

**PERFORMANCE EVALUATION OF ROLL/RPL PROTOCOL  
IN DELAY TOLERANT NETWORKS (DTN)**

**By**

**Elias Sfeir**

**A Thesis Submitted**

**In partial Fulfillment of the Requirements for the Degree of Master of  
Science in Computer Information System**

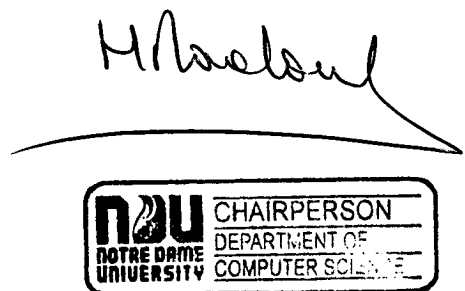
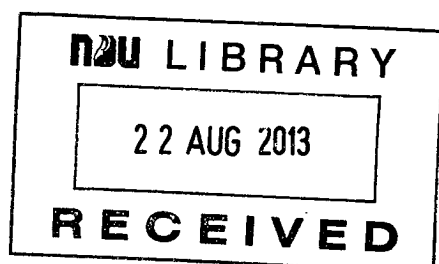
**Department of Computer Science**

**Faculty of Natural and Applied Sciences**

**Notre Dame University – Louaize**

**Zouk Mosbeh, Lebanon**

**Fall 2012**

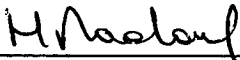


Performance Evaluation of ROLL/RPL Protocol in Delay Tolerant Networks (DTNs)

By

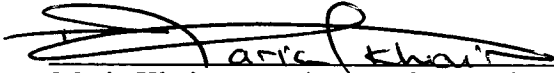
Elias Sfeir

Approved by:



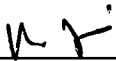
---

Hoda Maalouf: Associate Professor of Computer Science  
Advisor.



---

Marie Khair: Associate Professor of Computer Science  
Member of Committee.



---

Khaldoun El-Khaldi: Assistant Professor of Computer Science  
Member of Committee.

---

Date of Thesis Defense: November 8, 2012

## Abstract

Delay-Tolerant Networks (DTNs) are a type of emerging networks characterized by very long delay paths, frequent network partitions and high loss rates. The nodes are sparsely deployed and intermittently connected. The presenting network is used in a variety of environments including military, disaster situations and for long distance communications. Although it is used in critical sectors, the communication support lacks the high end technology usually found in important networks. The nodes participating in this network do not have stable end-to-end connections. They can communicate to each other only occasionally, through the opportunistically available links. Thus, routing becomes one of the most challenging problems. On the other hand, the Ripple Routing protocol (RPL) was introduced by a working group called Routing Over Low and Lossy Networks (ROLL) and aimed to be used as a routing protocol over Low and Lossy Networks (LLN).

This thesis aims to find a suitable protocol that can be applied to Delay tolerant Networks in order to improve and optimize the routing and to lower the delay and loss of packets between the nodes. Since the ROLL/RPL protocol is specialized for the Low power and Lossy Networks; therefore it could include Delay Tolerant Networks as well.

However, the ROLL/RPL protocol was mainly designed for dense wireless networks. On the other hand, a mobile DTN does not have much in common with a dense “always on” network. Therefore, we decided in this thesis to investigate the applicability of the ROLL/RPL on a DTN, and to calculate its efficiency depending on the network density. This thesis shall provide recommendations on how this protocol could be upgraded to suit sparse Low and Lossy Networks (LLNs).

## **Acknowledgements**

This thesis is to show my accomplishments throughout my long journey in this university, and how it all ended up to be a fruit of knowledge.

I would like to thank my parents for their endless support and encouragement through my college years; they were my backbone during the tough times and the good times.

I would like to express my gratitude also to my advisor Dr. Hoda Maalouf for her guidance and consistent support over this project that helped shape my skills and guide for the completion of this thesis.

## **Dedication**

Dedicated to my parents for they were my support and guidance throughout my life and university years.

## Table of Contents

Abstract .....	ii
Acknowledgements .....	iii
Dedication.....	iv
Table of Contents.....	v
List of Figures.....	viii
Chapter 1 Introduction and Problem Definition .....	1
1.1 Introduction.....	1
1.2 Problem Definition .....	1
1.3 Research Objectives .....	2
1.4 Approach.....	2
1.5 Thesis Organization.....	3
Chapter 2 Delay Tolerant Networks Overview.....	4
2.1 Introduction.....	4
2.2 Delay Tolerant Network Routing Protocols.....	8
2.2.1 Forwarding Protocols.....	8
2.2.2 Flooding Protocols.....	9
2.2.3 Hybrid based Protocols .....	10
2.3 Delay Tolerant Network Applications.....	10
2.4 Delay Tolerant Network Challenges .....	12
2.4.1 Power Consumption.....	12
2.4.2 Communication .....	13
2.4.3 Fault Tolerance and Adaptability .....	13
2.4.4 Scalability and Flexibility .....	13
2.5 Green Wireless Networks .....	14
2.6 Conclusion .....	14
Chapter 3 ROLL/RPL Protocol .....	15
3.1 Introduction.....	15
3.2 RPL Routing Requirements .....	15
3.3 RPL Protocol Definition .....	16
3.3.1 Metrics/Constraints in RPL.....	17

3.3.2 The Objective Function .....	19
3.3.3 RPL DODAG Architecture.....	21
3.4 RPL DODAG Messages .....	24
3.4.1 DIO.....	25
3.4.2 DAO.....	27
3.4.3 DIS .....	27
3.5 RPL DODAG Building Procedure .....	28
3.5.1 RPL Trickle Timer Process .....	28
3.5.2 DODAG Building Procedure.....	29
3.5.3 Loop Avoidance and Detection .....	32
3.5.4 Repair Mechanism .....	33
3.6 Previous Related Work .....	36
3.6.1 Performance Evaluation Study of RPL for LLN.....	36
3.6.2 Performance Analysis of the RPL.....	37
3.6.3 Simulation and Performance Evaluation of DAG Construction with RPL .....	39
3.6.4 A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL) .....	40
3.6.5 The P2P-RPL Routing Protocol for IPv6 Sensor Networks: Testbed Experiments.....	41
3.7 Conclusion.....	42
Chapter 4 Proposed ROLL/RPL Protocol for DTN and Simulation Results.....	43
4.1 Introduction .....	43
4.2 Simulation Building Process .....	43
4.2.1 Generate Nodes Functionalities .....	45
4.2.2 Run Simulation Functionalities.....	46
4.2.3 Energy Calculation.....	46
4.3 Assumptions.....	47
4.4 Network Lifetime and Power Consumption.....	47
4.4.1 Introduction .....	47
4.4.2 Calculation Procedure .....	48
4.4.3 Simulation Results .....	48
4.5 Network Success Rate.....	51
4.5.1 Introduction .....	51

4.5.2 Simulation Results .....	52
4.5.3 Tradeoffs Analysis .....	52
4.6 Conclusion .....	54
Chapter 5 Conclusions .....	55
5.1 Introduction.....	55
5.2 Main Contributions .....	55
5.3 Suggested Future Work.....	56
Appendix A Simulation software code snippets.....	57
Bibliography .....	63



## List of Figures

Figure 2.1- Store and Forward Protocol .....	4
Figure 2.2 - OSI Layers VS DTN Layers .....	5
Figure 2.3 - DTN Routing Protocols .....	8
Figure 3.1 - Different Objective Functions.....	21
Figure 3.2 - DODAG Multiple Instances.....	24
Figure 3.3 - DIO Process .....	30
Figure 3.4 - DODAG Building Process.....	32
Figure 3.5 - Loop Repair Mechanism.....	35
Figure 3.6 - Network Topology used for the Simulation.....	37
Figure 3.7 - Network Topologies with 20 and 100 nodes.....	38
Figure 3.8 - RPL Communication .....	42
Figure 4.1 - DTN/RPL Initial Screen .....	43
Figure 4.2 - Network Average Power Consumption .....	49
Figure 4.3 - Average Lifetime for Different Network Densities.....	50
Figure 4.4 - Percentage of Power Saving using the Sleep Mode .....	51
Figure 4.5 - Network Success Rate .....	52
Figure 4.6 - $p$ and $S$ as function of the network density .....	53

## List of Acronyms

DTN	Delay Tolerant Network
LLN	Low Power and Lossy Networks
DODAG	Destination Oriented Directed Acyclic Graph
RPL	Ripple Routing Protocol
ROLL	Routing Over Low and Lossy networks
IETF	Internet Engineering Task Force
LBR	LowPAN Border Router
DIS	DODAG Information Solicitation
DIO	DODAG Information Object
DAO	DODAG Destination Advertisement Object
ETX	Expected Transmission Count

# Chapter 1

## Introduction and Problem Definition

### 1.1 Introduction

Delay Tolerant Networks (DTN), also known as intermittently connected mobile networks; present a challenge nowadays, for they play an important role in case of critical situations such as disasters [1], space communications [2] and underwater communications [3]. Communications between the nodes rely on the store-carry and forward protocol in order to deliver packets from the source to the destination. Thus, the connectivity of the whole network relies on the nodes where the transmission range is intersected, and the waiting node will store the message until an opportunity of sending it will be available.

In this thesis, we will introduce the ROLL/RPL protocol [4] concept and check its suitability as a routing protocol for Delay Tolerant Networks. The ROLL/RPL protocol is defined to be an IPv6 routing protocol specified for Low Power and Lossy Networks (LLN) that defines a way to build a graph known as Destination Oriented Directed Acyclic Graph (DODAG) that will be implemented on top of a network in order to optimize its routing process. This graph is based on a collection of metrics and constraints that will be constructing the routing mechanism along with an objective function extracted from several parameters such as the link quality, shortest path, and others.

### 1.2 Problem Definition

The basic architecture of a delay tolerant network consists of communication nodes connected to each other wirelessly. Those nodes may not be connected all time which will include long periods of time with no connectivity between them. Thus, routing is the main issue addressed in these types of networks. The main workflow of the routing mechanism in DTNs is to find and choose the best available route in order to send the incoming data packets. In standard networks, the cost of the link present between two nodes is easily

calculated in order to define the best path, while in the DTN environment, the same cost is not easily defined for the connectivity between the nodes is variable depending on many factors. Those factors consider the power consumption of every node, the presenting link quality, the delay and the loss of the packets along the delivery process [5].

### **1.3 Research Objectives**

This research aims to achieve a cost-effective connectivity and a near optimized solution for the routing problem found in DTN, through the introduction of the ROLL/RPL protocol.

Since this proposed protocol usually addresses dense networks that present low power and lossy architecture (LLN), we shall first investigate in this thesis its applicability to DTNs which are LLNs but are usually not dense.

In fact, DTNs suffer from disconnectivity problem. Hence, our main objective would be to find the minimum acceptable node density where ROLL/RPL protocol could be efficiently implemented in a DTN.

### **1.4 Approach**

In this thesis, we are using computer based simulations to solve our problem and to achieve the desired thesis objectives.

Our simulation program is written in C# where different scenarios of wireless networks are implemented.

Once the simulations are completed, recommendations shall be given regarding the usability of ROLL/RPL in DTNs.

## **1.5 Thesis Organization**

This thesis is organized into five chapters. In addition to the current one, Chapter 2 will provide a general idea concerning the Delay Tolerant Network definition and characteristics. Chapter 3 gives an overview of the ROLL/RPL protocol architecture and applications. Chapter 4 focuses on mechanism defended in this thesis that will include the protocol application on the DTN architecture. It also gives some numerical and analytical results using C# technology. Finally, Chapter 5 includes summaries of the main results and suggests some potential future work.

## Chapter 2

### Delay Tolerant Networks Overview

#### 2.1 Introduction

In this chapter, we will have an overview of delay tolerant networks. It is a mobile network environment that consists of the absence of a reliable communication and end to end connectivity. Thus, high latency is an imposed subject in this area, and it may reach sometimes hours and maybe days. The routing process is an important issue to be discussed in such a case where many routing protocols are applied in order to optimize the whole process.

The store and forward protocol (See Figure 2.1) is implemented in the Delay Tolerant Networks (DTNs), which consists of storing the incoming message in the receiving node and forwarding the message from the same source when it comes to a failure in the delivery of the message. The DTN routers are designed in such a way to include persistent storage devices for several reasons, consisting of:

- i. The communication link between the routers is not stable all the time, and may not be even usable all the time for transferring messages.
- ii. Nodes in the DTN are different, so there are differences related to technical issues, such as speed, reliability, etc.
- iii. Some of the messages that are being sent between the nodes of the network may need to be retransmitted due to occurring error during transmission.

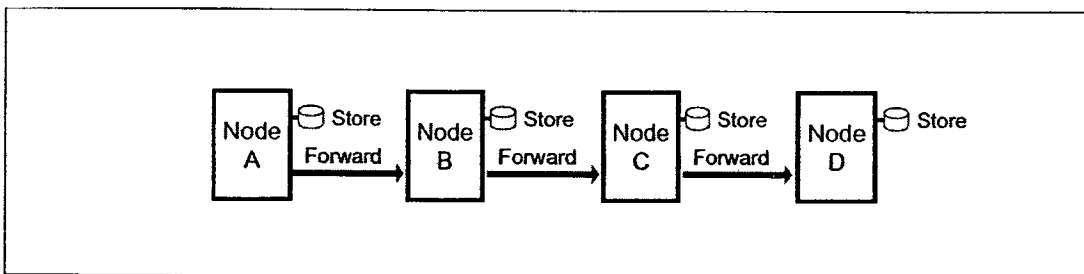


Figure 2.1- Store and Forward Protocol

The messages are controlled by protocol layers as shown in Figure 2.2 and the model is known as the Open Systems Interconnection (OSI) model.

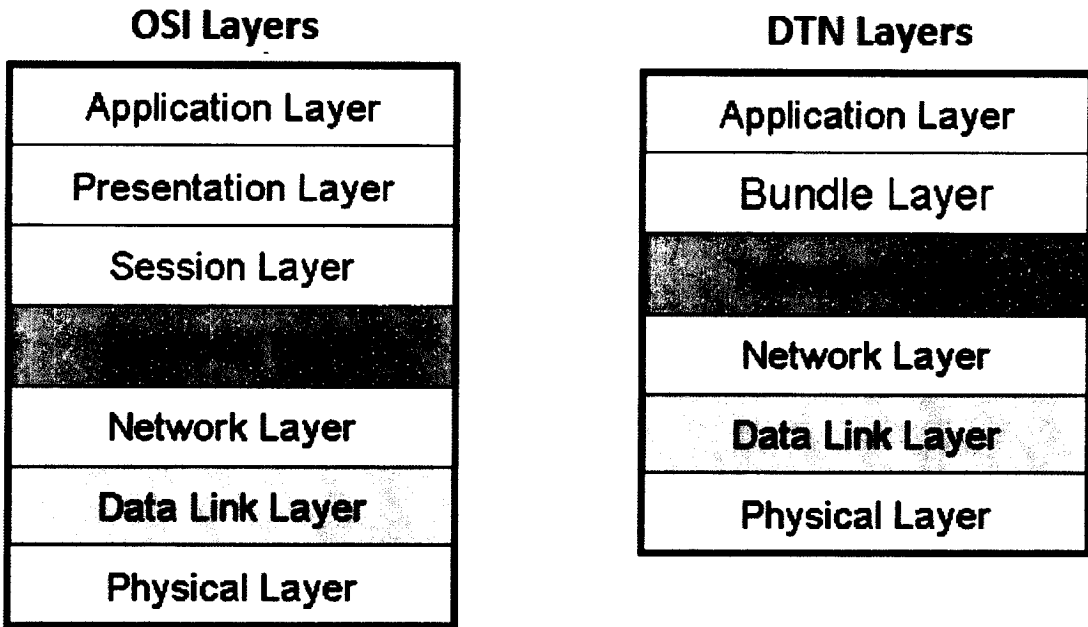


Figure 2.2 - OSI Layers VS DTN Layers

- The application layer produces the messages to be used in the network.
- The transport layer handles the end to end segmentation and the reassembly of the messages. The final result of the messages will include error control and flow control.
- The network layer controls the routing of the messages from the source to destination. Also, it will manage the fragmentation and the reassembly of the messages.
- The link layer handles the transmission and the reception of the link to link messages.

- The physical layer handles the link to link transmission and reception of the bit streams of the network.

The store and forward architecture in DTNs is implemented by creating an overlay on top of the given network structure, known as the “bundle layer” [2]. This overlay may exist anywhere between the two transport and application layers. This single bundle layer protocol is used across all networks that implement the DTN protocol. The bundles are also known as messages that are stored and forwarded between the nodes by the bundle layer.

The bundles are composed of the following:

- i. The user data source application.
- ii. The control information, which includes the processing, storing, disposing and the handling of the user data.
- iii. The bundle header.

Bundles extend the structure of the encapsulation of data objects implemented in other protocols such as the internet protocols. In other words, the bundle protocol can break the messages from the source node into fragments and then reassembles them at the destination node. The bundle layers do the communication between themselves and not through another layer. Retransmission is supported also at the two layers, the bundle and the transport layers. Custody transfers are managed and controlled between the bundle layers transmission phase. Thus, once a request is established between two nodes, a custody transfer is sent with a timer. If the acknowledgment was not received by the source, this node will retransmit the message.

The bundle layer presents the following classes of service [5]:

- a. Custody Transfer. This service provides the retransmission mechanism between the nodes through acknowledgment messages.
- b. Return Receipt. This service is a confirmation sent by the destination to the source to inform that the bundle is received.



- c. Custody Transfer Notification. This service includes a notification message that is sent to the source informing it that the destination node has accepted the custody request.
- d. Bundle Forwarding Notification. This service includes a notification message that is sent to the source informing it that the destination node has forwarded the bundle message.
- e. Priority of Delivery. This service informs the priority of the delivered message, such information includes if it is normal, bulk or expedited.
- f. Authentication. This service provides a kind of digital signature to the messages, usually used to check the sender's identity.

Flexibility and scalability are advantages of using the introduced approach. For example, one may easily link already existing TCP/IP networks with networks that will appear in the future for deep space communication purposes and will probably use their own transport, network and physical layers.

As stated in [6] Delay Tolerant Networks allow data transmission in challenging environments where the existing network architecture is known for its frequent and long duration partitioning and it may have no end-to-end connectivity between the source and the destination. The goal of implementing of such network structures is to increase the likelihood of finding a routing path in the network with extremely limited information. Real world deployment of the DTN routing protocols are often expensive and this is the reason behind simulation of such kinds of routing protocols, and it is highly dependent on the level of realism in the simulators.

## 2.2 Delay Tolerant Network Routing Protocols

Before going further in this thesis, we will define next the different types of routing protocols used for the DTN architecture as shown in Figure 2.3 [7]:

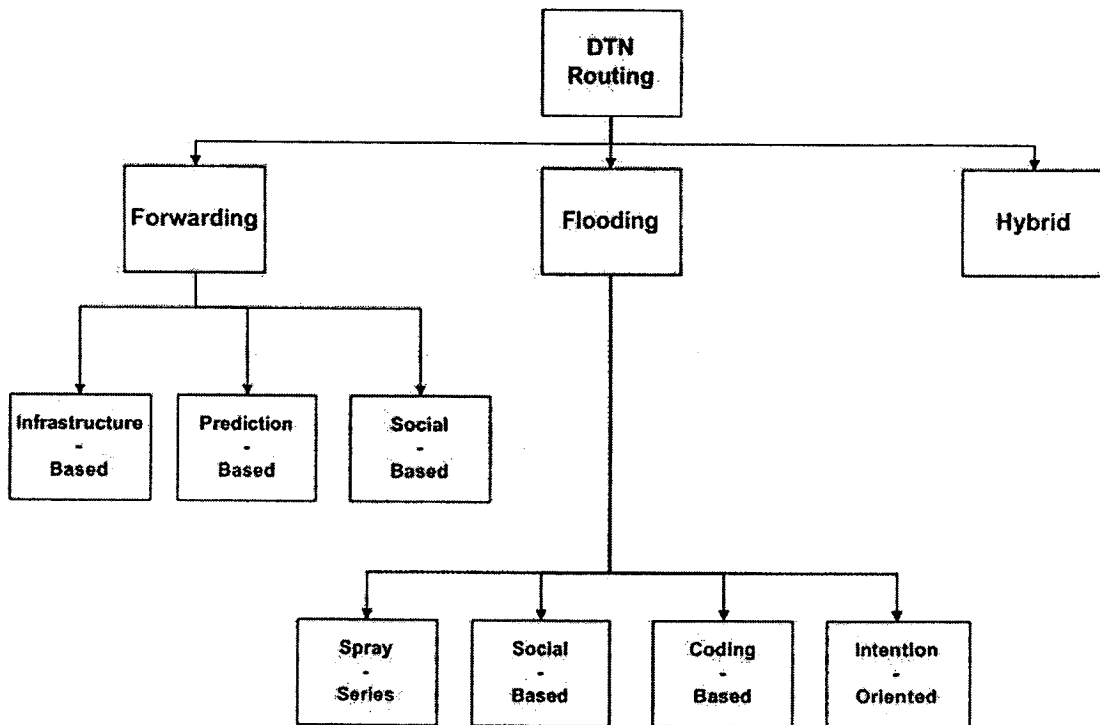


Figure 2.3 - DTN Routing Protocols

### 2.2.1 Forwarding Protocols

In the forwarding protocol, during the transmission phase in the network, only one copy is kept of the message. In such a protocol, every relay node should choose the next hop or the destination of the message being sent, and according to movement patterns, these forwarding protocols are divided into three sub-sections [6]:

- a) Infrastructure-based

Infrastructure based routing protocol can compensate the rugged environment of a DTN by deploying fixed mobile agents. Those agents will act as store-carry-forward paradigms that will carry the message from one node to another and fill in the gaps in the network.

b) Prediction-based

Prediction based routing protocol is based on the calculation and prediction of future knowledge such as message delivery probability, which is supported by history information. In that way, the next hop is predicted in order to improve the end-to-end delivery probability.

c) Social-based

Social based routing protocol depends on the common social relationships and people's daily life communication. Degrees, levels and priorities of social classes are used in order to determine the communication between the nodes inside of a network.

### **2.2.2 Flooding Protocols**

In the flooding protocol, duplications of a message are used in the network and by that the relay nodes will need the store-and-forward message mechanism. This protocol will reduce the delays of the message transmission and enhance the delivery method at the cost of huge network resource consumption. Some of the flooding based protocols are presented:

a) Spray-series

This strategy is based on a mechanism that "sprays" a few message copies in the network and then studies the routing paths of every sprayed message independently towards its destination. The generated copies of the message are studied and chosen in order to ensure that the transmission is controlled.

b) Social-based

This strategy is similar to the social-based forward routing protocol, but additionally this strategy includes the flooding mechanism in order to increase the delivery rate.

### c) Coding-based

This strategy takes into consideration the fragmentation of the message to be sent and then flooded in the network. The nodes will encode them into new packets which will be forwarded again to other nodes. In other words, the messages will be replicated throughout the whole network to assure that the messages will be received by their destination.

### d) Intention-oriented

This mechanism study the way messages are replicated in the system, which is why it is called intention routing. Based on the intention of the network, performance objectives are achieved such as average delivery rate, energy and bandwidth consumption.

## 2.2.3 Hybrid based Protocols

Some tried to achieve a balance between the two stated protocols, forwarding and flooding, which showed an increase of the delivery rate and by that reducing resource consumption. In [8], the authors present one of the best representatives of these strategies called Distributed Max-Contribution (DMC), which periodically scan for network topology changes and builds temporary disjoint groups of connected nodes by diameter-constrained algorithm.

## 2.3 Delay Tolerant Network Applications

A DTN could be used for the following applications[9] :

- a) Inter-Planet Satellite Communication Network. This is similar to the internet service in space and is characterized by:
  - Extremely long propagation delay.
  - Low transmission reliability (positioning inaccuracy and limited visibility).

- Low and asymmetric data rate.

b) Military Battlefield Network.

In this case, we have no consistent network infrastructure and frequent disruptions and hence such a network is characterized by:

- High intermittent connectivity.
- Mobility, destruction, noise, attack, interference.
- Low transmission reliability: positioning inaccuracy and limited visibility.
- Low, asymmetric data rate

c) Energy Constrained / Sparse Wireless Sensor Network

These networks are mainly used for coordinating the activities of multiple sensors to monitor science and hazard events (space, terrestrial and airborne environments) and are characterized by:

- Intermittent connectivity.
- Power saving, sparse deployment.
- Low and asymmetric data rate.

d) Village Area Networks:

These networks provide asynchronous digital connectivity in rural areas by transportation systems and are characterized by:

- Intermittent connectivity.
- Mobility and sparse deployment.
- High propagation delay.
- Asymmetric and heterogeneous data rate.

e) Underwater Acoustic Networks:

These networks are used for environmental monitoring, disaster prevention and assisted navigation and are characterized by:

- Intermittent connectivity.
- Mobility and sparse deployment.
- High propagation delay (1500 m/s).
- Transmission reliability, positioning inaccuracy and high attenuation.
- High transmission cost.
- Low asymmetric data rate.

f) Sparse Mobile Ad Hoc Networks:

These networks provide intermittent autonomous (opportunistic) communications among nodes and are characterized by:

- Intermittent connectivity.
  - Mobility and sparse deployment.
  - Large end to end delay.

## **2.4 Delay Tolerant Network Challenges**

DTNs face several challenges; some of them are unique due to their network infrastructure. These include: power consumption, communication, scalability and flexibility, fault tolerance and adaptability. Details are given in the next sections[10].

### **2.4.1 Power Consumption**

As stated in [11] the DTN nodes have limited power source. Routing strategies are by their nature energy consumers. Sending, transmitting, receiving, storing, listening and computing tasks continuously are the main reasons behind power consumption. There are

many strategies for the optimization of power saving in such scenarios. DTN nodes consume energy in both idle and active modes, leading to a huge energy drain. Some considered solutions are to have a power management solution in order to preserve the nodes energy source, such as renewable energy and sleep mode mechanism. Such power management strategies endure difficulties in the tradeoffs between performance and energy consumption.

#### **2.4.2 Communication**

The connection between nodes is a very important task in any network. The transmission techniques are based on the communication established between every node in the network. The main cause of the energy consumption in DTN is the transmission of messages between the nodes in order to build the structure of the network.

#### **2.4.3 Fault Tolerance and Adaptability**

Adaptability is a very important subject in the DTN region. Failure in the network should be handled in an appropriate manner. Such cases could include malfunctions in the link between the nodes which has a high probability in the studied environment. Other cases include nodes crash due to energy drain for instance.

#### **2.4.4 Scalability and Flexibility**

As defined in prior sections, DTNs are vulnerable for any wrong update in the network can cause failure for the whole network. Thus, the network density should not affect dramatically the performance of the protocol. It should be flexible enough to reach a level where the addition of a new node will not affect the reliability of the applied mechanism. Any new change to the network structure should not overload its performance and be scalable to a point of stability.

## **2.5 Green Wireless Networks**

Green Wireless Networks are energy efficiency networks where the main concern is to maintain the same productivity but with a low power consumption rate. This technique is now becoming a global demand. In [12], “Energy Efficiency” is defined as the ratio of useful work to the total supplied energy. Two basic methods were used to measure the energy; the first one is the ratio of efficient output power/energy to total input power/energy. The other one is simply the performance per unit of energy consumption.

In [13], the authors provided a solution where a mechanism is used to switch the participating actors off and on, to save energy. This mechanism is based on a prediction module, where it will identify low traffic period in the network. Thus the contributing sites will be turned off to conserve energy.

In [14], the authors showed that with proper power saving management techniques, the network could save energy in a considerable way. The tradeoffs taken into consideration in the paper was the power mode of the wireless card of the wireless nodes in order to save battery in DTN environments. The case where the energy saving had a big effects on its saving is when the wireless card was powered on and there was no need for transmission or reception of any packets.

## **2.6 Conclusion**

This chapter gave a brief description of the DTN structure, challenges and applications. The challenges introduced are still within the scope of the researcher’s studies progress in order to be optimized. Many protocols were proposed for such a topology in order to tackle the mentioned points of the DTN main challenges. In the next chapter, we will introduce a protocol called RPL, in which we will show how this mechanism can optimize some of the challenges of DTNs.



## **Chapter 3**

### **ROLL/RPL Protocol**

#### **3.1 Introduction**

The Ripple Routing Protocol (RPL) was introduced by a working group called ROLL (Routing Over Low and Lossy networks) in the year 2008. This working group was formed by the Internet Engineering Task Force (IETF) in order to form a solution for IP smart networks [4].

In this chapter, we will define the RPL term, architecture and the protocol process. We will explain the importance of this architecture in delay-tolerant networks as well as some of its drawbacks.

The primary goal is to design a protocol where its functionality would be to address the intersection of the application specific routing requirements (discussed in the next section). RPL was implemented to target LLNs where constrained devices are interconnected by lossy and high error links. Note that the RPL is still a work in progress.

#### **3.2 RPL Routing Requirements**

The IETF Working Groups usually, before defining any new protocol, produce a requirement document showing the reasons behind the launching of a new protocol, and also to pinpoint the challenges to be tackled, plus to highlight its features. Those requirements targeted mainly the routing issue for the LLNs, and they are summarized next [15]:

a) **Adaptive Routing:**

Dynamic paths are required to be computed on the spot if any changes occurred during transmission, and those paths should be capable to consider other constraints to adapt for.

b) **Constraint-Based Routing:**

The routing protocol should be able to support multiple constraints and it will be not only based on the provided link attributes but also on the node characteristics such as energy, memory and processing properties.

c) The Traffic Characteristics:

LLNs usually send data from the leaf nodes to a single destination node, where this kind is also referred to multipoint-to-point (MP2P) traffic. This routing protocol should support also point-to-multipoint (P2MP) and point-to-point (P2P).

d) Scalability:

Scalability is very important in this kind of networks, where large number of nodes could exist, thus the routing protocol should have specific rules to support this number.

e) Configuration and Management:

The routing protocol should support zero-configuration for any added nodes on the network. In other words, zero-configuration [16] means that a node can obtain an address and join the network on its own, without human intervention.

f) Node Attributes:

The routing protocol should take into account device characteristics such as battery operated nodes, and how to minimize the power consumption.

g) Security:

In [15] it is mentioned that “ the routing protocol must gracefully handle routing temporal security updates (e.g., dynamic keys) to sleeping devices on their ‘ awake ’ cycle to assure that sleeping devices can readily and efficiently access the network.”

### 3.3 RPL Protocol Definition

Most networks have some level of redundancy with more than one path between the source and the destination. It is the role of the routing protocol to find the best path generated by the computations of the metrics and objective functions. RPL basically is an IPv6 routing

protocol specified for LLN that defines a way to build a graph, further explained in detail in the next section, known as DODAG (Destination Oriented Directed Acyclic Graph), as a logical routing topology built over the physical network, that will be implemented from each node in the network to the DODAG root (usually an LBR-Lossy network Border Router) in order to optimize its routing process. RPL is a distance vector based protocol, and the main reason why it was designed in this way and not to be a link state protocol, is the nature of the nodes in LLNs. The link state topology is more powerful for it is known by all the nodes in the network but the only drawback in this protocol, is that it requires a vast amount of resources (memory, control traffic). DODAG is based on a collection of metrics and constraints that will be constructing the routing mechanism along with an objective function extracted from several parameters such as the link quality, shortest path, and others.

The challenging approach here is, first, how to choose wisely the required constraints and second, how to balance between the chosen constraints. One thought can come to mind, which is to combine all the constraints together to benefit from all; unfortunately such a mechanism is unrealistic and undesirable. All the constraints and metrics combined may not even be beneficial and also not possible due to the nature of the nodes.

That is why in this thesis; we shall choose carefully the required metrics since we are applying the ROLL/RPL protocol to a DTN and not any type of LLNs.

Multiple routing topologies can be used in a network, in a way that the participating nodes may join the existing graph or topology based on specific constraints.

### **3.3.1 Metrics/Constraints in RPL**

In [17] there exists a defined set of metrics and constraints used for the RPL protocol; in the following list some of these metrics, that are relevant to this thesis, will be discussed:

- a) Local or Global.

Global metric is propagated through the DODAG built graph, whereas the local metric indicates the node's local cost.

b) Aggregated Or Recorded.

Aggregated metrics is defined by the summation of the cost of all links along the chosen path, on the other hand, in some cases recorded metrics along some specific path might come at hand.

c) The Node State and Attributes Object (NSA).

The function of the NSA object is for node information. The node in a network is a critical point for reducing the traffic, thus based on the local defined policy it is sufficient to implement 1 flag that upon it, will be the decision whether to include that node in the routing path or not.

d) The Node Energy Object.

In this thesis, the energy is a very important element to be considered especially in the routing mechanism. Two main issues can be taken into consideration:

- Node Power Mode. Three flags are used indicating the following operation mode of the node: powered by main source energy, powered by battery source energy or powered by green source energy (solar system).
- Remaining lifetime. The remaining lifetime is computed by the simulation for the battery operated and green nodes.

e) Hop Count Object.

The number of hops counted along the path. The object may be used as a constraint to indicate the maximum or minimum number of hops that a path should traverse. The object can be used as a metric also by incrementing the count field for each hop the node makes. In this thesis, having a larger number of hops (for a predefined distance) would be a better choice for the routing path where the energy loss in wireless links is proportional to the square of the distance ( $\sim kd^2$ ).

The Distance Related Energy Loss Equation is given by:

$$(Kd_1)^2 + (Kd_2)^2 < K(d_1 + d_2)^2 \quad \text{Eq. 3.1}$$

f) Latency Object

This object is used to report the path latency or delay. We should note that the latency can be used as a metric or as a constraint. If it is used as a metric, usually we are interested in the total latency of the network or the maximum/minimum along the different paths. However, when the latency is used as a constraint, it is used to exclude some links that have large unacceptable delays.

g) Link Reliability Object

In LLNs, the links are lossy and hence the bit error rate (BER) can be high. In [18], the expected transmission count metric (ETX) is used as a reliability metric, which is the average number of packet transmissions required to transmit a given packet successfully. For the computation of the ETX, the method denotes to send regular probes in every direction to calculate the deliver ratio for specific links. Furthermore, the ETX equation could be defined by:

$$ETX = 1/df * dr \quad \text{Eq. 3.2}$$

Where  $df$  is defined as the probability that a packet will be received by a neighbor and  $dr$  is defined as the probability that the acknowledgment packet is received).

### 3.3.2 The Objective Function

The objective function consists of metrics and constraints used to be constructed and analyzed in order to choose the best path in the routing protocol. The difference between the two stated terms is that a metric can be defined as a quantitative expression to evaluate the path cost. As for the constraint, it can be defined as extra information added to the metric definition to be evaluated as criteria. Such metrics could be the state or energy of the node while the constraint could be the latency and reliability of this specific node. For example, the link metric is representative of the link delay, the path cost represents the total

propagation delay to the destination and the objective function may specify finding the shortest path based on the propagation delay. The constraint is used to keep or eliminate components (such as nodes or links) that do not meet defined and specific criteria.

In [17], a collection of metrics and constraints defined for links and nodes for RPL is stated. The important value of the objective function is the dynamic behavior, in other words, the parameters taken into consideration such as the link or the communication are variables subjects due to the LLN characteristics, thus the metrics and constraints defined should be taken into account when specifying this routing algorithm. There a slight issue to be discussed when it comes to dynamic values, where it can drain the whole functionality if it comes to reevaluating every time something change, by that some discrete value should be taken into consideration, such as the accuracy of the metrics used, therefore reducing computation on a frequent basis. With LLNs, several objective functions are used because of the variability of the different requirements in terms of path quality. (See Figure 3.1)

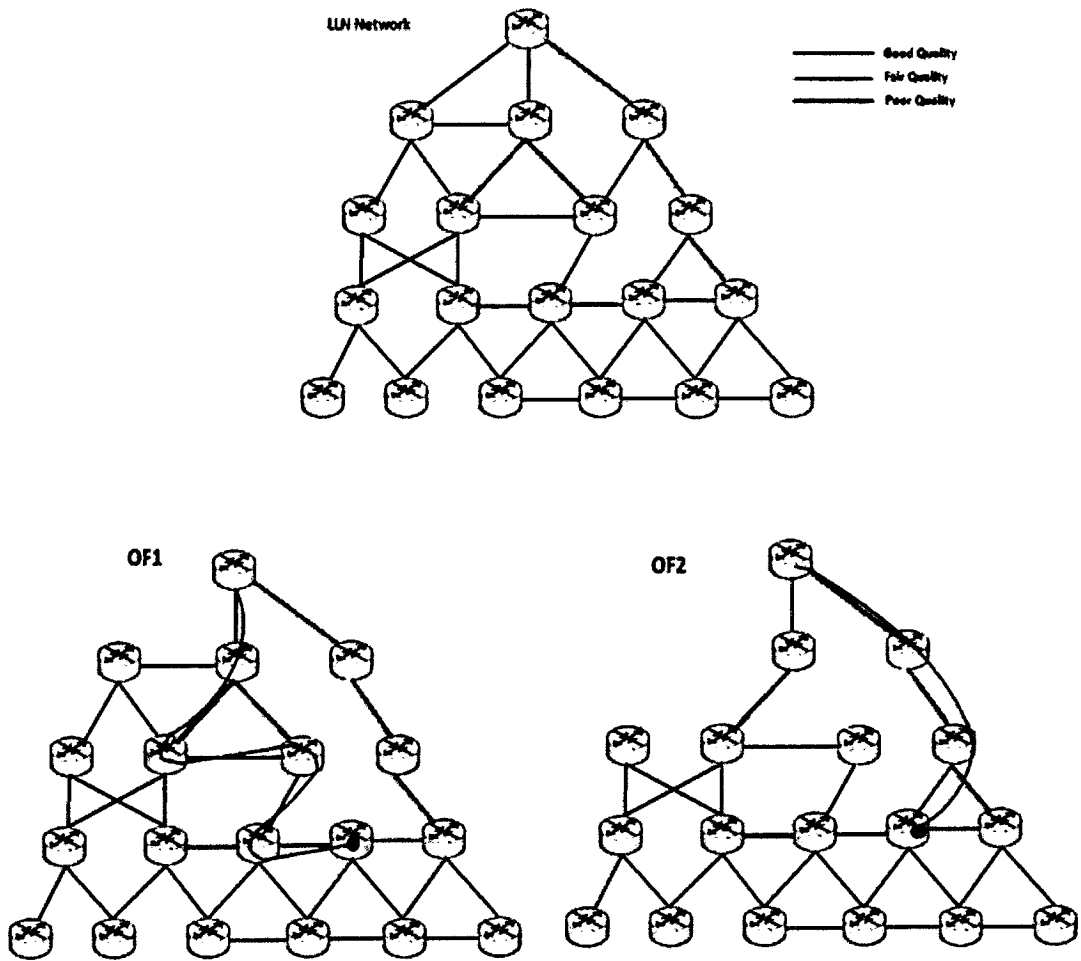


Figure 3.1 - Different Objective Functions

### 3.3.3 RPL DODAG Architecture

The RPL architecture in [19] is formed by constructing the DODAG graph which constitutes of the following:

The first step begins by the configuration of the root node which is also called the LBR (LowPAN Border Router) which is done by the administrator. The next step consists of sending messages and information between the nodes of the network. These sent instances are divided into three components:

- DIS defined as the DODAG Information Solicitation
- DIO defined as DODAG Information Object
- DAO defined as DODAG Destination Advertisement Object

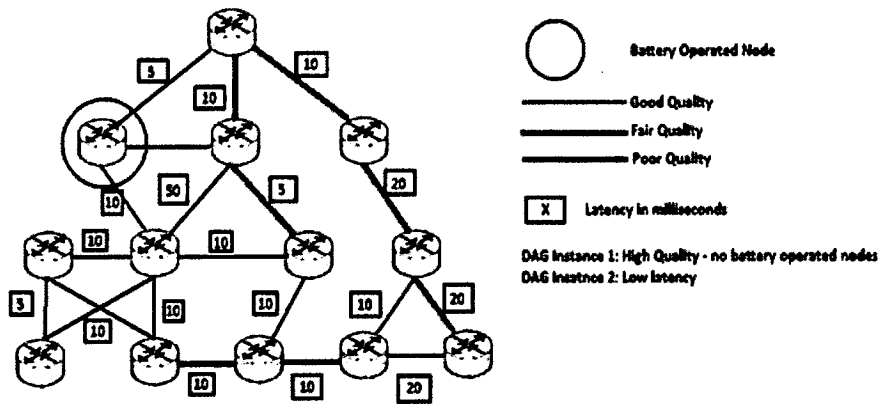
This set of instances will be an essential part of building the DODAG (will be explained in detail in the next sections), where information such as the DODAG-ID, the DODAG Rank (discussed in the next sections), along with others such as the set of path metrics and parameters discussed in previous section, will be based upon it to decide whether a node should join the DODAG or not. The DIS messages are used by the nodes in order to update the graph information and thus sending the updated messages to every node. The above process is used for the up traffic, which means the flow from the root towards the leaf nodes. For the reverse flow, DAO messages are used in order to do the advertisement from the leaf to reach the root. The body of the stated messages includes information concerning the graph such as the lifetime, prefix information, distance and others. The prefix is used by every node by adding entry information in the graph routing table thus building a complete path from the leaf till it reaches the root.

The whole process of sending the stated messages is based on the defined objective function (usually by the network administrator). The protocol of building the DODAG starts by the root where it will start sending the DIO message to all the neighboring nodes. The receiving nodes will capture the sent message and interpret it in order to decide whether to join the routing graph or not, based on the decision made the joined node will define itself as a rank in this graph hierarchy. The next step will include sending the updated graph information made by the joined node and advertised to its neighboring nodes. Thus the name Ripple protocol was defined where this step is propagated throughout all the nodes of the network. By the end of this step, each node will have a defined path towards its upper rank in the graph or by other words its parent node and it can have more than one parent based on the dedicated messages sent between them. In this manner the RPL protocol can eliminate the looping problem since every node will be connected to its parent and not the other way

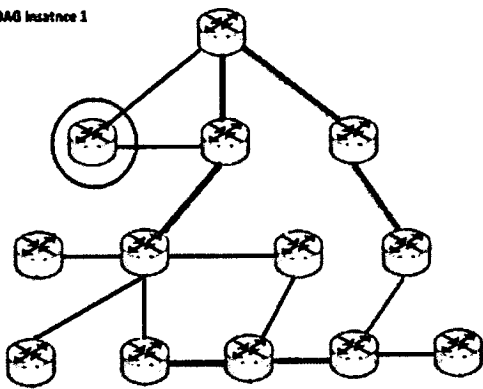


around. The RPL protocol also supports P2P (Point to Point) traffic, when a node sends a packet to another node, if the destination node is not reachable from the source node, it forwards the packet to its DODAG parent, therefore if the destination is reachable from one of the parent's children it will traverse the proper path to reach its destination.

Different DODAG instances can exist in the same network providing different paths to traverse based on defined criteria by the network (See Figure 3.2); such criteria could include critical and non-critical information. In other words, when the network is implemented in such a way to give importance level to each sent message, based on each node the message will be sent to different paths based on the built objective function to ensure that the high level important messages will have the higher probability to reach its destination than low level important messages. Thus the notion of the DODAG instance came along, and the idea behind it is to be able to construct multiple DODAGs over a given physical network. This is done for one reason which is to include different optimized paths according to different given requirements. An example could be given with the following scenario, consider the case where a network contains different types of nodes such as battery and main powered nodes, and the existing links between them varies based on bandwidth, reliability, throughput, latency, etc. . Two main applications will be using this network where one will consider a short delay and a highly reliable path function, whereas the other will consider non battery operated node path and low hop count function. The RPL protocol handles those two requirements by building two DODAGs graphs related to the two objective functions in order to compute two paths for different placed metrics (see figure 3.1).



DAG Instance 1



DAG Instance 2

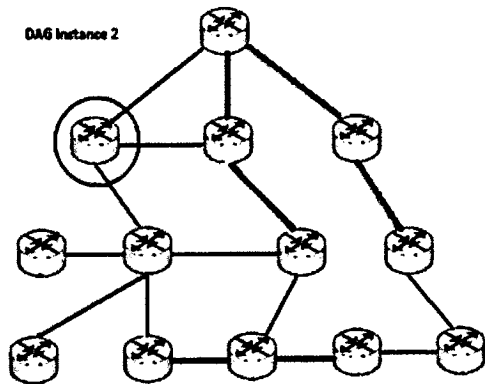


Figure 3.2 - DODAG Multiple Instances

### 3.4 RPL DODAG Messages

In the next sections, the RPL messages used in the building process of the DODAG will be explained based from this source [18], which are the following: the DODAG Information Object (DIO), the DODAG Destination Advertisement Object (DAO), and the DODAG Information Solicitation (DIS).

### 3.4.1 DIO

The DODAG Information Object known as DIO messages are used by the RPL nodes in order to advertise the DODAG being built and its properties. Thus, those messages will include mandatory and essential information related to the formation of the DODAG.

The following flags are found in a DIO message:

- The Grounded (G) flag is an indicator to show if the DODAG can accomplish the goal of the application defined.
- The Destination Advertisement Trigger (T) is a flag to trigger a complete refresh cycle of the routes.
- The Destination Advertisement Stored (S) is a flag to show that an ancestor, who does not have the root property, is storing the routing table entries.
- The Destination Advertisement Supported (A) is a flag to indicate whether the DODAG root supports the advertisements prefixes.
- The Destination Advertisement Trigger Sequence Number (DTSN) is an 8-bit field that is initialized by the node sending the DIO message.
- The DODAG Preference (Prf) is represented by a 3-bit field to define how preferable the root of this DODAG compared to other DODAG roots within the same instance.
- The DODAG Sequence Number is represented by the DODAG root and it characterizes the DODAG iteration.
- THE RPL Instance ID is an 8-bit field to show which RPL instance the DODAG is representing and provisioned at the DODAG root.
- The DODAGID is a unique identifier for the DODAG and represented by a 128-bit field.

- The DODAG Rank is the rank of the node that is sending the DIO message and represented by a 16-bit field.

#### 3.4.1.1 Node Rank

Based on [18] the rank of a node is defined by:

“A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root. Rank strictly increases in the Down direction and strictly decreases in the Up direction. The exact way Rank is computed depends on the DAG's Objective Function (OF). The Rank may analogously track a simple topological distance, may be calculated as a function of link metrics, and may consider other properties such as constraints.”

Another reason for implementing the rank is for loop avoidance. The rank itself is initialized from the routing metrics, as from the objective function. For instance, for the node to select its parent in the DODAG, cannot select as a parent a node with bigger rank. The more the rank is smaller means that the more the representing node is closer to the DODAG root, in other words, the rank of any node in the DODAG is always higher than the rank of any of its parents.

The node Rank is also used for selecting the parent of every node in the DODAG. The MinHopRankIncrease is a value advertised by the DODAG root and this value is the decimal point of the rank while the integer part of the rank represents a value called the floor calculated by  $\text{Rank} / \text{MinHopRankIncrease}$ . It signifies the minimum amount that a rank can increase on each hop and the other purpose is detecting the node siblings. While considering the floor formula, the rank itself can be compared between the nodes. For instance, if we have two nodes X and Y, if the node X has the floor less than the floor of Y will imply that the node X has a rank less than the rank of the node Y; and Vice Versa. But if the floor of two nodes was equal, will imply that the two nodes are siblings.

In later sections, in the process of building the DODAG, the rank will be explained and how its importance will be to avoid the loop between the nodes.

### 3.4.2 DAO

The purpose of the DODAG Destination Advertisement Object known as the DAO messages is to disperse the information related to the destination along the DODAG by populating the routing tables. The DAO messages include the following format:

- The DAO Sequence, which represents a counter for each DAO message sent by the node.
- The RPL Instance ID is the same as in the DIO message, it shows which RPL instance the DODAG is representing and provisioned at the DODAG root.
- The DAO Rank that will include the rank of the node explained in previous section.
- The DAO lifetime, is the lifetime of the message.
- The Route Tag will be used to tag specific routes in order to optimize the selection of the routes in certain nodes.
- The Destination Prefix contains the number of valid bits that will be included in the prefix of the message.
- Reverse Route Stack is used by nodes that does not have the ability to store the routing tables, thus storing the stack of the nodes where the routing tables are reached.

### 3.4.3 DIS

The DODAG Information Solicitation known as DIS messages, are used to solicit a DODAG Information Object (DIO) from RPL nodes and for the discovery of other DODAGs.

### 3.5 RPL DODAG Building Procedure

Before digging in the whole process of building the DODAG, the trickle timer used for sending the RPL messages is explained.

#### 3.5.1 RPL Trickle Timer Process

As in [20] and [21], the trickle timer is proposed to keep all the nodes updated in an optimized way, and especially without over heading the traffic in the network. The trickle algorithm controls the sending rates of the messages sent in the network in such a way that the nodes will hear enough packets to stay in an updated state under different situations. RPL will be using trickle timers in the DODAG building process. Thus, when there is an inconsistency occurring in the DODAG nodes, the RPL messages will be sent more often so that the DODAG nodes will become more stable. Furthermore, from time to time, a node will be transmitting data till it hears from mother transmissions occurring in the network that the data (such data could include routing state) being transmitted is redundant. The following format is used in the trickle timer algorithm:

The “I” field that corresponds to the current communication interval size.

The “T” field that includes the timer value and it ranges between two values: “I” and “I”/2.

The “C” field is a counter that is used for redundancy purposes.

The “K” field is a constant captured from the DODAG root and that is used for redundancy purposes.

The “I min” field that includes the smallest value of I captured from the DIO message and calculated by  $I_{min} = 2DIOIntervalMinms$  (DIOIntervalMin is advertised by the DODAG root and propagated by the DIO messages).

The “I max” field is the largest value that I can contain which is calculated by  $I_{max} = I_{min} * 2^{Idoubling}$ .

The “I doubling” field represents the number of times that I may have the value doubled before reaching a constant state. “I doubling” is captured from the DIO messages and advertised by the DODAG root as the DIOIntervalDoubling field.

The trickle variables are set by the RPL node captured from the DIO message, the counter is set to 0 and the two variables “I” and “I” min are equalized in value. The counter is incremented every time the node receives a DIO message from the DODAG parent.

When the timer expires, the counter is compared to the RPL constant to check whether to send multiple DIO messages or not. The RPL tickle timer is initialized all over again in several conditions: for instance, every time the DODAG is in an inconsistent state, or when a new node is to be joined to a DODAG topology, or when the RPL messages are propagating in the DODAG to state that there is an update and so forth.

### **3.5.2 DODAG Building Procedure**

When the initialization process starts in a node, two options are available. The first one is to stay in a silent mode until a DIO message is advertised to show that a DODAG exists, and thus this node will advertise by itself a DIS message to inform the neighborhood and to receive DIO messages from the other nodes that are neighbors. The second option is to start by its own DODAG and to begin advertising it by sending DIO messages. The rank is set to 1 if the advertising node is itself the DODAG root.

As shown in Figure 3.3, when the node receives a DIO message, it must first check if the message is corrupted, if it is then the node will discarded it. If the DIO message is not thrown away by the node, then it is processed. In that stage, to process and forward the message, a node may wait to send the message to the neighbor for example to check if the link is reliable. This is called local confidence between the nodes, especially between the neighbors.

If the rank of the DIO advertising node is less than the receiving node, and the DIO contains a more optimized path according to the Objective Function, then the DIO message is

managed by the node. Furthermore, the DIO message must be processed by the receiving node, if this advertising node changed the DODAG instance. Collisions between the DIO messages have a small probability to occur, since the sending of the DIO messages is controlled by the tickle timers and in a random way. But if it happens, it will be dropped and advertised again by the advertising node in a random time.

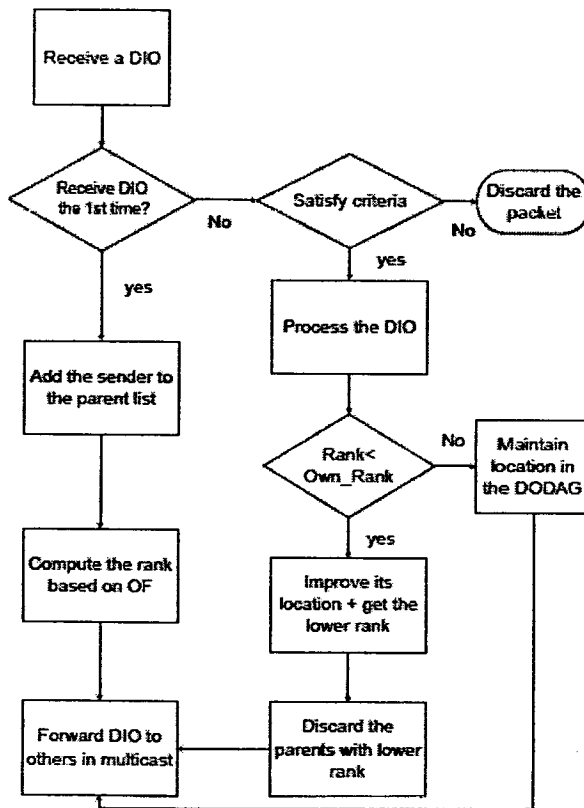


Figure 3.3 - DIO Process

As showed in Figure 3.4, the DODAG root starts by sending DIO messages. The Objective Function implemented in the figure is the minimum path with a constraint to avoid any nodes that has the battery as source energy. Thus, the process is built depending on the Objective Function and based on that every node start choosing its parent and siblings.



After the DODAG is constructed, the next step is to spread the routing tables along the nodes participating in the DODAG. For this stage, DAO messages are sent along the DODAG in order to accomplish it. The DAO message includes a sequence number in order to check if the information being populated is updated or outdated, then the outdated information or if there is duplication, those messages will be discarded. This sequence number will be incremented every time the nodes use it. The DAO messages are then sent from the nodes to their parents. The rank is found in the sent DAO messages in order to let the receiving node to be able to select the next hop based on the Objective Function.

Any node can choose a different DODAG at any time, for instance, if a DODAG was newly formed based on a different network requirements, a different Objective Function, then the node can join this new DODAG and select maybe a different DODAG parent. But the node must transmit all the queued packets before joining a different DODAG. Furthermore, the node will recompute its rank based on the new joined DODAG for re-computation will occur when the new DODAG is formed. If a new DODAG was built and a node was not able to join the DODAG based on the populated Objective Function, then the node can either not to join the graph or to join the graph but as a leaf node. As a leaf node, it can process the incoming DIO messages and send DAO messages.

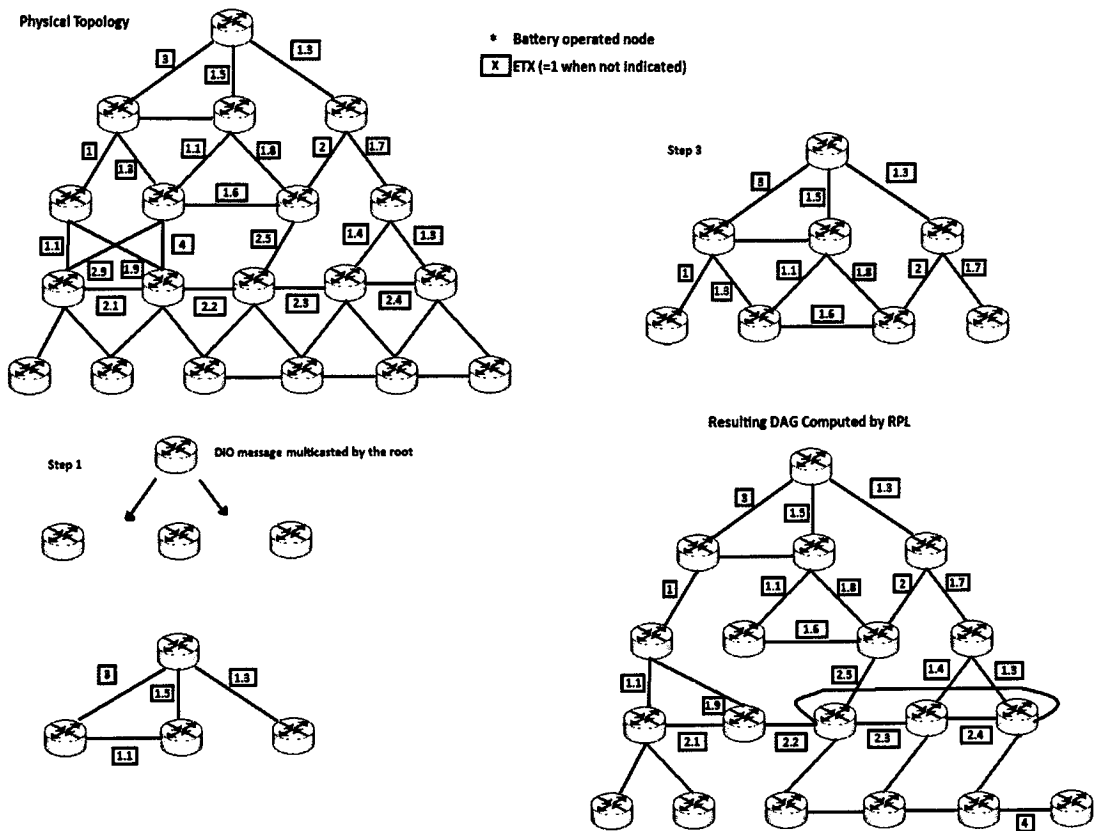


Figure 3.4 - DODAG Building Process

### 3.5.3 Loop Avoidance and Detection

Traffic is low in the Low Power and Lossy Networks, thus when a loop occur it does not leave a noticeable impact on the network. The RPL protocol uses the loop avoidance and detection mechanism rather than controlling if any loop did occur. Taking this strategy into consideration, there will be no standalone control mechanism over occurring loops in the network, only avoidance and detection. In the parent selection process in RPL, the node must not select another node as a parent without checking if the node's rank + DAG Mac Rank Increase are lower than the parent node rank. One of the reasons is reducing the possibility that a node will link to another node that is already in the same DODAG. For instance, in

Figure 3.2, if the node with the number 24 delete it the link between its parents and due to the network requirements, selected the node with the number 46 as a parent; with this decision, a loop will occur due to the path to the root from the node with number 46 is passing by the node with number 24. The node with number 24 has no ability to find if the node with number 46 belongs to its own DODAG.

Another case where a loop can occur is when a DIO or a DAO message is lost, in later sections; a repair solution will be discussed for RPL messages lost. If a node did not populate to its links with its parents that a specific link to a destination is broken, then a DAO loop may occur. In this case, the parents may still have the routing table showing that this destination is reachable, thus sending the packet to the broken link, and by that the child of the parent has an updated routing table showing that this destination is unreachable. Therefore, the child will resend the packet to the parent for another route possibility, while the parent will resend again to the same path, thus creating a loop.

The concept of the loop detection mechanism is to set a flag in the forwarded packets. For example, if a packet is to be forwarded from a node to a sibling to that node, this flag will be set to indicate that the packet has been forwarded to a sibling, when it will get to the next destination, if the packet need to be processed and sent again to another sibling, then the packet is dropped. DAO loops also can be detected by setting a bit in the packet, when a packet is sent and the routing table at the receiving node shows that the packet should be sent towards the DODAG root, then the DODAG is unstable and the packet should be discarded since a loop was detected.

#### **3.5.4 Repair Mechanism**

Links and nodes in any network are subjects of vulnerability and changeability, for that reason, some mechanisms has to exist in order to repair it. At first, if a topology has changed due to any element in the network that has malfunctioned or changed its properties, then a need to change the whole DODAG is brought into consideration. But in environments such as LLN, repairs should have the knowledge that the network in hand has lossy nature

and low power structure. One of solution proposed is rebuilding of the DODAG topology in order to be updated with the latest changes that occurred in the network. Therefore, the repair mechanism should be careful when it comes to the application of the mechanism, taking into account that the least error can cause the whole network to fail. Two techniques were proposed by RPL: Global and Local.

The local strategy is implemented by finding another route (any other available route and the chosen route may not be the best) other than the damaged or failed one, in order to send the message. This event is triggered by the node that discovered that the link, where the message is being sent, has been damaged. Global repairs are triggered by the DODAG root upon the generation of a new DODAG Sequence Number. Thus, with this trigger, the Objective Function will be reevaluated and the DODAG will be rebuilt all over again. This technique is also used as a re-optimization for the whole network of the DODAG.

During the repair mechanism, RPL makes sure that no loops will occur due to the Max-depth rule. By defining that a node does not have the right to advertise a rank less than or equal to any of its parents, RPL will avoid the loops during the repair process. Also, within DODAG iteration, a node does not have the right to advertise a rank more than  $L + \text{DAG Max Rank Increase}$ , where  $L$  is defined as the lowermost rank that the node has taken in this iteration, and the DAG Max Rank Increase is a variable promoted by the DODAG root. The Count To Infinity problem is defined whenever a node need to connect to another node and make it as its parent, but the parent should be a DODAG parent and not one of its children. Therefore, if this scenario took place, it will produce an infinite loop due to the continuous increment action of the node's rank. As showed in Figure 3.5, the repair mechanism is proposed and demonstrates how the loop is detected. An example scenario would be the following: The node 12 has a link to the DODAG root, suppose that this link has failed for a specific reason and the DAG Max Rank Increase is defined as 5. Suppose that in this case the node number 12 has the rank equal to 2. If we want to calculate the variable DAG Max Rank Increase plus the variable defined by the node which is the rank, the result would be 7. That would imply that the node with number 12 can join any other node with a rank that will not bypass the calculated result. If we want to take the best case where no loop will be formed,

the node with number 12 will select the node with number 21 as its DODAG parent. But for the sake of the chosen scenario, the node with number 12 selected the node with number 32 as its DODAG parent, and selected node was on the path of the selector. The node with number 12 will send a DIO message to update its own rank in the DODAG which will be equal to 5. The next chain will be as follows, the node with number 2 will update its own rank to value equals to 6, the node with number 32 will then update also its own rank so that will equals the value 7, which will push the node 12 to update its own rank again to be equal to 8. This final rank will violate the proposed rule of the DAG Max Rank Increase + L, and at this point the loop will be detected and stopped.

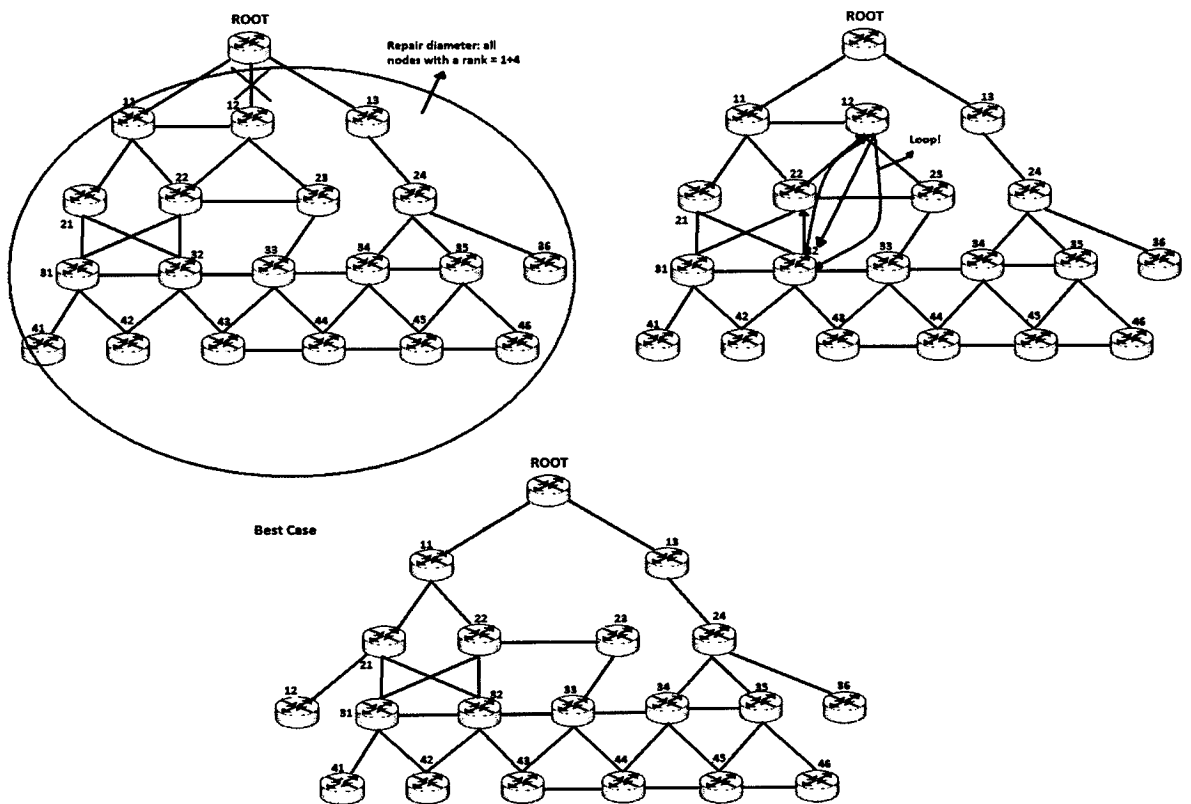


Figure 3.5 - Loop Repair Mechanism

### **3.6 Previous Related Work**

In what follows, we will state some previous work on the RPL protocol showing results studied by some papers. Various papers took different metrics to study the RPL protocol mechanism and how it responds to those chosen metrics through simulation results.

#### **3.6.1 Performance Evaluation Study of RPL for LLN**

In [22], the authors contribute to their study to provide results of the RPL protocol on a medium scale network using the following metrics:

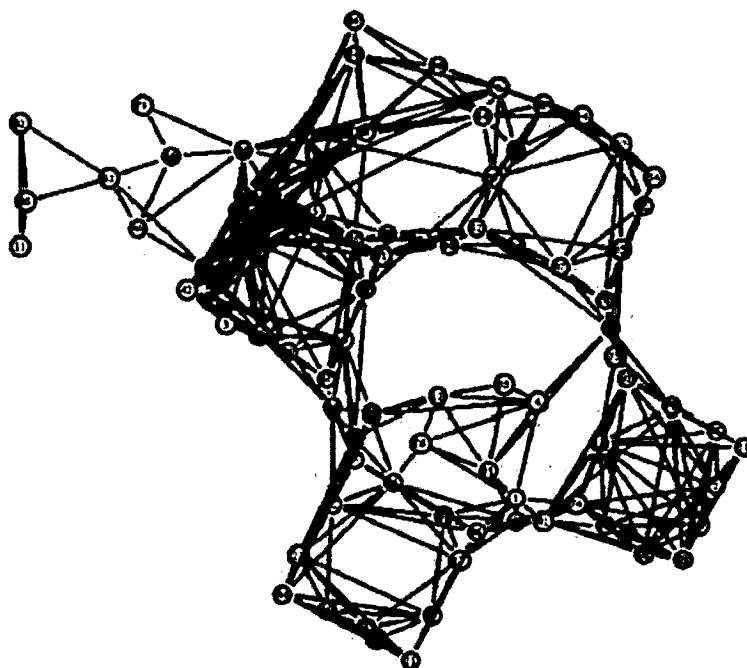
- a) Path Quality metric
- b) Control Plane metric
- c) Required resource constraints on nodes

The study started by giving a brief explanation of the RPL protocol process. Then, gathered some real data that has been used for two aspects in the simulation. The first one is the topology, 86 nodes were used with a single root, shown in Figure 3.6; the red links represent the links between nodes and their parents. The second aspect is the link failure model, which defined that some packets randomly will be dropped on random links.

The study used OMNET++ to perform the simulation, and based on the computed results, the following was produced. Based on the path quality metric, it turned out that for a given network architecture, 90% of paths will have a 5 hop count path as an optimized path for the shortest path routing protocol, whereas for the RPL protocol, 60% will have a 5 to 8 hop count path as an optimized path. While the ETX (Expected Transmission Count) metric shown in the simulation results, is close for both protocols from all the sources to all destinations. Another metric was studied also, which is the routing table size, showing that 90% of the nodes need, more or less, 21 entries in the routing table. Control packets were also studied in terms of the nodes transmission, the result was that the more close the node is to the root, the more data packets are transmitted, while the control packets traffic was negligible. The transmission of those packets is controlled by the trickle timers. When the

DODAG is stable, the control packets number is minimized, but when an updated DODAG is advertised the number is raised.

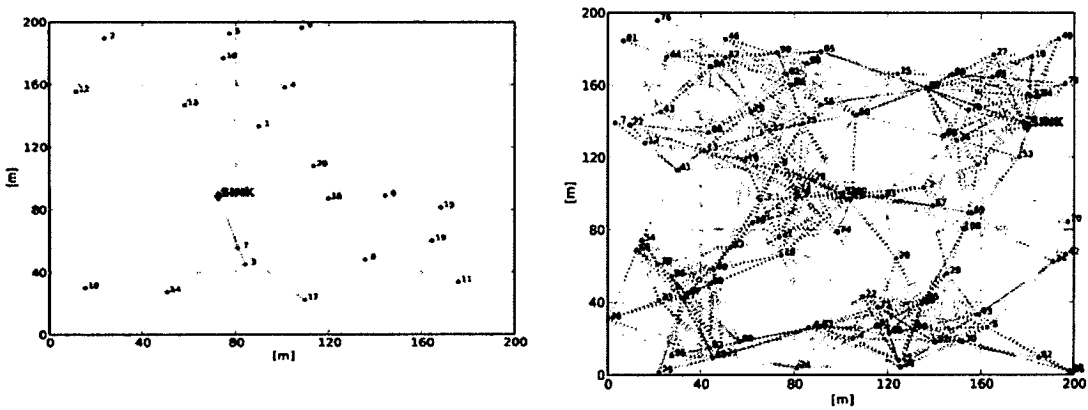
Therefore, the conclusion of the simulation results was that trickle timers have kept the control packets less than the data packets.



**Figure 3.6 - Network Topology used for the Simulation**

### **3.6.2 Performance Analysis of the RPL**

In [23], the authors present simulation results to show how the RPL can ensure a fast network set up especially in critical conditions. The study began with a brief explanation of the RPL protocol, and then presented the following simulation results. The RPL protocol was simulated on two wireless sensor networks that contain respectively 20 and 100 nodes (Shown in Figure 3.7).



**Figure 3.7 - Network Topologies with 20 and 100 nodes**

In these simulations, the time elapsed for the network to be stable was about 10 minutes. And hence the network took this amount of time to reduce its overhead with different Packet Error Rates (PER) from 100% to 25% for the first case (20 nodes) and from 100% to 50% for the second case (100 nodes).

Then the paper studied the aspect of the network throughput, thus showing that RPL allows a fast network set up, but the main difference is that the throughput from the first case (20 nodes) is lower than the second case (100 nodes). Also, it showed that the RPL signaling is very high compared to the data traffic, due to the dominant transmission of the DAO messages. The transmission of the DIS messages is done once at the beginning of the simulation but the DAO messages are advertised from the leaf nodes towards the root, which explains the signaling overhead.

Communication delays are directly related to the distance from the root, also known as the rank of the node. This paper shows that in both scenarios the delay did not pass the time limit of 2 seconds.

This study concluded that further studies could be done in order to reduce the overhead. And that the RPL protocol is a powerful technique due to the fast network set up.



### 3.6.3 Simulation and Performance Evaluation of DAG Construction with RPL

In [24], the authors provided simulations in order to show the performance of the network construction mechanism with the RPL protocol by varying different parameters. The authors started by giving a brief definition of the RPL protocol, and then presented the simulation process and the results. The simulation canvas was set to be a 600m \* 600m region, with one DAG root. The location of the root provided a significant impact on the simulation. The following performance metrics were analyzed:

- Average Power Consumption
- Average Hop Distance
- DAG Convergence Time
- Routing Table Size
- Packet Loss

The simulation in [24] showed that the power consumption is directly related to the number of nodes in the network; also, it showed that the energy consumption is less when the root is in the middle than if it was at the border of the network for the nodes will be forwarding more packets to their neighbors. The same results were shown with the hop count, where the average is less if the DAG root was in the middle than at the border. On the other hand, these simulations showed that the convergence time (DODAG construction process time) is independent from the number of the nodes in the network. The routing table size was proportional to the number of the nodes and the location of the DAG root.

The simulation took another turn to show the results with different objective functions. The first objective function is  $OF_0$  which is to minimize the hop count while the second  $OF_1$  is connected to the ETX. The convergence time was the same in the two objective functions, whereas the number of hops and the energy consumption was lower in the  $OF_0$  than in the  $OF_1$ .

The paper concluded that new objective functions should be studied in order to optimize the required application requirements.

### **3.6.4 A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)**

In [25], the authors provided an evaluation with observation for the characteristics protocol. This paper started by a brief definition of the RPL protocol, and then several aspects were taken into consideration.

One of these aspects is the traffic flow, the paper stated that the upward traffic, specified from the node leafs to the root, is predominant; this is critical for the resulting effect will be a congestion at the root.

The fragmentation part should be handled carefully, for at the link layer packets are usually fragmented when they reached a max size, so in order to be transmitted packets are divided into parts. This process could generate more traffic to the network, where this part should be avoided since RPL is intended for lossy networks.

The nodes can operate under two modes, storing or non-storing, which lead to a problem with a network consisting a large number of nodes when it comes to store the routing table in every node.

Parent nodes are selected based on the DIO messages, with no verification from the nodes that the selection is the best, in other words, there is no reply from the children nodes that the chosen parent is the best choice.

The implementation of the RPL protocol is expensive from different aspects such as security, manageability, auto configuration features and more. But RPL was designed for LLN, due to all the features cost, will this protocol support all the features when it comes to the implementation.

Control messages of the RPL are not well specified, such as the DAO messages and the DAO-ACK. Thus, the implementation of the RPL with no specific mechanism for such messages would include a bad performance.

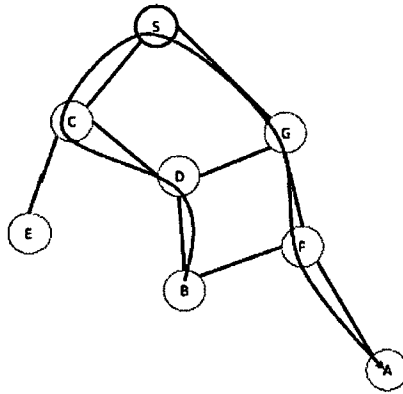
The link quality in real application of LLN environments vary very often. Taking into consideration the convergence time, where DIO messages are emitted in order to construct the DODAG will not be optimized due to frequent repairs.

Loops are handled by the RPL protocol where detection mechanisms are used. But this process should be studied in a way to minimize the congestion problem if a global repair is triggered.

The paper concluded by addressing further exploration of the listed aspects and limitation of the RPL, which need to be tested in real world applications.

### **3.6.5 The P2P-RPL Routing Protocol for IPv6 Sensor Networks: Testbed Experiments**

In [26], the authors provided some tradeoffs to the RPL protocol in order to propose an extension based on some evaluation results. Sensor networking is the future when it comes to the Internet of things. The main role of the sensors is distributing and monitoring various parameters from temperature, movement, and several activities by an automated manner. Specific network protocols are required to enable the communication between these sorts of nodes. Thus, a brief definition of the RPL protocol is provided to handle this responsibility. The tradeoffs listed in the paper provide some points of the RPL protocol where some changes are to be tweaked. The size of the forwarding and routing tables in the protocol is kept minimal, by not providing paths from the root back to the sensors and between arbitrary sensors (Figure 3.8). However such excluded paths are used in some types of networks such as industrial and home automation.



**Figure 3.8 - RPL Communication**

In order to introduce shorter path, a mechanism called P2P-RPL is provided. Control messages are included in the DIO messages by introducing the address to be reached in order to discover and establish paths to that address. By applying the protocol, results showed that the average route length is approximately half compared to the normal protocol of RPL. Another side of the results also showed that the traffic near the root was decreased. The objectives of the study were the discovery of shorter paths and reducing the root traffic, thus it concluded that further experiments could be done around this subject while keeping track of the tradeoffs of both protocols through different types of network requirements.

### 3.7 Conclusion

This chapter gave a brief description of the RPL protocol process. Furthermore, we provided with the mode of operation of this protocol; the DODAG building process along with its stages and discussed some of the related work done about this subject. This protocol is still a work in progress, and most of the papers that define this protocol from the IETF group are still internet drafts. Note that the RPL protocol is still under debate from different point of views, and still under study to reach its final and steady version. We gave a general idea of the whole protocol mechanism, and the next chapter shall tackle specific points related to our thesis study concerning DTN architecture.

# Chapter 4

## Proposed ROLL/RPL Protocol for DTN and Simulation Results

### 4.1 Introduction

This chapter is intended to defend our thesis study by going through the generated simulation results. The software was made in the C# language and its role was to show a simulated environment of the RPL protocol process over a network with a DTN nature. The following sections will describe the simulation process and show the generated results and provide with a conclusion at the end.

### 4.2 Simulation Building Process

A software application has been developed in this thesis for simulation of the RPL protocol within a DTN environment. Figure 4.1 illustrates the interface of the simulation software.

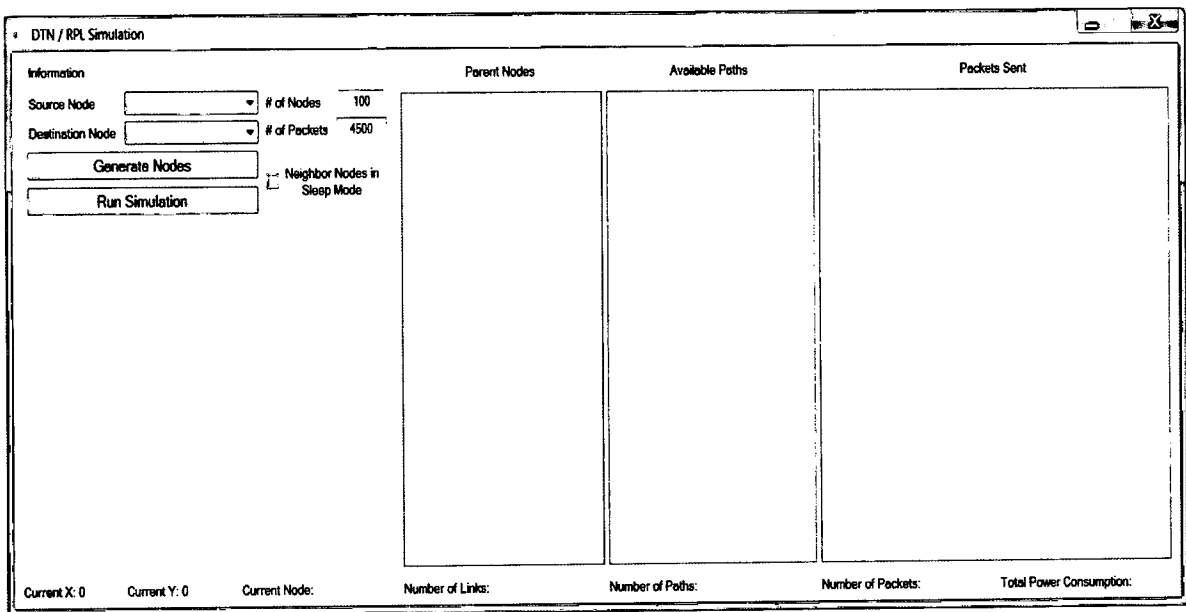


Figure 4.1 - DTN/RPL Initial Screen

It includes the following features:

- **Source Node:** Allows the user to specify the source node used to transmit the packets from.
- **Destination Node:** Allows the user to specify the destination node used to receive the packets.
- **Number of Nodes:** Specifies the density of the network in terms of number of nodes.
- **Number of Packets:** Specifies the number of packets desired to transmit.
- **Parent Nodes:** Specifies the nodes parents in the network with the list of their children. Every child will include the distance from the parent, and the assigned power energy. The number of links that are found within the whole network is displayed at the end of the tree view.
- **Available Paths:** This list will include the generated available paths based on the specified source and destination nodes. The total number of paths will be displayed at the bottom of the list.
- **Packets Sent:** This tree view specifies the simulation packets transmission results, which includes the packet ID, the path used linked with the calculated power consumption of the whole path and the calculated distance. Additionally, the number of successful transmitted packets is displayed at the bottom of the tree view.
- **Total Power Consumption:** Specifies the total power consumed by the chosen path for 4500 packets per second.
- **Sleep Mode:** Specifies whether the nodes in the network are operating under the sleep or no-sleep mode (discussed in detail in the next section).
- **Generate Nodes:** Discussed in detail in the next section.
- **Run Simulation:** Discussed in detail in the next section.

#### 4.2.1 Generate Nodes Functionalities

Based on the number of nodes specified by the user, this event will randomly generate the nodes and display them within the white canvas. Each object node has the following properties:

- Node ID: Represents the unique identifier of the node.
- Node Name: Represents the name of the node.
- X-Position: Represents the x coordinate of the node within the canvas.
- Y-Position: Represents the y coordinate of the node within the canvas.
- Node Power: Represents the power energy of the node.
- Child Nodes: Represents a list containing the child nodes linked to the node ID.
- Child Distances: Represents a list containing the distance from every child node to the parent node.
- Is Solar Powered: A Boolean variable representing whether the node is solar powered or not.

Additionally, Set and Get functions are created for each attribute.

The next step will draw the generated nodes within the canvas by assigning the X and Y coordinates with random numbers. After generating the nodes we need to specify the children for every node. This is done by calculating the distance between the nodes. If the distance is less than 40, then a link is established and the node will be considered as a child. Note that the given range of every node is randomly assigned by a value between 10 and 40. Furthermore, the parent nodes tree view will be populated with the generated data showing each parent node with its children.

#### 4.2.2 Run Simulation Functionalities

After generating the nodes and specifying the links between them, we must choose the source and destination nodes in order to run the simulation. As a first step, based on the RPL protocol functionalities, we need to find all the available paths between the chosen source and destination nodes. Those paths are displayed in the “Available Paths” list box. The second step will be building the DODAG layer on top of the network. In other words, it finds the best minimum distance path based on the objective function defined to be the destination metric. Furthermore, the packets to be sent are routed based on the process of the RPL protocol, specifically the paths built by the DODAG mechanism. Based on the chosen path, the nodes power will be decreased by a calculated value discussed in the next section. Additionally, the nodes are recharged by a green energy source. Thus, we are assuming that the list of all generated available paths is recharged for a certain specified process discussed in the next section.

#### 4.2.3 Energy Calculation

Choosing the best path depends on three main constraints that are used in the simulation:

- **Total Distance:** Represents a list containing the total distances for all the generated available paths.
- **Total Power:** Represents a list containing the current total power consumption for all the generated available paths.
- **Detail Power:** Represents a list containing the detailed link power consumption for all the generated available paths. This detail will contain the energy consumed by every link found in the path.

The energy calculation process is defined as follows [27]:

This is the transmission energy calculation formula

$$E_t = l(e + ux^k)$$

Eq. 4.1



This is the receiving energy calculation formula

$$E_r = le \quad \text{Eq. 4.2}$$

Where  $l$  is defined as the packet size to be sent,  $e$  is defined as the energy consumed for sending and receiving packets,  $u$  is defined as the energy consumed for power amplification, and  $x^k$  is defined as the consumed energy proportional to the distance taking as  $k = 2$  (wireless transmission in free space).

Note that  $e$  is considered to be equal to 50 nj/bit and  $u$  to be equal to 50 pj/bit/m<sup>2</sup>.

### 4.3 Assumptions

In the simulation process, the following assumptions must be taken into consideration:

- The number of the generated nodes can vary between 50 and 200.
- The number of packets sent is equal to 4500/sec. (see section 4.42 for calculation).
- The packet size is equal to 1500bytes.
- The packet loss factor is considered negligible and shall not be taken into consideration.
- The generation of the nodes is taken in a random manor.
- The initial power of the nodes is equal to 12096 Joules equivalent to a battery of 1.2 Volts.
- The maximum range of a node is set to be equal to 40meters.
- The area of the network is set to be equal to 460×340 m<sup>2</sup>(width × length).

### 4.4 Network Lifetime and Power Consumption

#### 4.4.1 Introduction

The network lifetime is expressed in this thesis as the unavailability of the chosen path. In other words, if the chosen path had at any time  $t$  a node that run out from energy, then it will become unavailable. Furthermore, the lifetime of the network is directly dependent to the energy related to every node. The calculation of the node power

consumption is based on Equation 4.1 and Equation 4.2. In this section, two modes of operation were considered, sleep and no-sleep modes. The first one is implemented in a way where the neighbors of the transmitting node will disable their radio activities, i.e. listening process. In the second mode, the nodes will consume energy by listening to their parent node.

#### 4.4.2 Calculation Procedure

The studied network in this thesis is based on the 802.11 G standard, which is used for wireless communications. This standard offers a transmission bit rate of 54 Mbps and a packet size of 1500 bytes (equivalent to 12,000 bits). Using this standard, we can calculate the number of packets sent per second through the following equation:

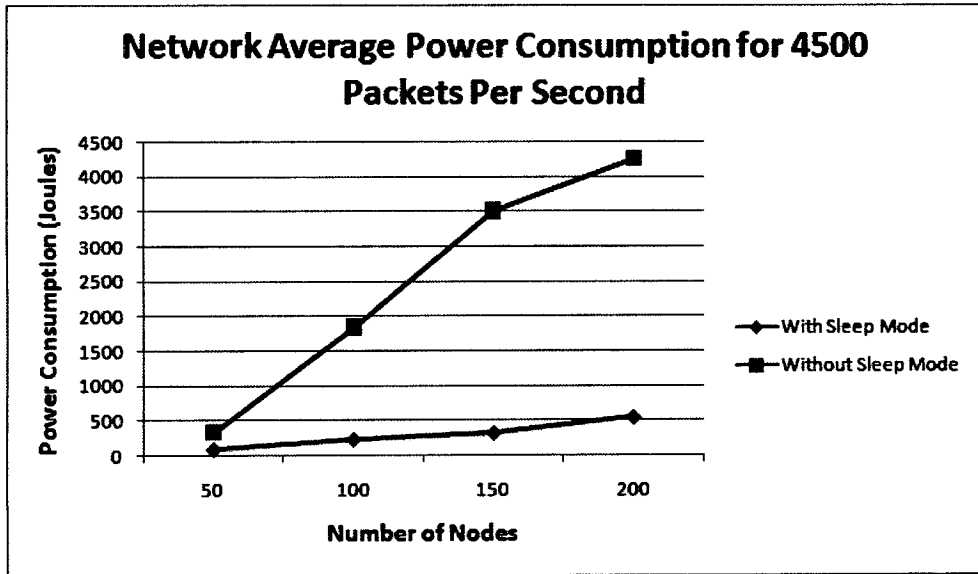
$$N_{packets} = \frac{54\,000\,000\,bits}{12000\,bits} = 4500\,packets / sec. \quad \text{Eq. 4.3}$$

Knowing the number of packets that can be transmitted per second, we can deduct the average power consumption per second for different densities of the network. Since we have taken the initial energy of the node equal to 12096 Joules, and through the usage of Equation 4.1 and Equation 4.2 to calculate the transmitted and received energies of the nodes, we can calculate the network lifetime through the following equation:

$$Network\ Lifetime = \frac{12096\,Joules}{Average\ Power\ Consumption} \quad \text{Eq. 4.4}$$

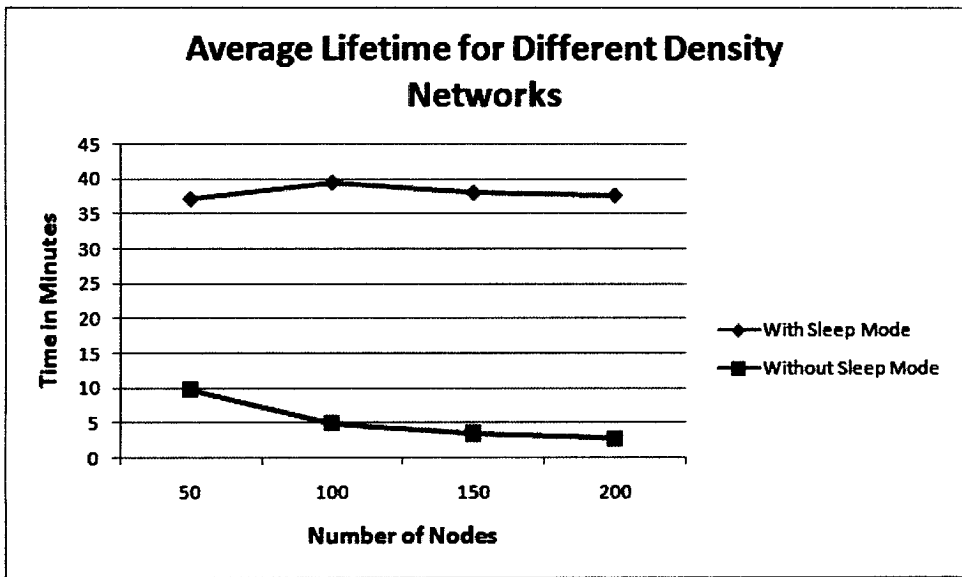
#### 4.4.3 Simulation Results

For this simulation, four scenarios were taken into consideration, where each one contains different network densities (50, 100, 150 and 200 nodes). The data rate for all the scenarios is equal to 4500 packets per second. The simulation was executed several times for all the chosen network densities with two node operation mode: sleep and no-sleep.



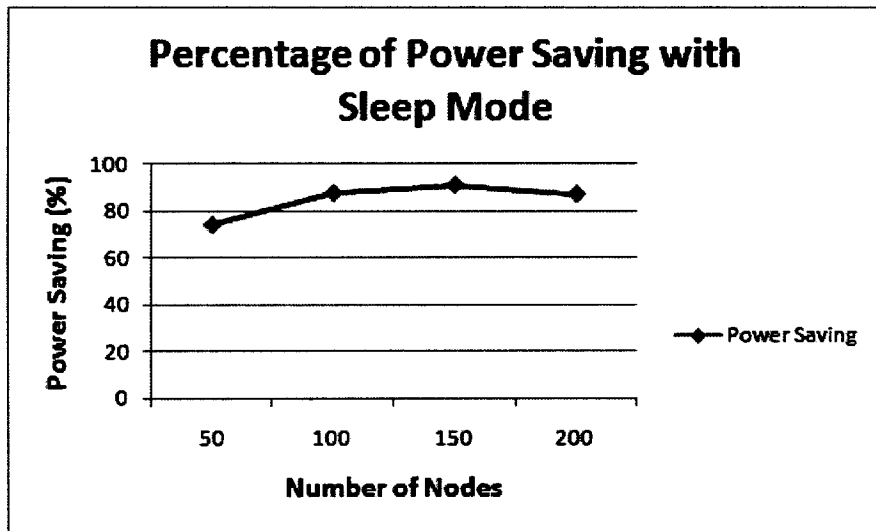
**Figure 4.2 - Network Average Power Consumption**

Figure 4.2 show us the differences between the values of the energy consumed for different network densities, proposed on two modes of operation: sleep and non-sleep. We can notice the high rate of the energy consumed in the non-sleep mode, where all the neighboring nodes of the transmitting node are consuming energy as well while listening to the transmitted packets. In the sleep mode the same process is performed but with less energy consumption, where only the transmitting nodes will lose energy. But the difference here is the variability between the two operating modes. In the sleep mode the variability of the power consumption is negligible, whereas in the non-sleep mode the variability is significant. This is mainly due to the large number of neighbors these nodes have along the path.



**Figure 4.3 - Average Lifetime for Different Network Densities**

Figure 4.3 illustrates the lifetime of the network for different densities under two operation modes: sleep and non-sleep. The results show, with the sleep mode, that as the density grows the average lifetime of the network has a value on average equals to 38 minutes. The reason behind this stability is that the larger the density the better possibility of finding another path to the destination node. Therefore, the nodes in the discovered path will still have enough energy to send the packets. Whereas in the non-sleep mode, we noticed that the larger the number of nodes occurs, the smaller the network lifetime will be achieved. Thus, if one of the chosen paths died for energy drain reason, then the remaining chosen paths will not have enough energy to continue the transmission process.



**Figure 4.4 - Percentage of Power Saving using the Sleep Mode**

Figure 4.4 shows that the average power saving across all the network densities was approximately around 85%. This value showed the high rate achieved while the network was operating on the sleep mode.

## **4.5 Network Success Rate**

### **4.5.1 Introduction**

The network success rate is expressed in this thesis as the probability of finding a path between a random source node and a destination node. In particular, we shall calculate the minimum required density of the network in which any random selected source can transmit a packet successfully to any random selected destination node. Furthermore, the success rate of the network is directly dependent on the available links between the nodes.

The purpose of this section is to find the minimum required number of nodes in a network where the RPL protocol is still optimally applicable.

### 4.5.2 Simulation Results

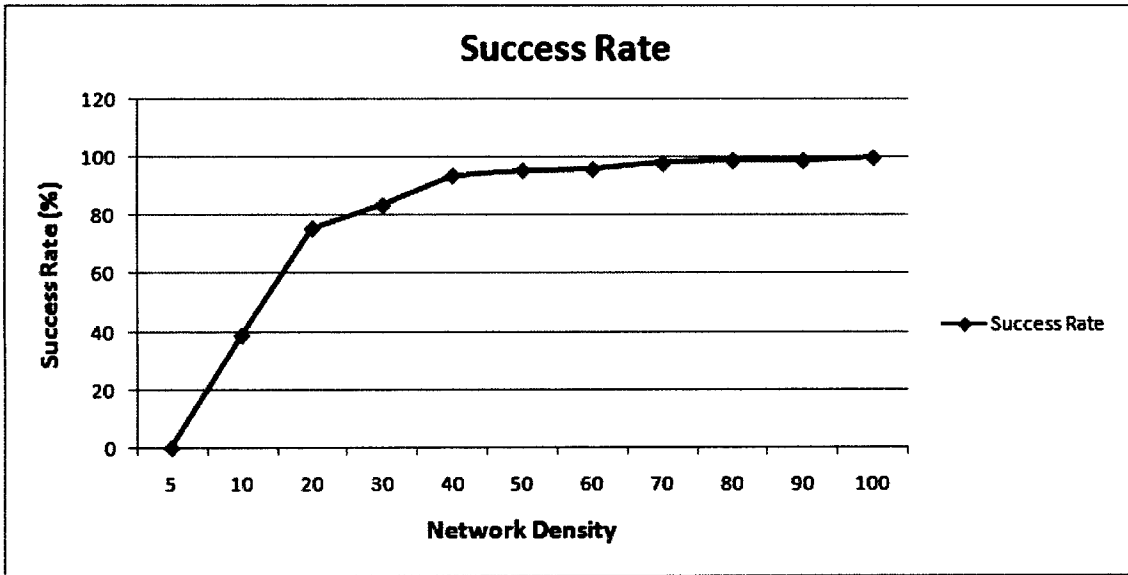


Figure 4.5 - Network Success Rate

Figure 4.5 demonstrates the generated results of the percentage of the success rate in a DTN populated on different node densities. The figure shows that with 5 nodes in the network we had unsuccessful results, while the rest of the densities the percentage of success increased till we started to notice an acceptable success ratio around 94% with the 40 nodes in the network. The smaller the network density occurs, the smaller the possibility for the nodes to have connections between each other, and hence the smaller the unsuccessful transmitting rate.

### 4.5.3 Tradeoffs Analysis

As found earlier, increasing the node density rate in the DTN would improve the success rate of the ROLL/RPL algorithm implementation. However, by increasing the network density the overall power consumption in the network would increase as well especially in the non-sleep mode of operation. In this section we shall found the node density that keeps both the success rate and the power consumption at acceptable level.

In order to find a compromise between the success rate and the power consumption we should be able to draw both variables in the same diagram. For that purpose, we propose the following new normalized power variable:

$$\bar{p} = \left(1 - \frac{\tilde{p}}{p_{max}}\right) \times 100 \tag{Eq. 4.5}$$

Where  $\tilde{p}$  is the average power consumption and  $p_{max}$  is the maximum power consumption in a given plot.

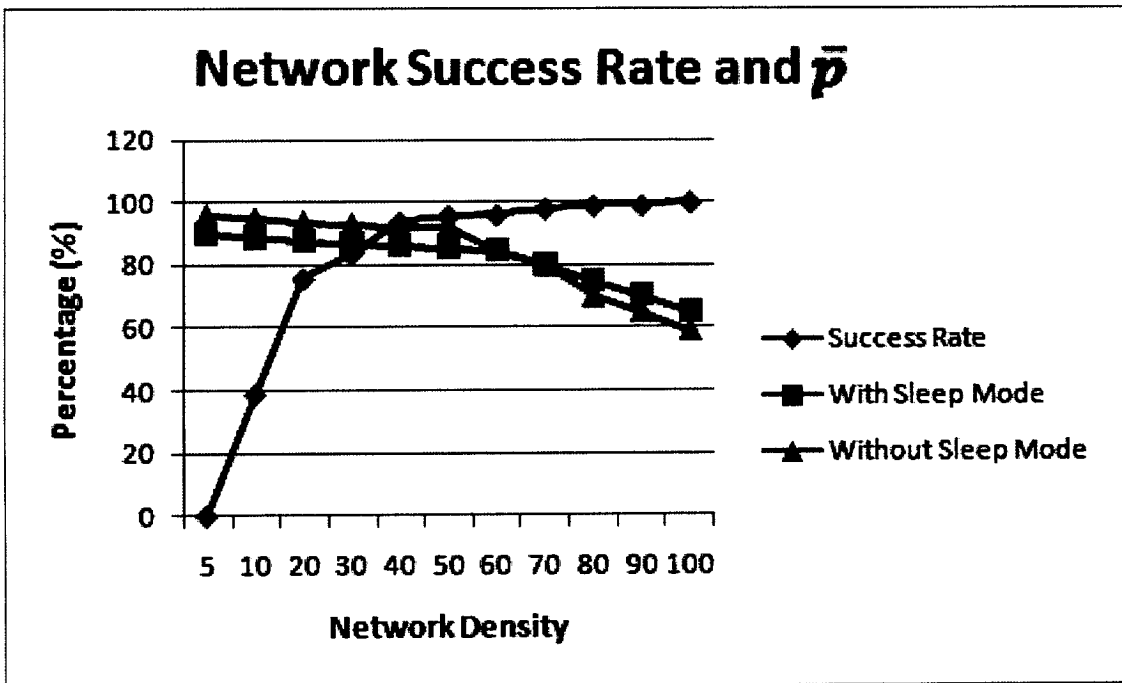


Figure 4.6 -  $\bar{p}$  and  $S$  as function of the network density

Figure 4.6 represents  $\bar{p}$  and the success rate  $S$  as function of the network density. This figure shows that the intersection point is approximately between the 40 and 50 nodes. This observation could demonstrate that the minimum balanced network density is around 40 nodes keeping in mind that this choice also includes a balanced power consumption rate between the sleep and non-sleep mode.

## 4.6 Conclusion

In this chapter, we have simulated the usage of ROLL/RPL over DTNs and calculated some major performance parameters such as the average power consumption, the network lifetime, the average power saving when using a sleep mode and the average success rate in the network.

Let us summarize now the main findings of this chapter. At first, the sleep mode is more energy efficient than the non-sleep mode, therefore the possibility of finding another optimal path through the RPL protocol is higher when using a sleep mode. Secondly, the results showed that we have less probability to apply the RPL protocol if the network is less dense. In fact, the RPL protocol needs to have a network with nodes that have many links and connections in order to be optimized. The lower the number of paths that exist between nodes, the lower is the efficiency of the RPL protocol. Finally, we noticed that the minimum required density for a DTN (with similar network characteristics) should be around 40 nodes. Obviously, having different network parameters (such as different packet transmission rate, packet sizes, bit energy consumption, etc) would lead to different optimums of network densities.



## **Chapter 5**

### **Conclusions**

#### **5.1 Introduction**

This thesis studied the possibility of using RPL protocol over DTNs architecture and simulated different aspects of these implementations. In particular, this chapter will present a summary of the studied subject, the proposed work and the simulation results.

#### **5.2 Main Contributions**

In this thesis, we studied Delay Tolerant Networks and provided the basic aspects of their architectures. Then we described the ROLL/RPL protocol by stating its primary functionalities and the flow of its routing process. We simulated the implementation of this routing protocol over DTNs taking into consideration different network performance parameters such as power consumption, traffic rate, network lifetime, link success rate, network disconnectivity, etc. Finally, we presented the generated simulation results and stated the concluding observations. The main purpose of this study was to investigate the deployment of the RPL protocol on the DTN framework taken into consideration the energy attribute as well as the density of the network. It is well known that RPL can be used over low and lossy networks (LLNs) but how good it is in DTNs scenarios? This is the question we were trying to answer in this thesis.

In fact, we took in this thesis the basic notion of the RPL protocol and by that we mean the DODAG building process, and we applied it on a DTN environment. The DTN nature is known by having a low probability when it comes to the connections between the nodes of the network. By applying the protocol on the network, we were able to calculate different important network parameters such as network lifetime, connection success rate and minimum acceptable network density of the LLN where ROLL/ RPL could be implemented successfully.

### **5.3 Suggested Future Work**

In this thesis, the implementation of the RPL protocol over a DTN was done after taking certain assumptions into consideration and neglecting certain parameters such as the bit error rate in the links. In order to have a more complete simulation, the packet error rates, the acknowledgement packets and retransmissions of the lost packets should all be taken into consideration.

Secondly, the DODAG building process should be tested more on different case scenarios for many errors could occur during the mechanism and need to be handled. One of the case scenarios that are important is the loop issue, where it has a big impact on the network. Therefore, the mentioned repair mechanism should be studied and tested in a way to optimize the renovated solution to the network. Another case scenario is the parent selection process where if a parent died for some reason the children should automatically choose another parent that is ideal for the network and for the RPL optimization plan.

Finally, other network architectures could be integrated with the RPL protocol, our thesis was intended to study the DTN structure that is a branch of the Lossy and Low power networks group. However, more network types could be tested while using the RPL protocol.

## Appendix A

### Simulation software code snippets

```
public bool RunSimulationNodes(int sourceNodeID, int destinationNodeID, IEnumerable<Node>
query, List<Node> subNodes)
{
    bool isDestinationReached = false;

    /* Base Case: If the source node has zero children,
       * Display in a message box the available DODAGs,
       * and return true to stop the recursive function.
       */

    if (this.listDODAG.Count >= 7)
    {
        return true;
    }

    if (nodes[sourceNodeID].getChildCount() == 0)
    {
        isDestinationReached = true;
    }

    return isDestinationReached;
}

//If the number of child nodes are greater than zero, meaning the current
node still have nodes attached to it:

for (int j = 0; j < query.ElementAt(0).getChildCount(); j++)
{
    Node childNode = query.ElementAt(0).getChildNode(j);

    if (childNode.getNodeID() == destinationNodeID)
    {
        subNodes.Add(childNode);
        this.fillDODAG(sourceNodeID, subNodes);
        subNodes.Remove(childNode);
    }
}

//If the destination was not reached, search inside the current node's
children recursively:

if (subNodes.Count > 0 || nodes[sourceNodeID].getChildCount() > 0)
{
    for (int j = 0; j < query.ElementAt(0).getChildCount(); j++)
    {
        Node childNode = query.ElementAt(0).getChildNode(j);

        //Not to get stuck in an infinite recursion:
        if (childNode.getNodeID() == sourceNodeID)
        {
```

```

continue;
        }

//Not to get stuck in an infinite recursion:
boolisFound = false;

for (int k = 0; k < subNodes.Count; k++)
    {
    if (subNodes[k].getNodeID() == childNode.getNodeID())
        {
        isFound = true;
        break;
        }
    }

if (isFound == true)
    {
    continue;
    }

subNodes.Add(childNode);

varsubQuery = from x in nodes
where x.getNodeID() == childNode.getNodeID()
select x;

        isDestinationReached = this.RunSimulationNodes(sourceNodeID,
        destinationNodeID, subQuery, subNodes);

        //Remove the last item in the list in case the recursion didn't
        reach a correct path.

if (isDestinationReached == false && subNodes.Count > 0)
    {
    subNodes.Remove(subNodes[subNodes.Count - 1]);
    }
else
    {
    return true;
    }

return isDestinationReached;
    }

```

```

public void RunObjectiveFunction()
{
    int nbrOfPacketsSent = 0;

    //Get maximum power value found within the paths:
    for (int listIndex = 0; listIndex < this.lastDODAG.Count; listIndex++)
    {
        string[] tempPath = this.lastDODAG[listIndex].Split('-');
        double totalPower = 0;
        int totalDistanceQuality = 0;
        string distanceDetailed = "";
        bool isNodeOffline = false;

        for (int i = 0; i < tempPath.Length; i++)
        {
            if (nodes[Convert.ToInt32(tempPath[i])].getPower() > 0)
            {
                totalPower += nodes[Convert.ToInt32(tempPath[i])].getPower();

                //Didn't reach the last element yet
                if (i < tempPath.Length - 1)
                {
                    for (int j = 0; j < nodes[Convert.ToInt32(tempPath[i])].getChildCount(); j++)
                    {
                        if (nodes[Convert.ToInt32(tempPath[i])].getChildNode(j) ==
                            nodes[Convert.ToInt32(tempPath[i + 1])])
                        {
                            //Distance between 10 and 14 has ratio equal to 1
                            if (nodes[Convert.ToInt32(tempPath[i])].getChildDistance(j) < 15)
                            {
                                totalDistanceQuality += 1;
                                distanceDetailed += "1-";
                            }
                            //Distance between 15 and 24 has ratio equal to 4
                            elseif (nodes[Convert.ToInt32(tempPath[i])].getChildDistance(j) < 25)
                            {
                                totalDistanceQuality += 4;
                                distanceDetailed += "4-";
                            }
                            //Distance between 25 and 34 has ratio equal to 9
                            elseif (nodes[Convert.ToInt32(tempPath[i])].getChildDistance(j) < 35)
                            {
                                totalDistanceQuality += 9;
                                distanceDetailed += "9-";
                            }
                            //Distance 35 and above has ratio equal to 16
                            else
                            {
                                totalDistanceQuality += 16;
                                distanceDetailed += "16-";
                            }
                        }
                    }
                }
            }
        }
        break;
    }
}

```

```

        }
    }
else
    {
this.pathPowerConsumption.Add(0);
this.pathDistance.Add(1000000);
this.pathDistanceDetailed.Add("");
isNodeOffline = true;
break;
    }
}

if (isNodeOffline == false)
    {
this.pathPowerConsumption.Add(totalPower);
this.pathDistance.Add(totalDistanceQuality);
this.pathDistanceDetailed.Add(distanceDetailed);
    }
}

doubleoverallPowerConsumption = 0;

for (int packets = 1; packets <= Convert.ToInt32(this.txtNbrOfPackets.Text);
packets++)
    {
intbestLinkDistance = this.pathDistance.Min();
intindexMinDistance = !this.pathDistance.Any() ? -1 :this.pathDistance.Select((value,
index) =>new { Value = value, Index = index }).Aggregate((a, b) => (a.Value<b.Value) ?
a : b).Index;

doublebestPower = this.pathPowerConsumption[indexMinDistance];
doublepathTotalPowerConsumption = 0;

if (bestLinkDistance != 1000000)
    {
TreeNodeTreeViewNode = newTreeNode("Packet ID : " + packets + " - Power Consumption :
" + bestPower + " - Distance : " + bestLinkDistance);
treeViewNode.Nodes.Add(this.lastDODAG[indexMinDistance]);
this.trvAvailablePaths.Nodes.Add(treeViewNode);

//Calculate the power consumption of the selected path.
//Must use the formula of the power consumption.
string[] bestPath = this.lastDODAG[indexMinDistance].Split('-');
string[] detailedPath = this.pathDistanceDetailed[indexMinDistance].Split('-');

for (int i = 0; i <bestPath.Count() - 1; i++)
    {
doubletransmissionEnergyResult = 0;
doublereceivingEnergyResult = 0;

//Transmitting Node
if (nodes[Convert.ToInt32(bestPath[i])].getPower() > 0)
        {

```

```

transmissionEnergyResult = packetSizeL * (energyPerBitE + (energyAmplification *
Convert.ToInt32(detailedPath[i])));
receivingEnergyResult = packetSizeL * energyPerBitE;

nodes[Convert.ToInt32(bestPath[i])].setPower(nodes[Convert.ToInt32(bestPath[i])].getPo
wer() - transmissionEnergyResult);
pathTotalPowerConsumption += transmissionEnergyResult;

if (this.chkIsSleepMode.Checked == false)
{
foreach (NodechildNodein nodes[Convert.ToInt32(bestPath[i])].childNodes)
{
//If the child node is different from the receiving node, set the new power.
if (childNode != nodes[Convert.ToInt32(bestPath[i + 1])])
{
childNode.setPower(childNode.getPower() - receivingEnergyResult);
pathTotalPowerConsumption += receivingEnergyResult;
}
}
}
//Receiving Node
if (nodes[Convert.ToInt32(bestPath[i + 1])].getPower() > 0)
{
receivingEnergyResult = packetSizeL * energyPerBitE;
nodes[Convert.ToInt32(bestPath[i + 1])].setPower(nodes[Convert.ToInt32(bestPath[i +
1])].getPower() - receivingEnergyResult);
pathTotalPowerConsumption += receivingEnergyResult;

if (this.chkIsSleepMode.Checked == false)
{
foreach (NodechildNodein nodes[Convert.ToInt32(bestPath[i + 1])].childNodes)
{
//If the child node is different from the transmitting node, set the new power.
if (childNode != nodes[Convert.ToInt32(bestPath[i])])
{
childNode.setPower(childNode.getPower() - receivingEnergyResult);
pathTotalPowerConsumption += receivingEnergyResult;
}
}
}
}
}
else
{
TreeNodetreeViewNode = newTreeNode("Packet ID : " + packets + " - Power Consumption :
" + bestPower + " - Distance : " + 0);
this.trvAvailablePaths.Nodes.Add(treeViewNode);
}

if (bestPower > 0)
nbrOfPacketsSent++;

```

```
this.pathPowerConsumption[indexMinDistance] -= pathTotalPowerConsumption;  
overallPowerConsumption += pathTotalPowerConsumption;  
}
```

```
this.pathPowerConsumption.Clear();  
this.pathDistance.Clear();  
this.pathDistanceDetailed.Clear();
```

```
this.lblSentPackets.Text = "Number of Packets: " + nbrOfPacketsSent.ToString();  
this.lblTotalPowerConsumption.Text = "Total Power Consumption: " +  
Math.Round(overallPowerConsumption, 2).ToString();
```

```
this.SaveResultsToFile();  
}
```



## Bibliography

1. *A Wireless Sensor, AdHoc and Delay Tolerant Network System for Disaster Response*. **H. Chenji, A. Hassanzadeh, M. Won, Y. Li, W. Zhang, X. Yang, R. Stoleru, G. Zhou**. Texas : Department of Computer Science and Engineering, Texas A&M Univeristy, 2011.
2. *Dealy Tolerant Networks*. **G., Volodymyr**. s.l. : Albert-Ludwigs-Universität Freiburg, 2010.
3. *A Mobile Delay-tolerant Approach to Long-term Energy-efficient Underwater Sensor Networking*. **Eugenio M., Jiejun K., Uichin L., Mario G., Paolo B., Antonio C.** Bologna : Wireless Communications and Networking Conference, IEEE, 2007.
4. *Routing Over Low Power and Lossy Networks (ROLL)*. The Internet Engineering Task Force (IETF). [Online]
5. *Delay Tolerant Networking for Sensor Networks*. **Loubser, Max**. Sweden : Swedish Institute of Computer Science, 2006.
6. *A Survey of Routing Protocols and Simulations in Delay Tolerant Networks*. **Mengjuan Liu, Yan Yang, and Zhiguang Qin**. [ed.] LNCS 6843 Y. Cheng et al. (Eds.): WASA 2011. s.l. : © Springer-Verlag Berlin Heidelberg 2011, 2011, pp. pp. 243–253.
7. *Delay Tolerant Networks (DTNs): A Tutorial Challenges and Applications of Delay Tolerant Networks*. **Choi, David (Bong Jun)**. 2009.
8. *Max-Contribution: On Optimal Resource Allocation in Delay Tolerant Networks*. **Kyunghan L., Yung Y., Jaeseong J., Hyungsuk W., Injong R., Song C**. s.l. : IEEE, 2010.
9. *Home Automation Routing Requirements in Low Power and Lossy Networks*. **A. Brandt, J. Buron, G. Porcu**. Internet Engineering Task Force (IETF). 2009.
10. *A Delay-Tolerant Network Architecture for Challenged Internets*. **F., Kevin**. Berkeley : Intel Research, 2003.
11. *Power Management in Delay Tolerant Networks: A Framework and Knowledge-Based Mechanisms*. **Hyewon J., Moustafa H. A., Ellen W. Z.** Atlanta, Georgia : Georgia Institute of Technology, College of Computing.
12. *Network Energy Saving Technologies for Green Wireless Access Networks*. **Tao C., Yang Y., Honggang Z., Haesik K., Kari H.** Finland : IEEE, 2011.
13. *Energy savings in mobile broadband network based on load predicitions: opportunities and potentials*. **Saulius S., Torben B.P., Troels B.S., Gilbert M**. s.l. : IEEE, 2012.
14. *Power Saving Trade-offs in Delay/Disruptive Tolerant Networks*. **Oscar T., Julian M., Jose M., Jorge G**. s.l. : IEEE, 2011.

15. *Routing Metrics used for Path Calculation in Low Power and Lossy Networks*. **JP. Vasseur, M. Kim, K. Pister, H. Chong**. The Internet Engineering Task Force (IETF). [Online] December 3, 2009. <http://www.ietf.org/>.
16. *Building Automation Routing Requirements in Low Power and Lossy Networks*. **Martocci J., De Mil P., Vermeylen W., Riou N.** The Internet Engineering Task Force (IETF). [Online] January 28, 2010. <http://www.ietf.org/>.
17. *RPL: The IP routing protocol designed for low power and lossy networks*. V. **JP, A. Navneet, H. Jonathan, S. Zach, B. Paul, C. Cedric**. April 2011, IPSO Alliance, p. 20.
18. *RPL: IPv6 Routing Protocol for Low power and Lossy Networks*. **Winter T., Thubert P. Vasseur JP**. The Internet Engineering Task Force (IETF). [Online] March 13, 2011. <http://www.ietf.org/>.
19. *A high-throughput path metric for multi-hop wireless routing*. **Bicket J, De Couto D, Morris R, Aguayo D**. San Diego, California, USA. : ACM, 2003. pp. 134-146.
20. *The Emergence of a Networking Primitive in Wireless Sensor Networks*. **Levis P., Brewer E., Culler D., Gay D., Madden S., Patel N., Polastre J., Shenker S., Szewczyk R., Woo A.** 7, July 2008, Communications of the ACM, Vol. 51.
21. *The Trickle Algorithm*. **Levis P., Clausen T., Hui J., Gnawali O., Ko** . The Internet Engineering Task Force (IETF). [Online] January 10, 2011. <http://www.ietf.org/>.
22. *A Performance Evaluation Study of RPL: Routing Protocol for Low Power and Lossy Networks*. **J. Tripathi, J. C. de Oliveira, J. P. Vasseur**. Philadelphia : IEEE, 2010.
23. *Performance Analysis of the RPL Routing Protocol*. **N. Accettura, L. A. Grieco, G. Boggia, P. Camarda**. Istanbul, Turkey : IEEE, 2011.
24. *Simulation and Performance Evaluation of DAG Construction with RPL*. **Oifa G., Anis K., Shafique C., Miled T., Rihab C., Mohamed A.** Tunisia, Saudi Arabia, Portugal : CES Research Unit-National School of Engineers of Sfax , COINS Research Group-Al\_Imam Mohamed bin Saud University , CISTER Research Unit-Polytechnic of Porto.
25. *A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)*. **Thomas C., Ulrich H., Matthias P.** s.l. : IEEE, 2011.
26. *The P2P-RPL Routing Protocol for IPv6 Sensor Networks: Testbed Experiments*. **Emmanuel B., Matthias P., Mukul G.** France, USA : IEEE, 2011.
27. *Performance Optimization in Structured Wireless Sensor Networks*. **Amine M., Hoda M.** 5, Lebanon : The International Arab Journal of Information Technology, 2009, Vol. 6.