. Therefore, this model is simple but lacks flexibility. In addition, the device-to-device model is unsuitable to be employed in a large area scenario because of the devices' poor communication quality at a long distance.

2.1.2.2 Device-to-Cloud Communications Model

In this model, IoT devices connect directly to an Internet cloud service, such as an application service provider, as figure 2.2 shows. Therefore, the communication is between devices and the cloud server in reality. This model usually uses existing communication methods, including Ethernet or Wi-Fi, to deploy a connection between the device and the Internet or IP network, which eventually attaches to the cloud service.

This strategy is usually employed in data collection scenarios, such as the camera around the crossing road and environment monitoring devices. In addition, some smart home devices can also use the strategy. However, this model has interoperability limitations. For example, the device and cloud

Figure 2.2: Device-to-Cloud Communications Model [26].

service are usually from the same provider if exclusive data protocols are applied between the device and the cloud service. Therefore, the provider may require the consumer to use the specific IoT devices and cloud service as a set rather than choose the device and cloud application separately.

2.1.2.3 Device-to-Gateway Communications Model

The difference between the device-to-gateway and the device-cloud method is that there is an application layer gateway between devices and the cloud service, as figure 2.3 shows. In brief, application software runs on a local gateway device, regarded as an intermediary. In this model, the intermediary application gateway can enhance security and provide other functions like protocol translation.

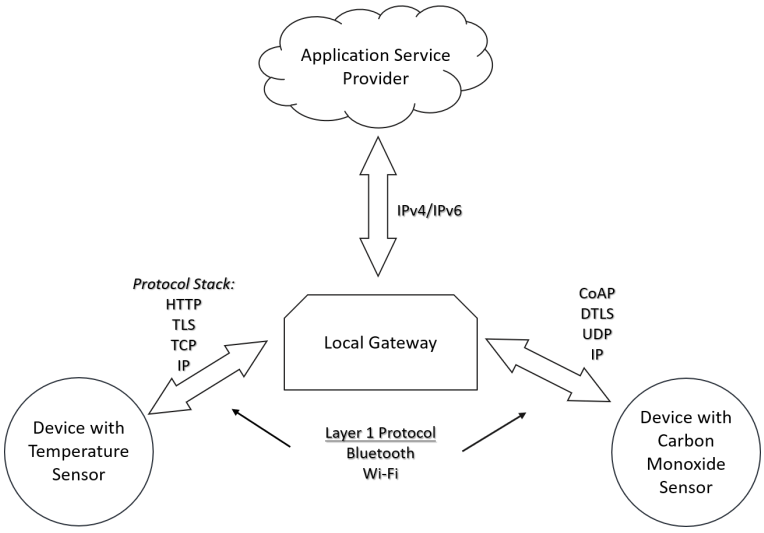

Figure 2.3: Device-to-Gateway Communications Model [26].

As for the application, some forms of this model are usually used in

consumer devices. For example, the local gateway is frequently applied on a smartphone as an app to communicate with a device and relay data to a cloud service.

2.1.2.4 Back-End Data-Sharing Model

The back-end data-sharing model can be considered an improvement of the single device-to-cloud communication model. This strategy supports the user to access the data from the cloud service, which is collected from other sources. Therefore, a back-end sharing strategy enables the data collected from unitary IoT device data sources to be integrated and analyzed.

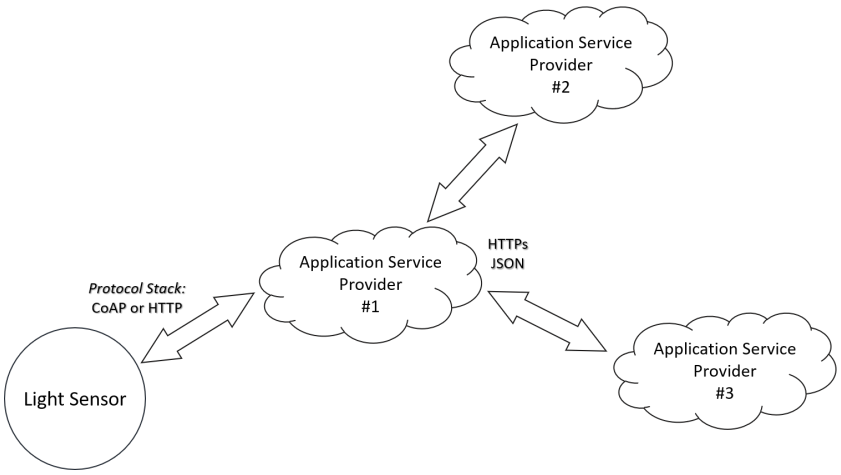

Figure 2.4: Back-End Data-Sharing Model [26].

For example, in a traffic flow monitoring system, the administrator can consolidate and analyze the traffic data produced by all sensors deployed on different crossing roads. In the simple device-to-cloud model, the data of each IoT sensor is stored in a stand-alone data repository. However, a back-end data-sharing strategy allows the administrator to easily access and analyze the data in the cloud server produced by all devices in the system. In addition, the data strategy allows the administrator to move the data quickly when they switch between IoT services.

2.2 RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks

RPL is a distance-vector routing protocol for a wide range of practical LLN applications, usually considered a device-to-device model. There are two main characteristics of LLNs. Firstly, it consists of many constrained nodes, typically working under constrained situations, including limited processing power, small memory, and even constrained energy conditions. Besides the constrained nodes, these nodes are interconnected by lossy links and support low data delivery rates. Secondly, LLN has different traffic patterns and needs to support multiple communication types, including point-to-point, multipoint-to-point, and point-to-multipoint.

Consequently, RPL was designed to satisfy the LLNs' characteristics. Firstly, RPL splits the packet processing and forwarding tasks from the routing optimization objectives to satisfy LLNs' requirements in large application domains. The routing optimization objectives include minimizing latency, meeting the requirements of constraints, or reducing energy resumption. Secondly, since the LLN requires multiple traffic patterns without typically predefined topologies, the routing protocol needs a comprehensive topology to support various communication types. Eventually, RPL organizes the topology as a Directed Acyclic Graph (DAG), which is a tree-like structure, including all edges oriented in one way towards one or several roots without loops in the topology. One step further, RPL splits the whole DAG network into several Destination-Oriented Directed Acyclic Graphs (DODAGs), which means one group includes only one root node. Therefore, RPL builds the network with one or multiple DODAGs [12]. In addition, one or more DODAGs can form a RPL Instance. DODAGs in one RPL Instance share the same Instance ID. The DODAG, RPL Instances, and RPL information packets will be introduced in detail in the next several sections.

2.2.1 Destination Oriented Directed Acyclic Graph (DODAG)

Regarding LLNs' requirements, RPL forms its topology as a Destination-Oriented Directed Acyclic Graph (DODAG), a unique situation of the Directed Acyclic Graph (DAG). A DAG contains one or more sinks without cycles, where all nodes can follow one or multiple paths toward one or multiple root nodes. As figure 2.5 shows, there are two root nodes R_1 and R_2, and several

sensor nodes that have more than one path towards root nodes without a cycle [12].

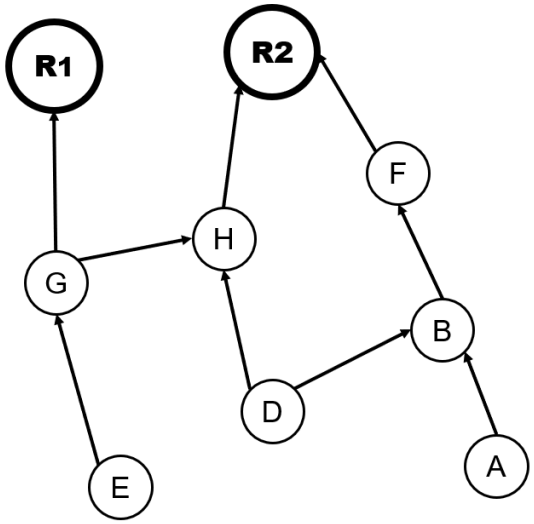

Figure 2.5: Directed Acyclic Graph (DAG), no cycles and two root nodes R_1 and R_2.

Unlike the DAG, a DODAG contains several sensor nodes but only one root node, as figure 2.6 shows. In a DODAG, there are still multiple paths toward the root node without cycles, but only one root node exists. In the RPL strategy, the DODAG root node as the sink is applied as a border router and gateway for the LLN. In addition, the sink can aggregate all routes in the DODAG. For example, the sink can collect the information from the RPL network and deliver the data to the backbone network for further processing [27].

Besides the RPL Instance ID, each DODAG has a DAG ID. The DAG ID is unique in each DODAG, which indicates the destination root node that the sensor nodes deliver packets.

2.2.2 RPL Instance

Besides DODAG, the RPL Instance is also an important component of the RPL network [12]. In general, a RPL Instance is a group that includes one or more DODAGs that share the same RPLInstanceID. The RPLInstanceID is a specific identifier in RPL networks. A RPL sensor node can belong to multiple RPL Instances but can only belong to one DODAG per RPL Instance at the same time [28]. In this project, three different RPL strategies are related to

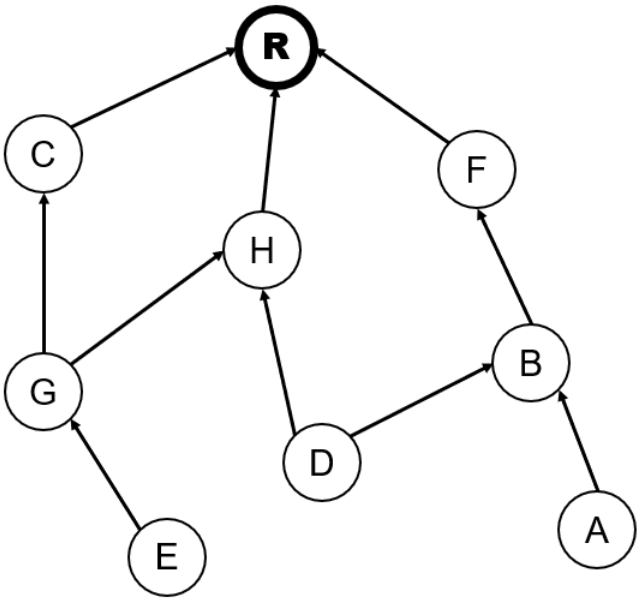

Figure 2.6: Destination-Oriented Directed Acyclic Graph (DODAG),with only one root node R.

different RPL Instance types.

In the protocol, the downward route means the route from the DODAG root to the sensor node [12]. Correspondingly, the route from the sensor node to the DODAG root is called the upward route. The RPL network has two modes: storing mode and non-storing mode. These two methods are also important factors when building the RPL network, which is applied for maintaining downward routes. In storing mode, nodes store the downward routing information for their sub-DODAG. Therefore, each node determines the next hop by checking its routing table. However, nodes do not store the downward routing information in non-storing mode. Instead, the DODAG root determines the routing path.

2.2.3 Rank and Objective Function

Rank is significant in a RPL network, which indicates the node's relative position to the DODAG root and other nodes within a DODAG. The rank value strictly increases if the node is far away from the root and decreases if the node is near the root. In the configuration of Contiki-NG, if the rank equals 65535, the route will be considered unreachable. The rank can be calculated according to the Objective Function (OF), which also defines the method of selecting

the parent node in a DODAG. There are two objective functions Objective Function Zero (OF0) and Minimum Rank with Hysteresis Objective Function (MRHOF) [12].

The goal of OF0 is to allow a node to join a DODAG Version, which offers feasible connectivity to a specific destination, for example, a specific set of nodes [29]. Practically, OF0 is designed to find the nearest node as the preferred parent without load balancing. The rank of a node R_N is calculated by adding a strictly positive scalar value $rank\_increase$ to the rank of the preferred parent node R_P, as the first equation shows. The $rank\_increase$ is calculated by $rank\_factor$ (R_f), $step\_of\_rank$ (S_p), $stretch\_of\_rank$ (S_r) and $MinHopRankIncrease$ as the second equation shows. Where the $step\_of\_rank$ is applied to calculate the amount to increase the rank along a specific link. $MinHopRankIncrease$ is the unit where the variable $rank\_increase$ is applied. The $rank\_factor$ and $stretch\_of\_rank$ are two configurable parameters. The R_f is used to multiply the effect of the link properties. The S_r indicates the maximum augmentation to the S_p of a preferred parent to allow the selection of a feasible successor [30].

$$R_N = R_P + rank\_increase \tag{2.1}$$

$$rank\_increase = (R_f * S_p + S_r) * MinHopRankIncrease \tag{2.2}$$

Another objective function is MRHOF, which is designed to prevent excessive churn in the network topology [31]. There are two phases to achieve the goal of this objective function. Firstly, it finds the path with minimum rank. Then if the minimum rank path is shorter than the current path by at least one given threshold, it changes to the minimum rank path. The second method is called ”hysteresis” in this objective function. By applying MRHOF, a sensor node computes the path cost of a neighbor by adding two components, including the value of the neighbor node or link's metric and the value of the defined metric recorded in the Metric Container [30]. MRHOF is also required to apply several metrics, including hop count, latency, and ETX.

2.2.4 RPL Control Messages

There are four types of RPL control messages, including DODAG Information Solicitation (DIS), DODAG Information Object (DIO), Destination Advertisement Object (DAO) and Destination Advertisement Object Acknowledgement (DAO-ACK) [12].

DODAG Information Solicitation (DIS)

The DIS message is sent by a RPL normal node, which is used to solicit a DIO message from another RPL node nearby. In the RPL network, the DIS message can be used to probe the nearby DODAGs by a node. The message format is shown in figure 2.7. There are three parts of the DIS Base Object Format. The Flags and Reserved fields are both 8-bit and unused. The protocol also requires these two fields to be configured as zero by the sender and ignored when a node receives it.

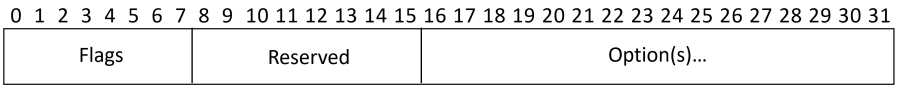

Figure 2.7: The DIS Base Object.

DODAG Information Object (DIO)

A node can discover a RPL Instance through the DIO message. Besides discovering a new RPL Instance, the DIO message can also be used to collect the configuration parameters, choose a DODAG parent set, and preserve the DODAG. The DIO Base Object Format is shown in figure 2.8.

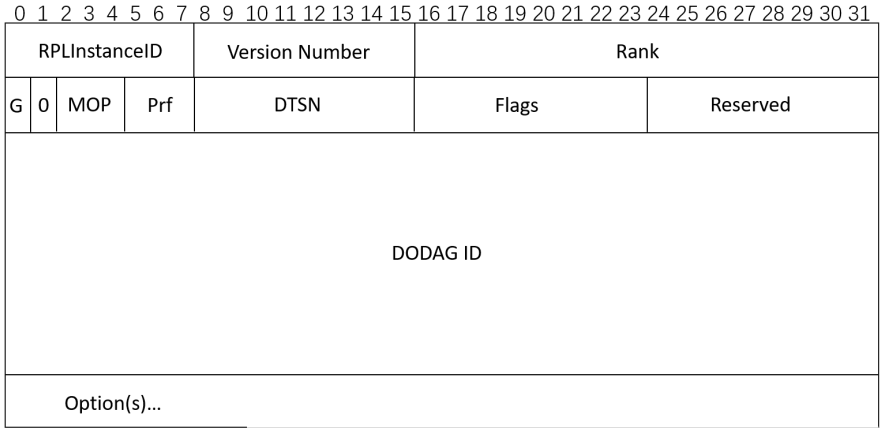

Figure 2.8: The DIO Base Object.

In the message format, RPLInstanceID is set by the DODAG root that shows which RPL Instance the node belongs to. The Version Number is an 8-bit unsigned integer which is determined by the DODAG root to indicate the DODAGVersionNumber. The DODAGVersionNumber is a component of

the DODAG Version tuple. Besides DODAGVersionNumber, RPLInstanceID and DODAGID also belong to the DODAG Version tuple. In addition, Rank and Mode of Operation (MOP) are also important parameters of RPL, which are included as two fields in the DIO messages. Rank indicates the position relative to other nodes within a DODAG. The MOP field represents the operation mode of the RPL Instance, which is related to the DODAG root. All nodes need the MOP to determine the storing mode and if they support the multicast function. The two most important modes are storing mode and non-storing mode, as indicated in section 2.2.2. Prf (DODAGPreference) is also an important field, which shows the node prefers one DODAG root to other DODAG roots within the instance.

Destination Advertisement Object (DAO)

The DAO message is applied to transmit information upward along the DODAG, which means the direction towards the DODAG root. The DAO messages are slightly different with different MOP configuration. In Storing mode, the DAO message is sent to the selected parent node from the child node. However, in Non-Storing mode, the DAO message is sent to the DODAG root directly. The destination may acknowledge the DAO message with a Destination Advertisement Object Acknowledgement (DAO-ACK) message back to the sender from the destination node.

Figure 2.9: The DAO Base Object.

Figure 2.9 is the DAO Base Object format, whose fields are similar to DIO and DIS message except for DAOSequence. The DAOSequence will increment at each specific DAO message from a node and the response DAO-ACK message.

Destination Advertisement Object Acknowledgement (DAO-ACK)

The DAO-ACK message is a response from a DAO parent or the DODAG root to the node that sent the DAO message. Figure 2.10 shows the DAO-ACK Base Object format. Like the DAO message, the DAOSequence number is incremented at the DAO message and the echoed DAO-ACK message. The status field indicates the completion. For example, the status code set to 0 means unqualified acceptance, which indicates the node receiving the DAO-ACK is not rejected. In addition, the DAO-ACK is an optional message defined by RPL standards [12]. Therefore, the DAO-ACK is not enabled in the Contiki-NG system by default.

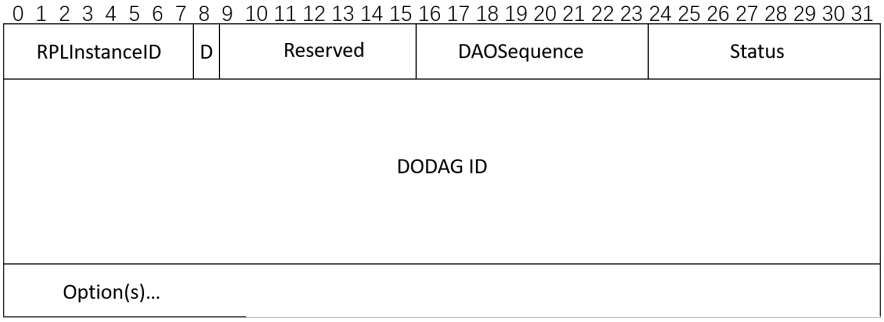

Figure 2.10: The DAO-ACK Base Object.

2.3 Contiki-NG and Cooja

This section aims at introducing Contiki-NG and Cooja, which are important in the research of the RPL. Contiki-NG is an operating system for next-generation IoT devices. Meanwhile, Cooja is a simulation platform that is integrated with Contiki-NG.

2.3.1 Contiki-NG

Contiki-NG is a cross-platform, open-source operating system for networked, memory-constrained systems and low-power wireless Internet of Things devices. Contiki-NG means Contiki Next Generation. Therefore, there is an old version called Contiki OS. Compared to Contiki OS, Contiki-NG is a more popular system, which focuses on dependable low-power communication and standard protocols, such as 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)), CoAP (The Constrained Application

Protocol (CoAP)) and RPL. Contiki-NG uses the uIP stack to support TCP/IP network stacks. The uIP stack is a lightweight open-source implementation of TCP/IP stacks for small 8- and 16-bit microcontrollers, which Adam Dunkels developed [32]. However, the uIP implementation only has a minimal set of features supporting a full TCP/IP stack since the uIP stack was designed to minimize the memory footprint and processing power required for TCP/IP communication. Therefore, it can only handle a single network interface and contains the IP, ICMP, UDP, and TCP protocols. In addition, the Contiki-NG can also support standard-based IPv6 communication on some modern IoT platforms, including ARM Cortex M3 and other 32-bit MCUs [15].

Contiki-NG provides multiple functions, including memory usage calculation, energy monitoring, and detailed log files. Therefore, users can collect detailed information following the related implementation of these areas. In addition, as Contiki-NG is an open-source system, some sections of codes can be modified to support multiple functions and improve the default network in simulation. Therefore, various types of research in RPL are based on the Contiki-NG [33] [13]. Besides the simulation, the improved network can also be implemented on real devices since the system construction and implementation are the same on both real devices and simulation platforms.

2.3.2 Cooja

Cooja is a simulation tool that is integrated with Contiki-NG. The RPL simulation is based on Cooja with several available motes. The motes can be generated by compiling the node file written in C language. In this project, Z1 motes are compiled and implemented as the sensor and root nodes. Different nodes' positions can be adjusted following the simulation requirement in Cooja through a simulation file. For example, the nodes can be set as a grid pattern or in random positions. Cooja provides several functions to collect the data under different conditions, including different Tx/Rx (transmission successful/reception successful) ratios and the transmission range of each node. Cooja can also collect the log information and nodes' output. In this project, different network models are applied to verify the different performance of different strategies in different scenarios. The log file and radio message are also collected to analyze each strategy's performance.

In Cooja, the transmission range represents the simulation's radio range. The Contiki-NG uses Unit Disk Graph Model (UDGM) as the transmission model. All nodes have unit graphs with the same radius as their transmission areas. Therefore, the node can communicate with the node located at the circle

center if it is located in the circle. Additionally, Contiki-NG has two loss methods, including constant loss and distance loss. In the constant loss model, the node has a fixed transmission and reception successful ratio value as long as it is located within the transmission range. However, in the distance loss model, only the node located at the limited distance has the same successful ratio as the configuration. While the node is in the circle, the successful ratio increases as the distance between the node and the circle center decreases. In this project, the experiments use the constant loss method.

2.4 Related Work

As RPL is an essential protocol in the IoT field, many types of research are related to RPL in many areas, including limitations, challenges, and formation strategies. Therefore, this section introduces previous work in the aspects mentioned above. Since the protocol formation strategy is the main idea of this project, some previous research related to RPL formation strategies is the main explanation in this section.

Ghaleb et al. had a detailed review of RPL's limitations [30]. They first concluded the comprehensive overview of limitations related to RPL essential operation, which can be divided into three types: OF limitations, RPL downward route limitations, and routing maintenance limitations. For example, one risk of RPL is involved in objective functions [34]. Since there is a lack of load-balancing configuration, the sensor node consistently sends messages through a specific link after determining the preferred connection, even suffering network overload. The drawback may cause high energy consumption related to network disconnection and unreliability issues. Besides the limitation of objective functions, another drawback is the storage limitation when applying the storing mode. While applying storing mode, every node needs to store the routing information of nodes whose routing links to the sink are passing through itself [12]. However, since the RPL focuses on constrained memory devices but still supports large-scale networks, the constrained devices may be overloaded because of the large amount of routing maintenance information. After explaining several limitations, Ghaleb et al. also update some research that is aimed at dealing with the drawbacks [30]. Kiraly et al. provide a method to deal with the storage limitation by not delivering the information that a node cannot store because it cannot be delivered to the destination. However, a drawback of this mechanism is that some paths are built but have yet to be available because of the method [35]. Then, Kiraly et al. provide another method called D-RPL, which

improves the RPL storing mode by applying multicast. The node and its children are registered in a multicast group if unreachable. Then the DODAG root applies the multicast address and the multicast group to communicate with the unreachable special nodes through standard mechanisms. Besides some effective solutions, some emphasized limitations still need to be fully addressed. For example, load balancing is still a problem. Although much research focuses on load-balancing with some outcomes, it is evident that the instability of frequently changing preferred parents may undermine the advantages of load-balancing [36].

In addition to the research on limitations, plenty of studies are related to the RPL formation strategy. Carels et al. introduce the virtual root mechanism and verify that it can be applied in RPL with minimal complexity [13]. As the network scale increases, more sinks are implemented to reduce traffic pressure and provide backup. However, the default Contiki-NG implementation requires multiple sinks to work with multiple DODAGs, which leads to higher memory usage for sensor nodes within the DODAGs, as they must store various sets of RPL information for each sink [37]. Besides the sensor nodes, each root must maintain a big DODAG because of multiple independent DODAGs. In contrast, sensor nodes in the virtual root strategy only need to store a set of RPL information, although there are multiple sinks actually. The multiple roots can maintain one DODAG together. Since the memory usage to store the routes within the DODAG will be allocated to multiple possible sinks. They also provide a detailed discussion of implementing the virtual root strategy, including the synchronization of roots and the route of sensor nodes within DODAGs.

In addition to the virtual root strategy, several studies have related to the multiple-instance strategy. Mai et al. [11] and Jeremy et al. [14] have significantly contributed to this approach. Mai et al. explained the concept of the multiple-instance strategy and discussed its advantages and disadvantages. The author also compared the multiple-instance strategy and single-instance single-root strategy. Then they verified that the multiple-instance strategy performs better than the single-instance strategy regarding convergence time and packet delivery ratio. However, since Contiki-OS does not fully support the multiple-RPL-instance strategy, the authors implemented a system including two independent DAGs with two objective functions to serve as a multiple-instance strategy. Then, based on Mai et al.'s work [11], Jeremy et al. conducted a detailed analysis of the multiple-instance strategy in Contiki-OS. The authors explained the storage structure of instances and routing information, and conducted experiments using Cooja simulations and

testbed experiments [38]. They provided an approach to satisfy the multiple-instance requirement in the Contiki-OS system, which involved storing multiple instances and multiple sets of routing information and improving the metric container. Finally, the authors implemented large-scale experiments with several evaluation metrics, including packet delivery ratio, latency, RPL control packets, and hop count fields. The experiment indicates the result with different multiple-instance networks, which includes the different number of concurrent instances. As the result, the experiment compared the network performance with various number of concurrent instances.

Chapter 3

Methodology

3.1 Introduction

This project aims to evaluate the performance of three different RPL strategies in various scenarios. Each strategy applies multiple sinks to provide redundancy and distribute the traffic load. This section introduces the multi-sink mechanism and three RPL strategies.

3.1.1 Multiple Sinks

Before the widely used IoT technology, the multipoint-to-point pattern was widely used in Wireless Sensor Network (WSN). Therefore, previous protocols are usually designed to guarantee the packet transmission towards a dependent sink [39]. However, many sensor nodes are deployed in a large area in a sensor network because of the modern requirement of IoT technology. Since the distance between sensor nodes and the root node increases, one root node cannot deal with all received packets from many sensor nodes which are far away. For example, some sensors are deployed among several intersections to collect traffic data. In this case, a sink cannot receive the packet from a long distance and process the data in time because of the propagation delay. Therefore, deploying multiple sinks is necessary to shorten the packet transmission distance [40]. Besides the distance problem, another multi-sink application is to deploy a backup sink in a sensor network. Since the sensor network is usually a real-time system, if a sink goes down, the system needs to be suspended and await maintenance. Therefore, system redundancy is suitable for a network that needs to operate continuously. Consequently, multi-sink can satisfy the requirement of redundancy and backup in the sensor

network [41].

More specifically, multi-sink is widely deployed in the application of RPL networks because one RPL Instance can contain one or more DODAGs according to the protocol. If there are multiple DODAGs in a RPL network, the network contains multiple sinks. The goal of deploying multiple root nodes in RPL networks is similar to other sensor networks. Firstly, it can shorten the distance between sensor nodes and the DODAG root and reduce the traffic load of a single DODAG root. Secondly, it can deploy a backup in the RPL network. For example, if one DODAG root fails in a system where one RPL Instance contains two DODAGs, another DODAG still works. Therefore, this RPL Instance can still work and wait for the failed link to be repaired and back online. In this project, all three strategies include multiple sinks to satisfy the redundancy requirement and reduce the heavy traffic load of root nodes.

3.2 Multiple DODAGs in one RPL Instance

As figure 3.1 shows, The first strategy is the Multiple DODAGs in one RPL Instance strategy. In this strategy, RPL Instance containing two disjointed DODAGs with common nodes, which is the simplest multi-sink strategy of RPL networks [12]. The node C in the figure is a common node of the two DODAGs. The RPL Instance ID is identical to the two DODAGs, but the two DAG IDs are different. The objective function is MRHOF. As the two DODAGs are independent, each node must choose one DODAG root if it is a common node of the two DODAGs. Therefore, if the connection between one sensor node and the root fails, the sensor node can repair the route quickly by communicating with the back-up DODAG root. In addition, deploying the RPL network in practical applications is convenient since the strategy is simple.

However, the recovery time is a limitation for the RPL network when a DODAG root fails. Although each sensor node has a backup route, it costs much time to build communication between the node and the backup DODAG root. Since multiple DODAG roots have different versions in the RPL network, the RPL control messages include different information from separate DODAG roots. Therefore, if one root cannot work, various sensor nodes of the RPL network receive the RPL information from the other RPL Instance, which takes some time. Besides the time consumption, data packets cannot be delivered to the destination during route rebuilding processing. [42].

In this project, the multiple-DODAG strategy in one RPL Instance is Contiki-NG and Cooja's default configuration.

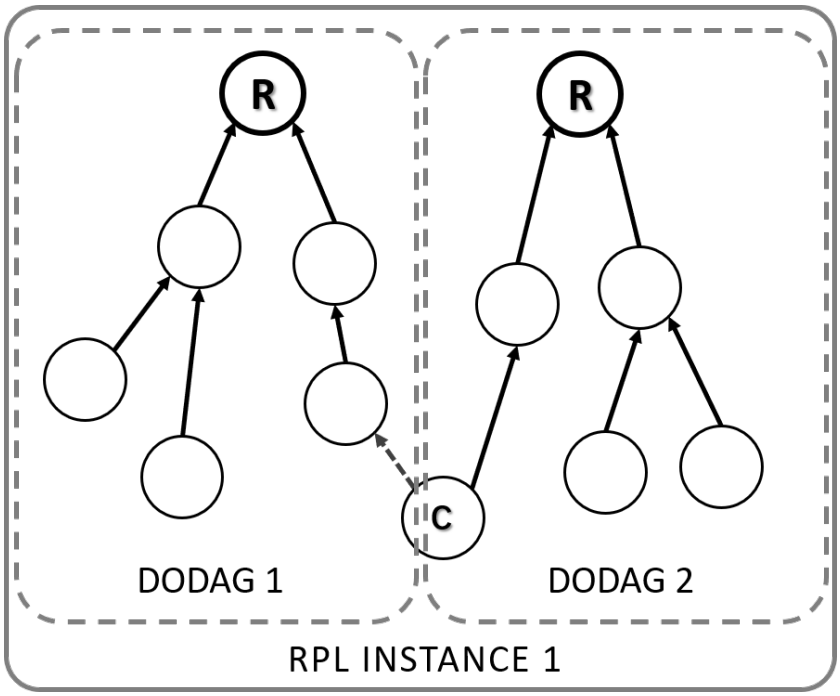

Figure 3.1: Multiple DODAGs in one RPL Instance.

3.3 Virtual Root in one RPL Instance

Figure 3.2 shows another strategy called virtual root [12]. There are still two DODAGs within one RPL Instance in this strategy. However, the difference between the virtual root strategy and multiple-DODAG strategy is the relationship between DODAGs. The two DODAGs are independent in the previous strategy despite sharing the same RPL Instance ID. However, according to virtual root configuration, sensor nodes consider only one DODAG root of the whole RPL network, although there are two sinks in reality. The approach is to synchronize the two DODAG roots by setting the same DODAG Version, including the same RPL Instance ID, DAG ID, and Version Number. According to the synchronization of multiple DODAG roots, the sensor nodes consider the multiple sinks as one DODAG root in the RPL network. Therefore, the special DODAG root is called a virtual root since the root does not exist in reality. Compared to multiple DODAGs in one RPL Instance strategy, there are two main advantages when employing the virtual root strategy, including memory and process requirements [27].

Firstly, in both storing and non-storing modes, the sink must store the routes to each node in the network, including nodes of the other DODAGs.

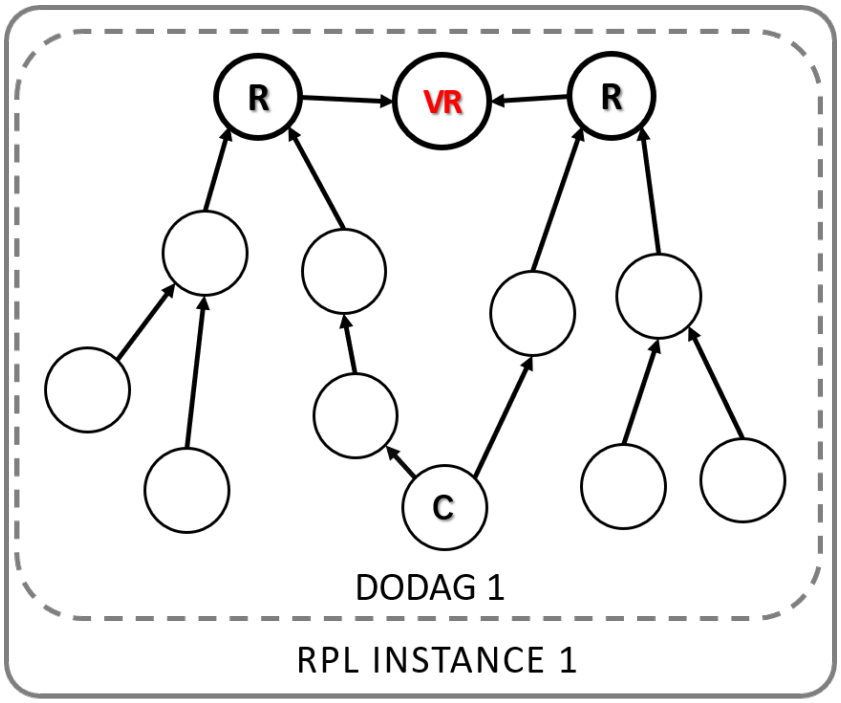

Figure 3.2: Virtual Root in one RPL Instance.

Therefore, there is a large amount of routing data that one DODAG root stores, which results in high memory consumption. The memory usage per sink is similar to the memory consumption when deploying only one sink in the RPL network. If considering downward routes from the DODAG root, each sink of the multiple-DODAG strategy may cost more memory than one sink in the RPL network. However, the physical root node does not store the route information of all nodes in the RPL network when applying the virtual root. Therefore, memory consumption is spread over different roots. One root only can store the information of its own DODAG, like one sensor node stores the information of its sub-DODAG in the single root strategy. In addition, the position of reality sinks will influence the number of children nodes per sink. [13].

Secondly, this strategy simplifies the process of nodes. A common node can receive messages from various physical sinks in the virtual root strategy. However, although multiple messages are from different sinks, the sensor node only stores one set of RPL information because of the synchronized configuration in the virtual root strategy. In contrast, the sensor node receives two sets of DODAG Version with the same RPL Instance ID but different DAG IDs and Version Numbers in multiple-DODAG strategy. Therefore, the sensor

node in the virtual root network receives messages from various sinks but does not need to maintain each connection between the sensor node and the sink.

In this project, to satisfy the synchronization of the multiple actual sinks, the code of RPL simulation in Contiki-NG needs to be modified. As mentioned before, the main idea is to set the same information for two DODAGs, and the most critical parameter is the DAG ID. Some codes are modified to implement the virtual root strategy based on Contiki-NG.

3.4 Multiple-Instance Strategy

3.4.1 Overview of Multiple-Instance Strategy

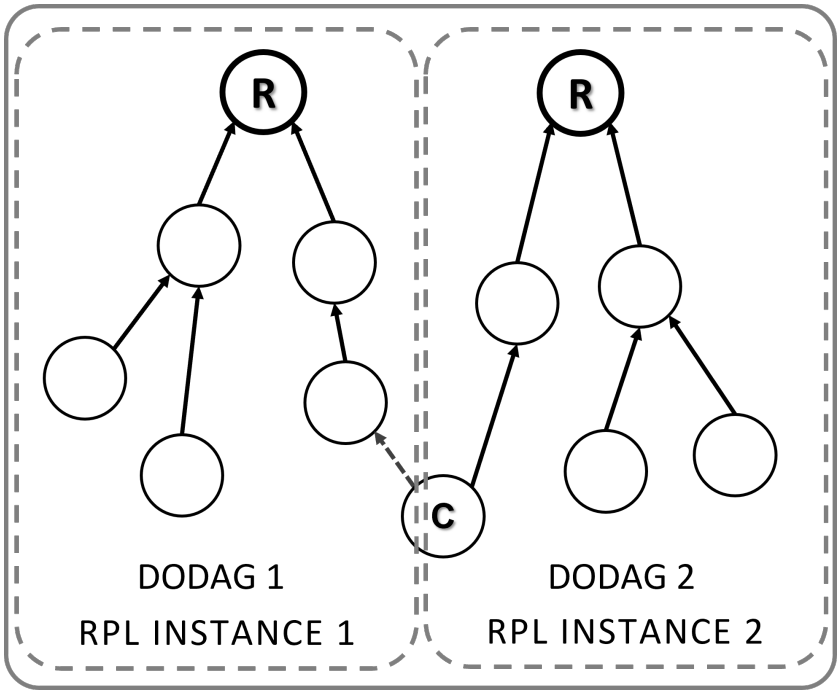

Figure 3.3: Multiple Instances in one RPL Network.

Figure 3.3 shows the last strategy, multiple-instance topology. There are two DODAGs located in two RPL Instances separately with common nodes. In this strategy, two DODAGs are fully independent except for common nodes because of the different RPL Instance IDs and different DAG IDs. An instance can contain multiple DODAGs in the multiple-instance strategy, but a node can only join one DAG per instance. As figure 3.3 shows, like the multiple-DODAG strategy, the common node must store two sets of information,

especially two sets of instance information. Besides the RPL information, nodes in the multiple-instance strategy store multiple sets of parents and routes within different instances. While transmitting packets, the node must determine the next hop according to the specific destination. The multiple-instance mechanism was mentioned in RPL's standard [12], but only a few research focus on the multiple-instance method. Furthermore, the Contiki-NG does not fully support the UDP packet transmission toward multiple sinks in the multiple-instance strategy.

The obvious problem is that the multiple-instance system generally bears high message overhead and transmission costs if the node is registered into two independent instances. Although nodes in the multiple-DODAG strategy also store information from different DODAGs, they only deal with the RPL control packets within one instance. Unlike the multiple-DODAG strategy, nodes in the multiple-instance strategy deal with messages from both instances, which causes a high packet delivery cost.

However, since a node can join multiple instances, the system can be more flexible. In the multiple-instance strategy, two instances can focus on different requirements, which allows a node focuses on different tasks within multiple instances. For example, if a node joins the first instance, it is sensitive to packet transmission latency because of the latency optimization implementation of the first instance. In addition, the node can also join the second instance optimized for delivery reliability [11]. But the flexibility of the multiple-instance strategy is not the main concern of this project. In this project, the research focuses on other performance metrics, including performance of PDR, the number of RPL control packets, and energy consumption in different scenarios.

3.4.2 Deployment of Multiple-Instance Strategy

Since the strategy has multiple instances with several sinks, an interesting discussion is whether a root can participate in other instances and act as a sensor node. This idea refers to the dynamic sink implementation of Voravit T. et al. [27], which allows a sink to act as a sensor node when the traffic pressure is low. However, if the traffic pressure of the network increases, the dynamic sink starts to act as a sink and decrease the traffic pressure on other sinks. Based on previous research, the first implementation allows the root to join other instances. However, in the experiment with the default Contiki-NG system, if a root joins other instances, it will not function as a root node in the following experiment. For example, the sink cannot handle incoming DAO

messages from sensor nodes. According to Voravit's research, root nodes and sensor nodes can decide if a dynamic sink is active as a root node or inactive as a sensor node. In addition, the dynamic research network has an outside coordinator, which is implemented out of the RPL network to receive and deal with the message from the sink in the RPL network. Therefore, the coordinator can analyze the traffic flow, send signal messages to the dynamic sink, and activate the inactive sink with traffic pressure increases. However, since the network is closed in this project, the root node cannot receive signals from the outside coordinator and restart as a sink after joining another instance. Besides the coordinator, allowing nodes to make decisions to restart the root node costs more energy and increases the complexity of the network. As the project aims to compare different strategies, the decision-making pattern is improper because it changes the essential function and logic of root nodes. Therefore, in this project, an instance only has a root node, which is responsible only for its instance rather than joining others.

After determining the root node's behavior, the Contiki-NG source code needs to be modified to satisfy the multiple-instance strategy. Although the Contiki-NG is a system in IoT application and single-instance RPL network, the system cannot fully support the multiple-instance RPL network. There are three main problems of the Contiki-NG to support the multiple-instance strategy, including instance selection, preferred instance determination, and correct route selection.

Firstly, the instance information selection section is unsuitable for the multiple-instance strategy. In the Contiki-NG system, each node has an instance_table structure to store the joined instance. In the default system, a node can store multiple instances' information, including the instance id, version, rank information, and preferred parent. However, there is no proper method to invoke the multiple instances' information, especially in the data packet delivery task from sensor nodes to the root node. For example, before delivering the packet to the root, the node must select the preferred instance and restructure the packet header in the network layer by inserting the instance information. However, the default system cannot check and select the correct instance from the instance table but only select the first value. Since the DAG ID of a DODAG is the same as the IP address of the DODAG's root node, Contiki-NG applies the DAG ID as the root destination while transmitting data packets. Therefore, if the node selects the DAG ID of the second instance in the instance table as its destination address, the DAG ID and the preferred instance are not matched while restructuring the packet header. As a result, the root node can receive the packets but cannot parse the packet because of the

incorrect instance information. Consequently, selecting the correct instance from the instance table according to the destination address is important. As the destination address is determined with the imperfect instance selection, the modification checks the instance table to find the matched instance of the destination address. In more detail, firstly, the destination address is used as the DAG ID to get the DAG to which it belongs. Then, the instance id stored in the DAG is applied to find the correct instance from the instance table. At last, the correct instance information is inserted into the packet header before transmitting the packet.

Even if the problem of matching the destination address and the instance information has been solved, destination determination is the second problem. Since the destination address is the same as the DAG ID of the root's DODAG, the destination of packet delivery is related to the DAG. Since each instance contains only one DAG, the destination decision is ultimately related to the preferred instance. However, the Contiki-NG system does not consider the multiple-instance situation like the first problem. Instances are not selected according to specific rules but based on their sequences in the instance table. Therefore, if the first instance in the instance table is not the best option, the sensor node may still select it as the preferred instance and choose the DAG ID in the instance as the destination address. Therefore, it is important to establish rules allowing each node to distinguish its preferred instance and find the corresponding destination address in packet delivery tasks. In this project, the preferred instance is determined when the sensor node joins the instance. If the sensor node joins the first instance, it considers that instance as its preferred instance. If the sensor node joins a second instance, it compares the rank value of the two instances and selects the instance with the smaller rank value as the preferred instance. Specifically, the sensor node stores the rank value of the first instance, and when it joins a second instance, it checks whether the rank value of the second instance is smaller than that of the first one. If it is, the sensor node chooses the second instance as its preferred instance. Therefore, the sensor node can properly determine its preferred instance and select the corresponding DAG ID as the destination address.

Although packets are transmitted toward the proper destination with the correct packet header, route selection is the third problem in the project. Like the previous two issues, route storage and selection are unsuitable for multiple instances. Therefore, before transmitting data packets, packets must be restructured with instance information from the RPL layer and delivered along the route in the uIP stack, similar to the network layer. In the RPL layer, the default route can be distinguished with different instances and stored

separately. However, the uIP stack cannot store multiple default routes related to multiple instances. For example, if a sensor node joins two instances, the uIP stack only stores the second instance's default route but ignores the first one's. Therefore, enabling the uIP stack to distinguish and store multiple default routes related to multiple instances is crucial. With this modification, the uIP stack can determine the correct route according to the instance and deliver the packet towards the correct destination. The default route structure in the uIP stack was modified to support multiple instances to achieve the determination function. In addition to the storage structure, each default route has a new direction parameter representing the corresponding instance. Since the direction parameter is same as the instance ID, the default route is determined by referring to the instance ID before transmitting packets toward the correct destination.

Chapter 4

Topology and Experiment Metrics

4.1 Network Topology

This section will introduce the physical topology of experiments. As this project aims to compare the performance of different RPL strategies and define which strategy suits a specific scenario, various topologies in this section are as similar to the practical application or common situation as possible.

4.1.1 Grid Topology

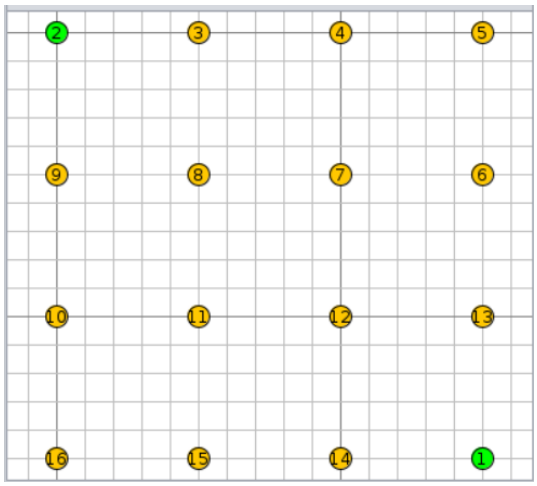

Figure 4.1: Grid Topology

Grid topology can be considered placing sensors evenly over an area as a grid topology, with two sinks and 14 sensor nodes. As figure 4.1 shows, node 1 and node 2 are two sinks in this topology. In addition, the topology is formed as a 4x4 square with two sinks set at both ends of the diagonal. All nodes are distributed uniformly in this topology, with the same intervals between every two nodes. In this topology, nodes at the corner have two neighbors, nodes at the sides have three neighbors, and nodes in the middle of the square have four neighbors.

4.1.2 Linear Topology

Linear topology is to place sensor nodes evenly along a straight line, with two sinks and 14 sensor nodes. Node 1 and node 2 are two sinks placed at the quarter position of the line. As figure 4.2 shows, node 1 is placed at the fifth position from the left end, while node 2 is placed at the fifth position from the right end. In this topology, every node has two neighbors except for the two end nodes, which only have one neighbor.

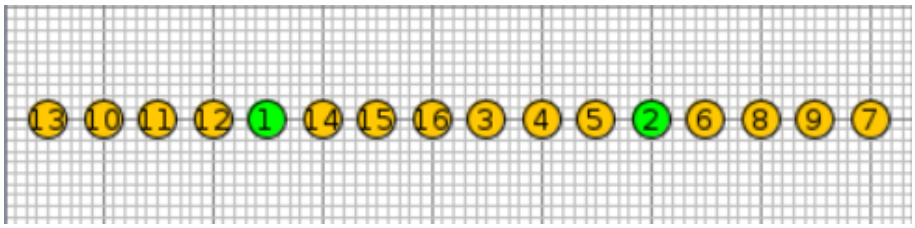

Figure 4.2: Linear Topology

4.1.3 Crossing Topology

Crossing topology can be considered a scenario that places several sensors on street lights around a crossing road. This topology also has two sinks and 14 sensor nodes, as figure 4.3 shows.

The two nodes are placed at the quarter position of the vertical and horizontal lines separately. As figure 4.3 shows, root node 1 is placed at the third position from the top, while root node 2 is placed at the second position from the left. In this topology, four end nodes only have one neighbor, the center node has four neighbors, and the other nodes all have two neighbors.

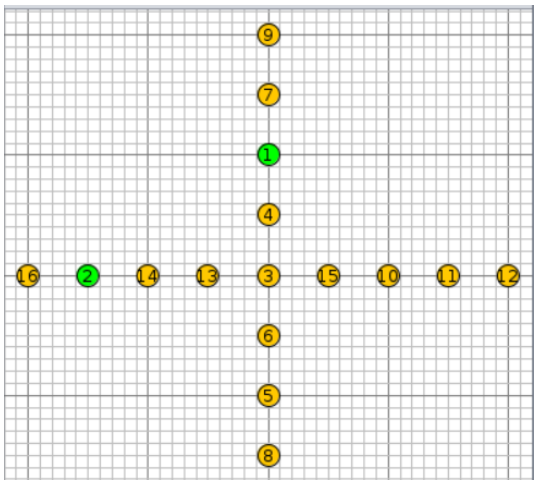

Figure 4.3: Crossing Topology

4.1.4 Circular Topology

The circular topology includes two sinks and 14 sensor nodes. As figure 4.4 shows, all nodes in this topology form a circle. In this topology, two sinks are placed at the y-axis, and three sensor nodes are placed in each quadrant. In this topology, all nodes are uniformly distributed according to the angle. For example, if drawing lines to connect sensor nodes and the circle's center, the angles between the line and the x-axis are $22.5°$, $45°$, and $67.5°$.

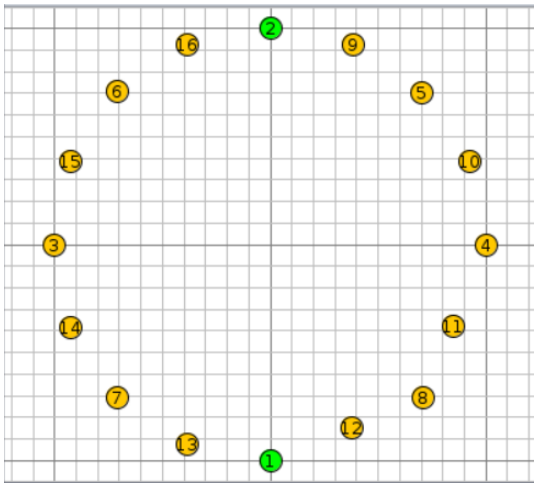

Figure 4.4: Circular Topology

4.1.5 Random Topology

The last topology is the random topology that does not have regular patterns but contains two sinks and 14 sensor nodes at stochastic positions, as figure 4.5 shows.

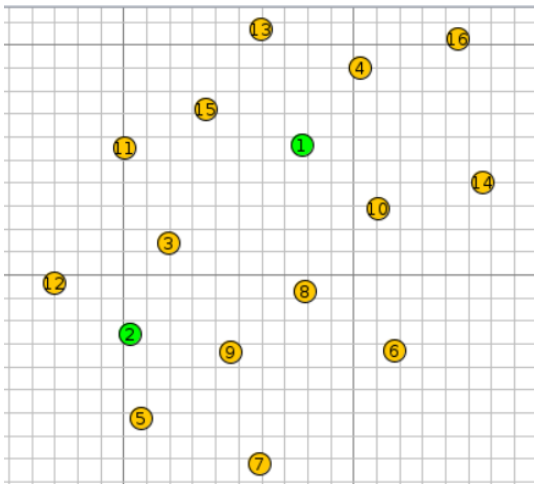

Figure 4.5: Random Topology

Therefore, nodes are not in specific positions in this topology and do not have uniform distances from other nodes. In addition, most nodes have two or three neighbors.

4.2 Experiment Metrics

This project has three metrics to compare the performance of different strategies: total number of RPL control packets, Packet Delivery Ratio (PDR), and energy consumption.

4.2.1 Total Number of RPL Control Packets

The first evaluation calculates the total number of RPL control packets, including DIS packets, DIO packets, and DAO packets, as mentioned in section 2.2.4. The number of RPL control packets represents the overhead of establishing and maintaining the whole network, which is a critical parameter related to the traffic load in practice. Therefore, if a strategy has a lot of RPL control packets, the strategy will cost many resources and much energy to maintain the connection between nodes and take up much traffic overhead.

For example, if one network produces much more RPL control packets than the other, the network's performance may be affected.

4.2.2 Packet Delivery Ratio (PDR)

The second evaluation is to calculate the Packet Delivery Ratio (PDR). In RPL networks, sensor nodes usually transmit data packets to the root node. Therefore, the PDR means the ratio of the total packets received from sensor nodes to the total packets sent to sinks in the RPL network, as formula 4.1 shows [43]. Therefore, the PDR represents the communication quality in the network.

$$Packet\ Delivery\ Ratio = \frac{\sum Number\ of\ packet\ receive}{\sum Number\ of\ packet\ send} \tag{4.1}$$

4.2.3 Energy Consumption

Energy consumption is a critical and common metric for evaluating the performance of networks, which is also related to the deployment, availability, and even costs of practical applications. As for the IoT network, energy evaluation can be divided into root node consumption and sensor node consumption. Some research focuses on comparing the different nodes' performances under different conditions. Meanwhile, some evaluation only focuses on the energy consumption of the whole network. In this project, the energy consumption metric of a network is the total energy cost of all nodes in the network.

Chapter 5

Experiment Setup

This chapter aims at the detailed setup, mainly containing the configuration with Cooja. The experiment of each topology includes the comparison among multiple-DODAG strategy, virtual root strategy, and multiple-instance strategy with three different packet loss values. The following sections indicate three main implementation areas to build the experiment: topology implementation, node implementation, and data collection from three experiment metrics.

5.1 Topology Implementation

All topologies are deployed on Cooja, with 2 sinks and 14 sensor nodes in each topology. The distance between every two nodes is the same as the transmission range of the node except for the random topology and circular topology. Modifying the simulation file allows the nodes to be placed precisely at the required place by setting the coordinate parameter.

The distance between every two sensor nodes differs for the random and circular topology. Since the project uses the constant loss model, the reception successful ratio is a specific value if the node is only located in the transmission range. In circular topology, the parameter is calculated according to the trigonometric function. In random topology, the topology was built by the random placement function in Cooja. However, some adjustment is necessary to ensure that a node is located in the transmission area of the nearest nodes in the topology.

5.2 Node Implementation

Z1 motes were used to compile all nodes in this project, and modifications were made to add additional functions to complete all experiments. This section introduces several parameters of the node implementation, including the configuration of Contiki-NG, transmission and reception successful ratios, transmission range, the processing time of the experiment, and the sending rate of UDP data packets.

It is necessary to specify the configuration of the Contiki-NG system. To ensure simplicity in the experiment implementation, most parameters and configurations remain at their default values in the Contiki-NG system. For instance, the Contiki-NG offers two implementations of RPL: RPL lite and RPL classic [15]. Since RPL lite does not support the multiple-instance implementation, this project applies the RPL classic pattern. In addition, Contiki-NG provides various implementations of the MAC layer [44]. This project applies the Carrier-Sense Multiple Access (CSMA), the default implementation in the Contiki-NG system. Besides the configuration, the Contiki-NG provides a retransmission function to support packet transmission tasks, which allows the packet to be retransmitted several times. However, the PDR is usually a high value with a large number of packet retransmission. Therefore, this project's retransmission parameter is 2, smaller than the default value.

An essential parameter in the simulation experiment is packet loss since communication experiences some loss in practical applications. In this project, the packet loss is emulated by changing the transmission and reception successful ratio. In all experiments, the transmission successful ratio is set to 100%, while the reception successful ratio is set from 90% to 70% with a 10% decrease per step. Specifically, the reception successful ratio corresponds to the following packet loss values: 10%, 20%, and 30%.

The third parameter is the transmission range, which indicates the radio range of each node, as discussed in section 2.3.2. In this project, the transmission range is set to a default value of 50 m for all experiments.

The processing time of every experiment is 30 minutes and 5 seconds, which allows the energy calculation function to print the last piece of data in the log file. Since the energy calculation function records the energy consumption per minute, the last 5 seconds allow the energy consumption module to record all logs. In the experiment, the sending rate of sensor nodes is set to one packet per second, which equates to 60 packets per minute. During the first two minutes, sensor nodes do not send UDP packets but wait for the

RPL network to converge. After the network converges, each node sends one packet per second for 27 minutes and 30 seconds, which is 1650 UDP packets in total. Then the final UDP packet is transmitted at 29 minutes and 30 seconds. The experiment then runs longer to ensure the root nodes receive all packets.

5.3 Data Collection

This section introduces the data collection implementation in the project, which includes the number of RPL control packets, the PDR, and the energy consumption as mentioned in Section 4.2. In addition, all experiments repeat three times to get the average PDR and average energy consumption results.

5.3.1 Total Number of RPL Control Packets

The Cooja has a function of recording all radio messages and storing the messages in a packet capture file, which Wireshark can retrieve. Then, the number of RPL control packets can be filtered using Wireshark. Therefore, the figure includes the RPL control packets of three strategies under different packet loss conditions. In this experiment, the number of RPL control packets is from one experiment rather than the average value of multiple experiments. Because the number is specific to an experiment, and the average value is meaningless. Nevertheless, the experiments in the different rounds have the same trend in the number of RPL control packets metric.

5.3.2 Packet Delivery Ratio (PDR)

The second evaluation calculates the Packet Delivery Ratio (PDR) by counting the number of delivered UDP data packets. In this project, the sensor node and the sink are modified to send and receive UDP packets and record the number.

The sensor node is modified to send UDP packets with the increment sequence number and record the transmission event in the log file. Correspondingly, the sink records the message with the sequence number stored in the packet in the sink's log file. Therefore, if a UDP packet cannot be delivered to the sink, the sensor node's log file still records the event with a sequence number, but the sink will miss the message. Figure 5.1 shows the whole process. According to log files, the PDR can be calculated directly.

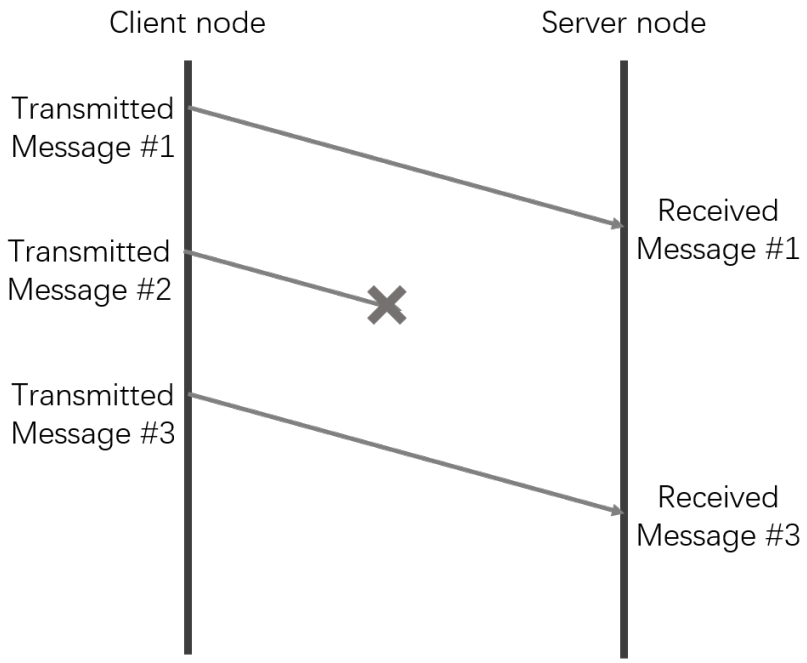

Figure 5.1: UDP Transmission Process

5.3.3 Energy Consumption

The third evaluation is energy consumption. An energy test module called Energest is integrated into Contiki-NG and can be applied in Cooja. On the simulation platform, the Energest module can calculate the time a system has spent in multiple states [45]. There are five predefined Energest modes: CPU in active, CPU in low power mode, CPU in deep low power mode, radio transmission (Radio Tx), and radio listening (Radio Rx). In addition, the simple Energest module can record summary messages once per minute. A message includes the sequence number of this message, the total number of ticks in the accounting period, and the number of ticks in different modes [15]. Therefore, the time spent in each state can be calculated according to the ticks. Finally, the current consumption can be calculated using the time distribution for each state with a platform-specific current consumption model. As this project applies Z1 mote, the corresponding current consumption model, as table 5.1 shows, must be applied to calculate the energy. Since Z1 mote does not include the CPU deep low power mode, only four states are in the table. Therefore, the energy consumption can be calculated by the current consumption, processing time, and voltage, as shown in formula 5.1. In this equation, T_a, T_l, T_r, and T_t represent the processing time of four states: CPU

active, CPU low power mode, Radio Rx and Radio Tx. Meanwhile, I_a, I_l, I_r, and I_t represent the current consumption of four states, and V is the voltage provided by the system to the component, which is 3.0 Volts for Z1 mote. Therefore, the current consumption variable is substituted by the data in table 5.1 and conclude the final equation.

State	Current Consumption
CPU active	10 mA
CPU low power mode	23 uA
Radio Rx	18.8 mA
Radio Tx	17.4 mA

Table 5.1: State and Current Consumption of Z1 Mote

$$\begin{aligned} E &= (I_a \times T_a + I_l \times T_l + I_r \times T_r + I_t \times T_t) \times V \\ &= (10T_a + 0.0023T_l + 18.8T_t + 17.4T_r) \times 3.0 \end{aligned} \tag{5.1}$$

Chapter 6

Results and Analysis

6.1 Overview

This chapter introduces the result and data analysis with different topologies. In each section, the result and figure are shown first, followed by the analysis based on results. As mentioned before, each topology has three types of results, with one figure in one type of result. Therefore, the analysis is also based on the three parameters and focuses on comparing the strategies. As each experiment with the same parameters repeats three times, this project applies the average value to indicate the PDR and energy consumption results. In addition, the variances are also marked in the figures. However, the number of RPL control packets is from one experiment rather than the average value, as mentioned in section 5.3.1.

The abbreviation in the figure needs to be expressly indicated:

- **MD** means Multiple-DODAG strategy in an instance.
- **VR** means Virtual Root strategy.
- **MI** means Multiple-Instance strategy.

6.2 Grid Topology

6.2.1 Result

With the packet loss increase, the total number of packets increases, as figure 6.1 shows. The number of packets in the multiple-instance strategy is much more than in the other two. In addition, the number of packets in

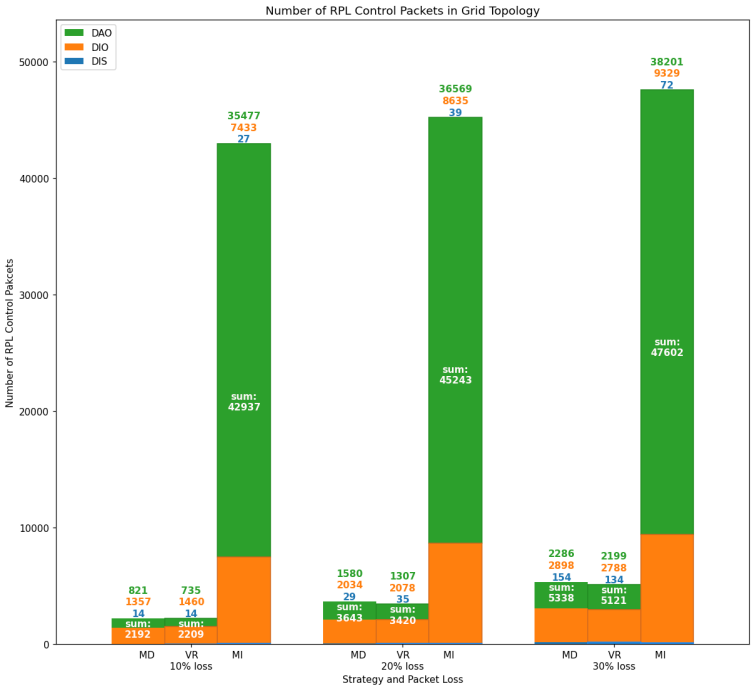

Figure 6.1: Number of RPL Control Packets in Grid Topology

the virtual root strategy is smaller than that in the multiple-DODAG strategy, especially under high packet loss conditions. Since a node transmits the DIS message to find the nearby DODAG, the message only occurs if a node is not registered in a DODAG. Therefore, the DIS packets only occur in the construction period in the beginning with the high packet loss because nodes are always connected with the DODAG during the experiment. However, DIS packets occur during the experiment with high packet loss because some nodes are disconnected from the DODAG and need to join the DODAG again. In addition, with the increase in packet loss, the difference between the number of packets in the multiple-DODAG and virtual root strategies becomes larger.

As figure 6.2 shows, the second result is PDR. The virtual root strategy performs better than the multiple-DODAG strategy in packet delivery ratio, especially under high packet loss conditions. The multiple-DODAG strategy only performs slightly better than the virtual root strategy with low packet loss conditions. With the packet loss increases, the PDR performance of the virtual root becomes much better. In addition, the multiple-instance strategy performs worse than the other two under low packet loss conditions but performs better with high packet loss.

The last parameter is the energy consumption of the RPL network. As

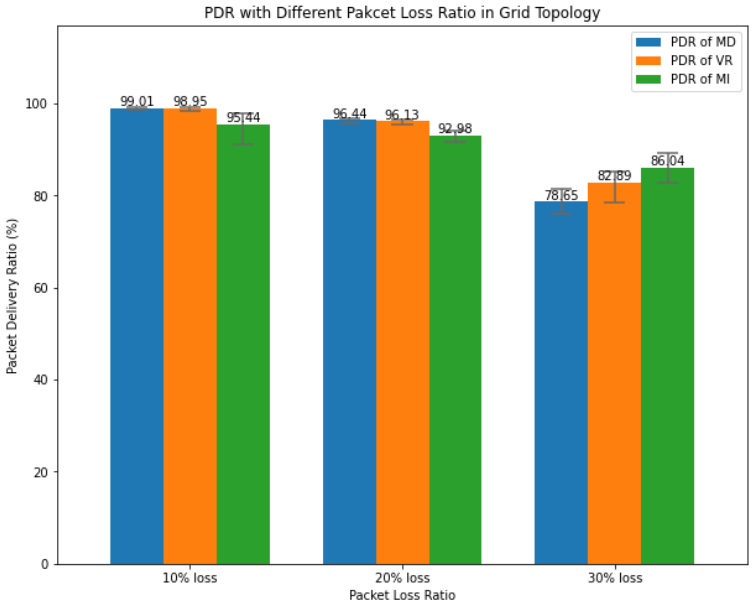

Figure 6.2: Packet Delivery Ratio in Grid Topology

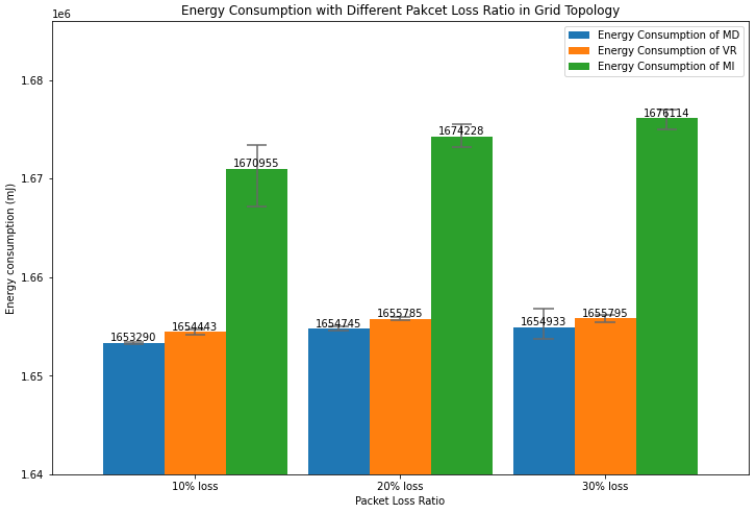

Figure 6.3: Energy Consumption in Grid Topology

figure 6.3 shows, the multiple-instance strategy costs much more energy than the other two. Moreover, besides the multiple-instance strategy, the virtual root strategy performs worse than the multiple-DODAG strategy, costing more energy.

6.2.2 Data Analysis

According to the result and figures previously, the multiple-instance strategy performs much worse than the other two in all three fields except for the PDR performance with high packet loss. Since the multiple-instance network contains two instances, nodes must transmit many packets to maintain two instances. As packet transmission and reception tasks cost energy, multiple-instance implementation costs a lot of energy with high packet overhead. However, the multiple-instance strategy has better PDR performance with the high packet loss condition. Compared to the multiple-instance strategy, the PDR of multiple-DODAG and virtual root strategy decreases significantly with the high packet loss. One reason is that sensor nodes change the destination of data packet transmission according to the parent and route switch, which is mainly related to the loop detection function. In the loop detection function, if a node detects its parent has a higher rank value than the threshold, the node will change the preferred parent and route. The route switch function can maintain the packet transmission task with low packet loss because the network has a high-quality connection. However, while the whole network has a high packet loss condition, nodes cannot have high-quality communication even if they change the transmission route. In addition, the loop detection function may pause a node's packet transmission task because the node needs time to establish a new route. The packet transmission task is significantly affected if the packet loss condition negatively affects the route switch processing. Therefore, the implementation causes some packets not to arrive at the destination with a high packet loss ratio during the switching period. As a result, the PDR performance of the multiple-DODAG and virtual root strategy decreases significantly with the high packet loss.

In addition to the multiple-instance strategy, the virtual root strategy performs better than the multiple-DODAG strategy in both packet overhead and packet delivery fields. However, the network with the virtual root strategy costs more energy than the other. As for the number of RPL packets, the number of DAO messages is the main difference between the two strategies. Since sensor nodes must send DAO messages to multiple sinks to maintain the connection within two DODAGs, the DAO messages in the multiple-DODAG strategy are more than in the virtual root strategy. As for energy consumption, the multiple-DODAG strategy performs better than the other. One reason is that the virtual root costs more energy in the loop detection function and parent switch process. Since the virtual root network acts as a big DAG, nodes in this network have more opportunities to change the parent and the route when the

parent's rank value is high. Meanwhile, nodes cannot change the route simply in the multiple-DODAG network because the neighbor node may belong to the other DAG. Therefore, the node in the multiple-DODAG strategy cannot send packets correctly. As a result, the number of data packets in multiple-DODAG decreases, resulting in low PDR and low energy.

6.3 Random Topology

6.3.1 Result

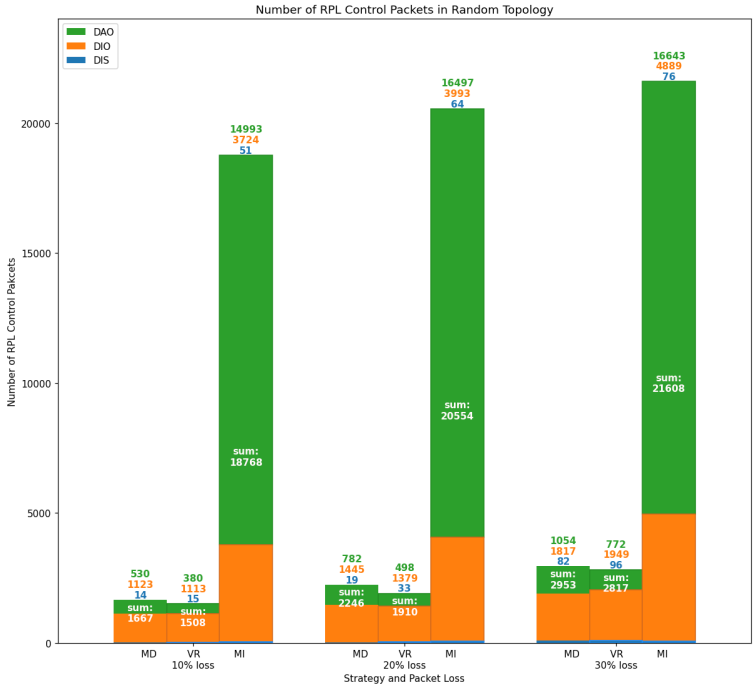

Figure 6.4: Number of RPL Control Packets in Random Topology

With the packet loss increases, the total number of packets increases, as figure 6.4 shows. The virtual root strategy performs better than the multiple-DODAG strategy. The number of DAO messages of the virtual root strategy is smaller than that of the multiple-DODAG strategy. In addition, the number of RPL control packets in the multiple-instance strategy is significantly more than in the other two.

As figure 6.5 shows, the multiple-instance strategy performs slightly better than the other two strategies with high packet loss but much better under high

packet loss conditions. In addition, the difference in PDR performance is not apparent for the virtual root and multiple-DODAG strategies under low packet loss conditions. But the PDR performance of the virtual root becomes better than the multiple-DODAG strategy with the high packet loss.

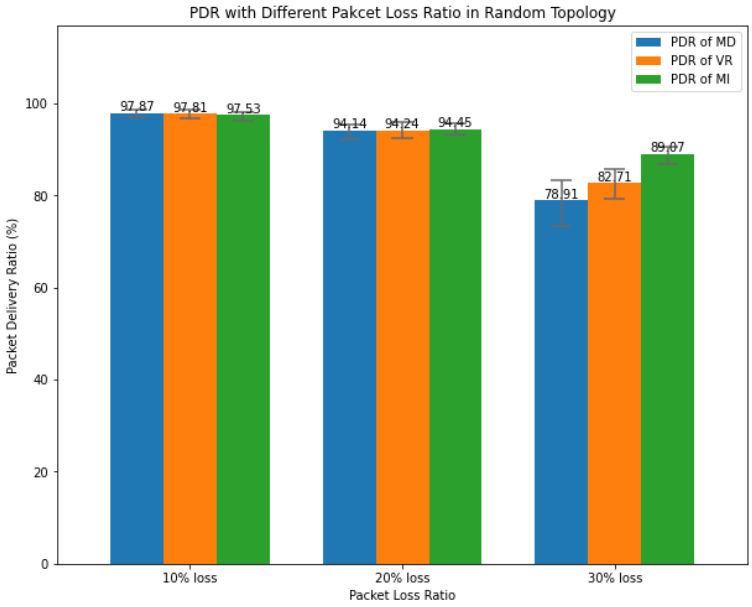

Figure 6.5: Packet Delivery Ratio in Random Topology

The last parameter is the energy consumption of the RPL network. As figure 6.6 shows, the energy consumption increases with the increasing of packet loss ratios. The virtual root strategy costs more energy than the multiple-DODAG strategy. Meanwhile, the multiple-instance strategy costs much more energy than the other two.

6.3.2 Data Analysis

As for the multiple-DODAG and virtual root strategies, the previous results and figures show that the virtual root strategy performs better than multiple-DODAG in the number of RPL control packets field. In the multiple-DODAG strategy, common sensor nodes must communicate with the backup sink to maintain the backup connection. Therefore, the number of DAO messages in the multiple-DODAG strategy is higher than in the virtual root strategy. As for the packet delivery ratio, both strategies perform well with low packet loss. However, the PDR decreases obviously with high packet loss. One main reason is that the root switch period negatively influences the packet

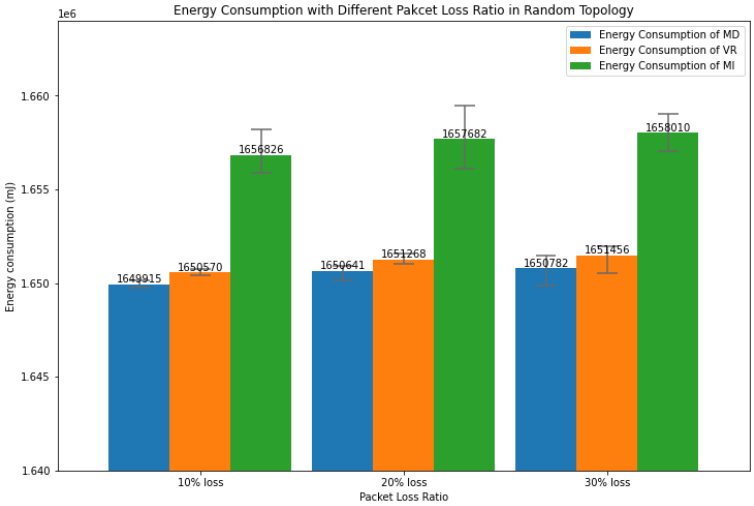

Figure 6.6: Energy Consumption in Random Topology

delivery ratio. Regarding energy consumption, the multiple-DODAG strategy costs less energy than the virtual root strategy. One possible reason is that the virtual root network nodes can change the parent and route within the DAG. Therefore, the virtual root network produces more packets with the retransmission function, resulting in high PDR and energy consumption.

In addition, the multiple-instance strategy produces significantly more RPL control packets than the other two because each node joins and maintains two instances. Since the packet transmission task costs energy, more packets in the multiple-instance network cost more energy. However, the multiple-instance strategy has good PDR performance with high packet loss because the preferred instance implementation reduces sink exchange. The parent switch processing may be affected negatively under the high packet loss condition since the switch processing costs time and pauses the packet forward task, resulting in low PDR. Therefore, the multiple-instance strategy has good PDR performance because nodes do not change the parent and route.

6.4 Linear Topology

6.4.1 Result

With packet loss increases, the total number of packets increases, as figure 6.7 shows. The multiple-instance strategy produces the most RPL control packets. Apart from the multiple-instance strategy, the difference in performance

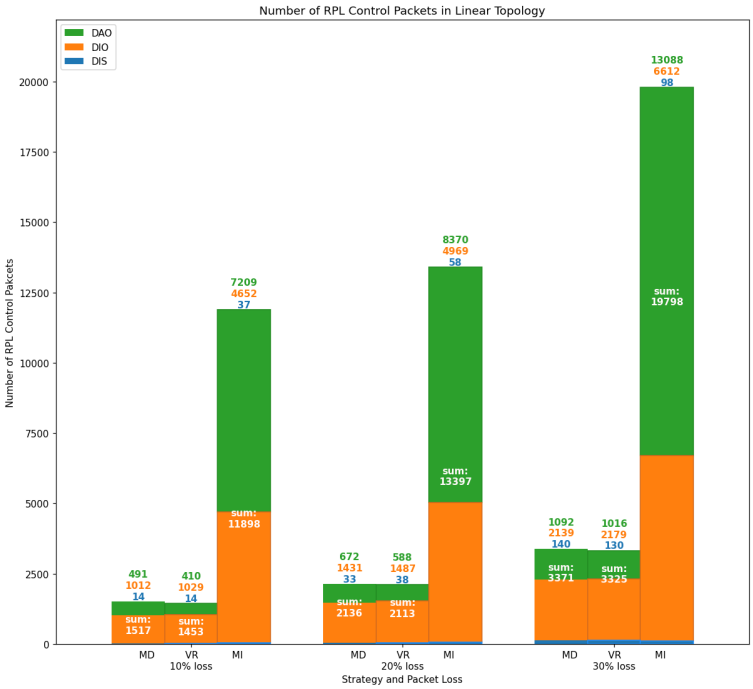

Figure 6.7: Number of RPL Control Packets in Linear Topology

between the other two strategies is not apparent. The virtual root strategy produces fewer RPL control packets than the multiple-DODAG strategy with low and high packet losses but more packets with medium loss. The figure shows that the number of DIO messages in the virtual root strategy is larger than in the multiple-DODAG strategy. In addition, the multiple-DODAG strategy has more DAO messages than the virtual root strategy.

As figure 6.8 shows, the performance of three strategies in PDR varies depending on loss conditions. As the packet loss ratio increases, the performance difference becomes more apparent. All three strategies have good PDR performance with low packet loss, whereas the multiple-instance strategy performs better than the other two with high packet loss. One particular case is that the variance of the multiple-instance strategy is significantly a significant value. Besides the multiple-instance strategy, the virtual root strategy performs better than the multiple-DODAG strategy with high packet loss.

The final parameter is the energy consumption of the RPL network. As shown in figure 6.9, energy consumption increases as packet loss increases. The multiple-instance strategy costs much more energy than the other two strategies. In addition, the multiple-DODAG strategy performs better than the

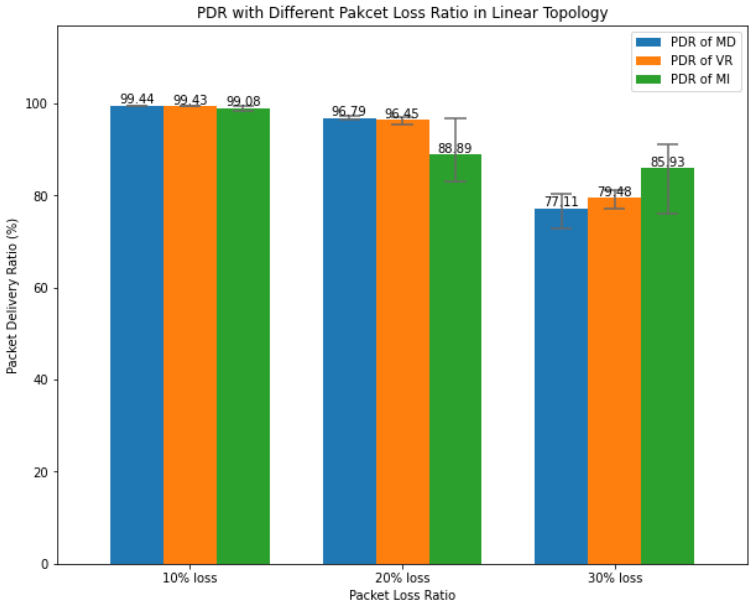

Figure 6.8: Packet Delivery Ratio in Linear Topology

virtual root strategy with less energy consumption.

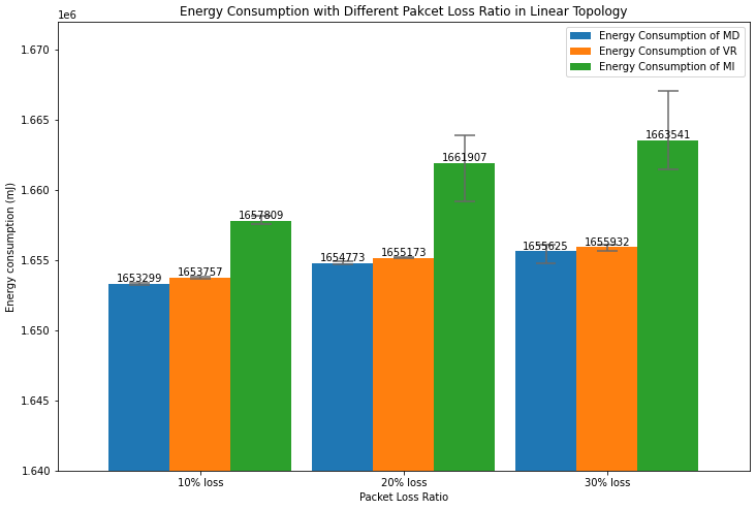

Figure 6.9: Energy Consumption in Linear Topology

6.4.2 Data Analysis

The multiple-instance strategy performs better in the PDR field with high packet loss because the linear topology is a simple symmetric topology.

Except for the lowest packet loss condition, the multiple-instance strategy has a significant variance in PDR and energy consumption metrics. As the paths are limited, connection maintenance and rebuilding or traffic congestion affects the whole network. One possible reason for the high variance is that different experiments have different events affecting the connection, resulting the different performances and high variance. Depending on the root selection algorithm of the multiple-instance strategy, the data packet transmission is simple without determining and switching the root, leading to good PDR performance in the multiple-instance strategy with high packet loss. However, the energy consumption is much more than the other two strategies because of the large number of RPL control packets.

Besides the multiple-instance strategy, the virtual root strategy performs better than the multiple-DODAG strategy in the RPL control packet field but without much advantage. In contrast, the multiple-DODAG strategy has good performance in the energy consumption area. As for the number of DAO messages, sensor nodes need to transmit DAO messages to DODAG roots to maintain the connection and two DODAGs. One single node can handle the transmission overhead since packets are only transmitted from two directions. Under high packet loss conditions, the virtual root strategy performs better than the multiple-DODAG strategy. Since the virtual root network has a big DAG, common nodes can change the parent simply within a DAG, resulting in a stable connection and high PDR. However, the parent switch produces more packets and retransmission packets than the multiple-DODAG strategy, which costs more energy. The topology is simple with limited paths, so the virtual root network produces more control and data packets, but the difference is insignificant. Therefore, the difference between the two strategies' energy consumption performance is insignificant.

6.5 Circular Topology

6.5.1 Result

With the increase of packet loss, the total number of packets increases, as figure 6.10 shows. The multiple-instance strategy produces significantly more RPL control messages than the other two. In addition, the difference in RPL control packets between the virtual root strategy and multiple-DODAG strategy is not apparent.

As shown in figure 6.11, the difference between the multiple-DODAG and virtual root strategies is insignificant regarding PDR performance under low

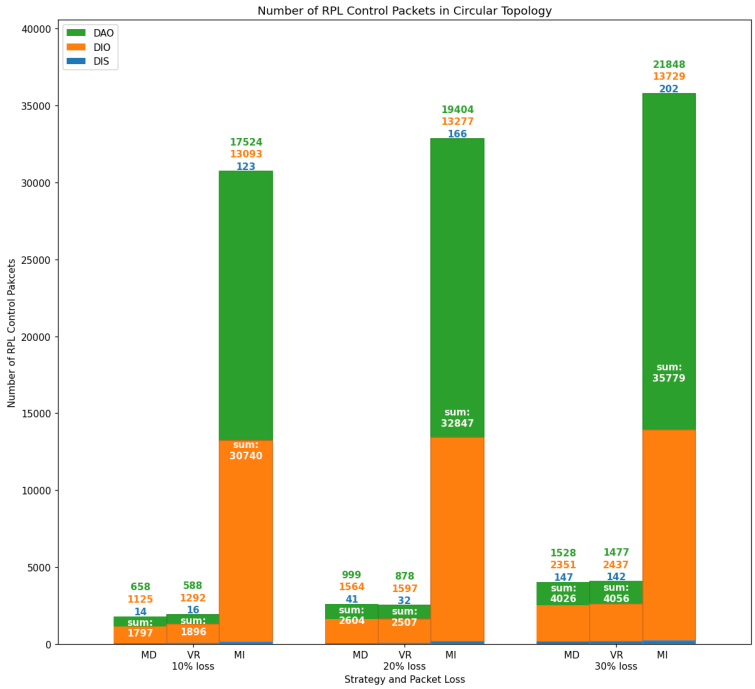

Figure 6.10: Number of RPL Control Packets in Circular Topology

packet loss conditions. However, the multiple-DODAG strategy demonstrates better PDR than the virtual root strategy, especially in high packet loss conditions. Furthermore, while the PDR performance of the multiple-instance strategy has worse PDR performance than the others under low packet loss, it outperforms the other two strategies in high packet loss scenarios.

Figure 6.12 shows the energy consumption in circular topology. Compared to the multiple-DODAG strategy, the virtual root strategy costs more energy in the low packet loss scenario but less energy than the multiple-DODAG strategy with the high packet loss. In addition, the multiple-instance strategy costs much more energy than others.

6.5.2 Data Analysis

In the circular topology, the virtual root strategy performs worse than the multiple-DODAG strategy in all three aspects. The simple topology allows sensor nodes to determine two paths towards the primary and secondary sinks in two directions without interference. However, according to the data packet record file, sensor nodes between the two sinks switch the destination more frequently because of the route switch or loop detection function in the virtual

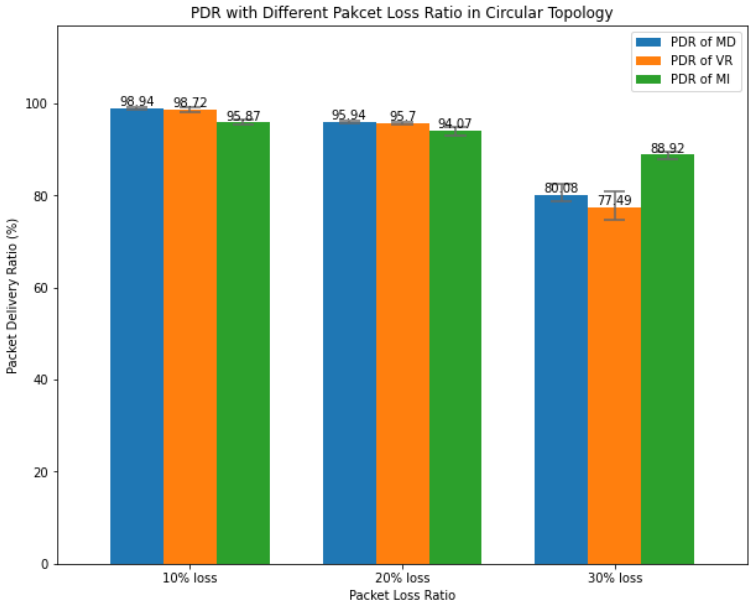

Figure 6.11: Packet Delivery Ratio in Circular Topology

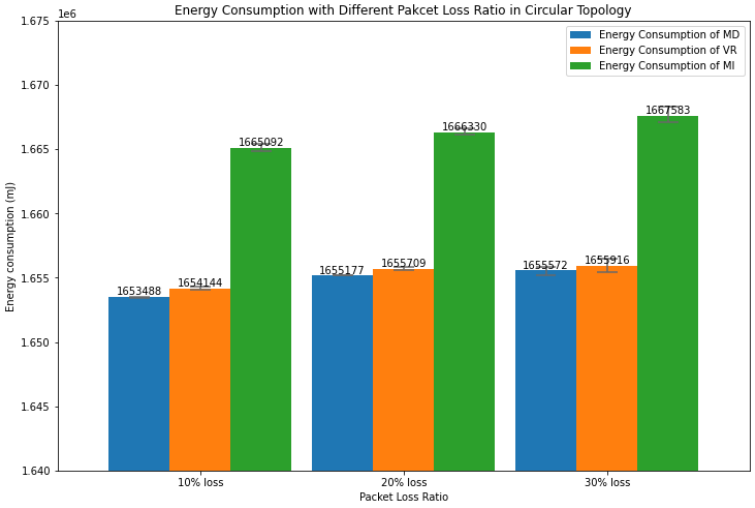

Figure 6.12: Energy Consumption in Circular Topology

root network, resulting in higher packet loss during destination exchange. In addition, as all nodes are common in the topology, the whole network has a big DAG and many node hops in the virtual root strategy. Therefore, if a sensor node detects a large rank value, it will choose the other neighbor. If the other neighbor has a big rank value either, the node must pause and wait for the new version to be updated. This problem can be solved but need some

time. Since the packet transmission speed is high and the virtual root network has a big DAG, the virtual root strategy causes low PDR. Consequently, the multiple-DODAG strategy performs better PDR than the virtual root strategy. Additionally, the virtual root strategy consumes more energy compared to the multiple-DODAG strategy.

As for the multiple-instance strategy, it performs worse than the other two strategies regarding RPL control packets and energy consumption. The establishment and maintenance of two instances provide a high overload of RPL control packets, leading to decreased PDR performance in low packet loss conditions. However, due to the implementation of preferred instances, the strategy performs better in the PDR field than the others in a high packet loss scenario.

6.6 Crossing Topology

6.6.1 Result

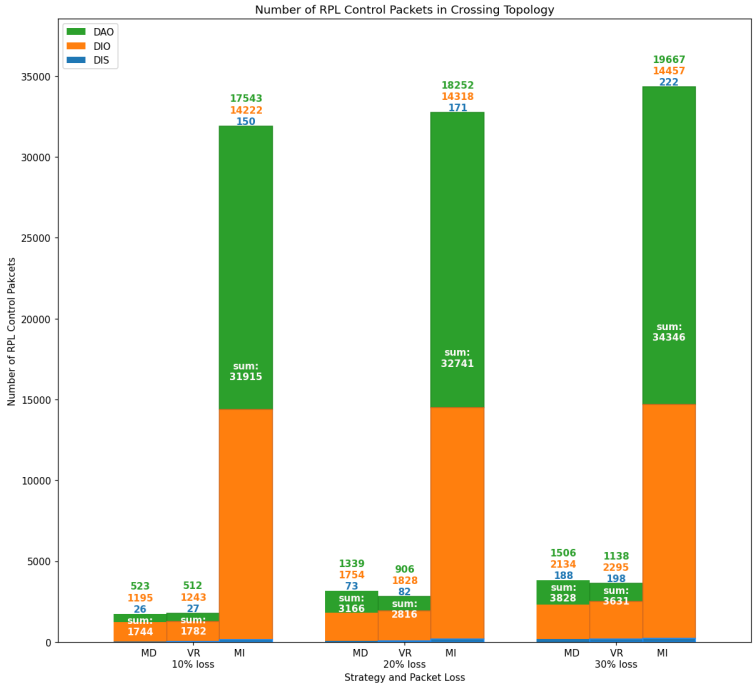

Figure 6.13: Number of Packets in Crossing Topology

As figure 6.13 shows, the virtual root strategy produces fewer RPL control packets than the multiple-DODAG strategy. The difference in DAO messages

is most significant in the two strategies. However, the virtual root strategy has more DIS and DIO messages than the multiple-DODAG setup. In terms of the multiple-instance network, the number of RPL control packets in the multiple-instance strategy is much more than in the other two.

As figure 6.14 shows, the PDR performance decreases significantly with the packet loss increase. The virtual root strategy performs better than the multiple-DODAG strategy. However, the PDR performance of both the virtual root and multiple-DODAG strategies decreases significantly with high packet loss. As for the multiple-instance strategy, the PDR performance is worse than the other two strategies with low packet loss conditions. However, the multiple-instance strategy has the best PDR performance among all three strategies with the high packet loss.

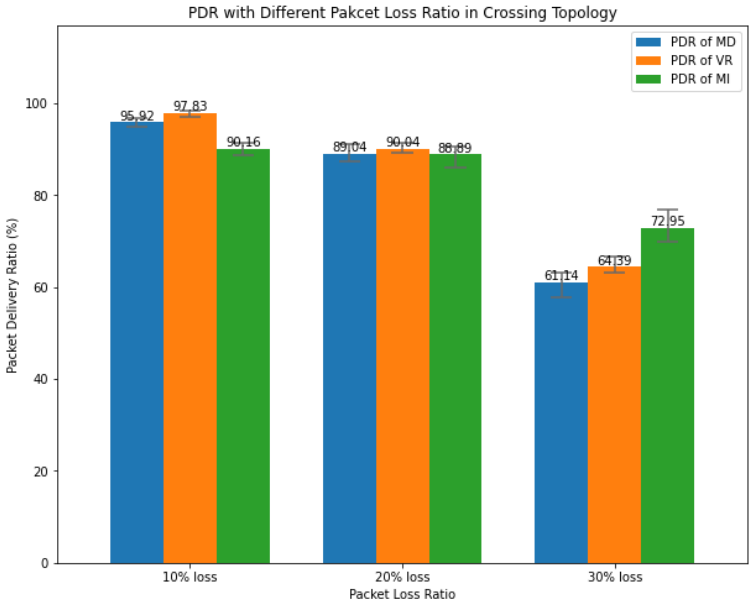

Figure 6.14: Packet Delivery Ratio in Crossing Topology

For the energy consumption, as the figure 6.15 shows, the multiple-instance strategy costs the most energy under each packet loss condition. In addition, the virtual root strategy costs more energy than the multiple-DODAG strategy, but the difference is insignificant.

6.6.2 Data Analysis

According to the results and figures, the virtual root strategy performs better than multiple-DODAG except for energy consumption, even if the strength is

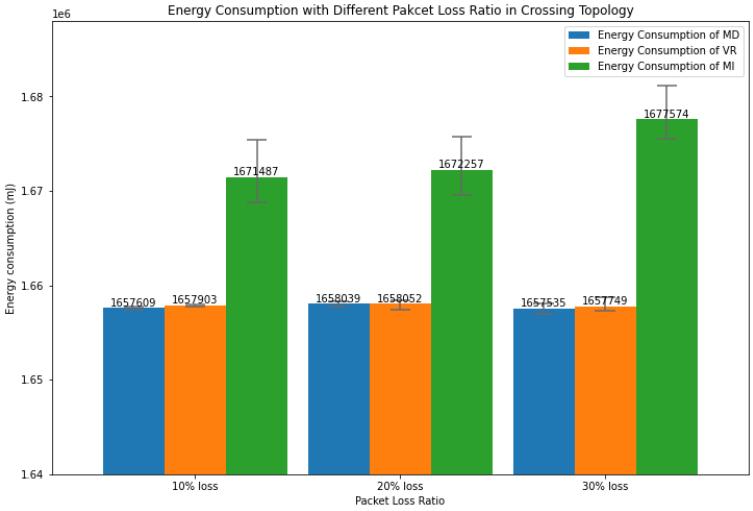

Figure 6.15: Energy Consumption in Crossing Topology

insignificant. However, both virtual root and multiple-DODAG strategies have significantly bad PDR performance with high packet loss, while the multiple-instance performs well. One possible reason is that the sensor node, located at the center, affords much traffic pressure from all four directions, leading to significant packet loss during the destination exchange. Another factor is that some nodes are located far away from two sinks. Therefore, the virtual root and multiple-DODAG strategy has bad performance with high packet loss because of the long distance, destination switch, and traffic pressure on the central node. Meanwhile, since the multiple-instance arranges the preferred instance, the packet loss during the destination switch period does not affect the data packet transmission, leading to higher PDR than the other two strategies. However, since the PDR decreases significantly under the high packet loss condition, the long distance and traffic pressure factors also negatively affect the packet delivery ratio in the multiple-instance network.

6.7 Discussion

To summarize the above sections, some regular patterns can be concluded. This section mainly discusses the virtual root and multiple-DODAG strategies. In addition, it just concludes with an overview of the multiple-instance strategy since it produces significant RPL control packets and costs much more energy than the others.

6.7.1 Discussion without Multiple-Instance Strategy

In this section, a topology is referred to as a ”dense topology” if most nodes have more than two neighbors in the topology, indicating that nodes are located in a dense area. The grid and random topologies are examples of dense topologies, where nodes have multiple paths to reduce traffic overhead. On the contrary, a topology is considered a ”sparse topology” if most nodes have only two neighbors, as nodes are located more loosely rather than bounded to a small area. All topologies except for the grid and random topologies belong to the sparse topology category, where most nodes only have two neighbors.

1. The multiple-DODAG strategy usually consumes less energy than the virtual root strategy, especially in dense topologies. For example, the difference between the two strategies in the energy consumption field in the grid and random topology is more than in the circular and linear topology. One reason the difference in sparse topologies is not apparent is that nodes have limited paths towards the next hop, leading to low consumption of analyzing and identifying the actual destination. In addition, loop detection and route switch are also the reasons. If a node's parent has a larger value than a threshold, the node will set the parent as invalid and change the route. Sensor nodes in the virtual root network can simply switch the parent as the network performs as a big DAG. Therefore, the virtual root strategy has good PDR performance because the parent switch design can maintain the data packet transmission task. Meanwhile, more packet transmission costs more energy. In addition, since nodes in dense topology have many neighbors, the route switch experiences more paths and produces more packets, which causes a larger energy consumption difference between the two strategies than the sparse topology.

2. Regarding the number of RPL control packets, the virtual root strategy outperforms the multiple-DODAG strategy in dense and crossing topologies. However, the virtual root strategy has slightly more packets in the linear and circular topologies than the multiple-DODAG strategy. As the virtual root strategy can simplify the establishment and maintenance of the RPL network with the big DAG, the strategy produces fewer RPL control packets than the multiple-DODAG strategy, especially in dense topologies. Meanwhile, the virtual root network does not reduce the RPL control packets in the sparse topology because the topology has specific paths. As for different kinds

of RPL control packets, the main difference lies in the number of DAO messages. The DAO message is used to transmit information upward along the link. Due to nodes in the multiple-DODAG strategy must maintain connections with two root nodes with different DAG information, the number of DAO messages is larger than in the virtual root strategy. Additionally, the virtual root strategy generates more DIO messages than the multiple-DODAG strategy, especially in dense topologies. There are two primary reasons. Firstly, the transmission of DIO messages is triggered by DIS messages, which the virtual root strategy produces more frequently than the multiple-DODAG strategy. Secondly, DIO messages are used to maintain the upward route in the RPL network. Although sensor nodes in the virtual root strategy do not need to maintain connections with two sinks, DIO messages are still necessary to join the DODAG, particularly during the sink switch period. Furthermore, a node with multiple neighbors may send additional DIO messages along different links with different sink information.

3. For the packet delivery ratio section, the virtual root strategy's performance is better than the multiple-DODAG strategy in most topologies. The main reason is that nodes in the virtual root strategy do not need to maintain the backup link. Therefore, nodes in the virtual root strategy can use more transmission capacity to deliver packets rather than maintain the RPL network connection. In addition, nodes in the virtual root strategy can switch the parent and route simply because nodes are located in a big DAG network. The route switch function can maintain the packet transmission task. However, for the circular topology, common sensor nodes in the virtual root network switch the destination frequently, leading to packet loss during the exchange period. As all nodes are common in the topology and belong to a big DAG with many hops, the loop detection function may cause bad performance. For example, if a sensor node detects a large rank value, it will choose the other neighbor. If the other neighbor also has a big rank value, the node must pause and wait for the new DAG version to be updated. This problem can be solved but need some time. Since the packet transmission speed is high and the virtual root network has a big DAG, the virtual root strategy cannot deliver data packets correctly during the root exchange period. Besides comparing the two strategies, the parent switch affects the PDR negatively in both

strategies. Therefore, the two strategies both have lower PDR than the multiple-instance strategy with high packet loss.

In conclusion, the virtual root strategy demonstrates a high packet delivery ratio and reduces the transmission pressure of RPL control packets in most scenarios, particularly in dense topologies. However, the multiple-DODAG strategy outperforms the virtual root strategy regarding energy consumption. In addition, in the circular topology, the multiple-DODAG strategy performs better than the virtual root strategy across all three aspects. The crossing topology presents a unique case where the central node experiences a high traffic overhead. Although both strategies indicate poor PDR performance in this topology, the virtual root strategy outperforms the multiple-DODAG strategy. Even the difference in energy consumption between the two strategies is not significant. The hops between sinks and distant sensor nodes along the central node are negative factors in the crossing topology. Moreover, as sensor nodes switch their destination, the exchange period introduces packet loss, negatively impacting the PDR.

Overall, the virtual root strategy is more suitable than the multiple-DODAG strategy in various scenarios, especially in dense topologies. Since the virtual root strategy costs more energy but the difference is insignificant, the strategy can be selected as long as the system is not a highly energy-sensitive system.

6.7.2 Multiple-Instance Discussion

The multiple-instance strategy is a special topic because of its complex implementation and complicated performance. Although the standard of RPL mentions the availability of the multiple-instance strategy, some modifications are necessary to allow the Contiki-NG to support some functions in the multiple-instance network. More specifically, the source code needs to be modified to satisfy the UDP packet transmission task in this project. According to experiments, it is obvious that the multiple-instance strategy produces much more RPL control packets than the other two because each node in the multiple-instance strategy must receive and deal with information from two independent instances. In addition, sensor nodes also send many RPL control packets toward the root to maintain the DODAG connection in two instances. Besides the number of packets, nodes in the multiple-instance strategy cost a lot of energy because nodes must process many packets. In addition, the heavy RPL control packet overhead occupies traffic resources, leading to a slightly

lower delivery ratio of the UDP packet transmission under low packet loss conditions.

However, the multiple-instance strategy has good PDR performance with high packet loss in each topology. In contrast, the number of RPL control packets and energy consumption is much more than the other two strategies. The main reason is that the multiple-instance implementation applies the preferred instance design, leading to little packet loss during the destination switch period. This design provides high PDR performance, but it reduces the flexibility in the multiple-instance network. Although the multiple-instance strategy demonstrates good performance in scenarios with high packet loss, it underperforms compared to the other two strategies regarding PDR with low data transmission speed, as observed in some supplementary experiments. Some additional experiments were designed to transmit one UDP data packet per three seconds. In these extra experiments, the multiple-instance strategy consistently exhibits a lower packet delivery ratio than the other two under all packet loss conditions.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this project, the main idea is to compare the different RPL strategies in different situations and conclude some regular patterns as a reference while implementing the RPL network. There are three strategies with multiple sinks designed in this project. Multi-sink mechanism of RPL is mainly applied to a large-scale network and provides redundancy and backup. However, three different strategies have different features. The multiple-DODAG strategy is the default strategy with the most straightforward implementation process. The virtual root strategy needs a unique configuration but simplifies the RPL network. The multiple instances strategy is the most complex strategy with much modification for the default system. Contiki-NG and Cooja are applied to build simulation experiments to compare the performance of three strategies in multiple scenarios. As for evaluating the performance in different scenarios, the project implements five topologies: grid, random, linear, circular, and crossing topology. Among the five topologies, grid and random topology are called dense topology because nodes are located in a dense area, and most nodes have more than two neighbors. On the contrary, the other topologies are called sparse topologies since nodes are not clustered in a certain area, and most nodes have two neighbors. The crossing topology is unique because most nodes have two neighbors at most, but the central node has four neighbors. After determining the topologies, the evaluation metrics are determined in the same section. There are three evaluation areas: the number of RPL control packets, PDR, and energy consumption. Then the implementation of the node and topology are indicated in detail. The method of collecting data from the three evaluation metrics is determined before introducing the results.

By analyzing the result, the multiple-instance strategy performs worst among all three strategies except for some specific scenarios. The multiple-instance strategy has better PDR performance under the high packet loss condition. Apart from the multiple-instance strategy, the virtual root strategy performs better in the number of packets and PDR field but worse in energy consumption than the multiple-DODAG strategy in most scenarios, especially in dense topologies. While applying the circular topology, the virtual root strategy performs worse than the multiple-DODAG strategy across all three aspects.

7.2 Limitations

The first limitation of this study is that the experiments were conducted on a simulation platform. Although the configuration of the Contiki-NG can be simulated, practical experiments are preferred for IoT networks. For example, practical experiments can help identify issues during practical use and allow adjustments to ensure the system works well in real-world scenarios.

The second limitation of this project is the number of topologies used in the experiments. Although five topologies are enough to conclude several regular patterns, they can only contain some scenarios. Therefore, the conclusion and rules are more solid with additional experiments and topologies. However, due to limitations in the length of the thesis, only five topologies were used in this project.

The third limitation is related to implementing the multiple-instance strategy in the experiments. While this strategy was implemented to evaluate the system's performance, there are some remaining flaws. For instance, it is unclear how the system would perform if the root were allowed to join another root's instance. Additionally, the preferred instance function negatively affects the flexibility of the multiple-instance network. As this project's primary goal is not to provide a detailed implementation and deployment of the multiple-instance strategy, the multiple-instance network can be applied to finish the project, but the implementation is limited.

7.3 Future Work

Based on the work shown in this project, there are several areas for potential future improvement.

Firstly, the experiments could be expanded to include more scenarios, such as larger network scales, diverse topologies, increased data packet numbers,

and experiments conducted on practical devices. Due to limitations in the laptop's performance used in this project, certain ideas, like extensive network scales and more data packets, cannot be completed in this project. Therefore, improving the experiments makes the conclusions and rules more accurate, enabling more efficient comparisons between different strategies.

Secondly, the implementation improvement is limited since deploying the virtual root strategy and the multiple-DODAG is straightforward. However, one potential modification involves enhancing the virtual root strategy's synchronization method among practical sinks. Furthermore, the experiment with different mote types and MAC layers can be applied to evaluate the performance in different scenarios.

Lastly, further research can be conducted on the multiple-instance strategy. In this project, sensor nodes in the multiple-instance strategy were assigned the preferred instance, resulting in good PDR performance with high packet loss. However, this design limits the flexibility of the RPL network. Unlike sensor nodes in the other two strategies, nodes in multiple-instance networks are unable to switch destinations frequently. In addition, further research can examine how the network would perform if the root were permitted to join another instance. Different implementations of the multiple-instance strategy may lead to different results.

References

[1] P. Gralla, *How the Internet works*. Que Publishing, 1998. [Page 1.]

[2] J. Mo, R. La, V. Anantharam, and J. Walrand, "Analysis and comparison of tcp reno and vegas," in *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, vol. 3, 1999. doi: 10.1109/INFCOM.1999.752178 pp. 1556–1563 vol.3. [Page 1.]

[3] Z. Bi, L. D. Xu, and C. Wang, "Internet of things for enterprise systems of modern manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1537–1546, 2014. doi: 10.1109/TII.2014.2300338 [Page 1.]

[4] W. He, G. Yan, and L. D. Xu, "Developing vehicular data cloud services in the iot environment," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1587–1595, 2014. doi: 10.1109/TII.2014.2299233 [Page 1.]

[5] P. O. Kamgueu, E. Nataf, and T. D. Ndie, "Survey on rpl enhancements: A focus on topology, security and mobility," *Computer Communications*, vol. 120, pp. 10–21, 2018. [Pages 1 and 2.]

[6] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information systems frontiers*, vol. 17, no. 2, pp. 243–259, 2015. [Page 1.]

[7] P. Sethi and S. R. Sarangi, "Internet of things: architectures, protocols, and applications," *Journal of Electrical and Computer Engineering*, vol. 2017, 2017. [Page 1.]

[8] S. Marksteiner, V. J. Exposito Jimenez, H. Valiant, and H. Zeiner, "An overview of wireless iot protocol security in the smart home domain,"

in *2017 Internet of Things Business Models, Users, and Networks*, 2017. doi: 10.1109/CTTE.2017.8260940 pp. 1–8. [Page 1.]

[9] R. Bhalerao, S. S. Subramanian, and J. Pasquale, "An analysis and improvement of congestion control in the coap internet-of-things protocol," in *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2016. doi: 10.1109/CCNC.2016.7444906 pp. 889–894. [Page 1.]

[10] 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, no. 4, pp. 2502–2525, 2017. doi: 10.1109/COMST.2017.2751617 [Page 1.]

[11] M. Banh, H. Mac, N. Nguyen, K.-H. Phung, N. H. Thanh, and K. Steenhaut, "Performance evaluation of multiple rpl routing tree instances for internet of things applications," in *2015 International Conference on Advanced Technologies for Communications (ATC)*, 2015. doi: 10.1109/ATC.2015.7388321 pp. 206–211. [Pages 1, 2, 20, and 28.]

[12] R. Alexander, A. Brandt, J. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: https://www.rfc-editor.org/info/rfc6550 [Pages 2, 11, 12, 13, 14, 17, 19, 24, 25, and 28.]

[13] 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, no. 1, pp. 1–23, 2014. [Pages 2, 18, 20, and 26.]

[14] 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. doi: 10.1109/DCOSS52077.2021.00015 pp. 10–17. [Pages 2 and 20.]

[15] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes, "The contiki-ng open source operating system for next generation IoT devices," *SoftwareX*, vol. 18, p. 101089, 2022. doi: https://doi.org/10.1016/j.softx.2022.101089 [Pages 3, 18, 39, and 41.]

[16] R. V. Labaree, "Research Guides: Organizing Your Social Sciences Research Paper: Quantitative Methods." [Online]. Available: https://libguides.usc.edu/writingguide/quantitative [Page 3.]

[17] "Correcting the IoT History," Mar. 2016, section: 4th Wave. [Online]. Available: http://www.chetansharma.com/correcting-the-iot-history/ [Page 7.]

[18] K. Ashton *et al.*, "That 'internet of things' thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009. [Page 7.]

[19] K. Rose, S. Eldridge, and L. Chapin, "The internet of things: An overview," *The internet society (ISOC)*, vol. 80, pp. 1–50, 2015. [Pages 7 and 8.]

[20] "IoT connected devices worldwide 2019-2030." [Online]. Available: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ [Page 7.]

[21] "Where and how to capture accelerating IoT value | McKinsey." [Online]. Available: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/iot-value-set-to-accelerate-through-2030-where-and-how-to-capture-it [Page 7.]

[22] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of iot: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things journal*, vol. 1, no. 4, pp. 349–359, 2014. [Page 7.]

[23] T. T. Thakur, A. Naik, S. Vatari, and M. Gogate, "Real time traffic management using internet of things," in *2016 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, 2016, pp. 1950–1953. [Page 7.]

[24] T. Malche and P. Maheshwary, "Internet of things (iot) for building smart home system," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE, 2017, pp. 65–70. [Page 7.]

[25] H. Ahmadi, G. Arji, L. Shahmoradi, R. Safdari, M. Nilashi, and M. Alizadeh, "The application of internet of things in healthcare: a

systematic literature review and classification," *Universal Access in the Information Society*, vol. 18, pp. 837–869, 2019. [Page 8.]

[26] H. Tschofenig, J. Arkko, D. Thaler, and D. R. McPherson, "Architectural Considerations in Smart Object Networking," Internet Engineering Task Force, Request for Comments RFC 7452, Mar. 2015, num Pages: 24. [Online]. Available: https://datatracker.ietf.org/doc/rfc7452 [Pages ix, 8, 9, and 10.]

[27] V. Tanyingyong, R. Olsson, M. Hidell, and P. Sjödin, "Scalable iot sensing systems with dynamic sinks," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7211–7227, 2022. doi: 10.1109/JIOT.2021.3094275 [Pages 12, 25, and 28.]

[28] 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*, vol. 26, no. 3, pp. 952–968, 2021. [Page 12.]

[29] P. Thubert, "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)," Internet Engineering Task Force, Request for Comments RFC 6552, Mar. 2012, num Pages: 14. [Online]. Available: https://datatracker.ietf.org/doc/rfc6552 [Page 14.]

[30] B. Ghaleb, A. Y. Al-Dubai, E. Ekonomou, A. Alsarhan, Y. Nasser, L. M. Mackenzie, and A. Boukerche, "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, no. 2, pp. 1607–1635, 2018. [Pages 14 and 19.]

[31] O. Gnawali and P. Levis, "The Minimum Rank with Hysteresis Objective Function," Internet Engineering Task Force, Request for Comments RFC 6719, Sep. 2012, num Pages: 13. [Online]. Available: https://datatracker.ietf.org/doc/rfc6719 [Page 14.]

[32] A. Dunkels, "uip-a free small tcp/ip stack," *UIP*, vol. 1, pp. 1–17, 2002. [Page 18.]

[33] N. Tsiftes, J. Eriksson, and A. Dunkels, "Low-power wireless ipv6 routing with contikirpl," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, ser. IPSN '10. New York, NY, USA: Association

for Computing Machinery, 2010. doi: 10.1145/1791212.1791277. ISBN 9781605589886 p. 406–407. [Online]. Available: https://doi.org/10.1145/1791212.1791277 [Page 18.]

[34] P. Karkazis, H. C. Leligou, L. Sarakis, T. Zahariadis, P. Trakadas, T. H. Velivassaki, and C. Capsalis, "Design of primary and composite routing metrics for rpl-compliant wireless sensor networks," in *2012 international conference on telecommunications and multimedia (TEMU)*. IEEE, 2012, pp. 13–18. [Page 19.]

[35] C. Kiraly, T. Istomin, O. Iova, and G. P. Picco, "D-rpl: Overcoming memory limitations in rpl point-to-multipoint routing," in *2015 IEEE 40th Conference on Local Computer Networks (LCN)*. IEEE, 2015, pp. 157–160. [Page 19.]

[36] 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)*. IEEE, 2015, pp. 265–273. [Page 20.]

[37] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon, "Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks," in *Information Networking. Convergence in Broadband and Mobile Networking: International Conference, ICOIN 2005, Jeju Island, Korea, January 31-February 2, 2005. Proceedings*. Springer, 2005, pp. 264–274. [Page 20.]

[38] J. Eriksson, N. Finne, N. Tsiftes, S. Duquennoy, and T. Voigt, "Scaling rpl to dense and large networks with constrained memory," in *International Conference on Embedded Wireless Systems and Networks (EWSN 2018), Madrid, Spain, 14-16 Feb 2018*, 2018. [Page 21.]

[39] P. Ciciriello, L. Mottola, and G. P. Picco, "Efficient routing from multiple sources to multiple sinks in wireless sensor networks," in *Wireless Sensor Networks*, K. Langendoen and T. Voigt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. ISBN 978-3-540-69830-2 pp. 34–50. [Page 23.]

[40] Z. Vincze, R. Vida, and A. Vidacs, "Deploying multiple sinks in multi-hop wireless sensor networks," in *IEEE International Conference on Pervasive Services*, 2007. doi: 10.1109/PERSER.2007.4283889 pp. 55–63. [Page 23.]

[41] Q.-D. Nguyen, J. Montavont, N. Montavont, and T. Noël, "Rpl border router redundancy in the internet of things," in *Ad-hoc, Mobile, and Wireless Networks*, N. Mitton, V. Loscri, and A. Mouradian, Eds. Cham: Springer International Publishing, 2016. ISBN 978-3-319-40509-4 pp. 202–214. [Page 24.]

[42] M. Abdullah, I. Alsukayti, and M. Alreshoodi, "On the Need for Efficient Load Balancing in Large-scale RPL Networks with Multi-Sink Topologies," *International Journal of Computer Science & Network Security*, vol. 21, no. 3, pp. 212–218, 2021. doi: 10.22937/IJCSNS.2021.21.3.29 Publisher: International Journal of Computer Science & Network Security. [Online]. Available: https://koreascience.kr/article/JAKO202121055613992.page [Page 24.]

[43] E. Fazeldehkordi, I. S. Amiri, and O. A. Akanbi, "Chapter 2 - literature review," in *A Study of Black Hole Attack Solutions*, E. Fazeldehkordi, I. S. Amiri, and O. A. Akanbi, Eds. Syngress, 2016, pp. 7–57. ISBN 978-0-12-805367-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128053676000028 [Page 37.]

[44] "RPL — Contiki-NG documentation." [Online]. Available: https://docs.contiki-ng.org/en/develop/doc/programming/RPL.html?highlight=rpl%20classic [Page 39.]

[45] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," in *Proceedings of the 4th Workshop on Embedded Networked Sensors*, ser. EmNets '07. New York, NY, USA: Association for Computing Machinery, 2007. doi: 10.1145/1278972.1278979. ISBN 9781595936943 p. 28–32. [Online]. Available: https://doi.org/10.1145/1278972.1278979 [Page 41.]