GOA COLLEGE OF ENGINEERING
FARMAGUDI, GOA
DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

2019 - 2020

Design of optimised IoT networks using contiki os and Tmote sky motes

by

Anirudha Chari (P.R.No.:201704585)
Ruta Kalangutkar (P.R.No.:201704566)
Sushanti Oli (P.R.No.:201711487)
Aaron Fernandes (P.R.No.:201704582)

A project submitted
in partial fulfilment of the requirements
for the degree of
Bachelor of Engineering
in
Electronics & Telecommunication Engineering
GOA UNIVERSITY

under the guidance of

Prof. Sonia Kuwelkar
Assistant Professor,
Electronics & Telecommunication Department
Goa College of Engineering

CERTIFICATE

This is to certify that the project entitled

"Design of optimised IoT networks using contiki OS and Tmote sky motes"

submitted by

Anirudha Chari	**P.R. No.:201704585**
Ruta Kalangutkar	**P.R. No.:201704566**
Sushanti Oli	**P.R. No.:201711487**
Aaron Fernandes	**P.R. No.:201704582**

has been successfully completed in the academic year 2019-2020 as a partial fulfilment of the requirement for the degree of BACHELOR OF ENGINEERING in Electronics & Telecommunication Department, at Goa College of Engineering, Farmagudi.

Internal Examiner
(Prof. Sonia Kuwelkar)

External Examiner

Head of Department,
Dr. H. G. Virani,
Professor, ETC Dept.

Place: Farmagudi, Ponda, Goa
Date:

PROJECT APPROVAL SHEET

The project entitled

"DESIGN OF OPTIMISED IOT NETWORKS USING CONTIKI OS AND TMOTE SKY MOTE"

by

Anirudha Chari	**P.R. No.:201704585**
Ruta Kalangutkar	**P.R. No.:201704566**
Sushanti Oli	**P.R. No.:201711487**
Aaron fernandes	**P.R. No.:201704582**

completed in the year 2019-2020 is approved as a partial fulfilment of the requirements for the degree of **BACHELOR OF ENGINEERING in Electronics & Telecommunication Engineering** and is a record of bonafide work carried out successfully under our guidance.

Project Guide,
Sonia Kuwelkar
Assistant Professor,
ETC Dept.

Head of Department
Dr. H. G. Virani
Professor, ETC Dept.

Principal
Dr. Krupashankara M. S.
Goa College of Engineering

Place: Farmagudi, Ponda, Goa
Date:

Declaration

We declare that the project work entitled "Design of Optimised IoT Networks using contiki os and Tmote sky mote" submitted to Goa College of Engineering, in partial fulfillment of the requirement for the award of the degree of B.E. in Electronics and Telecommunication Engineering is a record of bonafide project work carried out by us under the guidance of Prof. Sonia Kuwelkar. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

(Anirudha Chari)
P.R. No.:201704585

(Ruta Kalangutkar)
P.R. No.:201704566

(Sushanti Oli)
P.R. No.:201711487

(Aaron Fernandes)
P.R. No.:201704582

Date:

Acknowledgement

The success of our work is incomplete unless we mention the names of our respected teachers who made it possible, whose guidance and encouragement served to beacon light and crowned our efforts with success.

First, I wish to express my sincere gratitude to our guide, Prof. Sonia Kuvelkar, for her patience, and the constant motivation to achieve our goals. Without her support and guidance, this project would not have been possible.

We would also like to extend our gratitude to the Head of the Department, Dr. H.G.Virani for providing us with the required facilities. We wish to acknowledge the help provided by each and every staff member of the Electronics and Telecommunication Engineering Department. I am grateful to all of those with whom I have had the pleasure of working during this projects. Lastly we thank the almighty for keeping us in good health throughout the course of the project and showing us the light whenever needed

Contents

List of Figures

List of Tables

Abstract

Low-Power and Lossy Networks (LLNs) are a class of networks in which both the routers and their interconnects are constrained. LLN routers typically operate with constraints on processing power, memory, and energy (battery power). Their interconnects are characterized by high loss rates, low data rates, and instability. Thus these networks use routing protocols such as the Routing Protocol for Low-Power and Lossy Networks (RPL). RPL is a distance vector routing protocol which uses rank of a node to decide network routes, which is calculated using an objective function. This project proposes a new objective function with a combined metric calculated using Fuzzy Inference to dictate the rank of a node. We propose a combined metric Quality based on ETX, Delay, Hop count and Energy in order to select the best path.

Chapter 1

Introduction

1.1 Preamble

Internet of Things (IoT) refers to the network of physical objects having the ability to communicate with each other in an ubiquitous way through different technologies. They are used to ensure communication services in many application scenarios such as healthcare, industrial automation, and smart homes. However, sensor nodes are small and battery powered. Thus, changing their batteries is a very challenging task. Low power and Lossy Network (LLN) possess two key features: the limited resources of nodes and the lossy links between them. By considering these specific characteristics, Routing Over Low power and Lossy networks (ROLL) working group has initially proposed the IPv6 standard routing protocol for low power and lossy networks (RPL).

second para

1.2 Motivation

This is how you create bullets:

- Standard routing protocols like OSPF are inefficient for low power lossy networks (LLNs) due to the constraints and fuzzy nature
- Using just one metric for the objective function makes the routing protocol unable to accommodate application requirements many a times
- Some kind of metric combination is required. In this project we use Fuzzy logic to combine ETX, PDR, Hop count and Delay to a metric called Qos(Quality of service).

1.3 Outline

The IPv6 Routing Protocol for Low power and Lossy Networks (RPL) is the standard IPv6 based routing protocol for Low Power, Lossy Networks (LLNs) which consist largely of constrained nodes with limited power, memory, and processing resources like WSNs(Wireless Sensor Networks).RPL uses Objective functions(OFs) to build networks according to these constraints.

RPL supports traffic flows like:

- Multipoint-to-poitn - the dominant traffic pattern
- Point-to-multipoint - using destination advertisement mechanism
- Point-to-Point

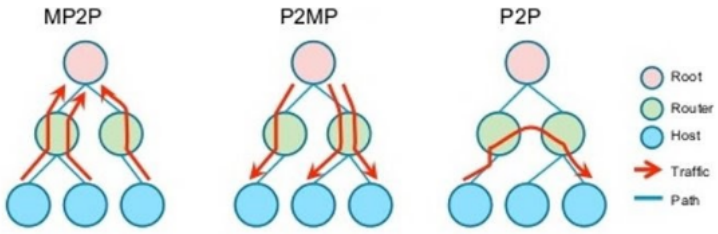


Figure 1.1: traffic flow in rpl

1.3.1 Topology

RPL organizes a network topology as a Directed Acyclic Graph (DAG) that is partitioned into one or more Destination Oriented DAGs (DODAGs), one DODAG per sink(root).

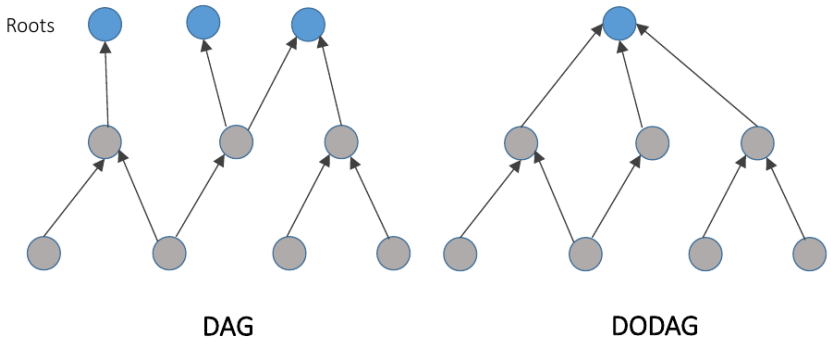


Figure 1.2: Topology in RPL

- The topology building starts at the root (initially, the only router that is part of the dodag). It sends DIO messages in its neighbourhood(This message contains all common communication parameters, including root ID, modes of operation, timer values etc.)Upon receipt of a number of such messages, neighbour nodes may participate in the DODAG according to the objective function (OF), select their parents and then start emitting their own DIO messages.This process spreads gradually to cover the whole network as new nodes join the DODAG.

- Prefered Parent- Only one node among parent's nodes (the preferred parent)acts as the nexthop on the path towards theroot

- RPL pro-actively creates and maintains the topology, by regularly sending ICMP control messages in the vicinity. The frequency of these exchanges are governed by the Trickle algorithm.

1.3.2 Important terminology

RPLInstanceID: A RPLInstanceID identifies a set of one or more Destination Oriented DAGs (DODAGs). A network may have multiple RPLInstanceIDs, each of which defines an independent set of DODAGs, which may be optimized for different Objective Functions (OFs) and/or applications. The set of DODAGs identified by a RPLInstanceID is called a RPL Instance. All DODAGs in the same RPL Instance use the same OF.

Rank: Rank defines individual node positions with respect to the DODAG root.

DODAGVersionNumber: A DODAG version is a specific iteration of a DODAG with a given DODAG ID. DODAGVersionNumber is a sequential counter incremented by the Root to form a new DODAG Version.

DODAGID: The combination of RPLInstanceID and DODAGID uniquely identifies a single DODAG in the network. A RPL Instance may have multiple DODAGs, each of which has an unique DODAGID.

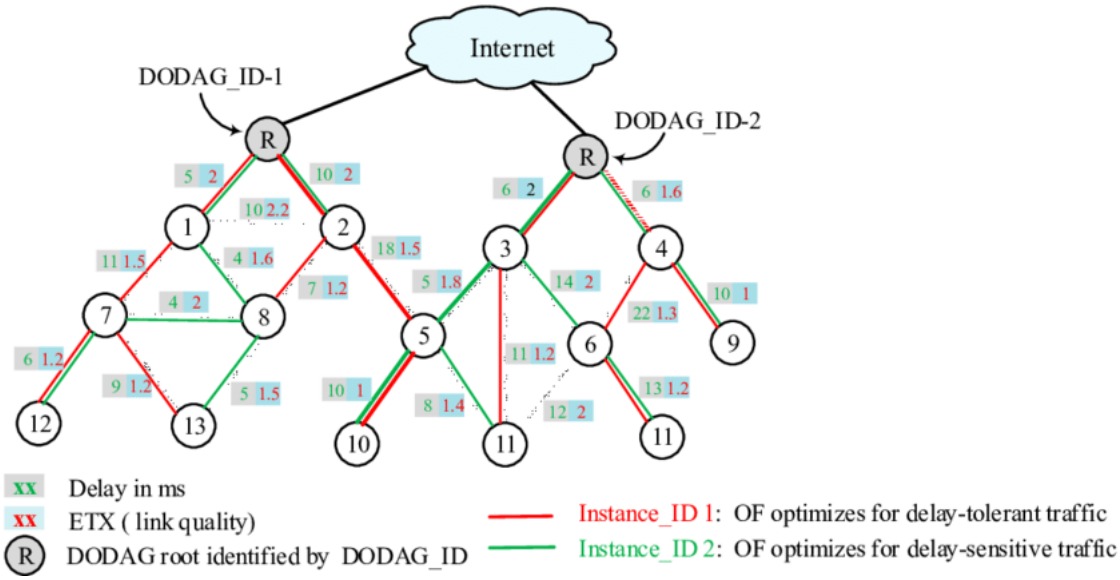


Figure 1.3: Terminologies used in RPL

1.3.3 ICMPv6 RPL control message

A RPL control message is identified by a code and composed of a base that depends on the code (and a series of options).

In accordance with [RFC4443], the RPL Control Message consists of an ICMPv6

header followed by a message body. The message body is comprised of a message base and possibly a number of options as illustrated in Figure 1.4.

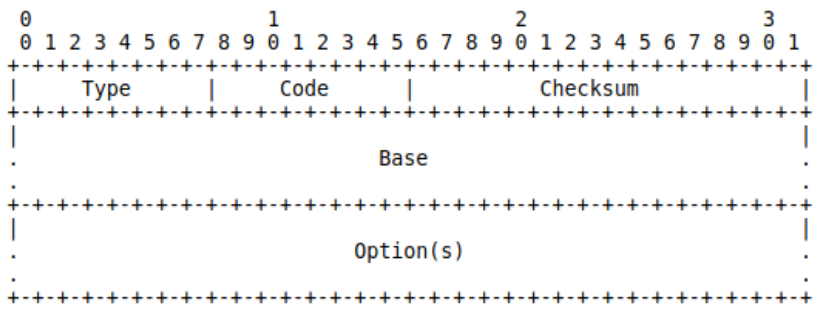


Figure 1.4: RPL control message

DODAG Information Solicitation (DIS)

The DODAG Information Solicitation (DIS) message may be used to solicit a DODAG Information Object from a RPL node. Its use is analogous to that of a Router Solicitation as specified in IPv6 Neighbor Discovery; a node may use DIS to probe its neighborhood for nearby DODAGs.

DODAG Information Object(DIO)

The DODAG Information Object carries information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG.

The Destination Advertisement Object (DAO)

The DAO is used to propagate destination information Upward along the DODAG. In Storing mode, the DAO message is unicast by the child to the selected parent(s). In Non-Storing mode, the DAO message is unicast to the DODAG root. The DAO message may optionally, upon explicit request or error, be acknowledged by

its destination with a Destination Advertisement Acknowledgement (DAO-ACK) message back to the sender of the DAO.

DAO Acknowledgement

The DAO-ACK message is sent as a unicast packet by a DAO recipient (a DAO parent or DODAG root) in response to a unicast DAO message.

Chapter 2

Literature Survey

2.0.1 Papers Reviewed

A paper proposed by Patrick Olivier Kamgueu et. al focuses on the analysis of combining several metrics criteria for the implementation of RPL objective function, the new routing standard for the Internet of Things. The general problem is known as NPcomplete, we propose the use of fuzzy inference system for finding a good trade-off among the various chosen metrics. Many routing solutions tend to favour increase on network lifetime, neglecting some other network performance aspects. In this work, they consider : the expected number of transmission needed to successfully send a packet to its final destination, to meet reliability; the latency, to minimize end-to-end delay; in addition to the remaining power draw by node, for network lifetime extension. Implementation was done on Contiki and simulations were carried out on its emulator Cooja. Obtained results show im-

provements compared with those from the most common implementation, namely the one that uses ETX as unique routing metric[1].

The paper proposed by Adeeb Saaidah et. al discusses the nature of the Low power and lossy networks (LLNs) requires having efficient protocols capable of handling the resource constraints. LLNs consist of networks that connect different type of devices which has constraints resources such as energy, memory and battery life. Using the standard routing protocols such as Open Shortest Path First (OSPF) is inefficient for LLNs due to the constraints that LLNs need. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) was developed to accommodate these constraints. RPL is a distance vector protocol that uses the object functions (OF) to define the best tree path. Choosing a single metric for the OF found to be unable to accommodate applications requirements. In this paper, an enhanced (OF) is proposed namely; OFRRT-FUZZY relying on several metrics combined using Fuzzy Logic. In order to overcome the limitations of using a single metric, the proposed OFRRT-FUZZY considers node and link metrics. Namely, Received Signal Strength Indicator (RSSI), Remaining Energy (RE) and Throughput (TH). The proposed OFRRT-FUZZY is implemented under Cooja simulator and then results were compared with OF0, MHROF in order to find which OF provides more satisfactory results. And simulation results show that OFRRT-FUZZY outperformed OF0 and MHROF[2].

The paper proposed by Muneer Bani et. al describes how recently, IETF standard-

ised a powerful and flexible Routing Protocol for Low Power and Lossy Networks (RPL). It selects the ideal routes from a source to a destination node based on certain metrics injected into the Objective Function (OF). In this study, the performance of RPL has been investigated in terms of two OFs (i.e. Minimum Rank with Hysteresis Objective Function (MRHOF) and Objective Function Zero (OF0)) in various topologies (grid, random) which makes this work distinctive. To study the RPL OFs performance, various parameters are considered Packet Delivery Ratio (PDR), Power Consumption and RX. The evaluation has been conducted based on these parameters (RX, topology) and compared for both OFs. The simulation results revealed that these parameters have a great impact on the PDR and achieved saved energy levels in the given networks. Our results have also indicated that the performance of RPL within light density networks for MRHOF can provide a better RPL behavior that OF0 could not provide[3].

The paper proposed by Hanane Lamaazi RPL routing protocol is designed to respond to the requirements of a large range of Low-power and Lossy Networks (LLNs). RPL uses an objective function (OF) to build the route toward a destination based on routing metrics. Considering only a single metric, some network performances can be improved while others may be degraded. In this paper, we present a flexible Objective Function based on Expected Transmission Count (ETX), Consumed Energy and Forwarding Delay (OF-ECF) built on a combination of metrics using an additive method. The main goal of this proposed solution is to balance energy consumption and minimize the average delay. To improve the

reliability of the network, a flexible routing scheme that provides the diversity of paths and a higher availability is presented. Simulation results show that the new objective function OF-ECF outperforms the OF-FUZZY, and the standards OF0 and MRHOF. In terms of network lifetime and reliability[4].

The paper proposed by Tayyab Mehbood demonstrates the scheme regarding Internet of Things (IOT) which is well thought-out the next generation of Internet. IOT explicitly elaborates the assimilation of human beings and physical systems, as they can cooperate with each other so leading towards a sort of encroachment in networking by interconnecting things together while making use of wireless embedded systems, said to be the building blocks of IOT, that are capable to be given an IP address and thus making them part of the global internet. Several essential approaches that entail in IOT and supports this innovation are being argued in this paper. 6LoWPAN (IPV6 Low Power Personal Area Networks) is a protocol used to appropriately and efficiently use IPV6 addresses. Control messages of RPL routing protocol for low power devices are discussed to understand the working of RPL protocol. In the end Contiki OS based COOJA Network simulator is used to demonstrate the working of how these routing and compression protocol works in real time simulation[5].

Chapter 3

Project Objectives

3.1 Project Objectives

Following is the list of objectives that we wish to achieve in this project:

- To design and implement an Fuzzy optimised Objective function for IPV6 routing protocol for low power and lossy networks(RPL) by combining metrics such as energy, delay, hopcount and ETX for the preferred parent selection decision.
- Optimising an IOT network hence improving the exchange of data and information among various sensor devices in the physical environment
- Comparative performance evaluation of the present objective functions (OF0 and MRHOF) and the optimized objective function.

3.2 Project Methodology

Carrying out test simulations for different number of motes using the MRHOF and OF0 objective functions and comparing their performance

- We are going to consider a network topology having 70, 90, 100 motes respectively. Network simulations are carried out using the mrhof and of0 objective function respectively.The Minimum Rank with Hysteresis Objective Function, (MRHOF) is designed to find the paths with the smallest path cost whereas the objective function zero (OF0) is based on the hop count. The implementation of MRHOF is suitable for use in sensor networks that require data delivery in the reliable network. While OF0 is suitable for use in sensor network that require fast formation network link and low power consumption

Designing a fuzzy controller using matlab

- The Fuzzy Logic Controller block implements a fuzzy inference system (FIS) in Simulink®. You specify the FIS to evaluate using the FIS name parameter.
- We are using the 'Mamdani' model of the FIS.
- Input and output membership functions are defined
- We are implementing it in 2 stages

- If-then rules are defined.
- Simulink is used to connect the 2 stages and give the final defuzzified output.Simulink, an add-on product to MATLAB, provides an interactive, graphical environment for modeling, simulating, and analyzing of dynamic systems

Designing the new objective function which optimizes the iot network using fuzzy logic:

- Objective function is written in C language

Carrying out test simulations using the new objective function:

- Using the same topology of 70,90,100 motes , the new objective function will be Evaluated and performance will be compared with the performance obtained using mrhof and of0 objective functions.

Carrying out implementation using motes and evaluating data transfer between the motes

- Physical motes such as TelosB will be used to check the actual data transfer between the motes.

Applying the optimised network in the physical environment

- The optimised IOT network will enhance the data transfer between the devices connected in the network

- IOT network has applications in many fields such as process automation, home automation, smart cars, decision analytics, and smart grids.

Chapter 4

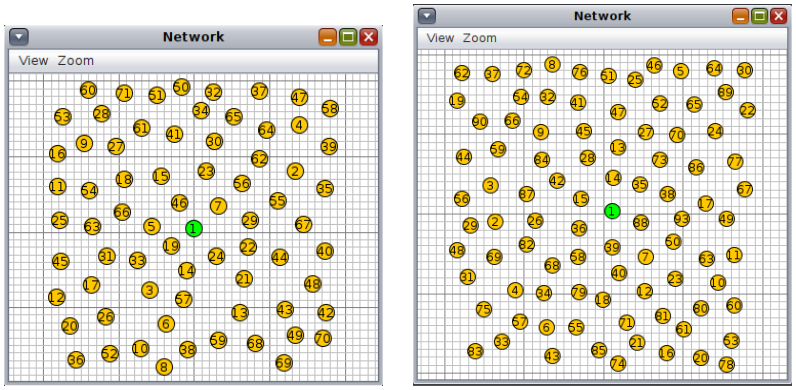
Performance Evaluation

4.1 Simulations Environment

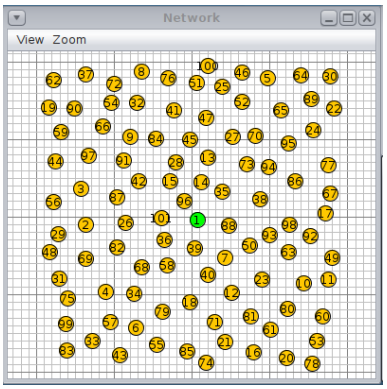
A set of simulations were performed for evaluating the performance levels of existing Objective functions namely OF0 and mrhof. The simulations were performed under Contiki OS 2.6 which provides a network simulator called COOJA which permits the emulation of real hardware platforms. COOJA is the application of Contiki OS concentrating on network behavior. It is capable of simulating wireless sensor networks without any particular mote. Cooja supports the following set of standards; TR 1100, TI CC2420, Contiki-RPL, IEEE 802.15.4, uIPv6 stack and uIPv4 stack.[3] There are 4 propagation models in COOJA simulator which must be selected for a simulation namely unit disk graph medium (UDGM): Distance loss, UDGM: Constant loss, Directed graph radio medium (DGRM), no radio traf-

fic, multipath ray-tracer medium (MRM).

The simulations were conducted for 70, 90 and 100 motes using of0 and mrhof. All simulations were conducted using UDGM and same random seed for the same number of motes for Tmote sky motes. The performances of the two OFs were



(a) random topology for 70 motes (b) random topology for 90 motes



(c) random topology for 100 motes

Figure 4.1: simulation of 70, 90 and 100 motes

compared based on the following criteria:

- Average latency: It refers to the average time a transmitted packet consumes from sender node to sink node. The following equation can be used for calculating it:

$$AverageLatency = \frac{\Sigma_{k=1}^{n} recv\ time - send\ time}{total\ packets\ recieved}$$

- Packet delivery ratio (PDR): It refers to the ratio of nodes' number of received and sent packets. The following equation can be used for calculating it:

  $PacketDeliveryRatio = \frac{total\ recieved\ packets}{total\ sent\ packets}$

- Energy consumption (mJ): Energy anode requires to exchanges data through the network between nodes. The following equation can be used for calculating it:

  $EnergyConsumption = \frac{(transmit*19.5mA\ +\ Listen*21.5mA\ +\ CPU_time*1.8mA+LPM*0.0454mA)*3V}{32768}$

- Control traffic overhead: Represents the total number of control messages DIO, DAO & DIS used by ICMPv6, it is calculated based on the following equation.

  $ControlTrafficOverhead = \Sigma_1^n DIO\ +\ \Sigma_1^n DIS\ +\ \Sigma_1^n DAO$

4.1.1 Results of simulation

The results obtained from the simulation were plotted using a python script to compare the two obejctive functions

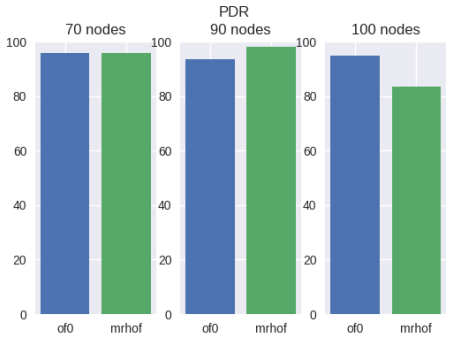


Figure 4.2: Simulation plot for PDR

PDR: Power delivery ratio is seen to be almost equal for less dense network but is degraded to 83% in case of Mrhof for a network of 100 nodes.

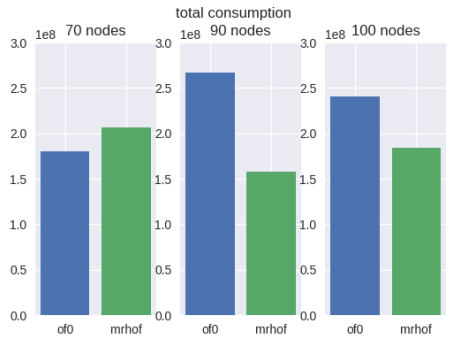


Figure 4.3: Simulation Plot for Energy

ENERGY: From this graph we can observe that mrhof performs better in terms of energy consumption for a large density of nodes

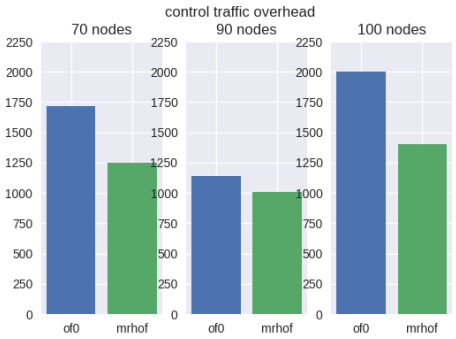


Figure 4.4: Simulation Plot for Energy

Control Traffic Overhead: From this graph we can confirm that mrhof performs significantly better in terms of traffic overhead for large density networks

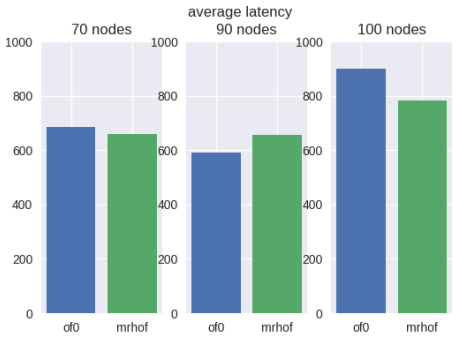


Figure 4.5: Simulation Plot for Energy

Average Latency: Here we observe that mrhof has lower average latency as compared to OF0 for large networks

Chapter 5

Design and Implementation

5.1 Conceptual Design

5.1.1 Fuzzy Inference System Design

A fuzzy inference system (FIS) allows the use of fuzzy set theory to map inputs (etx, delay, hopcount, energy) to outputs (Quality). Due to it's simplicity in implementation we use the mamdani FIS.

A FIS can essentially be broken down into the following steps:

- Fuzzification: take a crisp value input and determine its degree of membership (fuzziness) for the appropriate fuzzy sets.
- Fuzzy inference: Apply combination rules to fuzzified inputs and compute a

fuzzy output.

- Aggregation: If an output depends on more than one rule, this step unifies all values into one.
- Defuzzification: Convert the fuzzy output obtained at the previous step into a crisp value.

5.1.1.1 Linguistic variables

Node's performances knowledge are represented as linguistic variables:

- ETX - The expected number of required transmissions before a packet reaches the destination.
- Delay - The average time for a packet to reach its destination.
- Energy - The energy cost of the path, also energy of the node having the smallest remaining battery level on the path.

5.1.1.2 Fuzzification process

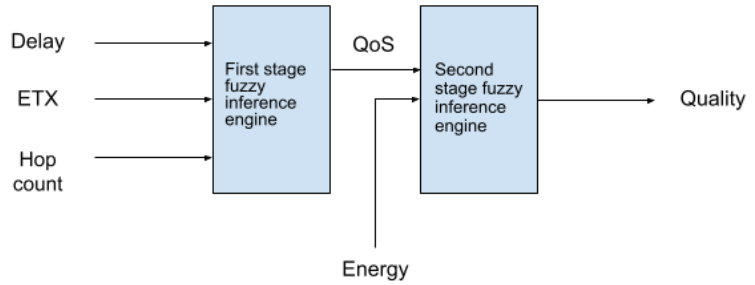
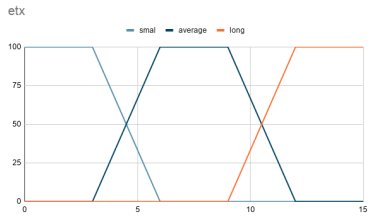
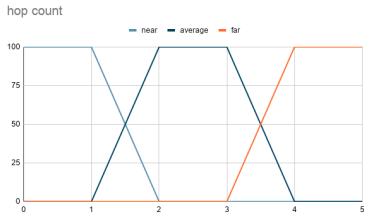


Figure 5.1: fuzzy inference engine

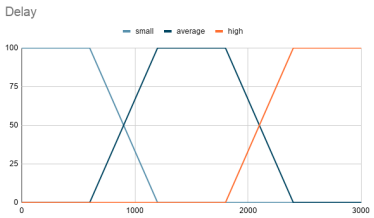
The fuzzification process is done in two stages to simplify the implementation. **Stage 1:** The first stage combines etx, delay and hopcount into a compound metric Quality of Service (QoS). We use three membership functions for each linguistic variable, etx is divided into *short*, *average*, *long*; delay is divided into *small*, *average*, *high*; and finally hopcount is divided into *near*, *average* and *far*. Figure 4.3 shows the membership functions for the 3 input parameters. Table



(a) etx membership functions



(b) hop count membership functions



(c) delay membership functions

Figure 5.2: membership functions for first stage of fuzzification

4.1 shows the relationship between the three input linguistic variables for the computation of QoS.

Stage 2: We are combining the result obtained in the first stage (QOS) with the final metric to be considered i.e. 'energy'. We are using trapezoidal membership function to represent energy . Energy is divided into low,medium and full subsections. The output of this stage is what we require that is 'quality'. Quality of every parent is compared and the best parent is selected based on quality.

etx	delay	hop count	qos
short	small	near	very fast
short	average	near	fast
short	high	near	average
short	small	average	fast
short	average	average	fast
short	high	average	average
short	small	far	fast
short	high	far	fast
short	high	far	average
average	small	near	fast
average	average	near	average
average	high	near	slow
average	small	far	fast
average	average	far	average
average	high	far	slow
long	small	near	average
long	average	near	slow
long	high	near	slow
long	small	average	average
long	average	average	slow
long	high	average	slow
long	small	far	average
long	average	far	slow
long	high	far	very slow

Table 5.1: QoS Output Metric

Table 4.2 shows how the output metric (quality) is decided based on the linguistic variables.

Figure 4.4 shows membership functions for Energy

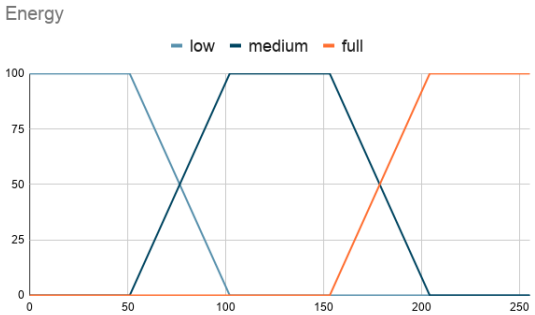


Figure 5.3: membership function for Energy metric

Qos / energy	low	medium	full
very slow	awful	bad	average
slow	bad	degraded	average
average	degraded	average	acceptable
fast	average	acceptable	good
very fast	avrerage	good	excellent

Table 5.2: Quality Output Metric

5.2 Detailed Design

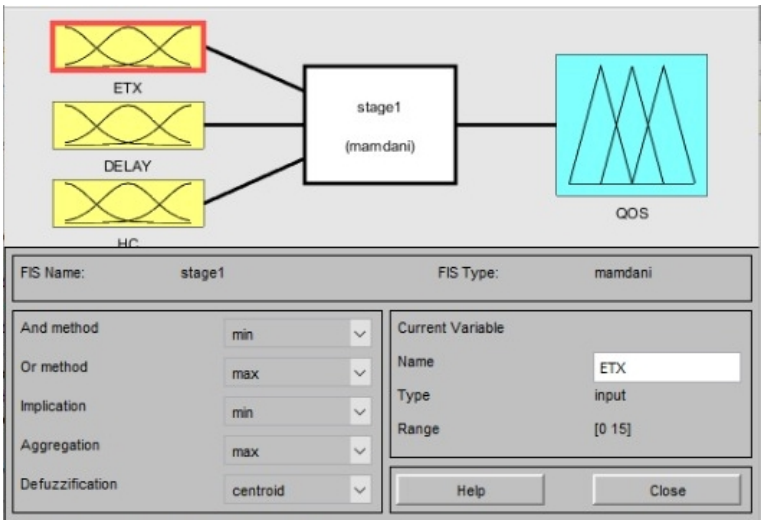
5.2.1 Implementation of Fuzzy Logic Controller in MATLAB

Fuzzy Logic Toolbox™ provides MATLAB® functions, apps, and a Simulink® block for analyzing, designing, and simulating systems based on fuzzy logic. The product guides you through the steps of designing fuzzy inference systems. Functions are provided for many common methods, including fuzzy clustering and adaptive neurofuzzy learning.

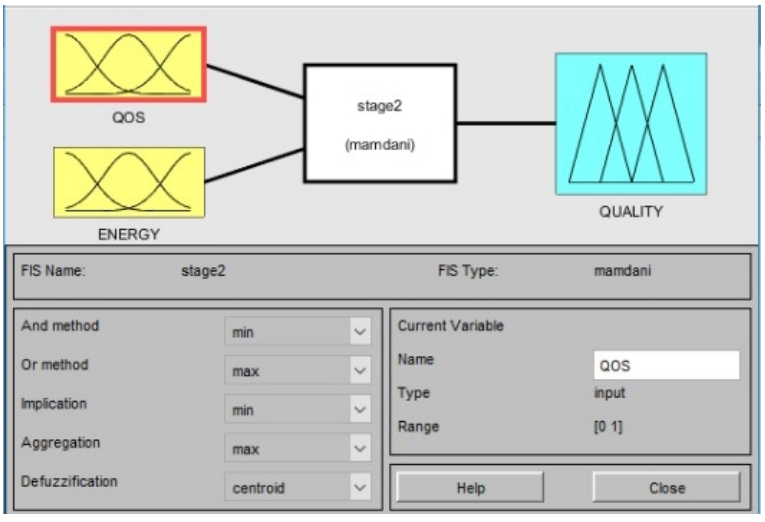
The toolbox lets you model complex system behaviors using simple logic rules, and then implement these rules in a fuzzy inference system. You can use it as a stand-alone fuzzy inference engine. Alternatively, you can use fuzzy inference blocks in Simulink and simulate the fuzzy systems within a comprehensive model of the entire dynamic system.

5.2.1.1 Fuzzy Logic Designer

Use the Fuzzy Logic Designer app or command-line functions to interactively design and test fuzzy inference systems. You can add or remove input and output variables. You can also specify input and output membership functions and fuzzy if-then rules. Once you have created fuzzy inference system, you can evaluate and visualize it.



(a) Stage 1



(b) Stage 2

Figure 5.4: Fuzzy Controller Designed in MATLAB

5.2.2 Requirements for implementing an OF in Contiki OS

Some importtant files that have the gist of RPL are ”rpl-config.h”, ”rpl-dag.c” and ”rpl.c”. Some important sections and parameters are mentioned here

RPL_DAG_MC

```
/* RPL_CONF_DAG_MC */
#ifdef RPL_CONF_DAG_MC
#define RPL_DAG_MC RPL_CONF_DAG_MC
#else
#define RPL_DAG_MC RPL_DAG_MC_NONE
#endif
```

Here RPL metric container of the type using ETX and ENERGY are supported. However if you develop an objective function and an associated metric container can be supported.

RPL_OF

```
/* RPL_CONF_OF */
#ifdef RPL_CONF_OF
#define RPL_OF RPL_CONF_OF
#else
#define RPL_OF rpl_mrhof
```

```
#endif
```

The RPL_CONF_OF parameter configures the RPL objective function. ETX is the default objective function here. This should be defined to be the name of an rpl_of object linked into the system image, e.g., rpl_of0. Here you can also define it to be your own developed Objective function.

RPL_INIT_LINK_METRIC

```
#ifndef RPL_CONF_INIT_LINK_METRIC
#define RPL_INIT_LINK_METRIC        5
#else
#define RPL_INIT_LINK_METRIC        RPL_CONF_INIT_LINK_METRIC
#endif
```

This is the initial metric attributed to the link when ETX is unknown. It can be changed to any desirable value.

An objective function basically uses a link metric and has function which subject to a contraint tries to choose the best path for routing. To define a completely new Objective Function file(without modifying the existing one) the following functions must be defined in them. Also the makefile should be accordingly modified and care should be taken that your new file should not run into compilation

and linking errors. Some of the API for RPL Objective function are:

- $reset(dag)$: Resets the objective function state for a specific DAG. This function is called when doing a global repair on the DAG.
- $neighbor\_link\_callback(parent, status, etx)$: Receives link-layer neighbor information.
- $best\_parent(parent1, parent2)$: Compares two parents and returns the best one, according to the OF.
- $best\_dag(dag1, dag2)$: Compares two DAGs and returns the best one, according to the OF.
- $calculate\_rank(parent, base\_rank)$: Calculates a rank value using the parent rank and a base rank.
- $update\_metric\_container(dag)$: Updates the metric container for outgoing DIOs in a certain DAG. If the objective function of the DAG does not use metric containers, the function should set the object type to $RPL\_DAG\_MC\_NONE$.

5.2.3 Implementation of a FIS in C

5.2.3.1 Fuzzification

The required trapezoidal membership functions can be implemented in C using the following functions

```
unsigned int DOWN(unsigned int x1, unsigned int x2, unsigned int X) {
return (TRUE * ((long)(X) - (long)(x2)))/((long)(x1) - (long)(x2));
}
unsigned int UP(unsigned int x1, unsigned int x2, unsigned int X) {
return DOWN(x2, x1, X);
}
```

The $DOWN()$ and $UP()$ functions implement the falling and rising edges of a trapezoidal membership function using the equation of a line with a maximum value of $TRUE = 100$ and a minimum value of 0

The membership functions over the entire range of input values are implemented using the $FIRST\_T\_NORM()$ for the lower values of the input range, $T\_NORM()$ for the central portion of the input and $LAST\_T\_NORM()$ is used for the higher values. Thus these produce the 3 trapezoidal membership functions required for all the input vairables in our project.

```
unsigned int FIRST_T_NORM(unsigned int b1, unsigned int b2, unsigned int px) {
if (px <= b1) return TRUE;
if (px > b1 && px <= b2) return DOWN(b1, b2, px);
return FALSE;
}
unsigned int T_NORM(unsigned int b1, unsigned int b2, unsigned int b3, unsigned in
if (px <= b1) return FALSE;
if (px > b1 && px <= b2) return UP(b1, b2, px);
```

```
if (px > b2 && px <= b3) return TRUE;

if (px > b3 && px <= b4) return DOWN(b3, b4, px);

return FALSE;

}

unsigned int LAST_T_NORM(unsigned int b1, unsigned int b2, unsigned int px) {

if (px <= b1) return FALSE;

if (px > b1 && px <= b2) return UP(b1, b2, px);

return TRUE;

}
```

The membership functions as described in section 5.1.1 and illustrated in Figure 4.3, and Figure 4.4 for ETX, delay, hopcount and Energy can be implemented as:

```
#define ETX_MAX 15

#define ETX_B1 ETX_MAX / 5

#define ETX_B2 ETX_B1 * 2

#define ETX_B3 ETX_B1 * 3

#define ETX_B4 ETX_B1 * 4

#define etx_short(etx) FIRST_T_NORM(ETX_B1, ETX_B2, etx)

#define etx_avg(etx) T_NORM(ETX_B1, ETX_B2, ETX_B3, ETX_B4, etx)

#define etx_long(etx) LAST_T_NORM(ETX_B3, ETX_B4, etx)


#define DELAY_MAX 3000

#define DLY_B1 DELAY_MAX / 5
```

```
#define DLY_B2 DLY_B1 * 2

#define DLY_B3 DLY_B1 * 3

#define DLY_B4 DLY_B1 * 4

#define dly_small(dly) FIRST_T_NORM(DLY_B1, DLY_B2, dly)

#define dly_avg(dly) T_NORM(DLY_B1, DLY_B2, DLY_B3, DLY_B4, dly)

#define dly_high(dly) LAST_T_NORM(DLY_B3, DLY_B4, dly)


#define HOPCOUNT_MAX 5

#define HC_B1 HOPCOUNT_MAX / 5

#define HC_B2 HC_B1 * 2

#define HC_B3 HC_B1 * 3

#define HC_B4 HC_B1 * 4

#define hc_near(hc) FIRST_T_NORM(HC_B1, HC_B2, hc)

#define hc_avg(hc) T_NORM(HC_B1, HC_B2, HC_B3, HC_B4, hc)

#define hc_far(hc) LAST_T_NORM(HC_B3, HC_B4, hc)


#define ENERGY_MAX 255

#define ENG_B1 ENERGY_MAX / 5

#define ENG_B2 ENG_B1 * 2

#define ENG_B3 ENG_B1 * 3

#define ENG_B4 ENG_B1 * 4

#define energy_low(eng) FIRST_T_NORM(ENG_B1, ENG_B2, (eng))

#define energy_medium(eng) T_NORM(ENG_B1, ENG_B2, ENG_B3, ENG_B4, (eng))
```

```
#define energy_full(eng) LAST_T_NORM(ENG_B3, ENG_B4, (eng))
```

5.2.3.2 Fuzzy Inference

The *min*() and *max*() functions which are used to evaluate the antecident in the fuzzy rule base are implemented using simple macros such as

```
#define max(a, b) x>y?x:y
#define min(a, b) x<y?x:y
```

The fuzzy rulebase can then be represented using max and min equivalents for AND and OR logic in the if-them statements

5.2.3.3 Aggregation

Aggregation is done using the Center of gravity method for QoS and Quality respectively

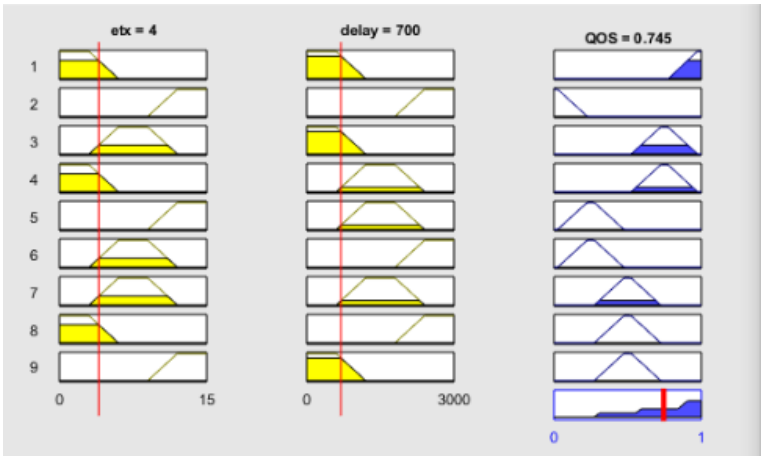
Chapter 6

Results and Discussion

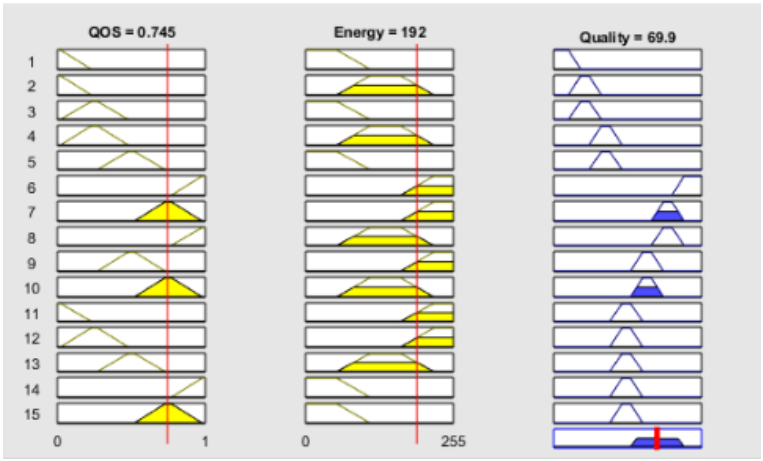
6.1 Results

Results obtained from the implementation of FIS in MATLAB are compared with the implemented code and the outputs are verified

6.1.1 Case1



(a) Stage 1 fuzzification outputs



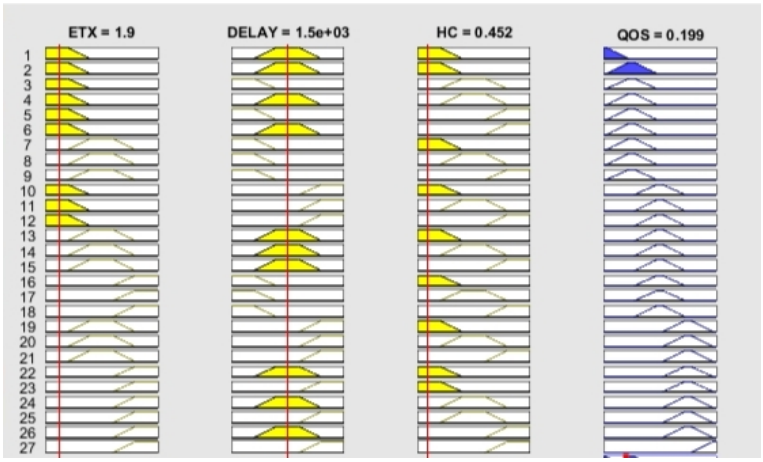
(b) Stage 2 fuzzification outputs

Figure 6.1: results of Fuzzy controller in MATLAB for 3 inputs

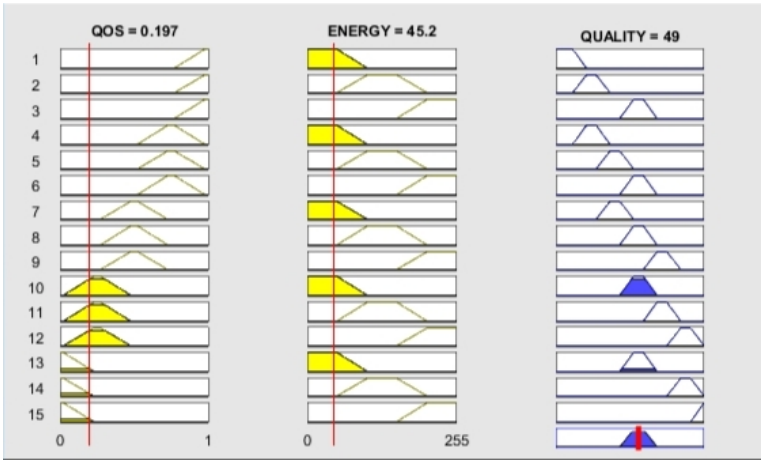
```
at reference $ ./FuzzyInference
etx: 4.000000
delay: 700.000000
hop count: 1.000000
energy: 192.000000
QoS: 78.571426
Quality: 73.705887
```

Figure 6.2: result from c implementation

6.1.2 Case 2



(a) Stage 1 fuzzification outputs



(b) Stage 2 fuzzification outputs

Figure 6.3: results of Fuzzy controller in MATLAB for 4 inputs

```
at fuzzy $ ./FuzzyInference
etx: 7.500000
delay: 1500.000000
hop count: 2.500000
QoS: 50.000000
Quality: 49.000000
```

Figure 6.4: result from c implementation

6.2 Discussion

To more precisely test the extent of the influence of Delays/Latency and Jitter in the Optimised Network we will have to carry out simulations for longer durations of time(1hr, 10hrs, 20hrs). We may also carry out the simulations using the optimised routing protocol for larger networks which may comprise of a few hundreds of nodes. We may also test the various existing OFs which use Fuzzy Logic for metrics combination with the OF we have designed.

Chapter 7

Conclusion

7.1 General Conclusion

We have designed a new Objective Function(OF) and implemented the same for Low Power and Lossy Networks. The New OF aims to Optimize more than one Network performance metrics. We have used a Fuzzy Inference System(FIS) to combine Hop count, Latency, Expected transmission count and Power dissipation into a unique value called Quality.

7.2 Future Work

In our future work, we will try to implement the optimized routing scheme in a heterogeneous wireless sensor network. In Heterogenous WSNs there are two or

more different types of nodes. These networks work on the principle of Clustering where a network is organized into groups or clusters where each cluster has a main node called Cluster Head.

We will also try to implement other forms of metric combinations (namely lexicographic and additive approaches) and compare their simulation results with those obtained with the fuzzy logic.

Bibliography

[1] Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio, Olivier Festor. "Fuzzy-based routing metrics combination for RPL". Doctoral Consortium Sensornets 2014. 2014, pp.8. hal-01093965

[2] Saaidah, Adeeb & Almomani, Omar & Al-Qaisi, Laila & Kamel, Mohammed. (2019). "An Efficient Design of RPL Objective Function for Routing in Internet of Things using Fuzzy Logic". International Journal of Advanced Computer Science and Applications. 10. 10.14569/IJACSA.2019.0100824.

[3] Altwassi, Hussien & Qasem, Mamoun & Bani Yassein, Muneer & Al-Dubai, Ahmed. (2015). "Performance Evaluation of RPL Objective Functions". 10.13140/RG.2.1.1790.2161.

[4] Lamaazi, Hanane & Ahmadi, Adnan & Benamar, Nabil & Jara, Antonio J.. (2019). "OF-ECF: A New Optimization of the Objective Function for Parent Selection in RPL". 27-32. 10.1109/WiMOB.2019.8923273.

[5] Mehmood, Tayyab. (2017). "COOJA Network Simulator: Exploring the Infinite Possible Ways to Compute the Performance Metrics of IOT Based Smart

Devices to Understand the Working of IOT Based Compression & Routing Protocols".