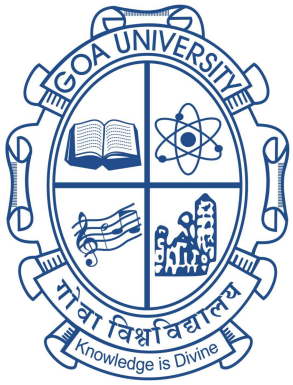
Optimization of Routing Protocols in Wireless Sensor Networks for Smart City Applications

A THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

In Electronics & Telecommunication Engineering

GOA UNIVERSITY



By

Ms. Sonia Santosh Kuwelkar

Research Centre in Electronics & Telecommunication Engineering Department
Goa College of Engineering

October 2024

DECLARATION

I, Sonia Santosh Kuwelkar hereby declare that this thesis represents work which has been carried out by me and that it has not been submitted, either in part or full, to any other University or Institution for the award of any research degree.



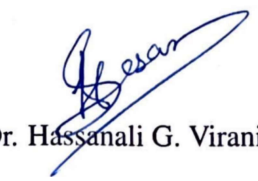
Place:Taleigao Plateau

Date: 04/10/2024

Sonia Santosh Kuwelkar

CERTIFICATE

I hereby certify that the above Declaration of the candidate, Sonia Santosh Kuwelkar is true and the work was carried out under my supervision.



Name of the Guide : Dr. Hassanali G. Virani

Affiliation of guide: Professor & Head

Electronics & Telecommunication Engg Department

Goa College of Engineering

THESIS APPROVAL

This thesis entitled **"Optimization of Routing Protocols in Wireless Sensor Networks for Smart City Applications"** submitted by **Ms. Sonia Santosh Kuwelkar** (P.R.No.:1631103001) is approved for the award of degree of **Doctor of Philosophy in Electronics & Telecommunication Engineering**

Examiners

1.....
Dr. Neelawar Shekhar Vittal Shet

2.....
Dr. Hassanali G. Virani

3.....
Dr. Anant Naik

Research Guide

.....
Dr. Hassanali G. Virani

Principal

.....
Dr. Krupashankara M. S.

Date: 04/10/2024

Place: Farmagudi, Ponda.

Acknowledgement

The time spent during the research period has been a mixed bag of innovative thoughts, challenges, opportunities, excitement, and frustration too. As I inch towards the end of the present research studies and move ahead to keep the flame of research lit always in the future, I take this opportunity to thank all those who guided and supported me during this amazing journey.

First and foremost, I would like to thank the almighty God for showing me the path and giving me the opportunity, strength, ability and knowledge to undertake the research and to persevere and complete it satisfactorily.

My deepest gratitude to Dr. Hassanali G. Virani, Prof & HOD, E&TC Dept, Goa College of Engineering and my research guide. He has been constantly supporting, motivating, and guiding me to march ahead with a positive mindset. He has been the focal point of the resourceful discussions, constructive arguments, and fruitful solutions during this research. I thank Dr. Hassanali G. Virani, wholeheartedly for the trust shown in my work on this research problem. This gratitude will be remembered for life.

The Director of Technical Education, Dr. Vivek Kamat has always been motivating the faculty under his purview for carrying out research work. I would like to thank his good office profoundly for all the support provided during the entire course of this research. I would like to thank Dr. Krupashankara M. S.,Principal of Goa College of Engineering and Dr. R. B. Lohani, Dr. V. N Shet, ex-principals, for the constant support in all the formal requirements during the research. A special thanks to Dr. R.B. Lohani, Prof, ETC dept, for his constant motivation and belief in me.

I would like to thank Dr. Mahabaleshwar S K, DRC committee expert, for thoughtful deliberations and constructive suggestions during this research. I would like to thank Dr. Nitesh Guinde, Professor, E&TC Dept and DRC committee expert for constant support, encouragement and constructive suggestions during this research.

Friends and colleagues always provide the strength you need to overcome the obstacles in your journey. I would like to thank all my colleagues today for the motivation they have provided during this research journey. Dr. Shajahan Kutty has been supporting me right from the start of this research. He has shared his immense expertise and experience in the form of valuable inputs during every discussion. My gratitude to him will remain forever. Dr. Sonali Pandit, a true motivator, she has been pushing me in the right direction and whose support, during the course of work, and support cannot be forgotten. I thank her wholeheartedly for being with me. I would like to extend my gratitude to Ms. Nayana Shenvi, my good friend for always lending me a patient ear. Ms. Amita Dessai has been a great companion during this research journey. I am grateful for her companionship and support. I express my deepest gratitude to all the teaching faculty and non teaching staff of E&TC dept, for their help and support during this journey.

A special thanks to my student Mr. Akhil Kushe for the technical assistance provided during

the entire research period. This research would not have been possible without his support. A special mention of my students Aniruddha Chari and Aaron Fernandes for their technical assistance.

Special thanks to Dr. Milind Sakhardande, Assoc. Prof, Mechanical Dept, for the valuable inputs provided by him during this research. Dr. Suraj Rane, has been a great source of inspiration and motivation. My special thanks to Shri Ketan Naik, Assistant Registrar, Goa College of Engineering, Shri Anand Naik & all the staff of Goa College of Engineering for their constant support.

I do understand that there are many other direct and indirect motivators and I acknowledge all of them at this juncture.

My acknowledgment would be incomplete without the biggest source of my strength, my family members.

I thank my late maternal grandfather Shri Dinkar Ramani, who cultivated the love for education and instilled moral values in me since my childhood.

My father, Shri Santosh Kuwelkar, for supporting me in all my decisions and believing in me always.

My mother, Meena, has been a source of motivation throughout my life. She has been always there for me on my call of 'mummy'. My mother has given me her unequivocal support throughout, for which my mere expression of thanks likewise does not suffice.

I am highly indebted and very much grateful to my father-in-law, Dr. Prakash Kurade and my mother-in-law, Smt.Kamal Kurade for their unconditional love, support blessings, and good wishes.

My deepest gratitude goes to my supportive and loving husband, Shri Siddhesh Kurade, who has been the major force behind my decision of joining PhD studies. His encouragement and support during the challenging times is what kept me going strong. Finally, big thanks, full of love and hugs, to my kids Sumukh and Sumedh for their love, understanding and wholeheartedly supporting me during this journey.

Thank you all.

Sonia Kuwelkar

Date: August 2024

Dedication

I dedicate this thesis to my parents for their endless love, support and encouragement throughout my pursuit for education

Abstract

Wireless Sensor Networks have evolved over the years and gained remarkable attention due to its widespread applications in various domains. A more recent version of Wireless Sensor Networks is the Internet of Things, expected to play a major role in smart cities applications, such as healthcare, waste management, air and water quality, noise monitoring, smart grids, home automation and traffic congestion. IoT network involves sensors with constrained memory, limited processing and power capabilities, thus characterizing it as a Low Power and Lossy Network. One crucial aspect of LLNs is the design of routing protocol, improvisations of which could lead to better outcomes and Quality of Service. The IETF ROLL proposed the IPv6 Routing Protocol for LLNs (RPL) as routing standard for these networks. In RPL, the network formation is governed by an Objective function that specifies a set of metrics or constraints which can be used as criteria for best parent and path selection, ensuring a faster route to the destination. The standard RPL use Objective functions which use a single metric for routing and are thus limited and inefficient. In this thesis, we have proposed three routing strategies for RPL, which incorporate multiple metrics for holistic routing and apply the techniques of fuzzy logic, neural networks and machine learning to improve network performance. Four routing metrics, namely Residual Energy, Expected Number of Transmissions (ETX), Delay and Hop Count are selected to overcome the limitation of single metric-based routing of standard RPL. Initially, an Objective Function OF-FZ is proposed, which applies Fuzzy Logic technique to compute the quality of each candidate neighbor node by evaluating the four routing metrics. The computed quality score is used as a single decision metric to select best parent and route from available options. Secondly, a Multi-layer Feed Forward Artificial Neural Network is designed to compute the best parent and optimal route. This version is termed ANN-RPL. Finally, RPL-ML(Random Forest) and RPL-ML(KNN) are proposed which apply the Random Forest and K-Nearest Neighbour Machine learning regression techniques respectively to select best route. Performance evaluation and deep dive of RPL-ML(Random Forest), RPL-ML(KNN), OF-FZ and ANN-RPL is done in two configurations: network scalability and increasing packet transmission rates using Contiki-2.7 Operating System and COOJA network simulator. OF-FZ, ANN-RPL, RPL-ML(KNN) and RPL-ML(Random Forest) give remarkable performance improvisations in terms of Average Power consumption, Packet Delivery Ratio, Latency and Control Overhead over standard RPL. The improvement achieved with RPL-ML(Random Forest) for power consumption is 16%, packet delivery ratio is 9%, latency is 25% and control overhead to the tune of 47% as compared to standard RPL. The objective function RPL-ML(Random Forest), which utilizes the Random Forest regression technique, gives superior performance as compared to the K-Nearest neighbour, Fuzzy and ANN based approaches.

Contents

1 Introduction **1**

1.1 Preamble 1
1.2 Research Motivation 3
1.3 Research Objectives 4
1.4 Organisation of the thesis 4

2 Literature Review **6**

2.1 Introduction 6
2.2 Wireless Sensor Networks for Smart City Applications 7
2.3 The IPv6 Routing Protocol for LLN's (RPL): Overview 9
2.4 IPv6 and 6LoWPAN stack : A brief overview 10
2.5 Need for RPL protocol 11
2.6 RPL DODAG Formation and Control Messages 12
2.7 RPL Objective Functions and Trickle Algorithm 15
2.7.1 Minimum Rank Hysteresis Objective Function (MRHOF) 15
2.7.2 Objective Function Zero (OF0) 16
2.7.3 Trickle Algorithm 16
2.8 RPL Metrics and Constraints 17
2.9 Shortcomings of Standard-RPL protocol 19
2.10 Related works on improvising RPL protocol performance 21
2.10.1 RPL improvement using Fuzzy Logic 21
2.10.2 RPL improvement using Evolutionary Algorithms 24
2.10.3 RPL improvement using Calculus and MCDM techniques 26

3 Simulation Setup **29**

3.1 Introduction 29
3.2 QOS Requirements in a network 31
3.3 Choice of Routing Metrics 32
3.4 Operating System 33
3.5 Specifications of Sensor Nodes 36
3.6 Performance Parameters Measured 38
3.7 Conclusion 39

4 Design of Efficient Objective Function for RPL using Fuzzy Logic **40**

4.1 Introduction 40

4.2 Research Objective . . . 41
4.3 Fuzzy Inference System . . . 41
4.4 Membership Functions for Routing Metrics . . . 42
4.5 Process for Best Parent Selection in Fuzzy Based-RPL implementation . . . 46
4.6 Design of Fuzzy Logic based Objective function RPL-FZ . . . 47
4.6.1 Fuzzy Inference System incorporation into Contiki OS . . . 50
4.6.2 Simulation Results of RPL-FZ . . . 51
4.6.3 Comparison of RPL-FZ with other existing Fuzzy logic-based implementations in literature . . . 59
4.7 Design of Fuzzy Logic based Objective Function OF-FZ . . . 61
4.7.1 Simulation Results of OF-FZ . . . 66
4.7.2 Performance Analysis of OF-FZ . . . 72
4.7.2.1 Comparison of OF-FZ with RPL-FZ . . . 72
4.7.2.2 Comparison of OF-FZ with existing four-metric fuzzy logic-based implementations in literature . . . 73
4.8 Conclusion . . . 77

5 Design of Efficient Objective Function for RPL using Artificial Neural Networks 79
5.1 Introduction . . . 79
5.2 Research Objective . . . 80
5.3 Design of ANN based objective function for RPL . . . 80
5.4 Simulation Results of ANN-RPL . . . 83
5.5 Performance Comparison of ANN-RPL with other existing works . . . 92
5.5.1 Comparison of ANN-RPL with DN-RPL . . . 93
5.5.2 Comparison of ANN-RPL with other Evolutionary Algorithm based works 95
5.6 Conclusion . . . 98

6 Design of Efficient Objective Function for RPL using Machine Learning 99
6.1 Introduction . . . 99
6.2 Research Objective . . . 100
6.3 Machine Learning Regression Analysis . . . 100
6.4 Linear Regression . . . 101
6.5 Random Forest . . . 102
6.6 K-Nearest Neighbour . . . 103
6.7 RPL using K-Nearest Neighbor Machine Learning technique . . . 106
6.8 Simulation Results of RPL-ML(K-NN) . . . 107
6.9 RPL-ML with Random Forest regressor versus RPL-ML (K-NN) with K=2,3,5 116
6.10 Performance Comparison of RPL-ML with other existing works . . . 121
6.11 Conclusion . . . 124

7 Performance Comparison of the proposed RPL implementations 126
7.1 Performance comparison with regard to Average Power consumption . . . 127
7.2 Performance comparison with regard to Packet delivery ratio . . . 130
7.3 Performance comparison with regard to Network Latency and Control Overhead 131

7.4 Conclusion 132

8 Conclusion **134**

8.1 General Conclusion 134

8.2 Research Contributions 137

8.3 Challenges 138

8.4 Future Work 138

Appendices **xvi**

A Appendix I **xvii**

List of Figures

2.1 6LoWPAN stack 10
2.2 RPL DODAG 12
2.3 DIO message format 13

3.1 Smart city application scenario 29
3.2 Flowchart depicting Sink-Sender communication and parent selection process . 30
3.3 Network view in COOJA simulator 30 and 50 nodes 35
3.4 Collect View application window 36
3.5 Zolertia Z1 sensor mote 37

4.1 Membership Functions of ETX, HC, Delay and RE 43
4.2 Process for Best Parent Selection using OF-FZ 47
4.3 Fuzzy Inference System in RPL-FZ 47
4.4 Membership Function for FIS output-Quality Score 48
4.5 Performance of RPL-FZ versus OF0 and MRHOF for varying network density . 52
4.6 Average Power Consumption with RPL-FZ, OF0 and MRHOF for varying data rates 54
4.7 Packet Delivery Ratios with RPL-FZ, OF0 and MRHOF for varying data rates . 56
4.8 Network Latency with RPL-FZ, OF0 and MRHOF for varying data rates 57
4.9 Control Traffic Overhead with RPL-FZ, OF0 and MRHOF for varying data rates 58
4.10 Comparison of OF-EC [77], OF-FUZZY [110], OPP-FL [78], OFRRT-FUZZY [79] and RPL-FZ 60
4.11 Fuzzy Inference System in OF-FZ 61
4.12 Membership function of Quality in Stage 1 of OF-FZ FIS 62
4.13 Membership function of QA Score in Stage 2 of OF-FZ FIS 64
4.14 Node with choice of candidate parents 64
4.15 Average Power Consumption with varying data rates for OF-FZ 67
4.16 PDR with varying data rates for OF-FZ 68
4.17 Network Latency with varying data rates for OF-FZ 69
4.18 Control Overhead with varying data rates for OF-FZ 70
4.19 Path traced from Nodes N1 and N2 in RPL networks with standard OF0, MRHOF and the proposed OF-FZ 71
4.20 Comparison of OF-FZ with RPL-FZ 72
4.21 PDR of OF-FZ, FL-HELR [81] and OF-FL [76] at 1 ppm and 5 ppm 74
4.22 Power Consumption with OF-FZ, FL-HELR [81] and OF-FL[76] 75
4.23 Network Latency and Overhead with OF-FZ, FL-HELR [81] and OF-FL [76] . 76

5.1 Artificial Neural network for selecting parent in RPL network. 81
5.2 Parent selection process using ANN-RPL 83
5.3 Power consumption and PDR with OF0, MRHOF and ANN-RPL for varying node density 84
5.4 Latency and Control Traffic Overhead for OF0, MRHOF and ANN-RPL for varying node density 85
5.5 Route chosen by Nodes N1 and N2 to Sink with OF0, MRHOF and ANN-RPL 87
5.6 Average power consumption for OF0, MRHOF and ANN-RPL at varying data rates 88
5.7 Packet Delivery ratios for OF0, MRHOF and ANN-RPL at varying data rates . 89
5.8 Latency for OF0, MRHOF and ANN-RPL at varying data rates 91
5.9 Control Traffic Overhead for OF0, MRHOF and ANN-RPL at varying data rates 92
5.10 End-to-End Delay with DN-RPL[111] and ANN-RPL 93
5.11 Packet Delivery Ratio with DN-RPL[111] and ANN-RPL 94
5.12 Energy Consumption with DN-RPL[111] and ANN-RPL 94
5.13 Comparison of PDR with RPL-GCA[83], HOFESA[112], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL 95
5.14 Comparison of Energy Consumption with RPL-GCA[83], HOFESA[112], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL 96
5.15 Comparison of Network Latency with RPL-GCA[83], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL 96
5.16 Comparison of control overhead with proposed ANN-RPL and existing works . 97

6.1 K-Nearest Neighbor Algorithm 103
6.2 K-Nearest Neighbor Algorithm Accuracy vs K 104
6.3 K-Nearest Neighbor Algorithm Error vs K 104
6.4 Prediction Accuracy for LR, RF and K-NN Regression 105
6.5 Prediction time and Training time for LR, RF and K-NN Regression 105
6.6 Avg Power consumption and PDR for OF0, MRHOF and RPL-ML(KNN) at varying node density 107
6.7 Latency and Overhead for OF0, MRHOF and RPL-ML(KNN) at 1 ppm 108
6.8 Avg Power consumption for OF0, MRHOF and RPL-ML(KNN) at varying data rates 109
6.9 Packet Delivery ratio for OF0, MRHOF and RPL-ML(KNN) at varying data rates 111
6.10 Network Latency for OF0, MRHOF and RPL-ML(KNN) at varying data rates . 112
6.11 Control Overhead for OF0, MRHOF and RPL-ML(KNN) at varying data rates . 113
6.12 Routes taken by Nodes N1 and N2 towards sink node with a) RPL -ML(KNN) and b) Standard RPL 115
6.13 Power consumption for RPL-ML (KNN) vs RPL-ML (Random Forest) 117
6.14 Network Latency for RPL-ML (KNN) vs RPL-ML (Random Forest) 118
6.15 Routes taken by Nodes N1 and N2 towards sink node with a) RPL -ML (KNN K=3) and b) RPL-ML (Random Forest Regressor) 120
6.16 Packet Delivery Ratio for RPL-ML with KNN vs Random Forest 121
6.17 PDR with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , HOFESA [112], FL-HELR [81] and MC-RPL[90] 122

6.18 Energy Consumption with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , HOFESA [112], FL-HELR [81] and MC-RPL[90] 122
6.19 Network Latency with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , FL-HELR [81] and MC-RPL[90] 123
6.20 Control Overhead with RPL-ML(KNN), RPL-ML(RF) versus E-RPL [84] , EEOPS-RPL [85] and FL-HELR [81] 123

7.1 Average Power Consumption with OF-FZ, ANN-RPL and RPL-ML(RF) 127
7.2 Percentage Improvement with RPL-ML(RF) over OF-FZ and ANN-RPL 128
7.3 Paths traced for two nodes N1 and N9 towards the sink node with RPL-ML(RF) and OF-FZ 129
7.4 Packet delivery ratio with OF-FZ, ANN-RPL and RPL-ML(RF) at 5 ppm and 10 ppm 131
7.5 Latency with OF-FZ, ANN-RPL and RPL-ML(RF) for 50 and 70 nodes at varying data rates 131
7.6 Percentage Improvement in Latency with and RPL-ML(RF) 132

8.1 Screen Shot of Published Paper in IETE Journal of Research, Taylor & Francis xiv
8.2 Screen Shot of Published Book Chapter in MDPI xv

List of Tables

2.1 Summary Table of RPL-Fuzzy Logic Based Literature 23
2.2 Summary Table of RPL- Evolutionary Algorithms Based Literature 25
2.3 Summary Table of RPL- Calculus and MCDM based works 28

3.1 Simulation Parameters 35

4.1 Rules to Compute Quality Score in RPL-FZ 49
4.2 Improvement in Avg Power Consumption and PDR with RPL-FZ over OF0 and MRHOF with varying node density at 1 ppm 52
4.3 Improvement in Network Latency and Control Overhead with RPL-FZ over OF0 and MRHOF with varying node density at 1 ppm 53
4.4 Improvement in Avg Power Consumption at 50 and 100 nodes with RPL-FZ over OF0 and MRHOF for varying data rates 55
4.5 Improvement in PDR at 50 and 100 nodes with RPL-FZ 57
4.6 Improvement in Network Latency with RPL-FZ for varying data rates 58
4.7 Improvement in Control Overhead at 50 and 100 nodes with RPL-FZ over OF0 and MRHOF for varying data rates 59
4.8 Comparison Table of RPL-FZ vs other 3-metric Fuzzy Based works existing in literature 60
4.9 Rules to Compute Quality in Stage 1 FIS for OF-FZ 63
4.10 Rules to Compute Quality Assurance Score in Stage 2 FIS for OF-FZ 64
4.11 Improvement in Avg power Consumption with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates 67
4.12 Improvement in PDR with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates 68
4.13 Improvement in Network Latency with OF-FZ over OF0 and MRHOF at 100 nodes for varying data rates 69
4.14 Improvement in Control Overhead with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates 70
4.15 Improvement of OF-FZ over RPL-FZ 73
4.16 Design Considerations: OF-FZ vs OF-FL [76] vs FL-HELR-OF [81] 73
4.17 Improvement in PDR of OF-FZ vs FL-HELR-OF vs OF-FL at 1ppm and 5 ppm 74
4.18 Improvement in Power Consumption of OF-FZ vs FL-HELR-OF vs OF-FL . . 75

5.1 Training Dataset samples 81
5.2 Four Metric values and computed Score by ANN 82
5.3 Improvement in Pavg and PDR with ANN-RPL over OF0 and MRHOF with varying node density at 1 PPM 84

5.4 Improvement in Network Latency and Control Overhead with ANN-RPL over OF0 and MRHOF with varying node density at 1 PPM 86
5.5 Four Metric values and computed Score by ANN for neighbors of N1 and N2 . 86
5.6 Improvement in Average Power Consumption with ANN-RPL at 50 and 100 nodes for varying data rates 89
5.7 Improvement in PDR with ANN-RPL at 50 and 100 nodes for varying data rates 90
5.8 Improvement in Network Latency with ANN-RPL at 50 and 100 nodes for varying data rates 90
5.9 Improvement in Control Overhead with ANN-RPL at 50 and 100 nodes for varying data rates 92
5.10 Comparison of ANN-RPL with other EA based works 97

6.1 Training Dataset samples 101
6.2 Linear Regression Performance Parameters 102
6.3 Random Forest Performance Parameters 102
6.4 K-NN Algorithm Performance Parameters 104
6.5 Improvement in Average Power consumption and PDR with RPL-ML(KNN) over OF0 and MRHOF at 1ppm 108
6.6 Improvement in Latency and Overhead with RPL-ML(KNN) over OF0 and MRHOF at 1ppm 108
6.7 Improvement in power consumption with RPL-ML(KNN) for varying data rates 110
6.8 Improvement in PDR with RPL-ML(KNN) for varying data rates 111
6.9 Improvement in Network Latency with RPL-ML(KNN) for varying data rates . 112
6.10 Improvement in Control Overhead with RPL-ML(KNN) for varying data rates . 114
6.11 CPU power for RPL-ML (KNN) versus RPL-ML (Random Forest) 117
6.12 Power consumption of RPL-ML (KNN) vs RPL-ML(Random Forest) at varying data rates 118
6.13 Improvement in Network Latency with RPL-ML (Random Forest) over RPL-ML (K-NN K=3) 119
6.14 Quality Scores of Candidate Parents from N1 to Sink with KNN and RF 120
6.15 Quality Scores of Candidate Parents from N2 to Sink with K-NN and RF . . . 120
6.16 Comparison of RPL-ML with other existing works in literature 124

7.1 Improvement with proposed works over standard RPL 127
7.2 QA scores computed by Fuzzy logic and Random Forest regressor for candidate parent nodes in path from N1 to sink 129
7.3 QA scores computed by Fuzzy logic and Random Forest regressor for candidate parent nodes in path from N9 to sink 130

8.1 Research Objectives to Publications Mapping 139

List of Abbreviations

ANN Artificial Neural Network

ANN-RPL Artificial Neural Network and four metric based RPL Objective Function

DAG Directed Acyclic Graph

DAO Destination Advertisement Object

DIO DODAG Information Object

DIS DODAG Information Solicitation

DODAG Destination Oriented Directed Acyclic Graph

ETX Expected Number of Transmissions

HC Hop Count

IoT Internet of Things

IETF Internet Engineering Task Force

IPv6 Internet Protocol Version 6

6LoWPAN IPv6 in Low-Power Wireless Personal Area Networks Working Group

KNN K-Nearest Neighbour Regression technique

LLN Low Power and Lossy Network

LBR LLN Border Router

MRHOF Minimum Rank Hysteresis Objective Function

OF Objective Function

OF0 Objective Function Zero

OF-FZ Fuzzy Logic and four metric based RPL Objective Function

PDR Packet Delivery Ratio

RE Residual Energy

RF Random Forest

RFC Request for Comments

RPL IPv6 Routing Protocol for Low power Lossy Networks

ROLL Routing over Low Power and Lossy Networks

RPL-FZ Fuzzy Logic and three metric based RPL Objective Function

RPL-ML(KNN) K-NN machine learning technique and four metric based RPL Objective Function

RPL-ML(RF) Random Forest machine learning technique and four metric based RPL Objective Function

WSN Wireless Sensor Networks

Chapter 1

Introduction

1.1 Preamble

In the past decade there has been significant progress in sensor technology, embedded processors and wireless communications. This had led to rapid advancement in the Wireless Sensor Networks domain. The sensors convert environmental phenomenon into electrical signals and transmit them to other sensors. By deploying WSN, temperature and humidity in rooms can be monitored, home appliances can be remotely controlled using automation in smart buildings, electricity and water billing meters can be read automatically, health of senior citizens can be remotely supervised, Traffic on roads can be monitored and smart parking can be enabled in smart cities, water quality and pipeline damage or leakage can be monitored in smart distribution networks. Apart from these applications in smart cities, WSN can also be used in military surveillance, industrial automation, weather monitoring and Logistics [1].

A more recent and popular evolution of WSN for smart cities is the Internet of Things and it is predicted that by 2025 there will be more than 27 billion IoT connections globally. IoT is a giant communications network connecting billions of devices like smart phones, wearable devices, sensors, actuators and home appliances over the Internet [2] . It aims to revolutionize the quality of human life by manifesting a plethora of applications in various domains

A sensor in IoT is a compact device that uses a microcontroller with limited computing ability such as the MSP430 with memory storage in range 64 KB to 256-KB and is battery powered [3]. Unlike conventional networks, the traffic in these networks can be point-to point, point to multipoint or multi-point to point. The communication network also has several drawbacks like instability, low data rates, high packet loss, limited transmission range, frame size limitation and dynamically varying network topology. Besides these aspects, the links connecting the devices

can use a variety of technologies like Bluetooth, Low power WIFI and wired PLC. Another aspect to be considered is that in large scale deployments, network can scale to thousands of nodes. Due to these innumerable constraints on devices and underlying communication technology, IoT network is characterized as a Low Power and Lossy Network (LLN).

In any network, the Routing protocol decides the routes and forwarding of data from nodes to the sink. A node has option of multiple routes to reach the sink. The key challenge is selecting the best route from the available choices to ensure optimal utilization of nodes resources. As the choice of route taken largely affects the network lifetime, design of an efficient Routing protocol for LLNs becomes a very challenging task. One major achievement by the Internet Engineering Task Force is the 6LoWPAN specification, which specifies how to carry IPv6 datagrams over LLNs. To devise a Routing solution for LLNs, the IETF constituted the Routing over Low Power and Lossy Networks (ROLL) working group and assigned them the task. The working group evaluated the existing routing protocols of WSNs like AODV, DSR, OSPF and OLSR. They realized that these are inefficient for LLNs with regards to power, latency, overhead and reliability. Eventually, the IETF ROLL working group proposed the IPv6 routing protocol for LLNs (RPL) as a routing solution [4]. RPL provides IPv6 connectivity to battery operated wireless embedded devices which uses low power radio to communicate. Traditionally, routing protocols like Collection Tree Protocol (CTP) and Hydro were designed for wireless sensor networks. Collection Tree Protocol [5] is a distance vector protocol and uses the ETX metric for selecting the best route during transferring data from one or more sensors to one or more root nodes. RPL builds upon Collection Tree Protocol (CTP) used in the WSN domain, but is re-designed and extended to incorporate IPv6. Over a period RPL has attained maturity and is standardized as the routing protocol for LLNs. The IP-stack of many operating systems such as Contiki OS, TinyOS, T-Kernel, EyeOS, LiteOS and RIOT includes RPL as standard routing protocol [6]. Over the years, several researchers have evaluated RPL performance and highlighted certain limitations thus giving scope for further improvisations.

In this thesis, we propose three routing strategies for the RPL protocol, designed using techniques of Fuzzy Logic, Artificial Neural Networks and Machine learning to overcome the limitation of single metric based routing in standard RPL. We evaluate the performance of the three proposed RPL implementations in terms of Power Consumption, Packet Delivery Ratio, Latency and Control Traffic Overhead.

1.2 Research Motivation

The network in IoT comprises of sensors that have constraints in power, computing ability and storage capacity The deployment environment is noisy and the links which connect the nodes support low data rates and have comparatively lower packet delivery ratios. The nodes are responsible for routing the packets of other nodes in the network in addition to forwarding data of their own. If the routing decisions are not intelligent, then it will lead to more number of retransmissions thus depleting the nodes of resources and in turn reducing the network lifetime. As revealed by several studies, the conventional routing protocols designed for WSN, ADHOC networks and MANET's were found to be unsuitable to adapt to the lossy and dynamic nature of these networks. Hence, design of Routing protocol for such networks posed a challenge to researchers. After rigorous exercise, The Internet Engineering Task Force (IETF) ROLL working group proposed the IPV6- Routing Protocol for Low power and Lossy Network (RPL) as a de-facto routing standard for these networks.

Although RPL was able to resolve major issues there were still certain limitations that needed addressing. For instance, standard RPL makes the routing decision considering a single metric which leads to the selection of non-optimal paths and thus reduces network lifetime. RPL also suffers from unbalanced load distribution between parent nodes and has slow convergence.

These shortcomings give scope to improve RPL further and have driven us to design an efficient solution with higher Quality of Service. In this thesis, we have addressed the issue of single metric-based routing in RPL. It is seen that selecting a route by taking only a single factor such as Hop count into consideration leads to poor choices. It can lead to certain nodes and routes being repeatedly selected, thus depleting them rapidly of energy and reducing the overall network lifetime. We devised a solution that utilizes four metrics in making the routing decision. The metrics chosen are Residual Energy, Hop Count, ETX and Delay. The challenge faced was in formulating a multi criteria decision making computation technique to determine the best route. Taking into account the resource constraints faced by nodes, this technique had to be accurate but not computation extensive. This ruled out the conventional hard computing methods. Artificial Intelligence and Soft Computing has drawn focus over the past few years and gained popularity in solving issues in various application domains. In this research, we have focused on these techniques and their application in augmenting the RPL protocol. The novel versions of RPL protocol proposed in this thesis are designed with an aim to provide better Quality of Service than existing solutions.

1.3 Research Objectives

The main aim of this research is to design a routing strategy that overcomes the single metric-based routing limitation of existing standard RPL protocol. It is modeled as a multi-constrained optimal path problem, where best route decision is based on multiple routing metrics and Soft Computing techniques are employed to determine the optimal route toward the sink node. The objectives of this research are:

1. Apply the technique of Fuzzy Logic to design a routing strategy for RPL that selects the best route based on three (ETX, residual energy and delay) and four routing metrics (ETX, residual energy, delay and hop count).
2. Design a routing strategy for RPL that applies a Multilayer Feed Forward Artificial Neural Network.
3. Apply the Random Forest and K-Nearest Neighbor regression techniques of Machine Learning to take the best route decision by evaluating the four metrics. Analyze the performance of the three approaches.

1.4 Organisation of the thesis

The thesis report is organised in the form of eight chapters. The first chapter introduces the research topic and presents the aims and objectives of the research. The second chapter provides an overview of the RPL protocol. The need of RPL protocol, its working and the various metrics which can be utilized during routing are discussed. The open issues in RPL protocol are listed. This chapter also includes the detailed literature review in all the concerned areas of research. Chapter three describes the simulation setup. The metrics chosen in design, the Operating system, simulation parameters and the performance parameters to be measured are described in this chapter. The chapter four presents the detailed design process of novel Objective Function using fuzzy logic approach. The design of two stage fuzzy inference system to perform multi criteria decision making is described. The performance analysis of this novel fuzzy based RPL version is presented by comparing it with the existing standard RPL. The simulations are carried out by varying node density from 30 to 100 and packet transmission rates from 1 to 10 packets per minute. Chapter five details the design steps of novel Artificial Neural network based RPL version and the process of incorporating it into the Contiki OS. Chapter six presents the machine learning regression techniques of Random Forest, K-Nearest Neighbor

and Linear regression and their applicability in improvising the RPL protocol. The performance results of novel ML based RPL version are presented. We present the Performance Comparison of the proposed three novel versions in chapter seven with regard to power consumption, latency, packet delivery ratio and control overhead. The research contributions, conclusions and future research directions are presented in chapter eight.

Chapter 2

Literature Review

2.1 Introduction

The Wireless Sensor Networks have created new opportunities across the spectrum of human endeavors, including health care, forest fire tracking, monitoring and control of environmental systems, engineering design and manufacturing, battlefield surveillance, disaster management, and critical infrastructure protection [7]. The deployment of sensors in a WSN depends on the nature of the application. Once deployed, sensor nodes self-organize into an autonomous wireless ad-hoc network, which requires very little or no maintenance [8],[9]. Sensor nodes then collaborate to carry out the tasks of the application for which they are deployed. Despite the disparity in the objectives of sensor applications, the main task of wireless sensor nodes is to sense and collect data from a target domain, process the data, and transmit the information back to specific sites where the underlying application resides [10]. Achieving this task efficiently requires the development of an energy-efficient routing protocol to set up paths between sensor nodes and the data sink. The path selection must be such that the lifetime of the network is maximized. The characteristics of the environment within which sensor nodes typically operate, coupled with severe resource and energy limitations, make the routing problem very challenging.

A more recent evolution of WSN are Low power and Lossy Networks(LLN) which are the dominant wireless networks comprising the Internet of Things and find applications in smart cities [11]. The devising of a routing protocol for LLNs is especially a challenging task owing to the lossy and constrained nature of such networks. The Internet Engineering Task Force constituted the Routing over Low Power and Lossy Networks (ROLL) working group and assigned them the task of devising a Routing solution for LLNs. The working group evaluated the existing

routing protocols of Ad-HOC networks like AODV, DSR, OSPF and OLSR. They realized that these are inefficient for LLNs with regard to power, latency, overhead and reliability. Eventually, the IETF ROLL working group proposed the IPv6 routing protocol for LLNs (RPL) as a routing solution [4].The RPL protocol was proposed in 2012 as RFC 6550. Since then, researchers have taken a deep interest in the standard and come up with techniques to improve its performance or methods to overcome its limitations [12]. In this chapter, we provide a brief review of the RPL protocol and the existing works in literature that have tried to overcome the limitation of RPL protocol using various computing approaches.

2.2 Wireless Sensor Networks for Smart City Applications

The vast urbanization of population has provoked the need for smart cities. The improvement of sensing and communication technologies in Wireless Sensor Networks (WSNs) is making it conceivable to adopt this technology in smart cities [13]. Sensor nodes can be deployed in areas that are hardly accessible to humans, thanks to their limited size and to their communication capabilities [14].
The applications of WSN from the smart cities perspective includes:

1. Smart Water: The water supply is one of the most significant perspectives for cities. With intelligent sensors, the water quality and levels can be checked progressively. These sensors can send triggers and alarms to key chiefs for low or high water levels. The spillages and water dispersion can be monitored using sensors. The water distribution network can be efficiently managed using sensors. A complete outline of the water supply with GPS directions can be given to water specialists with IoT frameworks[15].

2. Smart Grid : Balancing of Load distribution and cable monitoring can be accomplished using WSN in energy grids [16]. One of the critical difficulties for state run administrations is to decrease energy utilization. Electronic meters with automatic meter reading capability and intelligent lighting frameworks are a portion of the components that are utilized by legislatures to oversee energy effectively. The power dispersion guides can show ongoing energy utilization levels, spillages, and upkeep plans. The IoT empowered arrangements can improve the energy of the board for urban areas [17].

3. Smart Transportation: Automated toll collection, accident avoidance, parking management and traffic monitoring are domains which can largely benefit using WSN. Traffic is one of the significant problem areas for residents. With IoT sensors, traffic can be

controlled better [18]. The sensors are associated with traffic lights and send data to an incorporated server. The approaching vehicles are followed utilizing these sensors. When the quantity of vehicles arrives at a limit, signals are shipped off to the drivers to redirect. A Smart parking facility installed with sensors can detect and communicate available parking spots to civilians in designated parking spaces.

4. Smart Agriculture : Various activities like crop monitoring, plants watering, crop disease inspection and pesticide administration can be effortlessly accomplished using sensor devices in fields [19].

5. Smart Healthcare: Clinics and medical services frameworks are significant marks of administration. The smart city requires a state of the art medical services framework that can follow quantifiable advancement concerning residents well being. With the aid of sensors health of citizens, especially the elderly can be monitored [20], [21].

6. Pollution Control With Sensors: Urbanization has prompted an uncommon expansion in contamination levels. The rising contamination levels are causing medical problems for residents. With IoT empowered sensors, contamination can be estimated progressively. The contamination sensors send data to an incorporated server.

7. Waste and Garbage Management: Waste management system within a city can be improved with intelligent sensors and IoT Frameworks. The trash containers can utilize intelligent sensors to demonstrate when they should be discharged. This diminishes the times that vehicles are expected to gather the trash from the receptacles and evades what is going on of waste flood.

8. Smart Surveillance Systems: Sensor devices with cameras can assist the public authority with controlling crime percentages in a city. The sensors can be utilized for surveying and monitoring essential establishments.

Wireless sensor nodes sense and collect data from a target domain, process the data, and transmit the information back to specific sites where the underlying application resides. Achieving this task efficiently requires the development of an energy-efficient routing protocol to set up paths between sensor nodes and the data sink. The routing protocols in WSN are classified according to the manner in which information is acquired and maintained and the method used to compute paths based on the acquired information. They differ in the network architecture and structure adopted and the technique of used for data dissemination and gathering. Almost all of the routing protocols can be classified as data-centric, hierarchical or location-based [22]. The popular

routing protocols employed in WSN are SPIN [23], Directed Diffusion [24], [25],Gradient-based routing (GBR) [26], COUGAR [27],[28], ACQUIRE [29], LEACH [30], PEGASIS [31] , TEEN [32] , APTEEN [33], MECN [34], SMECN [35] and GEAR [36].

The lack of an IP-based network architecture precluded existing routing protocols from inter-operating with the Internet, limiting their real-world impact. Furthermore, without an IP-based architecture, these systems require an application-layer gateway to communicate with other systems or even the wider Internet. Application gateways are complex to design and manage since they perform significant functional and semantic translation and carry an application-layer state. Given this disconnect, the IETF chartered the 6LoWPAN and RoLL working groups to specify standards at various layers of the protocol stack with the goal of connecting low-power and lossy networks to the Internet. The IETF proposed the IPv6 Routing Protocol for LLNs (RPL) and several researchers have worked to improvise the initially proposed version of RPL.

2.3 The IPv6 Routing Protocol for LLN's (RPL): Overview

In a Wireless Sensor Network the individual nodes forward their data packets to the gateway or sink via other neighbor nodes. The formation of routes is governed by the routing protocol and it is responsible for making smart routing choices. To devise a Routing solution for Low Power and Lossy Networks, the Internet Engineering Task Force constituted the Routing over Low Power and Lossy Networks (ROLL) working group and assigned them the task. The working group evaluated the existing routing protocols of WSNs like AODV, DSR, OSPF and OLSR. They realized that these are inefficient for LLNs with regard to power, latency, overhead and reliability. Eventually, the IETF ROLL working group proposed the IPv6 routing protocol for LLNs (RPL) as a routing solution [4]. The Working Group defined the RPL routing protocol, which builds a robust topology over lossy links with minimal state requirements. RPL provides IPv6 connectivity to battery-operated wireless embedded devices which use low-power radio to communicate. RPL builds upon popular routing protocols like CTP and Hydro used in the WSN domain but is re-designed and extended to incorporate IPv6.

RPL can be deployed in networks that scale to thousands of nodes, with majority of nodes having constrained resources and where the network is controlled by a single central sink or super-node. The traffic patterns supported by RPL include multipoint-to-point, point-to-multipoint and point-to-point. As of early 2011, RPL was deemed "ready" by the IETF, for publication as a "Proposed Standard" in RFC (Request for Comments) 6550. A protocol is labeled "Proposed Standard" implies that it is generally stable: community reviewed and well understood,

no known design issues pending, and with some community support [37]. This protocol was standardized by the IETF as RFC 6550 in March 2012. In fact, the RPL is designed with the goal of meeting the routing requirements defined by the IETF ROLL WG in 2009 for four LLNs application areas: home automation LLNs (RFC 5826) [38], building automation LLNs (RFC 5867) [39], urban LLNs (RFC 5548) [40] and industrial LLNs (RFC 5673)[41]. RPL is included in IP stack of many operating systems such as Contiki OS, TinyOS, T-Kernel, EyeOS, LiteOS and RIOT.

2.4 IPv6 and 6LoWPAN stack : A brief overview

In the past decade, significant efforts have been dedicated by the IETF to the design of a complete IPv6-based network architecture for LLNs to provide interoperability with existing IP devices and services [42]. The two major milestones obtained by IETF are the specification of the 6LoWPAN adaptation layer, which specifies how to carry IPv6 datagrams over constrained links [43] and, more recently, Routing Protocol for Low-Power and Lossy Networks (RPL). The IPv6 in Low-Power Wireless Personal Area Networks Working Group was chartered to standardize necessary adaptations of IPv6 for networks that use the IEEE 802.15.4 physical (PHY) layer, and has defined how to carry IPv6 datagrams over IEEE 802.15.4 links and perform necessary configuration functions to form and maintain an IPv6 subnet [44]. The 6LoWPAN is an Internet Engineering Task Force (IETF) standard defined in RFC6282. As the full name implies – "IPv6 over Low-Power Wireless Personal Area Networks" – 6LoWPAN is a networking technology or adaptation layer that allows IPv6 packets to be carried efficiently within small link layer frames, such as those defined by IEEE 802.15.4. The figure 2.1 shows the 6LoWPAN stack. It

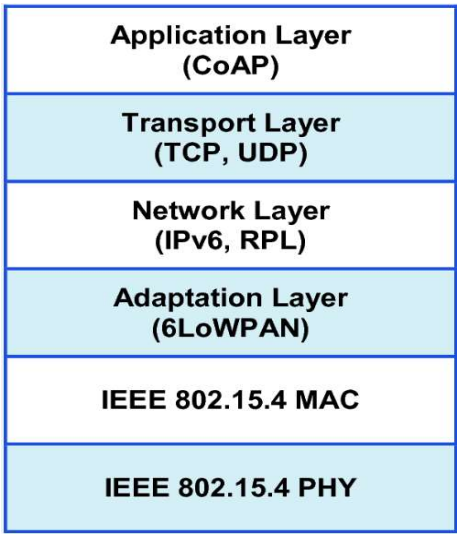


Figure 2.1: 6LoWPAN stack

is defined for low data rates (20/40/250 kbps)and network sizes to a max of 100 nodes. When sending data over MAC and PHY layers, an adaptation layer is always used. For example, RFC 2464 defines how an IPv6 packet is encapsulated in an Ethernet frame. The same is also used for IEEE 802.11 Wi-Fi. For 6LoWPAN, RFC 6282 defines how an IPv6 data frame is encapsulated over an IEEE 802.15.4 radio link. The main focus of the IETF working group, 6LoWPAN WG, was to optimize the transmission of IPv6 packets over low-power and lossy networks (LLNs) such as IEEE 802.15.4 and led to the publication of RFC 6282 specifying : Header compression, stateless auto configuration, fragmentation and reassembly. By 2030, 50 billion devices (things) are expected to be connected to the Internet to form the Internet of Things (IoT) world[45]. In general, two main categories of networks are used in the IoT: short-range and long-range low-power networks. IPv6 over low-power wireless personal area networks (6LoWPAN) are considered to be a crucial network in short-range low-power networks where 6LoWPAN motes will account for the majority of short-range low-power things [46]. Also, LoRaWAN and SigFox are two major networking landscapes and players in long-range, low-power networks or oftentimes called low-power wide area networks (LPWAN).

2.5 Need for RPL protocol

Two types of routing protocols are there in WSNs: Proactive and Reactive. The Reactive protocols are also termed as On-Demand routing protocols. The routes here are discovered only when required. The route request packets are flooded through the network in order to discover the routes. This amounts to large control traffic during the time of route construction, in turn increasing the time taken for route construction. Reactive protocols are infeasible in real-time applications owing to the delays involved. Two popular reactive protocols are AODV [47] and DSR [48]. Experiments conducted reveal that RPL is more efficient than AODV in terms of energy efficiency [49], [50]. Proactive protocols on the other hand create the routes before they are actually required by the nodes for data transfer. These protocols exchange control messages periodically to find the best routes and propagate them in the network as soon as the network is deployed. An up-to-date routing table is maintained at each node by frequently querying its immediate neighbors for routing information. Some commonly used proactive protocols are OLSR and DSDV [51]. These are link-state protocols and require a significant amount of memory to store the routing information, hence unsuitable in resource-constrained LLNs. This rendered most of the existing routing protocols of ADHOC networks and MANETs infeasible for LLNs. RPL is a proactive routing protocol designed specifically for resource-constrained LLNs. The

control traffic overhead is kept to a minimum and just sufficient to run network operations. A number of routing protocols are proposed for Wireless Sensor Networks, some popular ones being LEACH, PEGASIS, SPIN, TEEN, GAF, GEAR, COUGAR and ACQUIRE [52]. However, these do not possess IPv6 capability and have limited scalability, thus ruled out for IoT applications. A more recent version of AODV protocol is the Lightweight On-demand Ad hoc Distance-vector Routing Protocol (LOADng) which is designed for IPv6 based IoT scenarios. The use of radio transceiver determines the battery usage and thus the lifetime of a node. In the popularly used IoT nodes such as the Tmote Sky platform, the power consumption on account of radio transmission is three times higher than CPU processing. This implies that control traffic overhead and number of hops are the determining factors for node lifetime. Studies show that the control traffic overhead of LOADng does not scale well with dense deployments of smart home applications. RPL fared better over LOADng in this regard. RPL promised faster network convergence, better data delivery, lesser traffic overhead, better energy efficiency and network scalability in IoT scenarios [53]. RPL's ability to quickly build network routes and effectively adapt to network topology makes it a perfect option for 6LoWPAN devices [54] .

2.6 RPL DODAG Formation and Control Messages

RPL is a proactive distance-vector based routing protocol. The network is organized as a Destination Oriented Directed Acyclic Graph (DODAG) in which the destination is the Root node or LLN Border Router (LBR). The LBR acts as the gateway for connecting the local network to the Internet. When a network is deployed, the Root node initiates the construction of DODAG [55]. A node can join the DODAG by selecting a preferred parent node.

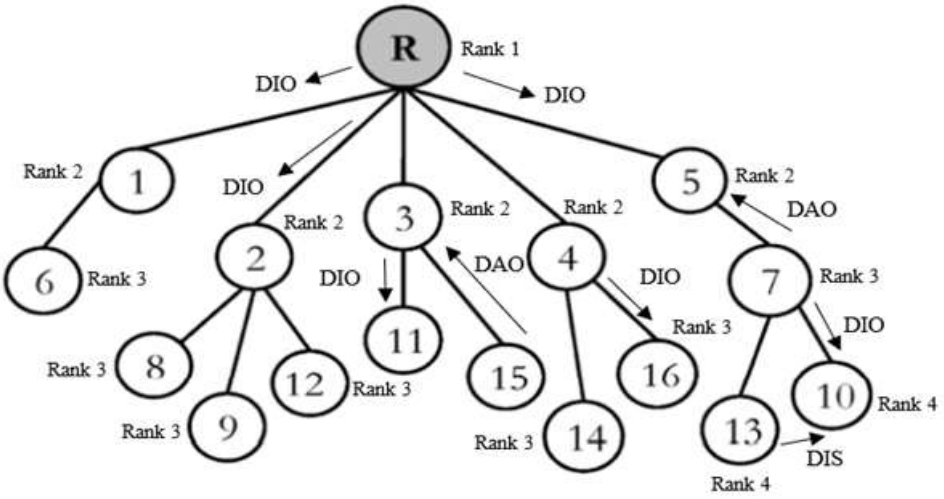


Figure 2.2: RPL DODAG

A DODAG is depicted in figure 2.2. Each node is assigned a Rank, which is directly proportional to the distance from root node. LBR has the lowest rank and the rank increases as we move away from the root. Every parent node will have a rank lower than its children's nodes.
To make all this functional and share the routing sensitive information, RPL protocol specifies four ICMPv6 control messages.
The control messages are:

1. DODAG Information Object (DIO): This message contains routing crucial information like Objective function, rank, metrics, DODAG ID, trickle timer parameters and RPL Instance ID. This information helps a node to select a parent, identify a RPL instance and join DODAG . The DIO message body comprises of two parts, DODAG configuration option and DAG metric container, where the important information is embedded as shown in figure 2.3. DIO messages are essential for constructing upward routes and maintaining the DODAG.

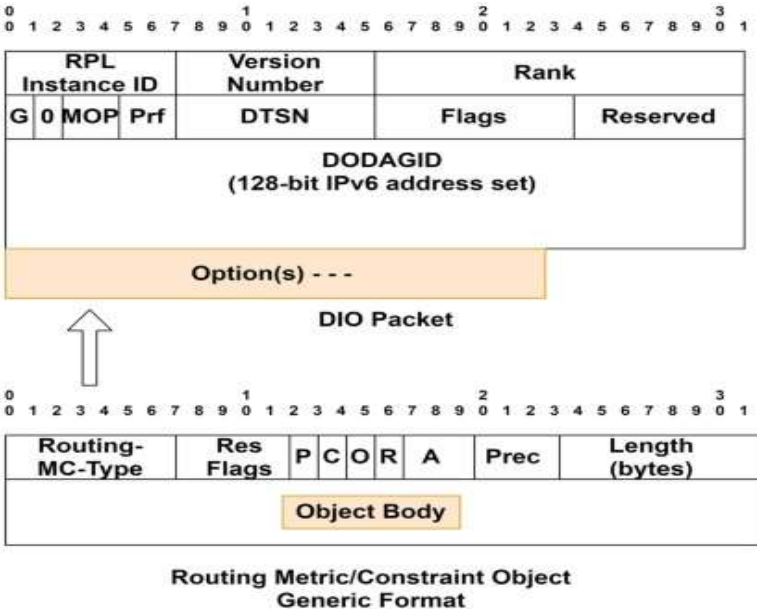


Figure 2.3: DIO message format

2. Destination Advertisement Object (DAO): These messages are primarily used to propagate destination information in up direction towards the DODAG root. This information in used for construction of downward routes between DODAG root and associated nodes.

3. DODAG Information Solicitation (DIS): A node who wants to join a DODAG and has not received a DIO can use the DIS message to solicit a DIO from its neighboring nodes.

4. Destination Advertisement Object Acknowledgement (DAO-ACK): Upon receiving DAO messages, a node can unicast a DAO-ACK to the DAO sender to acknowledge the receipt

of that DAO.

In a RPL based infrastructure, one or more RPL instances can exists, where each instance is identified by an unique RPLInstance_ID. Each instance can have its own set of metrics and routing policy independent of its neighboring instance. Multiple DODAGs can exists within a single RPL instance. To distinguish the DODAG they are assigned a DODAG_ID. Every node can participate in multiple DODAGs. In case of any inconsistency in the network, the sink can trigger DODAGs to reconstruct themselves. An incremental DODAG Version_Number is assigned to the DODAG by the sink every time it reconstructs itself. The exact version of a particular DODAG can be identified by the tuple of RPLInstance_ID, DODAG_ID and DODAG Version_Number. RPL protocol creates Destination Oriented Directed Acyclic graphs (DODAG), which consist of a single root node and several leaf nodes. The gateway or border router is termed as Root node. All other sensor nodes in the network transmit their data towards the gateway. At the beginning, to construct a DODAG, the root node broadcasts a DIO (DODAG Information Object) message [6] within the network. Neighbors of the root receive and process the DIO message. The processed DIO message is transmitted to other nodes one by one. A particular node, upon receiving a DIO message from its neighbor, adds this neighbor to the Parent list as a potential candidate parent node (CPN). To route the data towards the DODAG root, a node has to choose its best parent from the list of candidates. The choice of best parent is dependent on the metrics specified by the objective function. Some of the metrics specified in the RFC are Hop Count, Link Quality, Received signal strength ratio, throughput, energy level and delay. After selecting the best parent, a node computes its own rank relative to the rank of its parent and replies with a DAO message to its parent to inform its participation. Every node has a rank higher than its parent node. The Root node is assigned the least Rank within the DODAG. Rank computed by the other nodes increases uniformly with the hops from root node. All of the neighbors repeat this process until all of the nodes join the DODAG. After every routing decision, an updated DIO message is transmitted by a node to its neighbors. When any node joins the network anew, it can request a neighbour for the DIO using a DIS (DODAG Information Solicitation) message. Transmission of DIO messages is activated by the expiration of Trickle Timer. To minimize the control traffic overhead, the interval of trickle timer expiry is doubled after every DIO transmission. The routing information for Upward routes is contained in the DIO messages. Sensors transmit Destination Advertisement objects (DAO) to parent nodes and children to advertise membership. These DAO messages are useful for updating downward route information. If a node senses its parent increment DAO trigger sequence number, then it can trigger children to transmit DAO messages. If a node adds a

neighbor to its candidate parent list it can trigger a DAO. DAOs are also triggered on detection of downward route inconsistency. These are transmitted on expiry of a DelayDAO timer.

2.7 RPL Objective Functions and Trickle Algorithm

The root node initiates network formation by broadcasting a DIO message. Whenever a node receives a DIO, it is able to determine and join the RPL instance, add nodes to its candidate parent list, choose the best parent out of the candidates and replace a previous parent. The DIO message contains an Objective function along with other information. The term Objective Function (OF) is used to describe the set of rules and policies that governs the process of route selection and optimization, in a way that meets the different requirements of the various applications. In technical terms, the OF is used for two primary goals: first, it specifies how the Rank can be derived from one or a set of routing metrics (e.g.link reliability, throughput, latency, Hop count and energy); and second, it defines how the rank should be used for selecting the preferred parent. The nodes select the best route by constraining or optimizing certain metrics specified by the Objective function. Currently, two OFs have been standardized for RPL namely, the Objective Function Zero (OF0) [56] and the Minimum Rank with Hysteresis Objective Function (MRHOF) [57].

2.7.1 Minimum Rank Hysteresis Objective Function (MRHOF)

The MRHOF objective function uses the path cost or Expected transmission count (ETX) as a routing metric to select the preferred parent in the DODAG [57]. ETX is the number of retransmissions required before a packet is successfully received at the destination. MRHOF is designed with an aim to prevent frequent changes in the preferred parent. In MRHOF the path cost of each candidate neighbor is calculated by summing up two components, the value of the neighbor's node or link cost and the selected metric value (e.g., rank) advertised in metric container. Once the path cost is computed for all the candidates in the list, the one having the lowest value is selected as preferred parent node. MRHOF defines a PARENT_SWITCH_THRESHOLD which is taken into consideration before a parent switch is made. If the path cost of the new candidate parent exceeds the present parent node by this threshold, only then the parent is switched. This is the hysteresis part of MRHOF. For example, if a present parent A has a link cost of 120 and a rank of 256 then its total cost is 376. Let the PARENT_SWITCH_THRESHOLD be set at 100. Now, if a second candidate node B is available with path cost 82 and rank 256, then the

total cost for this candidate B is 338. In spite of having a lower cost, preferred parent will not be switched to B since the difference in cost between A and B is 38 which is lower than the specified threshold of 100.

2.7.2 Objective Function Zero (OF0)

It depends on Hop Count as a metric to make routing decisions [56]. A candidate parent with the lowest hop count, which implies that a node closest to the root will be chosen as the preferred parent. The node will then compute its Rank (Rn) by adding a scalar Rank_increase to the rank of selected parent Rank(parent).

$$Rn = Rank(parent) + Rank_increase \tag{2.1}$$

The rank_increase is computed as follows:

$$Rank_increase = (Rf * Sp + Sr) * MinHopRankIncrease \tag{2.2}$$

Where Rf is rank factor, Sp is step_of_rank and Sr is stretch_of_rank. Rf and Sr are normalization factor with default values 1 and 0 respectively. Sp has a default of 3 and MinHopRankIncrease has value 256. For instance, if a node A received a DIO from a node B whose rank is 256, then a RPL implementation with OF0 will choose B as the parent node and rank of A will be computed as 1024.

Both these Objective functions have the shortcoming of taking into account only single metric for routing decision. They lack the holistic approach essential for providing QOS. To overcome this, RPL specification has provided the designer the flexibility to incorporate more metrics into the decision-making process.

2.7.3 Trickle Algorithm

One of the key design principles of RPL is minimizing the routing control overhead and signaling data in order to reduce energy consumption and enhance reliability. In this regard, RPL employs the Trickle algorithm [58], [59] to govern the transmission of the signaling traffic used to construct and maintain the DODAG. The basic idea behind Trickle is to adjust the frequency of message transmission based on network conditions. Trickle relies on two simple mechanisms to disseminate routing information efficiently. The first is to change adaptively the signaling rate

according to conditions currently present in the network [59]. Specifically, Trickle increases the transmission rate when a change in routing information is discovered (i.e.,an inconsistency is detected) as a means to populate the network rapidly with up-to-date information[59]. As the network approaches its steady phase, Trickle exponentially reduces the transmission rate to limit the number of transmissions when there is no update to propagate. The second mechanism used by Trickle is the suppression mechanism in which a node suppresses the transmission of its control packet if it detects that enough of its neighbors have transmitted the same piece of information, thus limiting redundant transmissions. The time in Trickle is divided into intervals of variable size. A node running Trickle schedules a message to be sent at a randomly selected time in each interval. The transmission of a scheduled message is governed by Trickle parameters, variables, and steps.

The trickle timer runs for a definite interval depending on three configuration parameters:

a) Imin, the minimum interval size denoted in units of time like seconds or milliseconds.

b) Imax, the maximum interval size is specified in terms of the number of doublings of Imin.

c) k, a redundancy constant which is an integer greater than zero.

Besides these, three other parameters involved are: the current interval size I, a counter c and time t within the current interval. At the start of execution, current interval I is set to a value in between the range [Imin, Imax]. When the interval starts, counter c is reset to zero and the time t is set to a random value between [I/2, I]. When trickle receives packets consistently, the counter c is incremented monotonically and at time t, trickle transmits if the value of counter c is less than k [59]. Each time the interval I expire, its value is doubled till it reaches the value of Imax, after which it is maintained constant at Imax. If the trickle senses an inconsistent transmission, then it resets the timer by setting I to Imin and starts a new interval.

2.8 RPL Metrics and Constraints

Some of the metrics which have been used in combination are:

1. Node Residual Energy (RE): This metric is a measure of the remnant battery level of a sensor node. By considering this metric it is possible to avoid the selection of parents with lower energy levels. This is most desirable in networks with energy efficiency concerns. The residual energy (RE) is given as:

$$RE = Eo - Ec \tag{2.3}$$

2. End-to-End Delay or Latency: The average time required for a packet to travel from sender to destination is called latency. It is particularly crucial in real-time applications to have a very low delay.

3. Expected transmission count (ETX): The number of re-transmissions expected before a data packet is successfully delivered to the sink node is termed ETX. It's a rough estimate of the link quality. A lower value of ETX signifies a reliable and good link. It is given by the formula

$$ETX = \frac{1}{df * dr} \tag{2.4}$$

where df is the probability of successful delivery of a packet in forward direction and dr is the probability of successfully receiving the acknowledgment.

4. Hop Count (HC): It is a measure of distance between the DODAG root and the node in terms of number of hops. IoT applications that have real-time execution constraints prefer low hop counts for faster processing.

5. RtMetric: Multiple paths exist between a pair of nodes. Depending on the routes chosen by nodes the congestion level for every route would be different. Rtmetric is a cost associated with every route within the network. The protocol will use this route metric to select the path with the least congestion.

6. Buffer Occupancy (BO): Every node within a wireless sensor network may have different rates of packet arrival and transmission. Hence each node has buffers to store and forward packets. Buffer occupancy BO denotes the number of packets queued at the buffer for re-transmission.

7. Received signal strength indicator (RSSI): This metric is associated with the RF transceiver signal power. Basically, RSSI is the strength of the signal over the link between two neighboring nodes. It is an important factor in mobile networks. The nodes use the CC2420 radio chip for communication which has an in-built RSSI register. By reading this register, the RSSI value can be measured. Before the RSSI VALID bit is set, the RSSI value is averaged over minimum eight symbol periods and is given by

$$RSSI = RSSI_VALUE + RSSI_OFFSET \tag{2.5}$$

where RSSI_OFFSET is the front-end gain, computed analytically by manufacturers.

8. Mobility timer (MT): If a radio link within a mobile environment, responsible for forwarding packets to DODAG root suddenly becomes unavailable due to movement of that node then all the forwarded packets will be dropped. This problem will persist until a new link is identified or the link failure is detected. A metric termed as mobility timer MT is introduced to address this issue. The value of Mobility Timer is calculated based on the distance between mobile node and the candidate parent node (CPN) considering the maximum speed that this mobile node can attain. The approximate distance between mobile node and CPN can be computed from the RSSI equation as follows:

$$RSSI = -10 * n * \log D + C \tag{2.6}$$

Where D is the approximate distance, C represents fixed constant and n is the path loss exponent factor. The remaining distance (RD) before losing connectivity with the CPN is calculated by subtracting the approximate distance D from the CPNs communication range CR.

$$RD = CR–D \tag{2.7}$$

Later, to compute the value of Mobility Timer, we have to divide the remaining distance RD by the MNs max speed S.

$$MT = \frac{RD}{S} \tag{2.8}$$

9. Children Number: The number of children affiliated to a particular parent or neighbor node are reflected by this metric. Ideally, the number of children have to be distributed uniformly, to avoid saturation or overloading of few parents. By using this metric during routing load balancing can be done, by promoting the selection of a less loaded neighbor as parent. The CN can be calculated by monitoring the ICMPv6 messages, particularly the DAO and DAO-ACK messages exchanged to cancel or confirm a candidate parent during the selection process.

2.9 Shortcomings of Standard-RPL protocol

RPL could cater major requirements of LLNs thus making implementation practically possible. Over the years several researchers have studied and experimented with RPL. In this section we discuss some shortcomings of the existing standard RPL which have been reported by researchers working in this domain, addressing which can lead to further QoS enhancements.

1. Standard RPL uses an objective function which depends on a single metric, MRHOF on ETX and OF0 on Hop Count, to make the routing decision. The other parameters like node energy, delay is not considered while selecting best parent. This can lead to a non-optimal node being repeatedly selected as parent thus depleting it of its energy. This can adversely affect the network lifetime [60]. These standard objective functions tend to overlook latency aspect during parent selection. As a result, for applications with real time data delivery requirements, these standard OFs prove to be inefficient [61], [62], [63].

2. RPL has an inherent load balancing issue. The number of children nodes assigned to a particular parent can be much higher than some other parent. This will exhaust the parents' energy and reduce overall network performance. There is no defined scheme in RPL which regulates the number of children nodes across available parents [64].

3. RPL employs passive link estimation technique where the number of required re transmissions (ETX) is used to determine the link quality. This is inaccurate as compared to active link monitoring techniques, can converge slowly and may lead to RPL choosing long and unreliable links due to incomplete knowledge of neighborhood [6], [65].

4. Mobile IoT has recently gained popularity, but since RPL was primarily designed for stationary applications, adapting to the dynamic and fluctuating nature of mobile applications is a challenge. Many researchers have contributed to tune and provide mobility support for RPL, however still gaps exists before it can be made a standard for mobile IoT [66], [67].

5. In heterogeneous networks, the nodes have diverse hardware specifications, especially their transmission and reception queue buffers. RPL is does not differentiate between homogeneous and heterogeneous networks, which can lead to unbalanced traffic load and congestion in the networks [68], [69].

6. In construction of downward routes, DAO messages play an important role. RPL does not explicitly specify the timing of DAO transmission. This may lead to improper employment of the protocol leading to unproductive results [70]. In a RPL implementation, if root does not receive the DAO messages from all the intermediate nodes along a path, then it won't be able to determine the source route to that destination

7. RPL is subjected to security attacks like Neighbor DIS, Replay Attacks, Route Table Falsification, Local Repair, DAO Inconsistency, Replay Attacks, Version Number, Rank,

Worst Parent, DIS, Routing Table Overload and DAG inconsistency [71]. Several mechanisms are proposed to tackle the security issues in RPL[72], [73].

2.10 Related works on improvising RPL protocol performance

The IETF standardized the RPL protocol in 2012, and since then deep dive analysis of the protocol is been done by researchers across the globe. In this section, we have provided a brief description of the various techniques and methods devised to improvise the RPL protocol. Our primary focus is on research conducted to suggest approaches to overcome the limitation of single metric based routing in standard RPL. Related works in literature are scrutinized and the prominent ones are presented here. We have divided the literature into three categories: Fuzzy Logic based, Evolutionary Algorithm based and Calculus-based.

2.10.1 RPL improvement using Fuzzy Logic

Fuzzy Logic is seen as one popular choice of researchers to improvise RPL. The main reason for favoring it, is the ease of downloading the fuzzy logic into the tiny memory of the motes. The fuzzy inference system occupies unsubstantial memory space and computation time is minimal.

The pioneer work using fuzzy logic is by Kamgueu [74], [75], in which the routing decision is formulated as a a Multi-Constrained Optimal Path problem. A fuzzy logic controller is designed in two stages which takes three routing metrics delay, ETX and Energy as inputs and computes the quality of all the nodes in vicinity. The one with highest quality is chosen as parent. This approach is seen to outperform the ETX based RPL in terms of packet loss ratio, routing stability, energy efficiency and delay. The performance is evaluated in Contiki OS with 20 nodes and varying packet rates and also with 28 Telos B nodes in practical deployment.

Gaddour has proposed the OF-FL [76] objective function for RPL which adds the hop count metric to the work of Kamgueu. Another difference being the single stage design of the fuzzy controller in OF-FL with a Rule base of 81 rules. Simulations are carried out comparing the performance of OF-FL with standard RPL objective functions for network sizes up to 100 Tmote Sky motes. Better Network Lifetime, PDR, End to end delay and lower frequency of parent change is achieved as compared to standard RPL version.

Hanane [77] has made a similar attempt by using three metrics namely Hop Count, Energy consumption and ETX to propose an objective function OF-EC. A single stage fuzzy controller computes the quality of neighbors. The performance of OF-EC is evaluated in random and

grid topology by varying density from 10 to 50 nodes at packet rate of 1 ppm. The energy consumption, latency, PDR, control overhead, Network lifetime, number of parent changes and number of nodes alive are monitored. OF-EC is shown to outperform standard RPL and OF-FUZZY.

Kechiche [78] have proposed an objective function OPP-FL, which utilizes a novel routing metric Children Number (CN) in combination with ETX and HC. The CN metric is a measure of the number of children presently affiliated to a given neighbor. OPP-FL is tested in simulation with Zolertia Z1 motes at network densities upto 80 in random and linear topology at 1 ppm data rate. OPP-FL is compared with OF-EC and MRHOF in terms of PDR, overhead and end to end delay.

Adeeb Saidaah [79] has proposed the OFRRT-FUZZY objective function which differs from the above works in choice of metrics. It employs the RSSI, throughput and Remaining Energy metric during routing decision. OFRRT-FUZZY is compared in performance to OF0 and MRHOF by running simulations in network with Sky motes upto density of 45 deployed in grid topology. The average energy consumed, PDR, control overhead and Latency for the three are measured and OFRRT-FUZZY is found to outperform OF0 and MRHOF.

In a novel approach, Shreyas [69] have modeled the optimal path problem as a MADM-FUZZY [80] problem. Every node has a buffer associated with it to store and forward packets. If the buffer gets full then the packets which come in later are dropped. This reduces the packet delivery ratio and affects the network performance. RPL does not ensure load balancing, so the distribution of children is not uniform among the parent nodes. If a parent1 had five children nodes and some other parent2 has three children, the chances of parent1 buffer getting full is more as compared to parent2. To overcome this issue, a metric called buffer occupancy is utilized in this method to ensure uniform distribution. It corresponds to the number of packets queued at the buffer of a node for transmission. Another metric used is Rtmetric, which is a measure of congestion on any particular route. Along with Buffer occupancy and Rtmetric, ETX metric is also used for decision making during routing. This novel work, termed as CAUF is seen to improve the throughput and reduce packet loss over standard RPL. Darabkh [81] has proposed a novel FL-HELR-OF which computes the optimal path based on four metrics- RSSI, HC, EC and Latency. A single stage fuzzy controller is designed to determine the Quality of the best route. The performance is tested in Contiki OS with Zolertia Z1 motes by increasing density till 100 with data rate of 1 ppm in random topology. Upon comparing the PDR, power consumption, overhead, latency and Average Hop count with that of MRHOF and OF0, it is seen that FL-HELR-OF excels. The table 2.1 summarizes the above described literature works.

Table 2.1: Summary Table of RPL-Fuzzy Logic Based Literature

Year	Work	Metrics	OS/ Motes Used	Nodes	Data Rate	Parameters Evaluated	Limitation
2014	OF-FUZZY	ETX, Delay, Energy	Contiki OS/ COOJA Tmote Sky	20	1 ppm 2 ppm 12 ppm 20 ppm	Packet reception ratio and energy exhaustion	Not tested for more than 20 sender nodes
2015	OF-FUZZY	ETX, Delay, Energy	Contiki OS/ Practical /TelosB Motes	28	1 pkt/ 5 min	packet loss ratio, routing stability, energy efficiency and delay	Tested for only 28 sender nodes with very low packet rate.
2014	OF-FL	ETX, Delay, Energy, Hop Count	Contiki OS/ COOJA/ Tmote Sky	100	1 ppm 6 ppm	Average Hop Count, Delay, Packet loss ratio, Number of parent changes and Network lifetime	Not tested in Practical deployment .
2018	OF-EC	HC, EC, ETX	Contiki OS/ COOJA/ Tmote Sky	10 to 50	1 ppm	energy consumption, latency, PDR, control overhead, Network lifetime, number of parent changes and number of nodes alive are monitored.	Not tested for data rate higher than 1 ppm and density more than 50.
2019	OPP -FL	CN ETX and HC	Contiki OS/ COOJA/ Zolertia Z1	80	1 ppm	PDR, overhead and end to end delay	Not tested for data rate higher than 1 ppm.
2019	OFRRT-FUZZY	RSSI, Thrghput RE	Contiki OS/ COOJA/ Sky Motes	45	1 ppm	The average energy consumed, PDR, control overhead and Latency	Not tested at higher data rates and density more than 50.
2019	CAUF	Buffer occup-ancy, Rtmetric ,ETX	Contiki OS/ COOJA Sky Motes	25	1 ppm	throughput, packet loss	Not tested at higher data rates and density higher than 50.
2022	FL-HELR -OF	RSSI, HC, EC, Latency	Contiki OS/ COOJA/ Zolertia Z1	100	1 ppm	Improves PDR, power consumption, overhead, latency, Average Hop count	Not tested at higher data rates

2.10.2 RPL improvement using Evolutionary Algorithms

In this section we discuss the research done towards optimizing the RPL protocol using Evolutionary Algorithms. Evolutionary algorithms [82] like genetic algorithm, particle swarm optimization and ant colony optimization are popularly known for their application in solving real-time problems where conventional methods fail.

Cao [83] has proposed a novel RPL-CGA which uses a chaotic genetic algorithm to decide the optimal path. A composition metric is defined which comprises of five metrics namely, Queue length, Delay, RER, HC and ETX. Weighting factors are assigned to each metric within the composite metric. A genetic algorithm with a chaotic mechanism is devised to compute the weighting factors. The average success rate, end to end delay, network lifetime, average hop count, frequency of parent change is evaluated for node density varying from 50 to 400.

Preeth [84] have proposed an ant colony optimization based RPL implementation termed as E-RPL. In this method, the Children Number, Residual energy (RER), metrics link quality (ETX) and Rank value are used. The link quality and rank value are considered as pheromone factors, whereas the heuristic factors taken are RER and children number. The child-parent relationship denotes the pheromone evaporation factor. The pheromone, heuristic factors and evaporation factor are combined to form an objective function using weights. The protocol is tested in Contiki OS COOJA simulator in a network with node density between 30 to 70 deployed in random topology at 1ppm data rate.

In a novel approach the researchers [85] have used firefly optimization algorithm to improve RPL. This proposed version is termed as EEOPS-RPL. Here ETX and Residual energy (RER) metrics are chosen as the attractiveness parameters and the distance between nodes as the movement parameter to select the optimal parent in the DODAG. EEOPS-RPL is found to give faster convergence and longer network lifetime. EEOPS-RPL improves the packet transmission ratio and lifespan of the network by 2% to 5%, and 5% to 10%, respectively, compared with EPC-RPL and E-RPL. The protocol is tested in COOJA simulator in a network scenario with node density from 30 to 70 deployed in random topology at 1ppm data rate.

In another work focusing on the security aspect of RPL, the authors have devised a secure scheme for RPL based on Moth flame optimization algorithm [86]. Distance is the single decision metric. MFO-RPL uses the petal algorithm to select the next-hop nodes and construct the optimal path between the source and sink in the DODAG. Rank attacks in RPL are detected using the Moth-Flame algorithm to prevent malicious nodes from being selected as the preferred parent. Simulation findings under different scenarios revealed that MFO-RPL has fewer lost

packets, convergence time, rank switching and attacks than comparative schemes. MFO-RPL is compared with MRHOF and Sec-trust RPL. The table 2.2 summarizes the EA based RPL optimization works in literature.

Carlos et.al. [87] have proposed a novel RPL design, utilizes CatBoost, a Gradient Boosted Decision Trees (GBDT) algorithm, to optimize routing decisions. The ML model, trained on a dataset of routing metrics, predicts the probability of successfully reaching a destination node. The performance evaluation, reveals that ML-RPL significantly improves the packet delivery ratio and minimizes end-to-end delay, making it a promising solution for more efficient and responsive Wireless Smart Grid Networks.

Table 2.2: Summary Table of RPL- Evolutionary Algorithms Based Literature

Year	Work	Metrics	Technique Used	Nodes	Tools Used	Parameters Evaluated	Limitation
2018	RPL -CGA	Queue-length, Delay, RER, HC, ETX	Genetic Algorithm with chaotic mechanism	50 to 400	software simulator	Average success rate, end-to-end delay, network lifetime, average hop count, frequency of parent change	Not tested in network simulators like COOJA Not tested for higher data rates
2020	E-RPL	Children-Number, Residual-energy, ETX, Rank-value	Ant Colony Optimization	30 to 70	Contiki OS COOJA	PDR Energy consumption Overhead Delay	Not tested for higher data rates than 1 ppm
2021	EEOPS -RPL	RER, ETX Distance	Firefly optimization algorithm	30 to 70	Contiki OS COOJA	PDR Energy consumption Overhead Delay	Not tested at data rates higher than 1 ppm
2022	MFO-RPL	Distance	Moth Flame Optimization	50 100 150	MATLAB R2019a	Packets lost, convergence time, rank switching and attacks	Not tested in network simulators like COOJA Not tested for higher data rates

2.10.3 RPL improvement using Calculus and MCDM techniques

This section describes the literature works that have used conventional calculus methods and Multi Criteria Decision Making (MCDM) techniques to optimize the RPL protocol.

In a novel work the authors [88] have proposed RPL-OC which utilizes four metrics Energy, RSSI, delay and hop count. The combination of the four metrics presents as a multi-constrained optimal path problem. The authors have applied Operator calculus to solve the optimal path problem. The operator calculus algorithm [89] is primarily developed to solve pre-computed routing problems. OC is based on the properties of matrix computation and the definition of algebraic structures. RPL-OC is evaluated in Contiki OS with 25 nodes at data rates of 1 to 60 p/min. It is compared with MRHOF and OF-FL for throughput, energy consumption, radio duty cycle and delay.

Farzaneh [90] have proposed a novel MC-RPL, which uses the traditional VIKOR multi criteria decision making approach. The four metrics, ETX, hop count, delay and energy are used for decision making. MC-RPL is evaluated in Contiki OS with 30 T-mote sky motes with packet rate of 12 ppm. The PDR, throughput, delay and energy consumption is determined. An energy efficient cross-layer OF ELITE, which introduces a novel routing metric, SPR is proposed [91].

Strobe per Packet Ratio (SPR) indicates the number of transmitted strobes per packet due to Radio Duty Cycling (RDC) policies in the MAC layer. ELITE tries to select a path, which imposes less number of strobe transmissions to its nodes. It is evaluated in network scenario with 25, 50 and 75 Zolertia Z1 motes at packet rates of 0.5 p/min, 1.5 p/min and 2.5 p/min. The energy consumption, delay, PDR and Overhead achieved with ELITE is compared MRHOF and OF-FL.

To addresses the issues of packet losses, routing overhead and load imbalance in RPL a unique Dual Context-based Routing and Load Balancing (DCRL-RPL) is proposed [92]. A grid construction is implemented by splitting the network area into a grids of unequal sizes. The nodes within a grid are classified as grid members and a single grid head is chosen based on a ranking selection process . The grid head schedules the member nodes using a reputation based scheduling method. The grid head classifies the data received from members with an Adam deep Neural Network (ADNN) into sensitive and non-sensitive data.

For sensitive data transmission, the O^2F is selected, which adopts an optimisation algorithm. For the non-sensitive data the MOORA algorithm is used. The MOORA algorithm considers the energy consumption, Hop count, channel capacity, parent rank, and sub-tree size as criteria

for optimal parent node selection. An adaptive trickle timer mechanism is adopted to curtail the control messages transmission. The efficacy of DCRL-RPL is analyzed using the Network Simulator 3.26 (NS3) tool with 50 nodes for network lifetime, overhead, PDR, delay, packet loss and load balancing capability.

The authors [93] have proposed a new objective function, CAOF, which recursively examines the last states of the chain of nodes in the path toward the root. It considers the remaining power of the parent chain in the route toward the root and combines it with ETX and parent rank. Moreover, CAOF avoids Thundering Herd Phenomenon by gradual shifting from a high rank value toward the real rank value. A new routing metric, known as Context-Aware Routing Metric (CARF) is utilized. This metric considers Queue Utilization and remaining power of parents' chain instead of a single parent, thus helping in both power and load balancing. The protocol is analysed in COOJA simulator with 50 Tmote sky nodes for packet loss, energy consumption and network lifetime.

CORPL, an enhanced version of RPL is proposed for cognitive smart grid AMI networks. CORPL [94] utilizes an opportunistic forwarding approach that not only ensures protection to PUs but also fulfils the utility requirements of the secondary network. It is implemented and tested in MATLAB for 9 PU transmitters. The opportunistic forwarding approach consists of two key steps: selection of a forwarder set, where each node in the network selects multiple next hop neighbors, and a coordination scheme to ensure that only the best receiver of each packet forwards it. It is seen to improve the reliability by 50%.

In another MCDM based approach, the authors have focused on the TOPSIS method to combine three routing metrics -hop count, RSSI, and ETX and design a novel objective function called TFUZZY-OF [95]. This technique helps solve the load balancing problem by choosing the best parent node from a set of available parents' sets that is a less or non-congested parent in the network. TFUZZY-OF is evaluated in Contiki OS with 30 Zolertia Z1 motes for PDR and delay. The table 2.3 summarizes the various works towards the enhancement of RPL using calculus and MCDM techniques.

Table 2.3: Summary Table of RPL- Calculus and MCDM based works

Year	Work	Metrics	Technique Used	Nodes	Tools Used	Parameters Evaluated	Limitation
2014	CORPL	ETX	Path Cost Function	9	MATLAB	Reliability	Not tested in network simulators like COOJA
2018	CAOF	RE ETX Rank	Calculus	50	Contiki OS COOJA	PDR, Energy consumption, Lifetime	Not tested for more than 50 nodes
2020	DCRL -RPL	RE Rank' HC channel capacity	ADNN MOORA O^2F	50	NS3	PDR, lifetime Energy consumption Overhead Delay, load bal	Not tested for more than 50 nodes.
2020	ELITE	SPR	Calculus	25 50 75	Contiki	EC, PDR overhead delay	Not tested for data rate more than 3 ppm
2022	RPL-OC	Energy HC RSSI delay	Operator Calculus	25	Contiki COOJA	throughput delay EC, radio duty cycle	Not tested for more than 25 nodes.
2022	MC-RPL	Energy HC ETX Delay	VIKOR MCDM	30	Contiki COOJA	PDR Delay EC thruput	Not tested for more than 30 nodes.
2023	TFUZZY -OF	RSSI HC ETX	TOPSIS MCDM	30	Contiki COOJA	PDR Delay	Not tested for more than 30 nodes.

Since the inception of the RPL protocol in 2012, researchers have taken a deep interest in the standard and come up with techniques to improve its performance or methods to overcome its limitations. In this chapter, we made an attempt to provide a brief review of existing works in literature that have tried to overcome the limitation of RPL using computing approach of Fuzzy Logic, Evolutionary Algorithms, MCDM and Calculus.

Chapter 3

Simulation Setup

3.1 Introduction

An Internet of Things setup consists of the user application that can access and control a network remotely over the cloud and the deployed local wireless sensor network which senses the data and transmits it to the sink or central gateway. The figure 3.1 depicts a sensor network deployed for smart parking application. The gateway or sink node communicates over the Internet to the Cloud. The sensor nodes communicate with the gateway over the wireless 6LoWPAN network.

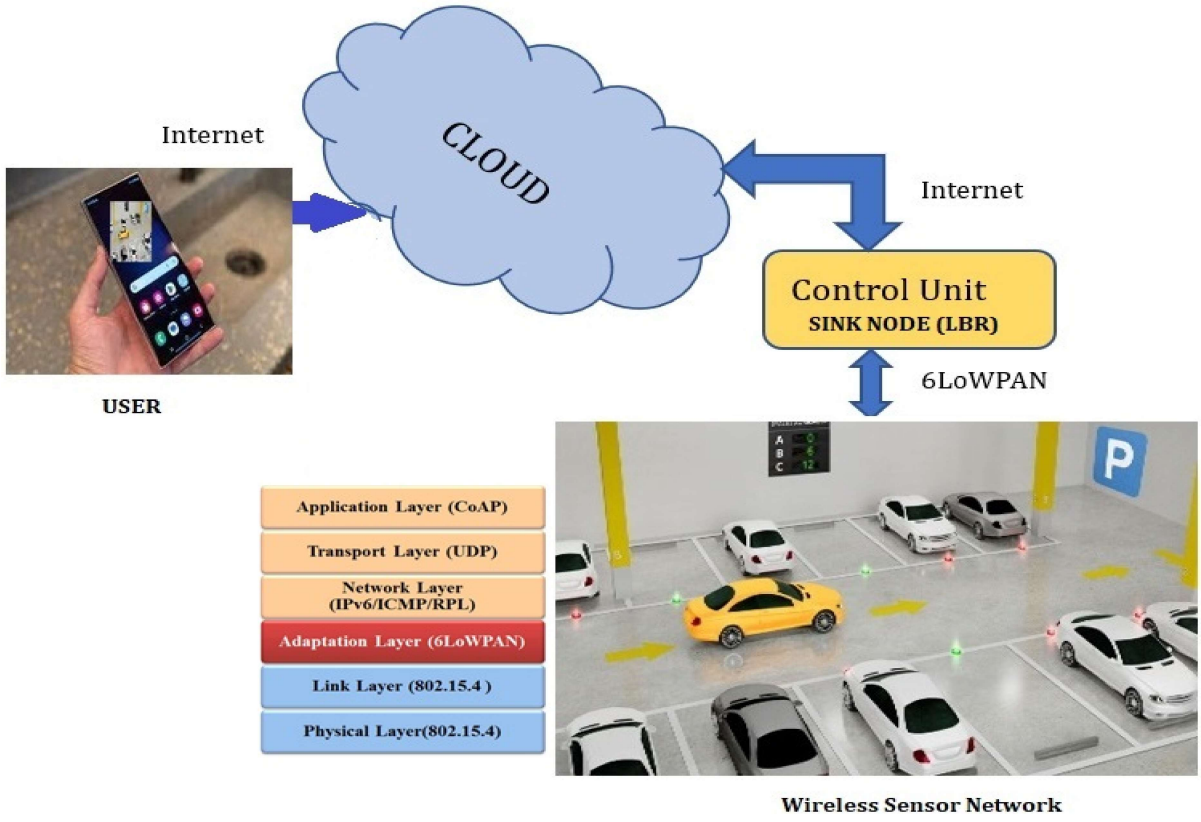


Figure 3.1: Smart city application scenario

Routing is the linchpin within a network that ensures efficiency and quality of service in IoT. The IPV6 Routing protocol for Low power and lossy networks (RPL) is standardised by the IETF to meet routing criteria in LLNs. The local sensor network uses the 6LoWPAN standard and RPL protocol for routing. The standard RPL selects routes based on single parameter or metric such as hop count or link quality. A holistic routing strategy with multiple metrics ensures better performance than a single metric based approach.

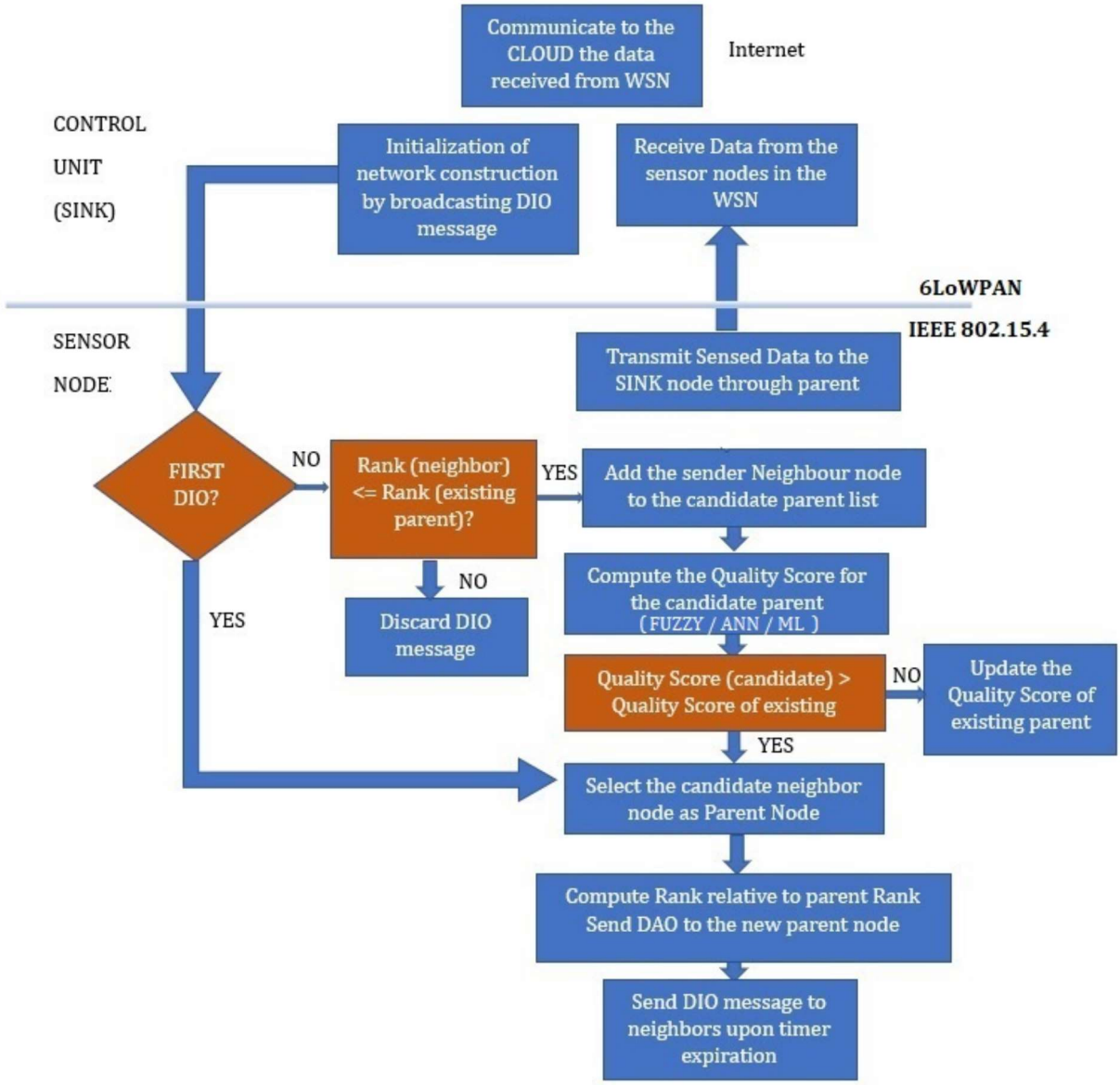


Figure 3.2: Flowchart depicting Sink-Sender communication and parent selection process

In this research our primary aim is to develop novel objective functions for RPL which consider multiple metrics while deciding the best parent nodes for optimal route to sink. To accomplish this goal, a multi-criteria decision method is required to evaluate the multiple metrics and reach the final decision. We have conducted this research in three main phases. In the initial phase, a Fuzzy Logic Inference system is designed to perform the multi-criteria decision

making. In the second phase,an Artificial Neural Network is devised to determine the best path. In the last stage, Machine learning regression techniques of K-Nearest neighbor, Linear Regression and Random Forest are explored owing to their growing popularity in problem solving. The techniques of Fuzzy Logic, ANN or Machine learning are designed to evaluate multiple link or node metrics and compute the Quality of neighboring nodes. The neighbor with highest quality is selected as parent node in route towards sink. The flowchart in figure 3.2 depicts the parent selection process at sender node and the sink-sender communication.
The efficacy of these techniques in selecting the best route by evaluating multiple routing metrics is studied by implementing them in network scenario. The method adopted to evaluate the performance is constant for the three approaches. This chapter describes the simulation setup. Here we elaborate on the routing metrics chosen, the operating system, HW specifications of the motes, the simulation setting and the performance parameters measured.

3.2 QOS Requirements in a network

As we proceed with the development of a improvised objective function for RPL, we need to consider the properties of a sustainable network infrastructure essential to meet Quality of Service. Based on these, we have to select the appropriate routing metrics for design consideration. In a network, the key QoS criteria are listed as follows:

1. Energy Efficiency: A network is sustainable if it is energy efficient. Nodes deployed are basically battery operated. So minimum energy consumption should happen during network operation and communication to ensure longer lifetime. Nodes which have higher residual energy must be involved in construction of route. The nodes with lower levels of energy should be avoided and not drained further. The routing process should be sensitive to the battery levels of nodes, in order to extend the lifetime. For any given network, energy efficiency can be determined by computing residual energy, power consumption (average) and network lifetime.

2. Reliability: In a network, sensor nodes collect crucial information and transmit it to the sink. A good route should ensure that all this information reaches the gateway or sink without any loss. The route has to be reliable with minimal number of packets getting dropped. The quality of links connecting the nodes in LLNs determines the reliability. It can be determined using link estimators such as RSSI (Received Signal Strength Indicator), Number of Packets Dropped, Packet Delivery Ratio (PDR) and expected number of

transmissions for successful delivery (ETX).

3. Real Time: The time a message takes to reach the sink should be as small as possible. For applications which operate in real time, this is a crucial factor. A good route should have low end-to-end delay or latency. In optimization, we generally observe a trade-off, which means if we try to improve one factor then some other factor degrades. This makes routing optimization a challenging task. For example, a good reliable route may take more hops, which adversely affects the latency. For sustainable architecture, main focus is to achieve a balance so that all the above criteria are met.

3.3 Choice of Routing Metrics

In RPL DODAG construction, the metric specified by Objective Function governs the selection of best parent node in route towards the sink . The routing metric largely influences the QoS and overall network performance. This section describes the routing metrics carefully identified for the design of novel objective functions keeping in mind the QoS requirements listed above. Each chosen metric corresponds to a vital attribute of the candidate neighbor node. We have chosen two link and two node metrics. Considering the QoS requirements, the four metrics we utilize in our work are:

1. **Expected transmission count (ETX)**: It is a measure of the link reliability. ETX represents the expected number of transmissions required for a packet to be successfully delivered to its destination node (DAG root or sink). A good and reliable link should have a low ETX value. It is given by formula

$$ETX = \frac{1}{df\ x\ dr} \tag{3.1}$$

where df and dr are the probabilities of successful delivery of a forward packet and acknowledgement receipt by the sender respectively.

2. **Residual Energy (RE)**: Initially at time of deployment, a node has full battery level which is denoted by Eo. During various network operations certain amount of energy is consumed. The energy consumed is given by Ec. The residual energy (RE) is given as

$$RE = Eo - Ec \tag{3.2}$$

Considering the node residual energy helps to avoid repeated selection of a node having low energy levels as the best parent. This ensures longer network lifetime.

3. **Delay**: It is the average time taken by a packet to reach its destination. This factor directly governs the overall network latency.

4. **Hop Count (HC)**: Represents the number of hops between a candidate parent node and the root node in a DODAG. Applications that need to do processing in real-time require low hop count to ensure low delay.

The methodology of applying the techniques of Fuzzy Logic, ANN and Machine Learning algorithms to make the optimal path decision is based on these four routing metrics is explained in the next chapters. These four metrics are fed as inputs to the Fuzzy Inference System or ANN or ML algorithms.

3.4 Operating System

To test the practical applicability of the proposed design, it has to be integrated into an operating system available for IoT motes. Several operating systems are developed and available for IoT applications. Some of the best IoT operating systems are TinyOS [96], Contiki OS [97], RIOT [98], Tizen [99], OpenWrt [100], Mbed OS [101], T-Kernel [102] and LiteOS [103].
Contiki is a state-of-the-art, open-source operating system for connecting low-power, low-cost micro controllers to the internet and a toolbox for creating complicated wireless systems [97]. Contiki is designed to adhere to the highest Internet standards, such as full support for IPv4 and IPv6. It's written in C language to provide a rapid programming environment in a single download, and it includes an active community to make any user feel at ease. Considering the popularity and its advantages over others, we decided to use Contiki OS for implementing our proposed designs.
Contiki operating system consists of the kernel, libraries, the program loader, and a set of processes. It provides libraries for memory allocation, linked list manipulation and communication abstractions. It is the first operating system that provided IP communication. It is developed in C, all its applications are also developed in C programming language, and therefore it is highly portable to different architectures like Texas Instruments MSP430. Contiki is an event-driven system in which processes are implemented as event handlers that run to completion. A Contiki system is partitioned into two parts: the core and the loaded programs. The core consists of the Contiki kernel, the program loader, the language run-time, and a communication stack

with device drivers for the communication hardware. The Program loader loads the programs into the memory and it can either obtain it from a host using communication stack or can obtain from the attached storage device such as EEPROM. The Contiki operating system provides modules for different tasks (layers). It provides the routing modules in a separate directory "contiki/core/net/rpl" and consists of a number of files. The crucial files in Contiki OS during design and simulation are listed below:

1. /contiki-2.7/core/net/rpl/rpl-conf.h. The configuration option such as the objective function used during simulation can be modified using this file.

2. /contiki-2.7/core/net/rpl/rpl-of0.c. An implementation of RPL's objective function 0 is in this file.

3. /contiki-2.7/core/net/rpl/rpl-mhrof.c. The Minimum Rank with Hysteresis Objective Function (MRHOF) is implemented in this file.

4. /contiki-2.7/core/net/rpl/rpl-fuzzy.c. The proposed Fuzzy Logic based Objective Function is implemented in this file.

5. /contiki-2.7/core/net/rpl/rpl-NNof.c. The proposed ANN based Objective Function is implemented in this file.

6. /contiki-2.7/core/net/rpl/rpl-ML-of.c. The proposed Machine Learning based Objective Function is implemented in this file.

7. /contiki-2.7/examples/ipv6/rpl-collect/collect-common.c. This file is used to change the data rates during simulation.

8. /contiki-2.7/core/net/rpl/rpl-dag.c. This file contains the Logic for Directed Acyclic Graphs in RPL

9. /contiki-2.7/core/net/rpl/rpl-icmp6.c. The icmpv6 message generation is implemented in this file.

The C programs for the proposed Fuzzy Logic, ANN and ML algorithms are written, tested and debugged using Visual Studio. The finalised designs are integrated into the Contiki 2.7 Operating System. It is an Linux based OS. This OS comes with an in-built simulator COOJA which emulates the network scenario.

Cooja is a Java-based simulator designed for simulating sensor networks running the Contiki

sensor network operating system [104]. The simulator is implemented in Java but allows sensor node software to be written in C. One of the differentiating features is that Cooja allows for simultaneous simulations at three different levels: Network Level, Operating System Level and Machine code instruction level [9]. Cooja can also run Contiki programs either compiled natively on the host CPU or compiled for MSP430 emulator. In Cooja all the interactions with the simulated nodes are performed via plugins like Simulation Visualizer, Timeline, and Radio logger. It stores the simulation in an xml file with extension 'csc' (Cooja simulation configuration). This file contains information about the simulation environment, plugins, the nodes and its positions, random seed and radio medium etc.Cooja Simulator runs the Contiki applications whose files are placed in another directory and may also contain a "project-conf.h" file which provides the ability to change RPL parameters in one place.

The sender nodes and sink nodes can be added to the network and deployed in random or grid topology. The network view in COOJA is shown in figure 3.3.

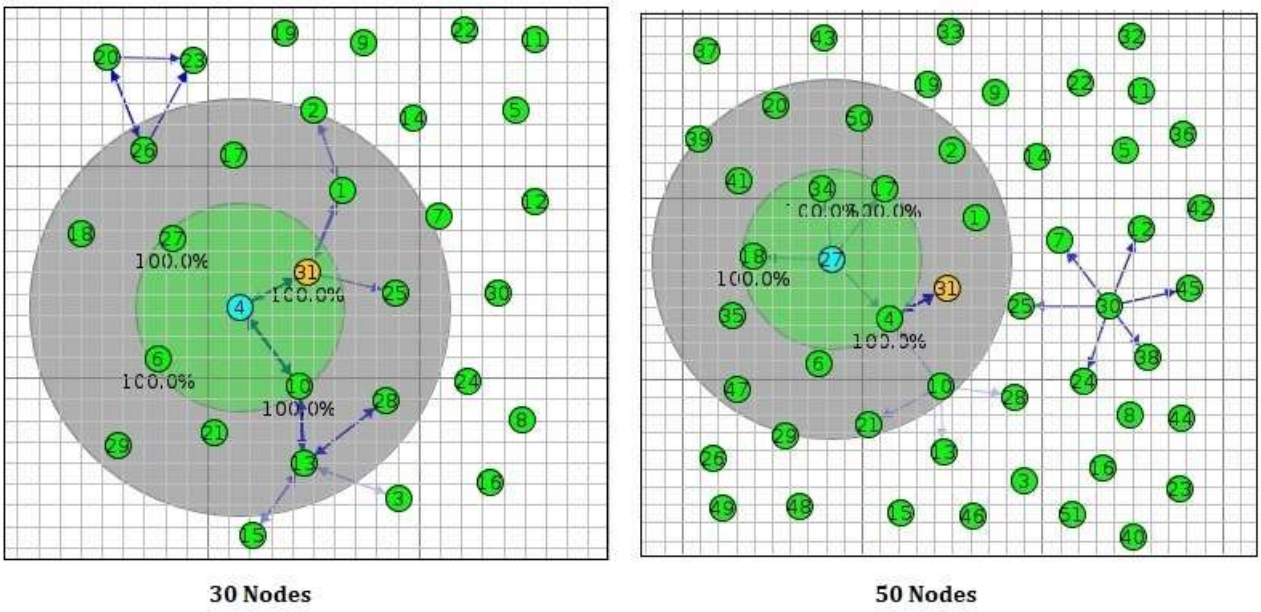


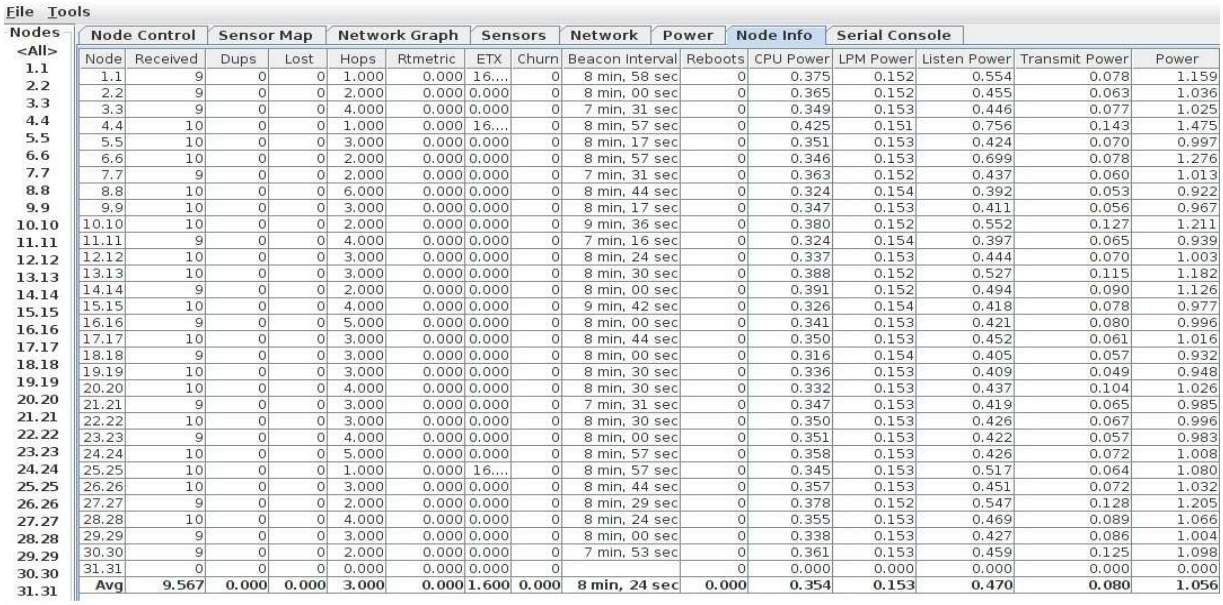
Figure 3.3: Network view in COOJA simulator 30 and 50 nodes

The parameter used during all the simulations are shown in table 3.1

Table 3.1: Simulation Parameters

Area of Deployment	400m * 400m
Simulator	Contiki OS - COOJA
Transmission Range	50m
Number of Sender Nodes	30, 50, 70, 90, 100 static nodes
Number of Sink Nodes	1
MAC Layer	Contiki MAC
Topology	Random Deployment
Data Transmission	1ppm, 2ppm, 5 ppm and 10 ppm
Emulated Nodes	T-mote Sky or Zolertia Z1 or WiSMote
Transmission Ratio	100%

Contiki OS provides an application called Collect-View for obtaining network performance parameters. During Simulation, using the Collect-View tool we can get the LPM power, Transmit power, Receive power, CPU power, Average power consumption, Successfully delivered packets and Lost packets. Besides Collect View application, Perl Scripts are written to extract some of the performance parameters like network latency, setup time and control overhead. The Perl Scripts are executed on the testlogs generated during the simulations. A sample of the Collect-View application window is shown in figure 3.4.


File Tools

Nodes

<All>
1.1
2.2
3.3
4.4
5.5
6.6
7.7
8.8
9.9
10.10
11.11
12.12
13.13
14.14
15.15
16.16
17.17
18.18
19.19
20.20
21.21
22.22
23.23
24.24
25.25
26.26
27.27
28.28
29.29
30.30
31.31

Node Control | Sensor Map | Network Graph | Sensors | Network | Power | Node Info | Serial Console

Node	Received	Dups	Lost	Hops	Rtmetric	ETX	Churn	Beacon Interval	Reboots	CPU Power	LPM Power	Listen Power	Transmit Power	Power
1.1	9	0	0	1.000	0.000	16....	0	8 min, 58 sec	0	0.375	0.152	0.554	0.078	1.159
2.2	9	0	0	2.000	0.000	0.000	0	8 min, 00 sec	0	0.365	0.152	0.455	0.063	1.036
3.3	9	0	0	4.000	0.000	0.000	0	7 min, 31 sec	0	0.349	0.153	0.446	0.077	1.025
4.4	10	0	0	1.000	0.000	16....	0	8 min, 57 sec	0	0.425	0.151	0.756	0.143	1.475
5.5	10	0	0	3.000	0.000	0.000	0	8 min, 17 sec	0	0.351	0.153	0.424	0.070	0.997
6.6	10	0	0	2.000	0.000	0.000	0	8 min, 57 sec	0	0.346	0.153	0.699	0.078	1.276
7.7	9	0	0	2.000	0.000	0.000	0	7 min, 31 sec	0	0.363	0.152	0.437	0.060	1.013
8.8	10	0	0	6.000	0.000	0.000	0	8 min, 44 sec	0	0.324	0.154	0.392	0.053	0.922
9.9	10	0	0	3.000	0.000	0.000	0	8 min, 17 sec	0	0.347	0.153	0.411	0.056	0.967
10.10	10	0	0	2.000	0.000	0.000	0	9 min, 36 sec	0	0.380	0.152	0.552	0.127	1.211
11.11	9	0	0	4.000	0.000	0.000	0	7 min, 16 sec	0	0.324	0.154	0.397	0.065	0.939
12.12	10	0	0	3.000	0.000	0.000	0	8 min, 24 sec	0	0.337	0.153	0.444	0.070	1.003
13.13	10	0	0	3.000	0.000	0.000	0	8 min, 30 sec	0	0.388	0.152	0.527	0.115	1.182
14.14	9	0	0	2.000	0.000	0.000	0	8 min, 00 sec	0	0.391	0.152	0.494	0.090	1.126
15.15	10	0	0	4.000	0.000	0.000	0	9 min, 42 sec	0	0.326	0.154	0.418	0.078	0.977
16.16	9	0	0	5.000	0.000	0.000	0	8 min, 00 sec	0	0.341	0.153	0.421	0.080	0.996
17.17	10	0	0	3.000	0.000	0.000	0	8 min, 44 sec	0	0.350	0.153	0.452	0.061	1.016
18.18	9	0	0	3.000	0.000	0.000	0	8 min, 00 sec	0	0.316	0.154	0.405	0.057	0.932
19.19	10	0	0	3.000	0.000	0.000	0	8 min, 30 sec	0	0.336	0.153	0.409	0.049	0.948
20.20	10	0	0	4.000	0.000	0.000	0	8 min, 30 sec	0	0.332	0.153	0.437	0.104	1.026
21.21	9	0	0	3.000	0.000	0.000	0	7 min, 31 sec	0	0.347	0.153	0.419	0.065	0.985
22.22	10	0	0	3.000	0.000	0.000	0	8 min, 30 sec	0	0.350	0.153	0.426	0.067	0.996
23.23	9	0	0	4.000	0.000	0.000	0	8 min, 00 sec	0	0.351	0.153	0.422	0.057	0.983
24.24	10	0	0	5.000	0.000	0.000	0	8 min, 57 sec	0	0.358	0.153	0.426	0.072	1.008
25.25	10	0	0	1.000	0.000	16....	0	8 min, 57 sec	0	0.345	0.153	0.517	0.064	1.080
26.26	10	0	0	3.000	0.000	0.000	0	8 min, 44 sec	0	0.357	0.153	0.451	0.072	1.032
27.27	9	0	0	2.000	0.000	0.000	0	8 min, 29 sec	0	0.378	0.152	0.547	0.128	1.205
28.28	10	0	0	4.000	0.000	0.000	0	8 min, 24 sec	0	0.355	0.153	0.469	0.089	1.066
29.29	9	0	0	3.000	0.000	0.000	0	8 min, 00 sec	0	0.338	0.153	0.427	0.086	1.004
30.30	9	0	0	2.000	0.000	0.000	0	7 min, 53 sec	0	0.361	0.153	0.459	0.125	1.098
31.31	0	0	0	0.000	0.000	0.000	0		0	0.000	0.000	0.000	0.000	0.000
Avg	**9.567**	**0.000**	**0.000**	**3.000**	**0.000**	**1.600**	**0.000**	**8 min, 24 sec**	**0.000**	**0.354**	**0.153**	**0.470**	**0.080**	**1.056**

Figure 3.4: Collect View application window

3.5 Specifications of Sensor Nodes

A number of sensor motes are available in the market ready to be deployed for IoT applications. Three most preferred sensor motes are T-mote Sky, Zolertia Z1 and the WisMote. Contiki OS provides support for these three motes in addition to several others. This section lists the hardware specifications of the motes. We will start with the Zolertia Z1 which is used in our simulations.

Zolertia Z1: The Zolertia Z1 mote is equipped with a second generation MSP430F2617 low power microcontroller, which features a powerful 16-bit RISC CPU at 16MHz clock speed, built-in clock factory calibration, 8KB RAM and a 92KB Flash memory. Also includes the well known CC2420 transceiver, IEEE 802.15.4 compliant, which operates at 2.4GHz with an effective data rate of 250Kbps. Z1 hardware selection guarantees the maximum efficiency and robustness with low energy cost. Z1 comes with built-in digital sensors ready to work: a digital programmable accelerometer (ADXL345) and a digital temperature sensor (TMP102) are on

the main board. Many sensors can be connected to the Z1 mote, Phidgets are supported out-of-the-box, also I2C sensors can be connected to the Ziglet defacto port, and UART/ADC/SPI communication is available. The commercially available Z1 mote is shown in figure 3.5.

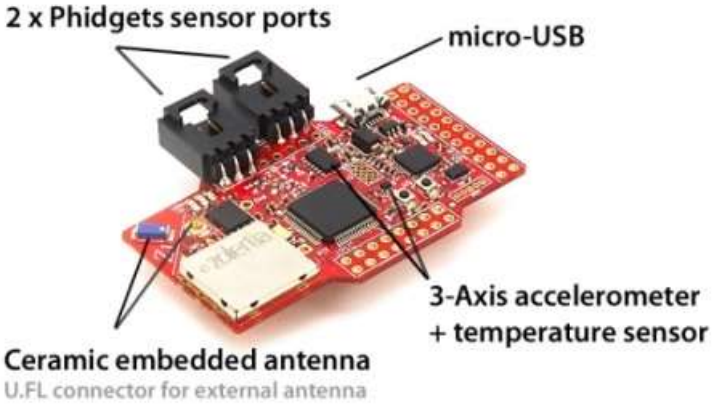


Figure 3.5: Zolertia Z1 sensor mote

The CC2420 chip is a true single-chip 2.4 GHz IEEE 802.15.4 compliant RF transceiver with baseband modem and MAC support. It features a DSSS baseband modem with 2 MChips/s and 250 kbps effective data rate. It is suitable for both RFD and FFD operation. It has a low current consumption (RX: 18.8 mA, TX: 17.4 mA), Low supply voltage (2.1 – 3.6 V) with integrated voltage regulator or Low supply voltage (1.6 – 2.0 V) with external voltage regulator.

T-mote Sky/TelosB platform:It features on board the 250kbps 2.4GHz IEEE 802.15.4 Chipcon Wireless Transceiver chip. It is equipeed with 8MHz Texas Instruments MSP430 microcontroller (10k RAM, 48k Flash). It comes with integrated ADC, DAC, Supply Voltage Supervisor, and DMA Controller. An integrated onboard antenna with 50m range indoors / 125m range outdoors is used for communication. Integrated Humidity, Temperature, and Light sensors are provided on board. This platform is Contiki's historical default platform. It is mostly used for emulation purposes in Cooja.

WiSMote: The WiSMote is equipped with a MSP430X series CPU, and support more memory than 64K. It has a 16KB RAM and a 256 KB Flash ROM. Also includes the well known CC2420 transceiver, IEEE 802.15.4 compliant, which operates at 2.4GHz with an effective data rate of 250Kbps. The platform supports an extendable memory of 1-8 MB (External Flash).

The design of Fuzzy Logic could be tested with all three motes. The ANN and machine learning algorithms needed higher memory, thus could be implemented successfully with Zolertia Z1 and WiSMotes.

3.6 Performance Parameters Measured

The following parameters are measured from the simulations to determine the efficiency of the network implementation:

- Packet Delivery Ratio (PDR): In wireless sensor networks, the sensed data is transmitted continuously to the root node by individual motes. If data rates are high, then some packets are dropped owing to buffers being full and collisions. The ratio of packets actually received at the root node to the total packets sent is called Packet Delivery ratio. The PDR is expressed by following equation:

$$PDR = \frac{Total\ Packets\ Received}{Total\ Packets\ Sent} \times 100 \tag{3.3}$$

- Average Power Consumption (Pavg): Hundreds of sensor motes deployed within a network consume varying amount of power depending on their activity, placement, topology and distance from root. The network lifetime is inversely proportional to the average of energy consumed by all the motes. Lower the power consumed, longer will be the network lifetime. The power consumption Pavg is computed by measuring the LPM power, CPU power, Transmit Power and Listen Power and averaging them for all nodes. The average power consumption of each node is sum of the average power consumption in CPU state (P_CPU) that is activated whenever the node is active, LPM state (P_LPM) that is activated when the sensor node goes to Low Power Mode, Rx state(P_Rx) that node is active in the radio receive mode and Tx state(P_Tx)that radio is active in transmit mode. Average power consumption is computed as shown in equation 3.5 [77],[81].

$$P = P_CPU + P_LPM + P_Rx + P_Tx \tag{3.4}$$

$$Average\ Power\ Consumption = \frac{\sum_{n=1}^{N} Pn}{N} \tag{3.5}$$

  where N is the total number of nodes in network.

- Network Latency: The time it takes for a data packet to travel from sender node to the receiver sink node is called end-to-end delay or network latency. If we average the delays of all the nodes within a network, we obtain average network latency as given below in equation 3.7 [77],[81]: The total latency is given by equation 3.6 [77],[81]:

$$Total\ Latency = \sum_{k=1}^{n} (Recv\ time\ (k) - Sent\ time\ \ (k)) \tag{3.6}$$

$$Average\ network\ latency = \frac{Total\ Latency}{Total\ Packets\ Received} \tag{3.7}$$

- Control Overhead: It is the total number of control messages transmitted by nodes during DODAG formation and to select the best parent between candidate neighbors. The control messages consist of the DAO, DIS and DIO messages. The control overhead is given by equation 3.8 [77],[81]

$$Control\ Overhead = \sum_{1}^{n} DIO + \sum_{1}^{n} DIS + \sum_{1}^{n} DAO \tag{3.8}$$

  The control messages are essential to circulate the routing information updates. But in excess can adversely affect the performance due to congestion and increased delay. To avoid congestion, the number of control messages should be restricted to minimal number. Equations 3.4 to 3.8 are as per the standard specified in the RFC 6550, RFC 6551 and followed by researchers working in the area of RPL protocol.

3.7 Conclusion

In this chapter we have described the simulation setup used to validate the proposed designs. ETX, Delay, Hop Count and residual energy are chosen as the inputs to the Fuzzy Inference System, Artificial Neural Network and Machine Learning algorithms designed in this research. The Contiki 2.7 Operating system is selected for implementation and the in-built network emulator COOJA is used for running the simulations. The simulation parameters and the specifications of sensor motes are listed. The performance parameters to be monitored are Packet delivery ratio, network latency, average power consumption and control overhead. These parameters help us to estimate the Quality of Service in deployed network.

The design of the Fuzzy Inference System, Artificial Neural Network and Machine Learning algorithms for deciding the best parent node in optimal route towards sink is detailed in the next three chapters.

Chapter 4

Design of Efficient Objective Function for RPL using Fuzzy Logic

4.1 Introduction

The IPv6 Routing protocol for LLNs (RPL) was standardised by IETF to enable the movement of IPv6 datagrams in networks which consists of resource-limited sensor devices and lossy links. RPL gained immediate popularity in the community with being included in the IP stack of major operating systems like Contiki OS and TinyOS. The standard version of RPL selected routes considering either Hop count or the expected number of re-transmission. The other crucial factors of nodes and links like remnant energy, delay, throughput, buffer occupancy and received signal strength were overlooked. The key focus of this research is to overcome the single metric limitation in RPL. Our aim is to improve the functionality of RPL protocol by considering multiple node and link metrics in making a holistic routing decision. To shortlist the best parent node based on multiple metrics, a mathematical technique is essential which will establish the relation between the metrics and suitability of a particular node. The conventional techniques of Multiple Criteria Decision Making (MCDM) such as AHP [105], ANP, WPM and TOPSIS involve matrix multiplications, are computation extensive and cannot be executed by the low-end micro controllers present in sensor nodes [106]. In problem scenarios where mathematical models cannot be applied, the domain of soft computing comes to the rescue. In this chapter, we have applied the technique of Fuzzy Logic to perform the multi-criteria decision making.
The concept of Fuzzy logic was proposed by Lotfi Zadeh in 1965 [107], and was applied in controlling steam engines by Mamdani in 1974 [108]. Fuzzy logic became practically useful in 1990s where many leading companies started using Fuzzy logic to design smart appliances.

Fuzzy logic closely represents the reasoning or decision-making ability of a human mind. Unlike a traditional computer, Fuzzy logic deals with a range of possibilities between 'TRUE' and 'FALSE'. Fuzzy logic is an accurate method widely used to combine several values to obtain a single precise output.

In this chapter, we describe the process of designing a fuzzy logic controller which accepts the multiple metrics and computes a quality score for candidate neighbour nodes. The best parent can then be determined on basis of its quality score. Initially, a novel objective function RPL-FZ is designed which operates with three routing metrics. Later, a fourth metric is included and this extended version is termed as OF-FZ. The performance of both RPL-FZ and OF-FZ is compared with standard OF0 and MRHOF versions.

4.2 Research Objective

The main aim of this research is to design a routing strategy that overcomes the single metric-based routing limitation of existing standard RPL protocol. It is modeled as a multi-constrained optimal path problem, where best route decision is based on multiple routing metrics and Fuzzy Logic technique is employed to determine the optimal route toward the sink node.

The objectives of research is: Apply the technique of Fuzzy Logic to design a routing strategy for RPL that selects the best route based on three (ETX, residual energy and delay) and four routing metrics (ETX, residual energy, delay and hop count).

4.3 Fuzzy Inference System

In this chapter, the detailed design of proposed Objective Functions RPL-FZ and OF-FZ is explained. The RPL standard allows the designer flexibility in choice of metrics for Objective Function design. Accordingly, in the initial phase we opted to use three routing metrics in designing the Objective Function RPL-FZ and later included the fourth metric in OF-FZ. Both these implementations invoke a Fuzzy Inference System to determine the best parent node in route towards the root. A Fuzzy Inference System consists of the following components:

- **Fuzzifier**: This unit converts crisp inputs into fuzzy sets. Crisp inputs are defined with linguistic variables and denoted graphically using membership functions. Membership function depict the degree of membership of an input value in a particular fuzzy set. Some commonly used membership functions are Gaussian, Trapezoidal, Triangular, sigmoidal

and generalized Bell. The crisp input variables in this work are the chosen four routing metrics - ETX, Delay, Hop Count and Residual Energy as described in chapter 3.

- **Rule Base**: It is an exhaustive collection of all the rules that govern the decision-making process in the fuzzy inference system. The rules are defined using IF-THEN conditions and operators like AND, OR and NOT.

- **Inference Engine**: This component of the FIS determines the specific set of rules triggered by a given input. The fuzzy output is generated by applying these rules to the input. If the output depends on more than one rule than it is unified using aggregation method.

- **Defuzzifier**: The fuzzy output is transformed to an explicit crisp output during defuzzification. This is the final stage of a fuzzy logic system. Some common defuzzification methods employed are Bisector, Centroid, Smallest of Maximum, Mean of Maximum, Largest of Maximum and Weighted Average method

4.4 Membership Functions for Routing Metrics

The first step of fuzzy logic process is fuzzification. In this we convert crisp input quantities in terms of fuzzy sets. Crisp inputs are defined with linguistic variables and denoted graphically using membership functions. In this work, the crisp inputs are the routing metrics ETX, RE, HC and Delay. Here, we will be defining the linguistic variables and membership functions for these four metrics. For the four metrics a range of values are set. which are chosen considering the Quality-of-Service requirements of a Network.
The membership functions of Residual energy, Delay, ETX and HC are as shown in figure 4.1. The linguistic terms are quantified using membership functions. Trapezoidal membership functions are chosen for all the crisp inputs. Studies [109] have shown that trapezoidal functions perform better than triangular membership functions. In this application scenario, trapezoidal membership functions are chosen as they have a wider core. This allows a range of values to have a complete membership value of 1. On the contrary, triangular membership functions assign complete membership to a single value and thus fails to represent this application scenario accurately.

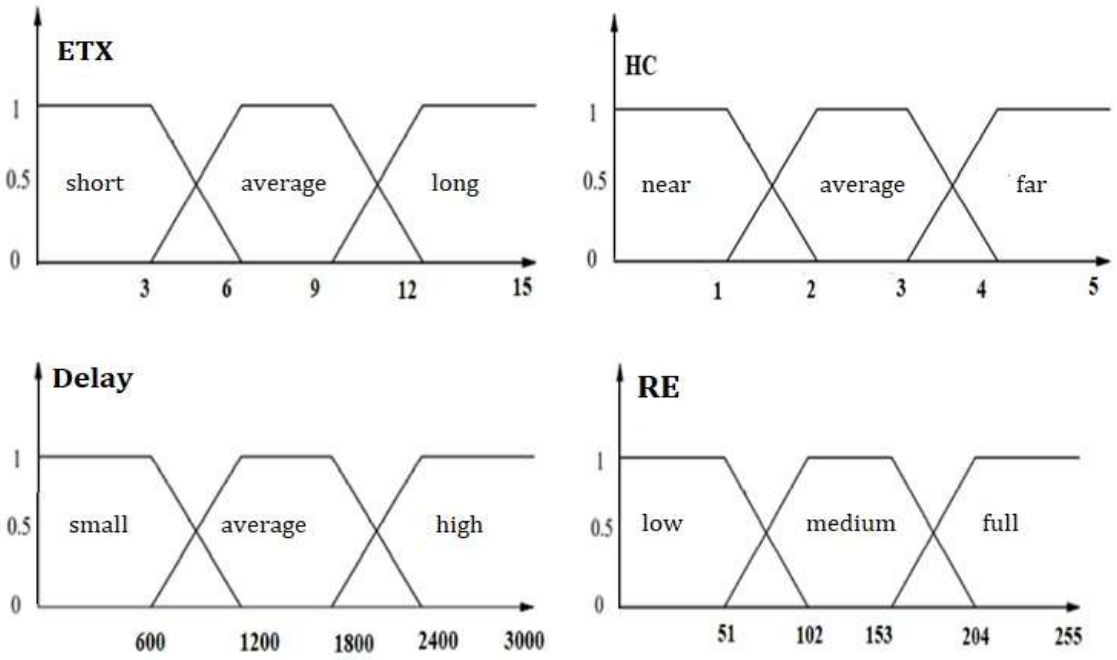


Figure 4.1: Membership Functions of ETX, HC, Delay and RE

- The ETX input is defined using linguistic variables - short, average and long. For ETX (Expected number of re-transmission), we choose a range of 1 to 15. A link with ETX value 1 to 3 is very good, since lesser number of re-transmissions signify faster delivery and good reliability. ETX between 3 to 12 is acceptable and above 12 is undesirable.
  The membership level in fuzzy set **SHORT** of ETX is defined by equation 4.1

$$.\mu\ (short)(\ ETX) = \begin{cases} 0 & ,\ ETX\ >\ 6 \\ \frac{6\ -\ ETX}{6\ -\ 3} & ,\ 3\ \leq ETX\ \leq 6 \\ 1 & ,\ ETX\ \leq 3 \end{cases} \tag{4.1}$$

  The membership level in fuzzy set **AVERAGE** of ETX is defined by equation 4.2.

$$\mu\ (average)(\ ETX) = \begin{cases} 0 & ,\ ETX\ <\ 3\ or\ ETX\ >\ 12 \\ \frac{ETX\ -\ 3}{6\ -\ 3} & ,\ 3\ \leq ETX\ \leq 6 \\ 1 & ,\ 6\ \leq ETX\ \leq 9 \\ \frac{12\ -\ ETX}{12\ -\ 9} & ,\ 9\ \leq ETX\ \leq 12 \end{cases} \tag{4.2}$$

The membership level in ETX fuzzy set **LONG** is defined by the equation 4.3.

$$\mu\ (long)(\ ETX) = \begin{cases} 0 & ,\ ETX\ <\ 9 \\ \frac{ETX\ -\ 9}{12\ -\ 9} & ,\ 9\ \leq ETX\ \leq 12 \\ 1 & ,\ ETX\ >12 \end{cases} \tag{4.3}$$

- Hop count is specified with linguistic variables near, average and far. For Hop Count we select a range of 1 to 5. Neighbor nodes which are 2 hops away from the Root node are very good. Neighbors with hops between 1 to 4 are considered average. Above 4 hops are undesirable as they are far from the Root node.

  The membership level in fuzzy set **NEAR** of Hop Count is defined by equation 4.4.

$$\mu\ (near)(\ HC) = \begin{cases} 0 & ,\ HC\ >\ 2 \\ \frac{2\ -\ HC}{2\ -\ 1} & ,\ 1\ \leq HC\ \leq 2 \\ 1 & ,\ HC\ \leq 1 \end{cases} \tag{4.4}$$

  The membership level in fuzzy set **AVERAGE** of Hop Count is defined by equation 4.5.

$$\mu\ (average)(\ HC) = \begin{cases} 0 & ,\ HC\ <\ 1\ or\ HC\ >\ 4 \\ \frac{HC\ -\ 1}{2\ -\ 1} & ,\ 1\ \leq HC\ \leq 2 \\ 1 & ,\ 2\ \leq HC\ \leq 3 \\ \frac{4\ -\ HC}{4\ -\ 3} & ,\ 3\ \leq HC\ \leq 4 \end{cases} \tag{4.5}$$

  The membership level in fuzzy set **FAR** of Hop Count is defined by equation 4.6.

$$\mu\ (far)(\ HC) = \begin{cases} 1 & ,\ HC\ >\ 4 \\ \frac{HC\ -\ 3}{4\ -\ 3} & ,\ 3\ \leq HC\ \leq 4 \\ 0 & ,\ HC\ \leq 3 \end{cases} \tag{4.6}$$

- The linguistic variables used to represent Delay are small, average and high. Range chosen for delay is 0 to 3000 units. A delay less than 600 units is considered very good. Delay values between 600 and 2400 are average and acceptable. Neighbor nodes with Delay values above 2400 are undesirable as parent nodes.

The membership level in fuzzy set **SMALL** of Delay is defined by equation 4.7.

$$\mu\ (small)(\ Delay) = \begin{cases} 0 & ,\ Delay\ >\ 1200 \\ \frac{1200\ -\ Delay}{1200\ -\ 600} & ,\ 600\ \leq Delay\ \leq 1200 \\ 1 & ,\ Delay\ \leq 600 \end{cases} \tag{4.7}$$

The membership level in fuzzy set **AVERAGE** of Delay is defined by equation 4.8.

$$\mu\ (average)(\ Delay) = \begin{cases} 0 & ,\ Delay\ <\ 600\ or\ Delay\ >\ 2400 \\ \frac{Delay\ -\ 600}{1200\ -\ 600} & ,\ 600\ \leq Delay\ \leq 1200 \\ 1 & ,\ 1200\ \leq Delay\ \leq 1800 \\ \frac{2400\ -\ Delay}{2400\ -\ 1800} & ,\ 1800\ \leq Delay\ \leq 2400 \end{cases} \tag{4.8}$$

The membership level in fuzzy set **HIGH** of Delay is defined by equation 4.9.

$$\mu\ (high)(\ Delay) = \begin{cases} 1 & ,\ Delay\ >\ 2400 \\ \frac{Delay\ -\ 1800}{2400\ -\ 1800} & ,\ 1800\ \leq Delay\ \leq 2400 \\ 0 & ,\ Delay\ \leq 1800 \end{cases} \tag{4.9}$$

- The input RE is specified using linguistic variables low, medium and full. When nodes are deployed, initially they have highest energy level of value 255. A neighbor node, who has a Residual Energy between 204 to 255 is a very good candidate for Parent. Nodes with RE between 204 and 51 are average and acceptable. Nodes whose residual energy is below 51 will not be chosen as Parent nodes.

  The membership level in fuzzy set **LOW** of Residual Energy is defined by equation 4.10.

$$\mu\ (low)(\ RE) = \begin{cases} 0 & ,\ RE\ >\ 102 \\ \frac{102\ -\ RE}{102\ -\ 51} & ,\ 51\ \leq RE\ \leq 102 \\ 1 & ,\ RE\ \leq 51 \end{cases} \tag{4.10}$$

  The membership level in fuzzy set **MEDIUM** of Residual Energy is defined by equation

4.11.

$$\mu\ (medium)(\ RE) = \begin{cases} 0 & ,\ RE\ <\ 51\ or\ RE\ >\ 204 \\ \frac{RE\ -\ 51}{102\ -\ 51} & ,\ 51\ \leq RE\ \leq 102 \\ 1 & ,\ 102\ \leq RE\ \leq 153 \\ \frac{204\ -\ RE}{204\ -\ 153} & ,\ 153\ \leq RE\ \leq 204 \end{cases} \tag{4.11}$$

The membership level in fuzzy set **FULL** of Residual Energy is defined by equation 4.12.

$$\mu\ (full)(\ RE) = \begin{cases} 1 & ,\ RE\ >\ 204 \\ \frac{RE\ -\ 153}{204\ -\ 153} & ,\ 153\ \leq RE\ \leq 204 \\ 0 & ,\ RE\ \leq 153 \end{cases} \tag{4.12}$$

The above equations 4.1 to 4.12 are our novelty and not defined in any existing literature. Once the membership equations are defined, the next step is to specify the Rule Base for the FIS. The Rule Base and De-fuzzification process in RPL-FZ and OF-FZ is detailed in the next sections.

4.5 Process for Best Parent Selection in Fuzzy Based-RPL implementation

The process for best parent selection amongst a number of candidate neighbors is shown in flowchart of figure 4.2. In RPL network, DIO messages are regularly transmitted after fixed periods determined by a trickle timer. Any node who has newly joined the network and has not received a DIO, can solicit from a neighbor by sending a DIS message. A node within the network, upon receiving the DIO message from a neighbor, checks if this neighbor is having a rank lower than itself. If true, then adds this neighbor as a potential candidate (CPN) to its candidate parent list. The link and node metrics such as Residual energy, Delay, ETX and hop count are extracted from the DIO message for this particular candidate parent. If this sender is the only candidate in parent list, then it is directly selected as the PPN (preferred parent node). Otherwise, if more than one candidate exists, then the FIS is invoked and the four extracted metrics are fed as input to the FIS. At the end of the defuzzification process, the computed QA scores of candidates are compared with QA score of prevailing PPN. A candidate parent with the highest QA score is selected as the next best parent or PPN. The new rank is calculated based on rank of selected parent. The previous PPN is replaced by the new one and an updated DIO message is multi-cast to neighbors. The DAO message is triggered when new parent is selected

and unicast to the selected parent node.

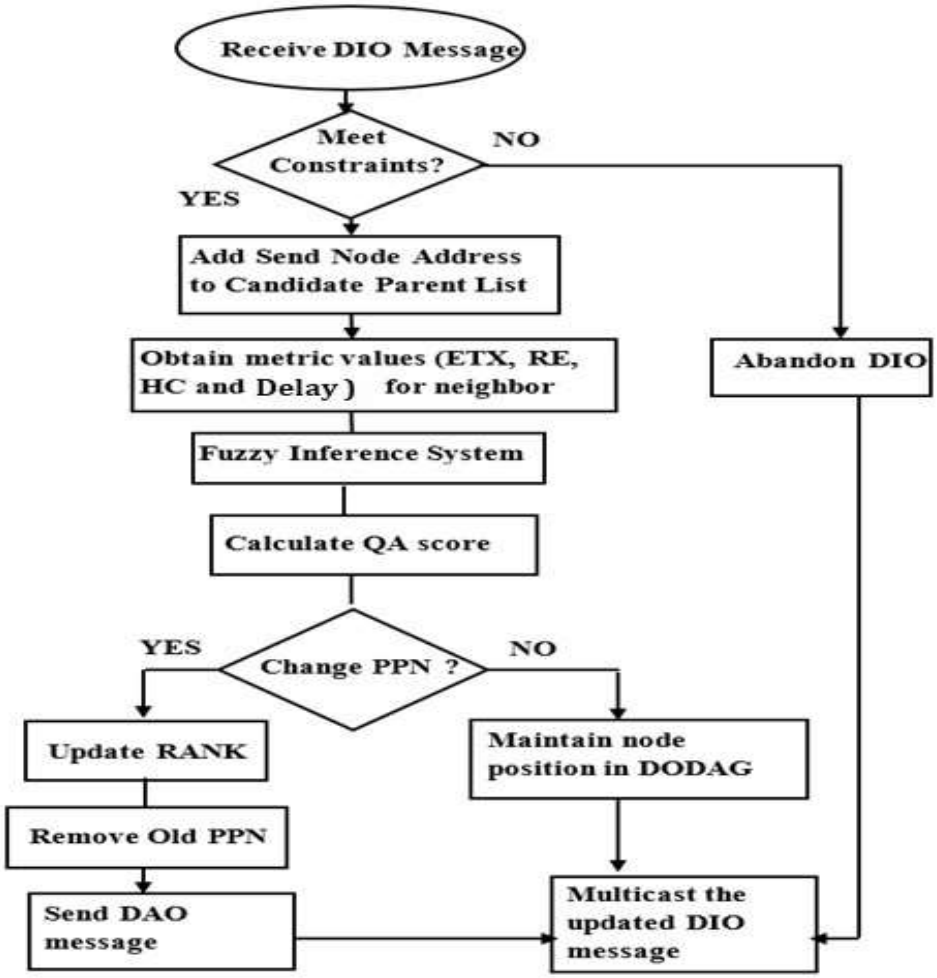


Figure 4.2: Process for Best Parent Selection using OF-FZ

4.6 Design of Fuzzy Logic based Objective function RPL-FZ

The design of the Fuzzy Inference System for novel objective function RPL-FZ which uses three routing metrics is detailed in this section.

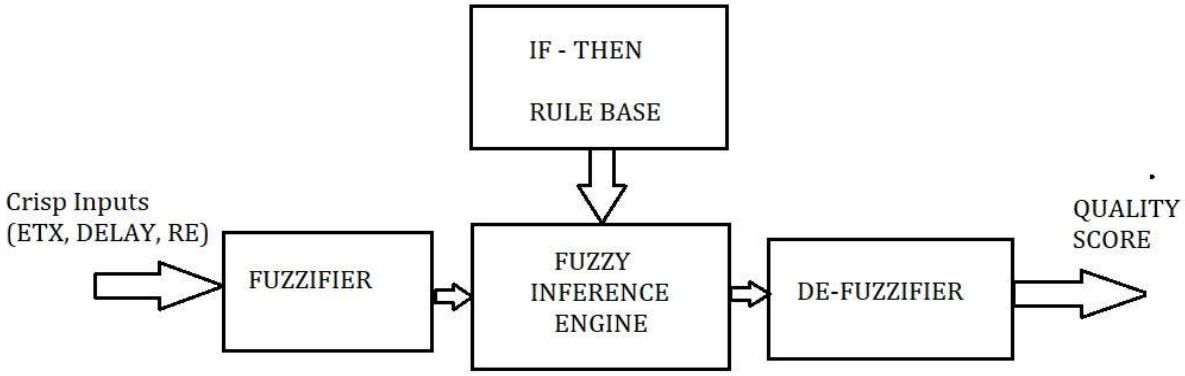


Figure 4.3: Fuzzy Inference System in RPL-FZ

Three routing metrics, namely ETX, Delay and Residual energy are chosen as the deciding criteria for best route. The three metrics are chosen considering the low power consumption, reliability and real time requirements of Quality of service. The membership functions for the three metrics are listed in Section 4.4. These three metrics are fed as input to the fuzzy controller and the output quality score for each neighbor node is calculated by the Fuzzy Inference System. A Rule Base has to be specified which relates the output to the inputs. Three metrics are involved and each metric has three fuzzy sets associated with it, thus the total number of Rules in the rule base are $3^3 = 27$. A set of IF-THEN fuzzy rules are stored in the database. These are applied to the fuzzy inputs in the fuzzy inference engine. The fuzzy inputs are connected by AND, OR and NOT fuzzy operators. The Fuzzy Inference System is shown in figure 4.3.
The output of the FIS, termed as Quality Score has associated with it seven fuzzy sets, namely very_bad, bad, degraded, average, satisfactory, good and excellent. The membership function of Quality is shown in figure 4.4.

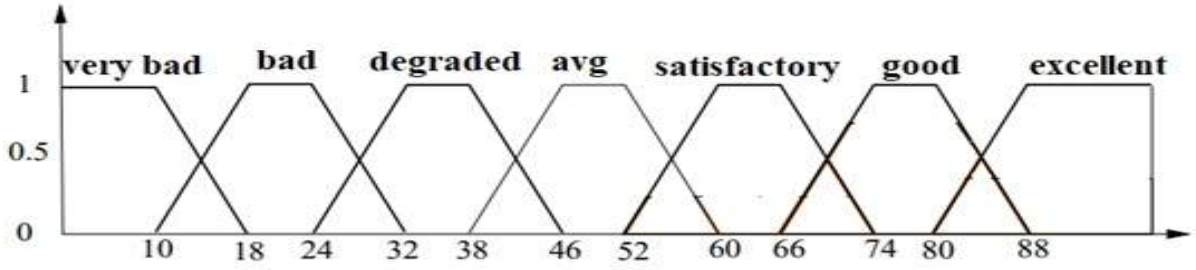


Figure 4.4: Membership Function for FIS output-Quality Score

Some examples of the rules as stored in database as listed below :

1. If ETX is short and Delay is small and RE is full then Quality is excellent

2. (If ETX is short and Delay is small and RE is medium then Quality is good) OR (If ETX is average and DELAY is small and RE is full then Quality is good)

3. (If ETX is short and DELAY is high and RE is full then Quality is satisfactory) OR (If ETX is long and DELAY is small and RE is full then Quality is satisfactory) OR (If ETX is average and DELAY is average and RE is full then Quality is satisfactory)

4. (If ETX is average and DELAY is average and RE is medium then Quality is Average) OR (If ETX is long and DELAY is small and RE is medium then Quality is Average) OR (If ETX is long and DELAY is high and RE is full then Quality is Average)

5. (If ETX is short and DELAY is average and RE is low then Quality is degraded) OR (If

ETX is average and DELAY is small and RE is low then Quality is degraded) OR (If ETX is long and DELAY is small and RE is low then Quality is degraded)

6. (If ETX is long and DELAY is average and RE is medium then Quality is bad) OR (If DELAY is high and ETX is long and RE is medium then Quality is bad)

7. If ETX is long and DELAY is high and RE is low then Quality is very_bad.

Table 4.1: Rules to Compute Quality Score in RPL-FZ

Inputs			Output
ETX	DELAY	RE	Quality Score
short	small	full	excellent
short	small	medium	good
short	small	low	average
short	average	full	good
short	average	medium	good
short	average	low	degraded
short	high	full	satisfactory
short	high	medium	average
short	high	low	degraded
average	small	full	good
average	small	medium	average
average	small	low	degraded
average	average	full	satisfactory
average	average	medium	average
average	average	low	degraded
average	high	full	satisfactory
average	high	medium	average
average	high	low	bad
long	small	full	satisfactory
long	small	medium	average
long	small	low	degraded
long	average	full	satisfactory
long	average	medium	bad
long	average	low	bad
long	high	full	average
long	high	medium	bad
long	high	low	very_bad

The Quality Score varies from 0 to 100. A Quality score is assigned to each candidate parent node. For a particular candidate node when the combination of metrics yields a good Quality score it can be selected as preferred parent. During defuzzification, the value of Quality is computed using the center of gravity (COG) defuzzification method:

$$M = \frac{\sum_{i=1}^{N} Vi \times \mu A\left(Vi\right)}{\sum_{i=1}^{N} \mu A\left(Vi\right)} \tag{4.13}$$

where M represents the defuzzified value, N corresponds to number of activated rules, Vi is the domain value of triggered rule i, and μA is the membership value. The 27 Rule combinations used to compute Quality Score are shown in Table 4.1.

4.6.1 Fuzzy Inference System incorporation into Contiki OS

The novel fuzzy logic based objective function has to be incorporated into operating systems meant for IoT motes to ensure that it can successfully be deployed in IoT networks. Contiki OS is a popular operating system build for wireless low power sensor motes used in Internet of Things applications. Contiki provides multitasking flexibility, has a built-in TCP-IP stack and requires around 10 KB of RAM and 30 KB of ROM. RPL is the standard routing protocol in Contiki. To improvise RPL, the proposed objective function has to be implemented into Contiki. In this section, we describe in brief the steps followed in this regard.

1. A new implementation file of the fuzzy based Objective function is created in core/net/rpl folder.
2. The created C file is added to makefile.rpl.
3. The new objective function has to be set in the conf.h file.
4. The file defining the fuzzy based objective function is named as rpl-of-fuzzy.c.
5. It consists of following functions:
    - calculate_rank to compute the rank by adding information
    - calculate_etx_path_metric to compute ETX of neighbor
    - calculate_latency_path_metric to compute the latency of overall path to DAG root
    - calculate_energy_path_metric to compute the energy of the neighbor
    - calculate_fuzzy_metric to compute the Quality score metric using the 3 selected routing metrics
    - best_parent to compute best parent based on ranks computed by the function calculate_fuzzy_metric
    - update_metric_container: this function is used to update the metric container.
6. The length of the DIO message is extended as the metric-container contains three routing metrics in case of RPL-FZ.

4.6.2 Simulation Results of RPL-FZ

In this section, we do the performance evaluation of the proposed objective function RPL-FZ. It is compared against the standard Objective functions OF0 and MRHOF. The three objective functions are evaluated under conditions of varying data transmission rates, and varying network density.

The COOJA simulator, which is an inbuilt tool available with CONTIKI-2.7 Operating system is used for performing the network simulations [104]. COOJA is an open-source tool ideal for checking implementation functionality by performing network simulations before actual deployment on sensor devices. A network with area of deployment 400m x 400m in random topology with many sender nodes and a single sink node in created in COOJA environment for simulations to be carried out. The test logs generated during the simulation in COOJA are saved. The values of power consumption and packet delivery ratio are obtained from collect view environment of COOJA. The delay and control overhead are obtained from the generated test logs by writing PERL scripts.

Scenario 1: Network Scalability

In this section, we evaluate the Objective functions RPL-FZ, OF0 and MRHOF by varying the density of nodes in the network from 30 to 100. A single sink node is used to ensure that network is strained. The sink node and sender nodes are static. Data transmission rate is 1 packet/min. The packet size is 128 bytes. The graphs of Average Power consumption, Packet Delivery Ratio, Latency and traffic Overhead are plotted in figure 4.5.

Simulation Results

Average power consumption: The graph is shown in figure 4.5. As number of nodes increase, average consumption also increases. Each node transmits one packet of 128 bytes per minute. 30 nodes will generate 30 packets per min.

When density rises to 100 nodes, three times higher data is being generated in the network. When density increases, the nodes have to becomes parents to a greater number of nodes. Thus, nodes have to be in ON mode for longer time, resulting in more energy consumption. RPL-FZ is found to give better performance by consuming 5% and 9% lesser energy as compared to OF0 and MRHOF respectively at density of 100 nodes. RPL-FZ makes the choice of preferred parent based on the residual energy of node, which considerably improves the average energy consumption and lifetime.

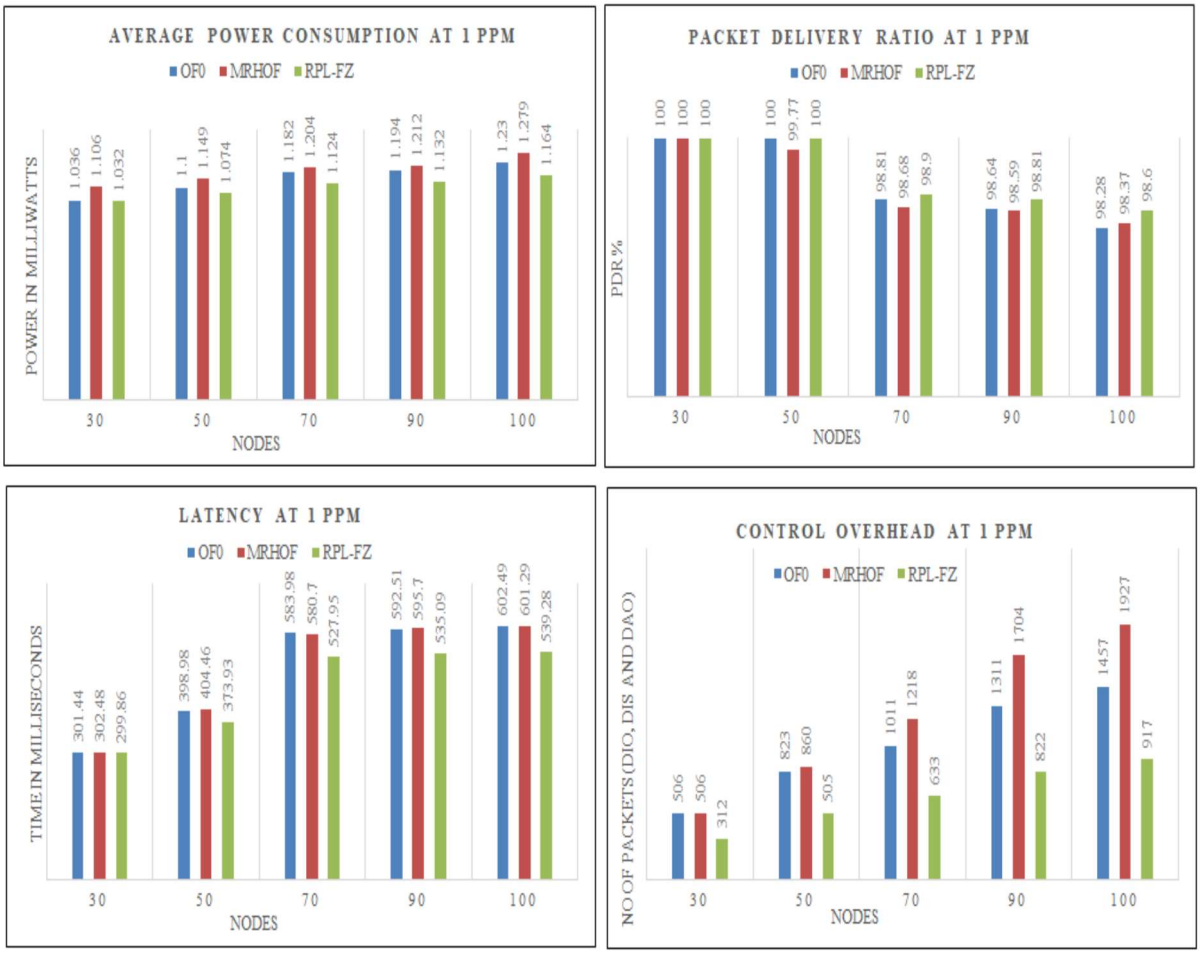


Figure 4.5: Performance of RPL-FZ versus OF0 and MRHOF for varying network density

Packet Delivery Ratio: The Packet delivery with RPL-FZ is 100% in a network with 30 nodes and 98.6 % as density is increased 100. Whereas OF0 and MRHOF are seen to lose more number of packets during transit. In terms of packet delivery success, RPL-FZ is seen to perform marginally better than OF0 and MRHOF at 1 ppm data rate. The table 4.2 shows the improvements in power consumption and PDR with RPL-FZ at 1 ppm as node density is varied from 30 to 100.

Table 4.2: Improvement in Avg Power Consumption and PDR with RPL-FZ over OF0 and MRHOF with varying node density at 1 ppm

	Average Power Consumption					Packet Delivery Ratio				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
30	1.036	1.106	1.032	Marginal	7%	100	100	100	Marginal	
50	1.1	1.149	1.074	2%	7%	100	99.77	100	Marginal	
70	1.182	1.204	1.124	5%	7%	98.81	98.68	98.9	Marginal	
90	1.194	1.212	1.132	5%	7%	98.64	98.59	98.81	Marginal	
100	1.23	1.279	1.164	5%	9%	98.28	98.37	98.6	Marginal	
Avg	1.1484	1.19	1.1052	4%	7%	99.146	99.082	99.262	Marginal	

Network Latency: The delay metric used in design of RPL-FZ ensures faster routes being chosen towards the root node. Both OF0 and MRHOF lack this advantage. The latency in RPL-FZ is 300 ms at 30 nodes and rises to 539 ms at 100 nodes. Whereas, OF0 and MRHOF have latency of above 600 ms at density of 100 nodes. RPL-FZ gives 10% lower latency over OF0 and MRHOF at 100 nodes. The table 4.3 shows the improvements in latency and control overhead with RPL-FZ at 1 ppm as node density is varied from 30 to 100.

Control Overhead: In standard RPL, the DAO messages are generated upon expiration of DelayDAO timer. The expiration of this timer happens at regular intervals leading to surplus DAO messages being generated in the network, which may not add any value. In this work, the DelayDAO timer expiration is triggered only upon the selection of a new parent node. This ensures that DAO messages are generated only when crucial routing updates must be sent and avoids unwanted overhead. The DIO messages, which carry the crucial routing information are generated on expiration of the Trickle timer. This timer expires at pre-defined fixed intervals and after a change in parent node. Together DIO and DAO ensure that updated routing information is circulated within the network. This leads to significant amount of reduction in the control overhead due to the mechanism adopted in transmitting DAO and DIO messages. The control overhead in RPL-FZ is significantly reduced to the tune of 38% and 49% over OF0 and MRHOF respectively.

Table 4.3: Improvement in Network Latency and Control Overhead with RPL-FZ over OF0 and MRHOF with varying node density at 1 ppm

	Network Latency (ms)					Control traffic overhead (Packets)				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
30	301.44	302.48	299.86	1%	1%	506	506	312	38%	38%
50	398.98	404.46	373.93	6%	8%	823	860	505	39%	41%
70	583.98	580.7	527.95	10%	9%	1011	1218	633	37%	48%
90	592.51	595.7	535.09	10%	10%	1311	1704	822	37%	52%
100	602.49	601.29	539.28	10%	10%	1457	1927	917	37%	52%
Avg	495.88	496.93	455.222	7%	8%	1021.6	1243	637.8	38%	49%

Scenario 2: Varying Data Rates

When sensor devices transmit packets at an increased data rate, the congestion in the network, can lead to packet loss and affect the network performance in terms of power consumption and eventually the lifetime. In our study, we perform simulations by transmitting packets of size 128 bytes at rates varying from 2 p/min up to 10 p/min. The network density is varied from 30 to 100 nodes and single sink node is used. The T-mote sky motes are deployed in a random topology of area 400m * 400m. Transmission range is set to 30m and ratio is 100%. The performance of

the objective function RPL-FZ is compared with OF0 and MRHOF.

Average Power Consumption

It is seen that RPL with MRHOF consumes highest power. This is because MRHOF considers only the transmission count in routing decision and does not consider energy of node. An RPL implementation with the MRHOF objective function can select paths with unreliable links since the link estimation is done using passive link monitoring techniques and RPL has no control over it. Thus, it is likely that the routing decisions with MRHOF may be based on outdated link statistics. It is seen form the experiments that MRHOF performs badly as the network density and data rate increases. The graphs in figure 4.6 depict the power consumption as data rates are increased from 1ppm to 10 ppm and network density from 30 to 100 for RPL-FZ, OF0 and MRHOF implementations.

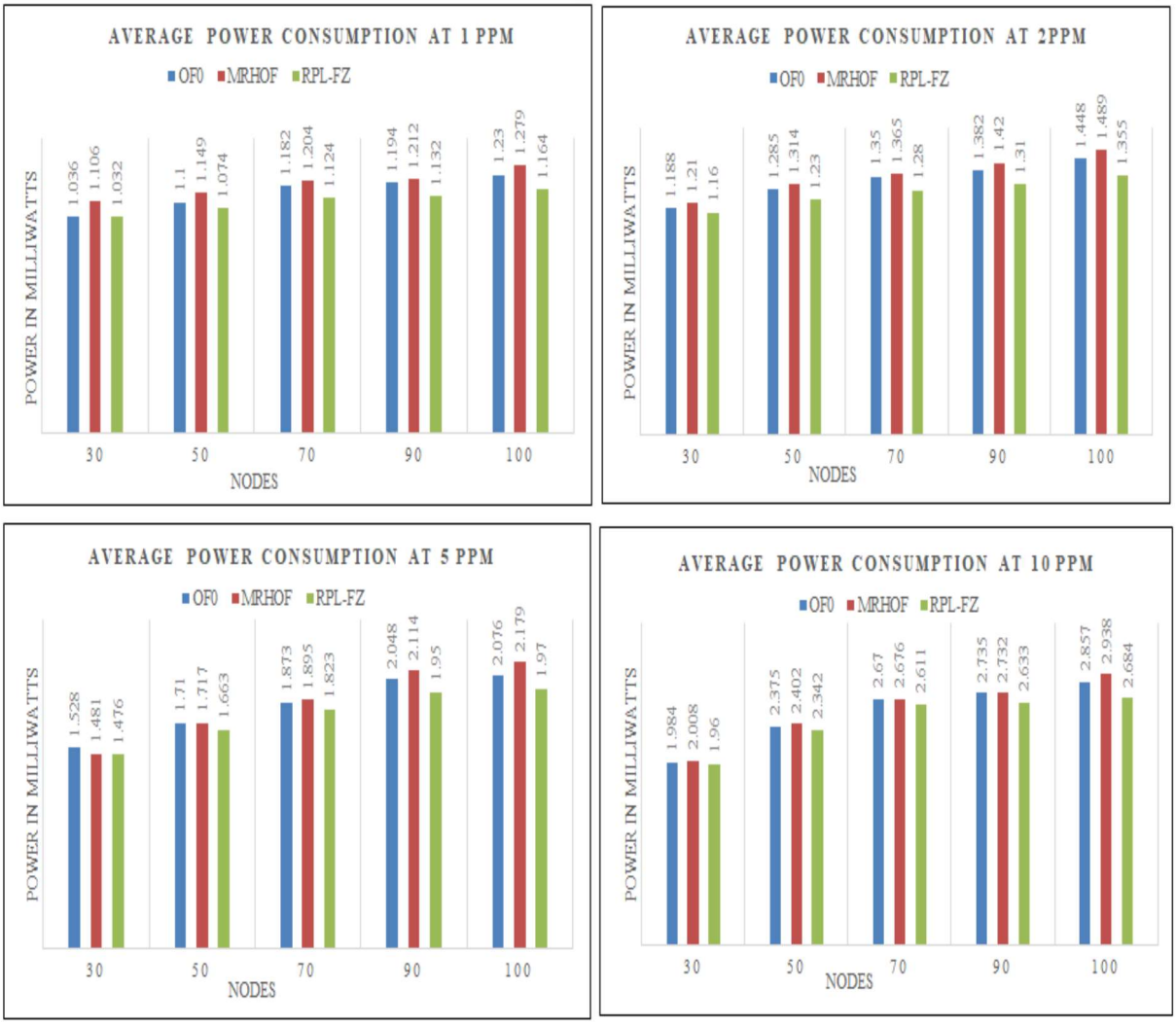


Figure 4.6: Average Power Consumption with RPL-FZ, OF0 and MRHOF for varying data rates

At 50 nodes and data rate of 1 p/min, the power consumption with MRHOF is 7% higher than with RPL-FZ. As the node density increases, more traffic in the network causes the power

consumption to rise. The sensors have to route more packets, causing them to be in ON mode for longer time duration. In a 100 node network setup, RPL-FZ consumes significantly lower power than OF0 and MRHOF to the tune of 6% and 10% at 10 ppm. As the packet rate increases to 10 ppm, power consumption increases further. Higher traffic leads to congestion, collisions and packet loss due to packets being dropped. This causes re-transmission of packets. Under these strained conditions of high density and high data rate, RPL-FZ consumes 9% lower power than MRHOF. Although the sensor nodes require extra computational power for the fuzzy process and calculation of rank based on the Quality score, the final power consumption of the Network with RPL-FZ is much lesser as compared to OF0 and MRHOF. RPL-FZ considers the remaining energy of a node during parent selection, thus ensuring that a node low on energy does not end up repeatedly being selected as parent and depleting its energy further. This shows that a network with RPL-FZ will have better lifetime as compared to OF0 or MRHOF. The power consumed at 50 nodes and 100 nodes and the corresponding improvement obtained with RPL-FZ is listed in table 4.4. RPL-FZ selects optimum parent nodes with low delays, small ETX values and higher energy levels. It ensures good packet delivery success, thus saving the energy spent in transmission and reception on packet loss.

Table 4.4: Improvement in Avg Power Consumption at 50 and 100 nodes with RPL-FZ over OF0 and MRHOF for varying data rates

	Power Consumption (mW) at 50 nodes					Power Consumption (mW) at 100 nodes				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	1.1	1.149	1.074	2%	7%	1.23	1.279	1.164	5%	9%
2 ppm	1.285	1.314	1.23	4%	6%	1.448	1.489	1.355	6%	9%
5 ppm	1.71	1.717	1.663	3%	3%	2.076	2.179	1.97	6%	10%
10 ppm	2.375	2.402	2.3	3%	4%	2.857	2.938	2.68	6%	9%
Avg	1.62	1.65	1.57	3%	5%	1.9	1.97	1.79	6%	9%

Packet Delivery ratio

The sensor nodes continuously transmit data to the Root node. Most of the packets are successfully received, but some are lost due to collisions and congestion. The ratio of total packets successfully received at the Sink to the total packets transmitted in the network is termed as Packet Delivery Ratio. The Packet Delivery ratio with RPL-FZ, OF0 and MRHOF is shown by graphs in figure 4.7 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm.

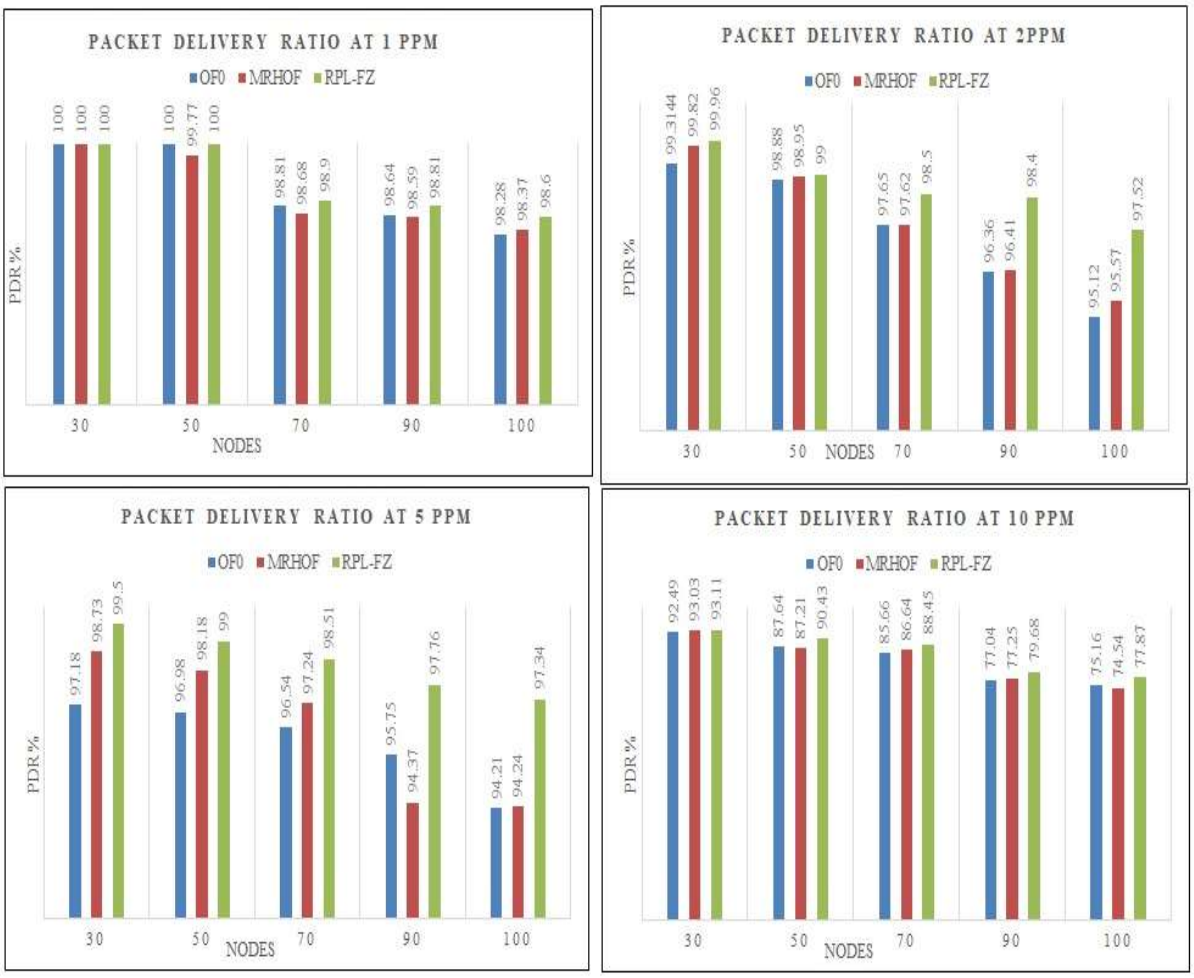


Figure 4.7: Packet Delivery Ratios with RPL-FZ, OF0 and MRHOF for varying data rates

It can be seen from the graphs that PDR is better at lower packet rates and low node density. At low packet rate of 1 ppm, 100% packet delivery is obtained with RPL-FZ. OF0 does not consider the link quality, and when selecting a parent with minimum rank it may actually end up selecting a congested parent and thus end up dropping packets. RPL-FZ considers ETX as a decision metric and thus ensures better packet delivery. As the sensor data transmission rate increases from 2 ppm to 10 ppm and node density from 30 to 100 it is seen that more packets are lost. This happens due to packet collisions and congestion in the network when packet rate increases. RPL is generally suitable for networks with low data rates. The PDR is also depended on the topology of the network. Each node has a buffer size of 4 packets. When the number of packets circulating in the network increases, the buffers overflow and many packets are dropped. This leads to the drop in PDR. In a network with 50 nodes, RPL-FZ gives packet delivery success of 99% upto 5 ppm. At 100 nodes and 10 ppm, RPL-FZ delivers packets with 4% higher success than OF0 and MRHOF. The table 4.5 shows the improvements with RPL-FZ at 50 and 100 nodes with data rate varying from 1 p/min to 10 p/min.

Table 4.5: Improvement in PDR at 50 and 100 nodes with RPL-FZ

	PDR at 50 nodes					PDR at 100 nodes				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	100	99.77	100	Marginal		98.28	98.37	98.6	Marginal	
2 ppm	98.88	98.95	99	Marginal		95.12	95.57	97.52	2%	2%
5 ppm	96.98	98.18	99	2%	Marginal	94.21	94.24	97.34	3%	3%
10 ppm	87.64	87.21	90.43	3%	4%	75.16	74.54	77.87	4%	4%
Avg	95.8	96.02	97.1	1%	1%	90.69	90.68	92.83	2%	2%

Network Latency

The Latency with OF0 , MRHOF and RPL-FZ is shown by graphs in figure 4.8 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. The latency is seen to increase from 300 ms to 2s with OF0 and MRHOF as data rates are increased from 1 ppm to 10 ppm. RPL-FZ gives 7% lower delays than other two on average. As nodes increase to 100 latency is 1.9 s in RPL-FZ, which is 9% lower than MRHOF. This is primarily because RPL-FZ uses the delay metric while routing and thus selects routes with shorter delay. The table 4.6 shows the improvements with RPL-FZ at 50 and 100 nodes over OF0 and MRHOF as data rate vary from 1 p/min to 10 p/min.

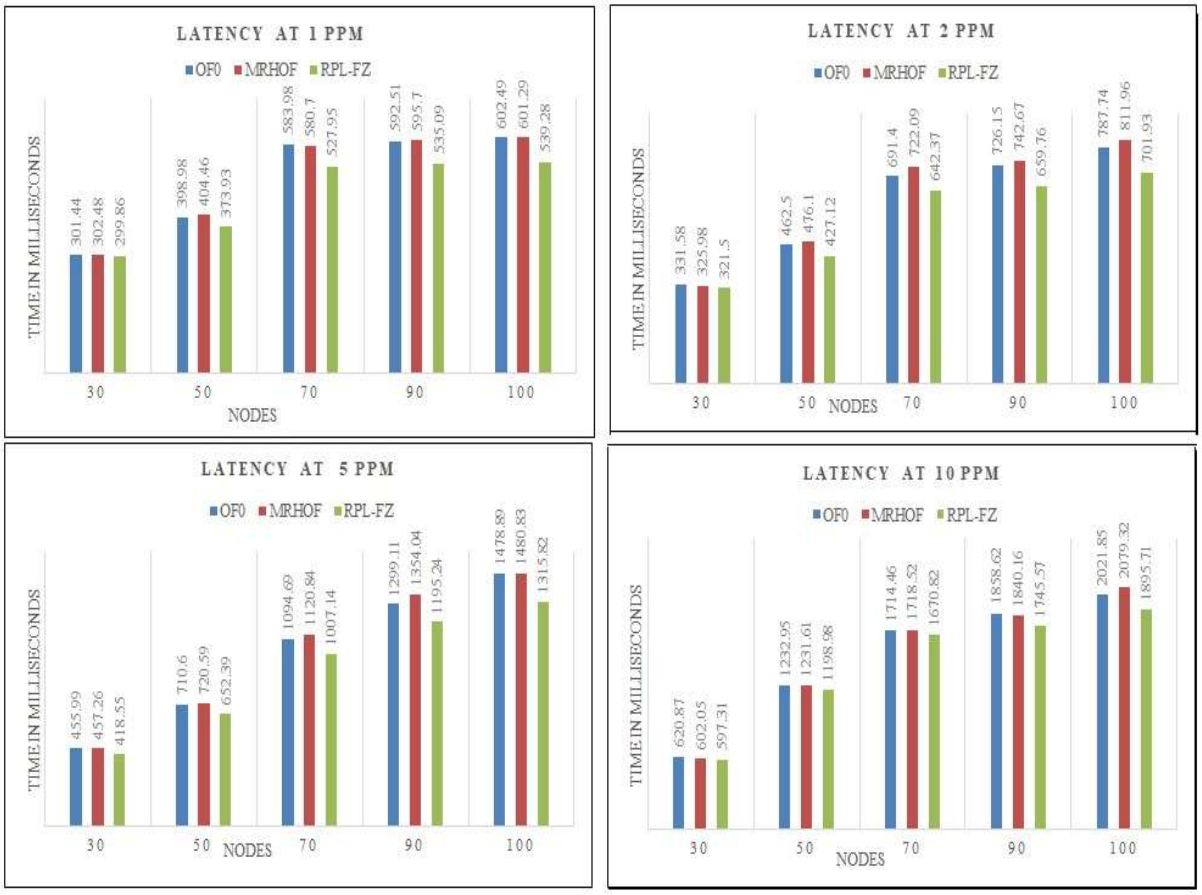


Figure 4.8: Network Latency with RPL-FZ, OF0 and MRHOF for varying data rates

Table 4.6: Improvement in Network Latency with RPL-FZ for varying data rates

	Latency (ms) at 50 nodes					Latency (ms) at 100 nodes				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	398.98	404.46	373.9	6%	8%	602.5	601.29	539.3	10%	10%
2 ppm	462.5	476.1	427.1	8%	10%	787.7	811.9	701.9	11%	14%
5 ppm	710.6	720.59	652.4	8%	9%	1478.9	1480.8	1316	11%	11%
10 ppm	1233	1231.6	1198.9	3%	3%	2021.9	2079.3	1896	6%	9%
Avg	701.26	708.19	663.1	6%	7%	1222.7	1243.4	1113	10%	11%

Control Overhead

In RPL-FZ , significant amount of reduction in the control overhead is achieved due to the mechanism adopted in transmitting DAO and DIO messages. The table 4.7 shows the improvements with RPL-FZ at 50 and 100 nodes over OF0 and MRHOF as data rate vary from 1 p/min to 10 p/min. The control overhead in RPL-FZ is significantly reduced to the tune of 33% and 49% over OF0 and MRHOF respectively. This reduction in control overhead contributes in lowering the congestion. Also the power consumption involved in transmission and reception is saved.

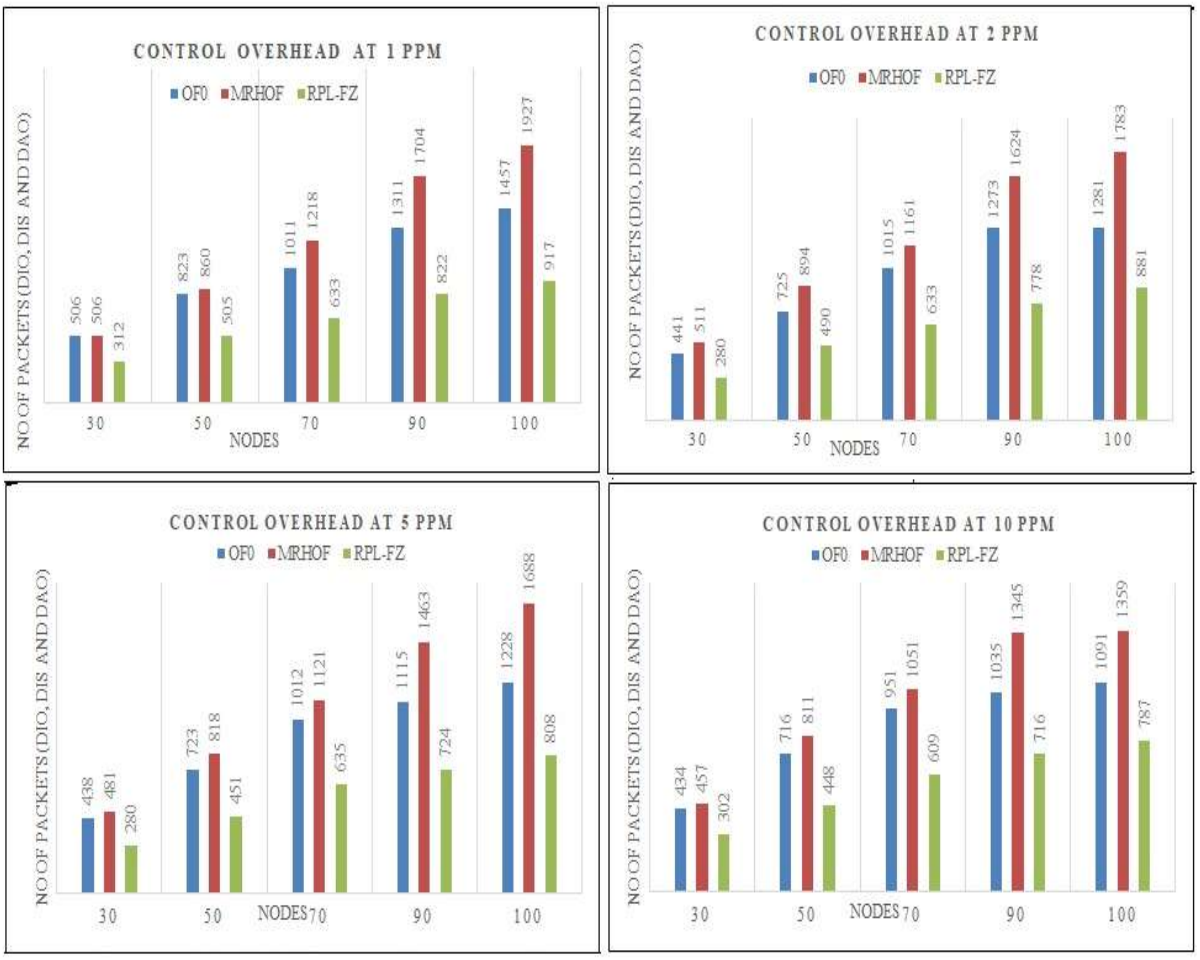


Figure 4.9: Control Traffic Overhead with RPL-FZ, OF0 and MRHOF for varying data rates

Table 4.7: Improvement in Control Overhead at 50 and 100 nodes with RPL-FZ over OF0 and MRHOF for varying data rates

	Control Overhead at 50 nodes					Control Overhead at 100 nodes				
	OF0	MRHOF	RPL-FZ	Improvement over		OF0	MRHOF	RPL-FZ	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	823	860	505	39%	41%	1457	1927	917	37%	52%
2 ppm	725	894	490	32%	45%	1281	1783	881	31%	51%
5 ppm	723	818	451	38%	45%	1228	1688	808	34%	52%
10 ppm	716	811	448	37%	45%	1091	1359	787	28%	42%
Avg	746	845	473	37%	44%	1264	1689	848	33%	49%

Simulations have been performed under varying data transmission rates. The results show that RPL-FZ based RPL-implementation performs better in terms of Packet Delivery Ratio, Power Consumption, Network lifetime and Network Latency as compared to OF0 and MRHOF based networks.

4.6.3 Comparison of RPL-FZ with other existing Fuzzy logic-based implementations in literature

Over the past few years, researchers have made an effort to improve RPL performance by utilizing two and three metrics to make the routing decision. In this section, we compare the performance of our 3-metric based RPL-FZ objective function with these existing works. Some popular and cited works chosen to compare are OF-EC [77], OF-FUZZY [110], OPP-FL [78] and OFRRT-FUZZY [79]. The graph of Packet delivery ratio, Average power consumption, latency and control overhead with the above techniques is plotted in figure 4.10. Results indicate that methods which utilize ETX and Residual energy in routing choice give better PDR. RPL-FZ achieves on average 3% higher PDR than these two or three metric based techniques. The figure 4.10 shows the average power consumption plotted for these techniques. The OPP-FL does not utilise residual energy and suffers from higher power consumption. It also relies on Children number which is the number of children nodes affiliated to a particular parent. The computation of this metric consumes relatively higher amount of power. The delay metric is not considered in OPP-FL. All these factors amount to 10% higher power consumption than RPL-FZ. OFRRT-FUZZY uses the RSSI metric instead of ETX and also experiences higher power consumption. The throughput metric in OFRRT-FUZZY needs more computation steps, thus consumes more power. RPL-FZ excels this work in regard to power consumption by 9% .The sizeable reduction of DAO messages in RPL-FZ consequently reduces the latency too. It reduces the latency by an average of 10% as compared to three metric-based works. The table 4.8 shows the percentage

improvements with OF-FZ over other fuzzy based works.

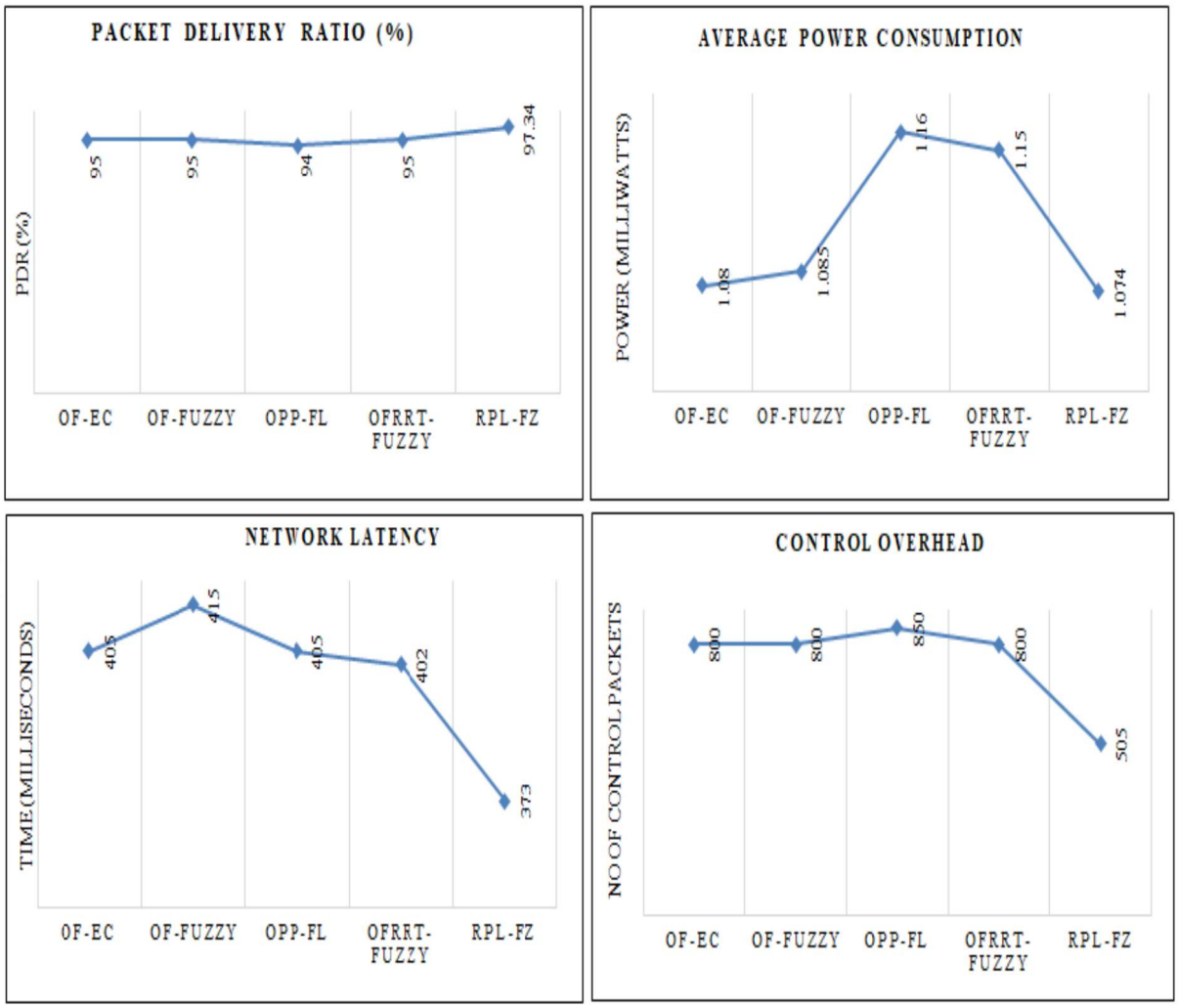


Figure 4.10: Comparison of OF-EC [77], OF-FUZZY [110], OPP-FL [78], OFRRT-FUZZY [79] and RPL-FZ

Table 4.8: Comparison Table of RPL-FZ vs other 3-metric Fuzzy Based works existing in literature

WORK	METRICS	PAVG (mW)	Imp in Pavg with RPL -FZ	PDR	Imp in PDR with RPL -FZ	Latency (ms)	Imp in Lat with RPL -FZ	Over-head	Imp in ovhd with RPL -FZ
OF-EC [77]	ETX, HC, EC	1.08	Marginal	95	2%	405	8%	800	37%
OF-FUZZY [110]	ETX, Delay, Energy	1.085	1%	95	2%	415	10%	800	37%
OPP-FL [78]	ETX, HC, CN	1.16	8%	94	3%	405	8%	850	41%
OFRRT-FUZZY [79]	RSSI, RE, Throughput	1.15	7%	95	2%	402	7%	800	37%
RPL-FZ (proposed work)	RE, Delay, ETX	1.074		97.34		373		505	

4.7 Design of Fuzzy Logic based Objective Function OF-FZ

The performance evaluation of 3-metric based objective function RPL-FZ has made it clear that a holistic routing decision involving multiple metrics gives better results. To improvise further, a fourth metric- Hop Count is introduced in addition to the three routing metrics taken in design of RPL-FZ. This novel objective function based on four routing metrics and fuzzy logic is termed as OF-FZ.

Hop Count metric has the advantage of choosing route with lesser number of hops thereby reducing delay. The four metrics ETX, Delay, Hop Count and Residual energy are chosen as the deciding criteria for best route in OF-FZ. The design of the Fuzzy Inference System for novel objective function OF-FZ is detailed in this section.

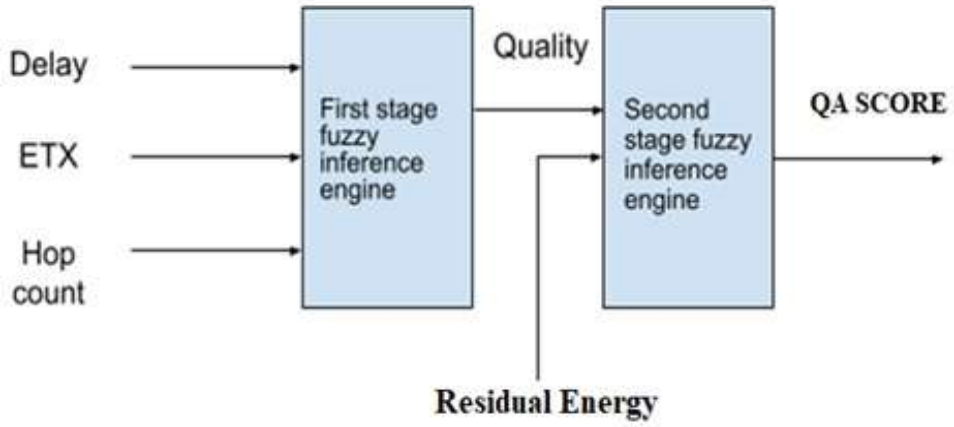


Figure 4.11: Fuzzy Inference System in OF-FZ

During the FIS design, the process of combining the four input variables to obtain a single output quantity is executed in two stages. The output of the FIS in case of OF-FZ is named as Quality Assurance (QA) score to distinguish it from the output of FIS in RPL-FZ. The two stage fuzzification process is shown in Figure 4.11.

In the stage 1 the input variables ETX, Delay and Hop count are fuzzified to obtain an output quantity termed as Quality after defuzzification. In the Stage 2, the Residual Energy (RE) and Quality are fuzzified to obtain the final output Quality Assurance Score. The membership functions for the four metrics are listed in Section 4.4. If the process was executed in single stage, as there are four metrics and each has three fuzzy sets associated with it, the total number of rules would be $3^4 = 81$. This would occupy memory space in a already constrained RAM. Hence, the process is split into two stages. In the Stage 1, only 27 rules are required and Stage 2 needs 15 rules. This approach saves on the memory space utilized.

In Stage 1, The Mamdani model [108] is used to compute the Quality value by applying

fuzzy rules to the fuzzy inputs. The Quality output is represented using five fuzzy sets, namely very_slow, slow, average, fast and very_fast. Figure 4.12 shows the membership function of quality.

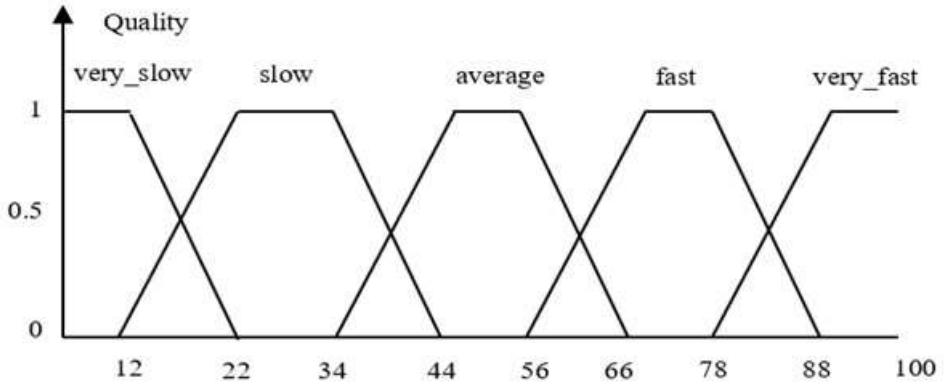


Figure 4.12: Membership function of Quality in Stage 1 of OF-FZ FIS

Some rules used to calculate the Quality are listed below:

1. If Delay is small and HC is near and ETX is short then Quality is very_fast.

2. (If Delay is average and HC is near and ETX is short then Quality score is fast) OR (If Delay is small and HC is average and ETX is short then Quality is fast) OR (If Delay is average and HC is average and ETX is short then Quality is fast) OR (If Delay is small and HC is far and ETX is short then Quality score is fast) OR (If Delay is average and HC is far and ETX is short then Quality score is fast) OR (If Delay is small and HC is near and ETX is average then Quality score is fast)

3. (If Delay is high and HC is near and ETX is short then Quality is average) OR (If Delay is average and HC is average and ETX is average then Quality is average) OR (If Delay is small and HC is near and ETX is long then Quality is average) OR (If Delay is average and HC is far and ETX is average then Quality is average)

4. (If Delay is average and HC is far and ETX is long then Quality is slow) OR (If Delay is high and HC is far and ETX is average then Quality is slow) OR (If Delay is high and HC is average and ETX is average then Quality is slow)

5. If Delay is high and HC is far and ETX is long then Quality is very_slow.

The 27 Rule combinations used to compute Quality are shown in Table 4.9.

Table 4.9: Rules to Compute Quality in Stage 1 FIS for OF-FZ

Inputs			Output
ETX	DELAY	HOP COUNT	Quality
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	average	far	fast
short	high	far	average
average	small	near	fast
average	average	near	average
average	high	near	slow
average	small	average	fast
average	average	average	average
average	high	average	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

In the Stage 2, the inputs residual energy and Quality are combined to obtain a final output quantity Quality Assurance Score. The fuzzy sets representing the Quality Assurance score vary from Very Bad to Excellent and are shown in figure 4.13.

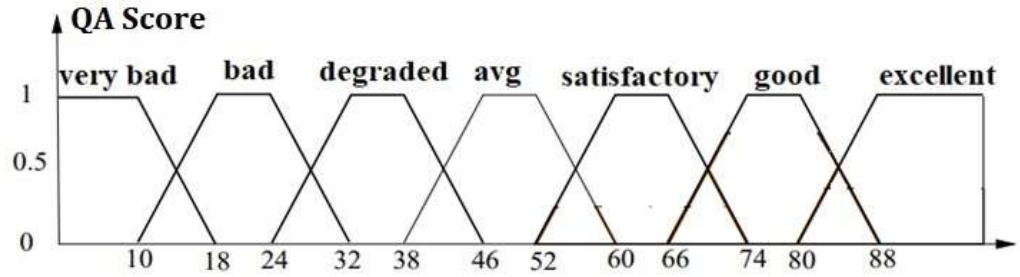


Figure 4.13: Membership function of QA Score in Stage 2 of OF-FZ FIS

The conditions used to define the 15 rules for computation of QA score are shown in table 4.10

Table 4.10: Rules to Compute Quality Assurance Score in Stage 2 FIS for OF-FZ

Quality — ENERGY	low	medium	full
very_slow	very bad	Bad	average
slow	bad	degraded	average
average	degraded	average	satisfactory
fast	average	satisfactory	good
very_fast	average	Good	excellent

The QA score ranges from 0 to 100. A Quality Assurance score is computed for each candidate parent node. For any potential candidate parent, when the combination of metrics results in a high QA score, it can be chosen as preferred parent. A crisp value of QA score is calculated during defuzzification using the centroid formula:

$$X = \frac{\sum_{i=1}^{K} Wi \times \mu\left(Wi\right)}{\sum_{i=1}^{K} \mu\left(Wi\right)} \tag{4.14}$$

where X is the defuzzified crisp value, K resembles the number of activated rules, Wi is the domain value of corresponding rule i, and $\mu(Wi)$ is the predicate truth of that domain value

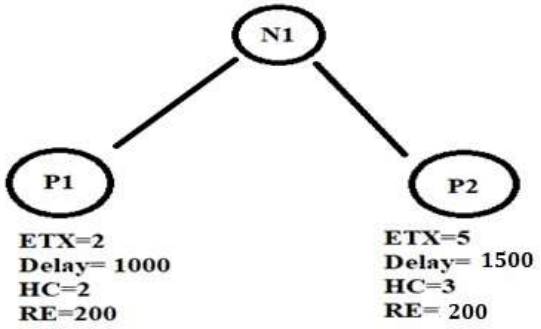


Figure 4.14: Node with choice of candidate parents

To illustrate the entire fuzzy process mathematically, let us consider some examples.

1. Assume a candidate parent 1, with ETX value 2, Delay of 1000, Hop count of 2 and residual energy 200 as shown in figure 4.14. Fist we have to evaluate the term Quality in the stage 1 of fuzzification. A value of 2 for ETX has membership in short fuzzy set with value 1. Delay of 1000 has membership in small and average fuzzy sets with values 0.33 and 0.667 respectively. HC of 2 has membership in near and average fuzzy sets with values 0 and 1 respectively. The membership values are computed using the equations 4.1 to 4.12. Four rules are triggered for quality and only the values "fast" and "very_fast" are activated. Using the max(min) implication, Rule 1 evaluates to a value of 0.333, value of Rule 2 is 0.667, value of rule 3 is zero and value of rule 4 is zero. The quality score is calculated using following centroid formula:

$$Q = \frac{0.333 * 72 + 0.667 * 72 + 0 * 94 + 0 * 72}{0.33 + 0.667 + 0 + 0} = 72 \tag{4.15}$$

   Where 72 and 94 are the centers of gravity for the quality "fast" and "very_fast" respectively. In the stage 2 of fuzzy process, a residual energy of 200 has membership in fuzzy set medium and full with values 0.078 and 0.9216 respectively. The quality of 72 has membership in set fast with value 1. Two rules are triggered for QA score with values "satisfactory" and "good". Using the max(min) implication, Rule 1 evaluates to a value 0.078 and value of rule 2 is 0.9216. The value of QA score is computed using the above centroid formula as

$$QA\ score\ = \frac{0.078 * 63 + 0.9216\ * 77}{0.078 + 0.9216} = 75.9 \tag{4.16}$$

   Where 63 and 77 are the centers of gravity for the QA score of "satisfactory" and "good" respectively.

2. Assume a candidate parent 2, with ETX value 5, Delay of 1500, Hop count of 3 and residual energy 200. Fist we have to evaluate the term Quality in the stage 1 of fuzzification. A value of 5 for ETX has membership in short and average fuzzy sets with value 0.33 and 0.667 respectively. Delay of 1500 is average with value 1. HC of 3 has membership in average fuzzy set with value 1. The membership values are computed using the equations 4.1 to 4.12. Two rules are triggered for quality and values "fast" and "average" are activated. Using the max(min) implication, Rule 1 evaluates to 0.334 and value of Rule 2 is 0.667. The defuzzified value of quality is computed using the centroid formula as

follows:

$$Q = \frac{0.33 * 72 + 0.667 * 50}{0.33 + 0.667} = 57.28 \tag{4.17}$$

Where 72 and 50 are the centers of gravity for the quality "fast" and "average" respectively. In the stage 2 of fuzzy process, a residual energy of 200 has membership in fuzzy set medium and full with values 0.078 and 0.9216 respectively. The quality of 57.28 has membership in sets average and fast with values 0.872 and 0.128 respectively. Four rules are triggered for QA score with values "satisfactory", "good" and "average". Using the max(min) implication, Rule 1 value is 0.872, Rule 2 evaluates to a value of 0.078, value of Rule 3 is 0.128 and Rule 4 evaluates to 0.078. The value of QA score is computed using the above centroid formula as

$$QA\ score\ = \frac{0.872 * 63 + 0.078\ * 49 + 0.128 * 77 + 0.078 * 63}{0.872 + 0.078 + 0.128 + 0.078} = 63.6 \tag{4.18}$$

Where 49, 63 and 77 are the centers of gravity for the QA score of "average", "satisfactory" and "good" respectively.

4.7.1 Simulation Results of OF-FZ

The designed OF-FZ is incorporated into Contiki OS in a manner similar to RPL-FZ. In this case, the DIO message size is increased to accommodate four metrics. In this section, we evaluate the Objective function OF-FZ against the standard OF0 and MRHOF by performing simulations using Contiki OS COOJA platform. The density of nodes in the network is increased from 30 to 100. A single sink node is used to ensure that network is strained. The sink node and sender nodes are static. Data transmission rate is varied from 1 packet/min to 10 p/min. The packet size is 128 bytes. The T-mote sky sensors are deployed in a random topology of area 400m * 400m. Transmission range is set to 30m and ratio is 100%. The performance of OF-FZ is compared to OF0 and MRHOF objective functions for power, PDR, latency and control overhead.

Average power consumption

The graphs of average power consumption are shown in figure 4.15. OF-FZ utilizes four metrics during the routing decision. This enables the nodes to take a holistic decision. The ETX metric ensures that less congested routes are chosen thus avoiding more number of re-transmissions. This helps to save on energy and improves the PDR. The delay metric helps to choose faster routes and lowers the overall network latency. Hop count metric further guides in selecting

shorter routes. The residual energy metric ensures that nodes with depleting energy levels are avoided. This helps to sustain their energy levels and prevents them from energy drain. This prolongs the network lifetime significantly. An overall performance improvement is observed with OF-FZ. As data rates is increased to 10 ppm, the single metric based OF's result in a power consumption 3 mW. Whereas, the consumption with OF-FZ is 2.6 mW at 10 ppm in a 100 nodes network. OF-FZ gives better performance by consuming 7% and 9% lesser energy as compared to OF0 and MRHOF respectively. Including the fourth metric is seen to give around 5% improvement over RPL-FZ at 2 and 5 ppm. The Hop count metric ensures that shortest routes get selected thus saving more on energy. The table 4.11 illustrates the improvement with OF-FZ at maximum node density of 100.

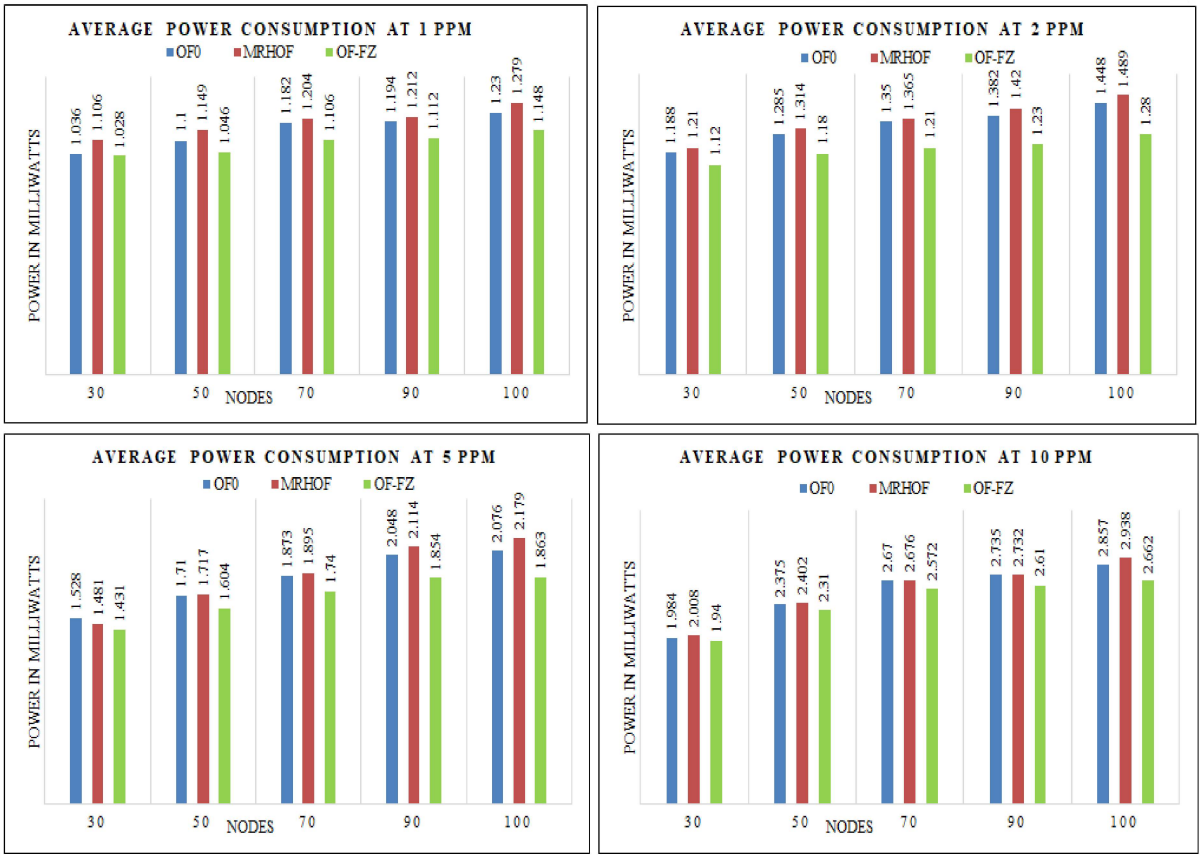


Figure 4.15: Average Power Consumption with varying data rates for OF-FZ

Table 4.11: Improvement in Avg power Consumption with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates

	Average Power Consumption (mW) at 100 nodes				
	OF0	MRHOF	OF-FZ	Improvement over	
				OF0	MRHOF
1 ppm	1.23	1.279	1.148	7%	10%
2 ppm	1.448	1.489	1.28	12%	14%
5 ppm	2.076	2.179	1.863	10%	15%
10 ppm	2.857	2.938	2.662	7%	9%

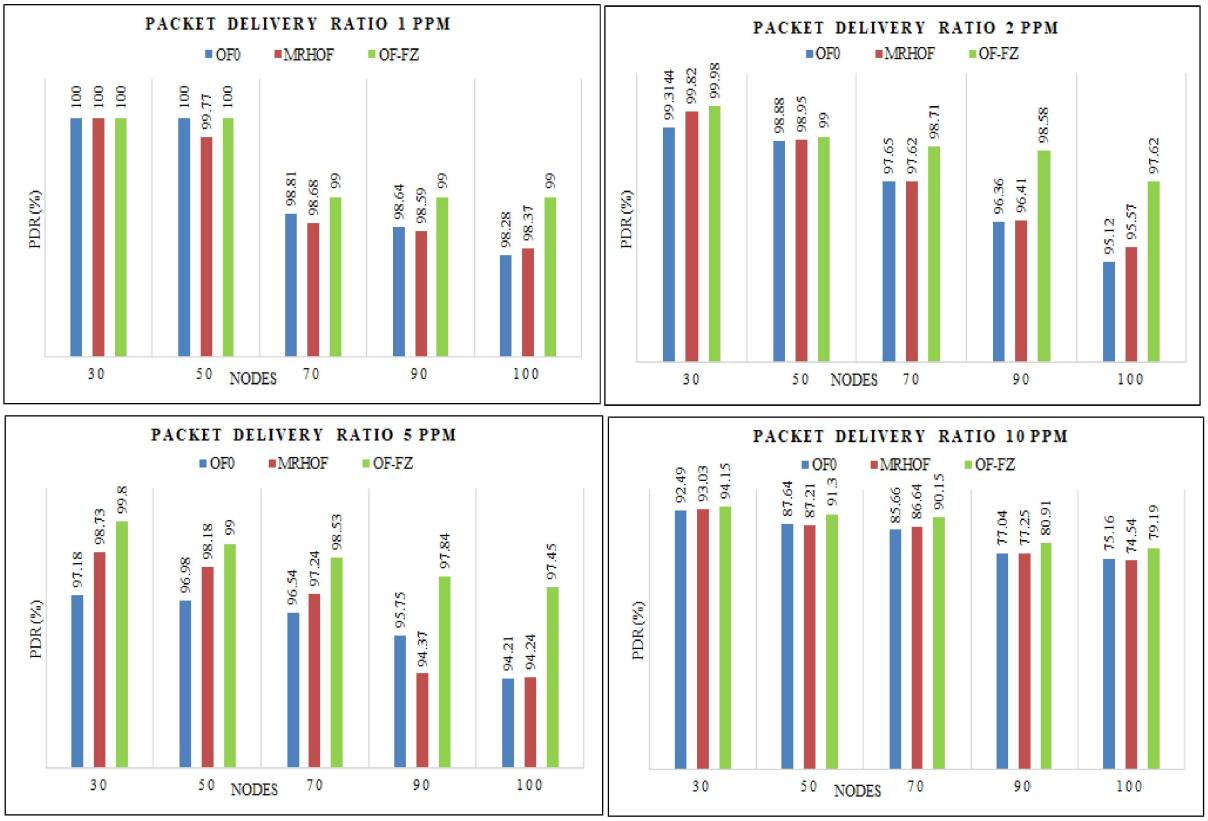


Figure 4.16: PDR with varying data rates for OF-FZ

Packet Delivery Ratio

The graphs in figure 4.16 depict the PDR achieved with OF-FZ, OF0 and MRHOF as data rates are varied from 1 ppm to 10 ppm. The PDR with OF-FZ is 6% higher than that achieved with OF0 and MRHOF. Upto 50 nodes and 1 ppm, packet delivery with OF-FZ is 100% successful. Few packets are dropped as number of nodes increase to 100. PDR is seen to vary from 99% to 97% for date rates till 5 ppm. As the rate is increased to 10 ppm , at 30 nodes PDR is 94% and then drastically drops to 80% at 100 nodes density. This being 6% higher than that with OF0 and MRHOF. The table 4.12 shows the improvement percentage with OF-FZ at 100 nodes for varying data rates. At high density of 100 nodes, the performance is seen to be strained remarkably, hence the comparison of OF0, MRHOF and OF-FZ is tabulated at this density.

Table 4.12: Improvement in PDR with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates

	Packet Delivery Ratio at 100 nodes				
	OF0	MRHOF	OF-FZ	Improvement over	
				OF0	MRHOF
1 ppm	98.28	98.37	99	Marginal	
2 ppm	95.12	95.57	97.62	3%	2%
5 ppm	94.21	94.24	97.45	3%	3%
10 ppm	75.16	74.54	79.19	5%	6%

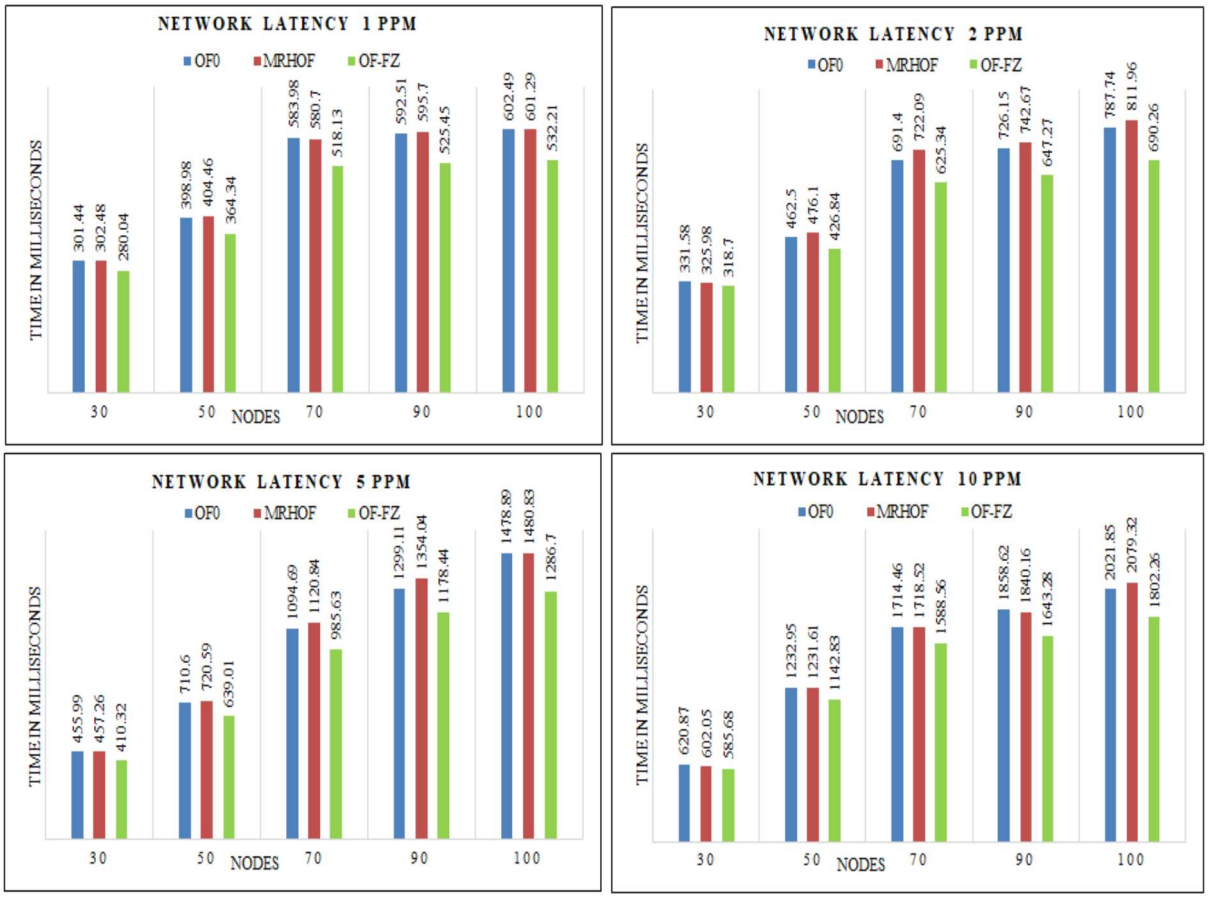


Figure 4.17: Network Latency with varying data rates for OF-FZ

Network Latency

The figure 4.17 shows the latency with OF-FZ, OF0 and MRHOF as data rates are varied from 1 ppm to 10 ppm. The delay and HC metrics used in design of OF-FZ ensures faster routes being chosen towards the root node. Both OF0 and MRHOF lack this advantage. In a 100 nodes network, latency with OF-FZ is 532 ms at 1 ppm and rises to 1.8 s at 10 ppm. Whereas, OF0 and MRHOF have latency of above 2 s at 10 ppm. OF-FZ gives 12% and 13% lower latency over OF0 and MRHOF. The table 4.13 lists the percentage improvements in latency with OF-FZ at 100 nodes for varying data rates.

Table 4.13: Improvement in Network Latency with OF-FZ over OF0 and MRHOF at 100 nodes for varying data rates

	Network Latency (ms) at 100 nodes				
	OF0	MRHOF	OF-FZ	Improvement over	
				OF0	MRHOF
1 ppm	602.49	601.29	532.21	12%	11%
2 ppm	787.74	811.96	690.26	12%	15%
5 ppm	1478.89	1480.83	1286.7	13%	13%
10 ppm	2021.85	2079.32	1802.3	11%	13%

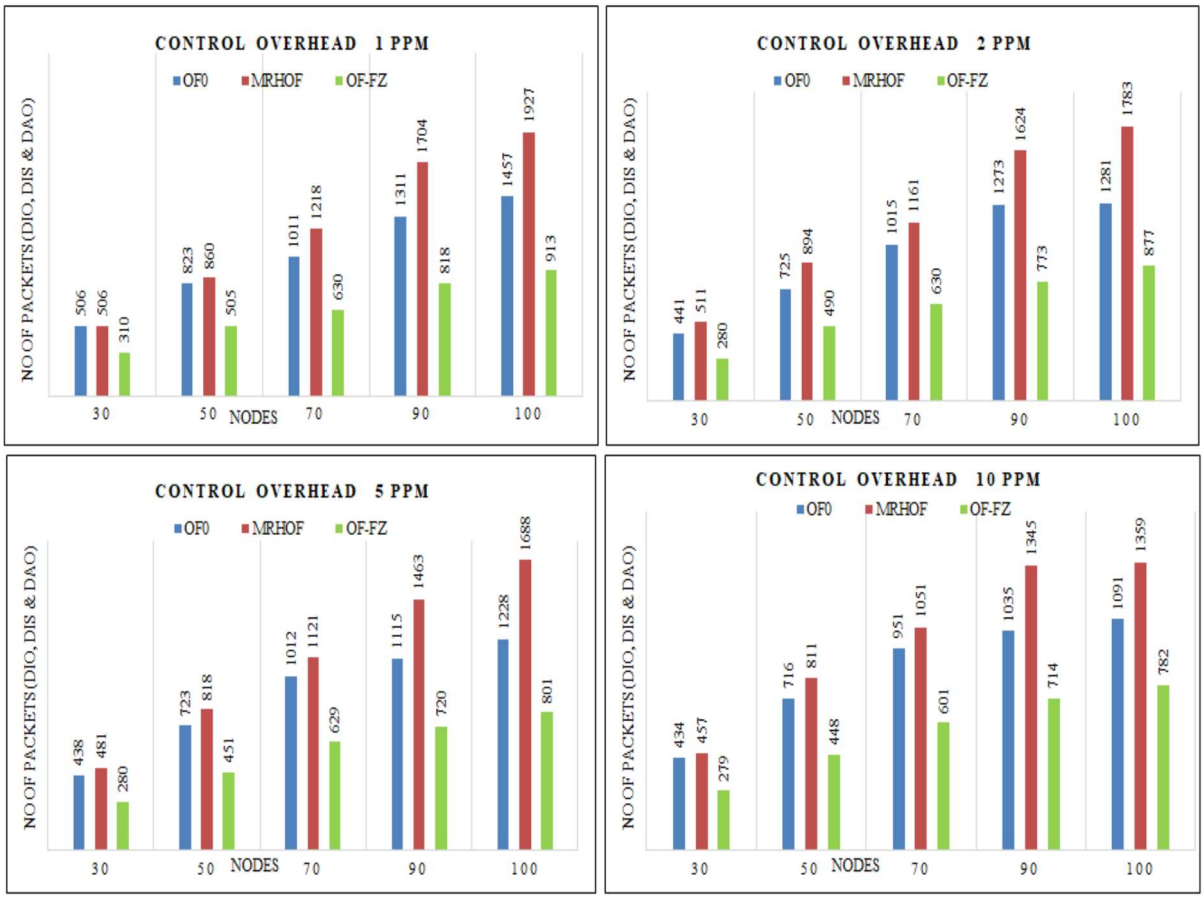


Figure 4.18: Control Overhead with varying data rates for OF-FZ

Table 4.14: Improvement in Control Overhead with OF-FZ over OF0 and MRHOF at 100 nodes with varying data rates

	Control traffic overhead at 100 nodes				
	OF0	MRHOF	OF-FZ	Improvement over	
				OF0	MRHOF
1 ppm	1457	1927	913	37%	53%
2 ppm	1281	1783	877	32%	51%
5 ppm	1228	1688	801	35%	53%
10 ppm	1091	1359	782	28%	42%

Control Overhead: The figure 4.18 shows the control overhead with OF-FZ, OF0 and MRHOF as data rates are varied from 1 ppm to 10 ppm. OF-FZ follows the same timer mechanism for DAO generation as that used in RPL-FZ. The DelayDAO timer expiration is triggered only upon the selection of a new parent node. This ensures that DAO messages are generated only when crucial routing updates must be sent and avoids unwanted overhead. This leads to significant amount of reduction in the control overhead due to the mechanism adopted in transmitting DAO and DIO messages. The control overhead in OF-FZ is significantly reduced to the tune of 36% and 48% over OF0 and MRHOF respectively. The table 4.14 lists the percentage

improvements in overhead with OF-FZ at 100 nodes for varying data rates.

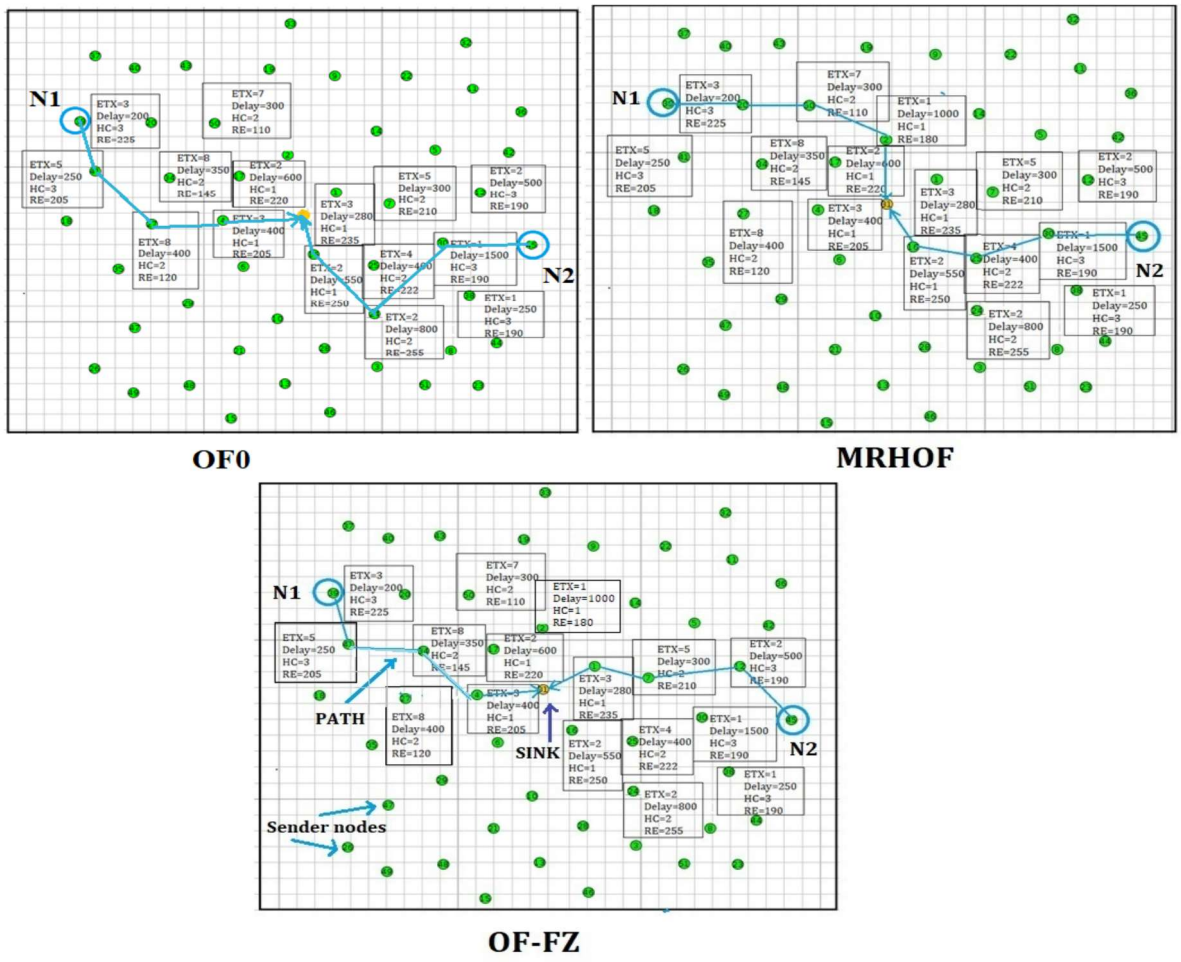


Figure 4.19: Path traced from Nodes N1 and N2 in RPL networks with standard OF0, MRHOF and the proposed OF-FZ

Path traced from nodes N1 and N2 to Sink

The figure 4.19 shows the paths taken by two distinct node N1 and N2 to send packets to the root node in the proposed OF-FZ and existing standard OF0 and MRHOF implementations. All the conditions of random topology, 50 sender nodes, data rate, network area are identical in all three scenarios. If we consider node N1 the path taken in OF-FZ network has a total delay of 1000 ms whereas the OF0 and MRHOF takes a path with delay of 1050 ms and 1500 ms respectively. MRHOF chooses paths based on single metric ETX. It can be seen that N1 with MRHOF chooses parents with residual energy of 225, 110 and 180. N1 with OF-FZ takes parents with RE of 205, 145 and 205. In the long run the parents taken by N1 in MRHOF will drain of energy faster. This plot of paths helps us to understand the advantage OF-FZ holds over the standard versions of RPL.

4.7.2 Performance Analysis of OF-FZ

In this chapter, a Fuzzy logic-based RPL objective function OF-FZ is proposed. To verify the efficiency of OF-FZ, the four performance parameters-Packet Delivery Ratio, power consumption, Network latency and control overhead are measured. In the previous section, the performance of OF-FZ is compared with the standard MRHOF and OF0 versions of RPL and it is seen that OF-FZ excels over them. This section compares OF-FZ with RPL-FZ and other existing fuzzy logic-based works in literature which use multiple metrics for decision making.

4.7.2.1 Comparison of OF-FZ with RPL-FZ

In this chapter, we have proposed two novel objective functions RPL-FZ and OF-FZ which evaluate multiple metrics to decide optimal path in a RPL network. The preliminary difference between these two is in the number of metrics used. RPL-FZ uses three, while OF-FZ uses an additional metric Hop Count for decision making. In this section, we compare the performance of OF-FZ and RPL-FZ. The Average power consumption, PDR, Latency and control overhead are plotted in the figure 4.20. The readings at node densities of 30,50, 70,90 and 100 are averaged and plotted at varying data rates.

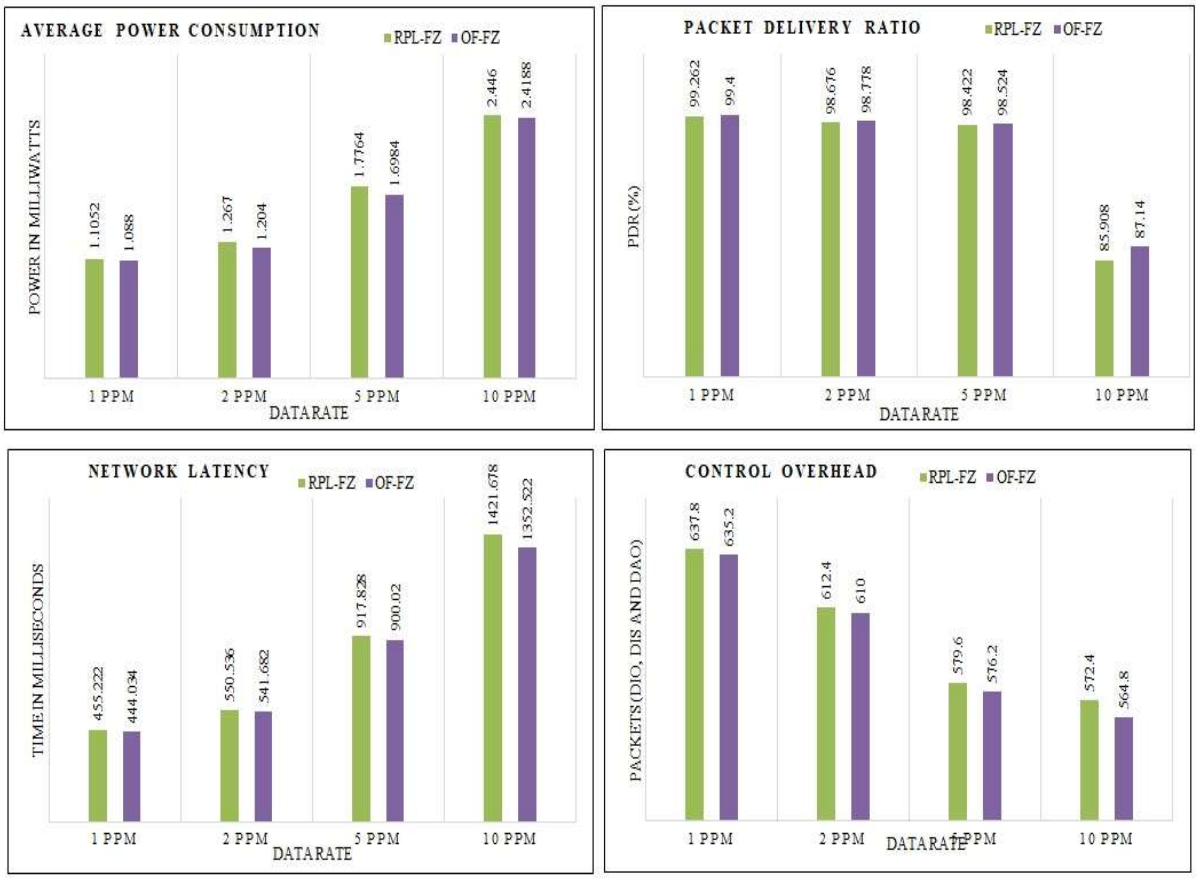


Figure 4.20: Comparison of OF-FZ with RPL-FZ

The inclusion of hop count results in a additional 14 bytes of data added to the DIO message. The rule base in a single stage FIS would increase from 27 (in-case of 3 metrics) to 81 rules, but due to the two stage design adopted by us, 15 rules get added extra in OF-FZ. The rise in computation power due to an additional metric is observed to be minimal. The advantages of adding the HC metric outweigh the downside. Inclusion of HC in decision making ensures that parent with minimum distant to sink node is chosen. This reduces the latency as can be seen from the graphs of figure 4.20. At packet rate of 10 ppm the improvement is 5%. At lower packet rates there is 2% reduction in latency. This in turn saves on the power consumed. The improvement ranges from 1% to 5%. The influence of HC on successful packet delivery is seen to be marginal. The control overhead is comparable as same mechanism is adopted for message generation on instance of parent change in both designs. The table 4.15 summarizes the percentage improvements of OF-FZ over RPL-FZ.

Table 4.15: Improvement of OF-FZ over RPL-FZ

	Improvement of OF-FZ over RPL-FZ			
	P avg	PDR	Latency	Control Overhead
1 ppm	2%	Marginal	2%	Marginal
2 ppm	5%	Marginal	2%	Marginal
5 ppm	4%	Marginal	2%	1%
10 ppm	1%	1%	5%	1%

4.7.2.2 Comparison of OF-FZ with existing four-metric fuzzy logic-based implementations in literature

In this section, the fuzzy logic-based novel objective function OF-FZ is compared with two other four metric-based objective functions OF-FL [76] and FL-HELR [81]. OF-FL uses the same routing metrics as OF-FZ whereas FL-HELR-OF replaces the ETX metric with RSSI. The differences in the design of OF-FZ, OF-FL and FL-HELR are shown in table 4.16. These two works are selected as they use the same simulation parameters as our work.

Table 4.16: Design Considerations: OF-FZ vs OF-FL [76] vs FL-HELR-OF [81]

Parameter	**OF-FZ**	**OF-FL [76]**	**FL-HELR-OF [81]**
Choice of Metric	HC, ETX, Delay, RE	HC, ETX, Delay, BL	HC, RSSI, Delay, EC
Design	Two Stage	Single Stage	Single Stage
Rule Base	Stage 1: 27 Rules, Stage 2: 15 Rules	81 Rules	81 Rules
Membership Function of Quality	Trapezoidal	Triangular	Triangular

Packet Delivery Ratio: The PDRs are plotted for the above three methods by varying the node

density from 30 to 100 and at packets rate of 1ppm and 5ppm as shown in figure 4.21. The table 4.17 shows the improvement in PDR of OF-FZ over the other two implementations. When the network is less dense with 30 nodes, the packet delivery rates are comparable. As the packet rates increase and the number of sensor nodes rises beyond 50, the packet loss increases in the case of FL-HELR. This is mainly due to the fact that FL-HELR does not consider the link quality during routing. The sensors choose poor routes which take more re-transmissions to deliver a packet. A hundred nodes transmitting 5 packets per min leads to a huge upsurge in data flowing within the network. In this situation, FL-HELR is observed to suffer from a significant packet loss to the tune of 16%.



Figure 4.21: PDR of OF-FZ, FL-HELR [81] and OF-FL [76] at 1 ppm and 5 ppm

OF-FZ makes smarter routing choices and provides successful packet delivery of 97.45%. The repeated attempt to transmit a packet in the case of FL-HELR leads to higher consumption of power. OF-FL utilizes the same routing metrics as OF-FZ. However, the distinguishing factor lies in the design of the fuzzy logic controller. The two-stage design in OF-FZ with a reduced rule base provides accurate score calculation for a given candidate parent.

Table 4.17: Improvement in PDR of OF-FZ vs FL-HELR-OF vs OF-FL at 1ppm and 5 ppm

	PDR at 1 ppm					PDR at 5 ppm				
NODE	OF-FZ	FL-HELR	OF-FL	% Improvement over		OF-FZ	FL-HELR	OF-FL	% Improvement over	
				FL-HELR	OF-FL				FL-HELR	OF-FL
30	100	99	100	1	0	99.8	98	98	2	2
50	100	93	99	7	1	99	90	97.5	9	2
70	99	90	98	9	1	98.53	87	96	12	3
90	99	88	96.5	11	3	97.84	85	93	13	5
100	99	87	95.5	12	4	97.45	82	92	16	6

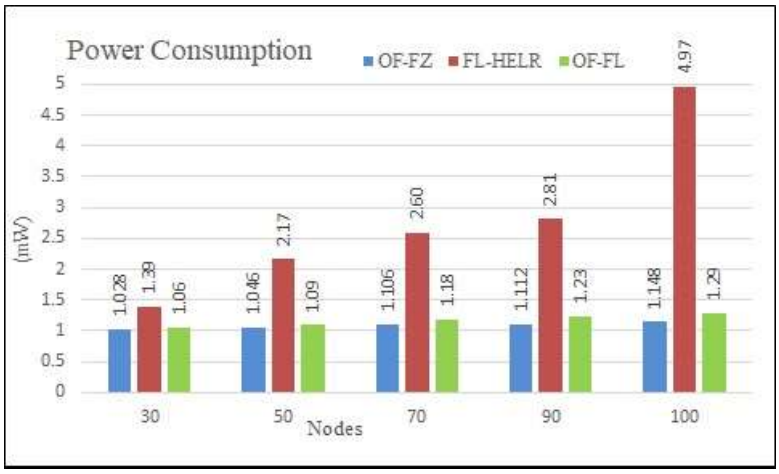


Figure 4.22: Power Consumption with OF-FZ, FL-HELR [81] and OF-FL[76]

Average Power Consumption: Figure 4.22 shows the power consumption with the three objective functions. At a low density of 30 nodes, the energy consumed with FL-HELR is 26% higher than with OF-FZ. As the density of nodes increases to 100, the power consumption with FL-HELR rises drastically to more than 60% than with OF-FZ. This is related to the fact that FL-HELR experiences higher packet loss, resulting in more re-transmissions and thus higher energy spent. OF-FL consumes on average 7% higher energy than OF-FZ. The routes taken from sender nodes to sink are traced for the three implementations as shown in figure **??**. The improvement with OF-FZ versus other works in terms of power is shown in table 4.18. OF-FZ reduces the advertisement control overhead (DAO messages) which considerably reduces number of packets in the network at any given time. This reduces packet collisions and congestion, thus improving the Packet Delivery ratio. The choice of metrics in OF-FZ and OF-FL (Delay, ETX, HC and RE) gives better PDR and Power consumption over FL-HELR-OF (EC, HC, Latency and RSSI) OF-FZ has higher rates of packet delivery. Thus, it saves the energy of packet re-transmissions owing to packet loss. FL-HELR-OF uses the RSSI metric. Its computation involves a series of calculations which further leads to more power being consumed by the nodes.

Table 4.18: Improvement in Power Consumption of OF-FZ vs FL-HELR-OF vs OF-FL

NODE	Avg Power Consumption (mW)				
	OF-FZ	FL-HELR	OF-FL	% Improvement over	
				FL-HELR	OF-FL
30	1.028	1.388	1.06	26%	3%
50	1.046	2.167	1.09	52%	4%
70	1.106	2.599	1.18	57%	6%
90	1.112	2.813	1.23	60%	10%
100	1.148	4.972	1.29	77%	11%
Avg	1.088	2.787	1.17	54%	7%

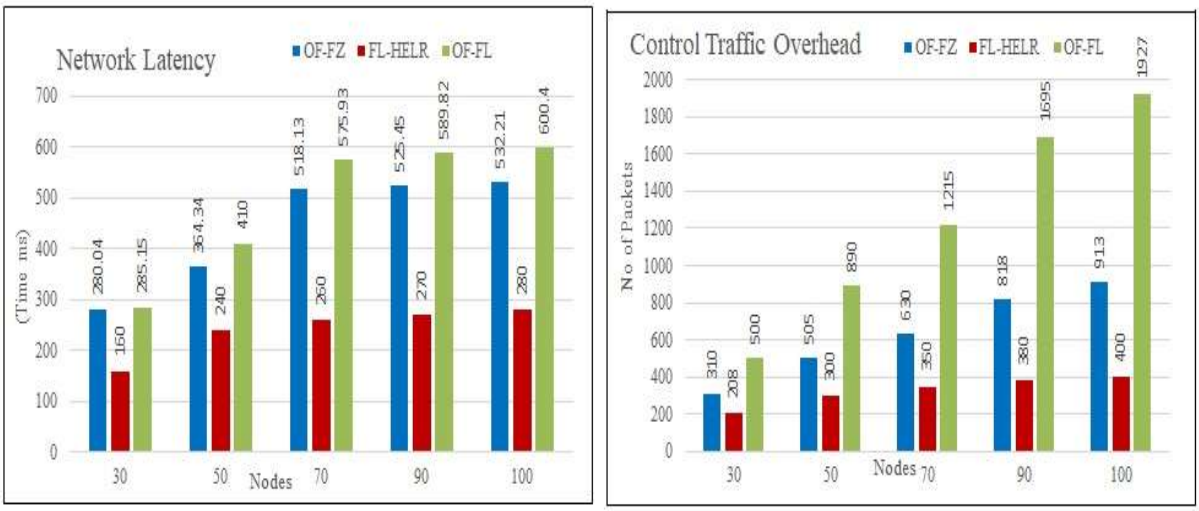


Figure 4.23: Network Latency and Overhead with OF-FZ, FL-HELR [81] and OF-FL [76]

Latency and Control Overhead: Figure 4.23 shows the Network Latency and Control Overhead of the three implementations. FL-HELR generates fewer control messages than OF-FZ. At 100 nodes, it transmits 46% lesser messages than OF-FZ. Generating less number of control messages will reduce traffic and improve the network latency, but it cannot be considered a desirable feature. As control messages are less frequent, the routing information updates will reach the nodes after longer intervals. If a link is down, and an update has not been reached, then the nodes will fruitlessly keep transmitting the packets to an inactive node. This could be another factor responsible for lower packet delivery and power consumption in FL-HELR. The reduction of overhead packets in OF-FZ is sufficient enough to reduce latency but not hamper the network performance by reducing the transmission of routing updates as in the case of FL-HELR [81].

Thus we come to the following conclusions:

1. ETX and Residual energy are seen to play a crucial role in governing the PDR and power consumption.

2. Choosing RSSI as a metric over ETX degrades the packet delivery success ratio and increases power consumption.

3. Computation time of Children Number, RSSI and Throughput metrics is high and consumes more CPU power, hence not recommended.

4. The choice of metric ETX along with Residual Energy and Delay in RPL-FZ gives optimal results compared to other three metric combinations.

5. The addition of Hop Count metric to ETX, RE and Delay in case of OF-FZ gives improvement in terms of Power Consumption(5% at 2 ppm and 5ppm) and Latency (2% at 2ppm and 5ppm and 5% at 10 ppm), but is seen to have marginal effect on the PDR. The computation cost of adding HC metric is negligible.

6. Routing decisions based on four metrics improve the overall performance in an RPL network.

7. A maximum of four metrics can be used. These are contained and transmitted via the DIO control messages. The size of routing metric and constraint is 6 bytes each. Two bytes for type and option length. Four metrics with constraints will need 56 bytes. The IPV6 header size is 5 bytes, ICMP type, code and checksum take 4 bytes. A DIO base object takes 24 bytes, 16 bytes for DODAG Configuration Option, 24 bytes for Route Information option. This makes a total packet size of 129 bytes. A control message of larger than 128 bytes is not ideal in an LLN.

8. The added overhead due to four metrics can be compensated by reducing the advertisement messages.

9. Reducing the control messages helps to reduce the latency, but curtailing them beyond a certain point may adversely affect the network performance.

10. Designing the fuzzy logic controller by splitting it into two stages and reducing the number of rules yields better results. Trapezoidal membership functions for the inputs and output linguistic variables may give better results than triangular membership functions. Reducing the Rule base saves the sensor memory space.

4.8 Conclusion

RPL is considered as the routing backbone in Internet of Things applications. The objective of our work is to optimize RPL protocol with focus on improving energy efficiency, reliability and speed. In this chapter, we proposed an improved objective function OF-FZ for RPL. The objective function (OF) is responsible for route formation. The existing standard OFs consider only a single metric for route selection and do not provide quality of service. OF-FZ uses four metrics (Hop Count, ETX, Delay and Residual Energy) to make the routing decision. The proposed approach uses the fuzzy logic technique to combine the four metrics and compute a single defuzzified metric Quality Assurance score. The QA score for each candidate neighbor

is computed by invoking the fuzzy controller. The neighbor with highest QA score is selected as preferred parent to route data to the sink. Since OF-FZ is a holistic objective function, it ensures that a route which is reliable, energy efficient and fast is selected. On an average, the RPL network with proposed objective function OF-FZ consumes 7% and 9% lesser power as compared to OF0 and MRHOF respectively. When the network is strained at packet rates of 10p/min, single sink node and 100 sender nodes, OF-FZ gives 5% higher Packet Delivery Ratio than OF0 and MRHOF. OF- FZ has 13% lower latency as compared to MRHOF under strained conditions. Another significant contribution with OF-FZ is the reduction in the DAO messages. It can be concluded that OF-FZ based RPL-implementation outperforms the standard objective functions and other existing works in literature with respect to power efficiency, network lifetime, latency, packet delivery ratio and network traffic overhead.

Chapter 5

Design of Efficient Objective Function for RPL using Artificial Neural Networks

5.1 Introduction

Artificial Neural Networks are computational mechanisms inspired by the functioning of human brain. They consist of a network of neurons connected by synaptic weights and configured to perform a specific task. They were invented in the late 1940s, when McCulloch and Pitt created computational models for neural networks. Hebb continued the work and proposed the Hebbian learning technique to train a neural network. Since then the domain of neural networks has evolved to a great extent and is applied to solve problems which cannot be addressed using conventional computing.

Neural Networks can be applied to solve problems in classification, regression, optimization and clustering domains. Some common application areas are stock market prediction, speech recognition, face recognition, aerospace, handwriting analysis, medical diagnosis, text translation and social media platforms. The application of neural networks in communication protocols is explored in this chapter.

In our work, a relation needs to be established between the four metrics-ETX, Delay, Hop count ,Residual Energy and the optimal route to gateway. No direct mathematical formula exists to relate these variables. Hence, the soft computing approaches need to be explored. In this section, we describe the design of a multi layer feed forward artificial neural network, which accepts the four routing metrics as input and computes the quality scores of parent nodes. This novel objective function proposed for RPL is termed as ANN-RPL. The performance analysis of ANN-RPL is carried out in Contiki OS COOJA Simulator by varying number of nodes and

data packet rates. An improvement in power consumption, latency, PDR and control overhead is obtained with this novel ANN based RPL version as compared to standard RPL.

5.2 Research Objective

The main aim of this research is to design a routing strategy that overcomes the single metric-based routing limitation of existing standard RPL protocol. It is modeled as a multi-constrained optimal path problem, where best route decision is based on multiple routing metrics and a Artificial Neural Network is employed to determine the optimal route toward the sink node.
The objectives of this research is:
Design a routing strategy for RPL that applies a Multilayer Feed Forward Artificial Neural Network and achieve an improvement in power consumption, latency, PDR and control overhead of the deployed network.

5.3 Design of ANN based objective function for RPL

This section describes the design steps of the proposed novel Artificial Neural Network based RPL version ANN-RPL. A three-layer perceptron Feed Forward network is designed, which will be applied to determine the optimal parent for a node within the DODAG.

As discussed in the previous chapter 4, the use of four metrics namely, Hop count (HC), ETX, Residual energy (RER) and End-to-end Delay in deciding the best route gave very good results. So, in this chapter too, the same four metrics are chosen as the criteria for selecting the optimum parent. These four metrics are fed as input to the feed forward neural network.

The neural network design consists of three layers of neurons. In layer1 there are 10 neurons, layer2 constitutes of 5 neurons and layer3 is the single output neuron. This is a typical example of network for performing regression. The metrics are applied as inputs to the layer 1. The network computes the quality score for each candidate parent depending on the values of ETX, HC, RER and Delay. The first step is to identify a dataset to train the neural network. The standard UCI Machine Learning Repository does not contain the required dataset for this application. Thus, we had to create our own exhaustive dataset to suit the requirement. The permissible range defined for ETX is from 1 to 15, HC is from 1 to 5, Delay is from 0 to 3000 ms and RE from 0 to 255. A python script was written to generate values of ETX ranging from 1 to 15 in steps of 1, HC from 1 to 5 in steps of 1, Delay from 0 to 3000 in steps of 100 and RE from 0 to 255 in steps of 5. A training data set was created with 85,686 sample values where all

exhaustive combinations of the four metrics were taken and a corresponding Quality value was computed for each combination using the FIS designed in previous chapter. As the FIS gave better performance over standard RPL, the quality values generated by the FIS were taken as base. If the dataset was used as it is, then results would be identical to those with Fuzzy Logic. Since we want to differentiate from FIS, the Quality values for innumerable samples were altered. In case of best-case scenario i.e ETX=1, HC=1, Delay=0 and RE=255, highest Quality value of 100 was assigned. As values of metrics degrade, lower Quality values were assigned. The assigned Quality value ranges from 15 to 100. Few samples of the data-set are shown in Table 5.1.

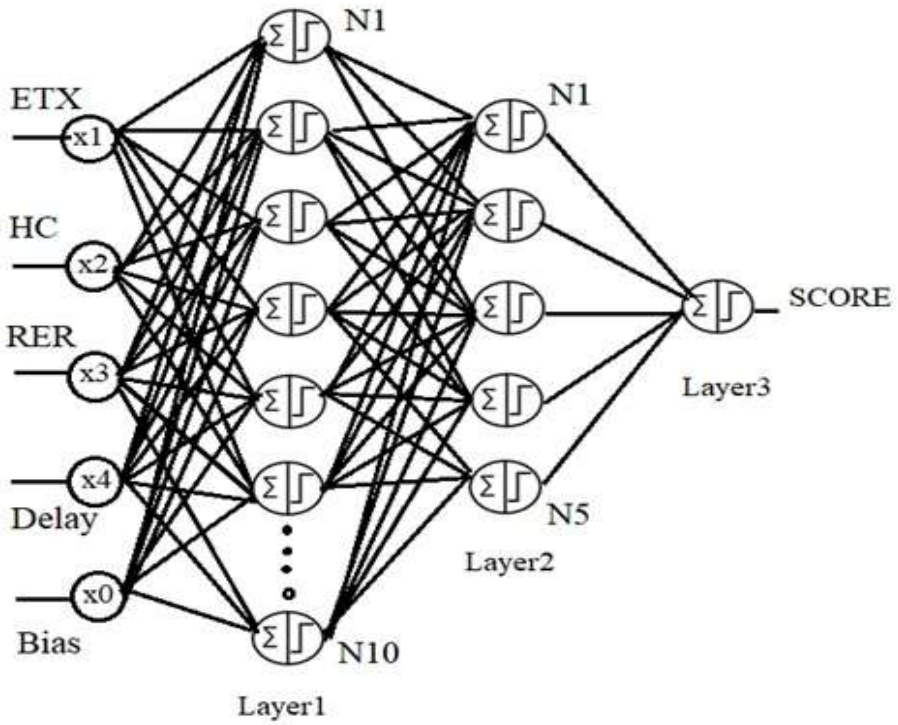


Figure 5.1: Artificial Neural network for selecting parent in RPL network.

Table 5.1: Training Dataset samples

Inputs				Output
ETX	Delay	Hop Count	Residual energy	Quality Score
1	1	1	255	100
2	71	4	251	94
3	1101	2	76	55
4	371	1	239	89
11	1501	1	111	48
14	301	4	61	45
14	2901	4	1	15
12	2801	4	16	20

This network is trained by splitting the dataset into 70% and 30% for training and validation.

The gradient descent-based Delta learning algorithm was used for training the network. The trained network gives an accuracy of 96%. The network is shown in figure 5.1. The weights connecting the neurons at each layer are listed in Appendix A.

The design and training of the neural network was done using tools like Google Collab and Jupyter notebook. Several other neural network configurations with different number of layers and neurons were tried and tested. But the one shown above gave better accuracy and practicality of implementation. Table 5.2 shows a list of metric values and the corresponding calculated score by the ANN. A node within a network has several neighbors. These nodes transmit the DIO messages to its neighbors at regular intervals. The values of ETX, HC, Delay and RE for a particular neighbor can be obtained from the received DIO message. The ANN is invoked and the extracted metric values are fed to the Feed Forward Neural network to compute the quality score of that neighbor. In a similar manner, for a given set of neighbors the scores of each can be computed. We can assume that a particular node X has 6 neighbors whose corresponding metric values are listed in table 5.2. The scores computed by the ANN for each neighbor are also listed in table. The node N2 has the highest score of 93, thus will be chosen as the parent by node X to route packets towards the root.

Table 5.2: Four Metric values and computed Score by ANN

	Inputs				Output
Neighbor	ETX	Delay	Hop Count	Residual energy	Q Score-ANN
N1	3	200	3	225	87
N2	2	550	1	250	93
N3	8	400	2	120	63
N4	3	400	1	205	86
N5	2	600	1	220	88
N6	1	1500	3	190	77

The steps involved in best parent selection using the ANN based RPL implementation are depicted in the algorithm of figure 5.2. Upon receipt of a DIO message, the receiving node adds the DIO sender neighbor as candidate parent to its candidate parent list. The values of the four metrics are extracted from the DIO metric container. The ANN is invoked and the score of the sender node is computed. If its score is higher than the score of the present parent node, then parent is switched. The new rank is computed and the change in routing information is communicated to neighbors by sending a DAO message.

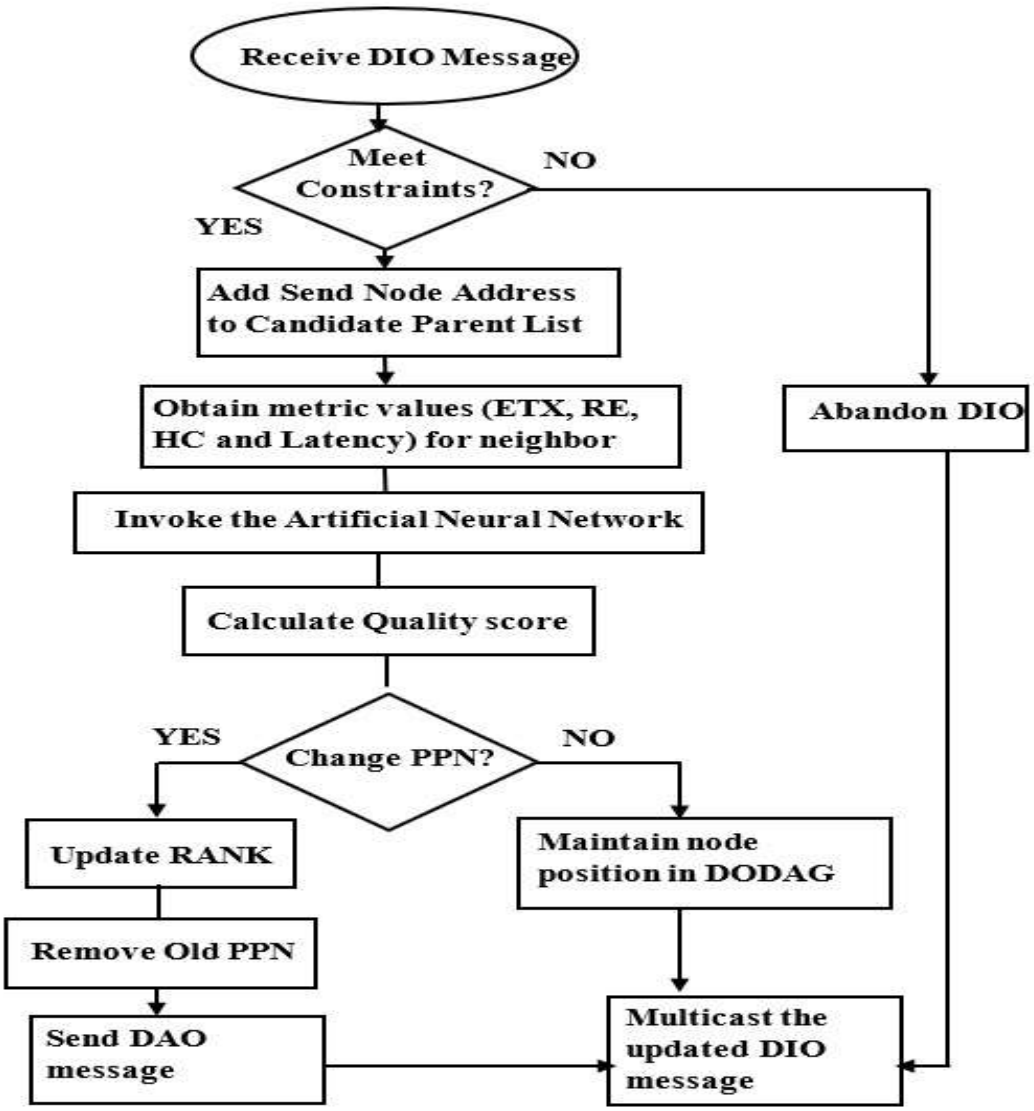


Figure 5.2: Parent selection process using ANN-RPL

5.4 Simulation Results of ANN-RPL

The novel ANN based objective function needs to be incorporated in the operating systems meant for IoT motes to ensure that it can successfully be deployed in IoT networks. Contiki OS is build for the wireless low power sensor motes used in Internet of Things applications. It has an in-built TCP-IP stack with multitasking flexibility. Thus, the neural network functionality is incorporated into Contiki OS for practical application of this version of RPL. To test its effectiveness, simulations are performed to compare it with the standard RPL objective functions OF0 and MRHOF. The detailed description of the standard objective functions OF0 and MRHOF is given in chapter 3. A network with area of deployment 400m x 400m in random topology with many sender nodes and a single sink node in created in COOJA environment for simulations to be carried out. The test logs generated during the simulation in COOJA are

saved. The values of power consumption and packet delivery ratio are obtained from collect view environment of COOJA. The delay is obtained from the generated test logs by writing PERL scripts. The control overhead is the sum of DIO, DIS and DAO messages generated in the network. PERL Scripts are written to analyse the test logs and count the total control packets generated. The proposed ANN-RPL is compared to standard RPL versions MRHOF and OF0 during simulations where number of nodes are varied from 30 to 100. Packets of length 128 bytes are transmitted at rate of 1p/min, 2 p/min, 5 p/min and 10 p/min. The simulations are performed in two case scenarios.

Scenario 1: In the first scenario, the packets are transmitted after an interval of 60 seconds and the network node density is varied from 30 to 100 nodes. The plot of average power consumed and the PDR is shown in figure 5.3. With the node density, power consumption also proportionally increases. ANN-RPL consumes lesser power than OF0 and MRHOF consistently at all node densities. At 30 nodes, the power consumption by ANN-RPL is 1 mW and gradually increases to 1.1 mW at 100 nodes. Whereas, OF0 and MRHOF result in 1.23 and 1.28 mW at 100 nodes. On an average, the improvement in power consumption with ANN-RPL is 8% and 10% over OF0 and MRHOF respectively.

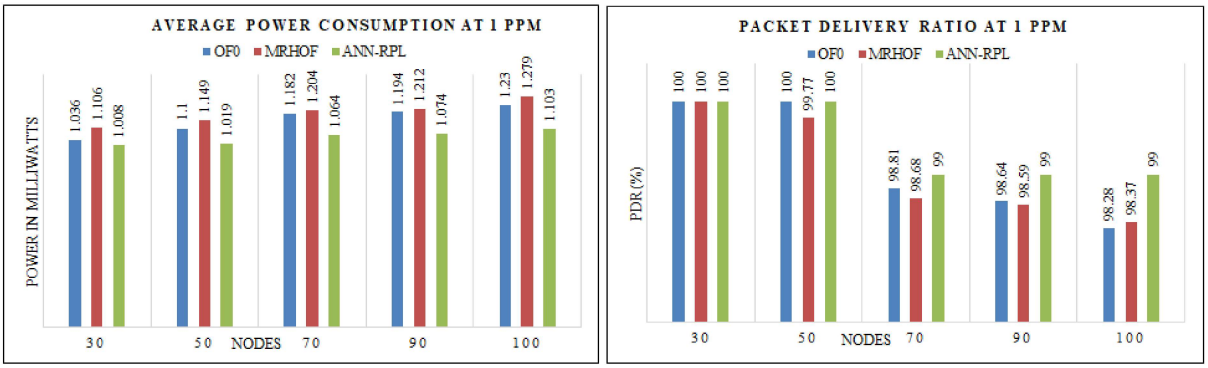


Figure 5.3: Power consumption and PDR with OF0, MRHOF and ANN-RPL for varying node density

Table 5.3: Improvement in Pavg and PDR with ANN-RPL over OF0 and MRHOF with varying node density at 1 PPM

	P(avg)(mW)			Improvement over		PDR			Improvement over	
	OF0	MRHOF	ANN-RPL	OF0	MRHOF	OF0	MRHOF	ANN-RPL	OF0	MRHOF
30	1.036	1.106	1.008	3%	9%	100	100	100	0.00%	0.00%
50	1.1	1.149	1.019	7%	11%	100	99.77	100	0.00%	0.23%
70	1.182	1.204	1.064	10%	11%	98.81	98.68	99	0.19%	0.32%
90	1.194	1.212	1.074	10%	11%	98.64	98.59	99	0.36%	0.41%
100	1.23	1.279	1.103	10%	14%	98.28	98.37	99	0.73%	0.64%
Average	1.1484	1.19	1.0536	8%	11%	99.156	99.08	99.4	0.26%	0.32%

The PDR with ANN-RPL is 100% at low density of 30 and 50 and drops to 99% as network

size increases to 100 nodes. While OF0 and MRHOF drops to 98.7% at 100 nodes. The readings of P(avg) and PDR obtained at various node densities and the corresponding percentage improvement of ANN-RPL over OF0 and MRHOF is listed in table 5.3.

The figure 5.4 depicts the Latency and Control Overhead with ANN-RPL versus OF0 and MRHOF. The latency varies from 300 to 600 ms with OF0 and MRHOF. In ANN-RPL it varies from 270 ms to 500 ms. At network size of 100 nodes, ANN-RPL ensures faster delivery by 100 ms, which translates to an improvement of 16%. ANN-RPL uses the delay and ETX metrics to choose faster and shorter routes. The readings of Latency and Overhead obtained at various node densities and the corresponding percentage improvement of ANN-RPL over OF0 and MRHOF is listed in table 5.4

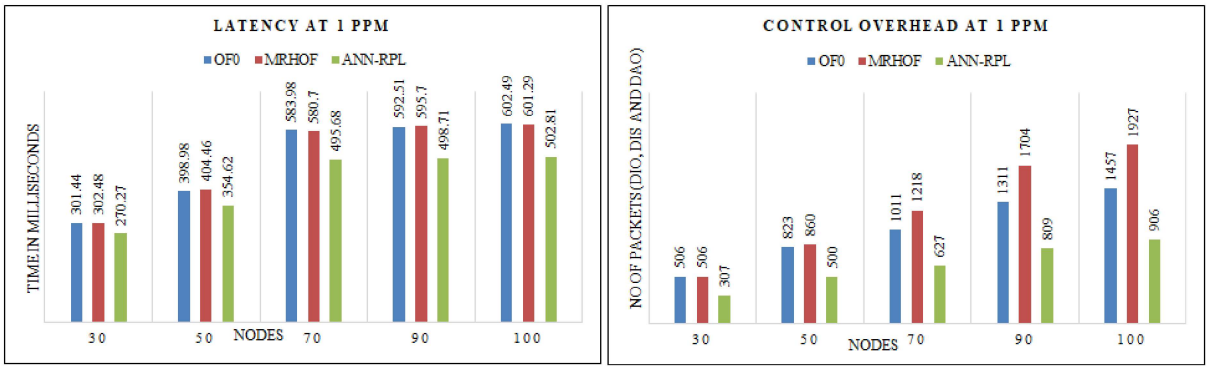


Figure 5.4: Latency and Control Traffic Overhead for OF0, MRHOF and ANN-RPL for varying node density

In RPL, three types of control messages are generated namely, DIO, DIS and DAO. The DAO messages are generated after the expiration of the DelayDAOtimer. In standard RPL, expiration of this timer happens at regular intervals leading to excess DAO messages circulating in the network. In this work the DelayDAO timer expiration is triggered on the selection of a new parent node. This ensures that DAO messages are generated only when crucial routing updates must be sent and avoids unwanted overhead. The DIO messages, which carry the crucial routing information are generated on expiration of the Trickle timer. This timer expires at predefined fixed intervals and after a change in parent node. Together DIO and DAO ensure that updated routing information is circulated within the network. This leads to significant amount of reduction in the control overhead due to the mechanism adopted in transmitting DAO and DIO messages. The control overhead in ANN-RPL is significantly reduced to the tune of 49%. This is an additional factor responsible for shorter delays in ANN-RPL. The reduced traffic overhead ensures lesser congestion and loss of packets owing to buffer overflow.

Table 5.4: Improvement in Network Latency and Control Overhead with ANN-RPL over OF0 and MRHOF with varying node density at 1 PPM

	Latency(ms)			Improvement over		Overhead			Improvement over	
	OF0	MRHOF	ANN-RPL	OF0	MRHOF	OF0	MRHOF	ANN-RPL	OF0	MRHOF
30	301.44	302.48	270.27	10%	11%	506	506	307	39.33%	39.33%
50	398.98	404.46	354.62	11%	12%	823	860	500	39.25%	41.86%
70	583.98	580.7	495.68	15%	15%	1011	1218	627	37.98%	48.52%
90	592.51	595.7	498.71	16%	16%	1311	1704	809	38.29%	52.52%
100	602.49	601.29	502.81	17%	16%	1457	1927	906	37.82%	52.98%
Average	495.88	496.93	424.42	14%	15%	1021.6	1243	629.8	38.35%	49.33%

Path chosen by Nodes-N1 and N2 to Sink with OF0, MRHOF and ANN-RPL

The paths taken by two nodes N1 and N2 towards the sink node are traced within a network comprising of 50 sender nodes. The paths are traced when network is operating using the Objective function of ANN-RPL, standard OF0 and MRHOF. The nodes in green are senders and one node in yellow is the sink. The figure 5.5 shows the paths chosen by two Nodes, N1 and N2 towards the sink node in a RPL instance. Consider the node N2, it has two neighbors within its transmission range, with metric values as follows: N12: = (ETX=8, RE= 230, Delay = 800 and HC= 3) and N11: = (ETX=7, RE= 175, Delay = 1200 and HC= 3). The ANN-RPL yields a Quality Score of 68 for neighbor N12 and 59 for N11. Since score of N12 is higher, ANN-RPL chooses N12 as parent for route towards sink. Whereas OF0 and MRHOF choose N11. Since N11 has lower energy and higher delay then N12, the performance of OF0 and MRHOF networks will be poor as compared to ANN-RPL. It is evident from the figure that ANN-RPL chooses optimal parents which are having higher energy levels and lower delays as contrary to that seen with OF0 and MRHOF. The table 5.5 show the metric values for the candidate parents of N1 and N2 and the quality scores computed by the ANN.

Table 5.5: Four Metric values and computed Score by ANN for neighbors of N1 and N2

	Inputs				Output
Neighbors of Node N1	ETX	Delay	HC	RE	Quality ANN-RPL
N3	8	1200	3	200	61
N4	2	1500	3	190	68
N5	5	800	2	150	63
N6	6	1000	2	225	70
N7	2	1000	2	200	74
N8	4	500	1	230	79
N9	5	700	1	200	71

	Inputs				Output
Neighbors of Node N2	ETX	Delay	HC	RE	Quality ANN-RPL
N10	6	900	3	180	64
N11	7	1200	3	175	59
N12	8	800	3	230	68
N13	5	400	2	190	72
N14	8	700	2	200	65
N15	11	500	2	250	68
N16	8	400	1	240	74
N17	9	500	1	250	73

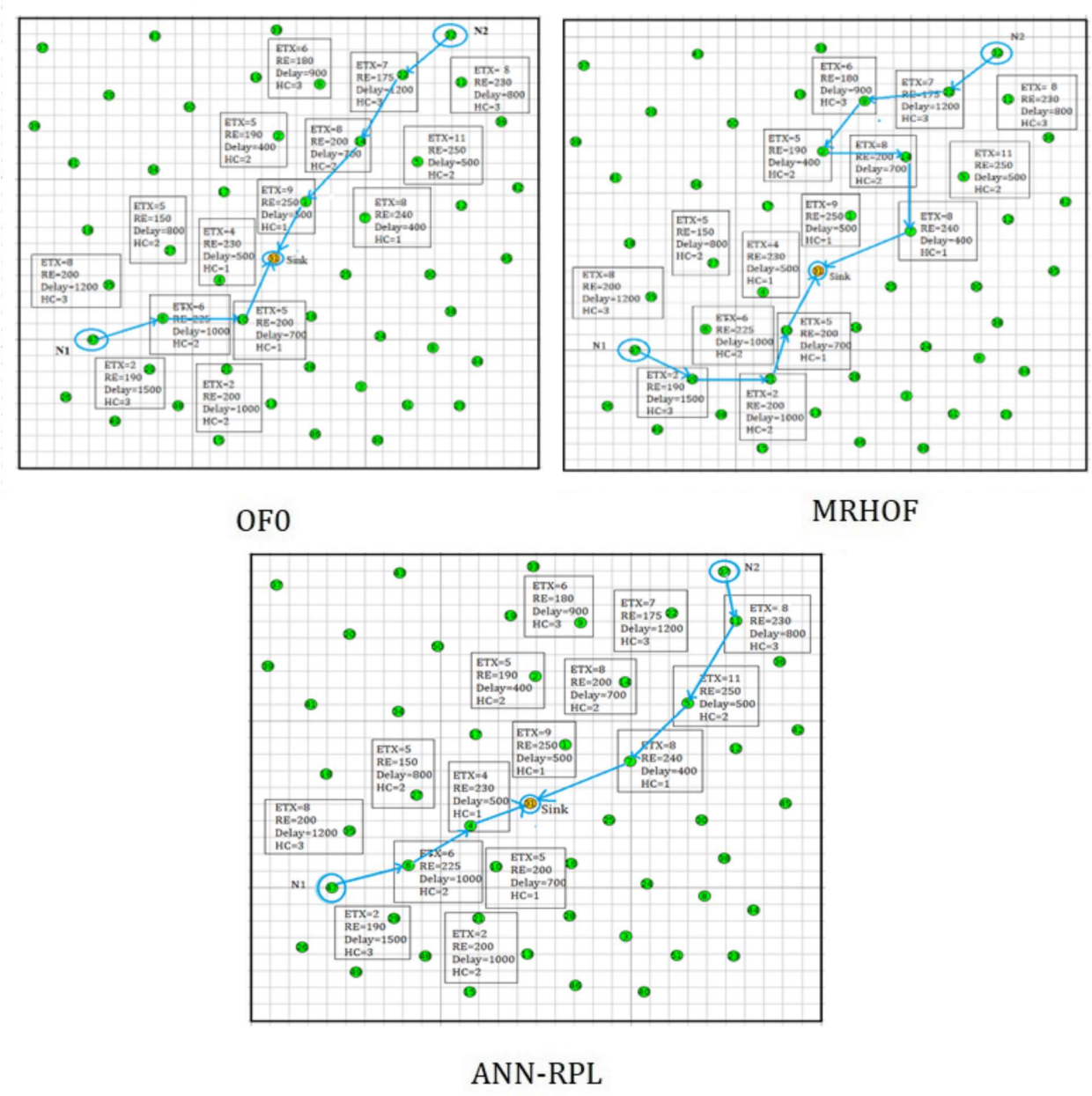


Figure 5.5: Route chosen by Nodes N1 and N2 to Sink with OF0, MRHOF and ANN-RPL

Scenario 2: In the second simulation setup, the data rates are varied from 2 ppm to 10 ppm. The graphs of power consumption, latency , PDR and overhead at the above data rates are plotted.

Average Power Consumption

The graphs in figure 5.6 depict the power consumption at varying data rates of 1ppm , 2 ppm, 5 ppm and 10 ppm as network density is increased from 30 to 100.

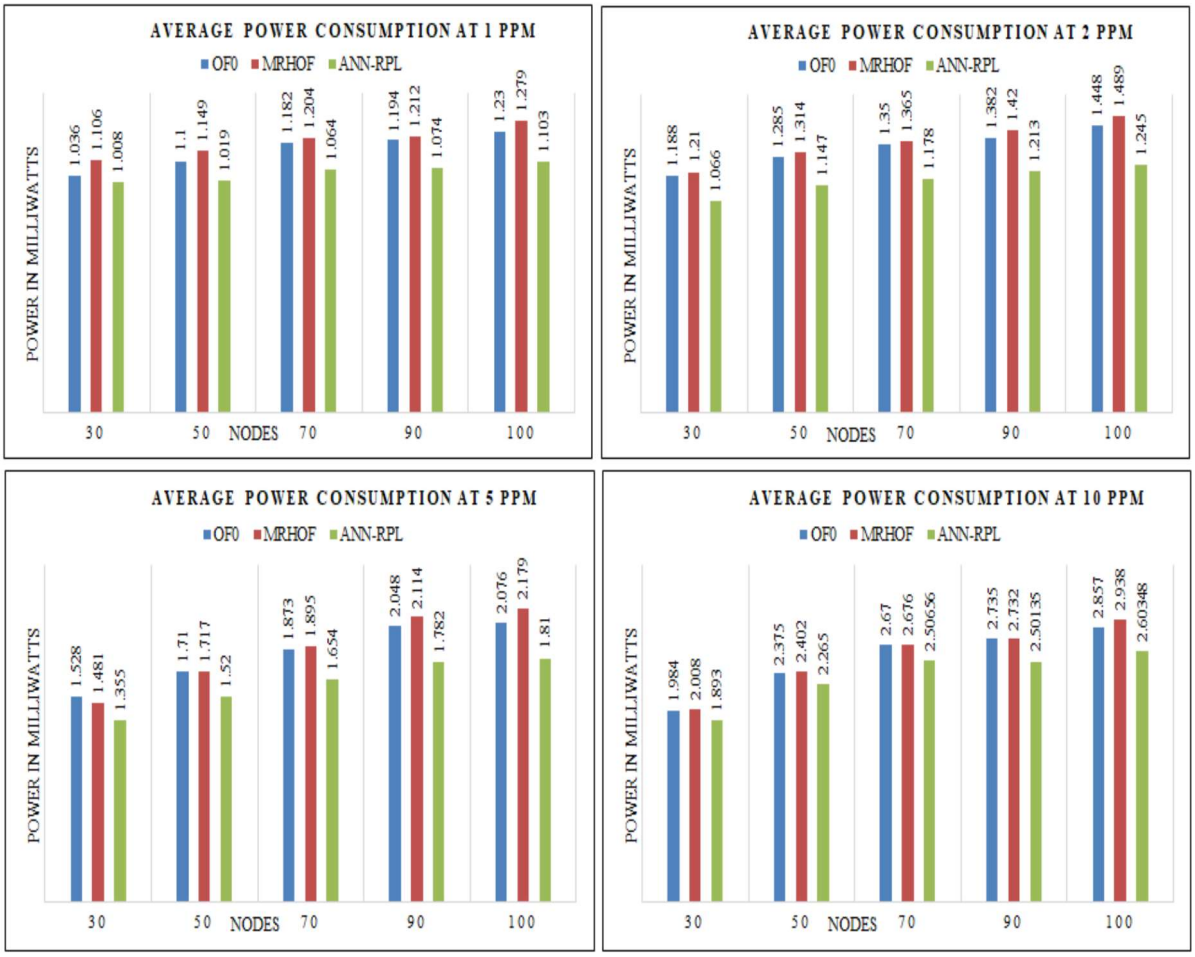


Figure 5.6: Average power consumption for OF0, MRHOF and ANN-RPL at varying data rates

The power consumed increases with the data rate, being highest at 10 ppm and 100 nodes. In a 100 node network setup, ANN-RPL consumes significantly lower power than OF0 and MRHOF to the tune of 9% and 11% at 10 ppm. The power consumed at 50 nodes and 100 nodes and the corresponding improvement obtained with ANN-RPL is listed in table 5.6.
The average power consumption of each node is sum of the average power consumption in CPU state (P_CPU) that is activated whenever the node is active, LPM state (P_LPM) that is activated when the sensor node goes to Low Power Mode, Rx state(P_Rx) that node is active in the radio receive mode and Tx state(P_Tx)that radio is active in transmit mode.

$$P = P_CPU + P_LPM + P_Rx + P_Tx \tag{5.1}$$

ANN-RPL selects optimum parent nodes with low delays, small ETX values and higher energy levels. It ensures good packet delivery success, thus saving the energy spent in transmission and reception on packet loss.

Table 5.6: Improvement in Average Power Consumption with ANN-RPL at 50 and 100 nodes for varying data rates

	Pavg (mW) at 50 nodes					Pavg (mW) at 100 nodes				
	OF0	MRHOF	ANN -RPL	Improvement over		OF0	MRHOF	ANN -RPL	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	1.1	1.149	1.019	7%	11%	1.23	1.279	1.103	10%	14%
2 ppm	1.285	1.314	1.147	11%	13%	1.448	1.489	1.245	14%	16%
5 ppm	1.71	1.717	1.52	11%	11%	2.076	2.179	1.81	13%	17%
10 ppm	2.375	2.402	2.265	5%	6%	2.857	2.938	2.60348	9%	11%

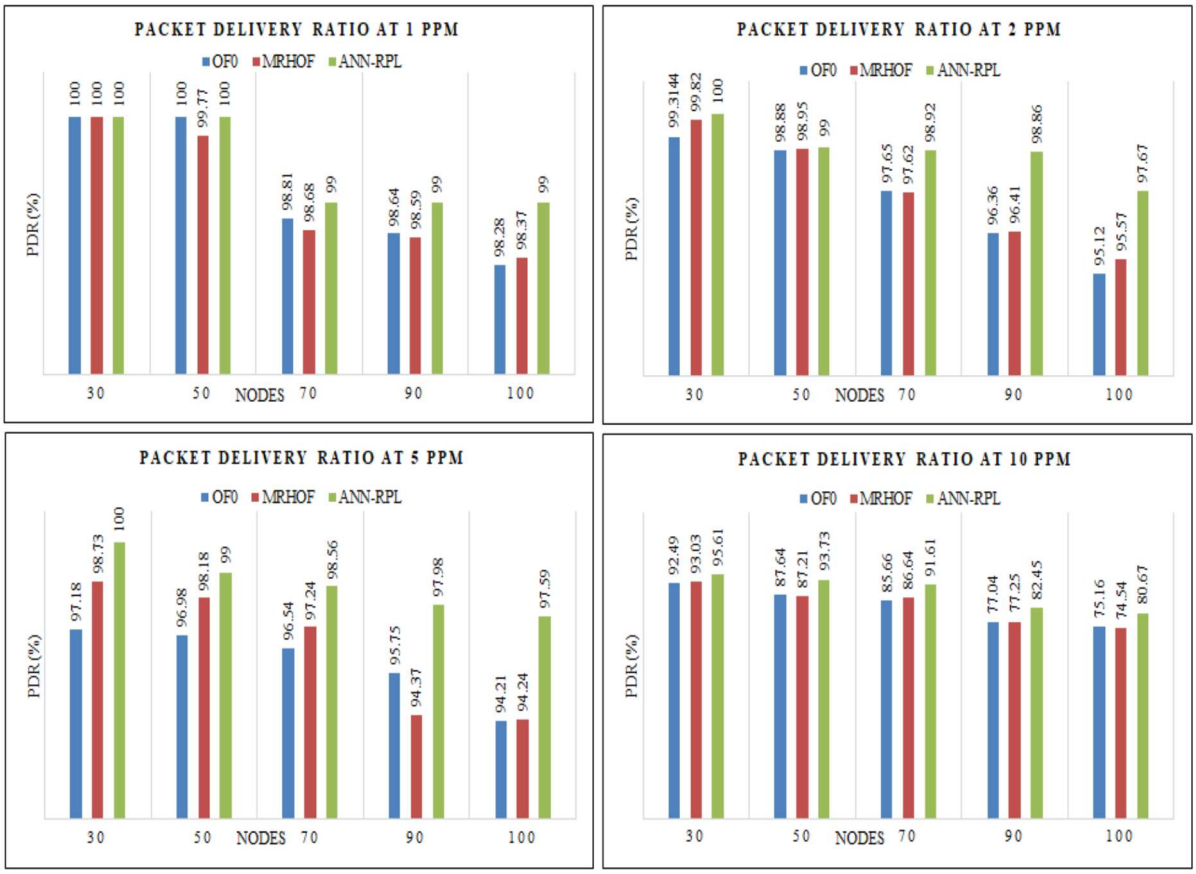


Figure 5.7: Packet Delivery ratios for OF0, MRHOF and ANN-RPL at varying data rates

Packet Delivery ratio

The Packet Delivery ratio with ANN-RPL , OF0 and MRHOF is shown by graphs in figure 5.7 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. In a network with 50 nodes, ANN-RPL gives packet delivery success of 99% upto 5 ppm. At 10 ppm, PDR with ANN-RPL is 94%, which is 7% higher than PDR achieved with OF0 and MRHOF. At higher node density, as packet rates are increased the PDR is seen to drop below 91%, maximum packet loss being at 10 ppm with 100 nodes density. A network with 100 nodes, each transmitting 10 packets /min , leads to a 1000 packets within network. Each node has a buffer size of 4 packets. When the number of packets circulating in the network increases, the buffers overflow and

many packets are dropped. This leads to the drop in PDR. ANN-RPL delivers packets with 8% higher success than OF0 and MRHOF at these strained conditions. The table 5.7 shows the improvements with ANN-RPL at 50 and 100 nodes with data rate varying from 1 p/min to 10 p/min.

Table 5.7: Improvement in PDR with ANN-RPL at 50 and 100 nodes for varying data rates

	PDR at 50 nodes					PDR at 100 nodes				
	OF0	MRHOF	ANN -RPL	Improvement over		OF0	MRHOF	ANN -RPL	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	100	99.77	100	0%	0%	98.28	98.37	99	1%	1%
2 ppm	98.88	98.95	99	0%	0%	95.12	95.57	97.67	3%	2%
5 ppm	96.98	98.18	99	2%	1%	94.21	94.24	97.59	3%	3%
10 ppm	87.64	87.21	93.73	6%	7%	75.16	74.54	80.67	7%	8%

Network Latency

The Latency with ANN-RPL , OF0 and MRHOF is shown by graphs in figure 5.8 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. The latency is seen to increase from 300 ms to 2s with OF0 and MRHOF as data rates are increased from 1 ppm to 10 ppm. At 50 nodes and 10 ppm, ANN-RPL gives 12% lower delays than other two. As nodes increase to 100 latency is 1.7 s in ANN-RPL, which is 18% lower than MRHOF. This is primarily because ANN-RPL uses the delay factor in addition to Hop Count while routing and thus selects faster routes. As shown in the figure 5.5 in case of route from Node N1 to root, OF0 and MRHOF choose paths with total delay of 1700ms and 3200 ms respectively whereas ANN-RPL chooses path from N1 to root with delay 1500s. In case of path from Node N2 to root, OF0 and MRHOF incur delays of 2400 ms and 3600 ms respectively, while ANN-RPL takes route with total delay of 1700 ms. The table 5.8 shows the percentage improvements in latency with ANN-RPL over OF0 and MRHOF at 50 and 100 nodes.

Table 5.8: Improvement in Network Latency with ANN-RPL at 50 and 100 nodes for varying data rates

	Latency (ms) at 50 nodes					Latency (ms) at 100 nodes				
	OF0	MRHOF	ANN -RPL	Improvement over		OF0	MRHOF	ANN -RPL	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	398.98	404.5	354.6	11%	12%	602.49	601.3	502.8	17%	16%
2 ppm	462.5	476.1	408.2	12%	14%	787.74	811.9	656.5	17%	19%
5 ppm	710.6	720.6	614.9	13%	15%	1478.9	1480.8	1219	18%	18%
10 ppm	1232.9	1231.6	1080.6	12%	12%	2021.9	2079.3	1704	16%	18%

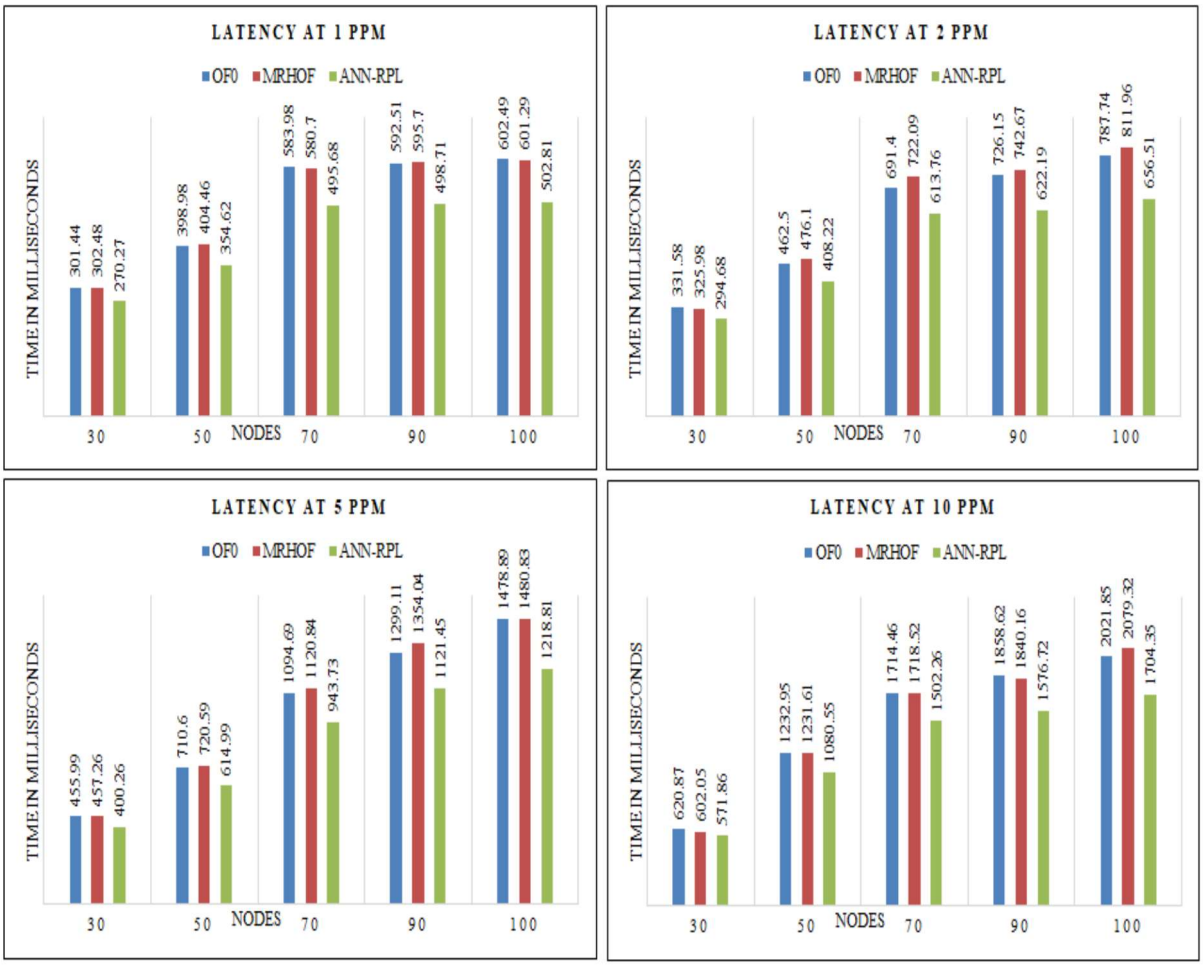


Figure 5.8: Latency for OF0, MRHOF and ANN-RPL at varying data rates

Control Overhead

The graphs in figure 5.9 depict the control overhead packets with ANN-RPL , OF0 and MRHOF as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm.

In RPL, three types of control messages are generated namely, DIO, DIS and DAO. In ANN-RPL, generation of the advertisement messages is curtailed. In standard RPL, DAO are generated after regular intervals leading to excess messages. In ANN-RPL, the DAO are transmitted only upon change in parent node, to update the same to neighbors. This mechanism avoids unwanted advertisement overhead.

The control overhead in ANN-RPL is significantly reduced to the tune of 47%. The reduced traffic overhead ensures lesser congestion and loss of packets owing to buffer overflow. The table 5.9 shows the percentage improvements in overhead with ANN-RPL over OF0 and MRHOF at 50 and 100 nodes.

Table 5.9: Improvement in Control Overhead with ANN-RPL at 50 and 100 nodes for varying data rates

	Control Overhead at 50 nodes					Control Overhead at 100 nodes				
	OF0	MRHOF	ANN -RPL	Improvement over		OF0	MRHOF	ANN -RPL	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	823	860	500	39%	42%	1457	1927	906	38%	53%
2 ppm	725	894	486	33%	46%	1281	1783	874	32%	51%
5 ppm	723	818	454	37%	44%	1228	1688	756	38%	55%
10 ppm	716	811	450	37%	45%	1091	1359	718	34%	47%

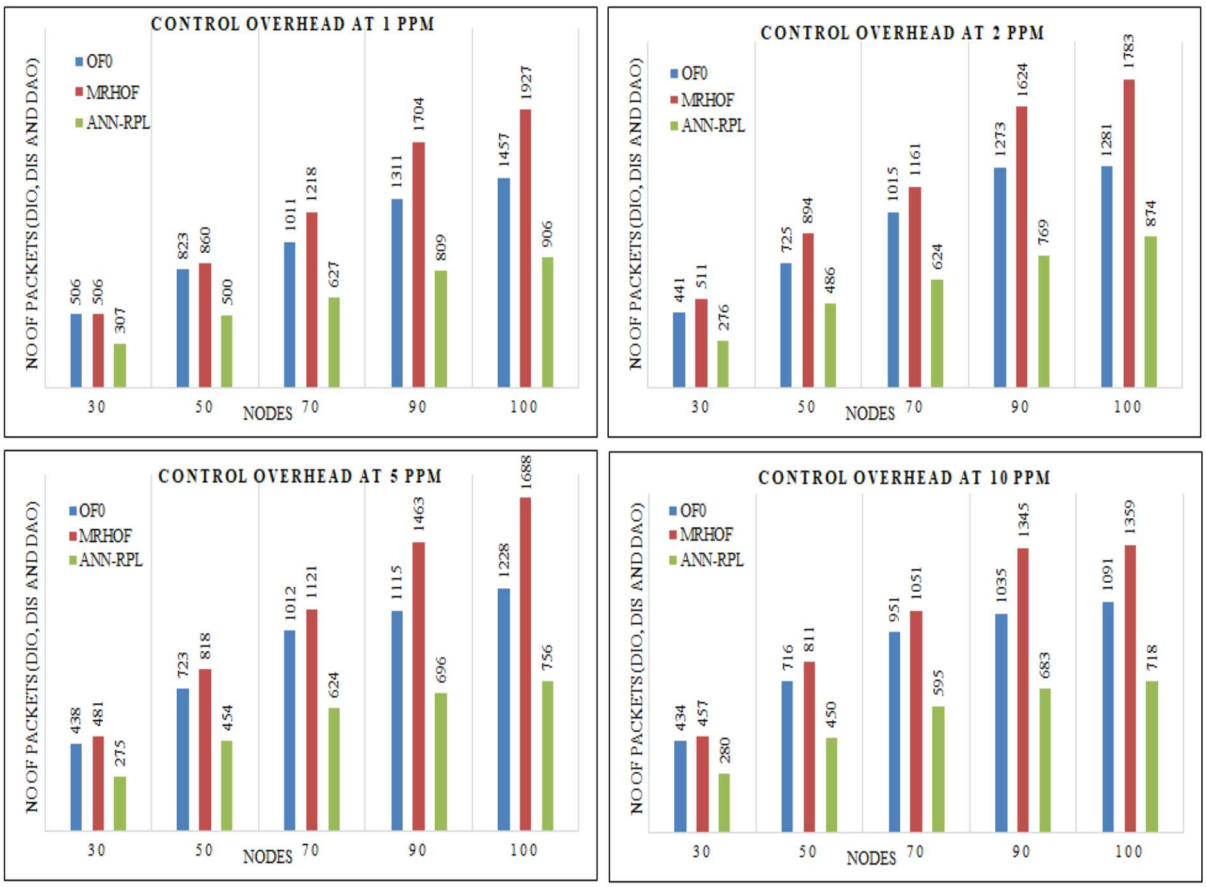


Figure 5.9: Control Traffic Overhead for OF0, MRHOF and ANN-RPL at varying data rates

5.5 Performance Comparison of ANN-RPL with other existing works

In this chapter an artificial neural network is proposed as an approach to compute the quality score of the candidate by evaluating the four metrics. In this section we provide performance analysis of ANN-RPL by comparing it to other popular works existing in literature. The works chosen are E-RPL [84], EEOPS-RPL [85], RPL-GCA [83] and DN-RPL [111]. The optimization techniques of Genetic Algorithm, Ant Colony optimisation and firefly optimization algo-

rithm are made use of by RPL-GCA, E-RPL and EEOPS-RPL respectively. DN-RPL employs deep nets to optimize the performance of RPL.

5.5.1 Comparison of ANN-RPL with DN-RPL

A deep dive analysis of the proposed ANN-RP and existing DN-RPL [111] is made by varying node density from 50 to 100 and data communication rate from 5 ppm to 10 ppm. The analysis of DN-RPL and ANN-RPL performance with regards to packet delivery success, energy consumption and end-to-end delay is done. The graphs in figure 5.10 exhibit the end-to-end delays for DN-RPL and ANN-RPL with varying node density and transmission rates. The end-to-end delay comprises of the time taken to transmit the packet by sender, propagation time through network before reaching the destination and the processing time at each intermediate node.

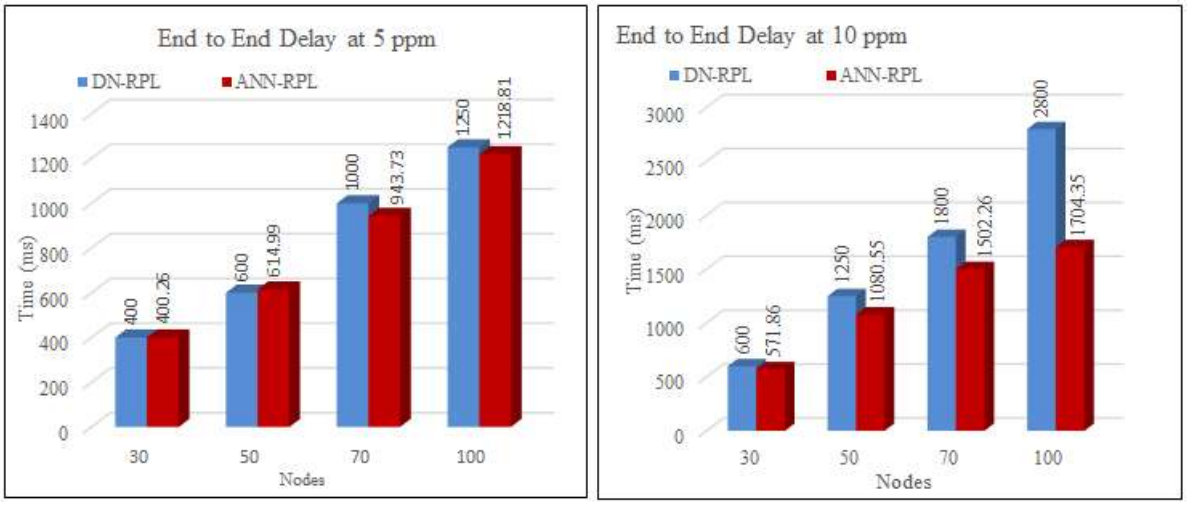


Figure 5.10: End-to-End Delay with DN-RPL[111] and ANN-RPL

The DN-RPL performs well at low node density of 30 and 50 nodes at transmission rate upto 5 ppm. It gives comparable delay as ANN-RPL in this setup. However as node density is increased to 70 and 100, then it starts suffering due to the large processing time taken by the deep-nets algorithm. At this point ANN-RPL gives 5% lower delays than DN-RPL. The performance of DN-RPL deteriorates with increasing packet rates.

At 10 ppm, ANN-RPL gives better results to the tune of 5%, 14%, 17% and 39% at 30, 50. 70 and 100 nodes respectively. The performance degradation is mainly attributed to the complexity of the deep-net calculation technique adopted in DN-RPL. DN-RPL relies on link quality, node energy, the node locations and euclidean distance between nodes. Additional time is required to compute the later two parameters besides the deep-net algorithm processing time. Another drawback of DN-RPL is that it does not consider the link delay in route construction.

This can lead to slower routes being taken during packet forwarding. The packet delivery success rate with varying network density is plotted in figure 5.11.

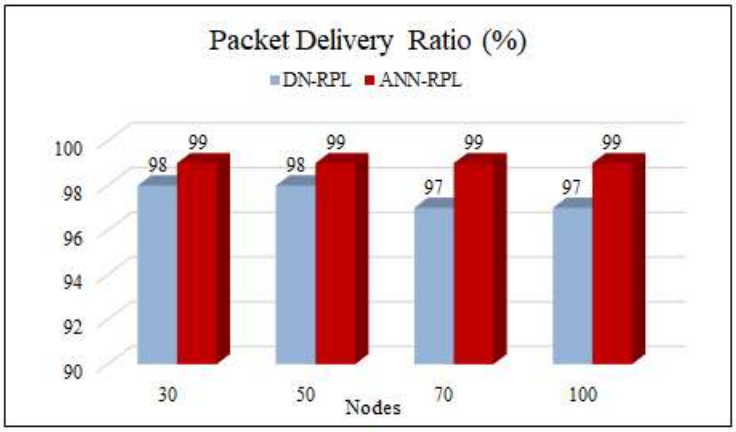


Figure 5.11: Packet Delivery Ratio with DN-RPL[111] and ANN-RPL

Both the techniques ensure packet delivery to destination with similar success rates. ANN-RPL scores an edge over DN-RPL by 2% at 70 and 100 nodes. The energy consumption plot is shown in figure 5.12. ANN-RPL has an average 17% and 25% lower energy consumption at 5 ppm and 10 ppm respectively than DN-RPL. The major energy consumption occurs during transmitting packets, receiving, listening, processing and forwarding packets. Contrary to DN-RPL, ANN-RPL requires lesser number of computations and thus saves on processing energy.

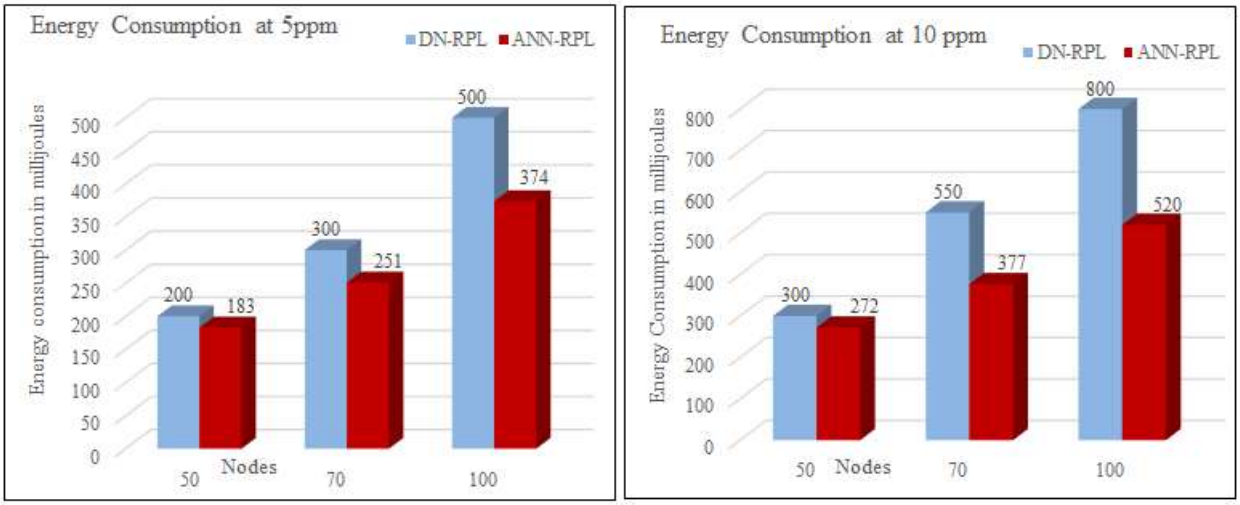


Figure 5.12: Energy Consumption with DN-RPL[111] and ANN-RPL

5.5.2 Comparison of ANN-RPL with other Evolutionary Algorithm based works

In this sub-section we compare ANN-RPL to other popular works existing in literature such as EPC-RPL, E-RPL [84], EEOPS-RPL [85], RPL-GCA [83] and HOFESA [112]. RPL-GCA uses the chaotic genetic algorithm for computing the weights in composition metric. EEOPS-RPL is based on firefly algorithm for finding best route. E-RPL and EPC-RPL use the ant colony optimization algorithms. HOFESA is a technique which uses multiple metrics and adopts a empirical value to prevent frequent parent changes thus maintaining stability.

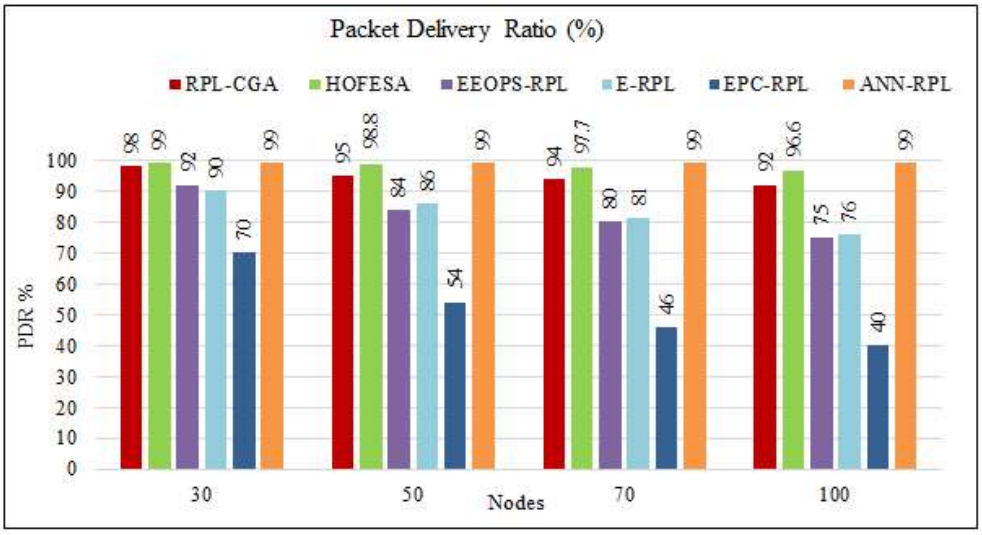


Figure 5.13: Comparison of PDR with RPL-GCA[83], HOFESA[112], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL

The graph of Packet delivery ratio with the above works is plotted in figure 5.13. The EPC-RPL has the least delivery success rate. RPL-GCA technique has good PDR of 98% at low node density, and reduces to 90% at 100 nodes. The firefly and ant colony techniques yield lower PDR of 92% at density of 30 and it decreases to 76% at 100 nodes. ANN-RPL scores above all these methods by ensuring 100% PDR at all node densities. The energy consumption with the above works is plotted in figure 5.14. Energy consumption with ANN-RPL is on an average 0.314 joules. It is lower as compared to that achieved with other existing works. RPL-CGA method has a comparable energy consumption of 0.3 J, this is primarily because the algorithm focuses on optimizing the energy consumption. HOFESA method has the highest energy consumption among all.

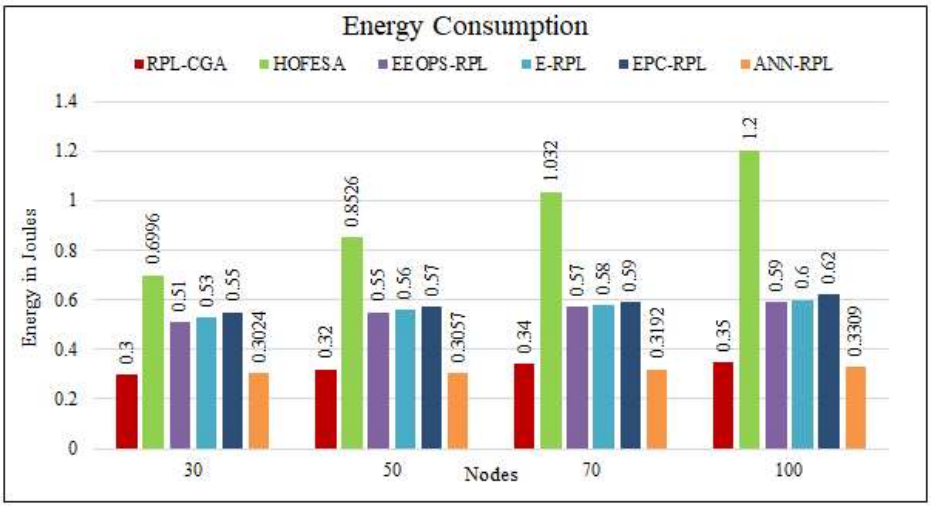


Figure 5.14: Comparison of Energy Consumption with RPL-GCA[83], HOFESA[112], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL

The plots in figures 5.15 depict the Network Latency or End to end delay in the works under consideration. Latency comprises of the transmitting time at sender, processing time at sender and forwarding time taken by each intermediate node. ANN-RPL method gives the least latency of 0.4 s. The other EA based methods have a latency of 0.4 mins. This is due to the complexity of the algorithms and time involved in processing at the sensor nodes.

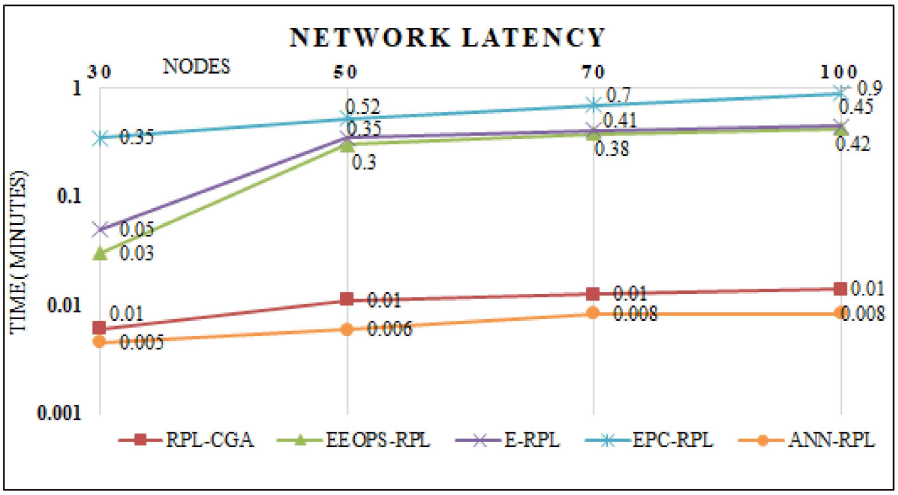


Figure 5.15: Comparison of Network Latency with RPL-GCA[83], E-RPL[84], EPC-RPL[84] , EEOPS-RPL[85] and proposed ANN-RPL

The control overhead with EPC-RPL, E-RPL and EEOPS-RPL and ANN-RPL is depicted in figure 5.16. ANN-RPL has marginally higher overhead than other three. Generally, lower overhead gives lower latency in any method. But in the case of EPC-RPL, E-RPL and EEOPS-RPL, the processing time overshadows the benefits reduced of control overhead.

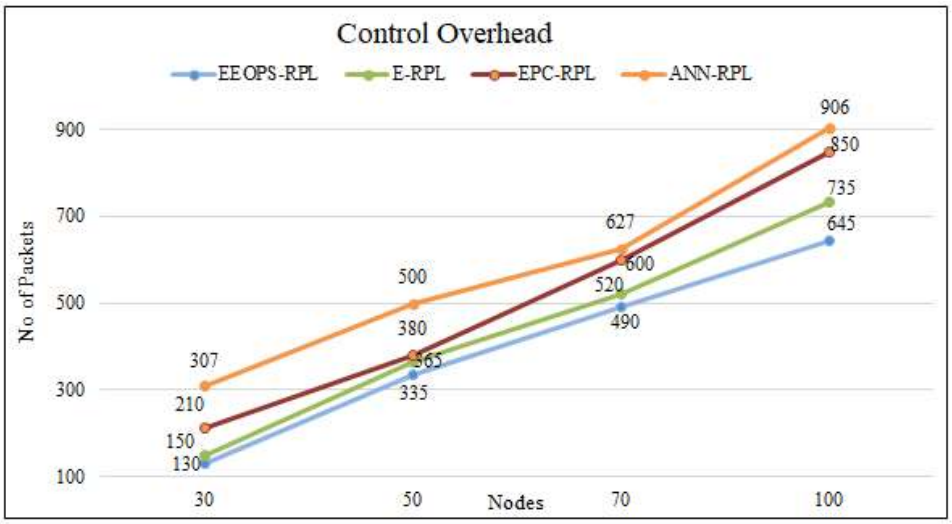


Figure 5.16: Comparison of control overhead with proposed ANN-RPL and existing works

Thus, it can be concluded that ANN-RPL is preferable regards to PDR and latency over the other methods with trade-off slightly higher energy consumption. The comparison summary of ANN-RPL versus other evolutionary algorithm based works is given in table 5.10.

Table 5.10: Comparison of ANN-RPL with other EA based works

Name	Technique Used	Metrics	PDR (%)	Energy Cons (J)	Delay	Control Ovhd
RPL-CGA [83]	Genetic Algorithm for optimizing weighting factors in composition metric	Queue length, Delay, RER, HC, ETX	95	0.33	0.65 s	-
EEOPS-RPL [85]	Firefly algorithm for optimization	RER, ETX and distance	82	0.56	0.28 min	400
E-RPL [84]	Ant colony optimization	Children count, RER, ETX, rank value	83	0.57	0.315 min	442
EPC-RPL [84]	Dynamic trickle and ACO	RER, ETX, rank value	52	0.58	0.59 min	535
HOFESA [112]	Hybrid Empirical stability aware	HC, RSSI, Energy	98	0.95	-	20500
ANN-RPL	ANN based Proposed work	ETX, HC, RE and latency	99	0.314	0.41 s	585

5.6 Conclusion

In this chapter a novel objective function ANN-RPL is proposed, which employs a Feed Forward Artificial Neural Network to choose the best route to the sink. The decision is taken based on multiple metrics. The performance of ANN-RPL is evaluated in comparison to the standard objective functions OF0 and MRHOF. They are analysed under conditions of varying network density and data rates. ANN-RPL uses the Residual Energy metric for choosing best parent, hence avoids using candidates with lower energy levels as parents. This preserves the node energy and improves overall efficiency. The delay and HC metric ensures that candidates with lower delay and shorter route are chosen as parents hence ensuring faster packet delivery. The latency in ANN-RPL is lower by 18% as compared to standard RPL at 10 nodes and 10 ppm. The ETX metric ensures better reliability and hence improves packet delivery. The PDR of ANN-RPL is higher by 7% at higher packets rates and network densities. It reduces the power consumption to the tune of 12%. There is significant amount of reduction in the control overhead due to the mechanism adopted in transmitting DAO and DIO messages. The traffic reduction to the tune of 49% is achieved. This novel ANN-RPL technique proves to be substantially better than standard RPL. The evolutionary algorithm based RPL implementations require more computing time due to the large number of iterations involved in the algorithms. On an average, the delay is seen to be around 0.4 mins. Also, they need more powerful microcontrollers with larger memory space. This considerably increases the deployment cost. These factors give ANN-RPL a leverage over the EA-based schemes. ANN-RPL has much lower computation time since it involves only a series of addition and multiplication operations at the 3 layers of neurons. The delay is approximately 0.4 s and significantly lower than EA-based RPL. It can work well on most commercially available microcontrollers like the MSP430. This extended and improved RPL version is suitable for the various applications of WSN and IoT domain

Chapter 6

Design of Efficient Objective Function for RPL using Machine Learning

6.1 Introduction

The advent of Machine Learning began in 1959, when Arthur Samuel, an expert in the field of artificial intelligence and computer gaming came up with the terminology. He defined machine learning as "the field of study that gives computers the ability to learn without being explicitly programmed" [113]. Machine learning finds application in domains like image and speech recognition, traffic prediction, medical diagnosis, stock market prediction, virtual personal assistant and self driving cars [114]. The advantage of ML techniques is highly accurate prediction, can easily identify trends and patterns, requires no human intervention and provides scope for continuous improvement [115].

Machine learning having proved its effectiveness in diverse applications, we decided to explore the possibility of applying machine learning to predict the best route within a network employing the RPL protocol. Machine learning solves problems of classification and regression type. In classification problems, the input data has to be appropriately classified as belonging to a particular category. Whereas in regression, a relation has to be established between one or more input variables and a single dependent output variable.

In a network scenario, the routing parameters like hop count, delay or energy are the deciding factors of best path. A relation needs to be developed between the best route and these routing metrics, thus regression analysis is needed here. This chapter evaluates three popular machine learning methods of Linear Regression, Random Forest and K-nearest neighbor. The Random Forest and the K-nearest neighbor techniques proves superior in terms of accuracy, and are cho-

sen to make the best route prediction. This novel version of RPL which applies the machine learning algorithm to choose the best parent node in route towards the root is termed as RPL-ML. The performance analysis of RPL-ML is carried out in Contiki OS COOJA Simulator by varying number of nodes and data packet rates. The packet delivery success ratio, energy consumption, latency and control overhead are found to be substantially lower with RPL-ML as compared to the conventional OF0 and MRHOF RPL implementations.

6.2 Research Objective

The main aim of this research is to design a routing strategy that overcomes the single metric-based routing limitation of existing standard RPL protocol. It is modeled as a multi-constrained optimal path problem, where best route decision is based on multiple routing metrics and machine learning regression techniques are employed to determine the optimal route toward the sink node.

The objectives of this research is:

To apply the Random Forest and K-Nearest Neighbor regression techniques of Machine Learning to take the best route decision by evaluating the four metrics.

6.3 Machine Learning Regression Analysis

Regression analysis is a powerful statistical process for estimating the relationship between a single dependent variable and one or more independent variables. Conventional regression analysis involves complex statistical models to estimate the dependent variable. As the nodes are equipped with micro controllers with limited computation power and memory, running the statistical models on them is not practically possible. Recent studies have shown that Machine Learning algorithms are effective in accurately predicting the dependent variable in regression analysis. Two popular Machine Learning regression techniques are Random Forest and K-Nearest Neighbor. The performance of these two is compared with the conventional linear regression method. This section describes these three techniques and presents their performance in terms of accuracy, prediction and computation time.

6.4 Linear Regression

Linear Regression comes from a family of statistical processes which date back to 1805. Originally the idea was coined by Sir Francis Galton, but later a precise mathematical model was presented by Sir Karl Pearson. It is a linear approach to model the relation between one or more explanatory variables and a single scalar response [116]. It relates one or more independent variables to a single dependent variable. In this case the independent variables are ETX, HC, RE and Delay. A relation must be established between these four and a single dependent variable called Quality score. Routes with optimal values of ETX, HC, RE and Delay should yield a higher quality score while poor values should result in low quality value. The constants a1, a2, a3 and a4 are defined to linearly relate the four metrics and Q score as below:

$$Qualityscore = a_1 * ETX + a_2 * HC + a_3 * Delay + a_4 * RE$$

The permissible range defined for ETX is from 1 to 15, HC is from 1 to 5, Delay is from 0 to 3000 ms and RE from 0 to 255. A training data set was created with 85,686 sample values where all exhaustive combinations of the four metrics were taken and a corresponding Quality value was assigned for each combination. In case of best-case scenario i.e ETX=1, HC=1, Delay=0 and RE=255, highest Quality value of 100 is assigned. As values of metrics degrade, lower Quality values are assigned.

Table 6.1: Training Dataset samples

Inputs				Output
ETX	Delay	Hop Count	Residual energy	Quality Score
1	1	1	255	100
2	71	4	251	94
3	1101	2	76	55
4	371	1	239	89
11	1501	1	111	48
14	2901	4	1	15
12	2801	4	16	20

Few samples of the data-set are shown in Table 6.1. This work is focused on using Machine learning to perform the Linear regression, so a python code was written in Jupyter notebook using the Scikit-learn models. Scikit-learn is an open source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem. The data-set was split as 70% training and 30% for validation. After training, the following relationship was established between the four metrics and the output Quality Score by linear regression.

$$Quality\ score =$$
$$(-1.8208) * ETX + (-0.5241) * HC + (-0.00885) * Delay + (0.1372) * RE + 62.9961$$

The method of linear regression gave an accuracy of 90%. An accuracy of 95% or higher is desirable, thus Linear Regression is not an ideal choice for this application. The accuracy, r2score and mean square error along with the training and prediction times achieved for Linear Regression are listed in table 6.2.

Table 6.2: Linear Regression Performance Parameters

Accuracy	90.698 %
R2score	0.9053
Mean Square Error	22.75
Training Time	0.338 s
Prediction Time	0.257 ms

6.5 Random Forest

Machine learning comprises of Supervised and Unsupervised learning Algorithms. In supervised learning a labeled data-set is needed for training. Random Forest is a supervised learning algorithm popularly used in solving classification as well as regression problems [117]. The original idea was proposed by Tin Kam Ho in 1995. The formal introduction of random forests was done some years later by Leo Breiman. Decision trees are constructed on different samples during training phase and the prediction mean/mode of individual trees is given as the regression output. This method provides high accuracy and can handle large data with multiple variables. Many trees are created on the subset of data and output of all the trees is combined. This helps to reduce the over fitting problem in decision trees as well as the variance thus improving the accuracy. The python code for Random Forest regression was written using Scikit-learn models. The data-set samples was used to train the random forest regressor. This method gave an accuracy of 99%. Random Forest regressor is considered a very powerful prediction technique compared to all other methods. The only drawback are the large training and prediction times involved. For applications where time is not a crucial factor and data sets are not huge, Random Forest is an ideal choice. The table 6.3 shows the performance parameters of RF.

Table 6.3: Random Forest Performance Parameters

Accuracy	99.19 %
R2score	0.9989
Mean Square Error	0.241
Training Time	6.52 s
Prediction Time	9.34 ms

6.6 K-Nearest Neighbour

The K-Nearest Neighbor algorithm is used for both regression and classification problems. It uses the concept of 'feature similarity' to predict output values for any given input data [118]. The new data point is assigned value depending on its resemblance to the points in the training data-set. The steps in the algorithms are as follows:

1. The distance between the new point and each training point is calculated. Some of the commonly used methods to calculate this distance are Manhattan (for continuous data), Euclidean and Hamming distance (for categorical). If the value A and the value B are the same then distance D = 0 otherwise D= A - B. If there are more than one independent variables, then the total distance is the summation of the difference in values of each variable as shown below in equation 1, where 'n' is the number of independent variables. A is the new point and B is the training point. In this manner the distance D for each training point is computed.

$$D = \sum_{i=1}^{n} |A_i - B_i|$$

2. Select the value of K. This number determines the number of neighbors to look at while assigning the value to a new point. The closest K data points are selected based on their distance.

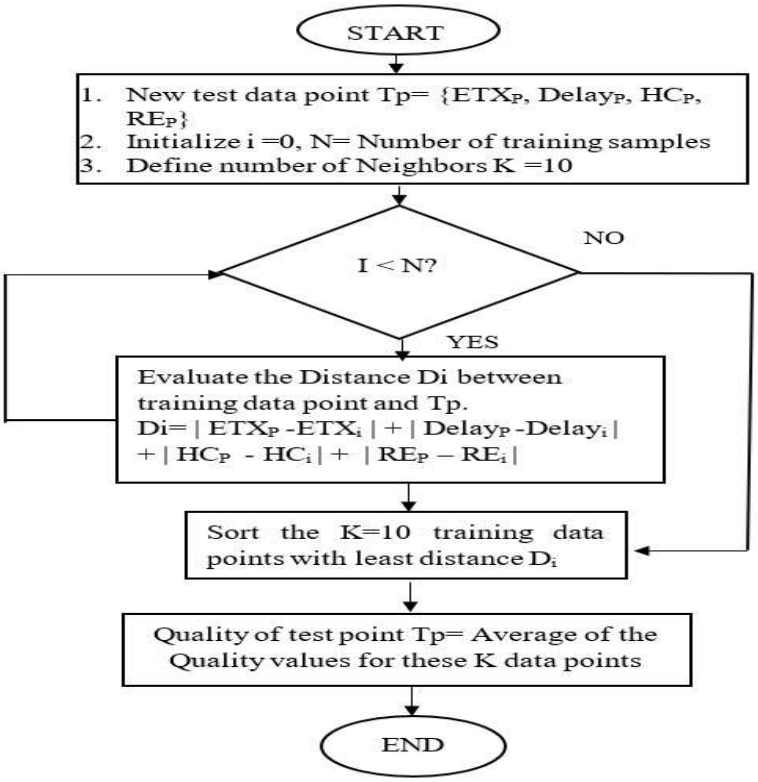


Figure 6.1: K-Nearest Neighbor Algorithm

The python code for K-Nearest Neighbor regression was written using sci-kit learn models. The K-NN regressor was trained using the same dataset as LR and RF. This method gave an accuracy of 98.17%. The K-NN algorithm is depicted in the flowchart of figure 6.1. The K-NN regression was performed by varying the K values from 1 to 40. The plots of Accuracy and mean error with k values are shown below in figure 6.2 and 6.3. The K=3 value is seen to give the best result of accuracy, R2score and prediction time.

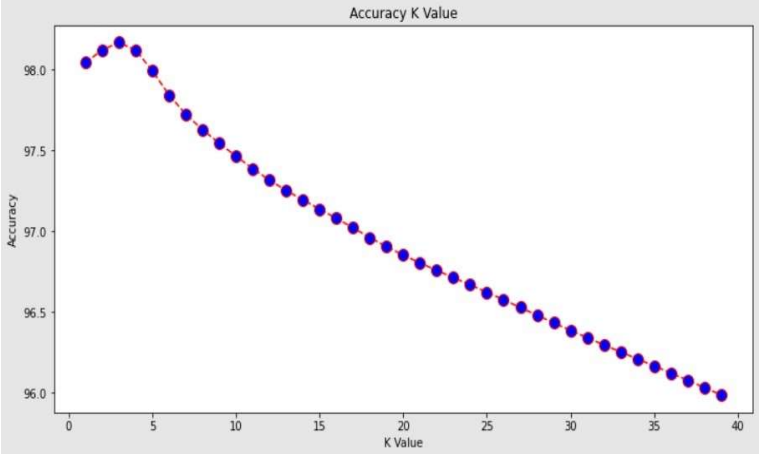


Figure 6.2: K-Nearest Neighbor Algorithm Accuracy vs K

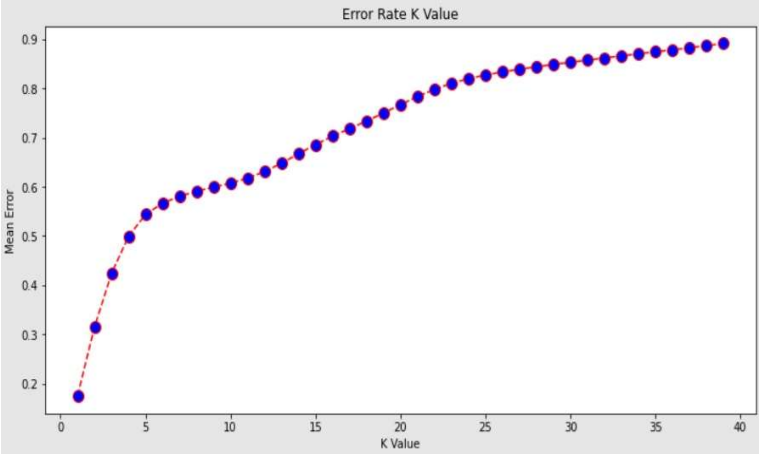


Figure 6.3: K-Nearest Neighbor Algorithm Error vs K

Table 6.4: K-NN Algorithm Performance Parameters

Parameter	K=2	K=3	K=4	K=5
Accuracy	98.12%	98.17%	98.12%	97.99%
R2score	0.993	0.997	0.994	0.994
Mean Error	0.314	0.4233	0.498	0.543
Training Time	28.55 s	29 s	29 s	32.7
Prediction Time	1.354 ms	1.29 ms	1.32 ms	1.31 ms

The prediction accuracy, computation time and training time for the above three regression techniques is compared as shown in the graphs 6.4 and 6.5.

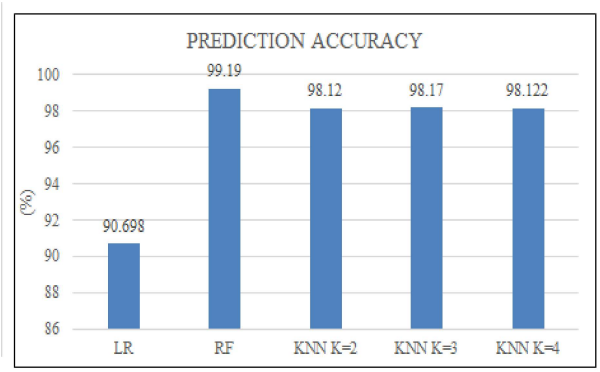


Figure 6.4: Prediction Accuracy for LR, RF and K-NN Regression

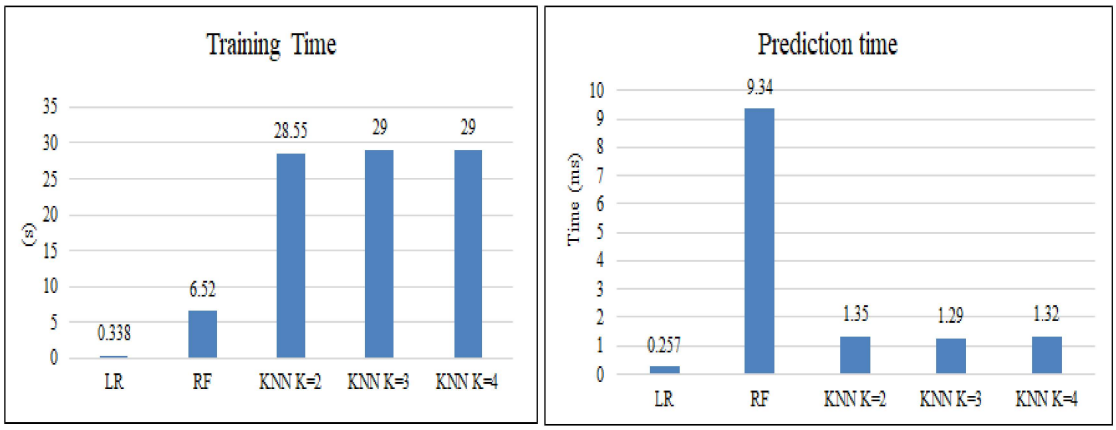


Figure 6.5: Prediction time and Training time for LR, RF and K-NN Regression

Random Forest and K-NN yield accuracy above 98%, while Linear regression gives lower accuracy of 90%. The prediction time needed by Linear regression is 0.257 ms and by Random Forest of 9 ms. The K-NN algorithm takes 1.3 ms for computation on an average. A training time of 6 s is required for Random Forest. This is due to the construction of multiple decision trees by the algorithm. K-NN regressor has training time of 29 s. Considering accuracy and prediction time, K-NN appears an ideal technique that can be applied for regression in our application scenario. So initially we experiment with the K-NN algorithm in RPL network. Many studies in classification suggest that the random forest is superior in performance to K-NN [119], [120]. So, in the later part of the chapter the Random Forest technique is also evaluated in the RPL network scenario.

6.7 RPL using K-Nearest Neighbor Machine Learning technique

The previous section has described three popular regression analysis methods.The K- Nearest Neighbor algorithm has an accuracy of above 98% and lower prediction time than the random forest technique. So the K-NN algorithm with K =3 value is selected to be implemented in the Contiki OS to attain the RPL-ML version. The C program for K-NN is integrated into the Contiki OS. To emphasize machine learning algorithm this novel version is termed as RPL-ML(K-NN). Along with the K-NN regression a threshold value PT is introduced in this RPL version. A Threshold Value PT is set that ensures less frequent changes in parent thus saving the energy required for network construction. The steps involved in optimal route selection using the K-NN based RPL implementation are as follows:

1. Upon receipt of a DIO message from neighbor node N, check if the rank of neighbor is less than rank of existing parent.
2. If $Rank(Parent) \geq Rank(neighborN)$ then add Neighbor N as a candidate parent to the parent list.
3. Extract the values of metrics HC, ETX, Delay, RER from the DIO message.
4. The K-NN regressor is invoked to decide the most suitable parent among the list of candidate parents.
5. Compute the regression score for all the candidate parents in the list.
6. Filter the candidate with maximum Quality score and compare it to the score of existing parent node.
7. Check if the candidate score exceeds the score of the present parent by the set threshold PT. If $QScore(Bestcandidate)$–$QScore(Existingparent) \geq PT$, then select this candidate as the new best parent.
8. Compute Rank of node proportional to the rank of the new selected parent using Rank equation.
9. If parent node is not changed then update the Quality score of the existing parent node by soliciting a DIO message from parent.
10. Send the DAO and DIO messages after expiry of respective timers.

6.8 Simulation Results of RPL-ML(K-NN)

Once K-NN algorithm is integrated into Contiki OS, the next step is to validate the performance of the designed model in a wireless network environment. This is done by running simulations in the in-built simulator COOJA. The simulation environment is set as shown in Table 3.1. The simulations are carried out in two scenarios. In first scenario, the density of nodes is varied from 30 to 100 and the performance is monitored at constant data rate of 1 ppm. In Scenario 2, the data rates are increased till 10 ppm. The four performance parameters-Power consumption, PDR, latency and control overhead are plotted from the simulation test logs generated by COOJA and the collect view tool.

Scenario 1:

The packet transmission rate is kept fixed at 1 ppm and the number of sender nodes in the network is varied from 30 to 100. Packets of length 128 bytes are transmitted at intervals of 60s. The value of K =3 is set in the K-NN algorithm for RPL-ML. The performance of RPL-ML is compared to OF0 and MRHOF. The figure 6.6 displays the graphs of power consumption and PDR for RPL-ML(KNN) and peers. An ideal network should be able to ensure delivery of maximum number of packets that too in real-time. The factors which can measure this is Packet Delivery Ratio (PDR) and Latency.

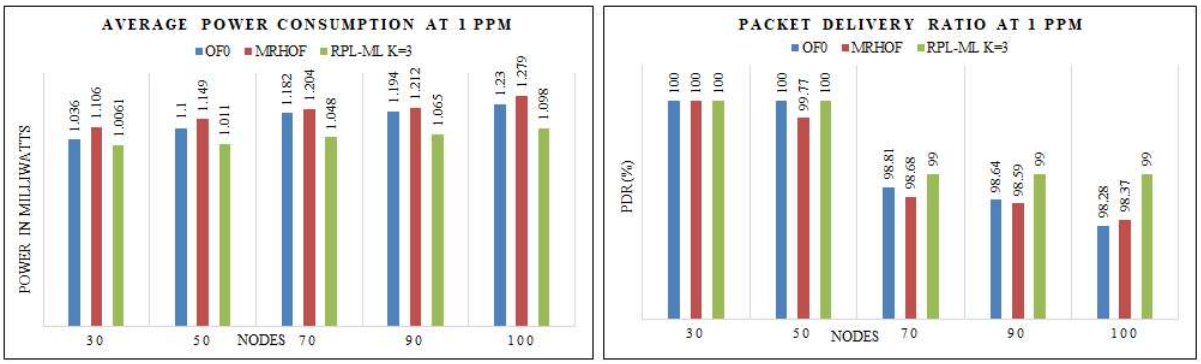


Figure 6.6: Avg Power consumption and PDR for OF0, MRHOF and RPL-ML(KNN) at varying node density

The power consumed should be minimum to ensure prolonged network lifetime. The motes require energy to transmit packets, listen for packets, receiving packets, forwarding packets, sensing and processing. The average power consumption gives an estimate of power spend by motes. The two types of packets circulating in the network include data and control overhead. The control packets carry routing information and advertisement messages. It is desirable to have minimal overhead to avoid congestion in network. At 30 nodes, the power consumption with MRHOF is 1.106 mW and with RPL-ML(KNN) it is 1 mW. As the network density

increases to 100 nodes, power consumed rises to 1.28 mW in MRHOF and 1.1 mW with RPL-ML(KNN).

Table 6.5: Improvement in Average Power consumption and PDR with RPL-ML(KNN) over OF0 and MRHOF at 1ppm

	Average Power Consumption(mW)					Packet Delivery Ratio				
	OF0	MRHOF	RPL-ML	Improvement over		OF0	MRHOF	RPL-ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
30	1.036	1.106	1.0061	3%	9%	100	100	100	0.00%	0.00%
50	1.1	1.149	1.011	8%	12%	100	99.77	100	0.00%	0.23%
70	1.182	1.204	1.048	11%	12%	98.81	98.68	99	0.19%	0.32%
90	1.194	1.212	1.065	11%	12%	98.64	98.59	99	0.36%	0.41%
100	1.23	1.279	1.098	11%	14%	98.28	98.37	99	0.73%	0.64%
Avg	1.148	1.19	1.046	9%	12%	99.146	99.082	99.4	0.26%	0.32%

The table 6.5 shows the comparison of proposed RPL-MK (KNN) with standard versions. On an average, the standard versions require 1.2 mW of power for transmission, reception, listening, forwarding and processing activities. With RPL-ML(KNN) a power of 1.05 mW is needed which is 12% lower than standard. The average PDR obtained with RPL-ML is 99.4%. RPL-ML constructs routes based on link quality and node energy. This helps to select good routes, ensuring successful packet delivery, thus saving energy of re-transmissions. The figure 6.6 shows the latency and control overhead in RPL with the three objective functions.

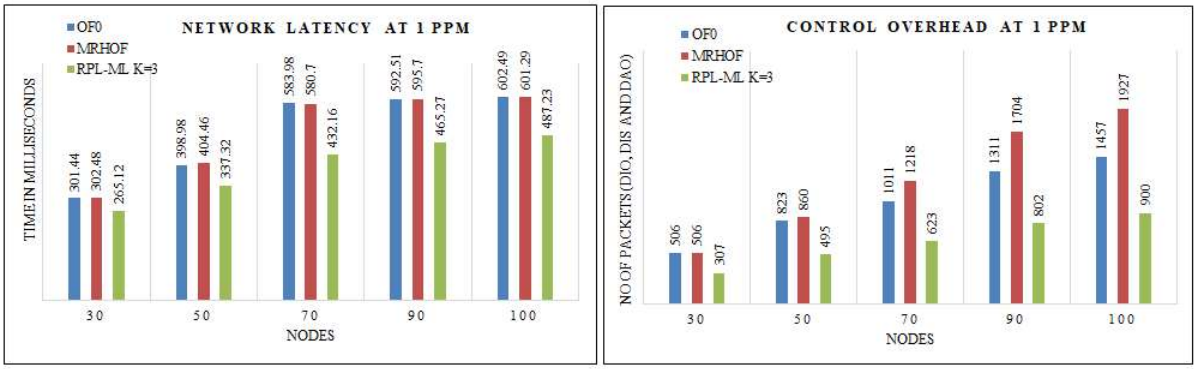


Figure 6.7: Latency and Overhead for OF0, MRHOF and RPL-ML(KNN) at 1 ppm

Table 6.6: Improvement in Latency and Overhead with RPL-ML(KNN) over OF0 and MRHOF at 1ppm

	Latency (ms)					Control Overhead				
	OF0	MRHOF	RPL-ML	Improvement over		OF0	MRHOF	RPL-ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
30	301.44	302.48	265.12	12%	12%	506	506	307	39%	39%
50	398.98	404.46	337.32	15%	17%	823	860	495	40%	42%
70	583.98	580.7	432.16	26%	26%	1011	1218	623	38%	49%
90	592.51	595.7	465.27	21%	22%	1311	1704	802	39%	53%
100	602.49	601.29	487.23	19%	19%	1457	1927	900	38%	53%
Avg	495.88	496.926	397.42	20%	20%	1021.6	1243	625	39%	50%

As seen from simulation results, the latency with MRHOF and OF0 varies from 300 ms to 600 ms as we go from 30 to 100 nodes. On the other hand RPL-ML(KNN) gives delays of 265 ms to 487 ms. On average, RPL-ML(KNN) results in 20% lower delays than standard RPL. The multi constrained optimal path routing in RPL-ML accounts for the reduction in latency. A secondary reason for lower packet loss and lower delays in RPL-ML is the reduction in traffic overhead, thus reducing congestion. The DIO and DAO advertisement messages in RPL-ML(KNN) are generated only when parent change occurs to advertise the routing updating information. This considerably reduces control traffic in RPL-ML to the tune of 50% as compared to standard OFs. MRHOF on other hand is designed to generates the control traffic at regular intervals depending on Trickle timer for DIO and DelayDAO timer for DAO messages. The table 6.6 shows the latency and overhead performance comparison of proposed RPL-MK (KNN) over standard versions.

Scenario 2: In the second simulation setup, the data rates are varied from 2 ppm to 10 ppm. The graphs of power consumption, latency , PDR and overhead at the above data rates are plotted.

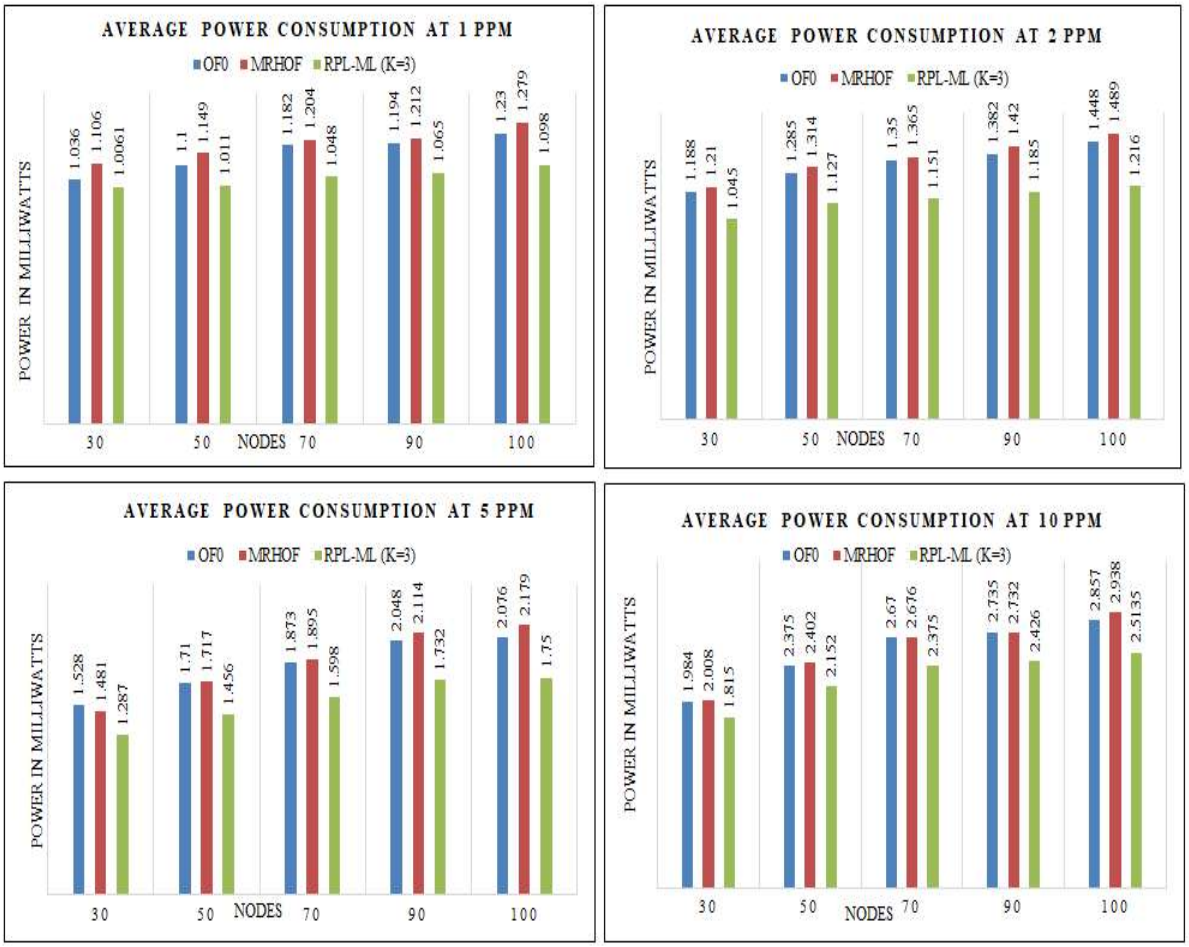


Figure 6.8: Avg Power consumption for OF0, MRHOF and RPL-ML(KNN) at varying data rates

Average Power Consumption

The graphs in figure 6.8 depict the power consumption at varying data rates of 1ppm , 2 ppm, 5 ppm and 10 ppm as network density is increased from 30 to 100. The power consumed increases with the data rate, being highest at 10 ppm and 100 nodes. In a 100 node network setup, RPL-ML(KNN) consumes significantly lower power than OF0 and MRHOF to the tune of 12% and 14% at 10 ppm.

Table 6.7: Improvement in power consumption with RPL-ML(KNN) for varying data rates

	Power Consumption (mW) at 50 nodes					Power Consumption (mW) at 100 nodes				
	OF0	MRHOF	RPL -ML	Improvement over		OF0	MRHOF	RPL -ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	1.1	1.149	1.01	8%	12%	1.23	1.279	1.098	11%	14%
2 ppm	1.285	1.314	1.13	12%	14%	1.45	1.489	1.216	16%	18%
5 ppm	1.71	1.717	1.46	15%	15%	2.08	2.179	1.75	16%	20%
10 ppm	2.375	2.402	2.15	9%	10%	2.86	2.938	2.51	12%	14%
Avg	1.62	1.65	1.44	11%	13%	1.90	1.97	1.64	14%	17%

The power consumed at 50 nodes and 100 nodes and the corresponding improvement obtained with RPL-ML(KNN) is listed in table 6.7. RPL-ML(KNN) selects optimum parent nodes with low delays, small ETX values and higher energy levels. It ensures good packet delivery success, thus saving the energy spent in transmission and reception on packet loss.

Packet Delivery ratio

The Packet Delivery ratio with RPL-ML(KNN) , OF0 and MRHOF is shown by graphs in figure 6.9 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. In a network with 50 nodes, RPL-ML(KNN) gives packet delivery success of 99% upto 5 ppm. At 10 ppm, PDR with RPL-ML is 95%, which is 8% higher than PDR achieved with OF0 and MRHOF. At higher node density of 100 and packet rate of 5ppm a 97% PDR is achieved with RPL-ML(KNN) which drops to 82% at 10 ppm. With OF0 and MRHOF, PDR is seen to drop drastically to 74% at 10 ppm with 100 nodes density. A network with 100 nodes, each transmitting 10 packets /min , leads to a 1000 packets within network. Each node has a buffer size of 4 packets. When the number of packets circulating in the network increases, the buffers overflow and many packets are dropped. This leads to the drop in PDR. RPL-ML(KNN) delivers packets with 8% higher success than OF0 and MRHOF at these strained conditions. The table 6.8 shows the improvements with RPL-ML(KNN) at 50 and 100 nodes with data rate varying from 1 p/min to 10 p/min.

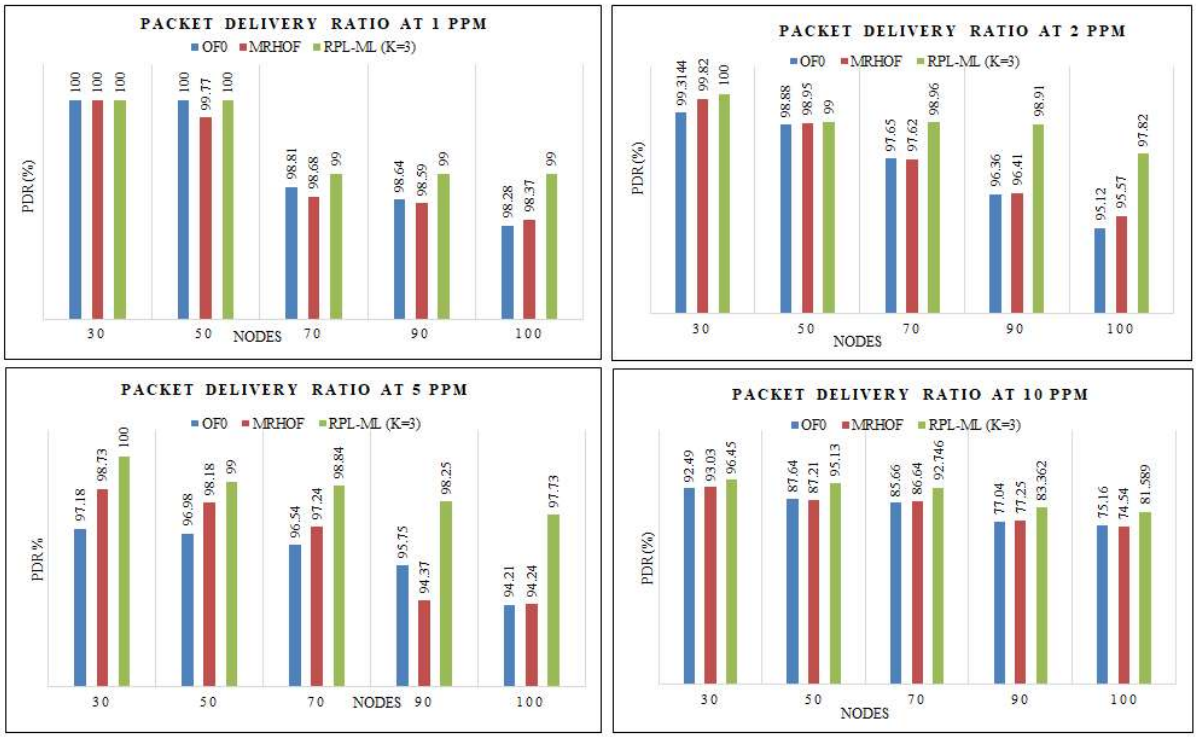


Figure 6.9: Packet Delivery ratio for OF0, MRHOF and RPL-ML(KNN) at varying data rates

Table 6.8: Improvement in PDR with RPL-ML(KNN) for varying data rates

	Packet Delivery Ratio at 50 nodes					Packet Delivery Ratio at 100 nodes				
	OF0	MRHOF	RPL -ML	Improvement over		OF0	MRHOF	RPL -ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	100	99.77	100	0%	0%	98.28	98.37	99	1%	1%
2 ppm	98.88	98.95	99	0%	0%	95.12	95.57	97.82	3%	2%
5 ppm	96.98	98.18	99	2%	1%	94.21	94.24	97.73	4%	4%
10 ppm	87.64	87.21	95.13	8%	8%	75.16	74.54	81.59	8%	9%
Avg	95.88	96.03	98.28	2%	2%	90.69	90.68	94.03	4%	4%

Network Latency

The Latency with RPL-ML(KNN) , OF0 and MRHOF is shown by graphs in figure 6.10 as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. The latency is seen to increase from 300 ms to 2s with OF0 and MRHOF as data rates are increased from 1 ppm to 10 ppm. At 50 nodes and 10 ppm, RPL-ML gives 15% lower delays than other two. As nodes increase to 100 latency is 1.6 s in RPL-ML(KNN), which is 20% lower than MRHOF. This is primarily because RPL-ML uses the delay factor in addition to Hop Count to perform multi constrained optimal routing and thus selects faster routes.

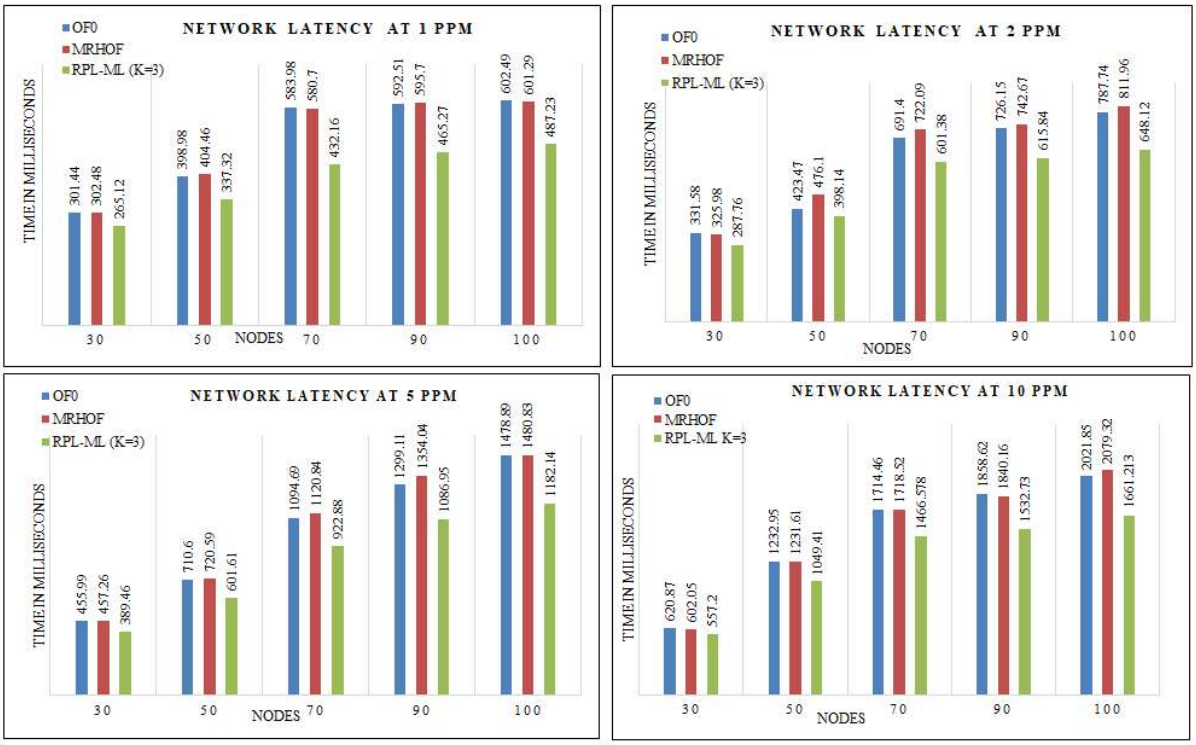


Figure 6.10: Network Latency for OF0, MRHOF and RPL-ML(KNN) at varying data rates

Table 6.9: Improvement in Network Latency with RPL-ML(KNN) for varying data rates

	Latency (ms) at 50 nodes					Latency (ms) at 100 nodes				
	OF0	MRHOF	RPL -ML	Improvement over		OF0	MRHOF	RPL -ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	398.9	404.46	337.3	15%	17%	602.5	601.29	487.2	19%	19%
2 ppm	462.5	476.1	398.1	14%	16%	787.7	811.96	648.1	18%	20%
5 ppm	710.6	720.6	601.6	15%	17%	1479	1480.83	1182.1	20%	20%
10 ppm	1233	1231.6	1049	15%	15%	2022	2079.32	1661.2	18%	20%
Avg	701.3	708.19	596.6	15%	16%	1222	1243.35	994.7	19%	20%

As shown in the figure 6.12 in case of route from Node N1 to root, MRHOF chooses paths with total delay of 1500ms respectively whereas RPL-ML chooses path from N1 to root with delay 1150s. In case of path from Node N2 to root, MRHOF incurs delays of 2450 ms , while RPL-ML(KNN) takes route with total delay of 1500 ms. The table 6.9 shows the improvements achieved with RPL-ML(KNN) in latency over standard at 50 and 100 nodes with varying data rates.

Control Overhead

The graphs in figure 6.11 depict the control overhead packets with RPL-ML(KNN) , OF0 and MRHOF as nodes increase from 30 to 100 and data rates increase from 1 ppm to 10 ppm. In RPL, three types of control messages are generated namely, DIO, DIS and DAO.

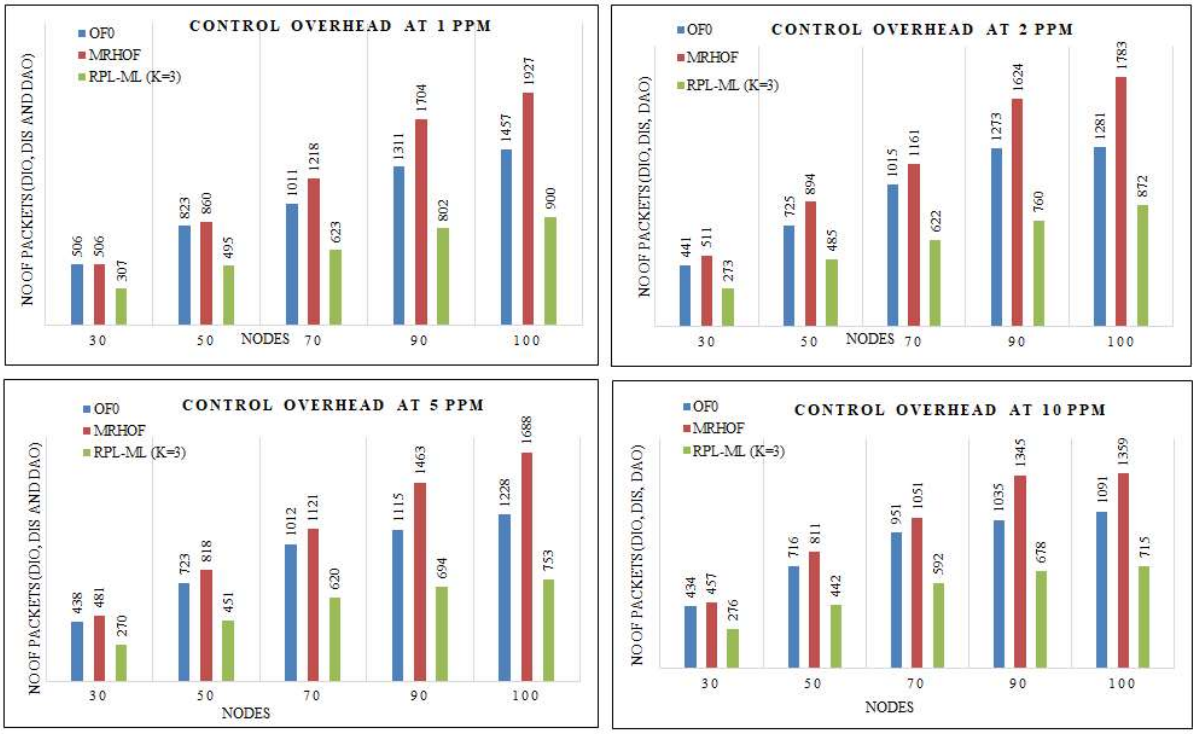


Figure 6.11: Control Overhead for OF0, MRHOF and RPL-ML(KNN) at varying data rates

In RPL-ML(KNN), generation of the advertisement messages is curtailed. In standard RPL, DAO are generated after regular intervals leading to excess messages. In RPL-ML, the DAO are transmitted only upon change in parent node, to update the same to neighbors. This mechanism avoids unwanted advertisement overhead, thus control overhead in RPL-ML is significantly reduced to the tune of 45%. The reduced traffic overhead ensures lesser congestion and loss of packets owing to buffer overflow. Every buffer at the node can hold up to 4 packets. With packets of size 128 bytes and 100 nodes transmitting 5 packets results in approximately 1 KB/s circulating in network. In addition to this sensed data, the control overhead is transmitted for routing updates. As can be seen from graph in figure the control overhead for RPL-ML(KNN) at 100 nodes is 753 packets. For OF0 and MRHOF it is 1228 and 1688 respectively. This additional overhead burdens the system and can cause packet loss due to limited buffer size at nodes. In this proposed version, the reduction achieved is around 50%, majorly the DAO advertisements being curtailed. RPL-ML(KNN) considers the delay of a route while making routing decision. This helps to avoid routes with longer delays. The table 6.10 shows the improvements achieved with RPL-ML(KNN) in control overhead against standard RPL at 50 and 100 nodes with varying data rates.

Table 6.10: Improvement in Control Overhead with RPL-ML(KNN) for varying data rates

	Overhead at 50 nodes					Overhead at 100 nodes				
	OF0	MRHOF	RPL -ML	Improvement over		OF0	MRHOF	RPL -ML	Improvement over	
				OF0	MRHOF				OF0	MRHOF
1 ppm	823	860	495	40%	42%	1457	1927	900	38%	53%
2 ppm	725	894	485	33%	46%	1281	1783	872	32%	51%
5 ppm	723	818	451	38%	45%	1228	1688	753	39%	55%
10 ppm	716	811	442	38%	45%	1091	1359	715	34%	47%
Avg	747	846	468	37%	45%	1264	1689	810	36%	52%

As number of packets circulating in the network are increased, there is depreciation in packet delivery rate, power consumption and latency across all three versions. Transmission of more packets demands additional energy and contributes more network traffic thus loading the network. The performance of OF0 and MRHOF are seen to degrade drastically in terms of packet delivery success rate. RPL-ML is seen to perform best at data rates between 1 ppm to 5 ppm. At these rates with 100 nodes, the packet delivery with RPL-ML(KNN) is 99% to 97.73% which is 4% higher than standard versions. RPL-ML(KNN) achieves 20% lower latency of 1182 ms as against delays of 1479 ms and 1480 ms for OF0 and MRHOF at 100 nodes, 5 ppm. At this setting, power consumption is 20% lower than MRHOF, thus making it preferable over standard RPL.

Path chosen by Nodes-N1 and N2 to Sink with OF0, MRHOF and RPL-ML(KNN)

RPL-ML(KNN) is designed to make better routing choices than standard RPL. To practically verify this,paths taken by two nodes N1 and N2 towards the sink node are traced within a network comprising of 50 sender nodes. The paths chosen by two nodes N1 and N2 in a particular RPL instance are depicted in figure 6.12. All the parent nodes with their corresponding values of ETX, delay, residual energy and hop count are plotted. Node N1 needs three hops to reach the sink node. At the first hop, there are two candidate parents:

- P1 with metric values [ETX=3, delay=200, HC=3 and RE=225]
- P2 with metric values [ETX=5, delay=250, HC=3 and RE=205].

At the second hop there are three candidate parents:

- P3 with metric values [ETX=7, delay=300, HC=2 and RE=110]
- P4 with metric values [ETX=8, delay=350, HC=2 and RE=145]
- P5 with metric values [ETX=8, delay=400, HC=2 and RE=120].

At the last hop, again choice of three candidate parents exists:

- P6 with values [ETX=1, delay=1000, HC=1 and RE=180]
- P7 with values [ETX=2, delay=600, HC=1 and RE=220]
- P8 with values [ETX=3, delay=400, HC=1 and RE=205]

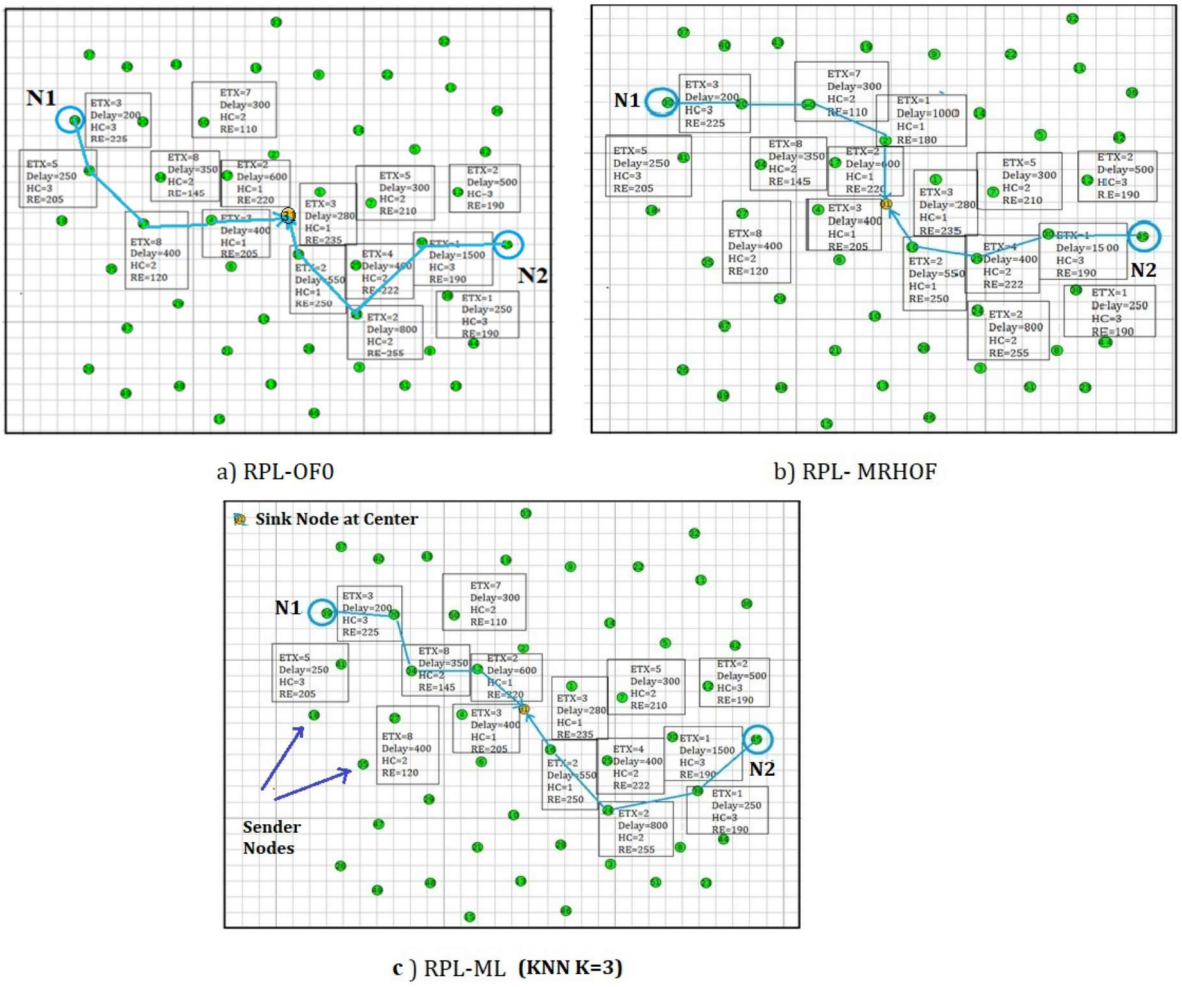


Figure 6.12: Routes taken by Nodes N1 and N2 towards sink node with a) RPL -ML(KNN) and b) Standard RPL

In RPL-ML(KNN) network, N1 chooses the parents P1, P4 and P7 in route towards the sink, whereas in RPL-MRHOF the parents P1, P3 and P6 are chosen depending on ETX values. As seen from the figure, node N1 chooses a route towards sink with RPL-ML resulting in a total delay of 1150 ms, while a delay of 1500 ms is obtained with standard RPL. The parent node P3 has low value of residual energy and runs the risk of draining out. The parent P6 has high delay value and low RE, thus not a very smart choice as parent. Node N2 takes a route with delay of 1600 ms with RPL-ML(KNN) as compared to higher delay of 2450 ms with standard RPL. RPL-ML succeeds in taking paths with lower delay and chooses parent with higher energy levels.This clearly indicates the advantage of RPL-ML design as regards delay. The average power consumption with RPL-ML(KNN) is lower than standard version by 17%.

RPL-ML has better success rates in packet delivery thus saving energy on re-transmissions. The parent change threshold helps to avoid frequent updating of routing information and network re-construction in-turn saving energy. The residual energy metric in RPL-ML ensures that parents with higher levels of energy are chosen. Further energy depletion of weak nodes is prevented thus prolonging network lifetime. Reduction in generation of advertisement messages further saves the energy of transmission and reception. The parent tracing diagram in figure 6.12 depicts the contrasting route choices which occur with RPL-ML and standard MRHOF.

6.9 RPL-ML with Random Forest regressor versus RPL-ML (K-NN) with K=2,3,5

In the K-NN algorithm the value of K determines the accuracy of prediction. The value of K=3 was seen to give best accuracy and least error as indicated in graph 6.2. Setting K=2 and K=5 in the K-NN algorithm and its corresponding effect on the performance of the network is studied in this section. The implementation of the Random Forest regressor to perform the multi constrained optimal path computation in RPL is also studied and compared with K-NN. The network density is varied from 30 to 100 nodes and the packet rates are varied from 1 ppm to 10 ppm. The plots of power consumption, latency and PDR are plotted for RPL-ML with K-NN algorithm for K=2,3 and 5 and RPL-ML with random forest regressor.

The figure 6.13 depicts the power consumption at varying data rates. The power consumption with K=2, K=3 and K=5 setting differs minimally, with K=3 giving the best results. The Random Forest algorithm was seen to outperform the KNN K=3 in RPL network scenario. RPL-ML (Random Forest) consumes 2% to 3% lower power than the RPL-ML(K-NN K=3). This is primarily due to the computation technique adopted in Random Forest. It involves true or false decisions and averaging, while the K-NN algorithm involves more computation intensive subtraction operations. Random Forest also requires comparatively lesser memory space than K-NN. This two factors make Random Forest more lucrative option over K-NN in RPL network implementation.

The power consumption P of each node is sum of the power consumed in CPU state (P_CPU) whenever the node is active and processing, LPM state (P_LPM) that is activated when the sensor node goes to Low Power Mode, Rx state(P_Rx) when node is active in the radio receive mode and Tx state(P_Tx) activated when radio is active in transmit mode. Average power con-

sumption is computed as shown in equation 6.2.

$$P = P_CPU + P_LPM + P_Rx + P_Tx \tag{6.1}$$

$$Average\ Power\ Consumption = \frac{\sum_{n=1}^{N} Pn}{N} \tag{6.2}$$

where N is the total number of nodes in network.

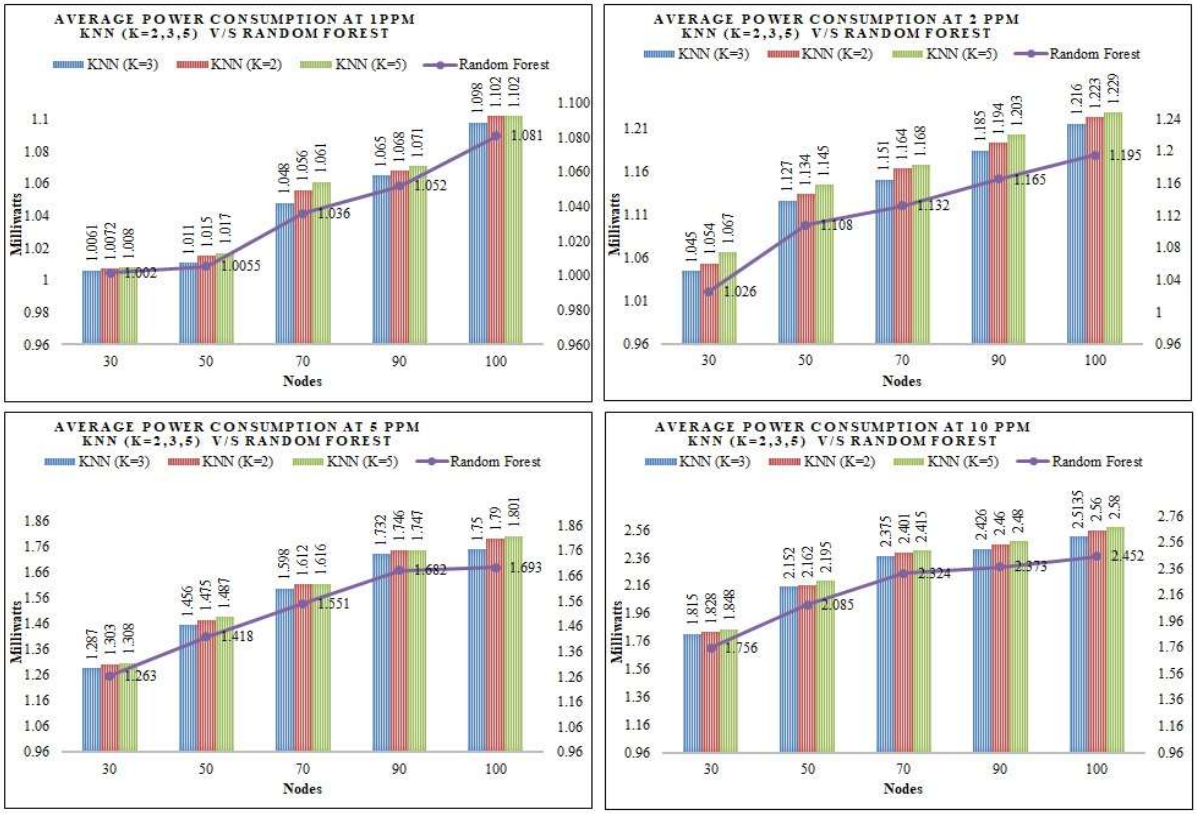


Figure 6.13: Power consumption for RPL-ML (KNN) vs RPL-ML (Random Forest)

The table 6.11 depicts the CPU power, LPM, Receive and Transmit powers consumed with the KNN (K=3) and Random Forest algorithms at varying nodes density and 1 PPM data rates. The values of P_CPU, P_LPM, P_Rx, P_Tx are obtained during simulation from the Collect View tool in COOJA. It is seen that at node density of 70, RF consumes 3% lower CPU power than KNN, and 4% lower at 90 and 100 nodes.

Table 6.11: CPU power for RPL-ML (KNN) versus RPL-ML (Random Forest)

	Average Power Consumption (mW) (RPL-ML K-NN)					Average Power Consumption (mW) (RPL-ML Random Forest)					CPU pwr %
Nodes	P_CPU	P_LPM	P_Rx	P_TX	Total PAVG	P_CPU	P_LPM	P_Rx	P_TX	Total PAVG	KNN >RF
30	0.342	0.141	0.443	0.080	1.006	0.338	0.141	0.442	0.080	1.002	1%
50	0.344	0.142	0.445	0.081	1.011	0.340	0.141	0.443	0.081	1.005	1%
70	0.356	0.147	0.461	0.084	1.048	0.344	0.146	0.462	0.084	1.036	3%
90	0.362	0.149	0.469	0.085	1.065	0.347	0.150	0.470	0.085	1.052	4%
100	0.373	0.154	0.483	0.088	1.098	0.358	0.153	0.482	0.088	1.081	4%

The table 6.12 shows the improvements in power consumption achieved with RPL-ML (random forest) at 50 and 100 nodes over the RPL-ML (K-NN) with varying data rates. At 100 nodes and 5 ppm, Random forest consumes 3% lower power than KNN K=3. Random Forest algorithm is found to be a more optimal solution for multi constrained optimal path decision than K-nearest neighbor.

Table 6.12: Power consumption of RPL-ML (KNN) vs RPL-ML(Random Forest) at varying data rates

	Power Consumption (mW) at 50 nodes					Power Consumption (mW) at 100 nodes				
	KNN K=3	KNN K=2	KNN K=5	RF	Improvement RF over KNN	KNN K=3	KNN K=2	KNN K=5	RF	Improvement RF over KNN
1 ppm	1.011	1.015	1.017	1.0055	1%	1.098	1.102	1.102	1.081	2%
2 ppm	1.127	1.134	1.145	1.108	2%	1.216	1.223	1.229	1.195	2%
5 ppm	1.456	1.475	1.487	1.418	3%	1.75	1.79	1.801	1.693	3%
10 ppm	2.152	2.162	2.195	2.085	3%	2.5135	2.56	2.58	2.452	2%
Average	1.44	1.45	1.46	1.40	2%	1.64	1.67	1.68	1.61	2%

The latency with RPL-ML K=2, 3 and 5 versus the latency with random forest regressor is plotted in figure 6.14 by varying the data rates. The latency values with K= 3,2 and 5 are minimally distinct. RPL-ML with Random Forest regressor gives the lower latency to the tune of 4% to 6% over K-Nearest Neighbor algorithm at all packet rates.

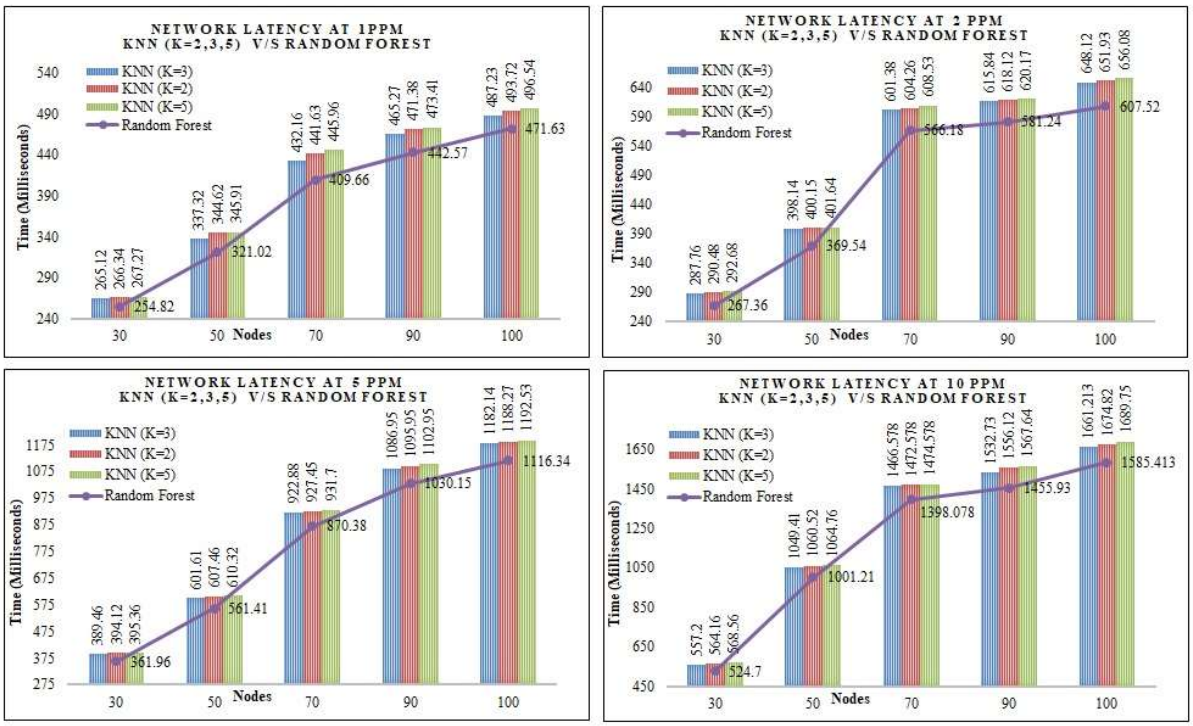


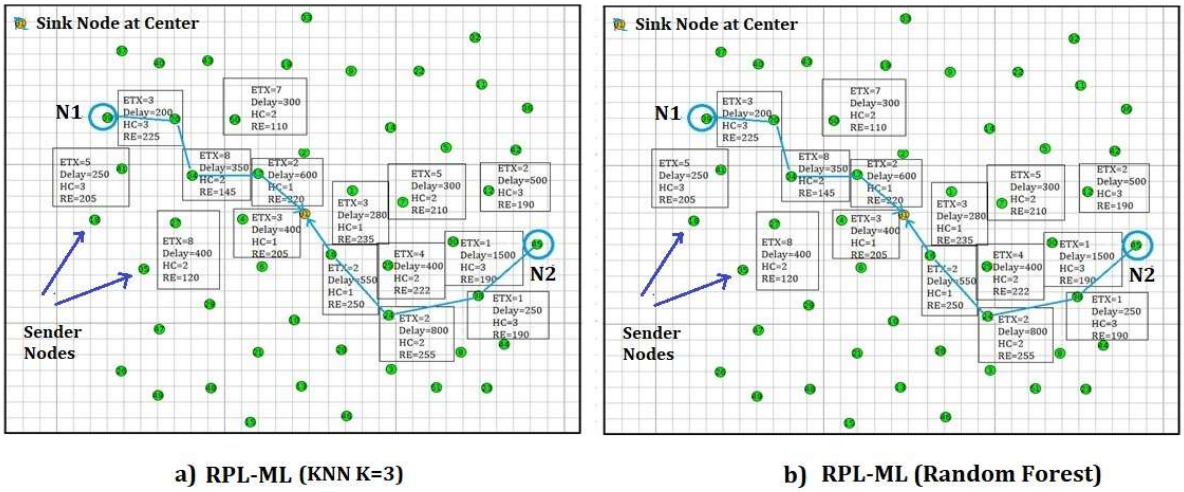
Figure 6.14: Network Latency for RPL-ML (KNN) vs RPL-ML (Random Forest)

Table 6.13: Improvement in Network Latency with RPL-ML (Random Forest) over RPL-ML (K-NN K=3)

	Network Latency (ms) at 50 nodes					Network Latency (ms) at 100 nodes				
	KNN K=3	KNN K=2	KNN K=5	RF	Improvement of RF over K-NN	KNN K=3	KNN K=2	KNN K=5	RF	Improvement of RF over K-NN
1 ppm	337.32	344.62	345.91	321.02	5%	487.23	493.72	496.54	471.63	3%
2 ppm	398.14	400.15	401.64	369.54	7%	648.12	651.93	656.08	607.52	6%
5 ppm	601.61	607.46	610.32	561.41	7%	1182.14	1188.27	1192.53	1116.34	6%
10 ppm	1049.41	1060.52	1064.76	1001.21	5%	1661.213	1674.82	1689.75	1585.413	5%
Average	596.62	603.19	605.66	563.30	6%	994.68	1002.19	1008.73	945.23	5%

The total network latency comprises of the transmission time, transit time and processing time. Processing in a sensor broadly consists of sensed data processing, received packets processing and the routing information updation. During the routing process, the prediction of best route by means of constructing decision trees in case of Random Forest is apparently faster than the difference and average method of K-NN algorithm. Both techniques are superior over the standard OF0 and MRHOF, but Random forest is preferable due to its relatively faster computation. The table 6.13 shows the improvements in latency achieved with Random Forest over K-NN at 50 and 100 nodes.

As seen above, the RPL implementation with Random Forest (RF) algorithm gives considerably lower latency. The main reasons for this reduction is the lower CPU processing time taken by random forest algorithm. To compare the routes taken with Random Forest with that of K-NN, we decided to trace the paths from nodes deployed in a RPL-ML (RF) network towards the root. Two distinct nodes N1 and N2 were chosen and the paths they choose towards the sink node were traced in both KNN (K=3) and RF networks with 50 sender nodes and 1 PPM data rate. The paths chosen by two nodes N1 and N2 in a particular RPL instance are depicted in figure 6.15. All the parent nodes with their corresponding values of ETX, delay, residual energy and hop count are plotted. We observed that Random Forest implementation takes the exact same routes as those of KNN K=3. This implies that the excess delay in K-NN is not due to non-optimal route selections. Thus we can attribute the increased latency to the extra CPU processing time in K-NN algorithm. The tables 6.14 and 6.15 show the Quality scores obtained with K-NN (K=3) and the Random forest algorithm for all the candidate parents in path from towards the sink from nodes N1 and N2.



a) RPL-ML (KNN K=3) b) RPL-ML (Random Forest)

Figure 6.15: Routes taken by Nodes N1 and N2 towards sink node with a) RPL -ML (KNN K=3) and b) RPL-ML (Random Forest Regressor)

Table 6.14: Quality Scores of Candidate Parents from N1 to Sink with KNN and RF

Q Score of Candidate Parent Nodes from N1 to Sink						
Sr No	ETX	Delay	HC	RE	Q Score KNN K=3	Q Score RF
1	3	200	3	225	88	89
2	5	250	3	205	86	82
3	7	300	2	110	65	65
4	8	350	2	145	68	69
5	8	400	2	120	63	62
6	3	400	1	205	84	84
7	2	600	1	220	86	87

As we travel from N1 to sink, at the outset two neighbors with Hop count of 3 are present. The quality scores computed for both neighbors are 88 and 86 with K-NN and 89 and 82 with RF. The node with ETX=3, Delay=200, HC=3 and RE=225 is of higher score and thus chosen as parent in case of both K-NN and RF. At the next level, three neighbors with hop count of 2 are present. The Q scores computed are 65, 68 and 63 with K-NN and 65, 69 and 62 by RF. The neighbor listed at Sr.No. 4 in table 6.14 with highest score (K-NN 68 and RF 69) is opted as parent in both networks. In the final step, two neighbors are present with hop count of 1. The node at Sr.No. 7 with highest Q score of 86 (K-NN) and 87 (RF) is selected as parent.

Table 6.15: Quality Scores of Candidate Parents from N2 to Sink with K-NN and RF

Q Score of Candidate Parent Nodes from N2 to Sink						
Sr No	ETX	Delay	HC	RE	Q Score KNN K=3	Q Score RF
1	1	250	3	190	86	87
2	1	1500	3	190	75	77
3	2	500	3	190	83	84
4	2	800	2	255	88	88
5	4	400	2	222	84	83
6	5	300	2	210	81	82
7	3	280	1	235	89	90
8	2	550	1	250	91	91

The PDR achieved with Random forest is marginally higher than that with K-NN algorithm. The control overhead generated is comparable with insignificant variations. The figure 6.16 shows the plots of RPL-ML with KNN (K=2,3,5) versus RPL-ML with Random forest regressor as data rates vary from 1 ppm to 10 ppm.

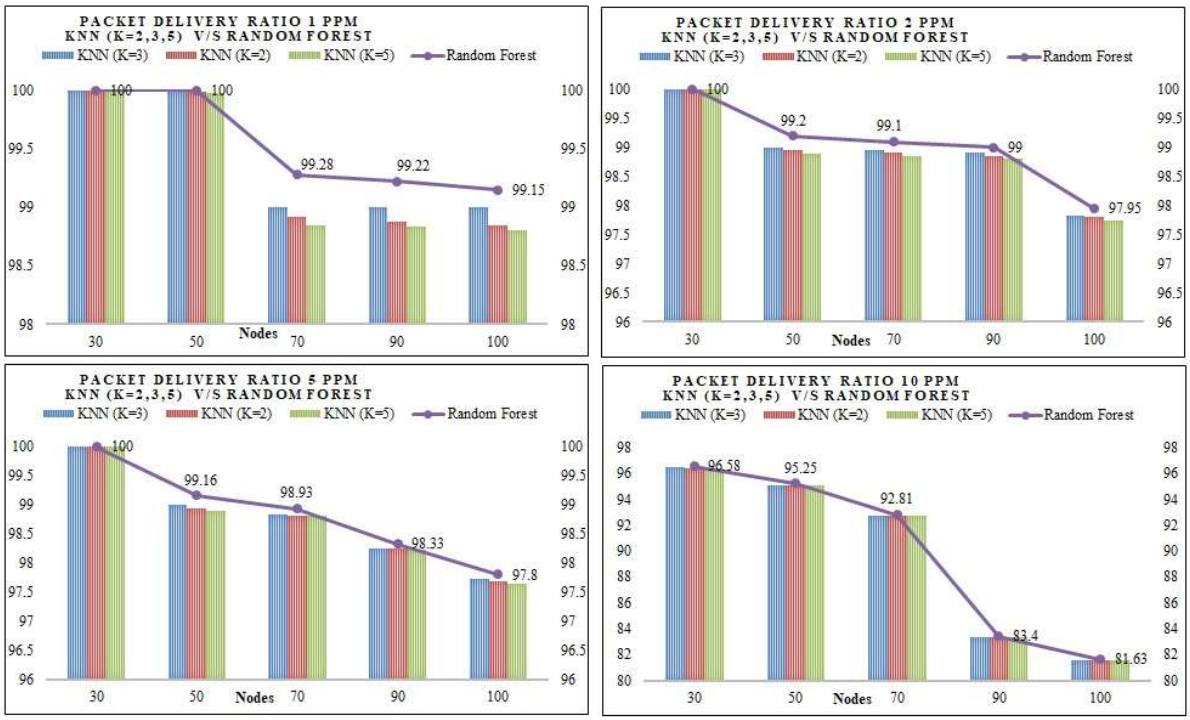


Figure 6.16: Packet Delivery Ratio for RPL-ML with KNN vs Random Forest

The above results indicate that RPL with Random Forest algorithm is a better option to perform the multi constrained optimal path decision over the K-NN (K=3) regressor with regards to power consumption, latency and PDR. Many studies in literature prefer the random forest over the K-NN for classification due to the low error rate of RF [121], [122]. Our findings are consistent with the results of these studies, and show that random forest performs better over K-NN.

6.10 Performance Comparison of RPL-ML with other existing works

In this sub-section we compare RPL-ML to other popular works existing in literature such as E-RPL [84], EEOPS-RPL [85], RPL-GCA [83], HOFESA [112], FL-HELR [81] and MC-RPL[90]. FL-HELR is a fuzzy logic based method and MC-RPL is a VIKOR MADM technique. The comparison is done at varying node densities from 30 to 100 and at packet rate of

1 ppm. Among all the various works available in literature, these works are chosen since the simulation parameters closely matched to those in our work.

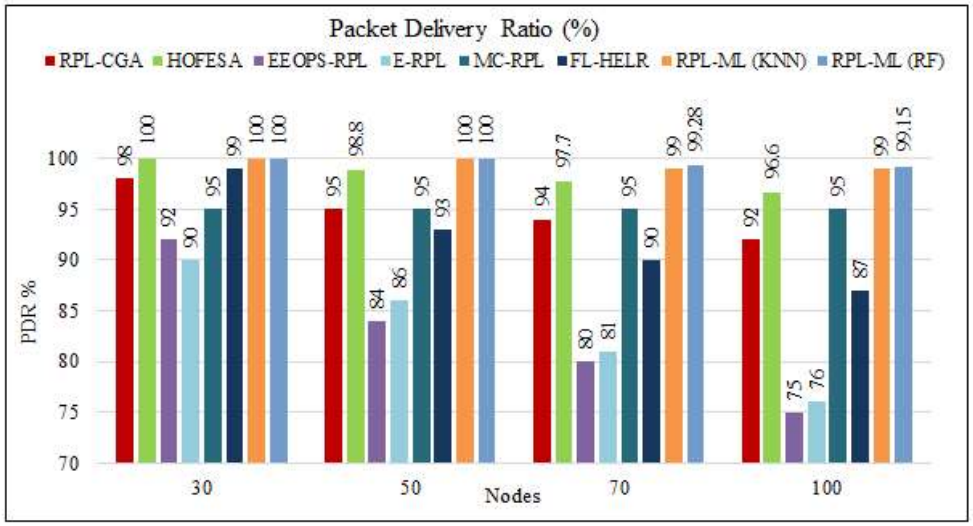


Figure 6.17: PDR with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , HOFESA [112], FL-HELR [81] and MC-RPL[90]

The graph of Packet delivery ratio with the above works is plotted in figure 6.17. The EPC-RPL has the least delivery success rate. RPL-GCA technique has good PDR of 98% at low node density, and reduces to 92% at 100 nodes. The firefly and ant colony techniques yield lower PDR of 92% at density of 30 and it decreases to 76% at 100 nodes. MC-RPL gives 95% PDR . FL-HELR varies from 99 at low density of 30 nodes to 87 at 100 nodes. RPL-ML scores above all these methods by ensuring above 99% PDR at all node densities.

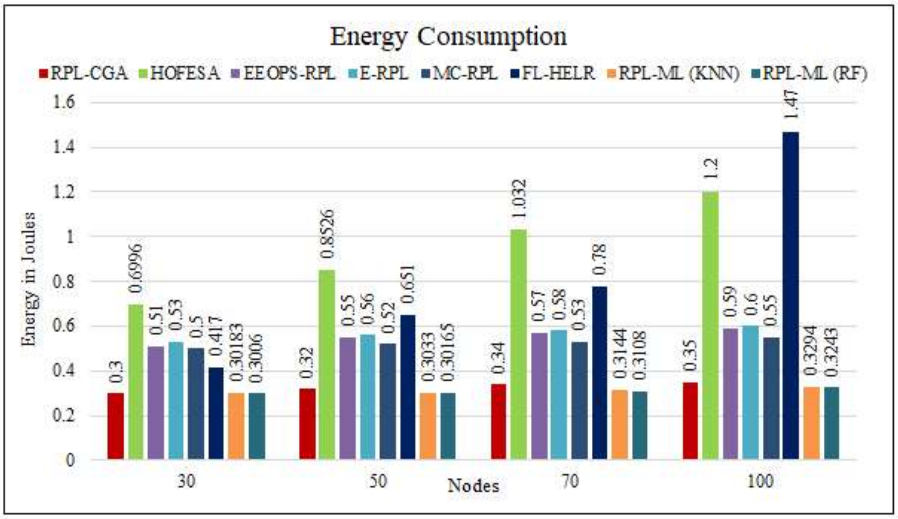


Figure 6.18: Energy Consumption with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , HOFESA [112], FL-HELR [81] and MC-RPL[90]

The energy consumption with the above works is plotted in figure 6.18. Energy consumption with RPL-ML(Random Forest) is on an average 0.30 joules. It is the lowest compared to other

existing works shown in the graph. RPL-CGA method has the comparable energy consumption to RPL-ML(Random Forest). FL-HELR method has the highest energy consumption among all. The EA based technique consume more energy to process the complex algorithms.

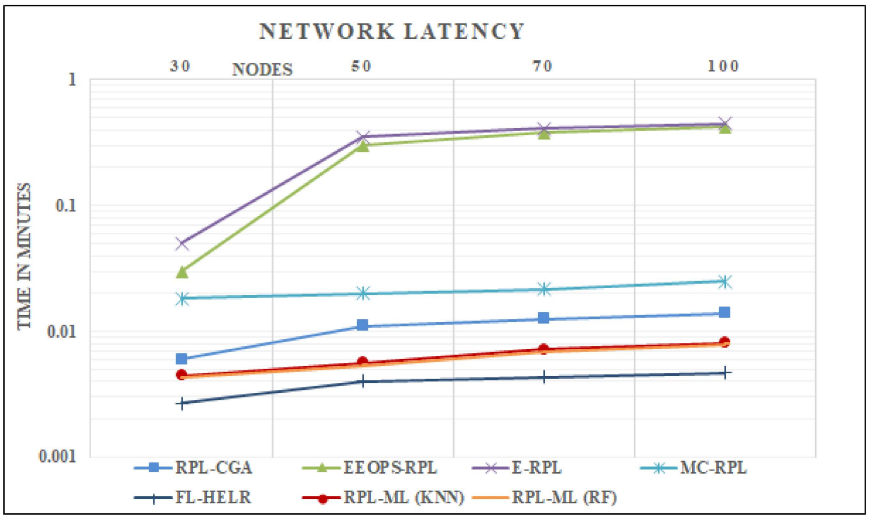


Figure 6.19: Network Latency with RPL-ML(KNN), RPL-ML(RF) versus E-RPL[84], EEOPS-RPL[85], RPL-GCA[83] , FL-HELR [81] and MC-RPL[90]

The plots in figures 6.19 depict the Network Latency in the works under consideration. Latency or End-to-end delay consists of the transmitting time at sender, processing time at sender and forwarding time taken by each intermediate node. RPL-ML (K-NN) method gives delay of 0.38 s, while the RPL-ML (Random Forest) results in a lower latency of 0.36 s. The other EA based methods have a latency of 0.4 mins. This is due to the complexity of the algorithms and time involved in processing at the sensor nodes. FL-HELR has lowest latency of 0.2 s among all. MC-RPL has a latency of 1.28 s.

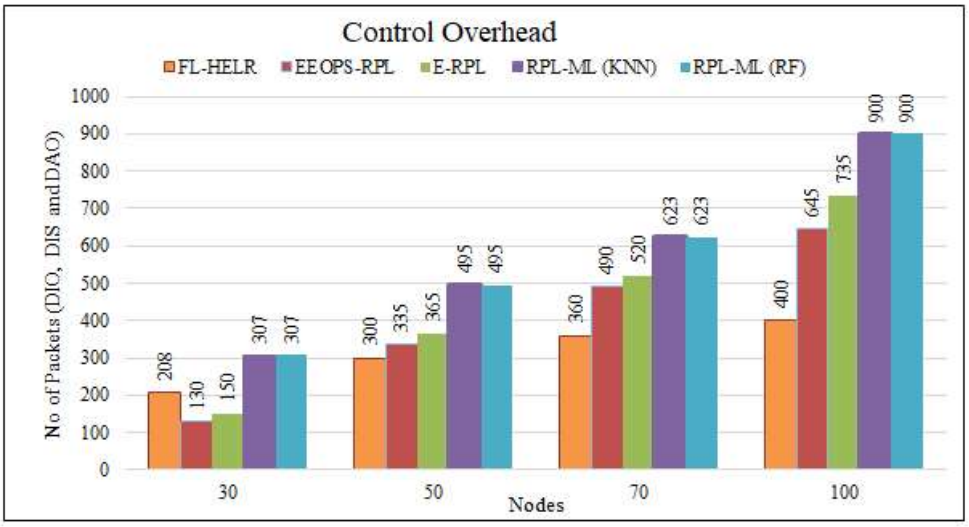


Figure 6.20: Control Overhead with RPL-ML(KNN), RPL-ML(RF) versus E-RPL [84] , EEOPS-RPL [85] and FL-HELR [81]

The control overhead with EPC-RPL, E-RPL and EEOPS-RPL and RPL-ML is depicted in figure 6.20. In addition to the application data, three types of control messages circulate in the network. These messages play a crucial role for initial network construction and routine routing information updation. RPL-ML has marginally higher overhead than E-RPL [84], EEOPS-RPL [85]and FL-HELR [81]. At the start, during the stage of network construction, RPL-ML requires more number of control messages. In a network with 100 nodes and simulation time of 60 mins, 6000 data packets are generated in the network. RPL-ML generates 900 control packets during this duration. Thus the ratio of Data to Control messages is 20:3. Generally, lower overhead gives lower latency in any method. But in the case of EPC-RPL, E-RPL and EEOPS-RPL, the processing time of complex EA overshadows the benefits of reduced control overhead. Thus, it can be concluded that RPL-ML is preferable regards to PDR, energy consumption and delay over the other methods with trade-off slightly higher control overhead.

Table 6.16: Comparison of RPL-ML with other existing works in literature

Name	Technique Used	Metrics	PDR(%)	Energy Cons (J)	Latency	Control Ovhd
RPL-CGA [83]	Genetic Algorithm for optimizing weighting factors in composition metric	Queue length, Delay, RER, HC, ETX	95	0.33	0.65 s	-
EEOPS-RPL [85]	Firefly algorithm for optimization	RER, ETX and distance	82	0.56	0.28 min	400
E-RPL [84]	Ant colony optimization	Children count, RER, ETX, rank value	83	0.57	0.315 min	442
MC-RPL [90]	VIKOR MADM based	Energy, ETX, Delay	95	0.53	1.28 s	-
FL-HELR [81]	Fuzzy logic based	Energy, ETX, Delay, HC	92	0.83	0.2 s	317
HOFESA [112]	Hybrid Empirical stability aware	HC, RSSI, Energy	98	0.95	-	20500
RPL-ML(K-NN)	K-Nearest Neighbor ML based proposed work	ETX, HC, RE and latency	99.5	0.31	0.38 s	581
RPL-ML(RF)	Random Forest ML based proposed work	ETX, HC, RE and latency	99.61	0.30	0.36 s	581

6.11 Conclusion

The impact of applying machine learning regression methods to optimise the routing decision in RPL protocol is explored in this chapter. Three machine learning regression techniques are studied namely Linear Regression, Random Forest and K-Nearest Neighbour. These techniques

are tested using a data-set which contains exhaustive values of the four routing metrics ETX, Delay, HC and RE and the corresponding desired Quality Score. The K-NN and RF have accuracy of 98% and higher than that of LR which is 90%. The K-NN algorithm with K=3 values is integrated into Contiki OS to achieve a improvised version of RPL termed as RPL-ML(KNN). The delay and HC metric ensures that candidates with lower delay and shorter route are chosen as parents hence ensuring faster packet delivery. The latency in RPL-ML(KNN) is lower by 20% as compared to standard RPL. The ETX metric ensures better reliability and hence improves packet delivery. The PDR of RPL-ML(KNN) is higher by 9% over MRHOF and OF0. A reduction in control overhead to the tune of 52% is achieved. Higher PDR and lower control overhead saves the node energy of re-transmission, receiving and forwarding messages. Besides this, Residual Energy metric aids RPL-ML(KNN) in avoiding candidates with lower energy levels from being selected as parents. This preserves the node energy and improves overall efficiency. A 17% reduction in power consumption is achieved with RPL-ML(KNN) as compared to standard RPL at density of 100 nodes. In the later part of the chapter, the evaluation of RPL with Random forest algorithm to perform the multi-criteria decision making is done. Random forest algorithm computes relatively faster than K-NN algorithm thus saving on power and speed. The values of power consumption and latency with Random Forest are lesser by 3% and 6% respectively than that achieved with K-NN. The technique of Linear Regression is not evaluated in Contiki OS due to its low accuracy of 90%. Thus, we conclude that the novel RPL-ML techniques prove to be significantly superior than standard RPL implementations OF0 and MRHOF with the Random Forest algorithm having an edge over the K-nearest neighbor method. RPL-ML with Random Forest achieves 20% lower Power consumption, 26% lower latency and 4% higher PDR than standard RPL.

Chapter 7

Performance Comparison of the proposed RPL implementations

The Internet of things is bound to define the human lifestyle in the coming decade. Billions of devices connected and communicating with each other will utilize large amounts of resources in terms of energy, bandwidth, sensing technology and processing hardware. The data within localized wireless networks will be routed by sensor nodes to the main gateway using the RPL protocol. Efficient routing which saves energy and time is crucial to sustaining the network. To improvise the routing process in RPL, this research proposed three routing approaches using Fuzzy Logic, Artificial Neural Networks and Machine Learning. Four routing metrics -ETX, Delay, HC and RE were chosen and these soft computing methods were applied to decide the best route. Initially, fuzzy logic based RPL-FZ was proposed which relied on three metrics to decide the best route. Later, the fourth metric was added and OF-FZ was designed. The network performance of OF-FZ excelled that of RPL-FZ . Thus, the four metrics used in OF-FZ were finalised as the decision criteria. Subsequently, it was decided to explore the domain of artificial neural networks and machine learning and analyse their suitability to perform MCDM in RPL networks. Towards this direction, three novel version, ANN-RPL, RPL-ML(K-NN) and RPL-ML(RF) were proposed. ANN-RPL used feed forward neural network and the other two were based on K-nearest neighbor and random forest regression methods of machine learning. RPL-ML (RF) scored better than RPL-ML(KNN). In the previous chapters, we have compared the performance of OF-FZ, ANN-RPL and RPL-ML(RF) with the standard RPL objective functions OF0 and MRHOF. The simulation results indicated that the objective functions proposed in this research have an significant edge over the standard objective functions. The table 7.1 shows the improvements attained with the proposed works over standard OF0 and MRHOF.

Table 7.1: Improvement with proposed works over standard RPL

	Metrics	Technique	Improvement over OF0 & MRHOF							
			Pavg		**PDR**		**Latency**		**Control Ovhd**	
			OF0	MRHOF	OF0	MRHOF	OF0	MRHOF	OF0	MRHOF
RPL-FZ	ETX, Delay, RE	Fuzzy	6%	9%	3%	4%	6%	9%	28%	42%
OF-FZ	ETX, Delay, RE, HC	Fuzzy	7%	9%	5%	6%	11%	13%	28%	42%
ANN-RPL	ETX, Delay, RE, HC	Neural Networks	9%	11%	7%	8%	16%	18%	34%	47%
RPL-ML(KNN)	ETX, Delay, RE, HC	K-NN	12%	14%	8%	9%	18%	20%	34%	47%
RPL-ML(RF)	ETX, Delay, RE, HC	Random Forest	14%	16%	8%	9%	23%	25%	34%	47%

This chapter aims to present the performance comparative analysis of the three approaches OF-FZ, ANN-RPL and RPL-ML(RF) with regard to Packet delivery ratio, power consumption, latency and control overhead.

7.1 Performance comparison with regard to Average Power consumption

In this section, we compare the three novel objective functions with each other. Primarily, we compare the power consumption with the three techniques. The average power consumption with OF-FZ, ANN-RPL and RPL-ML(RF) is plotted in figure 7.1 for varying data rate from 1 ppm to 10 ppm. RPL-ML (RF) performs very well at all node densities, with max advantage noted at 50 and 70 nodes.

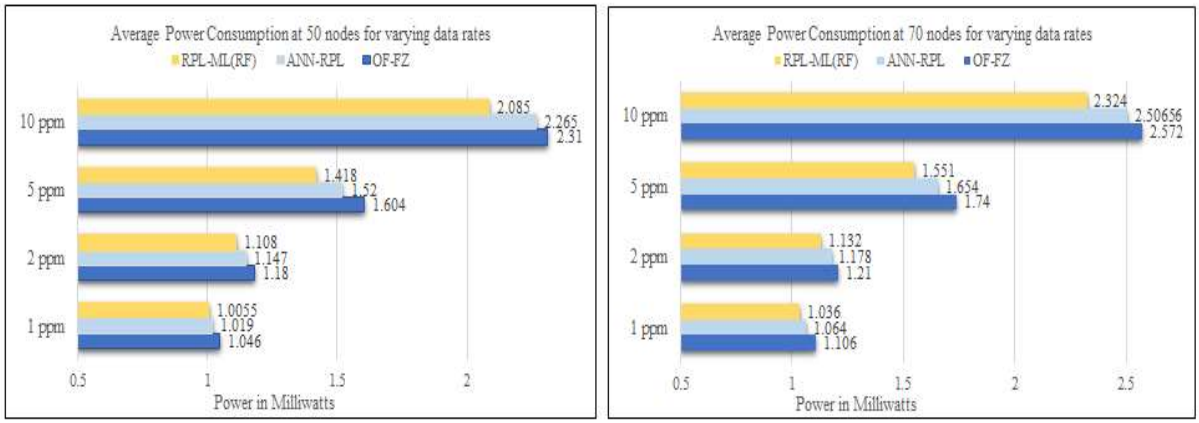
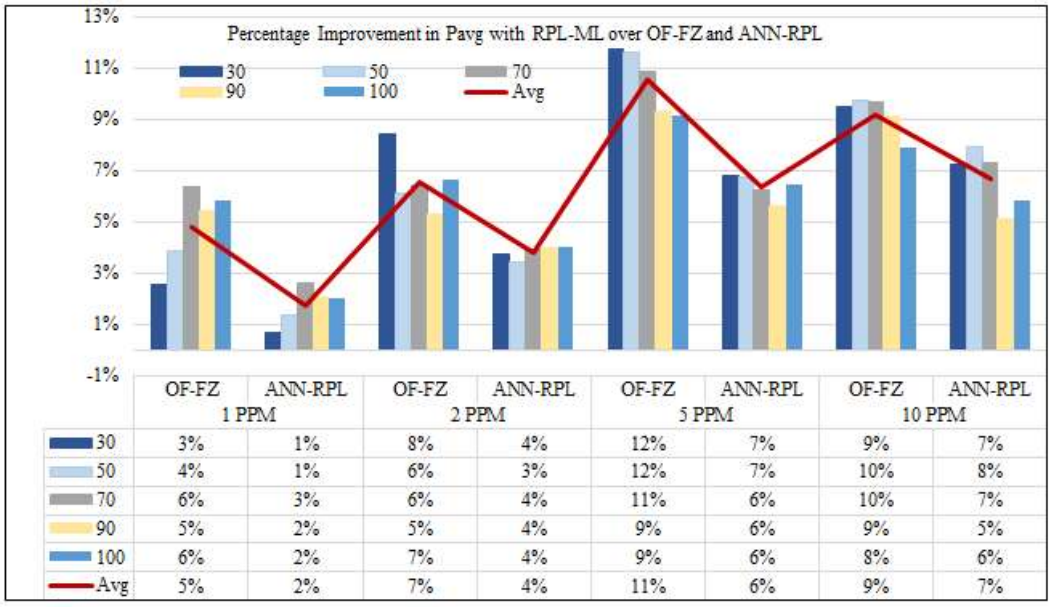


Figure 7.1: Average Power Consumption with OF-FZ, ANN-RPL and RPL-ML(RF)

The fuzzy logic based objective function OF-FZ, gives an improvement of 15% over MRHOF at 5ppm and 100 node density. On an average it gives 7% and 9% improvement over OF0 and

MRHOF. The Artificial Neural Network based ANN-RPL, further improves upon OF-FZ. The feed forward neural network gives higher accuracy in computation of Quality score for parent node than fuzzy logic. The fuzzy inference system in OF-FZ is designed to give a particular output for a range of values, as specified in the membership function. For example all nodes with residual energy between 102 and 153, will result in the same Quality or QA score. As a result it is unable to give distinct Quality or QA score output values for distinct inputs which lie in the specified range. This shortcoming is not there in the machine learning and ANN techniques. They are capable of computing a distinct Quality/QA score for the set of inputs lying in specified range. ANN-RPL gives on average a 3% improvement in power consumption over the OF-FZ approach. RPL-ML with the Random Forest algorithm is observed to give the best results among the three. RPL-ML(RF) attains 9% and 6% improvement over the OF-FZ and ANN-RPL methods at higher packet rates of 5 ppm and 10 ppm.



	OF-FZ (1 PPM)	ANN-RPL (1 PPM)	OF-FZ (2 PPM)	ANN-RPL (2 PPM)	OF-FZ (5 PPM)	ANN-RPL (5 PPM)	OF-FZ (10 PPM)	ANN-RPL (10 PPM)
30	3%	1%	8%	4%	12%	7%	9%	7%
50	4%	1%	6%	3%	12%	7%	10%	8%
70	6%	3%	6%	4%	11%	6%	10%	7%
90	5%	2%	5%	4%	9%	6%	9%	5%
100	6%	2%	7%	4%	9%	6%	8%	6%
Avg	5%	2%	7%	4%	11%	6%	9%	7%

Figure 7.2: Percentage Improvement with RPL-ML(RF) over OF-FZ and ANN-RPL

RPL-ML (RF) RPL-ML scores above the ANN-RPL method in terms of power consumption owing to the basic difference in computation technique in both. Both the Random Forest and Neural Networks are different techniques that learn differently but can be used in similar domains. The machine learning algorithm of random forest is pre-trained and then deployed in the network. It builds decision trees which basically involves true/false logical decisions and averaging. The CPU can execute this program effortlessly. Whereas the ANN method involves a number of matrix multiplication operations between the weights and inputs. Matrix Multiplication requires relatively higher processing effort than logical operations. Random Forest is less computationally expensive and does not require a GPU to finish training. The percentage

improvements with RPL-ML over the other two are depicted in the graph of figure 7.2.
To further understand the reason of improvement in case of RPL-ML(Random Forest) over OF-FZ, we decided to trace the route constructed from two nodes to sink with RPL-ML and OF-FZ. The paths traced for two nodes N1 and N9 towards the sink node with RPL-ML and OF-FZ is shown in figure 7.3.

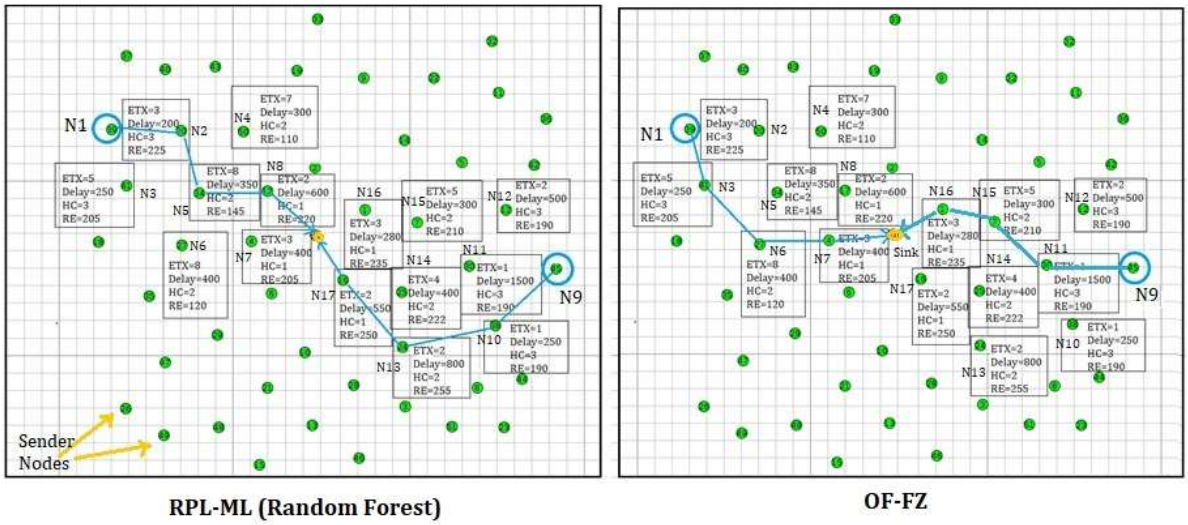


Figure 7.3: Paths traced for two nodes N1 and N9 towards the sink node with RPL-ML(RF) and OF-FZ

Table 7.2: QA scores computed by Fuzzy logic and Random Forest regressor for candidate parent nodes in path from N1 to sink

Node ID	ETX	Delay	HC	RE	QA score- FUZZY	QA score -Random Forest
N2	3	200	3	225	77	89
N3	5	250	3	205	77	82
N4	7	300	2	110	63	65
N5	8	350	2	145	63	68
N6	8	400	2	120	63	62
N7	3	400	1	205	92	84
N8	2	600	1	220	92	86

The tables 7.2 and 7.3 depict the QA scores computed by Random Forest regression and Fuzzy Inference System for various candidate parent nodes along the route from two nodes N1 and N9 toward sink. In case of N1, nodes N2 to N8 are present along the route towards sink. For N9, the nodes N10 to N17 are candidate parents. In case of N2 and N3, the FIS computes a score of 77 for both, while Random forest distinctly computes scores as 88 and 86 ans selects N2 as parent. It is clearly evident that N2 is a better choice due to lower ETX value, lower delay and higher residual energy. OF-FZ fails to distinguish between these two and is likely

to choose N3 as parent leading to higher power consumption owing to higher delay and more number of re-transmissions. Similarly with N4, N5 and N6, the FIS score is 63 for all three. With random forest, node N5 gets a higher score due to its higher residual energy. Preference is given to sustaining node energy, hence node with higher level is selected. OF-FZ on the other hand selects N6 in this case scenario, which has lower energy.

Table 7.3: QA scores computed by Fuzzy logic and Random Forest regressor for candidate parent nodes in path from N9 to sink

Node ID	ETX	Delay	HC	RE	QA score- FUZZY	QA score -Random Forest
N10	1	250	3	190	73	87
N11	1	1500	3	190	73	77
N12	2	500	3	190	73	84
N13	2	800	2	255	77	88
N14	4	400	2	222	77	83
N15	5	300	2	210	77	82
N16	3	280	1	235	92	90
N17	2	550	1	250	92	91

While traversing from N9 to sink, nodes N10, N11 and N12 get assigned same score of 73 by FIS. RF computes 86 score for N10 and selects it as parent. OF-FZ makes a poor choice of N11, which has a high delay of 1500 ms. The total path delay with OF-FZ from N9 to sink amounts to 2080 ms, while that with RF is 1600 ms. This implies that random forest is better suited to identify parent nodes as compared to OF-FZ. ANN-RPL takes same route as RPL-ML(RF) and is an eminent approach, provided the extra CPU processing effort can be condoned.

7.2 Performance comparison with regard to Packet delivery ratio

The packet delivery ratio is a measure of the successful delivery of packets to the destination node. The PDR reduces as more and more nodes are added to the network. Another factor influencing the PDR is the packet transmission rate. At rates above 5 ppm, the PDR is seen to drop considerably in standard RPL. Our work focuses on improvising the RPL protocol and its impact on improvement in PDR can be appreciated when the network is operating at high data transmission rate of 10 ppm. At packet rates of 1 ppm, 2 ppm and 5 ppm, the three perform alike with regards to packet delivery with marginal deviation. The fuzzy logic based OF-FZ gives 6% higher PDR over the standard OF0 and MRHOF in a 100 node network operating at 10 ppm data rate, ANN-RPL achieves 8% better and RPL-ML(Random Forest) excels by 9%.

The figure 7.4 shows the PDR with all three novel objective functions.

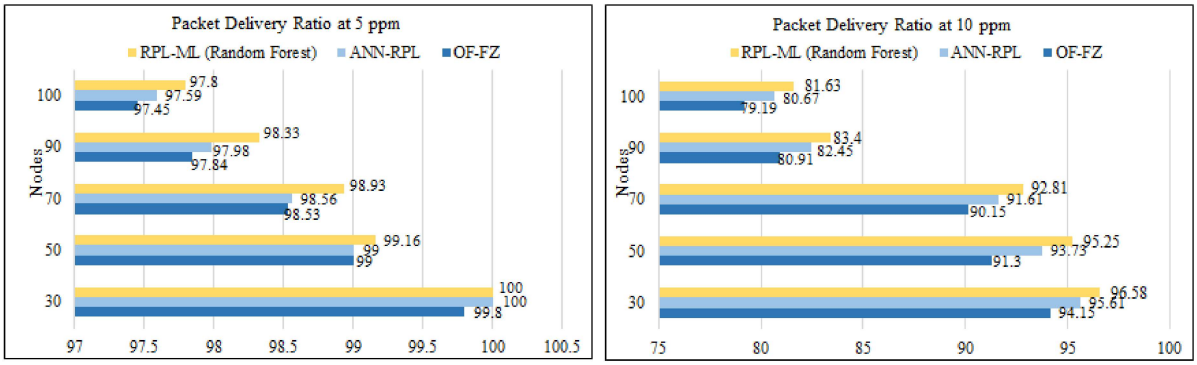


Figure 7.4: Packet delivery ratio with OF-FZ, ANN-RPL and RPL-ML(RF) at 5 ppm and 10 ppm

We observe that PDR with RPL-ML (Random Forest) exceeds ANN-RPL by 1% and OF-FZ by 3%. RPL-ML(Random Forest) is found to be very efficient at multi-constrained optimization by selecting the most optimal route to sink. ANN-RPL is also a comparably efficient technique for optimal routing. Both these techniques have an edge over the fuzzy logic approach. As seen in the paths diagram 7.3, OF-FZ tends to take lesser efficient routes than random forest. The lower PDR can be attributed to the packet loss occurring on non optimal routes take by OF-FZ.

7.3 Performance comparison with regard to Network Latency and Control Overhead

The graphs in figure 7.5 depict the latency at 50 and 70 nodes. The percentage improvements with RPL-ML(RF) are plotted in figure 7.6.

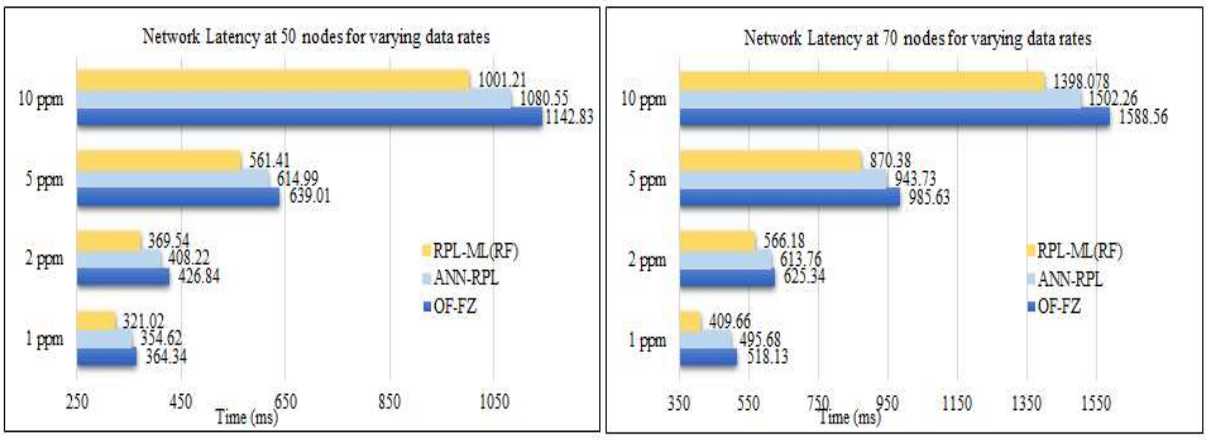
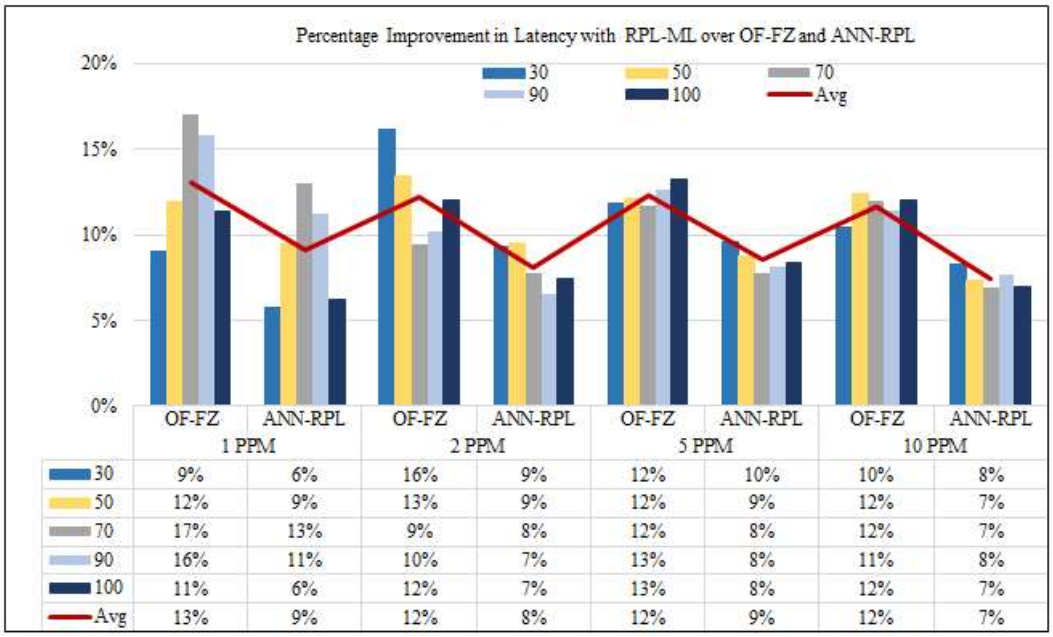


Figure 7.5: Latency with OF-FZ, ANN-RPL and RPL-ML(RF) for 50 and 70 nodes at varying data rates



	OF-FZ	ANN-RPL	OF-FZ	ANN-RPL	OF-FZ	ANN-RPL	OF-FZ	ANN-RPL
	1 PPM		2 PPM		5 PPM		10 PPM	
30	9%	6%	16%	9%	12%	10%	10%	8%
50	12%	9%	13%	9%	12%	9%	12%	7%
70	17%	13%	9%	8%	12%	8%	12%	7%
90	16%	11%	10%	7%	13%	8%	11%	8%
100	11%	6%	12%	7%	13%	8%	12%	7%
Avg	13%	9%	12%	8%	12%	9%	12%	7%

Figure 7.6: Percentage Improvement in Latency with and RPL-ML(RF)

The Random Forest regression can distinguish optimal routes with higher accuracy than the Fuzzy based OF-FZ. It selects routes with the lowest delay metric and results in least latency among the three. As compared to ANN method, random forest is preferable as the computation effort is lower. Faster processing ensures speedy delivery of packets to destination, thus reducing the network latency. RPL-ML (RF) achieves on an average 12% lower latency than OF-FZ. As compared to ANN-RPL it has a 8% lower latency. All the three techniques adopt the same mechanism for generating the control advertisement messages. Thus, control overhead generated is comparable with insignificant variations.

Impact of data transmission rate on the network performance

The network performs optimally at data rate of 5 ppm. The packets are delivered with success rate of 97%. If the data transmission rate is doubled to 10ppm, the packet delivery ratio drops by 25%. The power consumption is seen to rise by 30% and latency increases by 28%. Thus, we conclude that data rates of 1ppm to 5 ppm are recommended for optimal network performance. A network with 30 to 100 nodes is ideal with single sink node. Additional sink nodes need to be deployed in networks of sizes more than 100 nodes.

7.4 Conclusion

The performance of RPL-ML(RF), ANN-RPL and OF-FZ is compared in this section. RPL-ML(RF) is the best performer among peers. It has the lowest power consumption, 9% and 6%

lower than OF-FZ and ANN-RPL at 100 node density and 5 ppm data rate. The network latency with RPL-ML(RF) is observed to be 13% and 8% lower than OF-FZ and ANN-RPL at same configuration. The PDR is 97.8% which is marginally higher than peers. All three techniques adopt the same mechanism for generating the control advertisement messages. Thus, control overhead generated is comparable with insignificant variations. In general, the network performance is prudent and efficient for data transmission rates upto 5 ppm. Beyond that performance degrades drastically. Packet loss rises and latency rises dramatically.

Chapter 8

Conclusion

8.1 General Conclusion

This research focuses on developing a holistic routing solution in RPL based networks. Routes are evaluated based on multiple link and node metrics like ETX, Hop Count, Residual Energy and Delay. This is a classic Multi Attribute Decision Making problem, however the conventional MADM techniques cannot be applied here due to hardware constrains at nodes. Artificial Intelligence and Soft Computing have the potential to solve complex real world problems without being computation extensive. Therefore these methods are explored in this research in the context of improving the routing performance of RPL protocol.

Generally in a network, a node has several neighbors and it has to choose the best one out of these as parent. This implies that a Quality score has to be computed for every neighbor on the basis of which selection can be made.

Fuzzy logic is a widely used technique for decades in several application domains. In the initial part of our work, we have designed a Fuzzy Inference System which takes the four metrics as inputs and generates the Quality of neighbors as output. Linguistic variables, membership functions for the four metrics are defined and a rule base is created. Upon defuzzification, the crisp Quality Score is obtained. The MATLAB software is used to design FIS. There are several operating systems which are available for IoT networks. We have used the Contiki OS 2.7 in our entire work. The C program for the designed FIS is incorporated into Contiki OS and simulations are run in the COOJA simulator by varying node density, packet rates and number of sink nodes. The effect on network performance parameters like Power Consumption, Latency, Control Traffic Overhead and Packet Delivery Ratio is studied.

Later, we have explored Artificial Neural Networks and its application in computing the

Quality of neighbours by taking the multi-metrics as input. A Multi-Layer Feed forward Artificial Neural Network is designed using Jupyter Notebook and Python. A data-set with 85,686 sample values is used in training and testing this ANN. The python codes for ANN are converted to C language, since Contiki OS is in C. Visual Studio is used to write the C programs and debug the ANN. The simulations are run and impact of using ANN in RPL is evaluated by analysing the network performance parameters.

Lastly, we decided to apply Machine Learning Techniques to decide the best parent node. We have to use Regression techniques since a relation has to be obtained between the output and the multiple input metrics. Regression techniques of Linear regression, Random Forest and K-Nearest Neighbours (K-NN) are chosen and compared. The technique of K-NN and Random Forest resulted in higher Accuracy. So, K-NN and Random Forest algorithms were chosen and incorporated into Contiki OS to make the routing decisions. Simulations in COOJA were run and performance of K-NN and random forest based RPL was analysed. The detailed analysis and trade-off comparison of the above three techniques in improving the RPL protocol was done.

This work proposed three RPL implementations- OF-FZ, ANN-RPL and RPL-ML which applied Fuzzy Logic, neural networks and machine learning regression techniques to perform the multi criteria decision making. As part of the research work we come to the following conclusions:

1. When the network is strained at packet rates of 10p/min, single sink node and 100 sender nodes, RPL network with proposed objective function OF-FZ consumes 7% and 9% lesser power as compared to OF0 and MRHOF respectively. OF-FZ gives 5%- 6% higher Packet Delivery Ratio than OF0 and MRHOF. OF- FZ has 13% lower latency as compared to MRHOF under strained conditions. Another significant contribution with OF-FZ is the reduction in the DAO messages. A reduction in control overhead to the tune of 42% is achieved. The addition of Hop Count metric to ETX, RE and delay in case of OF-FZ gives improvement in terms of Power Consumption (5% at 2 ppm and 5ppm) and Latency (2% at 2ppm and 5ppm and 5% at 10 ppm) over RPL-FZ, but is seen to have marginal effect on the PDR. The computation cost of adding HC metric is negligible.

2. ETX and residual energy metrics play a crucial role in governing the PDR and power consumption. Combining ETX , residual energy, delay and hop count metrics as implemented in OF-FZ gives optimal results compared to other four metric combinations.

3. Every additional metric adds 14 bytes to the DIO message. In case of proposed objective

functions additional 42 bytes get appended to DIO message. A maximum of four metrics can be used since maximum packet size is 128 bytes. The added overhead due to four metrics can be compensated by reducing the advertisement messages. Reducing the control messages helps to reduce the latency, but curtailing them beyond a certain point may adversely affect the network performance.

4. In a network with 100 nodes, operating at 10 ppm, ANN-RPL reduces the power consumption to the tune of 11% over standard RPL. The PDR of ANN-RPL is higher by 7% at this configuration.The latency in ANN-RPL is lower by 18% as compared to MRHOF. This novel ANN-RPL technique proves to be substantially better than standard RPL. ANN-RPL scores above OF-FZ in terms of power and latency by 2% and 5% respectively. ANN-RPL is compared to evolutionary algorithm based RPL implementations. The EA schemes require more computing time due to the large number of iterations involved in the algorithms. On an average, the delay is seen to be around 0.4 mins. Also, they need more powerful microcontrollers with larger memory space. This considerably increases the deployment cost. These factors give ANN-RPL a leverage over the EA-based schemes.

5. The latency in RPL-ML(KNN) is lower by 20% and a 14% reduction in power consumption is achieved with RPL-ML(KNN) as compared to standard MRHOF at density of 100 nodes. The PDR of RPL-ML(KNN) is higher by 9% over MRHOF and OF0 at 10 ppm and 100 nodes. Random forest algorithm computes relatively faster than K-NN algorithm thus saving on power and speed. RPL-ML with Random Forest achieves 16% lower Power consumption, 25% lower latency and 9% higher PDR than standard RPL in a 100 nodes network. The values of power consumption and latency with Random Forest are lesser by 2% and 5% respectively than that achieved with K-NN. The RPL-ML techniques prove to be significantly superior than standard RPL implementations OF0 and MRHOF with the Random Forest algorithm having an edge over the K-nearest neighbor method.

6. RPL-ML(RF) has 8% and 6% lower power consumption than OF-FZ and ANN-RPL at 100 node density and 10 ppm data rate. The network latency with RPL-ML(RF) is observed to be 12% and 7% lower than OF-FZ and ANN-RPL at same configuration. The PDR is 97.8% which is marginally higher than peers. RPL-ML(RF) has an edge over the other two approaches.

7. RPL-ML(RF) having lowest latency is best suited for time crucial applications. As its power demands are on lower side, the network lifetime will be prolonged.

8. In general, the network performance is prudent and efficient for data transmission rates up to 5 ppm. Beyond that performance degrades drastically. Packet loss rises and latency rises dramatically. A network with 30 to 100 nodes is ideal with single sink node. Additional sink nodes need to be deployed in networks of sizes more than 100 nodes.

8.2 Research Contributions

The growing need for an efficient routing protocol for smart IoT objects and networks motivated us to carry on this research and we have successfully achieved the following:

- An improvised objective function for RPL protocol, termed as OF-FZ is proposed which improves the network power consumption by 9%, packet delivery ratio by 6%, latency by 13% and control overhead to the tune of 42% as compared to standard RPL. Routing decisions are based on four metrics- ETX, Hop Count, Delay and Residual Energy and fuzzy logic technique is applied to do the multi-criteria decision making.

- An Artificial Neural Network based Objective Function, ANN-RPL, is designed for RPL protocol to select the best route using four routing metrics thereby improving the power consumption by 11%, packet delivery ratio by 8%, latency by 18% and control overhead to tune of 47% as compared to standard RPL.

- Machine Learning based Objective Function RPL-ML(RF) which employs the random forest regression method and RPL-ML(KNN) which uses the K-Nearest Neighbor algorithm are proposed to perform multi-constrained optimal routing. The improvement achieved with RPL-ML(RF) for power consumption is 16%, packet delivery ratio is 9%, latency is 25% and control overhead to the tune of 47% as compared to standard RPL. A detailed Analysis and trade-off comparison of OF-FZ, ANN-RPL and RPL-ML implementations is done by comparing performance with other related works in literature in terms of Power Consumption, Packet Delivery Ratio, Latency and Control Traffic Overhead for varying node density and packet transmission rates. The objective function RPL-ML(RF), which utilizes the Random Forest regression technique, gives superior performance as compared to the Fuzzy, Artificial Neural Networks and K-nearest neighbor based approaches.

8.3 Challenges

During the research work the following challenges were faced;

- The Contiki operating system supports standard RPL. The standard RPL generates DIO messages with single metric. To incorporate our proposed work into Contiki OS , the length of the DIO message had to be increased to accommodate additional metrics.
- Creating an exhaustive data-set to train the Neural network and machine learning algorithms was challenging.

8.4 Future Work

The future research directions in this area of work are listed as follows :

- The Proposed Objective Functions can be implemented and tested on Real life Test Motes. The performance in simulation and real-life test beds can be analysed.
- The Proposed techniques can be assessed considering the nodes to be Mobile.
- The proposed techniques can be deployed and tested in Heterogeneous Networks.
- Alternate MCDM techniques like AHP, TOPSIS can be applied and evaluated in network scenarios.

PAPERS PUBLISHED

Table 8.1: Research Objectives to Publications Mapping

Sr No	Research Objective	Mapping with publications		
		Journal	Conference	Book Chapter
1	Apply the Fuzzy Logic technique to design a routing strategy for RPL that selects the best route based on three (ETX, residual energy and delay) and four routing metrics (ETX, residual energy, delay and hop count)	1, 2	3,4	5
2	Design a routing strategy for RPL that applies a Multilayer Feed Forward Artificial Neural Network.		6	5
3	Apply the Random Forest and K-Nearest Neighbour regression techniques of Machine Learning to take the best route decision by evaluating the four metrics. Analyze the performance of the three approaches.	7		

1. Sonia Kuwelkar, H. G. Virani,"OF-FZ: An Optimized Objective Function for the IPv6 Routing Protocol for LLNs", IETE Journal of Research, pp 1-19, Taylor & Francis, Oct 2021. https://doi.org/10.1080/03772063.2021.1990139. SCI, SCOPUS.

2. Sonia Kuwelkar, H. G. Virani, "Design of an Efficient RPL Objective Function for Internet of Things Applications", International Journal of Advanced Computer Science and Applications (IJACSA), 12(6), July 2021. http://dx.doi.org/10.14569/IJACSA.2021.0120625. E-SCI, SCOPUS.

3. S. Kuwelkar and H. G. Virani, "RPL-FZ: An optimized objective function for the RPL routing protocol", 6th International Conference on Microelectronics, Computing & Communication Systems (MCCS-2021), July 2021.

4. S. Kuwelkar and H. G. Virani, "Implementation of Fuzzy Optimized IPv6 Routing Protocol for Internet of Things Applications", 2021 IEEE Mumbai Section, Asian Conference on Innovation in Technology (ASIANCON) Pune, India. Aug 28-29, 2021.

5. Sonia Kuwelkar and H. G. Virani. "Enhancing the RPL protocol using ANN for sustainable IoT infrastructure". Book Chapter, Book- Transitioning to Sustainable Industry, Innovation and Infrastructure, MDPI, July 2023. https://doi.org/10.3390/books978-3-03897-869-5-5.

6. Sonia Kuwelkar and H. G. Virani. RPL protocol enhancement using Artificial Neural Network (ANN) for IoT applications. In IEEE International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT 2023), pages 51– 57, Jan 2023.

7. Sonia Kuwelkar and H. G. Virani, "RPL protocol enhancement using K-Nearest Neighbor machine learning approach for IoT application", SUBMITTED to Wireless Personal

Communications, Springer, SCI, SCOPUS
Research Square DOI: https://doi.org/10.21203/rs.3.rs-2392682/v1. (Under Review),

8. S. Kuwelkar and H. G. Virani, ”Performance Analysis of the Objective Functions in IPv6 Routing Protocol for LLN’s used in IOT Applications,” 2019 IEEE Pune Section International Conference (PuneCon), 2019, pp. 1-5, doi: 10.1109/PuneCon46936.2019.9105896.

9. S. Kuwelkar and H. G. Virani, ”Routing Protocols for Automatic Meter Reading systems using Wireless Networks: A Review,” IEEE 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) 2018, pp. 179-183, doi: 10.1109/I-SMAC.2018.8653680.

Bibliography

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[4] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, and R. Alexander, "Rpl: Ipv6 routing protocol for low-power and lossy networks," tech. rep., 2012.

[5] N. Ben Ottman, R. Liscano, and S. S. Heydari, "An analysis of the collection tree protocol (ctp) in mobile sensing environments," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–7, 2017.

[6] E. Ancillotti, R. Bruno, and M. Conti, "Reliable data delivery with the IETF routing protocol for low-power and lossy networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 3, pp. 1864–1877, 2014.

[7] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[8] C. S. Raghavendra, K. M. Sivalingam, and T. Znati, *Wireless sensor networks*. Springer, 2006.

[9] I. F. Akyildiz and M. C. Vuran, *Wireless sensor networks*. John Wiley & Sons, 2010.

[10] A. Khalifeh, K. A. Darabkh, A. M. Khasawneh, I. Alqaisieh, M. Salameh, A. AlAbdala, S. Alrubaye, A. Alassaf, S. Al-HajAli, R. Al-Wardat, *et al.*, "Wireless sensor networks for smart cities: Network design, implementation and performance evaluation," *Electronics*, vol. 10, no. 2, p. 218, 2021.

[11] J. Ko, A. Terzis, S. Dawson-Haggerty, D. E. Culler, J. W. Hui, and P. Levis, "Connecting low-power and lossy networks to the internet," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 96–101, 2011.

[12] K. A. Darabkh, M. Al-Akhras, J. N. Zomot, and M. Atiquzzaman, "Rpl routing protocol over iot: A comprehensive survey, recent advances, insights, bibliometric analysis, recommendations, and future directions," *Journal of Network and Computer Applications*, vol. 207, p. 103476, 2022.

[13] Z. Yang, S. Ping, H. Sun, and A.-H. Aghvami, "Crb-rpl: A receiver-based routing protocol for communications in cognitive radio enabled smart grid," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 7, pp. 5985–5994, 2016.

[14] T. A. Al-Amiedy, M. Anbar, B. Belaton, A. A. Bahashwan, I. H. Hasbullah, M. A. Aladaileh, and G. A. Mukhaini, "A systematic literature review on attacks defense mechanisms in rpl-based 6lowpan of internet of things," *Internet of Things*, p. 100741, 2023.

[15] H. Mohapatra and A. K. Rath, "Designing of fault-tolerant models for wireless sensor network-assisted smart city applications," in *Intelligent Technologies: Concepts, Applications, and Future Directions, Volume 2*, pp. 25–43, Springer, 2023.

[16] S. A. Abdel Hakeem, A. A. Hady, and H. Kim, "Rpl routing protocol performance in smart grid applications based wireless sensors: Experimental and simulated analysis," *Electronics*, vol. 8, no. 2, p. 186, 2019.

[17] D. Wang, Z. Tao, J. Zhang, and A. A. Abouzeid, "Rpl based routing for advanced metering infrastructure in smart grid," in *2010 IEEE International Conference on Communications Workshops*, pp. 1–6, IEEE, 2010.

[18] P. Chithaluru, A. Singh, M. S. Mahmoud, S. Kumar, J. L. V. Mazón, A. Alkhayyat, and D. Anand, "An enhanced opportunistic rank-based parent node selection for sustainable & smart iot networks," *Sustainable Energy Technologies and Assessments*, vol. 56, p. 103079, 2023.

[19] C. Fathy and H. M. Ali, "A secure iot-based irrigation system for precision agriculture using the expeditious cipher," *Sensors*, vol. 23, no. 4, p. 2091, 2023.

[20] K. S. Sankaran, T.-H. Kim, and P.N.Renjith, "An improved ai based secure m-trust privacy protocol for medical internet of things in smart healthcare system," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[21] R. Natarajan, G. H. Lokesh, F. Flammini, A. Premkumar, V. K. Venkatesan, and S. K. Gupta, "A novel framework on security and energy enhancement based on internet of medical things for healthcare 5.0," *Infrastructures*, vol. 8, no. 2, p. 22, 2023.

[22] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad hoc networks*, vol. 3, no. 3, pp. 325–349, 2005.

[23] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wireless networks*, vol. 8, pp. 169–185, 2002.

[24] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 56–67, 2000.

[25] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM transactions on networking*, vol. 11, no. 1, pp. 2–16, 2003.

[26] C. Schurgers and M. B. Srivastava, "Energy efficient routing in wireless sensor networks," in *2001 MILCOM Proceedings Communications for Network-Centric Operations: Creating the Information Force (Cat. No. 01CH37277)*, vol. 1, pp. 357–361, IEEE, 2001.

[27] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *ACM Sigmod record*, vol. 31, no. 3, pp. 9–18, 2002.

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

[29] N. Sadagopan, B. Krishnamachari, and A. Helmy, "The acquire mechanism for efficient querying in sensor networks," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003.*, pp. 149–155, IEEE, 2003.

[30] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd annual Hawaii international conference on system sciences*, pp. 10–pp, IEEE, 2000.

[31] S. Lindsey and C. S. Raghavendra, "Pegasis: Power-efficient gathering in sensor information systems," in *Proceedings, IEEE aerospace conference*, vol. 3, pp. 3–3, IEEE, 2002.

[32] A. Manjeshwar and D. P. Agrawal, "Teen: Arouting protocol for enhanced efficiency in wireless sensor networks.," in *ipdps*, vol. 1, p. 189, 2001.

[33] A. Manjeshwar and D. P. Agrawal, "Apteen: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Parallel and distributed processing symposium, international*, vol. 3, pp. 0195b–0195b, IEEE Computer Society, 2002.

[34] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE Journal on selected areas in communications*, vol. 17, no. 8, pp. 1333–1344, 1999.

[35] L. Li and J. Y. Halpern, "Minimum-energy mobile wireless networks revisited," in *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No. 01CH37240)*, vol. 1, pp. 278–283, IEEE, 2001.

[36] Y. Yu, R. Govindan, and D. Estrin, "Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks," 2001.

[37] T. Clausen, U. Herberg, and M. Philipp, "A critical evaluation of the ipv6 routing protocol for low power and lossy networks (rpl)," in *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 365–372, IEEE, 2011.

[38] A. Brandt, J. Buron, and G. Porcu, "Home automation routing requirements in low-power and lossy networks," tech. rep., 2010.

[39] J. Martocci, P. De Mil, N. Riou, and W. Vermeylen, "Building automation routing requirements in low-power and lossy networks," tech. rep., 2010.

[40] M. Dohler, T. Watteyne, T. Winter, and D. Barthel, "Routing requirements for urban low-power and lossy networks," tech. rep., 2009.

[41] K. Pister, P. Thubert, S. Dwars, and T. Phinney, "Industrial routing requirements in low-power and lossy networks," tech. rep., 2009.

[42] G. Moritz, F. Golatowski, C. Lerche, and D. Timmermann, "Beyond 6lowpan: Web services in wireless sensor networks," *IEEE transactions on Industrial Informatics*, vol. 9, no. 4, pp. 1795–1805, 2012.

[43] J. Hui and P. Thubert, "Compression format for ipv6 datagrams over ieee 802.15. 4-based networks," tech. rep., 2011.

[44] R. Tabish, A. Ben Mnaouer, F. Touati, and A. M. Ghaleb, "A comparative analysis of ble and 6lowpan for u-healthcare applications," in *2013 7th IEEE GCC Conference and Exhibition (GCC)*, pp. 286–291, 2013.

[45] R. Yadav and V. Kumar, "A systematic review paper on energy-efficient routing protocols in internet of things," *IETE Journal of Research*, pp. 1–23, 2023.

[46] H. A. A. Al-Kashoash and A. H. Kemp, "Comparison of 6lowpan and lpwan for the internet of things," *Australian Journal of Electrical and Electronics Engineering*, vol. 13, no. 4, pp. 268–274, 2016.

[47] C. Perkins, E. Belding-Royer, and S. Das, "Rfc3561: Ad hoc on-demand distance vector (aodv) routing," 2003.

[48] D. B. Johnson, D. A. Maltz, J. Broch, *et al.*, "Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks," *Ad hoc networking*, vol. 5, no. 1, pp. 139–172, 2001.

[49] A. Zrelli, "Hardware, software platforms, operating systems and routing protocols for internet of things applications," *Wireless Personal Communications*, vol. 122, no. 4, pp. 3889–3912, 2022.

[50] N. Accettura, L. A. Grieco, G. Boggia, and P. Camarda, "Performance analysis of the rpl routing protocol," in *2011 IEEE International Conference on Mechatronics*, pp. 767–772, IEEE, 2011.

[51] S. Mohapatra and P. Kanungo, "Performance analysis of aodv, dsr, olsr and dsdv routing protocols using ns2 simulator," *Procedia Engineering*, vol. 30, pp. 69–76, 2012.

[52] S. K. Singh, M. Singh, D. K. Singh, *et al.*, "Routing protocols in wireless sensor networks–a survey," *International Journal of Computer Science & Engineering Survey (IJCSES)*, vol. 1, no. 2, pp. 63–83, 2010.

[53] A. W. Burange and V. M. Deshmukh, "Trust based secured routing system for low power networks," *Journal of Integrated Science and Technology*, vol. 11, no. 1, pp. 431–431, 2023.

[54] H. P. Alahari and S. B. Yalavarthi, "A survey on network routing protocols in internet of things (iot)," *Int. J. Comput. Appl*, vol. 160, no. 2, pp. 0975–8887, 2017.

[55] J. Vasseur, N. Agarwal, J. Hui, Z. Shelby, P. Bertrand, and C. Chauvenet, "Rpl: The ip routing protocol designed for low power and lossy networks," *Internet Protocol for Smart Objects (IPSO) Alliance*, vol. 36, 2011.

[56] P. Thubert, "Objective function zero for the routing protocol for low-power and lossy networks (rpl)," tech. rep., 2012.

[57] O. Gnawali and P. Levis, "The minimum rank with hysteresis objective function," tech. rep., 2012.

[58] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *Proc. of the 1st USENIX/ACM Symp. on Networked Systems Design and Implementation*, vol. 25, pp. 37–52, 2004.

[59] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The trickle algorithm. internet eng," *Task Force Standard RFC6206*, 2011.

[60] H. Lamaazi and N. Benamar, "A comprehensive survey on enhancements and limitations of the rpl protocol: A focus on the objective function," *Ad Hoc Networks*, vol. 96, p. 102001, 2020.

[61] O. Gaddour and A. Koubâa, "(rpl) in a nutshell: A survey," *Computer Networks*, vol. 56, no. 14, pp. 3163–3178, 2012.

[62] H. Lamaazi and N. Benamar, "RPL enhancement using a new objective function based on combined metrics," *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*, pp. 1459–1464, 2017.

[63] P. O. Kamgueu, E. Nataf, T. Djotio, and O. Festor, "Energy-based metric for the routing protocol in low-power and lossy network," *SENSORNETS 2013 - Proceedings of the 2nd International Conference on Sensor Networks*, pp. 145–148, 2013.

[64] A. Kumar and N. Hariharan, "DCRL-RPL: Dual context-based routing and load balancing in RPL for IoT networks," *IET Communications*, vol. 14, no. 12, pp. 1869–1882, 2020.

[65] E. Ancillotti, C. Vallati, R. Bruno, and E. Mingozzi, "A reinforcement learning-based link quality estimation strategy for rpl and its impact on topology management," *Computer Communications*, vol. 112, pp. 1–13, 2017.

[66] A. Mohammadsalehi, B. Safaei, A. M. H. Monazzah, L. Bauer, J. Henkel, and A. Ejlali, "(armor): a reliable and mobility-aware rpl for mobile internet of things infrastructures," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1503–1516, 2021.

[67] S. Sanshi and J. Cd, "Fuzzy optimised routing metric with mobility support for (rpl)," *IET Communications*, vol. 13, no. 9, pp. 1253–1261, 2019.

[68] M. Shirbeigi, B. Safaei, A. Mohammadsalehi, A. M. H. Monazzah, J. Henkel, and A. Ejlali, "A Cluster-Based and Drop-aware Extension of (RPL) to Provide Reliability in IoT Applications," *15th Annual IEEE International Systems Conference, SysCon 2021 - Proceedings*, 2021.

[69] J. Shreyas, H. Singh, J. Bhutani, S. Pandit, N. N. Srinidhi, and D. S. Kumar, "Congestion Aware Algorithm using Fuzzy Logic to Find an Optimal Routing Path for IoT Networks," *Proceedings of 2019 International Conference on Computational Intelligence and Knowledge Economy, ICCIKE 2019*, pp. 141–145, 2019.

[70] B. Ghaleb, *Efficient Routing Primitives for Low-power and Lossy Networks in Internet of Things*. PhD thesis, 2019.

[71] B. Farzaneh, M. Koosha, E. Boochanpour, and E. Alizadeh, "A New Method for Intrusion Detection on RPL Routing Protocol Using Fuzzy Logic," *2020 6th International Conference on Web Research, ICWR 2020*, pp. 245–250, 2020.

[72] Z. A. Almusaylim, A. Alhumam, and N. Jhanjhi, "Proposing a secure rpl based internet of things routing protocol: A review," *Ad Hoc Networks*, vol. 101, p. 102096, 2020.

[73] D. Arshad, M. Asim, N. Tariq, T. Baker, H. Tawfik, and D. Al-Jumeily OBE, "Thc-rpl: A lightweight trust-enabled routing in rpl-based iot networks against sybil attack," *PloS one*, vol. 17, no. 7, p. e0271277, 2022.

[74] P. O. Kamgueu, E. Nataf, T. Djotio, and O. Festor, "Fuzzy-based routing metrics combination for rpl," 2014.

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

[76] O. Gaddour, A. Koubâa, N. Baccour, and M. Abid, "Of-fl: Qos-aware fuzzy logic objective function for the rpl routing protocol," in *2014 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 365–372, 2014.

[77] H. Lamaazi and N. Benamar, "Of-ec: A novel energy consumption aware objective function for rpl based on fuzzy logic.," *Journal of Network and Computer Applications*, vol. 117, pp. 42–58, 2018.

[78] I. Kechiche, I. Bousnina, and A. Samet, "A novel opportunistic fuzzy logic based objective function for the routing protocol for low-power and lossy networks," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pp. 698–703, 2019.

[79] A. Saaidah, O. Almomani, L. Al-Qaisi, and M. K. MADI, "An efficient design of rpl objective function for routing in internet of things using fuzzy logic," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 8, 2019.

[80] A. Pahlavani, "A new fuzzy madm approach and its application to project selection problem," *International journal of computational intelligence systems*, vol. 3, no. 1, pp. 103–114, 2010.

[81] K. A. Darabkh, M. Al-Akhras, A. F. Khalifeh, I. F. Jafar, and F. Jubair, "An innovative rpl objective function for broad range of iot domains utilizing fuzzy logic and multiple metrics," *Expert Systems with Applications*, vol. 205, p. 117593, 2022.

[82] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary computation*, vol. 1, no. 1, pp. 1–23, 1993.

[83] Y. Cao and M. Wu, "A novel rpl algorithm based on chaotic genetic algorithm," *Sensors*, vol. 18, no. 11, p. 3647, 2018.

[84] S. Preeth, R. Dhanalakshmi, R. Kumar, and S. Si, "Efficient parent selection for rpl using aco and coverage based dynamic trickle techniques," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 4377–4391, 2020.

[85] S. Sennan, R. Somula, A. K. Luhach, G. G. Deverajan, W. Alnumay, N. Jhanjhi, U. Ghosh, and P. Sharma, "Energy efficient optimal parent selection based routing protocol for internet of things using firefly optimization algorithm," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 8, p. e4171, 2021.

[86] A. Seyfollahi, M. Moodi, and A. Ghaffari, "Mfo-rpl: A secure rpl-based routing protocol utilizing moth-flame optimizer for the iot applications," *Computer Standards & Interfaces*, vol. 82, p. 103622, 2022.

[87] C. L. D. Santos, A. M. Mezher, J. P. A. León, J. C. Barrera, E. C. Guerra, and J. Meng, "Ml-rpl: Machine learning-based routing protocol for wireless smart grid networks," *IEEE Access*, vol. 11, pp. 57401–57414, 2023.

[88] A. Syarif, M.-e.-A. Brahmia, J.-F. Dollinger, A. Abouaissa, and L. Idoumghar, "Rpl-oc: Extension of rpl protocol for lln networks based on the operator calculus approach," in *Proceedings of Sixth International Congress on Information and Communication Technology: ICICT 2021, London, Volume 1*, pp. 47–58, Springer, 2022.

[89] R. Schott and G. S. Staples, "Generalized zeon algebras: theory and application to multi-constrained path problems," *Advances in Applied Clifford Algebras*, vol. 27, pp. 45–57, 2017.

[90] B. Farzaneh, A. K. Ahmed, and E. Alizadeh, "Mc-rpl: A new routing approach based on multi-criteria rpl for the internet of things," in *2019 9th International Conference on Computer and Knowledge Engineering (ICCKE)*, pp. 420–425, 2019.

[91] B. Safaei, A. M. H. Monazzah, and A. Ejlali, "Elite: An elaborated cross-layer rpl objective function to achieve energy efficiency in internet-of-things devices," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 1169–1182, 2020.

[92] A. Kumar and N. Hariharan, "Dcrl-rpl: Dual context-based routing and load balancing in rpl for iot networks," *IET Communications*, vol. 14, no. 12, pp. 1869–1882, 2020.

[93] S. Taghizadeh, H. Bobarshad, and H. Elbiaze, "Clrpl: context-aware and load balancing rpl for iot networks under heavy and highly dynamic load," *IEEE access*, vol. 6, pp. 23277–23291, 2018.

[94] A. Aijaz, H. Su, and A.-H. Aghvami, "Corpl: A routing protocol for cognitive radio enabled ami networks," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 477–485, 2014.

[95] A. K. Ahmed, B. Farzaneh, E. Boochanpour, E. Alizadeh, and S. Farzaneh, "Tfuzzy-of: a new method for routing protocol for low-power and lossy networks load balancing using multi-criteria decision-making," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 3, pp. 3474–3483, 2023.

[96] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, *et al.*, "Tinyos: An operating system for sensor networks," *Ambient intelligence*, pp. 115–148, 2005.

[97] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *29th annual IEEE international conference on local computer networks*, pp. 455–462, IEEE, 2004.

[98] E. Baccelli, O. Hahm, M. Günes, M. Wählisch, and T. C. Schmidt, "Riot os: Towards an os for the internet of things," in *2013 IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pp. 79–80, IEEE, 2013.

[99] G. Vashisht and R. Vashisht, "A study on the tizen operating system," *International Journal of Computer Trends and Technology*, vol. 12, no. 1, pp. 14–15, 2014.

[100] F. Fainelli, "The openwrt embedded development framework," in *Proceedings of the Free and Open Source Software Developers European Meeting*, p. 106, sn, 2008.

[101] R. Toulson and T. Wilmshurst, *Fast and effective embedded systems design: applying the ARM mbed*. Elsevier, 2012.

[102] L. Gu and J. A. Stankovic, "t-kernel: Providing reliable os support to wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, pp. 1–14, 2006.

[103] Q. Cao, T. Abdelzaher, J. Stankovic, and T. He, "The liteos operating system: Towards unix-like abstractions for wireless sensor networks," in *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)*, pp. 233–244, Ieee, 2008.

[104] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Proceedings. 2006 31st IEEE conference on local computer networks*, pp. 641–648, IEEE, 2006.

[105] N. F. Aziz, S. Sorooshian, and F. Mahmud, "Mcdm-ahp method in decision makings," *ARPN Journal of Engineering and Applied Sciences*, vol. 11, no. 11, pp. 7217–7220, 2016.

[106] M. Yang, S. Nazir, Q. Xu, S. Ali, *et al.*, "Deep learning algorithms and multicriteria decision-making used in big data: a systematic literature review," *Complexity*, vol. 2020, 2020.

[107] L. A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," *Fuzzy sets and systems*, vol. 1, no. 1, pp. 3–28, 1978.

[108] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International journal of man-machine studies*, vol. 7, no. 1, pp. 1–13, 1975.

[109] P. S and v. n.-p. y. DrS.S.Dhenakaran, journal=IJRDO - Journal of Computer Science and Engineering , "Comparison of triangular and trapezoidal fuzzy membership function,"

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

[111] S. Kharche and S. Pawar, "Optimizing network lifetime and qos in 6lowpans using deep neural networks," *Computers & Electrical Engineering*, vol. 87, p. 106775, 2020.

[112] A. E. Hassani, A. Sahel, A. Badri, *et al.*, "A hybrid objective function with empirical stability aware to improve rpl for iot applications," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 3, p. 2350, 2021.

[113] T. M. Mitchell *et al.*, *Machine learning*, vol. 1. McGraw-hill New York, 2007.

[114] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[115] B. Mahesh, "Machine learning algorithms-a review," *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.

[116] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[117] M. Belgiu and L. Drăguţ, "Random forest in remote sensing: A review of applications and future directions," *ISPRS journal of photogrammetry and remote sensing*, vol. 114, pp. 24–31, 2016.

[118] K. Chomboon, P. Chujai, P. Teerarassamee, K. Kerdprasop, and N. Kerdprasop, "An empirical study of distance metrics for k-nearest neighbor algorithm," in *Proceedings of the 3rd international conference on industrial application engineering*, vol. 2, 2015.

[119] P. Thanh Noi and M. Kappas, "Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using sentinel-2 imagery," *Sensors*, vol. 18, no. 1, p. 18, 2017.

[120] S. Shabani, S. Samadianfard, M. T. Sattari, A. Mosavi, S. Shamshirband, T. Kmet, and A. R. Várkonyi-Kóczy, "Modeling pan evaporation using gaussian process regression k-nearest neighbors random forest and support vector machines; comparative analysis," *Atmosphere*, vol. 11, no. 1, p. 66, 2020.

[121] A. Murugan, S. A. H. Nair, and K. S. Kumar, "Detection of skin cancer using svm, random forest and knn classifiers," *Journal of medical systems*, vol. 43, pp. 1–9, 2019.

[122] K. Shah, H. Patel, D. Sanghvi, and M. Shah, "A comparative analysis of logistic regression, random forest and knn models for the text classification," *Augmented Human Research*, vol. 5, pp. 1–16, 2020.

List of Publications

[1] Sonia Kuwelkar and H. G. Virani. Design of an efficient RPL objective function for Internet of Things applications. *International Journal of Advanced Computer Science and Applications*, 12(6):228–235, 2021.

[2] Sonia Kuwelkar and H. G. Virani. OF-FZ: An optimized objective function for the IPv6 routing protocol for LLNs. *IETE Journal of Research, Taylor & Francis*, pages 1–19, 2021.

[3] Sonia Kuwelkar and H. G. Virani. Enhancing the RPL protocol using ANN for sustainable IoT infrastructure. *Book Chapter, Book- Transitioning to Sustainable Industry, Innovation and Infrastructure, MDPI*, 2023.

[4] Sonia Kuwelkar and H. G. Virani. RPL protocol enhancement using K-Nearest Neighbor machine learning approach for IoT application. *Wireless Personal Communication, Springer-SUBMITTED-UNDER REVIEW*, 2023.

[5] Sonia Kuwelkar and Hassanali. G. Virani. Routing protocols for Automatic Meter Reading systems using wireless networks: A Review. In *2018 2nd IEEE International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, pages 179–183, 2018.

[6] Sonia Kuwelkar and Hassanali. G. Virani. Performance analysis of the objective functions in IPv6 routing protocol for LLN's used in IOT applications. In *2019 IEEE Pune Section International Conference (PuneCon)*, pages 1–5, 2019.

[7] Sonia Kuwelkar and Hassanali. G. Virani. Implementation of fuzzy optimized IPv6 routing protocol for Internet of Things applications. In *IEEE 2021 Asian Conference on Innovation in Technology (ASIANCON)*, 2021.

[8] Sonia Kuwelkar and Hassanali. G. Virani. RPL-FZ: An optimized objective function for the RPL routing protocol. In *6th International Conference on Microelectronics, Computing Communication Systems(MCCS-2021)*, 2021.

[9] Sonia Kuwelkar and Hassanali. G. Virani. RPL protocol enhancement using artificial neural network (ANN) for IoT applications. In *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pages 52–58, 2023.



IETE JOURNAL OF RESEARCH
https://doi.org/10.1080/03772063.2021.1990139

Taylor & Francis

Check for updates

OF-FZ: An Optimized Objective Function for the Ipv6 Routing Protocol for LLNs

Sonia Kuwelkar and Hassanali Gulamali Virani

Electronics & Telecommunications Department, Goa College of Engineering, Goa, India

ABSTRACT

Internet of Things (IoT) has seen a lot of traction over the last few years and is expected to play a major role in controlling devices and communication. IoT network involves sensors with constrained memory, limited processing and power capabilities. The process of network formation largely decides the Quality of service in these networks. IPv6 Routing Protocol for LLNs (RPL) is specially designed by the IETF ROLL working group to cater to routing requirements in such networks. In RPL, an Objective function specifies a set of metrics or constraints which can be used as criteria for best parent and path selection, ensuring a faster route to the destination. In our work, we propose a novel objective function OF-FZ, which uses four metrics to make routing decisions. Fuzzy Logic is used to combine the metrics hop count (HC), ETX, delay and node residual energy (RE) to obtain a single decision metric termed as Quality Assurance (QA) score. During routing, a neighbor node having the highest QA score is chosen as the best parent. The proposed Objective Function leads to significant savings in energy consumption (EC) (9% lower), Packet Delivery Ratio (PDR) (3% higher), End-to-end Delay (8% lower) as compared to existing OFs. Controlling advertisement messages circulating in the network leads to a significant reduction in control overhead to the tune of 45%. Performance evaluation and deep dive of OF-FZ is done in three configurations: network scalability, increasing packet transmission rates and multi-sink scenarios using Contiki OS and COOJA network simulator.

KEYWORDS

IoT; LLNs; routing protocol; RPL; objective function; fuzzy logic

1. INTRODUCTION

The Internet of Things (IoT) network is characterized as a Low power and Lossy Network (LLN) owing to the sensor nodes which are battery-operated, have limited memory and processing capability along with reduced radio coverage [1]. The routing process, largely governs the quality of service in such networks, thus making the task of protocol design quite challenging. The traditional protocols used for routing in WSNs, MANETs and ADHOC networks governed by an Objective Function. Nodes choose the best route by optimizing or constraining certain metrics contained in Objective Function. Some metrics listed in the specifications are Energy, HC, Latency, Throughput, Received Signal Strength Indicator (RSSI) and Link quality. The RPL standard has not imposed any restrictions on the choice of metrics. This gives a lot of design flexibility to meet application needs. In the specifications, an objective function zero, OF0 is specified [2]. It depends on a

Figure 8.1: Screen Shot of Published Paper in IETE Journal of Research, Taylor & Francis



Enhancing the RPL Protocol Using an Artificial Neural Network for Sustainable IoT Infrastructure

Sonia Kuwelkar and Hassanali G. Virani

1. Introduction

The IoT is a giant communications network connecting billons of devices, such as smart phones, wearable devices, sensors, actuators, and home appliances, over the Internet. It is predicted that by 2025 there will be more than 27 billion IoT connections. The IoT will revolutionize the quality of human life by manifesting a plethora of applications in healthcare, industrial automation (Pister et al. 2009), weather monitoring, smart urban cities (Dohler et al. 2009), home automation (Brandt et al. 2010), logistics, building automation (Martocci et al. 2010), and smart metering (Atzori et al. 2010). The network here comprises of sensors that are battery-operated, use microcontrollers, such as the MSP430, and have limited storage capacity. In contrast to conventional networks, the traffic in these networks can be point-to-point, point-to-multipoint, or multipoint-to-point. The communication network also has several drawbacks, such as instability, low data rates, high packet loss, limited transmission range, frame size limitation, and dynamically varying network topology. Besides these aspects, the links connecting the devices can use a variety of technologies, such as Bluetooth, low-power WiFi, and wired PLC. Another aspect to be considered is that, in large-scale deployments, networks can scale to thousands of nodes. Due to these innumerable constraints on devices and underlying communication technology, the IoT network is characterized as a low-power and lossy network (LLN) (Clausen et al. 2011). All of these unique requirements make the design of an efficient routing protocol a very challenging task (Hui and Culler 2008). One major achievement by the IETF is the 6LoWPAN specification, which specifies how to carry IPv6 datagrams over LLNs (Hui et al. 2012). To devise a

Figure 8.2: Screen Shot of Published Book Chapter in MDPI

Appendices

Appendix A

Appendix I

Weights assigned to Neurons at the three Layers in ANN-RPL

layer1 =

- 0.07435652,0.51822096,0.077307366,-0.6414657,-0.4316001,
  0.6259582,-1.0040765,-0.18634301,-0.7697773,0.06780366,
- 0.00056544034,0.00027489805,-0.43141633,0.10949936,-0.00022519623,
  0.08901061,0.17464672,-0.00055760844,0.0073442967,0.00998921,
- 1.3787638,0.49989197,-1.1029677,0.447331,0.54847676,
  -0.09533733,0.5761163,-1.090431,-1.1444011,0.6243079,
- 0.012745311,0.012527541,0.18448682,0.19263482,-0.024272503,
  -0.5123726,-0.5986534,0.0058499156,-0.2843482,-0.0058558993,
- -6.8658395,-7.0535574,9.080358,9.64777,6.8006287,
  -9.30843,11.598778,4.6842794,9.658299,10.869768;

layer2 =

- 0.165523,0.124461755,-0.5196377,0.34174114,-0.31841975
- -0.11245203,0.32940426,-0.5829561,0.3504779,-0.26547232
- -0.2840166,0.242611,-0.02847453,-0.11474156,-0.25086388
- 0.35463932,0.43074352,0.21673463,-0.6570365,0.108664304
- -0.41329414,-0.47529325,-0.3373336,-0.18833615,-0.3826401

- 0.1653832,-0.51819134,-0.15321259,0.61400867,0.1097562
- 0.22155711,0.6017437,0.11826656,0.2389555,-0.53169286
- 0.28087506,0.6309908,-0.47149956,0.51472926,-0.080736496
- 0.3917908,0.70016545,-0.018110754,-0.40906456,-0.48442152
- 0.6577122,-0.30955616,-0.011397076,-0.11844179,-0.51977944
- 9.541652,9.8262825,-9.366474,-9.083355,-9.470707

layer3 =

0.2923965 , 0.42190942,-0.17356397,-0.85945743, -0.2895543,9.089215;