Energy-Efficient Load Balancing Algorithms in Fog Computing

A

Thesis

submitted for the award of the degree of

DOCTOR OF PHILOSOPHY

by

Simar Preet Singh
(Reg. No.: 951603007)

Under the supervision of

Dr. Rajesh Kumar
Professor
Computer Science and Engineering Department
Thapar Institute of Engineering & Technology
Patiala, Punjab

Dr. Anju Sharma
Assistant Professor & Incharge
Punjab State Aeronautical Engineering College
MRS Punjab Technical University
Bathinda, Punjab

Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala, Punjab, India

September 2020

To my family members
for their love, support and encouragement

Certificate

This is to certify that the thesis entitled '**Energy-Efficient Load Balancing Algorithms in Fog Computing**', by Simar Preet Singh (Reg No: 951603007), a research scholar in the *Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala, Punjab, India*, for the award of the degree of '**Doctor of Philosophy**', is a record of an original research work carried out by him under our supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the Institute and in our opinion has reached the standard needed for submission.

The matter presented in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

(Simar Preet Singh)
Reg. No. 951603007

This is to certify that the above statement made by the candidate is correct and true to the best of our knowledge.

Dr. Rajesh Kumar
Professor
Computer Science and Engineering Department
Thapar Institute of Engineering & Technology
Patiala, Punjab

Dr. Anju Sharma
Assistant Professor & Incharge
Punjab State Aeronautical Engineering College
MRS Punjab Technical University
Bathinda, Punjab

Acknowledgements

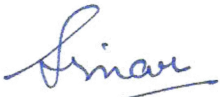
I would like to express my sincere thanks to all those people who made this research work possible. First and foremost, I would like to express my profound respect and gratitude to my supervisors, Dr. Rajesh Kumar and Dr. Anju Sharma, who has been guiding force behind this work. I am greatly bounded for their constant encouragement, invaluable guidance, and their valuable comments on my work. More importantly, I would like to thank for the patience they have shown in carefully reading and commenting on the manuscripts, and countless revisions of this dissertation. Their commitments and dedication to research have been and will continue to be a constant source of inspiration for me. I am fortunate enough to have such an supervisors who gave me the freedom to think independently and explore new ideas. I have no doubts that finishing my degree in proper and timely manner was impossible without their help and support. I am highly privilege to have got an opportunity to work with such wonderful advisors.

I would also like to thank my doctoral committee members Dr. Maninder Singh, Dr. Inderveer Chana, Dr. Sanmeet Kaur and Dr. Ankush Kansal for their invaluable suggestions, encouragements, and moral supports that helped me to improve my research work. I am also thankful to the Head of the Department, other faculty members and staffs, and Dean-RSP for their kind help carried out during my academic studies.

My special thanks to moral advisers Dr. Harsh Garg, Dr. Ashok Kumar and Dr. Harjeet Singh, for their guidance, constant support and insightful comments during the entire journey of my PhD life. I had a great time with many friends Mohd Abuzar Sayeed, Abhishek, Vivek, Bhavya, Deepali, and Sumedha to name a few at Thapar Institute of Engineering and Technology, Patiala. I would like to thank them for their support and encouragement.

I am grateful to my parents, sister and relatives, whose love, encouragement, and support made this research work possible.
I am thankful to Thapar Institute of Engineering and Technology, Patiala for providing the research scholarship to undertake my PhD research.

Finally, I would like to thank the Almighty God for bestowing me this opportunity and showering his blessings on me to come out successful against all odds.



Simar Preet Singh

Abstract

Fog computing is an amalgamation of many network technologies working at edges of networks, thus differentiating itself from cloud computing with an increased focus on traffic load balancing at edges. Fog computing combines shared, geographically distributed and heterogeneous resources to achieve high computational performance. The objective of fog computing is to provide enhanced responsiveness with reduced latency, that too near to the user (at one hop distance). Fog computing offloads huge tasks to cloud and performs latency-sensitive tasks with the help of collaborative heterogeneous fog nodes within the fog network, nearer to the end devices. This helps in reduction of excessive traffic on the network along with the optimal usage of the available resources. These resources may belong to homogeneous or heterogeneous environments like devices working in different institutions, different domains and may pose tasks that requires high computations. One of the major challenge in such heterogeneous and complex computing environments is devising energy-efficient load balancing algorithm(s) in fog environment. Such algorithm(s) should be efficient, robust, and scalable with optimal use of available resources.

To meet the growing traffic needs on the Internet as well as for optimal or minimal energy utilization between available fog nodes present in the fog zone, energy-efficient load balancing algorithm(s) are the current needs in fog computing environment. Efficient use of such algorithms will produce better Quality of Service (QoS) parameters (such as latency, responsiveness, availability, bandwidth, scalability, storage, energy consumption etc.) and increases performance of the system.

This research work mainly focuses on 'Energy-Efficient Load Balancing Algorithms in Fog Computing', which deals with designing of energy efficient fog load balancer and optimizing the fog network paths for better traffic management. Initially, an in-depth review of existing models, approaches and algorithms has been done. During the literature review, it has been observed that existing load balancers can be improved to meet the current emerging technologies necessities, and for that, there is a need to have proper design and optimal fog load balancer algorithm, embedded into our devised fog load balancer. The empirical results of various designs of fog load balancers shows that fuzzy based traffic management component is most appropriate for working with the imprecise nature of load passing through the interconnects of the network. In this research work, a fuzzy-based fog load balancer is devised using different levels of design (3-level, 5-level and 7-level) and tuning of fuzzy controls. This fuzzy logic based algorithm has been implemented for conducting load analysis of interconnects for managing traffic.

Use of optimization algorithms is one of the way to improve efficiency of the network in terms of responsiveness, latency and to avoid wastage of energy. Nano-Caches are

integrated for delivering contents efficiently, using search-based optimization techniques which are energy and response aware in nature. An algorithm namely Modified Teaching Learning Based Optimization (MTLBO) is devised and implemented in fog zone to find efficient route for forwarding contents using Nano-Caches and subsequently to improve content retrieval time. Mathematical distribution model of traffic/load is used for simulation process. MTLBO is compared with existing algorithms, namely, Teaching Learning Based Optimization (TLBO) algorithm and Simulated Annealing (SA) algorithm. The design of experiments (DOE) has been carried out to observe number of iterations, learning rate and network size.

The analysis shows that 3-level design is energy efficient for load balancing in fog zone due to reduced number of intervals in fuzzy design, reduced overheads in provisioning and improved responsiveness. Higher levels (5-level and 7-level) lead to creation of redundant fuzzy rules and wastage of resources. The optimization results show that Modified Teaching Learning Based Optimization (MTLBO) approach is better than Teaching Learning Based Optimization (TLBO) approach as it has less overheads in terms of memory (considering number of fog caches) and network size for delivering contents at remote areas. In comparison to the Simulated Annealing (SA) algorithm, MTLBO performs better in terms of execution time, overhead in terms of memory, and scalability as function of network size.

To validate the research work, various applications have been considered to check the evaluation of proposed energy efficient fog load balancer. The case presented in this thesis gives insights on how fog devices and data mining can be used for bringing people into main fold of the economy. The inclusiveness index of the person is computed on the basis of four aspects: fitness, her/his inner social circle, her/his reliability to remain in a place, and call analysis. While computing the fitness index, it was found that Naive Bayes (NB) algorithm has the maximum accuracy with respect to K-Nearest Neighbors (KNN), Decision Tree (DT) and Linear Discriminant Analysis (LDA). For computing inner social circle, Louvain algorithm helped to compute stability and strength of socio-economic ties of an individual. For geospatial and call analysis, insights from knowledge discovery algorithm such as FP-Growth helped to arrive at decision to qualify the person for inclusive program. The thesis ends with details on how to automate the inclusiveness index computation using machine learning. The research indicates that energy is the key constraint for implementing such programs. Hence, a theoretical analysis about energy efficiency is also explained in the thesis. In this research work, a unique system of computing inclusiveness score has been introduced and implemented using fog load balancer in fog network.

List of Publications

Refereed Journals: Published

1. **Simar Preet Singh**, Rajesh Kumar, Anju Sharma (2019), *"Efficient content retrieval in fog zone using Nano-Caches"*, *Concurrency and Computation: Practice and Experience*, Wiley, 32(2), pp. 1-20, IF: 1.148,
   https://doi.org/10.1002/cpe.5438,
   URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5438.

2. **Simar Preet Singh**, Anju Sharma, Rajesh Kumar (2020), *"Design and exploration of load balancers for fog computing using fuzzy logic"*, *Simulation Modelling Practice and Theory*, Elsevier, vol. 101, May 2020, 102017, IF: 2.677,
   https://doi.org/10.1016/j.simpat.2019.102017,
   URL: https://www.sciencedirect.com/science/article/abs/pii/S1569190X19301480

3. **Simar Preet Singh**, Anju Sharma, Rajesh Kumar (2020), *"Designing of Fog Based FBCMI2E Model Using Machine Learning Approaches for Intelligent Communication Systems"*, *Computer Communications*, Elsevier, vol. 163, pp. 65-83, IF: 2.816,
   https://doi.org/10.1016/j.comcom.2020.09.005,
   URL: https://www.sciencedirect.com/science/article/abs/pii/S0140366420319198

4. **Simar Preet Singh**, Rajesh Kumar, Anju Sharma, Anand Nayyar (2020), *"Leveraging Energy Efficient Load Balancing Algorithms in Fog Computing"*, *Concurrency and Computation: Practice and Experience*, Wiley, IF: 1.148,
   https://doi.org/10.1002/cpe.5913,
   URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5913

Contents

Title **Page No.**

Dedication **ii**

Certificate **iii**

Acknowledgements **iv**

Abstract **v**

Contents **viii**

List of Figures **xi**

List of Tables **xiii**

List of Notations **xv**

List of Abbreviations **xviii**

1 Introduction **1**
- 1.1 Introduction to Fog Computing 1
- 1.2 Communication Models for Fog Computing 3
- 1.3 Fog Computing Architecture 7
- 1.4 Fog Load Balancer 9
- 1.5 Challenges of Fog Computing 12
- 1.6 Motivation 13
- 1.7 Objectives of Thesis 14
- 1.8 Organization of Thesis 19

2 Literature Survey **22**
- 2.1 Fog Computing 22
- 2.2 Load Balancing in Fog Computing 24
  - 2.2.1 Taxonomy of Load Balancing 25
  - 2.2.2 Energy-Efficient Load Balancing 27
- 2.3 Research Challenges to Design Load Balancer 31
  - 2.3.1 Design Issues in Fog Load Balancing 31
  - 2.3.2 Comparative Analysis of Load Balancing Algorithms 33
- 2.4 Techniques for Optimizing Load Balancing 38
  - 2.4.1 Optimization Techniques for Fog Load Management 38

2.4.2 Comparative Analysis of Optimization Techniques 41
2.5 Tools Available for Fog Computing 46
2.5.1 Simulation and Emulation Tools for Fog Computing 46
2.5.2 Comparative Analysis of Fog Computing Tools 49
2.6 Applications of Fog Computing 49

3 Design Levels for Energy-Efficient Fog Load Balancer 53
3.1 Fuzzy Based Fog Load Balancer 53
3.1.1 Traffic Distributions for Fog Load Balancer 54
3.2 System Model of Fuzzy-Based Fog Load Balancer 56
3.3 Proposed Design Levels for Fuzzy-Based Fog Load Balancer 56
3.3.1 Design and Implementation of Fuzzy Based Fog Load Balancer . 57
3.4 Design of Fuzzy Logic Scenarios for Evaluation of Algorithms 61
3.4.1 Fuzzification and Membership Functions 62
3.4.2 Selection of Fuzzy Membership Functions and Inference Controller 73
3.5 Fuzzy Control Rules 74
3.5.1 Defuzzification 78
3.5.2 Strength of Rules 79
3.6 Analysis of Various Design Levels 80
3.6.1 Test Cases for Evaluation of 3-level, 5-level and 7-level Scenarios 80
3.6.1.1 3-Level Design of Fog Load Balancer 81
3.6.1.2 5-Level Design of Fog Load Balancer 86
3.6.1.3 7-Level Design of Fog Load Balancer 91
3.6.2 Comparative Analysis of 3-level, 5-level and 7-level Design Scenarios 97
3.7 Validation of Design Scenarios of Fog Load Balancer 98
3.7.1 Validation of 3-Level Design of Fog Load Balancer 99
3.7.2 Validation of 5-Level Design of Fog Load Balancer 104
3.7.3 Validation of 7-Level Design of Fog Load Balancer 108
3.7.4 Comparative Analysis of 3-level, 5-level and 7-level Fuzzy Load Balancer Designs 116
3.8 Summary and Conclusion 116

4 Optimization of Fog Load Balancer Using Energy-Efficient Algorithms 119
4.1 Teaching Learning Based Optimization (TLBO) 119
4.1.1 Teacher Phase 120
4.1.2 Learner Phase 120
4.2 Simulated Annealing (SA) Algorithm 121
4.3 Proposed Modified Teaching Learning Based Optimization (MTLBO) Algorithm 122
4.4 Implementation for MTLBO 123
4.4.1 Terminology and Definitions used for MTLBO 123
4.4.2 Fog Zone's System Model 125
4.4.3 Construction of Composite Metric for Fog Zone's System Model . 126
4.4.4 Normalization of Data 127
4.4.5 Implementation of Modified Teaching Learning Based Optimization Algorithm (MTLBO) 129
4.5 Analysis of Proposed MTLBO Algorithm 133

4.6 Comparison of MTLBO with TLBO and SA . . . 138
4.7 Summary and Conclusion . . . 139

5 Applications of Fog Computing Using Fog Load Balancer 141
5.1 Fog Based Communication Model for Inclusiveness of Informal Economy (FBCMI2E) . . . 142
5.1.1 Dataset for FBCMI2E . . . 142
5.2 Mathematical Trust Model of FBCMI2E . . . 147
5.2.1 Correlation Matrix . . . 147
5.3 Inclusiveness Qualification Computations for FBCMI2E . . . 152
5.3.1 Inner Circle Analysis . . . 153
5.3.1.1 Simulating Communities Using Erdos Renyi Pseudo Logic . . . 153
5.3.1.2 Community Detection Louvain Algorithm . . . 154
5.3.2 Mobile Call Analysis for FBCMI2E . . . 155
5.3.3 Geospatial Analysis for FBCMI2E . . . 156
5.4 Automating FBCMI2E Using Machine Learning . . . 158
5.4.1 Qualification Cut-Off for Geospatial Events . . . 159
5.4.2 Qualification Cut-Off for Mobile Calls events . . . 160
5.4.3 Measure of Modularity . . . 162
5.5 Energy Analysis for FBCMI2E . . . 163
5.5.1 Energy Model of FBCMI2E for Heterogeneous Device . . . 165
5.5.2 Energy Model of FBCMI2E for Nano-Cache Nodes . . . 166
5.5.3 Energy Model of FBCMI2E for Computational Intensive Nodes . 167
5.5.4 Computation of Total Energy for FBCMI2E . . . 167
5.6 Summary and Conclusion . . . 167

6 Conclusion and Future Directions 169
6.1 Conclusion . . . 169
6.2 Future Directions . . . 171

Appendices 173

Bibliography 177

List of Figures

Figure No.	Title	Page No.
1.1	Communication Model in Sensors Network using Ethernet	3
1.2	Communication Model in Sensors Network using Radio Links	4
1.3	Communication Model in Sensors Network using Wi-Fi	5
1.4	Communication Model in Sensors Network using Satellite	5
1.5	Communication Model for Cloud Bridge using Fog/Edge Computing	6
1.6	Transformation Era from Cloud Computing to Fog Computing [1]	7
1.7	Working of Fog Computing	8
1.8	Content Delivery Network (CDN)	11
1.9	Evaluation Strategy of Research Work	15
1.10	Block diagram of Phase 1	16
1.11	Block diagram of Phase 2	17
1.12	Flowchart of Optimization Algorithms	17
1.13	Workflow of Phase 3	18
2.1	Representation of a Load Balancer	24
2.2	Load Balancing Scenario	25
2.3	Types of Load Balancers	26
2.4	Scenario designed in iFogSim	47
2.5	Scenario in EmuFog	48
2.6	Scenario in FogNetSim++ Simulator	49
3.1	Phases of Fog Load Balancer	53
3.2	System Model of Fog Load Balancer	57
3.3	Load Balancer Framework (comprising of cloud and fog network zones)	59
3.4	Traffic Load considering Low Linguistic Term Value in 3-Level Design	64
3.5	Delay Sensitivity considering Medium Linguistic Term Value in 3-Level Design	65
3.6	Energy Consumption considering Medium Linguistic Term Value in 3-Level Design	65
3.7	Link Saturation considering High Linguistic Term Value in 3-Level Design	66
3.8	Traffic Load considering Low Linguistic Term Value in 5-Level Design	67
3.9	Delay Sensitivity considering Medium Linguistic Term Value in 5-Level Design	68
3.10	Energy Consumption considering Medium Linguistic Term Value in 5-Level Design	68
3.11	Link Saturation considering High Linguistic Term Value in 5-Level Design	69
3.12	Traffic Load considering LOW Linguistic Term Value in 7-Level Design	71

3.13 Delay Sensitivity considering Medium Linguistic Term Value in 7-Level Design . . . 71
3.14 Energy Consumption considering Medium Linguistic Term Value in 7-Level Design . . . 72
3.15 Link Saturation considering High Linguistic Term Value in 7-Level Design 73
3.16 Link Health considering Medium Linguistic Term Value in 3-Level Design 74
3.17 Example of Defuzzification . . . 79
3.18 Number of Rules Affected vs Strength in 3-Level Design . . . 83
3.19 Total Cases vs Cases Affected . . . 83
3.20 Total Fuzzy Rules vs Total Fuzzy Rules Affected . . . 83
3.21 Comparative Analysis of 3-level, 5-level and 7-level Designs . . . 98

4.1 Assumed Fog Zone and Components of System Model . . . 126
4.2 Varying the Number of Fog Caches . . . 134
4.3 Varying the Moderation Rate . . . 136

5.1 Inclusiveness Program for Qualifying Process in FBCMI2E . . . 143
5.2 System Model of FBCMI2E . . . 144
5.3 Caching and Service Interface Layer . . . 146
5.4 Fuzzy Logic Business Rules . . . 146
5.5 FBCMI2E (Fitness Dataset) [2] . . . 148
5.6 Correlation Matrix of FBCMI2E . . . 149
5.7 Comparison of Machine Learning Models on Fitness Data . . . 151
5.8 Community Detection of FBCMI2E . . . 154
5.9 Call Analysis of FBCMI2E Dataset . . . 155
5.10 Result of Call Analysis Using FP-Growth . . . 157
5.11 Geospatial Analysis of FBCMI2E Dataset . . . 159
5.12 Result of Geospatial Analysis . . . 160
5.13 Geospatial Events Cut-Off . . . 161
5.14 Qualification Cut-Off of Mobile Calls Events . . . 161
5.15 Measure of Modularity . . . 161
5.16 Comparison of Machine Learning Models . . . 164

I Coding (View 1) involved in Simulation Process . . . 174
II Coding (View 2) involved in Simulation Process . . . 175
III Results of Simulation Process . . . 176

List of Tables

Table No.	Title	Page No.
2.1	Design based Comparative Analysis of Related Work	34
2.2	Comparative Analysis of Literature Work	42
2.3	Comparison of Various Fog Computing Tools	50
3.1	Design Parameters Used in System Model	61
3.2	Descriptor Linguistic Terms Values	63
3.3	Input Variable: Data Boundaries for Traffic Load in 3-Level Descriptor	64
3.4	Input Variable: Data Boundaries for Delay Sensitivity in 3-Level Descriptor	64
3.5	Input Variable: Data Boundaries for Energy Consumption in 3-Level Descriptor	65
3.6	Input Variable: Data Boundaries for Link Saturation in 3-Level Descriptor	66
3.7	Input Variable: Data Boundaries for Traffic Load in 5-Level Descriptor	66
3.8	Input Variable: Data Boundaries for Delay Sensitivity in 5-Level Descriptor	67
3.9	Input Variable: Data Boundaries for Energy Consumption in 5-Level Descriptor	68
3.10	Input Variable: Data Boundaries for Link Saturation in 5-Level Descriptor	69
3.11	Input Variable: Data Boundaries for Total Load in 7-Level Descriptor	70
3.12	Input Variable: Data Boundaries for Delay Sensitivity in 7-Level Descriptor	70
3.13	Input Variable: Data Boundaries for Energy Consumption in 7-Level Descriptor	72
3.14	Input Variable: Data Boundaries for Link Saturation in 7-Level Descriptor	73
3.15	Output Variable: Data Boundaries for Link Health	73
3.16	3-Level Fuzzy Rules for Sensor Data Boundaries	75
3.17	3-Level Fuzzy Rules for Video Data Boundaries	75
3.18	3-Level Fuzzy Rules for Web Data Boundaries	76
3.19	5-Level Fuzzy Rules for Sensor Data Boundaries	76
3.20	5-Level Fuzzy Rules for Video Data Boundaries	77
3.21	5-Level Fuzzy Rules for Web Data Boundaries	77
3.22	7-Level Fuzzy Rules for Sensor Data Boundaries	77
3.23	7-Level Fuzzy Rules for Video Data Boundaries	78
3.24	7-Level Fuzzy Rules for Web Data Boundaries	78
3.25	3-Level Descriptors Case for analysis of Sensor Data	81
3.26	3-Level Descriptors Case for analysis of Video Data	83
3.27	3-Level Descriptors Case for analysis of Web Data	85
3.28	5-Level Descriptors Case for analysis of Sensor data	87
3.29	5-Level Descriptors Case for analysis of Video data	88
3.30	5-Level Descriptors Case for analysis of Web data	90

3.31 7-Level Descriptors Case for analysis of Sensor data 92
3.32 7-Level Descriptors Case for analysis of Video data 94
3.33 7-Level Descriptors Case for analysis of Web data 96
3.34 Sensor Data 98
3.35 Video Data 99
3.36 Web Data 99
3.37 3-level Cases for Sensor Data 100
3.38 3-level Design for Sensor Data 101
3.39 3-level Cases for Web Data 102
3.40 3-level Design for Web Data 103
3.41 3-level Cases for Video Data 104
3.42 3-level Design for Video Data 105
3.43 5-level Cases for Sensor Data 106
3.44 5-level Design for Sensor Data 107
3.45 5-level Cases for Web Data 108
3.46 5-level Design for Web Data 109
3.47 5-level Cases for Video Data 110
3.48 5-level Design for Video Data 111
3.49 7-level Cases for Sensor Data 112
3.50 7-level Design for Sensor Data 113
3.51 7-level Cases for Web Data 114
3.52 7-level Design for Web Data 115
3.53 7-level Cases for Video Data 116
3.54 7-level Design for Video Data 117
3.55 Comparative Analysis of 3-level, 5-level and 7-level Fuzzy Load Balancer Designs 118

4.1 Computation Parameters used for Experimental Work 130
4.2 Input Parameters for MTLBO 131
4.3 Experimental Configuration Parameters for MTLBO 132
4.4 Comparison of MTLBO with TLBO and SA 139

5.1 Fog/IoT/Wireless Medical Sensors and Devices Used as Analysers in Inclusive Technology 145
5.2 Performance of Machine Learning Models on Fitness data 150
5.3 Medical Ranges of parameters 152
5.4 Qualifying/Not-Qualifying Criteria 162
5.5 Performance of Machine Learning Models on Fitness data 162
5.6 Parameters Requirements for FBCMI2E 165
5.7 Heterogeneity Parameters for FBCMI2E 166

List of Notations

Chapter 3

H_x	Health matrix of link
P	packets
RP	routingpaths
HS	healthstatus
TL	trafficload
DS	delaysensitivity
EC	energyconsumption
LS	linksaturation
R	rules
RS	rulestrength
No_{load}	NoLoad
Min_{load}	MinimumLoad
$BAvg_{load}$	BelowAverageLoad
Avg_{load}	AverageLoad
$AAvg_{load}$	AboveAverageLoad
Max_{load}	MaximumLoad
$Peak_{load}$	PeakLoad
No_{delay}	NoDelay
Min_{delay}	MinimumDelay
$BAvg_{delay}$	BelowAverageDelay
Avg_{delay}	AverageDelay
$AAvg_{delay}$	AboveAverageDelay
Max_{delay}	MaximumDelay
TO_{delay}	TimeOutDelay
$VMinute_{consumption}$	VeryMinuteConsumption
$VLow_{consumption}$	VeryLowConsumption
$Low_{consumption}$	LowConsumption
$BAvg_{consumption}$	BelowAverageConsumption
$Avg_{consumption}$	AverageConsumption
$AAvg_{consumption}$	AboveAverageConsumption

Continued on next page

$Peak_{consumption}$	PeakConsumption
$VMinute_{saturation}$	VeryMinuteSaturation
$VLess_{saturation}$	VeryLessSaturation
$Low_{saturation}$	LowSaturation
$BAvg_{saturation}$	BelowAverageSaturation
$Avg_{saturation}$	AverageSaturation
$AAvg_{saturation}$	AboveAverageSaturation
$Full_{saturation}$	FullSaturation
$Low_{quality}$	LowQuality
$Med_{quality}$	MediumQuality
$High_{quality}$	HighQuality

Chapter 4

Avg_m	Average Marks
$C_{minscore}$	Min Score of Class
$C_{bestscore}$	Best Score of Class
T_c	Total Cycle Time
T_d	Transmission Delay
R_t	Render Time
T_{cyc}	Total Cycles
N_{cb}	Number of Content bytes
A_{crt}	Average Content Retrieval Time
N_t	Time Frame
E_{dist}	Euclidean Distance
E_a	Access Energy Consumed
E_t	Transmission Energy Consumed
E_r	Render Energy Consumed
E_c	Energy Consumed
J	Joules
R_u	Resource Utilization
P_r	Proximity
T_m	Minimum Threshold Matrix
C_g	Cloud Group
F_g	Fog Zone Group
A_t	Access Time,
T_d	Transmission Delay
R_t	Render Time

Continued on next page

List of Notations – *Continued from previous page*

B_w	Bandwidth
H_m	HealthMatrix
A_{B_w}	Available Bandwidth
U_{B_w}	Used Bandwidth
N_{bytes}	Request by source cache in bytes
No_{cache}	Number of Cache Servers
Mod_{Rate}	Learning/Moderation Rate
$MinB_w$	Minimum Bandwidth
$MaxB_w$	Maximum Bandwidth
$MinE_c$	Minimum Energy
$MaxE_c$	Maximum Energy
$MinA_{crt}$	Minimum Average Content Retrieval Time
$MaxA_{crt}$	Maximum Average Content Retrieval Time
$InitS_c$	Initial Score of Class
$MinS_c$	Minimum Score of Class
C_s	Cumulated Score
C_{score}	Current Score
P_{score}	Previous Score
T_{score}	Total Score

Chapter 5

P_{corr}	Pearson Correlation
T_{eexp}	Total Energy Expenditure
N	Number of fog devices
m_0	Fraction of computation intensive nodes
m	Fraction of Nano-cache nodes
$1-m$	Fraction of normal Nano-cache nodes
$N*m*m_0$	Total number of computation intensive nodes
$N*m*(1-m_0)$	Total number of Nano-cache nodes
E_0	Initial energy of the normal Nano-cache nodes
a	Extra energy in the Nano-cache nodes
b	Extra energy in the computation intensive nodes
$E_0(1+b)$	Energy of each computation intensive node
$E_0(1+a)$	Energy of each Nano-cache node

List of Abbreviations

IoT	Internet of Things
GA	Genetic Algorithm
PSO	Particle Swarm Optimization
LR	Logistic Regression
LDA	Linear Discriminant Analysis
KNN	k-Nearest Neighbors
CDN	Content Delivery Network
QoS	Quality of Service
FRAN	Fog Radio Access Networks
Wi-Fi	Wireless Fidelity
MHz	megahertz
BLE	Blue Low Energy
HR	Heart Rate
$\boldsymbol{SpO_2}$	Saturated Oxygen in the blood
ZB	zettabytes
LAN	Local Area Network
WAN	Wide Area Network
TCO	Total Cost of Ownership
L1	level 1
L2	level 2
L3	level 3
CDN	Content Delivery Network
DOE	Design of Experiments
VM	Virtual Machine
SDN	Software Defined Network
GA	Genetic Algorithm
PSO	Particle Swarm Optimization
ACO	Ant Colony Optimization
GbE	Gigabit Ethernet
TLBO	Teaching Learning Based Optimization
SA	Stimulated Annealing

Continued on next page

List of Abbreviations – *Continued from previous page*

MTLBO	Modifed Teaching Learning Based Optimization
NB	Naive Bayes
KNN	K-Nearest Neighbors
DT	Decision Tree
LDA	Linear Discriminant Analysis
ICI	Inner Circle Index
FIW	Fitness Index for Work
GDA	Geospacial Data Analysis
MLA	Mobile Call Log Analysis
TWA	Trust Worthiness Analysis
CPU	Central Processing Unit
DMo	Data In Motion
IoE	Internet of Everything
LBMM	Load balancing min-min
TCP	Transmission Control Protocol
IP	Internet Protocol
MMOG	Massive Multi-player Online Game
QoE	Quality of Experience
FOX	Fast Offset XPath
VANET	Vehicular Ad-hoc Network
OMNeT++	Objective Modular Network Testbed in C++
AWS	Amazon Web Services
SLA	Service Level Agreements
RTES	Real-time Efficient Scheduling
FCFS	First Come First Serve
MBFD	Modified Best Fit Decreasing
MBFH	Modified Best Fit Heuristic
LTE	Long Term Evolution
EPC	Evolved Packet System
ADM	Architecture Driven Modernization
MDE	Model Driven Engineering
OS	Operating System
VMM	Virtual Machine Monitor
BLA	Bees Life Algorithm
CSP	Constraint Satisfaction Problem
FRAS	Fuzzy-based Real-time AutoScaling
LVS	Linux Virtual Server

Continued on next page

List of Abbreviations – *Continued from previous page*

IGD	Inverted Generational Distance
PL	Plausibility Level
NS	Network Simulator
IO	Input-Output
FRAN	Fog Radio Access Network
NFV	Network Function Virtualization
CCFF	Cognitive Caching approach for Future Fog
SCC	Smart Collaborative Caching
VOD	Video-On-Demand
CS	Content Score
PPP	Poisson Point Process
PE	Processing Elements
RCPSP	Resource Constrained Project Scheduling Problem
HV	Hypervolume
GD	Generational distance
IGD	Inverted generational distance
LFU-LB	Least-Frequently Used Load Balancing
SE	Semi-Edge caching
ICN	Information Centric Networking
MM3C	Multi-Source Mobile Streaming in Cache-enabled Content-Centric Networks
TLBMO	Teaching Learning Bird Mating Optimization
VRTT	Virtual Round Trip Time
CCN	Content-Centric Networking
DNS	Domain Name Server
CLB	Cloud Load Balancing
STS	Science, Technology and Society
MPLS	Multi-Protocol Label Switching
GMPLS	Generalized Multi-protocol Label Switching
SIP	Session Initiation Protocol
WPAN	Wireless Personal Area Network
HTTP	Hyper Text Transfer Protocol
BAN	Body Area Network
PAN	Personal Area Network
WSAN	Wireless Sensor Area Network
IEEE	Institute of Electrical and Electronics Engineers
COG	Center of Gravity

Continued on next page

List of Abbreviations – *Continued from previous page*

OL	Overload
NL	Normal-load
UL	Under-load
NoL	No-load
TJ	Traffic Jam
PL	Peak-load
CL	Congestion
ISP	Internet Service Provider
PF	Pareto Front
ETLBO	elitist TLBO
FBCMI2E	Fog Based Communication Model for Inclusiveness of Informal Economy
CLA	Call Log Analysis
M	Modularity
HIPAA	Health Insurance Portability and Accountability Act
ER	Erdos Renyi
LDA	Linear Discriminant Analysis
PCA	Principal Component Analysis
GPRS	General Packet Radio Service
Q	Qualified
NQ	Not-Qualified
SVC	Support Vector Machine
NAN	Not a Number
NN	Neural Network

Chapter 1

Introduction

This chapter begins with the introduction to the fog computing and continues with discussion on evolution of various communication models that lead to present status of fog networks. The chapter gives information about emerging architectures that are currently in use for solving business and scientific problems related to fog computing. Besides this, the chapter covers ideas, terms, definitions, concepts, tools, and methods that are required for understanding the details of this research work. The basics of traffic management and load balancing in the fog zone are also elaborated in detail.

1.1 Introduction to Fog Computing

Fog computing is a way by which the devices, deployed at edges, carry out significant operations such as storage, computation, and communications. Fog computing is an amalgamation of many network technologies. Currently, the industry is passing through a phase of consolidation comprising various technologies. Problems of design, development and deployment of fog networks are getting lot of attention and new iteration of progress associated with fog computing is also underway.

CISCO developed fog computing in 2014 [3]. Fog computing consists of multiple fog nodes arranged in a network which are placed at just one hop from the user. Due to the placement of these multiple fog nodes that are placed very close to the user, the data is processed locally by fog nodes rather than migrating the data to cloud server placed far away from the user [4, 5, 6, 7]. Dastjerdi *et al.* [8] defined fog computing as the distributed computing environment extending the services of cloud computing to one hop distance from the user. Fog computing technology provides networking, computation, storage and management services between the end users (edges) and the cloud data centers [9, 10]. This supports protocols for communication, mobility, resources to perform computing, distributed analysis of data and integration of cloud for addressing latency-sensitive applications that needs minimum delay. In Marin-Tordera *et al.* [11], fog node is defined as

a device where the fog computing is deployed.

The layout of fog computing is designed in such a manner that the control decisions of outbound routing are separate from the decisions of inbound packets [12]. This way it enables computing operations and definitions to be realized at the edge more efficiently as compared to client-server design. Using this way of arrangement, proximity to the user base is achieved and consequently the efficiency of services such as video rendering, buffering, content retrieval, load balancing etc. increases [13].

Load balancing is one of the major thrust area in fog computing these days. Most of the organizations (OpenFog Consortium as well as many other research organizations) are working on it. Load balancing is a technique used to optimize network efficiency, reliability and capacity. Load balancing performs many functions such as distribution of client request to various network keeping in view the load across multiple active servers. This ensures availability, enables flexibility for addition or removing the servers. Load balancing is used to improve responsiveness and increases ability of applications to become more efficient. With the help of load management applied at edges, fog computing is able to support numerous technologies such as Internet of Things (IoT), augmented reality, content management, fog machines or robotics, and virtual reality [14]. The researchers are applying multiple methods and techniques to resolve the challenges of load balancing, responsiveness, buffering, etc. The techniques include the applications of optimization techniques such as genetic algorithm (GA), particle swarm optimization (PSO) and fuzzy logic [15, 16]. Machine learning algorithms such as neural network, logistic regression (LR), linear discriminant analysis (LDA), k-nearest neighbors (KNN) etc. are also in use [17]. Content Delivery Network (CDN) also provides the better solution for such load balancing problems. In terms of hardware, the researchers are using the idea of nano-devices, nano-server or nano-machines to improve the efficiency at the edges of the fog network.

An exploration of the current paradigms in fog computing shows that IoT systems, medical sensors, health instruments, ocean and mining observations need last-mile management for improving quality of service (QoS). This current research work also explores QoS levels, that have been redefined for the edge devices. There is a clear-cut division of the levels of services that are expected from a cloud network and fog devices that are engaged at the edges. This is because the degree of traffic pattern is incoherent, voluminous, and interconnects which have increased many folds. Due to this, these networks inherently have high degree of fuzziness in terms of traffic management.

Various QoS parameters such as latency, responsiveness, availability, bandwidth, scalability, storage, energy consumption, network performance etc. are taken into considerations. Currently, there are confirmatory signals from the industry and research community that the existing cloud service providers need to overcome the issue of latency and to fall back upon cloud-based solutions only, is not working properly. The cloud-based technology is

on maturity phase, but the services for which it was designed, is getting obsolete. There is an urgent need to use fuzzy mathematics, machine learning models and optimization techniques such as Nano-Caches to improve the latency of the network [18]. The fog load balancer, linked with cloud, receives data from sensors and in many cases, directly deal with live streams of data. The bandwidth may be constrained at the edges of the network. There may not be persistence storage possible at the last mile device network. The power budget of devices may be too low as compared to the amount of power provided to set the cloud service and fog network. Consequently, the network at edges of the cloud needs to share some degree of load [19, 20]. Due to this, the edge network has to be small and highly constrained as compared to the cloud. All these factors mentioned above, forces us to rethink the way by which current networks are designed and deployed. There is also a need to re-evaluate the way network performances are measured from the core of the cloud network to the edge of the network.

1.2 Communication Models for Fog Computing

In this section, evolution of the network technologies related to efficient communication has been discussed. Figure 1.1 gives glimpses of communication that is based on old paradigm (wired lines).

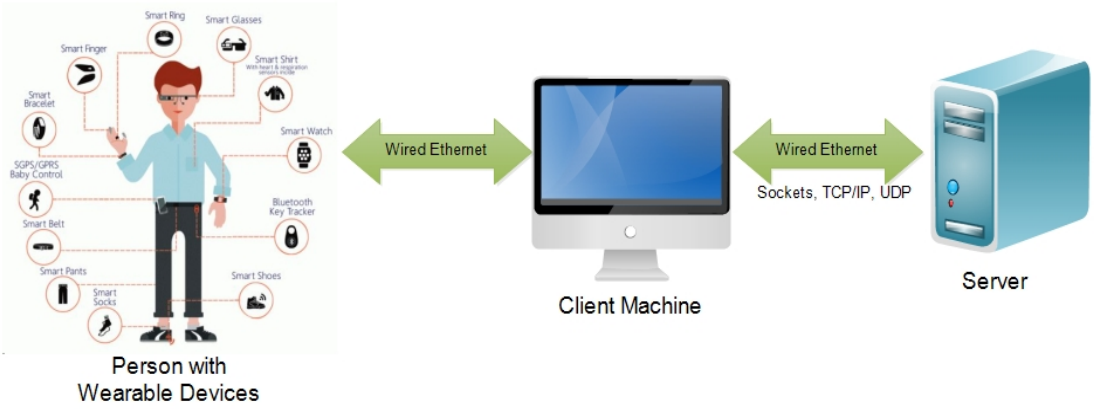


Figure 1.1: Communication Model in Sensors Network using Ethernet

In this model (Figure 1.1), the human body get connected with the help of wired sensors and Ethernet links to the clouds of insurance and hospital organizations. This model works well when the subject (person or participant) is under strict observation and is immobilized. This way of putting sensors and computers is based on the proven technologies such as Ethernet that came in the 1970 - 80 years. This is when microcontrollers were connected using Ethernet ports and cables, and it does not involve the use of radio links. Moreover, there is no requirement of large scale deployment. Hence, no concept of cloud

computing or specializations of grids/clusters of machines. There was no need of caching as there was direct connection with the server. The server stores information on all aspects and is interfaced using socket or ports [21]. The workflow is simple with limited application due to which they do not require any kind of sophisticated traffic engineering algorithms. All these factors force us to think hard on its use as inclusiveness technology. Such model can be useful to collect medical data but not limited to it. This may have few applications in computing inclusiveness based on health but no one considered factors such as mobility, social and financial.

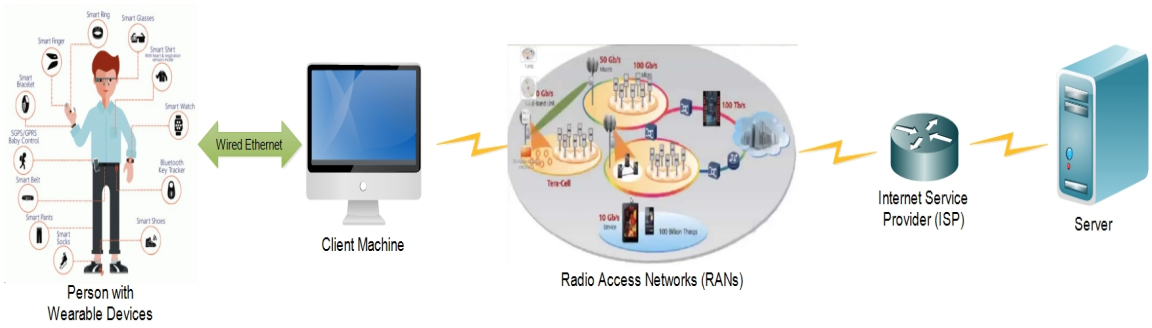


Figure 1.2: Communication Model in Sensors Network using Radio Links

Recent advances in radio technology (Figure 1.2) has led to a communication model that is called Fog Radio Access Networks (FRAN). Before these advances, communication over long distances was only possible using radio links. Mobility is the first advantage over the wired (Ethernet) connection. This network has now taken multiple iterations in terms of technical progress. Current literature shows the emergence of sophisticated algorithms such as neural network, fuzzy logic etc. for allocation and division of resources within the network. Since, such networks cover large distances, which involves the use of routers and switches. A large number of ultra-low powered wireless body sensors groups can be connected with the radio links either directly to long-range radio equipment or indirectly via interconnects. The radio cloud services interact with these radio devices. These networks, typically work in bands of 902-928 MHz and 2400-2483 MHz making them a low-budget technology implementation, especially, where the cellular network infrastructure or phones are not available. This technological setup qualifies as inclusive technology platform. It can transform the lives of many people living in remote areas as they can avail services such as remote telemedicine services for community.

In this arrangement (Figure 1.3) the smart IoT based networks get connected to cellular communication infrastructure. Such networks may have wired wearable or wireless devices [22]. The uplink connection requires a radio link transmitter, which consumes a fair amount of energy to reach cellular tower infrastructure. Such networks have application in setting up links to remote locations such as mountains, rivers, oceans and far away settlements. Such networks may require advance edge caching technologies to support

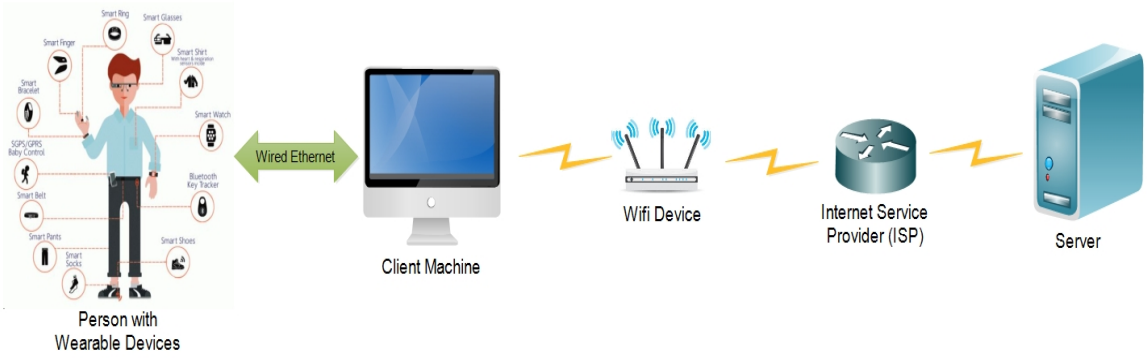


Figure 1.3: Communication Model in Sensors Network using Wi-Fi

the network performance. Traffic load management may also be required and lot of cross layer orchestration may be required.

Figure 1.4: Communication Model in Sensors Network using Satellite

Figure 1.4 shows the use of cellular technology and satellites for communication and monitoring. The enormous expanse of such networks can be helpful in connecting with large remote population. For secure and efficient working of such networks, there is a need to have multiple zones within the large networks. Such networks can support cognitive radio bands, radio resource virtualization and advance combinations of technologies. Such networks can support inclusiveness programs. A deep look into the network architecture shows that there are no systems that work on specific protocols and devices. There are multiple completing communication technologies that work together to form an efficient network.

Figure 1.5 is an advancement over the previous models by adding Bluetooth enabled devices as well as Wi-Fi-based devices. In this arrangement, the mobile application acts as gateways and can receive data from Bluetooth as well as Wi-Fi-enabled connection lines. The Bluetooth standard was created in 1998. It was added to 802.15.4 standards but it can also work as an independent group. It works in the same range of Wi-Fi (2400-2483 MHz), which is the free zone for licensee holder. Another version of Bluetooth is

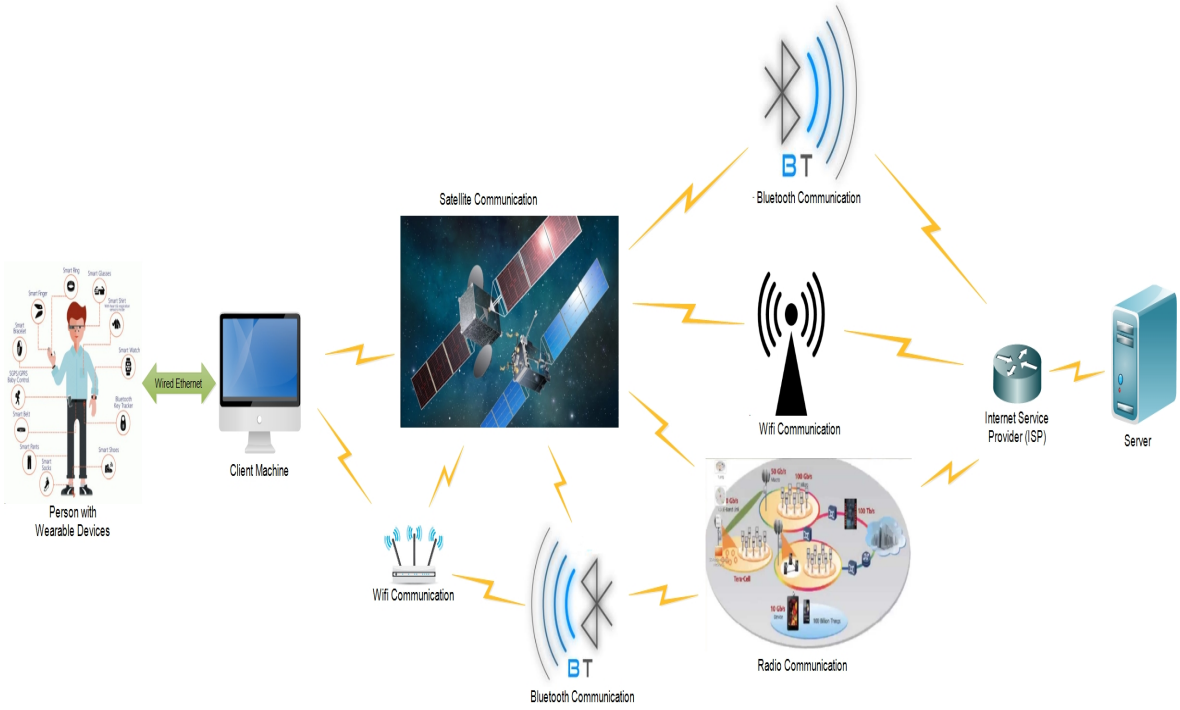


Figure 1.5: Communication Model for Cloud Bridge using Fog/Edge Computing

called BLE or simply Blue Low Energy. This model is useful for those medical devices that directly need to connect with a mobile phone rather than the Wi-Fi router and then connect with inclusive technology platform. At the same time, it also brings issues with respect to latency, jitter and delays in communication. Hence, the need for edge/fog zone optimization. This happens due to the need of high provisioning required by portable medical devices. The BLE based devices have predefined profiles for maintaining their functions and resources.

Figure 1.5 shows that the smartphones play an important role along with the Wi-Fi router. In order to monitor the subject (person or participant) body functions such as heart rate (HR), SpO_2 (saturated oxygen in the blood), is always desirable that such devices be light in weight, flexible and compact, so that it does not give any inconvenience to the subject. One of the best ways to do it is by including smartphone in the network topology, as smartphones are now widely present. The sensors, being Wi-Fi enabled and connected to smartphone (acting as Wi-Fi hotspot or Wi-Fi router), are available in the topology to connect with cloud.

The evolution of all these models is giving way to new kind of technological challenges. These challenges are creating renewed options for acceleration of next industrial revolution as well as in new social-economic order. The issues, however, still remains on how to organize to the potential of these technologies as there is no single policy mix or ideal technology matrix that can be used for social cause. The next section discusses about the architecture of such technology that is currently in use.

1.3 Fog Computing Architecture

The emergence of Internet of Things (IoT) technology leads to the development of a wide variety of IoT devices. Smart home applications, tablets, smart-phones, edge routers, cellular base stations, smart traffic systems, smart energy meters, connected vehicles, smart building controllers etc. use IoT devices. The rapid increase in these varieties of network edge devices results in huge data transmission on a continual fashion within a short span of time. This raw data needs to be processed and respond quickly.

Cloud computing consists of cloud data-centers, that are centrally deployed on a global scale. Without fog computing technology, these data-centers needs to process huge data coming from IoT devices. In addition, as the physical distance between the cloud and the user increases, transmission latency and response time increases with it. Users might become exhausted with the services provided [23, 24]. The processing speed in such a large environment is mostly dependent on the performance of the user device. The approachable solution to these problems is the fog computing.

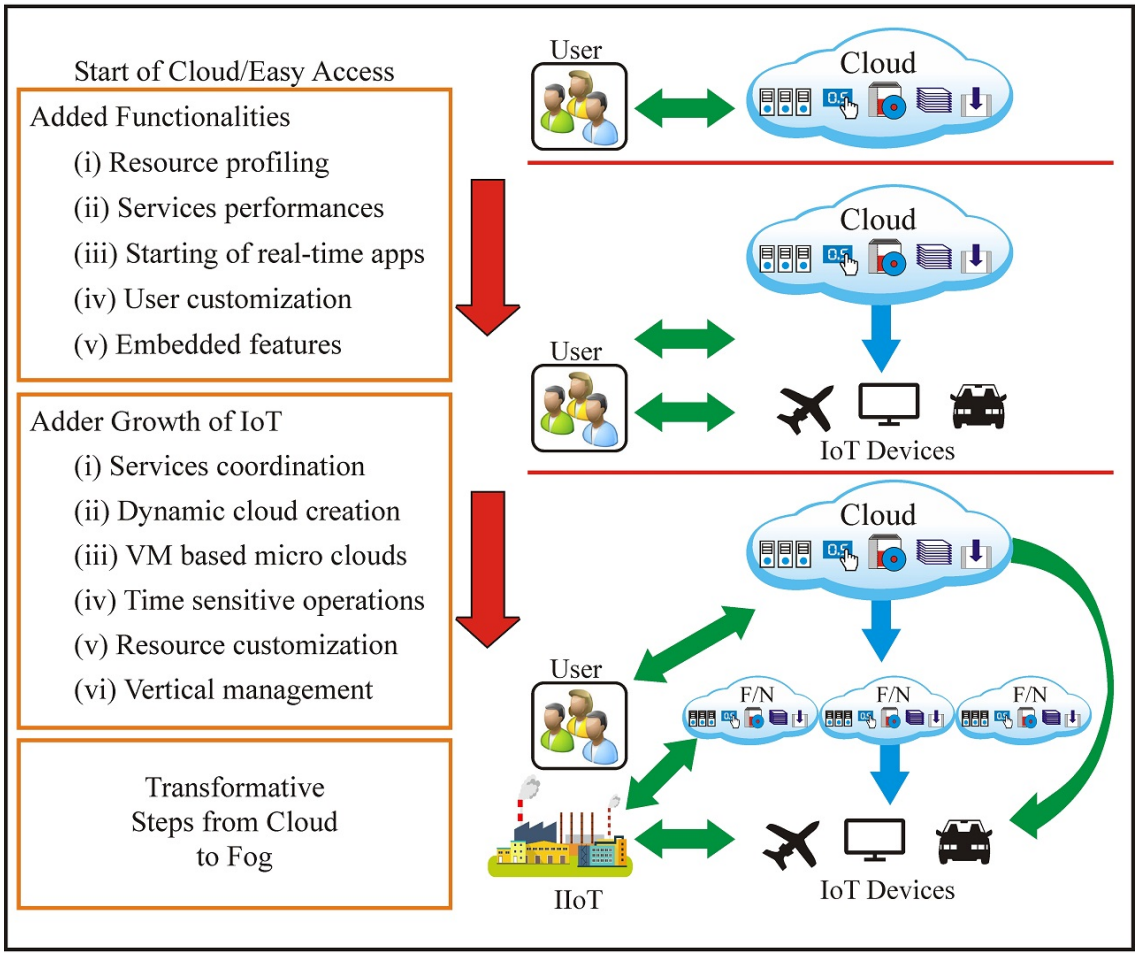


Figure 1.6: Transformation Era from Cloud Computing to Fog Computing [1]

Figure 1.6 represents the transformation of cloud computing to the fog computing with adding functionalities with change of technology. Initially, users' data was directly pro-

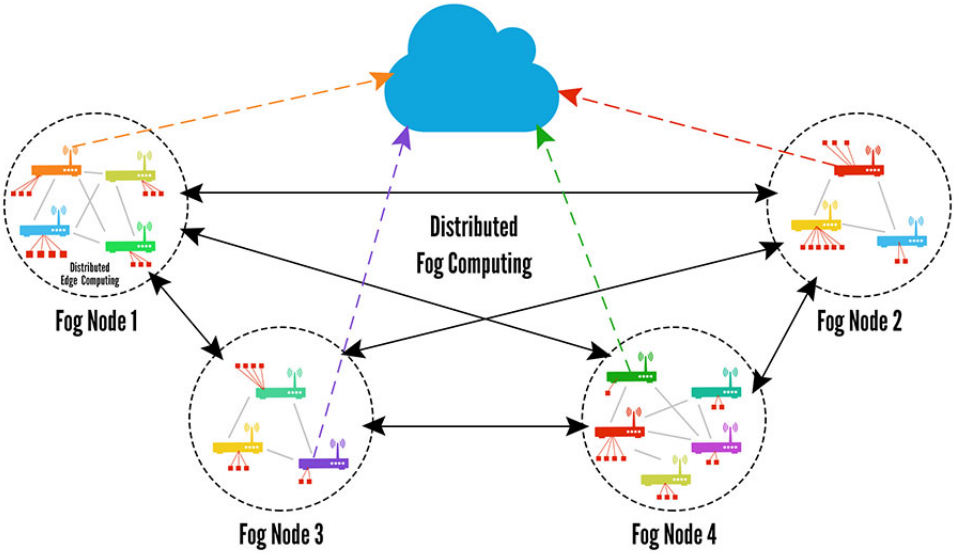


Figure 1.7: Working of Fog Computing

cessed by the cloud data centers. But with the emergence of fog computing, latency sensitive applications are processed by the fog nodes and latency in-sensitive applications are processed directly by the cloud datacenters.

Figure 1.7 represents fog nodes in a distributed arrangement within a fog network. Fog network consists of multiple fog nodes, each communicating with one another and also in sync with the cloud data center. Fog computing and cloud computing work together to enhance quality and performance of any system.

In today's era of computing, huge data is increasingly generated by the proliferating IoT devices at the core network. This large quantity raw data is initially processed by the fog networking or fog computing technology. Fog computing comprises of micro clouds (fog nodes) that are placed at the edge of the data originator. These micro clouds have the mechanism which tells how the data is to be processed before going on to the network. Hence, fog computing structure is useful in big IoT data analytics. This is on emerging phase and requires research initiatives on how effective knowledge can be obtained for smart decisions [12].

According to the CISCO estimation report, 847 zettabytes (ZB) of data will be generated by using IoT devices by 2021 [25]. Generated data is not useful and needs to be mined and filtered. Fog computing overcomes this processing of data at the edge of the network. As a result, fog computing overcomes some of emerging big data problems using fog load balancer, that is proposed and devised in this research work [26].

1.4 Fog Load Balancer

Cloud technologies are mature enough for giving birth to next generation of networks due to intrinsic challenges. The idea of cloud was conceived due to the necessity of the complexities faced in the LAN, WAN technologies, especially in context of maintenance and payload [27]. It is self-evident that due to explosion of internet related services, the conventional LAN, WAN technologies required more improvisations [28]. These improvisations were then compensated with the cloud technologies. These cloud technologies initially helped organizations to reduce the burden of total cost of ownership (TCO). It has improved the services by doing division and specialization of various kinds of business services. The companies no longer require maintaining their own hardware inventories nor do they have to worry about scaling up due to increase in growth of internet related economics. Now, companies are able to take resources on rent (storage, bandwidth, memory, and network), and they do not need to worry much on scaling up when their business grow at a faster speed as compared to the acquired hardware. With the passage of time, large number of cloud service providers mushroomed and the market became more service and subscription oriented. The companies now could proffer to various kinds of plans as per their payload and QoS parameters agreed with the customers [29].

Recently, it has been felt that the cloud technologies have capability issues. This is related to the last mile integration/customization with the cloud services and responsiveness [30, 31]. The sensors and the devices that are connected at the last mile are also required to be connected 24 hours with the cloud servers. For example, remote weather station on top of a mountain or a wireless network of sensors in a deep sea required a special attention to manage the resources at the edge of the cloud infrastructure for maintaining cloud connectivity. The sensors or the devices that generate highly time-sensitive data are required to sync with the cloud servers and fog devices [32, 33]. These have less sensitive data collection capabilities, thus, may require data aggregators in-between the cloud servers and devices at the last mile. Hence, depending upon the situation, the data payload needs to be managed [34, 35]. There are also sensors that generate data asynchronously with discrete timelines. Some sensors carry bio-physical data that can never be accepted with any corruption. In general, sensors generate data in bursts and in continuous manner.

There is thin line difference between the virtualized fog device and cloud device; it is the position and application for which they have been deployed [36, 37, 38]. If the device is deployed for customization of the last mile services or sensing some data then the device may be classified as fog device otherwise the device is a cloud device [39]. Another factor that may be considered for classification is resources with which it is made up of. If the device has sensing ability as well as limited computing infrastructure then also it may be considered as a fog device. These fog devices may be managed with the help of

self-serving interfaces and algorithms that manage the power budget and payload budget. These algorithms may be residing on the virtual fog device or may be managed from other device.

In datacenter, power budgets are computed either to reduce the operational cost or to conform to the green standards. The power budgets have direct relation to the task load and their limit is set dynamically based on the current demand of the services. Above all this, it must conform to the green technology standards. The same concepts and standards apply at the edges of the network. In a way, this is basis of Internet of Things (IoT) which allows millions processes to be controlled remotely [40, 41]. These can be accessed from distances in presence of right mix of network infrastructure (the edge, access, core, and datacenter networking infrastructure) [42]. Industry feedback shows that scaling, in terms of computing and management of the resources, will now come from software defined components, networks and intelligent algorithms rather than from hardware investment. Therefore, operations such as load balancing will be done with help of soft components.

The need for edge computing is to provide services with no buffering, no delay or without any jitter in transmission. There has been an evolution of technologies in this direction and software-defined devices have been designed with inbuilt algorithms to provide feedback loop which overcomes issues of content retrieval time. The evolution of such technologies primarily started from the hardware point of view, when the concept of the L1, L2, L3 cache came into existence [43, 44]. The purpose of cache was to increase the capacity for providing faster random read/write access. This concept was further extended into the creation of full fledge memory servers or cache servers, that would increase the power of the network in terms of improving persistence, temporary storage and memory access to the user, based on their requests [45]. With the technical advancements, specialized servers with specific configurations are in demands. Specialized servers played a significant role in the client-server architectures and networks as well. However, need is to build specialized software components that would help in improving the capacity and speed of the network data [46, 47]. These specialized servers as well as softwares use the concept of virtual memories and caches algorithms that helps virtual devices, computers and networks to optimize capacity and speed. Cache algorithms are either implemented as de-facto policies in the networks or as programmable instructions [48, 49].

With the explosion of the Internet and Web 2.0 technologies, everyone started using caching mechanisms for enhancing their business, which results in increased load on the network. To overcome this growing load, Content Delivery Networks (CDNs) came into existence. Most of the cloud applications now use CDNs or the content delivery servers. These are nothing but surrogate servers that provide smaller content retrieval time to the end-user by caching highly relevant data using proximity [47, 50, 51]. Figure 1.8 represents the Content Delivery Networks (CDNs).

Technological advancements also paved the way towards peer-to-peer caching and storing

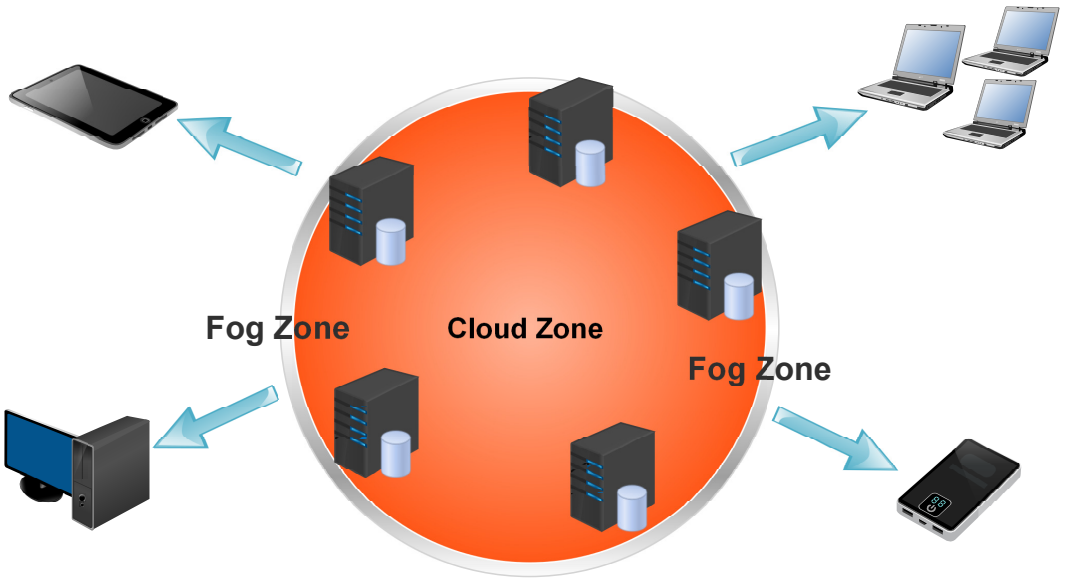


Figure 1.8: Content Delivery Network (CDN)

information for faster retrieval and green computing. For example, events such as 9/11 attacks in USA and flash crowds [52], accelerated the growth of caching technologies. With the birth of software-defined radios, videos and audio channels, the caching and content delivery technology have become an essential ingredient for construction of a responsive network. As the networks are expanding more towards remote locations such as oceans, mines, mountains ranges and deserts, there is a need for further innovation of caching content technologies at edges of network.

Transparent Caching is another technology which is gaining ground in the content delivery market [53, 54]. The caching technique has been added to existing versions of windows and other important operating systems as well. It is also a part of server farms and content networks. The mechanism is simply to cache the most frequently used data in some local source to act as an offline source.

The concept of content delivery networks came into existence due to the ever growing need to deliver quality content, not only in core network zone, but also in edge network zones. The properties of content has also been rapidly changing; the size and volume of files are increasing expeditiously. Their delivery requires considerable resources and orchestration. The popularity of the content dramatically changes the way a single or cluster of servers can handle the request. When an event such as olympic games are going on, the website holding the information of games is inundated with requests. The number of requests and types of content requested varies; the server might not be able to handle so many requests. To provide high quality of content to small devices such as mobile requires special arrangements. Hence, the need to manage content delivery becomes critically important. Content delivery network provides means and algorithms to manage a personalized inundate of content requests. However, rigorous regression testing and anal-

ysis is required for these technologies to be successful. Extensive experimentation and evaluation of empirical results is the correct way forward for advancing algorithms and test-beds. For such technologies, design of experiments (DOE) and profiling tools may be used.

1.5 Challenges of Fog Computing

Based on the findings of literature survey, following areas have been identified that require significant research attention.

(i) Load Balancing: It is a concept in fog computing which is very useful to achieve an energy efficient system. The research work proposed by Varghese *et al.* [55] contains heterogeneous nodes ranging from sensors to user devices to routers, mobile base stations, and switches. To perform general purpose computing on these different types of resources, both horizontal and vertical scaling is required.

(ii) Design of load balancer: Traffic management and load balancing have to deal with lot of uncertainty and inherent fuzziness of the traffic load. Limited work can be observed in current context that captures and maps the nature of traffic load using imprecise mathematics [56, 57].

(iii) Last mile optimization: Applications such as smart home, smart monitoring and rendering requires high degree of last mile optimization [58, 59]. Lot of attention is now been given to this area, but at the same time, lot of challenges too remain intact in this context.

(iv) Content management: The volume and variety of content has changed drastically in past decade. Users are accessing heavy bytes of video and other forms of data. The service level agreements have also become strict and tighter in their leverages. There is need to preemptively predict what the user's content likes and dislikes, hence, there is a challenge to manage traffic at user level [60, 61, 62].

(v) Responsiveness and Latency: There are many studies that point out the issue related to latency and responsiveness [63, 64]. Researchers talked about these challenges being faced by the users due to change in volume and variety of content being rendered at last mile of fog networks.

(vi) Scalability: The contemporary literature shows that in last decade lot of work has been delivered towards making the network scalable by adding new hardware at core zones of cloud. Limited work has been reported where studies give information on the scalability and availability issues in fog zone of the network [8, 65].

(vii) Resource provisioning: This may need to be carefully designed by using the priority model for better scheduling of resources [65, 66]. Resource discovery and sharing are critical for performance of applications in fog computing. The resource sharing optimization techniques can be formulated for maximizing the availability of resources and minimizing the energy consumption [8, 65]. Virtual machine (VM) scheduling needs a new design to provide an optimal solution for scheduling VMs [66].

(viii) Energy-Efficiency: Dastjerdi *et al.* [8] considered fog computing as a network consisting of a large number of fog nodes within which the computation is distributed. This can be less energy-efficient as compared to cloud computation model. The improved model can be proposed by taking energy optimization into consideration.

(ix) Quality of Service (QoS): This is an important metric for judging the efficacy of fog computing and it is judged considering the aspects as described: QoS can be improved by considering connectivity between all devices at optimized cost for data transfer and computations [65, 67]. The capacity of fog computing is determined by two factors: a) Network bandwidth for data transfer and b) Storage capacity of fog controller nodes. These factors affect the overall performance of the fog controller and need to be improved. In addition to these, data placement in fog and cloud needs to be looked upon for improving QoS. Yi *et al.* [65, 66] discussed that enormous delay can degrade the user satisfaction. Latency sensitive applications like complex event processing or stream mining must be handled by fog computing to provide processing in real-time streaming.

(x) Software Defined Network (SDN): SDN handles issues due to the reliability of wireless links and mobility of devices in the network and maintains connectivity. Issues like fog controller placement, communication between different controllers like constantly connected controller (at edge infrastructures) or intermittently connected controller (at end devices), and the designing of the distributed SDN system to meet the fog computing requirements can be considered [62, 65].

In order to balance the load and successful completion of work, it is necessary to find load balancing algorithms, approaches and models.

1.6 Motivation

Load balancing is no longer an act of simply distributing packets or jobs in a round robin manner nor it is a precise action to partition, divide or assign jobs or packets to a particular route or machine. The process of load balancing is not a simple process of stochastic computations to find best route. Now, there is a need to devise a heuristic or a self-learning

model that can work as an algorithm that has capabilities of self-learning the complexities of network traffic at a fine grain level. It is a process of balancing a complex traffic engineering scenarios that need minute-to-minute feedback due to inherent erraticness that reside in current networks. Even the optimization algorithms such as genetic algorithm (GA), particle swarm optimization (PSO), Bee Colony, Ant colony optimization (ACO), Honey Bee Foraging etc. that help to manage the traffic are showing that their results are approximate and not exact. This means the underline assumption to solve traffic engineering problems is that there is no precise solution but an optimal solution. All these optimization algorithms approximate the values of local and global minima/maxima. Hence, logically fuzzy mathematics is more applicable in context of managing traffic.
The related work points out to many challenges that are faced by the industry due to the issues of delay. It was found that the conditions in the cloud-fog network face a high degree of fuzziness due to multiple competing factors such as link saturation or erratic burst of data streams. This occurs, especially in cases where sensors are employed, which necessitates the reason to manage the cloud traffic load and fog load separately. The networks across the world have adopted over 100 Gigabit Ethernet (GbE) but the bandwidth as resource is still a constraint at the edges of the cloud network. This makes time-sensitive applications struggle due to which additional devices need to be deployed and separate management algorithms are required to manage the edge or fog devices. A case may arise when old and new elements of the network are working together, then the load analysis of each inter-node becomes important to monitor differently. If the cloud and fog network consists of old and new routing devices (new devices installed for scalability or deployed at the edge for better response time), the load analysis algorithms are unable to distinguish between 1 gigabit and 10-gigabit lines and utilization of the link is negatively impacted. Last, but not least in many cases it was found that there is a need for caching mechanism or data aggregation edge zone. This again compels to manage the fog zone with different approaches as compared to the cloud. Figure 1.9 represents the flow of the research work.

1.7 Objectives of Thesis

On the basis of literature review and research gaps, the following objectives are delineated and how each objective have been accomplished is included in this section.

i. *Objective 1: To explore and analyze the load balancing in fog computing to gain a systematic understanding of the existing techniques and approaches.*

   For the attainment of this objective, theoretical as well as practical review of load balancing algorithms has been done. This approach is exploratory and investigative in nature, which lead to formulation of the problem statement and defined the scope

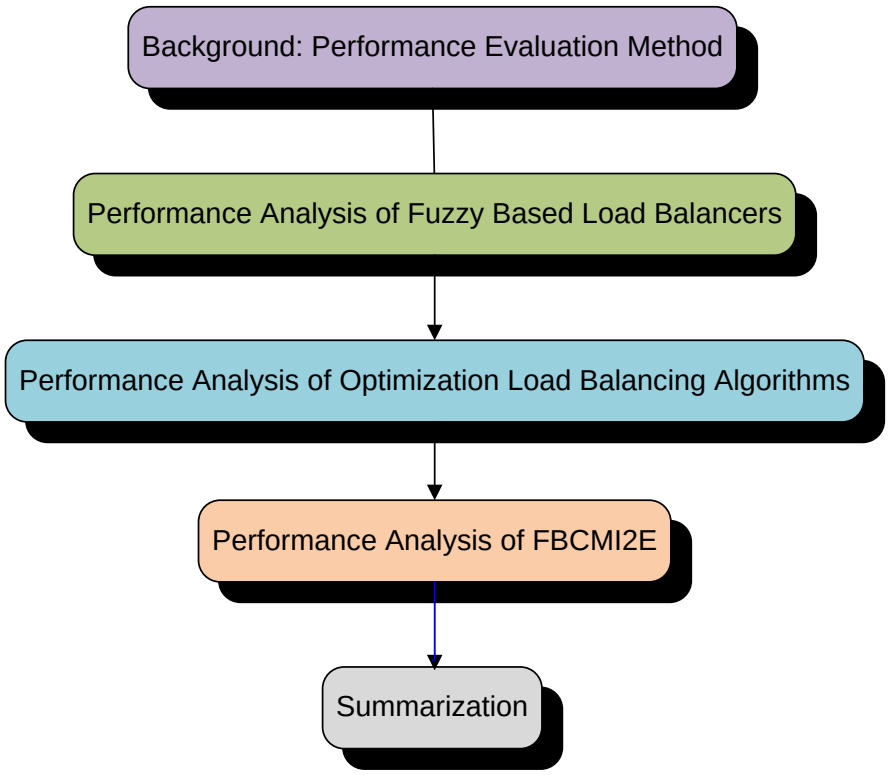


Figure 1.9: Evaluation Strategy of Research Work

of this work. Survey of tools, methods and algorithms has been carried out with the help of study material from reputed journals.

ii. *Objective 2: To propose an energy-efficient load balancing algorithm(s) in fog computing.*

This objective is obtained in two phases. In the first phase, fog load management algorithms have been designed. In this phase, various types of designs for managing fog zone load have been studied. Practical demonstration of the finalized design is carried out with the help of multiple simulations. The objectives of the design and demonstrated load balancers in the research work overcomes the issue of constrained capabilities of the fog devices. It was understood that the nature of fog load is different from the kind of properties a cloud network exhibits. The traffic at the fog zone has inherently higher levels of fuzziness due to last mile expectations of responsiveness. The QoS parameters in fog zone are tighter as compared to the core cloud zone traffic in many cases. It was felt that traditional approaches of finding best route for packets needs to be revisited and redefined in context of fog zone specifications. It was also felt that scaling of hardware is not a better option in case of fog zone devices, but use of fuzzy mathematics can help to overcome the current challenges. Therefore, a fuzzy logic based designs have been devised. The process of design of fog load balancer and system architecture are shown in Figure 1.10.

As shown in Figure 1.10, the fog load balancer design and its performance is eval-

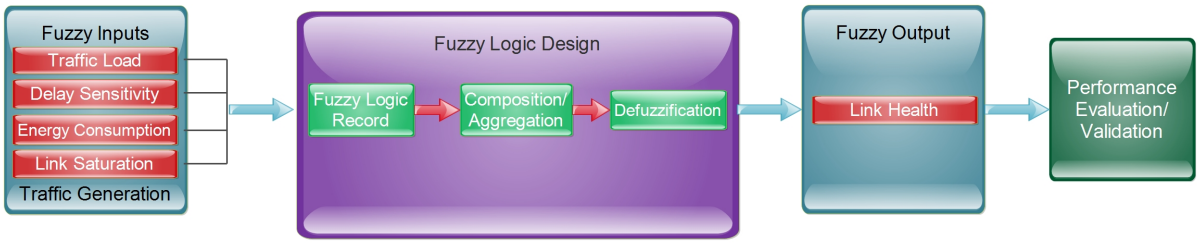


Figure 1.10: Block diagram of Phase 1

uated on the basis of three levels and testing is done for web, sensor and video data types of traffic. Multiple possible conditions and scenarios have been evaluated and each design deals with different levels (3-level, 5-level or 7-level) for intensive traffic. The fuzzy algorithm evaluates the health of the possible live links based on which the traffic load was split and routed. Candidate fuzzy rules are generated for managing the load balancing function and only those rules are considered for firing, if that hold some degree of impact (strength) on the process of traffic. Four parameters (traffic load, delay sensitivity, energy consumption and link saturation) are considered for constructing the rules.

From the phase 1, it is clear that the hardware scaling works well in case of managing traffic in cloud scenarios. Another way to improve efficiency in fog zone is augmentation of interconnects with temporary storage machines or Nano-Caches. Hence in this research, modified optimization approach of managing web, sensor and video data packets has been applied. In this phase, optimization methods such as Teaching Learning Based Optimization (TLBO), Stimulated Annealing (SA) etc. have been evaluated and an improved algorithm is constructed. Figure 1.11 can be referred for better understanding of the study done in this phase.

It can be seen in Figure 1.11 that the role of Nano-Caches is to help the fog zone network to become more responsive and reduce the overall execution time in fetching content. To do this and to acquire ability of scaling with reduced overhead, optimization algorithm called Modified Teaching Learning Based Optimization (MTLBO) has been devised and evaluated. It was found that this algorithm performance can be improved by changing the learning function. To further validate and compare MTLBO, SA algorithm based load balancer was also put under test.

The TLBO and MTLBO algorithms have two phases i.e. Teacher and Learner phase whereas the SA algorithm has all its functionality executed in single phase. These algorithms are given four parameters as input and executed with the objective function that minimizes the latency. Infact all the four parameters are combined to form composite matrix and on the basis of score, the best route and the most efficient Nano-Cache is found. The best route paves a way to find best performing Nano-Cache. The

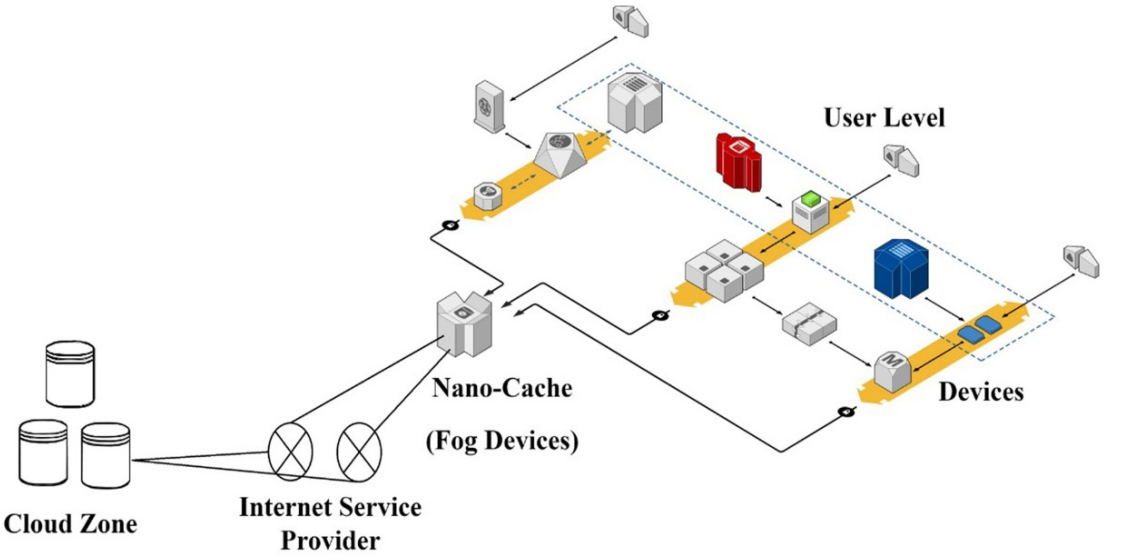


Figure 1.11: Block diagram of Phase 2

best performing Nano-Cache helps to reduce latency issues. Figure 1.12 represents the flowchart of optimization algorithms.

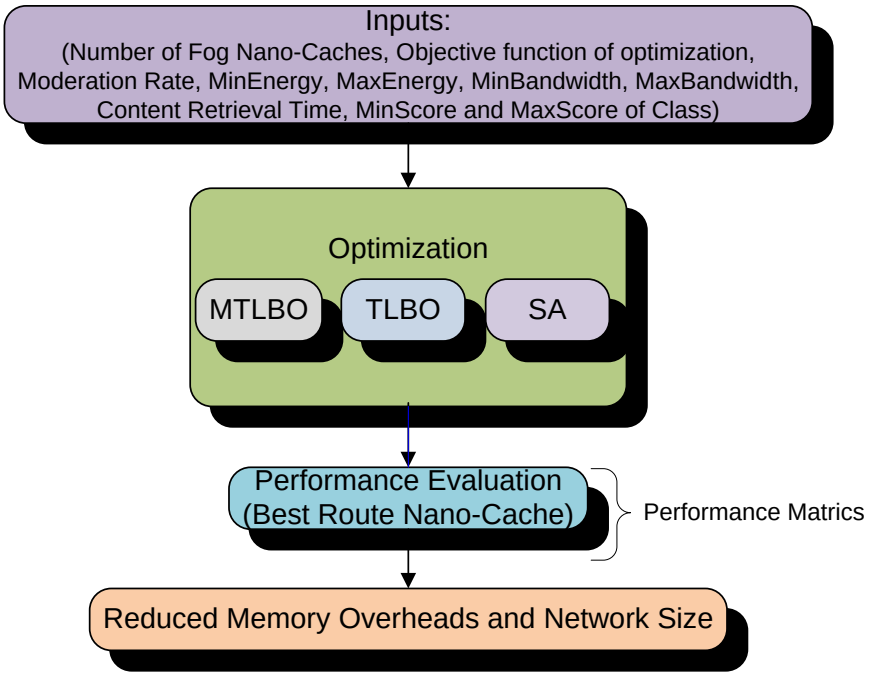


Figure 1.12: Flowchart of Optimization Algorithms

iii. *Objective 3: To implement the proposed algorithm(s) in simulated/real environment.*

To attain this objective, theoretical analysis of energy requirements for implementing

fog based networks is done. This phase includes work that demonstrates the working of fog based network made to collect multiple type of data and analysis of that data for a social cause. In this phase, the classic problem of finding right candidate for passing government benefits has been demonstrated using simulation process. The fog network collects four types of data. It includes fitness, geospatial, call log data and social inner circle data.

The collected data is analysed using structural methods and later on automated to find qualified candidate for government benefits. The automation of the process is done using supervised machine learning algorithms. The algorithms include Naive Bayes (NB), K-Nearest Neighbors (KNN), Decision Tree (DT) and Linear Discriminant Analysis (LDA). Figure 1.13 gives step-by-step information of phase 3.

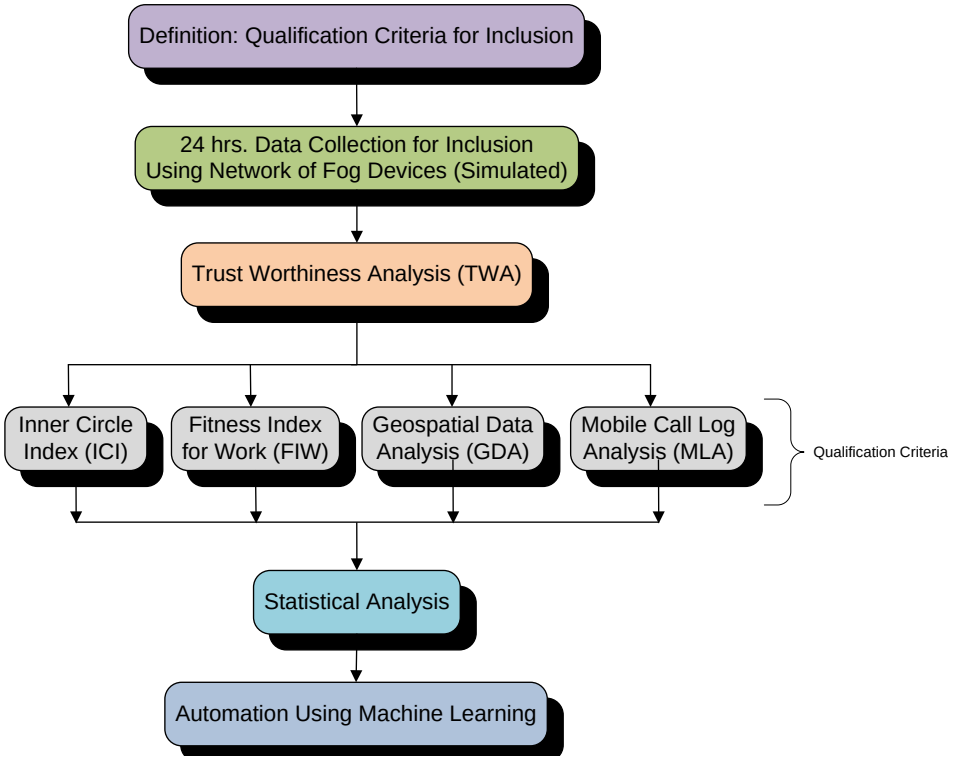


Figure 1.13: Workflow of Phase 3

iv. *Objective 4: To conduct a comparative analysis of different existing load balancing algorithms with proposed algorithm(s) in real/simulated fog computing environment.*

To achieve fourth objective, comparative analysis in each stage of the research was carried out. The phase 1 comprises comparison of three types of designs of fog load balancer, phase 2 consists of comparison of various optimization algorithms and phase 3 demonstrates comparison of data mining and machine learning algorithms for developing simulated fog computing application.

1.8 Organization of Thesis

This thesis describes the design of fog networks for load balancing, content management at the edge of a network, and application of fog network management algorithms for social cause. The thesis is organized into six chapters as described below:

Chapter 1: Introduction

This chapter discusses the research idea as well as problem formulation. The chapter also discusses ideas, terms, definitions, concepts, tools, and methods that are required for understanding the details of this research work. Discussion about evolution of various communication models that are required to run such arrangements is also included in this chapter. The basics of traffic management and load balancing in the fog zone are also elaborated in detail.

Chapter 2: Literature Survey [1]

This chapter insights the design issues, problems and challenges regarding load balancing algorithms in fog zone. This chapter contains three sections. The first section reviews the contemporary design and algorithms associated with the traffic management in cloud as well as in fog zone. The second section of the chapter focuses on methods that relate to the use of optimization methods and machine learning algorithms that help in overcoming traffic management issues such as congestion, load balancing and automation of availability and scalability functions. The last section elaborates tools for fog computing and investigates the application of fog zone technologies in social-economic sphere as well as its implications for the larger society.

Chapter 3: Design Levels for Energy-Efficient Fog Load Balancer [2]

This chapter focuses on the aspects related to the design of a load balancer for fog zone. The chapter discusses the evolution of the load balancers from initial days of overload and current methods of handling overload in the cloud. The chapter covers the conventional methods of load balancing and outlines their limitations. Furthermore, the chapter gives detail information on how these limitations can overcome by using fuzzy logic and Nano-Caches. Conventional designs of fuzzy-based load balancing algorithms are compared with higher-order/levels of fuzzy designs.

[1] Simar Preet Singh, Rajesh Kumar, Anju Sharma, Anand Nayyar (2020), *"Leveraging Energy Efficient Load Balancing Algorithms in Fog Computing"*, *Concurrency and Computation: Practice and Experience*, Wiley, IF: 1.148, https://doi.org/10.1002/cpe.5913, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5913

[2] Simar Preet Singh, Anju Sharma, Rajesh Kumar (2020), *"Design and exploration of load balancers for fog computing using fuzzy logic"*, Simulation Modelling Practice and Theory, Elsevier, vol. 101, May 2020, 102017, IF: 2.677, https://doi.org/10.1016/j.simpat.2019.102017, URL: https://www.sciencedirect.com/science/article/abs/pii/S1569190X19301480

Chapter 4: Optimization of Fog Load Balancer Using Energy-Efficient Algorithms [3]

This chapter highlights the use of content-centric networks. The use of conventional methods of improving responsiveness such as scaling of hardware has also been discussed. The chapter also covers the information in detail regarding the use of non-traditional methods of improving responsiveness in distributed systems. The current method advocates the use of Nano-Caches or tiny servers for reducing latency. The idea of using hardware (Nano-Cache) and optimization algorithms to scale up and reduce the latency worked well for optimizing the load in the fog zone. This has been established as a verified fact by doing a comparative study of three load balancing algorithms. An algorithm namely Modified Teaching Learning Based Optimization(MTLBO) was contrived and employed in fog zone to find an efficient route for forwarding contents using Nano-Caches and subsequently to improve content retrieval time. The mathematical distribution model of traffic was used and it was found that MTLBO is the best optimizer. MTLBO is compared with existing algorithms, namely, Teaching Learning Based Optimization (TLBO) algorithm and Simulated Annealing (SA) algorithm.

Chapter 5: Applications of Fog Computing Using Fog Load Balancer [4]

This chapter analyzes our research work applicability to a social problem. The chapter gives a graphical illustrations of the concepts of load balancing in fog computing that can be applied to the real-time problems. The chapter covers information on multiple types of fog devices, that collect different types of data to achieve a common social goal. The common social goal is to bring people living on the fringes of society into the main economy. The data collected by the fog devices are processed in real-time using data mining techniques to identify people who qualify for government benefits. Equipped with the discovery of new patterns in the dataset, the process of identification of the right person is automated using machine learning modeling.

Chapter 6: Conclusion and Future Directions

This chapter summarizes and concludes with respect to each step taken to achieve

[3] Simar Preet Singh, Rajesh Kumar, Anju Sharma (2019), *"Efficient Content Retrieval in Fog Zone using Nano-Caches"*, Concurrency and Computation: Practice and Experience, Wiley, 32(2), pp. 1-20, IF: 1.148, https://doi.org/10.1002/cpe.5438, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5438

[4] Simar Preet Singh, Anju Sharma, Rajesh Kumar (2020), *"Designing of Fog Based FBCMI2E Model Using Machine Learning Approaches for Intelligent Communication Systems"*, Computer Communications, Elsevier, vol. 163, pp. 65-83, IF: 2.816, https://doi.org/10.1016/j.comcom.2020.09.005, URL: https://www.sciencedirect.com/science/article/abs/pii/S0140366420319198

the aforementioned objectives. The chapter also highlights the novel contributions made in context of fog computing. This chapter also covers a section that highlights the sociological impact of current study. Subsequently, the last section of this chapter covers the ways and methods by which this research work can be further extended and meaningful contributions to the society can be made.

Chapter 2

Literature Survey

This chapter gives insights into design issues, problems and challenges regarding load balancing algorithms in fog zone. This chapter is divided into three sections. The first section reviews the contemporary design and algorithms associated with the traffic management in cloud as well as in fog zone. The second section of the chapter focuses on the use of optimization methods and machine learning algorithms that help in overcoming traffic management issues such as congestion, load balancing and automation of availability and scalability functions. The last section investigates the application of fog zone technologies in social-economic sphere and its implications for the larger society. Lastly, comparative analysis of literature work is given in tabular form. This chapter concludes with applications and tools that are used for implementation of fog computing scenarios in literature.

2.1 Fog Computing

The fog computing environment composed of multiple heterogeneous networks (fog nodes) that consist of the different configuration of devices. These heterogeneous networks communicate with each other to form the network of fog nodes. Fog computing is discussed by numerous researchers in their research.

Chiang and Zhang [62] discussed fog computing as a provider of various services at the edge of the network making them closer to the user. In their work, they explored new challenges and opportunities of fog computing and focused on how widely IoT network can use fog computing platform. They also discuss how various gaps in the technology can be filled with fog computing architecture, which leads to increase in new business opportunities. Latency constraints, network bandwidth constraints, resource constrained are some of the challenges for fog computing cited in their work.

Ivan Stojmenovic and Sheng Wen [6] discussed about privacy and security issues in fog computing. Their work involves man-in-the-middle attack and their research considers

Central Processing Unit (CPU) and memory consumption parameters.

Dastjerdi *et al.* [8] discussed about the emergence of fog computing along with its key characteristics. They presented a reference architecture for fog computing and discussed the recent developments and applications. They categorized various research challenges like resource management and energy minimization that are still in need for solutions in their work. They discussed about the IoT, commercial products including Cisco IOx (combination of networking operating system-IOS and Linux), Cisco Data In Motion(DMo) (provides analysis and data management at edge of network), LocalGrid (embedded software installed on network devices and sensors), Parstream (real-time IoT analytics platform). They also discussed the case study of a smart city, which is one of the key use-cases of the Internet of Things.

Kukreja and Sharma [68] discussed about the scalable distributed computing paradigms: fog, cloud, and dew computing. They explained dew computing as a novel structural layer in the available distributed computing hierarchy, which is placed as the base level for the fog and cloud computing paradigms. From the cloud computing, the various divisions: hierarchical, vertical and complementary to dew computing fulfills the requirements of low and high-end computing needs in day-to-day work and life. These novel computing paradigms decrease the cost and enhance the performance, specifically for applications and concepts i.e. Internet of Everything (IoE) and the Internet of Things (IoT).

Pranali More [69] defined fog computing as a model in which various applications along with the data was stored in devices which were placed at the edge nearer to the user rather than storing in the existing cloud. They described fog computing and the cloud computing's extension along with the various services of cloud computing extended to the edge of the network. They described the various features of fog computing and a case study along with the actual implementation of fog computing in traffic analysis to understand how it is applied in edge environment. They also discussed the differences between the cloud and the fog computing.

Mahmud *et al.* [34] discussed the fog computing environment along with challenges in fog computing. They discussed five types of fog nodes, their configuration and resource/service provisioning metrics. Security concerns in fog computing were explained in their work. Saharan and Kumar [70] discussed the main characteristics of the fog computing. They discussed the advantages along with the differences between the cloud computing and the fog computing and analyzed its applications for IoT. Their work concentrate on quality-of-service (QoS) parameters like delay and utilization of energy on the Internet, which is also possible for fog computing.

Yi *et al.* [65] discussed the unsolved cloud computing issues such as lack of mobility support, latency and location awareness and suggested fog computing as a solution to these issues. They discussed the fog computing definition and various issues that may arise during designing and implementation of fog computing and highlighted the issues

related to resource management, QoS, privacy, interfacing, and security.

Yi *et al.* [66] discussed the various challenges in fog computing platform and presented several application examples to explain the platform design like smart home, smart grid, smart vehicle, and health data management. A prototype for fog computing platform is implemented and evaluated for smart home applications.

Fog computing can also help other technologies to overcome their challenges. Chiang and Zhang [62] discussed about the IoT technology and various IoT challenges. Discussion on how the fog computing can help others to address various emerging IoT challenges are also elaborated by them.

As this research work is related to energy-efficient algorithms and load balancing, the next section describes the load balancing in fog computing.

2.2 Load Balancing in Fog Computing

In the era of computing, where most of the dealing is done with in streaming and out streaming of data, it becomes mandatory to check and guide the flow of data thorough desirable and efficient channels, so that it distributes the workloads across multiple resources used to handle or access data like network links, disk drives, smart gadgets, central processing units or computer cluster. The improved distribution is ensured by using a technique called load balancing.

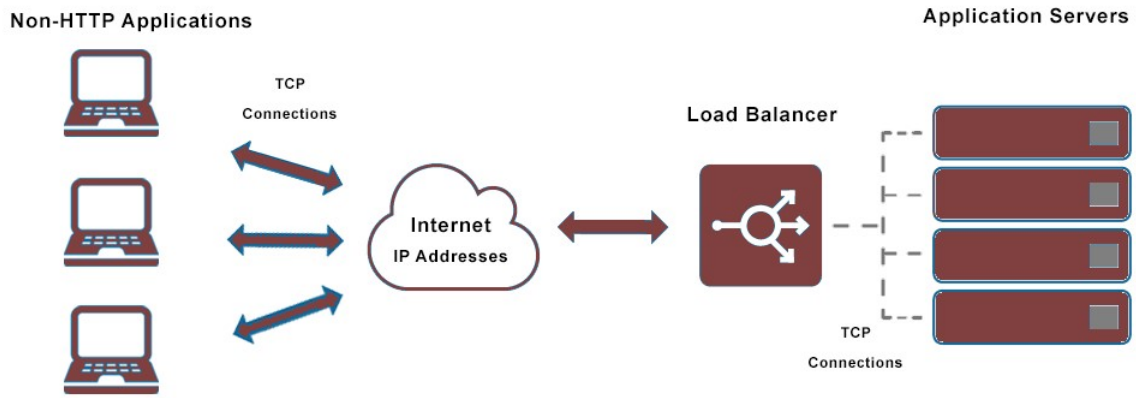


Figure 2.1: Representation of a Load Balancer

Load balancing is a technique used to optimize network efficiency, reliability and capacity. Load balancing performs many functions such as sending distributor client request to various networks, keeping in view the load across multiple active servers, it also ensures high availability, enables flexibility for addition or removing the servers. Load balancing is used to improve responsiveness and increases ability of applications. A load balancer is in-between the server farm and the client which accepts incoming application traffic

which distributes the traffic across group of available servers using different techniques. Figure 2.1 represents the load balancer. Figure 2.2 depicts the load balancing scenario.

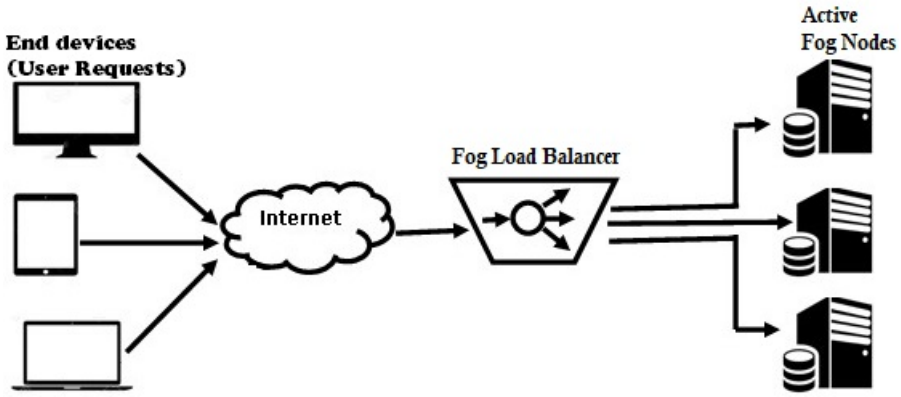


Figure 2.2: Load Balancing Scenario

There are various algorithms/techniques used for balancing the traffic load. Centralized load balancing is a technique in which there is a central node that controls all computations of load on the fog nodes in a distributed environment. Distributed load balancing is one in which there is no central node that manages the computation of each and every node in fog environment. The taxonomy of load balancing is discussed in the next section.

2.2.1 Taxonomy of Load Balancing

The term 'taxonomy' is defined as the branch of science that deals with classification. Thus, this section discusses about the taxonomy of load balancing in fog computing environment. The section lists the types of load balancers and explains their hierarchical relationship among them.
The behaviour of load balancers are elaborated in Figure 2.3:

(i) On the basis of state of cloud environment [71, 72]:

- *Static environment:* This comprises of homogeneous resources. Round Robin algorithm is an example of the static environment of load balancing.
- *Dynamic environment:* This comprises of heterogeneous resources. Load balancing min-min (LBMM) algorithm is an example of dynamic environment.

(ii) On the basis of type of load balancer [72]:

- *Hardware based load balancing:* This can route TCP/IP packets to various servers.
- *Software based load balancing:* This is a combination of application server and web server software packages. This is less costly and more flexible than hardware based load balancing.

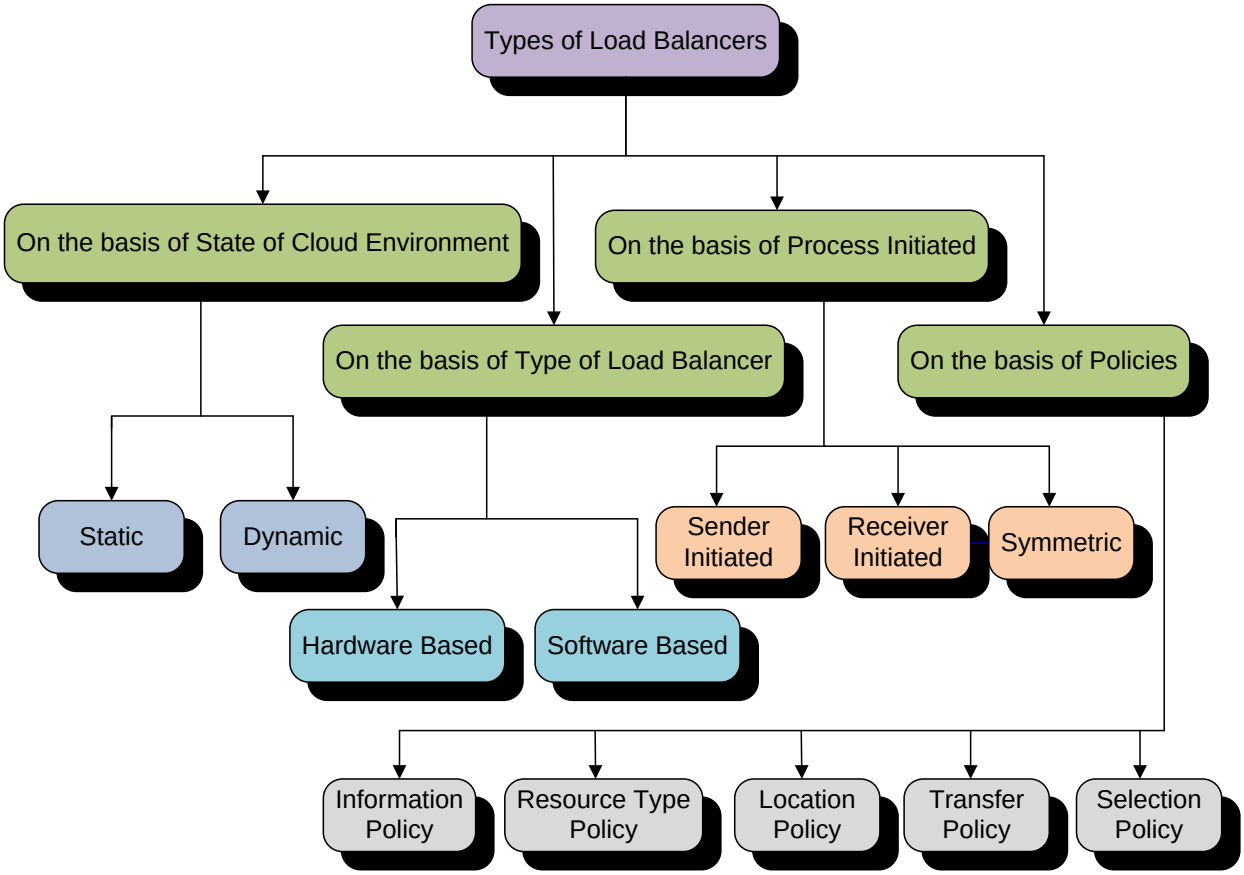


Figure 2.3: Types of Load Balancers

(iii) On the basis of process initiated [72]:

- *Sender initiated:* Load balancing is initiated by heavily loaded nodes which gather information about the load assigned to different nodes and according to the workload, a load is assigned to the low loaded node.
- *Receiver initiated:* Load balancing is initiated by low loaded nodes which gather information about the heavily loaded nodes and retrieve work from them.
- *Symmetric:* Load balancing is performed by coordination between sender and receiver depending on the conditions.

(iv) On the basis of policies [72]:

- *Information policy:* It defines what workload information is to be collected from nodes and from where and how.
- *Resource type policy:* In this, the resource is defined as either server or receiver of the process according to its availability.
- *Location policy:* It defines which destination node is to be selected for transferring the workload.

- *Transfer policy:* It defines which task is to be selected from local node to transfer to remote node.
- *Selection policy:* It defines which processors are involved in the load exchange.

2.2.2 Energy-Efficient Load Balancing

This section starts with the in-depth survey of energy efficient parameters, algorithms and applications to achieve the efficient fog load balancing in detail.

Lin and Shen [73] proposed a lightweight system named CloudFog for Massive Multi-player Online Game (MMOG) which uses fog computing for rendering game videos and streaming to reduce latency. For the enhancement of quality of experience (QoE), they proposed the receiver-driven encoding rate adaption strategy which increases the play-back continuity. PlanetLab real-world testbed and PeerSim simulator were used for their implementation.

Verma *et al.* [74] proposed Edge Balancing, an efficient load balancing algorithm for fog-cloud based architecture. This algorithm uses the data replication technique for managing the data in the fog network. They compared their proposed technique with the existing load balancing techniques. They used cloudsim 3.0 simulator for their implementation. A real-time efficient scheduling (RTES) algorithm for load balancing in fog computing was proposed by Verma *et al.* [75]. Cloudsim simulator tool was used for implementation in the fog environment. This proposed architecture can complete real-time tasks within their deadlines and increases the throughput so as to overcome growing demands of the end users.

Brennand *et al.* [76] presented FOX (Fast Offset XPath) for intelligent transport systems used in vehicular ad-hoc networks to detect congestion of traffic in vehicular ad-hoc network (VANETs). They used Network simulator: Objective Modular Network Testbed in C++ (OMNeT++) for their implementation.

Paya and Marinescu [77] discussed about energy-aware operation model. They mentioned that highest energy is consumed by the server. Thus they tried to increase the servers count operating in optimal regimes for energy reduction. They discussed scaling, types of scaling, server consolidation and five server operating regimes: optimal regime, two sub-optimal regimes, and two undesirable regimes. They proposed an algorithm for application scaling and load balancing along with many server consolidation policies. They used Amazon Web Services (AWS) for their implementation. Inter-clustering scaling was not addressed in their paper. They discussed other future challenges in the paper that need to be worked on.

A new architecture based on load balancing was proposed by Verma *et al.* [72] in a less complex and effective manner to meet the rapid rate of everyday increase in the number of end users. They discussed various approaches into which the load balancing

is categorized i.e. hardware based and software based load balancing. They mentioned various goals that must be considered while going towards load balancing techniques. They briefly explained various load balancing algorithms including Honeybee behavior inspired load balancing of tasks and genetic algorithm based load balancing strategy. To implement their proposed architecture they used modified Honey Bee based algorithm to balance the load. Their architecture resulted in better bandwidth utilization, reduced cost, fast access speed, maximum throughput and improved the computing needs of Internet of Things (IoT).

Load balancing schemes (static, dynamic, centralized, distributed, hierarchical and workflow-dependent) were presented by Katyal and Mishra [71] based on the service level agreements (SLA) requirements. They discussed three stakeholders of cloud (i.e. End User, Cloud Provider and Cloud Developer), their requirements and issues. For load balancing, they discussed two major tasks: resource provisioning and resource scheduling. They measured the effectiveness and efficiency of the load balancing algorithms using the cloudsim simulator. They also explained the comparison between various load balancing algorithms.

Kalra and Singh [78] presented a survey along with the comparision of various scheduling algorithms based on Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Genetic Algorithm (GA) meta-heuristic techniques. Vijindra and Shenai [79] also discussed various existing scheduling algorithms and compared few parameters with these algorithms.

Beloglazov and Buyya [80] discussed various VM scheduling and allocation methods along with their implementation in cloud environment using cloudsim simulator.

Vijaya and Srinivasan [81] discussed resource scheduling and its types. Scheduling technique was preferred when the request for the resources was large. They also discussed two levels of scheduling i.e. scheduling of tasks to virtual machine and scheduling of the virtual machine to physical machine. Consumption of energy, utilization of bandwidth, adhere to SLA agreements, load balancing, optimization of cost and techniques of virtualization were the metrics of their discussion. Scheduling was implemented to increase the performance and availability of the resource, minimize the cost and to reduce power. They discussed two types of scheduling algorithms i.e. static and dynamic. In their work, they defined Round Robin algorithm as a static algorithm whereas First Come First Serve (FCFS), ACO, honey bee algorithm and other particle swarm optimization algorithms are dynamic algorithms.

Sun and Zhang [82] presented a crowd-funding algorithm which integrates a spare dynamic pool of resources in the network. They presented a system structure based on a neural network of the human body in the form of characteristics of cloud and fog data centers. They discussed repeated game strategy for motivation and for supervising supporters to actively complete the tasks. This game was an infinite repeated game with no

final stage. They used Nase Equilibrium for calculating the results. Their goal was to get the highest capital return. They used basic framework of Hadoop system for their implementation. They detected SLA violation rate and time for completion of a task consisting of different task loads. They compared their algorithm with the Min-Min algorithm and Modified Best Fit Decreasing(MBFD) algorithm.

Agarwal *et al.* [83] proposed a cloud-fog architecture for scalable resource allocation to resolve resource allocation issues in fog environment. They discussed the cloud computing and its major challenges. They analyzed the obstacles that may occur during virtualization. They discussed the virtualization methods and their techniques. Usage optimization of resources, and minimizing response time and throughput maximization was the main motive of virtualization. They identified various issues, threats and attacks with virtualization.

Xiong and Xu [84] proposed virtual machine allocation algorithm for reducing the consumption of energy in data centers. The algorithm was based on Particle Swarm Optimization (PSO) and multi-resource allocation model. They used the total Euclidean distance as the fitness function, which is also the fitness function of PSO. They compared their results with Modified Best Fit Decreasing (MBFD) and Modified Best Fit Heuristic (MBFH) algorithms. They used cloudsim simulator for their implementation.

Another VM allocation technique was proposed by Saraswathi *et al.* [85] in which VMs are allocated to the user based on analyzing the characteristics of the job. Their main objective was to prioritize the low priority jobs (jobs having late deadline) than the high priority jobs (jobs having early deadline). They used dynamic allocation of VM resources for the user jobs within their deadline.

Fog computing environment consists of multiple fog nodes having virtual machines placed on the hypervisor of the physical machines. These VMs are migrated from one host to another to balance the load. Leelipushpam and Sharmila [86] discussed about the live virtual machine migration in the cloud environment. To balance the load, to achieve energy efficiency as well as to make the servers available for the new requests, they performed the live migration in the cloud data center. They discussed various categories of migration techniques i.e. load balancing migration techniques, energy efficient migration techniques and fault tolerant migration techniques with an example. Comparison among various migration techniques of load balancing was also discussed by taking different parameters into consideration.

Yi *et al.* [66] discussed the VM migration and the different ways of VM migration. Gupta and Mukhija [87] proposed an improved strategy for load balancing through virtual machine migration method. This method ensures that every node in the system will be utilized efficiently. They deal with two important matrices i.e. total migration time and service availability. They discussed the fog computing paradigm and described its characteristics. They also discussed various algorithms including load balancing and honey

bee behavior load balancing of tasks and the concept of migration and virtualization in fog computing.

Wang *et al.* [88] presented MOBISCUD, a traffic management model in the mobile network, which acts as a fast moving personal cloud. This architecture integrates technologies of the cloud and SDN together to form a standard mobile network. This task is performed in the backward compatible fashion. All the data is stored in the virtual machines which follow the user through the mobile network. For this movement, virtual machines use live migration and ensure low latency between the user and the cloud. They explained MOBISCUD by taking an example of a personal VM which moves with the user within distributed mobile network, making this architecture capable of supporting low latency applications. PhantomNet, an Long Term Evolution (LTE)/Evolved Packet System (EPC) testbed, was used to implement the prototype of MOBISCUD.

Bittencourt *et al.* [89] discussed layer of cloud that consists of cloudlets in the form of small clouds, that have low computing capabilities. They presented fog computing architecture that provides management components for operating the fog network. They focused on maintaining QoS by data migration of user among cloudlets. Jin *et al.* [90] discussed about live virtual machine migration approach. This approach uses checkpointing/recovery and trace/replay technology which provides LAN and WAN environments with fast and transparent VM migration. They implemented prototype which is based on ReVirt, which is a full-system trace and replay system. They compared their approach with XenMotion even in high-speed LANs.

Sabiri *et al.* [91] proposed method of virtual machine migration based on Architecture Driven Modernization (ADM). They explained the working of ADM. Their objective was to migrate processes from on-premise legacy system to the cloud environment without depending on their applications, features and cloud providers. They discussed the challenges in migration to the cloud. They compared the existing migration methods (CloudMIG, REMICS, ARTIST, Based on Services and PAAS Migration), based on the Model Driven Engineering (MDE) approach, and briefly listed the advantages and weakness.

Live migration of running Operating Systems (OSs) was discussed by Clark *et al.* [92] with liveness constraints in the data center and in heterogeneous environments. The concept of the writable working set was introduced and analyzed by them. They performed the designing, implementation and evaluated OS migration, which they built on top of the Xen VMM. Intel Xeon 2.4GHz machine was used for running Xen. Pre-copy approach was used for performing the live migration of the running OSes without stopping the running services on the machine being in migration.

Thus, from literature, energy is concluded to be the major parameter in devising fog load balancer. The next section discusses the various design and issues in fog load balancing.

2.3 Research Challenges to Design Load Balancer

The section discusses research challenges in designing fog load balancer. This section has been divided into two sub-sections. First sub-section discusses design issues in fog load balancing (Section 2.3.1) and second sub-section gives the comparative analysis of various load balancing algorithms (Section 2.3.2).

2.3.1 Design Issues in Fog Load Balancing

S. Kitanov and T. Janevski [42] discussed various possible design of traffic controllers considering fog to cloud, as a technological bridge. The authors recognized the need to incorporate a load balancing algorithm between various resources bounded by the interconnects and network links. They had proposed a quality of service (QoS) aware service distribution by deploying the nodes at the edges. The authors considered resource balancing as a multi-dimensional knapsack problem [42].

Zhou *et al.* [93] used simulation platforms to study the round trip time and transmission time when mobile fog computing infrastructure was put under stress. The paper gives information about the policy design for improving the response time with the help of feature comparison summaries and visual graphs. Fuzzy inference system has been demonstrated in mobile edge computing by the authors Zhou *et al.* [94] for the development of sustainable security service chain functions. The outcomes of the study was improved execution time, as compared to contemporary solutions. Wazid *et al.* [95] also used fuzzy logic to construct user authentication framework.

Nasrollahi and Kazem [96] conceptualizes the concept of resource discovery in large grid networks using fuzzy logic. The authors claimed that the use of fuzzy logic helps in real life conditions for the development of discovery services that are highly sensitive to humans in real life conditions. Thus, they made the sharing of resources very responsive in nature. Soleymani *et al.* [97] gave illustrations of trust model that worked on fuzzy logic mathematics. The model provided information on the fuzzy trust model between machine-to-machine. In the paper, imprecise nature of data between various vehicle nodes required fuzzy logic inherently.

S. Ningning *et al.* [56] worked on the logic of dividing or partitioning the graph into smaller graphs based on the factor analysis, that helped in balancing the utilization of the resources. These smaller components which match specific QoS parameters, were accepted and implemented. These graph partition problems fall under the class of NP-hard challenges. F. H. Rahman *et al.* [98] applied fuzzy logic mathematics to the concept of trustworthiness. Trustworthiness was computed with the help of a software defined component called fog broker. Its main function was to identify available fog devices that had good reputation and trust. The trust and reputation was computed based on feedback,

quality of service, security consideration and inputs from the load balancer about the current traffic scenario. The paper represents sketches, illustrations along with proof of concept (simulated) to demonstrate the working of the framework for load balancing.

According to S. Bitnam *et al.* [99], the traffic in the fog zone can be managed more effectively if bio-inspired algorithms were applied to job scheduling problems. In their paper, Bees Life Algorithm (BLA) has been applied to optimize the distribution of tasks in a network. The results claimed by the authors, mentioned that it outperforms Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) when execution time(reduced by 23-50 seconds) and memory is considered.

Y. Shu and F. Zhu [100] gave mechanism to offload overheads in 5G networks that had large fog/edge zone based node deployments. The authors used a term called Weak Load Balancing to demonstrate their algorithm in peer-to-peer association of fog nodes. They computed their offloading scheme based on minimization of multiple factors that helped to create balance between network quality and user experience. They proposed a multi-stage algorithm; In the first stage peer-to-peer connection was identified and the second stage executes the mobile sensing operations specific to the edges. Third stage mapped fog tasks with respect to available density of the nodes in the edge. The authors, with their algorithm, claimed that they were able to improve resource utilization and were able to optimize delay-duty cycle. This way, they improved overall efficiency of the system almost 87%. Similar research work was done by M. B. Kamal *et al.* [101] in their min-conflicts scheduling algorithm for solving Constraint Satisfaction Problem (CSP).

F.-H. Tseng *et al.* [102] discussed comparative analysis of four schemes that help to auto-scale fog platforms and traffic load balancing. All the schemes use custom made virtual network function to orchestra fuzzy based mechanism for auto-scaling. According to authors, their purposed scheme helped to reduce total operating expenses and increased fog network performance of three factors: responsiveness, reduced average delay and error rate. Similar work was carried out by X. Xu *et al.* [103] for enhancing the performance considering memory, bandwidth and CPU as resource parameters.

The scope of fog computing has been highlighted in recent studies that focus on upcoming technologies in 5G wireless networks [104, 105]. Green computing concepts, with the help of fog nodes, was referred in the contemporary literature [104]. S. Singh and R.K. Jha [105] mentioned about QoS and basis for determining the quality of the links for taking decision on maintaining correspondence between the radio nodes that were deployed at the edges. The proposed algorithm by D. Chen and V. Kuehn [106] benchmarked against the Quality of Experience (QoE) for traffic load balancing. The authors anticipated the need for load balancing, especially in cases where there were dense radio nodes working at the edges of the network [106].

2.3.2 Comparative Analysis of Load Balancing Algorithms

Comparative analysis of various algorithms given by different researchers is also presented in Table 2.1.

Table 2.1: Design based Comparative Analysis of Related Work

Ref. No.	Name of Author(s)	Year	Algorithm/Approach	Description	Parameter(s) Considered	Nature of Work
[95]	M. Wazid *et al.*	2019	Security model	Designed the secure key management, and authentication in fog network	Throughput, packet loss rate, cost and end-to-end delay	NS2 2.35 simulator and Ubuntu 14.04 LTS
[101]	M. B. Kamal *et al.*	2019	Algorithm	Presented Min-conflicts scheduling algorithm, a load balancing scheduling algorithm for solving Constraint Satisfaction Problem (CSP)	Cost, processing time, request processing time and delay time	Simulation using cloud-analyst simulator
[93]	Zhou *et al.*	2018	Policy design	Presented the policy design for improving the response time	Round trip time and transmission time	Simulated using NS2 and Matlab
[98]	F. H. Rahman *et al.*	2018	Algorithm	Trustworthiness was computed with the help of a software defined component called fog broker in fog computing	Arrival rate, time, number of fogs and service rate	Simulation using Matlab Fuzzy logic toolbox
[100]	Y. Shu and F. Zhu	2018	Algorithm	Discussed mechanisms to offload overheads in 5G networks that had large fog/edge zone based node deployments	Time, number of servers and number of mobile nodes	Simulation using OpenNet and Matlab

Continued on next page

Table 2.1 – continued from previous page

Ref. No.	Name of Author(s)	Year	Algorithm/Approach	Description	Parameter(s) Considered	Nature of Work
[99]	S. Bitnam *et al.*	2018	Algorithm	Applied bio-inspired algorithms to job scheduling problems for traffic management in fog zone	CPU execution time, cost, latency and memory size	Simulation using C++
[102]	F.-H. Tseng *et al.*	2018	Fuzzy-based real-time autoscaling (FRAS) mechanism	Discussed comparative analysis of four schemes that helped to autoscale fog platforms and traffic load balancing and presented fuzzy-based real-time autoscaling (FRAS) mechanism	Average delay, error rate and cost	Simulation using Apache JMeter and UnixBench
[103]	X. Xu *et al.*	2018	Algorithm	Enhanced performance considering memory, bandwidth and CPU as resource parameters	Number of fog services, number of computing nodes, number of node types, resource capacity, resource requirements of fog services and duration for each service	Simulation using Cloudsim

Continued on next page

Table 2.1 – continued from previous page

Ref. No.	Name of Author(s)	Year	Algorithm/Approach	Description	Parameter(s) Considered	Nature of Work
[94]	Zhou *et al.*	2017	Algorithm	Demonstrated fuzzy inference system in mobile edge computing for development of sustainable security service chain functions	Execution time, total processing delay, total CPU usage and Inverted Generational Distance (IGD) values	Implemented using OpenVSwitch, Nshk-mod, iptables, Linux Virtual Server (LVS) and nDPI
[97]	Soleymani *et al.*	2017	Security model	Illustrations of trust model that worked on fuzzy logic mathematics	Distance, Plausibility level (PL) and time	Simulation using Network Simulator (NS2) with SUMO and MOVE traffic simulator
[42]	S. Kitanov and T. Janevski	2016	Approach	Discussed the design of traffic controllers between the fog-to-cloud, as a technological bridge	Latency	Simulation
[96]	Nasrollahi and Kazem	2016	Approach	Proposed resource efficient searching using taboo table and fuzzy logic	Grid size, number of servers, number of nodes visited, server update and number of queries	Simulation using Matlab

Continued on next page

Table 2.1 – continued from previous page

Ref. No.	Name of Author(s)	Year	Algorithm/Approach	Description	Parameter(s) Considered	Nature of Work
[56]	S. Ningning *et al.*	2016	System model	Partitioned the graph into smaller graphs based on the factor analysis that helped in balancing the utilization of resources	Time, cpu, memory, input-output (IO), network and number of tasks	Simulation using Hadoop and JMeter
[106]	D. Chen and V. Kuehn	2016	Practical design of green network	Proposed Fog-Radio Access Network (F-RAN) based upon fog computing and benchmarked against the Quality of Experience (QoE) for traffic load balancing	Memory, bandwidth, number of users, total power consumption and cost	Simulation

Thus, design of any system effects the quality of service as well as has direct impact on performance. So, there is the need to choose the efficient design for selection of optimal fog load balancer. The next section discusses various optimization techniques that have discussed in literature related to fog load management.

2.4 Techniques for Optimizing Load Balancing

The section discusses techniques for optimizing load balancing in fog networks. This section has been divided into two sub-sections. First sub-section discusses optimization techniques for fog load management (Section 2.4.1) and second sub-section gives the comparative analysis of various optimization techniques (Section 2.4.2).

2.4.1 Optimization Techniques for Fog Load Management

Concepts related to Software-Defined Network (SDN) and Network Function Virtualization (NFV) were discussed by S. Yi *et al.* [65]. F. Al-Turjman [107] had focused on three types of caching methods: node functionality based caching, location based caching and content based caching. The paper in the later part shows the development of a fourth type of caching method named as Cognitive Caching approach for the Future Fog (CCFF), which was hybrid in nature. All of these caching methods were applied to different scenarios in various combinations.

F. Al-Turjman [107] had focused on multiple scenarios based on four performance metrics (age of data, popularity of on-demand requests, delay to receive the requested information and data fidelity) to come to the conclusion that there was a need for hybridization of these cache methods. Hence, the fourth method was proposed, called Cognitive Caching approach for the Future Fog (CCFF), which works in content-centric scenarios as well as in multi-functional/multi-layer/multi-hierarchical caching scenarios [107]. It was found that whenever there was an increase in the hit ratio, there was a need to develop a hierarchical cache system to split the load and transmission traffic. It was also found that the energy consumption was higher in cases where request type was on-demand as compared to periodic or emergency requests. This affects the overall reliability of the cache mechanisms. The degree of connectivity, and popularity of the specific content also impact the overall quality of delivery. Higher popularity and higher connectivity were always desirable for a successful cache mechanism. Another metric called percentage of in-network node equipped with cache was also studied by the author. It was found that higher number of caches in the sub-networks always help to improve the overall efficiency of the network. It was similar to the way of putting lot of Nano-Caches in sub-networks and edge networks [107].

S. M. Azimi *et al.* [108] managed fog cloud traffic load with two approaches. The

first approach towards load balancing was unaware of the available hardware. The second approach towards load balancing algorithm was aware of the fact that more hardware/resources were available. When more resources were available, the dynamic popularity of the content made the main criteria to become self-aware about the hardware resources [109]. In similar work, popularity factor was considered to put forward the traffic in a specific direction [110].

Smart Collaborative Caching means that the caching was done based on the algorithm that collect information from the available caches [111]. Based upon this information, algorithm decides whether to cache the content or store the content or perform no action on the content. The authors discussed the simple and complex content sharing mechanisms. The single content sharing mechanism was based upon the single circle of content request-response whereas the complex scenario was based upon many-to-many request responses. It was found that the concept of keys can be used to maintain the status of each cache to make the cache management smart. F. Song *et al.* [111] had proposed a collaborative caching scheme for assisting a network whose sole purpose was to provide information at a click of the button. The validation of their scheme was done on the basis of four experiments from which it was claimed that the Smart Collaborative Caching (SCC) algorithm and enhanced collaborative cache provide minimum latency and smaller content retrieval time.

Y. Hua *et al.* [112] discussed about dividing the cloud network into multiple levels called edge and semi-edge nodes. The paper illustrates that all networks and sub-networks in the cloud must have caches for routing and forwarding content at different levels. It also showed that by incorporating caches in the semi-edge and edge networks, latency can be reduced, especially for most popular content transmission.

F. Malandrino et at. [113] mentioned four types of architectures that must incorporate multiple caches to overcome the latency issues. These architectures include base stations, base station rings, aggregation layers and core layer cloud switches. These days Video-On-Demand (VOD) and other content are demanded by the users when they are mobile (eg. from their vehicles). This reason was mentioned by the authors behind these architectures. The authors developed a metric called the price of fog to judge the cache worthiness of content. By cache worthiness, the authors are trying to define a qualifying metric to decide whether to store content in caches or not.

Evidences can be found in many research papers [114, 115, 116, 117, 118, 119] about the use of zippers distribution to model and simulate popularity distribution for caches. The recent progression in the domain of Internet of Things (IoT) and edge/fog computing has forced many researchers to re-think about maintaining content retrieval time of the devices [120, 121, 122].

D. O. Mau *et al.* [123] integrated content centric network with caching scheme that works for mobile multimedia streaming and radio access networks. The congestion and trace

load was controlled by avoiding unnecessary pathways/routes and request. This was done by maintaining records of hit rate at various inter-nodes in the network [124]. A. Gonzfilez *et al.* [44] represented dual data cache scheme. This scheme promotes lazy caching and makes local prediction table that helps to maintain history and gives information for future requests and forward tasks so that traffic management becomes more effective. This ultimately leads to better load balancing.

Energy optimization is required at all levels of edge/fog networks as these devices may or may not have constant supply of energy/power. Guo-qing Wang *et al.* [125] have argumented the conventional cache policies with energy indicators for improving the energy efficiency of the network. The algorithm uses data structures in data plane of the content router, Content Score (CS) and forwarding information base for maintaining cache mechanisms. The results claimed by the authors show that average energy consumption improves with respect to cache size [125].

Contemporary literature also shows that there is an increasing trend of using distributed caches for improving content retrieval time in the IoT/fog networks [126]. Some authors have developed cache's solution using optimization algorithms such as Genetic Algorithm (GA) [127], Particle Swarm Optimization (PSO) [99, 128] and Simulated Annealing (SA) [129, 130]. The analysis of these results show that optimization technique is performing quite well in this field for improving video and audio streaming performances in the fog/IoT networks.

Lin *et al.* [131] considered the study of machine-to-machine and device-to-device communication network. The scenario demonstrated by the authors was about unmanned vehicles communicating with each other using Poisson Point Process (PPP) model. In the context of problems undertaken by the authors, the Poisson Point Process (PPP) models suggested that the positions of the device/machine/unmanned vehicle were distributed with infinite intervals, that has Poisson distribution. At the same time, the positions of the machines/devices/unmanned vehicles were distributed with disjoint intervals and are independent random variables. But still they follow some order and also have the characteristics of being a memoryless property of the distribution process. The outcome of the simulation showed that maximum popular content was most suitable in such scenarios. The authors finally proposed caching placement strategy based on Genetic Algorithm (GA).

Ma *et al.* [132] have drawn various kinds of inferences and conclusions by doing a simulated study of cache placement problem that has application in VANET. According to the authors, the proposed procedure followed by them produces better quality of experience in terms of maximization of caching gain and all this was done with the help of convex optimization technique called simulated annealing. Schlag *et al.* [133] in their paper claimed that they have conducted an extensive series of experiments with the help of partitioning edge zone area by using the concept of hyperbolic random graphs which leads

to better efficiency of the network. The authors have used parallel and split graph construction algorithm to demonstrate the effectiveness of their approach. Many researchers [130, 134, 135, 136, 137, 138, 139] used the simulated annealing algorithm for solving cache placement and improving the response time.

2.4.2 Comparative Analysis of Optimization Techniques

Comparative analysis of various algorithms given by different researchers is also presented in Table 2.2.

Table 2.2: Comparative Analysis of Literature Work

Ref. No.	Author(s) Name	Year	Algorithm/Approach	Parameter(s)	Type of Communication	Tools Used for Implementation
[131]	Lin *et al.*	2019	Poisson Point Process (PPP) Model	File size, transmit power and density	Machine-to-machine and device-to-device	Mathematical Model
[133]	Schlag *et al.*	2019	Algorithm	Time, vertex cut, number of instances and number of processing elements (PEs)	Node-to-node	ForHLR II
[140]	D. Joshi *et al.*	2019	TLBO for resource-constrained project scheduling problem (RCPSP)	Number of iterations, maximum schedules generated and population size. Objective: Minimizing total execution time	Device-to-device	Matlab R2008a
[141]	Elif Varol Altay and Bilal Alatas	2019	Algorithm	Number of iterations and fitness value	Device-to-device	Mathematical Model
[142]	J. Nayak *et al.*	2019	Approach Literature about TLBO from year 2011 to 2017	–	–	–
[143]	J. Kaur *et al.*	2019	Non-dominating sorting TLBO using group learning and learning from other experienced learners (NSGLTLBOLE)	Hypervolume (HV), generational distance (GD), inverted generational distance (IGD), spread and epsilon metrices	Device-to-device	jMetal Framework 4.3

Continued on next page

Table 2.2 – continued from previous page

Ref. No.	Author(s) Name	Year	Algorithm/Approach	Parameter(s)	Type of Communication	Tools Used for Implementation
[108]	S. M. Azimi *et al.*	2018	Approach	Latency, link capacity, cache storage and bandwidth	Edge-to-edge	Monte Carlo simulations
[110]	M. Enguehard *et al.*	2018	Least-Frequently Used Load Balancing (LFU-LB) Algorithm	Latency, cost, cache size, memory, service time, arrival rate and percentage of requests	Device-to-device	Packet-level simulator
[112]	Y.Hua *et al.*	2018	Semi-Edge caching (SE) Mechanism	Latency and cache hit ratio	Edge network communication	Information Centric Networking (ICN) simulator Icarus
[144]	M. K. Hussein and M. A. Mohammed	2018	Algorithm	Cache hit ratio, hit time, cache miss ratio, time of query and miss time	Device-to-device	Crawlers of Google-Bot, 4-shared system
[107]	F. Al-Turjman	2017	Cognitive Caching approach for the Future Fog (CCFF)	Age of data, popularity of on-demand requests, delay to receive the requested information and data fidelity	Device-to-device	Network Simulator (NS3)
[111]	F. Song *et al.*	2017	Smart Collaborative Caching (SCC) Scheme	Number of clusters, packet loss probability, total number of packets and average transmission latency	Device-to-device	–
[124]	J. A. Khan *et al.*	2017	Collaborative Content Caching Approach and Self-organizing Coalition Formation Algorithm	Cache hits, time slot and file size	Node-to-node	–

Continued on next page

Table 2.2 – continued from previous page

Ref. No.	Author(s) Name	Year	Algorithm/Approach	Parameter(s)	Type of Communication	Tools Used for Implementation
[132]	Ma *et al.*	2017	Simulated Anneal Algorithm	Hit ratio and latency	Node-to-node	Mathematical evaluations
[109]	Marat Zhanikeev	2016	Approach	Cache size, inter-cloud transfer count, volume (ratio of total content), log transfer count and ratio of inner/outer file transfers	Heterogeneous	–
[113]	F. Malandrino et at.	2016	Approach	Distance, price-of-fog, time and cache size	Device-to-device	WeFi app
[145]	D. Chen *et al.*	2016	Teaching-learning-based optimization algorithm (TLBO) with multi-classes cooperation and simulated annealing operator (SAMCCTLBO)	Diversity and generation (size)	Device-to-device	Matlab
[146]	R. Bellio *et al.*	2016	Algorithm	Cost and temperature	Device-to-device	Ubuntu Linux 13.04, Json2 run
[65]	S. Yi *et al.*	2015	Approach Discussed concepts, scenarios, issues and challenges	–	Device-to-device	–

Continued on next page

Table 2.2 – continued from previous page

Ref. No.	Author(s) Name	Year	Algorithm/Approach	Parameter(s)	Type of Communication	Tools Used for Implementation
[123]	D. O. Mau *et al.*	2015	Multi-Source Mobile Streaming in Cache-enabled Content-Centric Networks (MM3C)	Bandwidth, delay, jitter, buffer size, monitoring time interval and play rate	Device-to-device and machine-to-machine	OPNET
[147]	Zhang *et al.*	2015	Teaching Learning Bird Mating Optimization (TLBMO) algorithm	Population size, number of onlookers, tournament size, angle and distance	Homogeneous	Matlab R2010a
[125]	Guo-qing Wang *et al.*	2014	Virtual round trip time (VRTT) Approach in content-centric networking (CCN)	Average energy consumption and average hop	Device-to-device	Matlab
[148]	R. V. Rao and V. D. Kalyankar	2013	Algorithm	Number of iterations, cost, convergence rate and accuracy of the solution	Device-to-device	–
[149]	Babak Amiri	2012	Teaching Learning Based Optimization (TLBO) Algorithm	–	Homogeneous	Matlab 7.13

From the Sections 2.3 and 2.4, following points can be mentioned for better understanding between optimization approach and fuzzy mathematics for building load balancers in context of edge/fog computing.

(i) Optimization algorithms sometimes are not able to do full coverage of the search space. Due to which local minima and maxima are obstructed. In fuzzy logic, centroid of all points is computed, as a result some value will always come in the result.

(ii) During multi-objective optimization, the Pareto-optimal solution cannot be improved and the outcome is not dominated with reliable and feasible solution, especially when there are opposing objectives between the variables, whereas fuzzy logic is based on rules. For such cases, particular rule will not be fired in fuzzy logic.

(iii) Fuzzy systems are designed by experts in such a way that it supports execution of tasks that can be done with the help of non-fuzzy algorithms as well.

(iv) Fuzzy logic algorithms can be executed with the help of minimal resources and as compared to other conventional methods, its execution will be faster due to low overhead in firing fuzzy rules.

(v) Fuzzy logic algorithms have inbuilt mechanism to overcome any bias in the calculations. Thus, they are robust methods that can work with noisy and imprecise datasets.

Next section discusses tools that are used for fog computing.

2.5 Tools Available for Fog Computing

The section discusses tools that are available for simulating fog network scenarios. This section has been divided into two sub-sections. First sub-section discusses simulation and emulation tools for fog computing (Section 2.5.1) and second sub-section gives the comparative analysis of tools in fog computing (Section 2.5.2).

2.5.1 Simulation and Emulation Tools for Fog Computing

Various simulation tools for implementing fog computing environment were discussed by Kukreja and Sharma [68] as iCanCloud, OPNET, CloudAnalyst, Cloudsim, Network Cloud, Network Simulator 2 (NS2), OMNeT++, Open Cirrus, CDOSim and Xen Hypervisor.
Goyal *et al.* [150] discussed the working of cloudsim and its various versions. Modelling of different services and applications, and scheduling them on the cloud system was the

challenging task. They solved this issue by providing power to manage services and performing simulation with cloudsim simulator. They also discussed the layered architecture, working, and implementation of cloudsim.

Recently, a java-based fog computing simulator named iFogSim was designed by Gupta *et al.* [151] to model and simulate IoT, fog, and edge computing environments. iFogSim allows investigation and comparison of resource management techniques based on QoS criteria such as latency under different workloads. They described two case studies: latency-sensitive online game and intelligent surveillance through distributed camera networks. They demonstrated the effectiveness of iFogSim for evaluating resource management techniques including cloud-only application module placement, and a technique that pushes application towards edge devices when enough resources are available. iFogSim can perform simulations in context of IoT, fog and edge computing. Figure 2.4 shows general scenario designed in iFogSim.

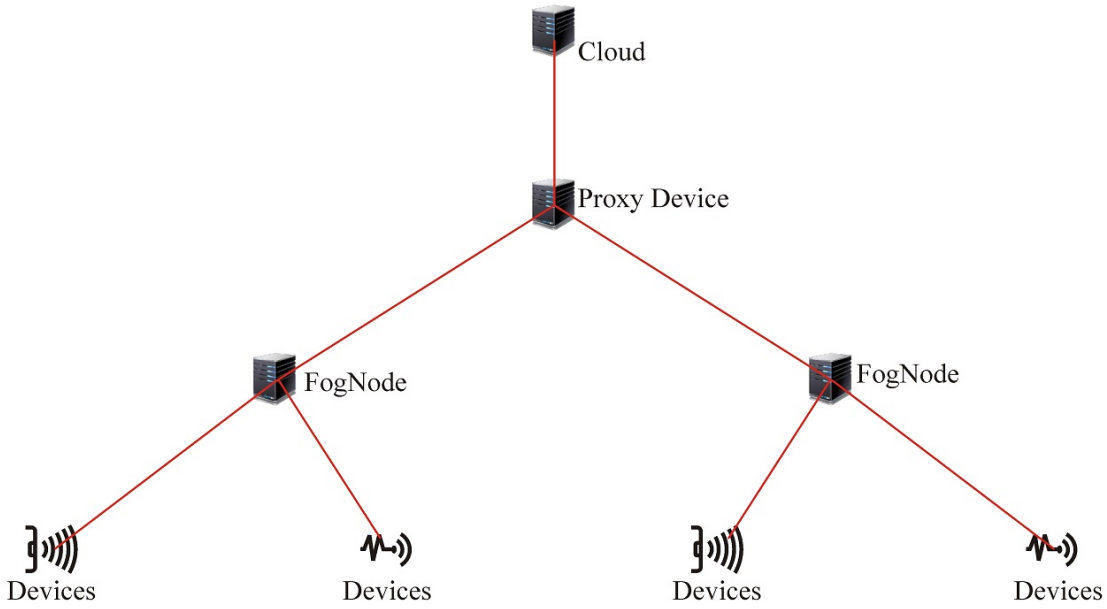


Figure 2.4: Scenario designed in iFogSim

There are various objectives of design that are implemented by EmuFog [152]. Large topology networks are emulated by EmuFog which allows the developer for studying the scenarios of fog computing that are in large scale. EmuFog makes implementation easier for the developers by making package of the applications and to run in the emulated environment. All the EmuFog components are replaceable and extensible with the help of the components that are custom build.

The workflow of EmuFog at the time of performing emulations of the fog computing scenarios are divided in four different ways:

(i) *Topology Generation:* BRITE is one of the topology generator which creates a network topology. This is the network topology which is loaded from file that also

allows the datasets to include with the topology of the real world.

(ii) *Topology Transformation:* In case of EmuFog, the network can be represented by an indirected graph and are connected with various links along with latency and certain throughput. The devices of networks are grouped with various autonomous systems. Thus, in this way, the topology can be translated into a specific model of topology of EmuFog [153].

(iii) *Topology Enhancement:* The topology network along with the fog nodes are enhanced. In this way there are two different steps that are performed in which, first, the edge in case of topology of network is determined and secondly, in the topology of the network the fog nodes are inserted which are done in accordance to the policy of placement.

(iv) *Deployment and Execution:* In the environment of emulation, the network topology, which is enhanced, is deployed. The fog nodes are also placed in the network, which are emulated within the application components, that are provided by the Docker containers, that are also deployed on the basis of fog nodes.

The developers of an application who are capable to evaluate the applications of fog computing in various environments of fog, uses the following ways. Different workflow steps in case of EmuFog can be implemented in accordance of the need of the users. EmuFog serves the huge set of emulation scenarios of fog computing that are useful in various implementations. A component to generate topology is also provided by the EmuFog for generating the topologies of their Internet scale. This is the topology that is based on the network of BRITE [154]. An adapter is also there which translates the topologies of BRITE network in the topologies of real world taken from CAIDA and ITDK, which is the model of network topology of EmuFog. These are various simple components of Emufog [155, 156, 157].
In case of EmuFog, a placement policy which is latency based, is implemented to keep a bounded latency that is between the clients who are connecting the edge of the network and the fog node (that is closest). Figure 2.5 depicts a scenario designed in EmuFog tool.

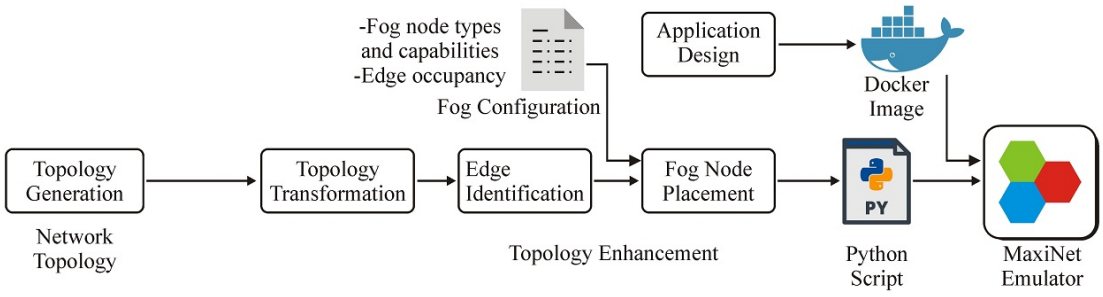


Figure 2.5: Scenario in EmuFog

Qayyum *et al.* [158] discussed about FogNetSim++, that was developed by extending OMNeT++, an open source tool, to provide simulation capabilities in fog computing. They presented booker node and fog node algorithms in their research. In addition to these, they compared their proposed simulator with multiple other fog computing simulators. Figure 2.6 shows the one of the scenario in FogNetSim++ simulator.

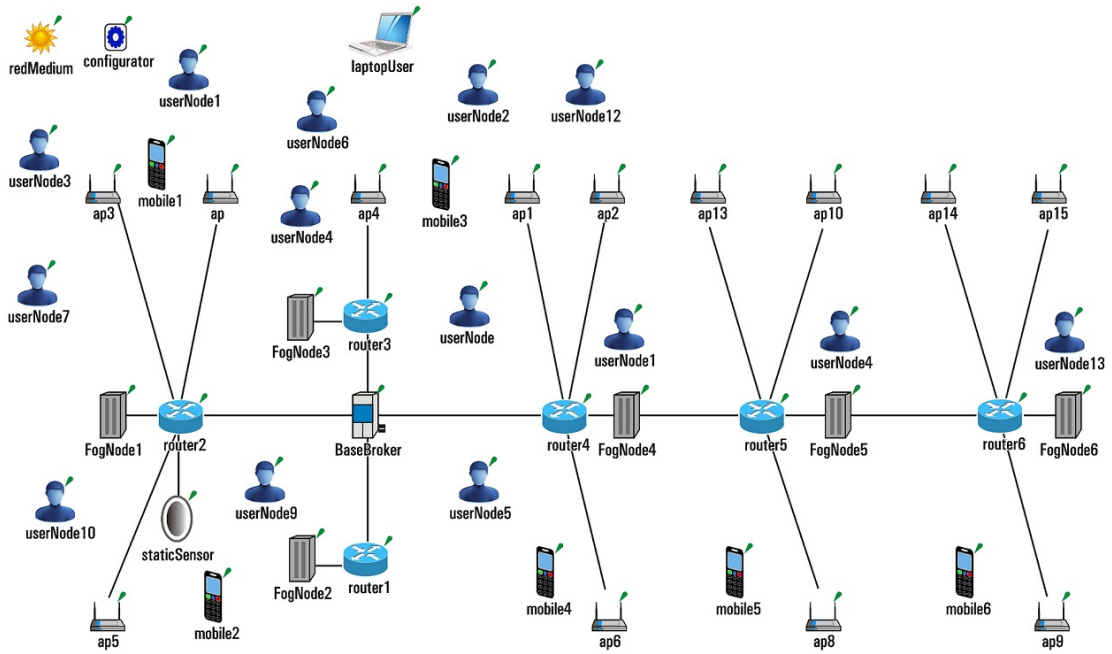


Figure 2.6: Scenario in FogNetSim++ Simulator

2.5.2 Comparative Analysis of Fog Computing Tools

The comparison of various fog computing tools is presented in Table 2.3.
Next section discusses about some of the applications that are available in fog computing.

2.6 Applications of Fog Computing

Bringing informal economy into the main fold has been the business of governments till date. However, with advent and use of new technological innovations such as smart watch, smart wear and smart bio-sensors etc., people from the different socio-economic stratification can be included in the formal economy.
The current technologies promise great deal of prosperity and inclusiveness for people who are currently out of main stream economy [159]. At the same time, the current literature portrays a debate on who should be the real beneficiaries of so-called inclusive technology. The current research on the topic shows that the label of informal economy is quite misleading, this is perhaps due to the fact that it was never a beneficial concept [160]. The concept of need economy is arguably better as it reflects both the so called

Table 2.3: Comparison of Various Fog Computing Tools

Simulators/ Emulators	**Use of Programming Languages**	**Based on Platform**	**Available as Open Source**	**Mobile Node**	**Mobility Models Customization**	**Scheduling Techniques**	**Handover of devices**	**Energy component**
MobloTSim simulator	Java-based	Linux-based	✓	Present	×	×	×	×
SimpleloT simulator	Java-based	Unix-based	×	Present	×	×	×	×
BlueMax, simulator by IBM	Java-based/ Python-based	Cloud-based	×	Present	×	×	×	×
Google IoT Sim simulator	NA	Cloud-based	×	Present	×	×	×	×
iFogSim/ MyiFogSim simulators	Java-based	All platforms	✓	Present	×	✓	×	×
Cooja simulator	C language	Linux	✓	Limited in Number	×	×	×	×
FogTorch simulator	Java-based	All platforms	✓	Not Present	×	×	×	×
RECAP simulator	N/A	–	N/A	Not Present	×	×	N/A	N/A
EmuFog tool	Python-based	All platforms	✓	Not Present	×	×	×	×
EdgeCloudSim simulator	Java-based	All platforms	✓	Present	×	✓	×	×
FogNetSim++ simulator	C++ language	All platforms	✓	Present	✓	✓	✓	✓

informal economy's interconnection to the global capitalism as well as nuances of its perceived empowering potential. At the same time, the need economy would potentially not be able to capture the complex nuances of real life problems [160, 161].

So, what should be the definition of need economy? Is it a sector that comprising of tiny, small and enterprises or something else. Empirical studies from the different parts of the world show that with the introduction of mobile telephony, life of many people has changed [162]. This is evident from the countries such as India [163], Indonesia [164], Philippines [160], Uganda [165], and West African countries [166]. Adoption of mobile banking changed the dynamics of street market (A place where information economy is visible enough) [167]. C. W. Larsson and J. Svensson [165] discussed the role of mobile technology in transforming the well-being of the people in Uganda. The concept of Science, Technology and Society (STS) program gives details on transformation and empowerment of small business transactions in informal economy with the use of mobile telephony. The paper discusses contradictory assumptions taken by the government for introducing mobile telephony for transaction and real acceptance level of people of Kampla, Uganda [165].

Traders in the formal building are required to have formal licences and these need to be stationary at place for some substantial amount of time [168]. This gives a chance for them to improve their inclusiveness factor in the economy. Now, the small traders can be tracked and trust models can be constructed for their qualification for formal economy [169]. People who are permanently disposed (peasants and petty producers) can be absorbed into the urban grind with help of mobile and other inclusiveness technologies. These technologies can overcome the problems of Informal economy that are characterized by temporary, uncontrollable and in-congruent.

Evidences can be found in current literature that in most of the countries, the welfare schemes are not efficient and have high order leakage. Identification of poor or the target person, who needs support, usually turn out to be complex, controversial and is characterized by high levels of corruption. To overcome such issues, a smartphone and sensors based monitoring system can help. Even, the people who are stigmatized by society like waste/garbage collectors, scavengers, animal skinners and sex workers can be included to change their paradigm of life.

On the technology front, the inclusive technologies are not without problems and challenges [170]. There have been augmentative series of debates for wearables, application of sensors in medical industry etc. [171, 172]. The advancements in wearable technology open doors to support and monitor people from distance. This raises questions of ethics as well as questions of comforts for many people. The current ecosystem is surrounded by debates on its quality of sensor data it produces: whether this data can be used for medical, insurance and for inclusiveness is rigorously debated in the circles of researchers.

Fog devices in the market promise to improve health/fitness with little scientific evidence.

As such, using these devices and data for forming drugs is risky but it can surely be used for record keeping of the status of health/fitness by upgrading these to standard medical devices. Such developments give rise to new applications in economy as well. The companies can track their subject's behavior, habits and compute risks. According to Vilela *et al.* [173], the informal economy requires health and life insurance more than the main stream segments of the society for obvious reason. They are more vulnerable and are at the mercy of entities that control the growth of the economy. The mobile commerce can get further boost from micro-finance sector and other informal economy segments of society [167].

The basis for such developments is propelled by the facts that trust based on financial transactions should not be the only reason and there are many alternative of computing goodwill, reputation and trust [174]. The components of 'trust' can be derived from the analysis of data collected from sensors and smartphone type of devices. The geospatial data of a person can give insights into mobility factor of the person. The mobile call data can give information on the inner circle of the person. The health sensors can give information on the fitness level and habits of the person. All these components can form the basis of computing the reputation or trust of the person. Hence, use of such alternative approaches can help to construct models of trust [163, 175]. In Indian case: "The characterization of the India specific inclusiveness" needs to be understood in terms of present day context.

The key objective of this research is to propose and implement fog technology based inclusiveness program for segments of people who are permanently outside the main circle of the economy [176]. Use case scenario approach with illustration/simulation of inclusiveness program using fog devices has been carried out. The data related to fitness, social grouping, calls and geospatial data has been used for qualification to the inclusiveness program rather than using financial criteria for qualification. Application of algorithms for design and implementation of classical problem i.e. 'Identification of individuals that qualify for the inclusive program' has been done. The second objective is to develop understanding the bottlenecks for implementing such concepts and offer solutions in terms of technological viability in context of energy usage.

Chapter 3

Design Levels for Energy-Efficient Fog Load Balancer

In this chapter, design of fuzzy based fog load balancer along with its working have been proposed. This is a layered approach. The various observations that are considered for designing an energy efficient fog load balancer have also been discussed in this chapter. Various design scenarios for our research work have been taken into consideration like 3-level, 5-level and 7-level fuzzy inputs. The evaluation and validation of various design levels have also been done and elaborated in detail.

3.1 Fuzzy Based Fog Load Balancer

This section briefly describes the three phases: input, process and output of fuzzy based fog load balancer, as represented in Figure 3.1.

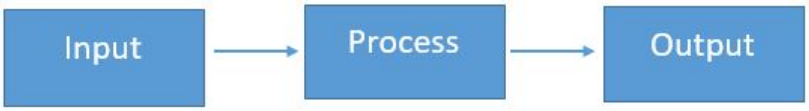


Figure 3.1: Phases of Fog Load Balancer

(i) Fuzzy inputs: This is the first phase of fuzzy based load balancer. The inputs that are entered into the fuzzy system are discussed below.

- *Fuzzy variables:* Fuzzy inputs are entered into fuzzy system in the form of fuzzy variables. For this research work, four fuzzy variables i.e. traffic load, delay sensitivity, energy consumption and link saturation have been considered.

- *Traffic distributions:* The traffic in this research work is mapped to various trapezoidal distributions considering the state of the traffic load on the fog network. These are discussed in Section 3.1.1.
- *Fuzzy levels and boundaries:* Fuzzy levels comprise of various states into which the fuzzy system is divided. This basically defines the design of the system model. We have considered 3-level, 5-level and 7-level designs for implementation of energy efficient fog log balancer.
- *Fuzzy membership functions:* These are the fuzzy conditions on the fuzzy variables that are represented as fuzzy membership functions in proposed system model.

(ii) Fuzzy processing: This is the second phase of fog load balancer, in which processing of the fuzzy variables based on fuzzy membership functions are taken into considerations with the help of fuzzy rules, fuzzy inference, as listed below.

- *Fuzzy rule:* Each rule comprises of all fuzzy variables and the conditions imposed on them.
- *Strength of fuzzy rule:* It is a measure which describes how much it effects the fuzzy process.
- *Fuzzy inference:* Fuzzy inference is a process in which the input variables along with set of logical operators forms set of rules to construct membership functions. These membership functions are assigned to the fuzzy output for producing meaningful information from the inputs.

(iii) Fuzzy output: This is the third phase of fuzzy based load balancer that deals with the output, which is obtained after the processing phase of the fuzzy based load balancer. This phase determines the strength and number of rules designed for this work. In this defuzzification and fuzzy interpretation have been considered.

- Defuzzification: Defuzzification is the process of getting crisp or numerical value of the fuzzy based load balancer.
- Fuzzy Interpretation: It is a set of logical inferences that can be derived from different fuzzy states. These states are defined by input vectors, set of rules, membership functions and the output vectors.

3.1.1 Traffic Distributions for Fog Load Balancer

To achieve the optimized traffic, a fuzzy based design of fog load balancer is considered. Following type of traffic distributions are studied for designing fuzzy based fog load balancer.

(i) The shape of trapezoid distribution resembles traffic load scenario in which initially the traffic load increases from lower bound 'a' till upper bound 'd' where $a < d$, beyond which no traffic event occurs (there is zero probability of new traffic coming at d). In addition to these two sharp bending points, the points 'b' and 'c' represent a uniform rate of incoming and outgoing traffic such that $a \leq b \leq c \leq d$. This is a scenario in which initially the traffic monotonically increases to a maximum level and then it plateaus (approximately linearly). Finally, the traffic load reduces and reaches to the initial level a.

(ii) The ramp distribution is a special case of trapezoidal distribution where initially the traffic increases steadily to reach a maximum point and then suddenly drops back to the initial point. The side ramp may not be perfectly linear but the values steadily increases or decreases and achieve maxima or minima to finally drop to the lowest level.

(iii) The triangular distribution is a case when $b = c$ where $a \leq c$, $c \leq b$. The value of c represents the mode, a is lower limit and b is upper limit. The triangular distribution in our research work simulates a traffic scenario in which after increase in load, there is some inflection point where either the load decreases or the performance of the network throughput decreases.

(iv) The S-Curve Shaped distribution and Sigmoid distribution simulates a traffic condition in which there is an increase in traffic load initially and in-between there is a slow down. Later on, the traffic increases to a higher level.

(v) The Gaussian distribution simulates the condition when the traffic increases in such a manner that it forms a bell-shaped curve. The traffic finally reaches the maximum point and starts decreasing slowly towards the original state. In double Gaussian distribution, surge occurs twice. These distributions represent a traffic condition when rate of increase and decrease are approximately similar.

It can further be observed that delay sensitivity parameter is directly proportional to the traffic load. Whenever significant changes happen in the values of delay sensitivity parameter, traffic load values change accordingly. This change leads to similar change in distribution pattern of traffic load parameter. When the traffic increases monotonically upwards, the delay is higher. If the traffic runs linearly at uniform rate, the behavior of delay sensitivity matrix shows the similar behavior. To implement the shape of distributions with various parameters, a system model is devised for designing the fuzzy based fog load balancer, as discussed below.

3.2 System Model of Fuzzy-Based Fog Load Balancer

Traditionally, there are two ways to balance traffic load in networks. First, by sending the data on different links (traffic data splitting) and second is to use additional hardware to handle the overload. In software approach, dynamic thresholds are computed either by using heuristic search or by using stochastic algorithms to do traffic splitting for load balancing. The purpose of putting edge/fog infrastructure gets defeated if the energy/power management and latency remains an issue due to imbalance utilization of resources. These problems get more pronounced due to unpredictable traffic patterns, which lead to erratic and unwanted changes in the energy/power management and latency of the network. Experimentally, it is also proven that algorithms working on global settings may not be sensitive to the traffic conditions at far end points. Hence, leading to conditions in the algorithms that are numerical unstable or hard to realize. The reason is that the edge network acquires a property of 'fuzziness' [177, 178]. The state of fuzziness needs to be addressed using software defined components that controls the resource utilization.

The concept of Software Defined Network (SDN) [105, 179, 180, 181] allows us to implement any device to act as a data aggregator, power/energy manager, data forwarder or a device that allows sending and receiving of data (gateway). It reduces the need to put specialized routing hardware at the edges of the cloud. A datacenter, typically has specialized routing servers called Ingress and Engress, that act as monitoring and analytical device for traffic management. They are based on MPLS/GMPLS (Multi-protocol Label Switching/Generalized Multi-protocol Label Switching) technology. Both these technologies are useful in high speed optical communication networks. However, this research work focuses on devices that can act as intelligent hotspots for sending and receiving data, which are energy efficient in nature. These devices might have constrained memory and computing power and may have operations other than routing also. These devices are normally used in wireless networking of fog devices that ultimately sync their data with cloud servers. Figure 3.2 shows the system model of fog load balancer.

3.3 Proposed Design Levels for Fuzzy-Based Fog Load Balancer

In literature survey, various challenges for energy efficient fog load balancer have been found which include issues related to the energy constraints, problems related to response time, deployment locations of the devices and the most important challenge is maintaining high degree of synchronization with the cloud servers. Hence, this research work focuses on investigating the impact due to traffic load factors, traffic arrival patterns (distributions

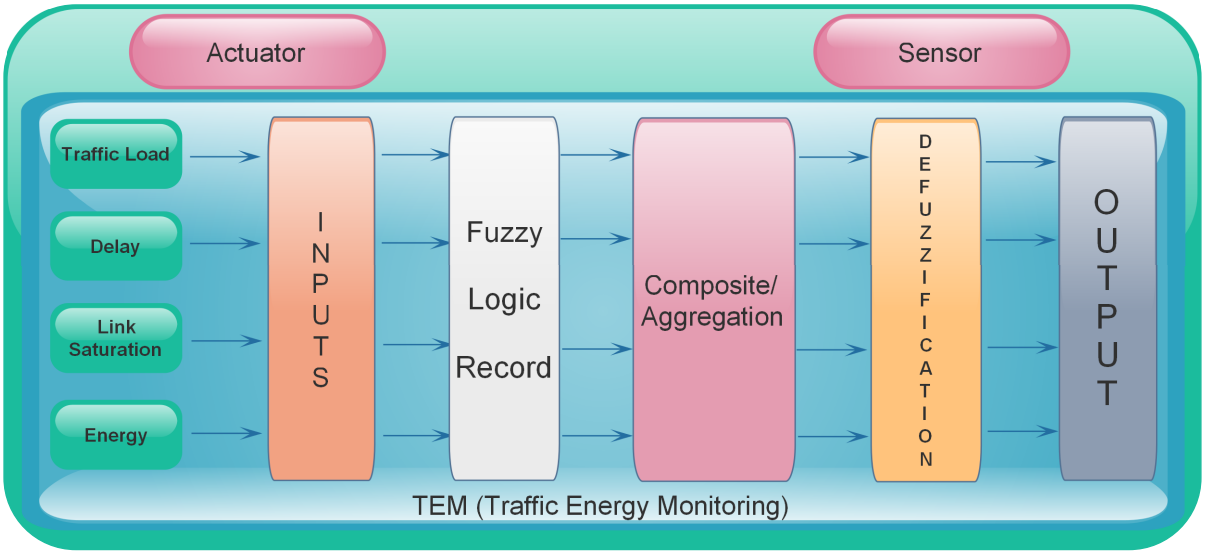


Figure 3.2: System Model of Fog Load Balancer

of data) and type of traffic (video, web, sensor data) in the edge/fog network. The analysis encompasses the design of the Rule Based Fuzzy algorithm that would help to control the imbalances that occur in terms of network utilization [182, 183].
This section discusses the proposed design, as mentioned in Algorithm 1, that is based on fuzzy logic. Next section gives details on the steps involved in design and fine-tuning of the fuzzy logic controller for the traffic load balancer.
The pseudo logic of the proposed algorithm is represented in Algorithm 2.

3.3.1 Design and Implementation of Fuzzy Based Fog Load Balancer

This section discusses the process of building fuzzy load balancer framework, that works for formulating the new tasks, payload and commands in edge, access, core, and datacenter sub-networks for better energy/power management of the network. Well-established approaches are followed to develop a simulated environment for edge, access, core and datacenter infrastructure right from the design of the network to the algorithms involved in traffic management and balancing. Various kinds of resource utilization parameters have also been explained. For this, we have used jperf simulator and fuzzylite api for implementation [184, 185, 186, 187, 188]. Simulation using Network Simulator (NS3) has been carried out in Ubuntu 18.04 [189]. The execution steps involved in simulation are as follows and screenshots of the same has been shown in Appendices - Figure I, Figure II and Figure III.

Execution Steps involved are as follows:

(i) Install operating system on machine (in our case Ubuntu 18.04).

(ii) Install network simulator (NS3).

Algorithm 1 Proposed Algorithm of Fuzzy Based Fog Load Balancer

1: Let L_x be a matrix that defines the quality of links.
2: Let N_x be number of total nodes in the fog zone.
3: Let C_f be the number of cloud-fog and cloud-mobile interconnects.
4: Let H_x be the matrix representing the health of the link L_x in equation 3.1.

$$H_x = \{Energy, TrafficLoad, Delay, LinkSaturation\}. \tag{3.1}$$

.
5: Observable mathematical relationships: The Fuzzy logic rules depend on the mathematical relationship between the various influencing factors as depicted in matrix 'H_x'(equation 3.1). It is abundantly clear that the 'PW' energy/power consumption is directly proportional to the workload/traffic load and at the same time is inversely proportional to the bandwidth. Higher bandwidth means lower consumption of the energies. Therefore, Power/energy consumption $\propto \frac{1}{Bandwidth}$.
6: High Throughput can be obtained if higher volume of bandwidth is available at any given time. Higher degree of free bandwidth in the network would lead to higher value of throughput. Therefore, Throughput $\propto$ Bandwidth.
7: High the current throughput and high degree of free bandwidth means lows latency. Which signifies that Latency $\propto \frac{1}{Bandwidth}$ and it has similar relation to throughput i.e. Latency $\propto \frac{1}{Throughput}$.

(iii) In NS3, we created a project and modules of fuzzylite. This will prepare exporters for C++ and other languages.

(iv) In *.cc file extension, we embed the fuzzylite C++ coding, which comprises of the following:

- Fuzzy input variables (TrafficLoad, DelaySensitivity, EnergyConsumption, LinkSaturation)
- Fuzzy output variable (LinkHealth)
- Linguistic terms
- Fuzzy Controller
- Defuzzification

(v) Once done, we save the C++ code, and run 'waf configure' command to configure the project.

(vi) The project is build and the results are obtained.

According to the network layout depicted in Figure 3.3, it can be seen that the network topology can be understood with the help of layered approach. The fuzzy based fog load balancer framework consists of following layers:

(i) Datacenter Layer: Each datacenter consists of multiple zones (like America & US) and each farm is having physical machines that are heterogeneously configured. Each physical machine works as a host for holding virtual machines (VMs) [190].

Algorithm 2 Pseudo Logic of Fog Load Balancer

Input: enum: design, packettype (sensor,video and web)

Initializations: packets P, threshold, routingpaths RP, healthstatus HS, trafficload TL, delaysensitivity DS, energyconsumption EC, linksaturation LS, rules R, rulestrength RS

```
initialization                                   ▷ Initialization of all values
for each RP in routepaths do
    for each P in packets do
        for each variable in design do
            compute fuzzy intervals of each design in range [0:1]
            Read:
            for each TL in trafficload do
                for each DS in delaysensitivity do
                    for each EC in energyconsumption do
                        for each LS in linksaturation do
                            M_fxn = compute membership function value
                            C_g = compute centroid
                            L_HS = aggregate link health status
                            for each R in rules do
                                if RS > threshold then
                                    invoke R in rule for each RP in routes
                                end if
                            end for
                        end for
                    end for
                end for
            end for
        end for
    end for
end for
```

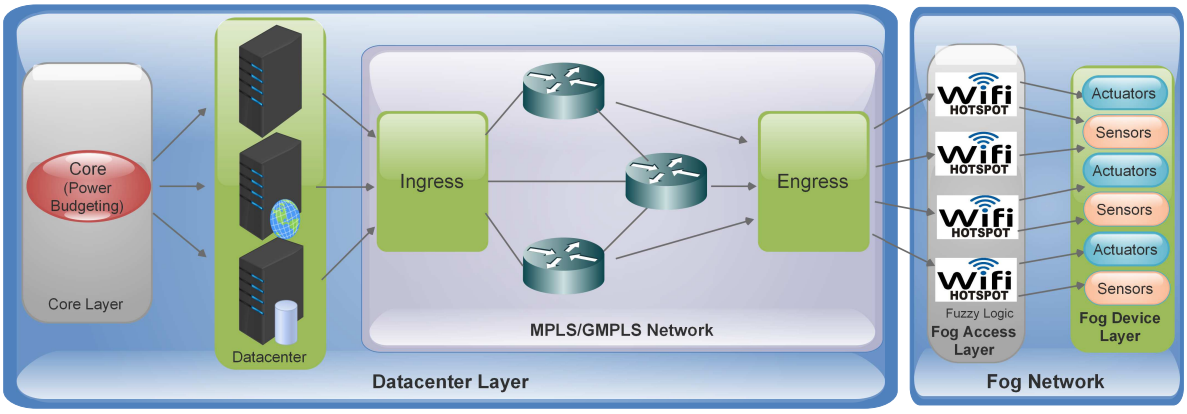


Figure 3.3: Load Balancer Framework (comprising of cloud and fog network zones)

These VMs have inbuilt capability for load balancing and load allocation when load is given to datacenter. The load in this research work is characterized into two definitions: fog load and cloud load.

(ii) Core Layer: This layer reside inside the datacenter. It is the place from where the operations of the datacenter are carried out such as power/energy budgeting etc. [191].

(iii) Fog Access Layer: This layer consists of connecting devices such as switches, routers, software components and different interfaces for managing the cloud network [32].

(iv) Edge/Fog Device Layer: The fog load basically consists of data request-response cycles from sensors, mobile phones and other wireless devices, which are collecting and aggregating data from moving vehicles [192] in their range. These sensors and actuators send data as a TCP/IP request-responses with the information from the edge to cloud networks. Different types of traffic loads that come across in fog network are listed below:

- *Type of Traffic or Workloads:* The traffic load types in the edge are not much different from the data that is processed in the core and datacenter of the cloud networks. The edge networks may normally consist of the data streams that are initiated either from a sensors, mobile telephony or Internet protocol based data packets. Therefore, following types of traffic streams have been taken into consideration and are listed below:
  - *Mobile VoIP:* The mobile device can act as data collector, a device to store data temporary and device for forwarding data [193]. It is a device for communicating via Session Initiation Protocol (SIP). The mobile data will be generated using Markov ON/OFF model. The ON and OFF period lengths were exponentially distributed with a mean of 1.2s (seconds) and 0.8s (seconds), respectively. The packet size would be 65 bytes and an average ON state data rate between 20.0 to 26.4 kbps. These devices may even gather data from bluetooth (BLE) devices using Wireless Personal Area Network (WPAN) technology.
  - *Video streaming:* The video streaming data is based on the streaming traffic model using a real trace file. This video stream has H.263 (video compression standard) codec.
  - *File downloading:* The end user may require different files such as word document/powerpoint or log/trace files on this device for analysis or editing purposes. Hence, this traffic type is also included. The file downloading traffic model constantly sends packets of size 1024 bytes at an average rate ranging between 128-256 kbps.
  - *Content Delivery & Browsing:* The HTTP request-response based cycle represent the web content delivery model which consists of many small objects

such as text, images etc., but in most cases it can be considered as exponential distribution (with a mean of 48,302 bytes).

– *Sensor data:* The sensor data can come from different types of network like: Body Area Network (BAN) or Personal Area Network (PAN), or from Wireless Sensor Area Network (WSAN), or any network that follows IEEE 802.15.4 specifications like ZigBee specifications.

Next section elaborates the evaluation of proposed algorithms with different parameters.

3.4 Design of Fuzzy Logic Scenarios for Evaluation of Algorithms

The parameters considered for designing the fuzzy logic scenarios for evaluations are: input variables (traffic load, delay sensitivity, energy consumption, link saturation) and output variable (link health). Input variables include traffic load, which is based on the packet size and total number of packets. The second parameter is the delay sensitivity, computed in terms of milliseconds. The input variable set also contains the parameter of energy, namely energy consumption, which the devices on that path would consume, if that particular path is considered for sending incoming data. And finally, link saturation is considered as fourth parameter that depends on the utilization of bandwidth and throughput, which the system is able to provide. The output variable is the selection of inter-connects or links that can provide the most optimized alternative to balance the current fog-zone traffic.

Table 3.1: Design Parameters Used in System Model

Input	**Design Terms**	**Description**
Traffic Load	3-level, 5-level and 7-level	A model that captures the traffic load for sensor data, web data and video data.
Delay Sensitivity		A model that captures delay for sensor data, web data and video data.
Energy Consumption		A model that captures energy consumption for sensor data, web data and video data.
Link Saturation		A model that captures saturation in the link for sensor data, web data and video data.

Various scenarios that are proposed, designed and evaluated are: 3-level, 5-level and 7-level. To design fuzzy based fog load balancer, multiple logic/sets have been used, as in formal logic approach only the conclusions that are either true or false (eg. yes or no)

was used. 3-level fuzzy design means that the outcome of the input is divided into three different fuzzy sets. These fuzzy sets may overlap the values of member variables and may be discrete in nature i.e. it may be disjoint sets. Similarly, a 5-level design means there are five imprecise sets of possible outcomes from the input values. Each imprecise outcome describes a linguistic variable that facilitates the expression of rules and facts. When we say a 5-level design, it means the linguistic variable describes five kinds of rules and facts. A 7-level design would mean that there are seven possible ways of describing the output. All these outputs are finally computed as a numerical value with the help of a process known as defuzzification.

By experimentation and literature survey, it is observed that the fuzzy logic is employed at those places where there are multiple influencing parameters on the same process. The influence of these parameters among each other and other aspects of the network system along with their descriptions are defined in Table 3.1 and Table 3.2. The working of the fuzzy logic traffic analysis is dependent on the 'terms' that describe the various degrees of the 'influencing factors' in human readable language. These terms needs to be defined along with their population distribution shape or the membership function. For this purpose, elicitation and interpolation is normally done. By elicitation means that get the source values from an expert and apply non-linear interpolation to have the idea of the shape of the population distribution such as Trapezoid, Gaussian, Sigmoid, S-Shape, Gaussian product, Ramp, Triangle from which membership function can be ascribed to [194, 195]. Henceforth to implement the various scenarios, fuzzification and membership functions need to be understand and described below.

3.4.1 Fuzzification and Membership Functions

The next step is to compute the fuzzy set boundaries for each fuzzy terms and it is determined by the shape of the membership function intervals. It can also be seen graphically the fuzzy set values for each term as shown in Tables 3.3 to 3.15. Instances of each level is also represented in Figures 3.4 to 3.16. Tabular data in various tables show how the fuzzy boundaries are computed on the basis of distribution and the number-ranges with respect to each membership function or shape of the distribution of parameter. Tables 3.3 to 3.6 represent data boundaries for 3-level design. Tables 3.7 to 3.10 represent for 5-level design and Tables 3.11 to 3.14 represent the 7-level design. All these are computed separately for each traffic type. It is assumed that the packet arrival time is based on Gaussian Product distribution in case of traffic load and delay sensitivity. It follows overlapping S-Shaped curve when the traffic load starts to increase. Hence, fuzziness may exemplify into the network.

The traffic load term computation equation (TL_{3S}) at any particular instance (i.e. Low eg. 0.200) of linguistic term value is represented in equation 3.2. Graphically, this is shown

Table 3.2: Descriptor Linguistic Terms Values

Type of Traffic	Design Parameter	Fuzzy Terms			
		Traffic Load (Packet Size in Bytes)	Delay Sensitivity (in milliseconds)	Energy Consumption (in joules)	Link Saturation (in term of normalized value based on available Bandwidth and Throughput)
Sensor Data, Video Data, Web Data	3-Level	Minimum Load, AverageLoad, PeakLoad	NoDelay, AverageDelay, TimeOutDelay	Low Consumption, AverageConsumption, PeakConsumption	LowSaturation, AverageSaturation, FullSaturation
Sensor Data, Video Data, Web Data	5-Level	Minimum Load, BelowAverageLoad, AverageLoad, AboveAverageLoad, PeakLoad	NoDelay, BelowAverageDelay, AverageDelay, AboveAverageDelay, TimeOutDelay	Low Consumption, BelowAverageConsumption, AverageConsumption, AboveAverageConsumption, PeakConsumption	LowSaturation, BelowAverageSaturation, AverageSaturation, AboveAverageSaturation, FullSaturation
Sensor Data, Video Data, Web Data	7-Level	NoLoad, MinimumLoad, BelowAverageLoad, AverageLoad, AboveAverageLoad, MaximumLoad, PeakLoad	NoDelay, MinimumDelay, BelowAverageDelay, AverageDelay, AboveAverageDelay, MaximumDelay, TimeOutDelay	VeryMinute Consumption, VeryLowConsumption, LowConsumption, BelowAverageConsumption, AverageConsumption, AboveAverageConsumption, PeakConsumption	VeryMinute Saturation, VeryLessSaturation, LowSaturation, BelowAverageSaturation, AverageSaturation, AboveAverageSaturation, FullSaturation

Table 3.3: Input Variable: Data Boundaries for Traffic Load in 3-Level Descriptor

Type of Traffic	Traffic Load		
	MinimumLoad	AverageLoad	PeakLoad
Sensor, Video and Web Data: Membership Function	Trapezoid(0.00, 0.08, 0.24, 0.33)	Trapezoid (0.24, 0.33, 0.57, 0.66)	Gaussian (0.500, 0.200)

in Figure 3.4.

$$TL_{3S} = 1.000/MinimumLoad + 0.000/AverageLoad + 0.325/PeakLoad. \quad (3.2)$$

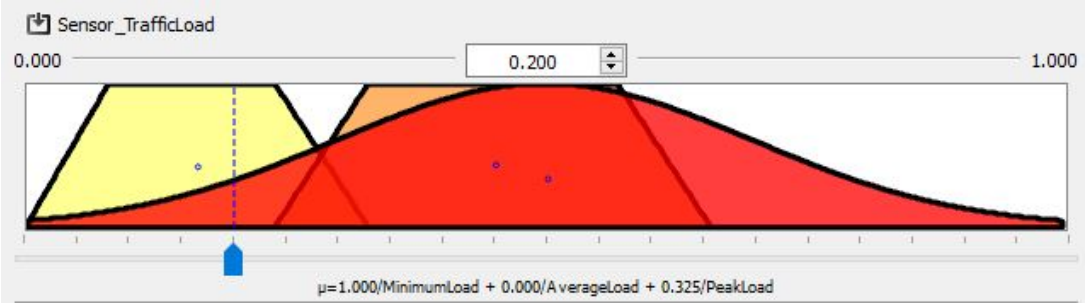


Figure 3.4: Traffic Load considering Low Linguistic Term Value in 3-Level Design

Table 3.4: Input Variable: Data Boundaries for Delay Sensitivity in 3-Level Descriptor

Type of Traffic	Delay Sensitivity		
	NoDelay	AverageDelay	TimeOutDelay
Sensor, Video and Web Data: Membership Function	GaussianProduct (0.500, 0.200, 0.500, 0.200)	Sigmoid (0.500, 20.000)	SShape (0.000, 1.000)

The term computation for delay sensitivity (DS_{3S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is presented by equation 3.3. This is represented graphically in Figure 3.5.

$$DS_{3S} = 1.000/NoDelay + 0.500/AverageDelay + 0.500/TimeOutDelay. \quad (3.3)$$

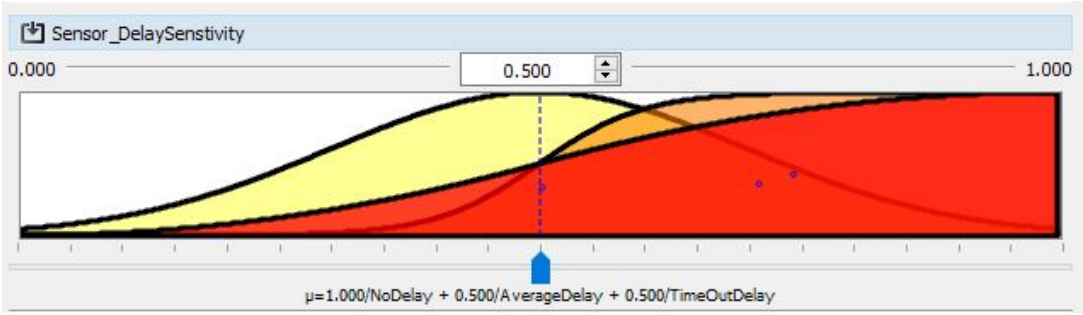


Figure 3.5: Delay Sensitivity considering Medium Linguistic Term Value in 3-Level Design

Table 3.5: Input Variable: Data Boundaries for Energy Consumption in 3-Level Descriptor

Type of Traffic	Energy Consumption		
	Low Consumption	AverageConsumption	PeakConsumption
Sensor, Video and Web Data: Membership Function	Sigmoid (0.510, 20.408)	Sigmoid (0.510, 20.408)	SShape (0.020, 1.000)

Energy consumption term computation equation (EC_{3S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is shown through equation 3.4 and graphically it is depicted in Figure 3.6.

$$EC_{3S} = 0.445/LowConsumption + 0.445/AverageConsumption + 0.478/PeakConsumption. \tag{3.4}$$

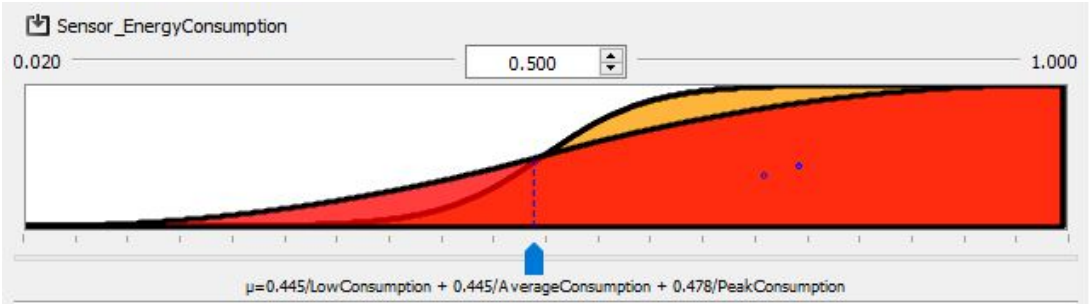


Figure 3.6: Energy Consumption considering Medium Linguistic Term Value in 3-Level Design

Link saturation term computation equation (LS_{3S}) at any particular instance (i.e. HIGH eg. 1.000) of linguistic term value is given by equation 3.5 and graphically it is depicted

Table 3.6: Input Variable: Data Boundaries for Link Saturation in 3-Level Descriptor

Type of Traffic	Link Saturation		
	LowSaturation	AverageSaturation	FullSaturation
Sensor, Video and Web Data: Membership Function	Ramp (0.000, 1.000)	Ramp (0.000, 1.000)	SShape (0.000, 1.000)

in Figure 3.7.

$$LS_{3S} = 1.000/LowSaturation + 1.000/AverageSaturation + 1.000/FullSaturation. \tag{3.5}$$

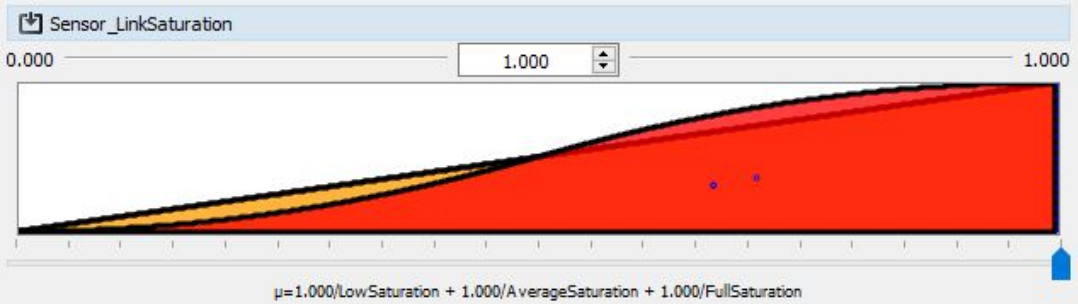


Figure 3.7: Link Saturation considering High Linguistic Term Value in 3-Level Design

Table 3.7: Input Variable: Data Boundaries for Traffic Load in 5-Level Descriptor

Type of Traffic	Traffic Load				
	MinimumLoad	BelowAverageLoad	AverageLoad	AboveAverageLoad	PeakLoad
Sensor, Video and Web Data: Membership Function	Trapezoid (0.000, 0.050, 0.160, 0.200)	Triangle (0.000, 0.500, 1.000)	Trapezoid (0.320, 0.400, 0.530, 0.600)	Triangle (0.000, 0.500, 1.000)	Gaussian (0.500, 0.200)

The traffic load term computation equation (TL_{5S}) at any particular instance (i.e. Low eg. 0.200) of linguistic term value is represented as equation 3.6. Graphically, this is shown

Table 3.8: Input Variable: Data Boundaries for Delay Sensitivity in 5-Level Descriptor

Type of Traffic	Delay Sensitivity				
	NoDelay	BelowAverageDelay	AverageDelay	AboveAverageDelay	TimeOutDelay
Sensor, Video and Web Data: Membership Function	Gaussian Product (0.500, 0.200, 0.500, 0.200)	Triangle (0.000, 0.500, 1.000)	Sigmoid (0.500, 0.200)	Triangle (0.000, 0.500, 1.000)	SShape (0.000, 1.000)

in Figure 3.8.

$$TL_{5S} = 0.000/MinimumLoad + 0.400/BelowAverageLoad + 0.000/AverageLoad + 0.400/AboveAverageLoad + 0.325/PeakLoad. \tag{3.6}$$

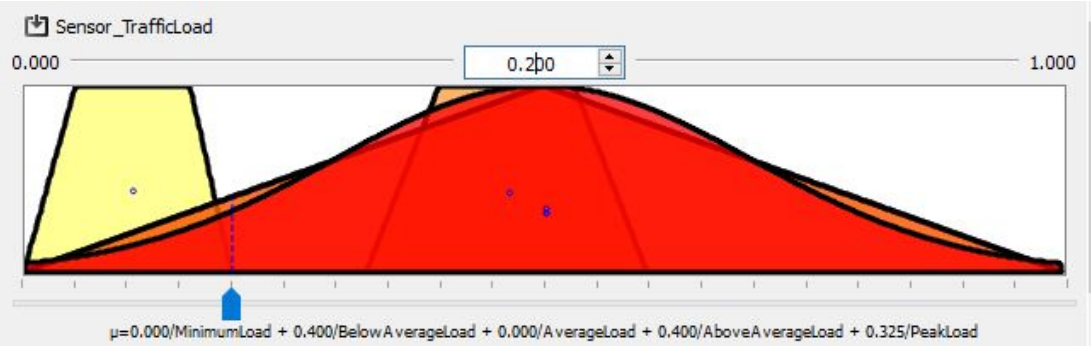


Figure 3.8: Traffic Load considering Low Linguistic Term Value in 5-Level Design

The term computation equation for delay sensitivity (DS_{5S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is presented as equation 3.7. This is represented graphically in Figure 3.9.

$$DS_{5S} = 1.000/NoDelay + 1.000/BelowAverageDelay + 0.500/AverageDelay + 1.000/AboveAverageDelay + 0.500/TimeOutDelay. \tag{3.7}$$

Energy consumption term computation equation (EC_{5S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is given by equation 3.8 and graphically it is

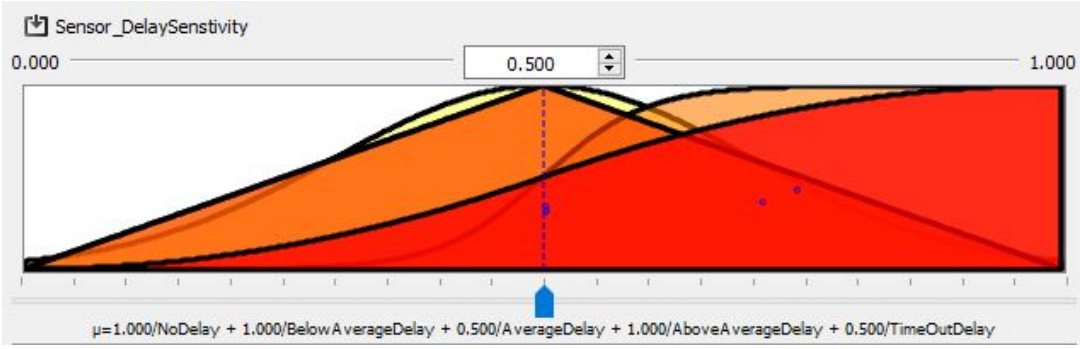


Figure 3.9: Delay Sensitivity considering Medium Linguistic Term Value in 5-Level Design

Table 3.9: Input Variable: Data Boundaries for Energy Consumption in 5-Level Descriptor

Type of Traffic	Energy Consumption				
	LowConsumption	BelowAverageConsumption	AverageConsumption	AboveAverageConsumption	PeakConsumption
Sensor, Video and Web Data: Membership Function	Sigmoid (0.510, 0.20408)	Triangle (0.020, 0.510, 1.000)	Sigmoid (0.510, 0.20408)	Triangle (0.020, 0.510, 1.000)	SShape (0.020, 1.000)

depicted in Figure 3.10.

$$\begin{aligned} EC_{5S} = {} & 0.445/LowConsumption + 0.978/BelowAverageConsumption \\ & +0.445/AverageConsumption + 0.978/AboveAverageConsumption \\ & +0.478/PeakConsumption. \end{aligned} \tag{3.8}$$

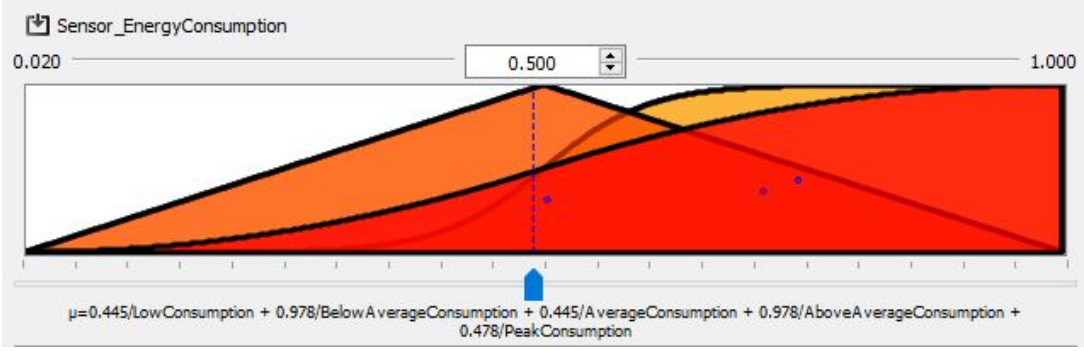


Figure 3.10: Energy Consumption considering Medium Linguistic Term Value in 5-Level Design

Table 3.10: Input Variable: Data Boundaries for Link Saturation in 5-Level Descriptor

Type of Traffic	Link Saturation				
	LowSaturation	BelowAverageSaturation	AverageSaturation	AboveAverageSaturation	FullSaturation
Sensor, Video and Web Data: Membership Function	Ramp (0.000, 1.000)	Triangle (0.000, 0.500, 1.000)	Ramp (0.000, 1.000)	Triangle (0.000, 0.500, 1.000)	SShape (0.000, 1.000)

Link saturation term computation equation (LS_{5S}) at any particular instance (i.e. High eg. 0.999) of linguistic term value is presented through equation 3.9 and graphically it is shown in Figure 3.11.

$$\begin{aligned} LS_{5S} = {} & 0.999/LowSaturation + 0.002/BelowAverageSaturation \\ & + 0.999/AverageSaturation + 0.002/AboveAverageSaturation \\ & + 1.000/FullSaturation. \end{aligned} \tag{3.9}$$

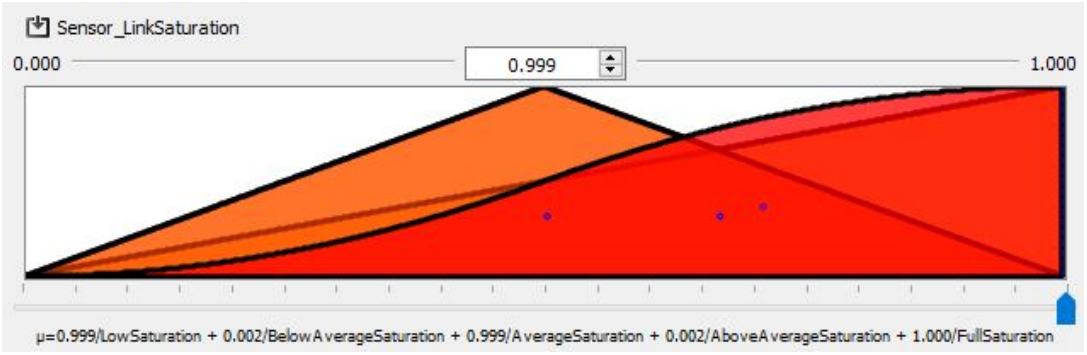


Figure 3.11: Link Saturation considering High Linguistic Term Value in 5-Level Design

The traffic load term computation equation (TL_{7S}) at any particular instance (i.e. Low eg. 0.200) of linguistic term value is represented by equation 3.10. Graphically, this is shown in Figure 3.12.

$$\begin{aligned} TL_{7S} = {} & 0.000/NoLoad + 1.000/MinimumLoad + 0.000/BelowAverageLoad \\ & + 0.000/AverageLoad + 0.000/AboveAverageLoad + 0.000/MaximumLoad \\ & + 0.325/PeakLoad. \end{aligned} \tag{3.10}$$

Table 3.11: Input Variable: Data Boundaries for Total Load in 7-Level Descriptor

Type of Traffic	Total Load						
	NoLoad	MinimumLoad	BelowAverageLoad	AverageLoad	AboveAverageLoad	MaximumLoad	PeakLoad
Sensor, Video and Web Data: Membership Function	Constant (0.000)	Trapezoid (0.110, 0.150, 0.240, 0.290)	Trapezoid (0.240, 0.290, 0.390, 0.440)	Trapezoid (0.390, 0.440, 0.540, 0.600)	Trapezoid (0.540, 0.600, 0.690, 0.740)	Trapezoid (0.690, 0.740, 0.830, 0.880)	Gaussian (0.500, 0.200)

Table 3.12: Input Variable: Data Boundaries for Delay Sensitivity in 7-Level Descriptor

Type of Traffic	Delay Sensitivity						
	NoDelay	MinimumDelay	BelowAverageDelay	AverageDelay	AboveAverageDelay	MaximumDelay	TimeOutDelay
Sensor, Video and Web Data: Membership Function	Gaussian Product (0.500, 0.200, 0.500, 0.200)	Triangle (0.000, 0.500, 1.000)	Triangle (0.000, 0.500, 1.000)	Sigmoid (0.500, 0.200)	Triangle (0.000, 0.500, 1.000)	SShape (0.000, 1.000)	SShape (0.000, 1.000)

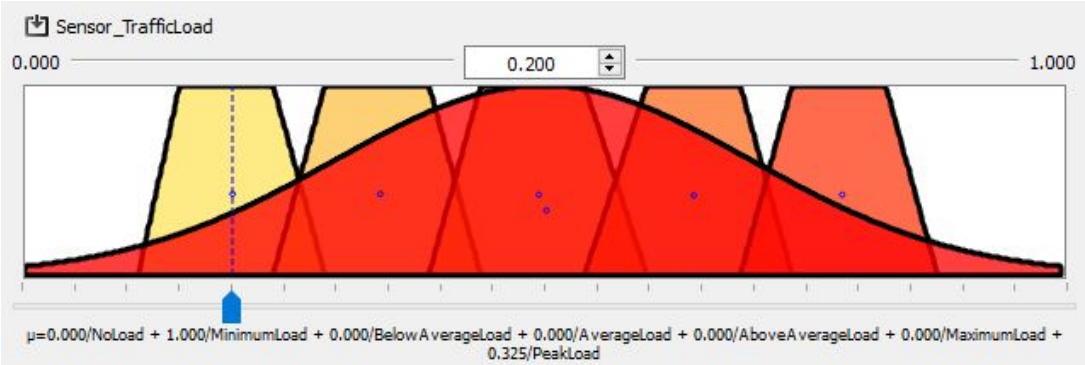


Figure 3.12: Traffic Load considering LOW Linguistic Term Value in 7-Level Design

The term computation equation for delay sensitivity (DS_{7S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is presented in equation 3.11. This is represented graphically in Figure 3.13.

$$\begin{aligned} DS_{7S} = 1.000/NoDelay + 1.000/MinimumDelay + 1.000/BelowAverageDelay \\ +0.500/AverageDelay + 1.000/AboveAverageDelay \\ +0.500/MaximumDelay + 0.500/TimeOutDelay. \end{aligned} \tag{3.11}$$

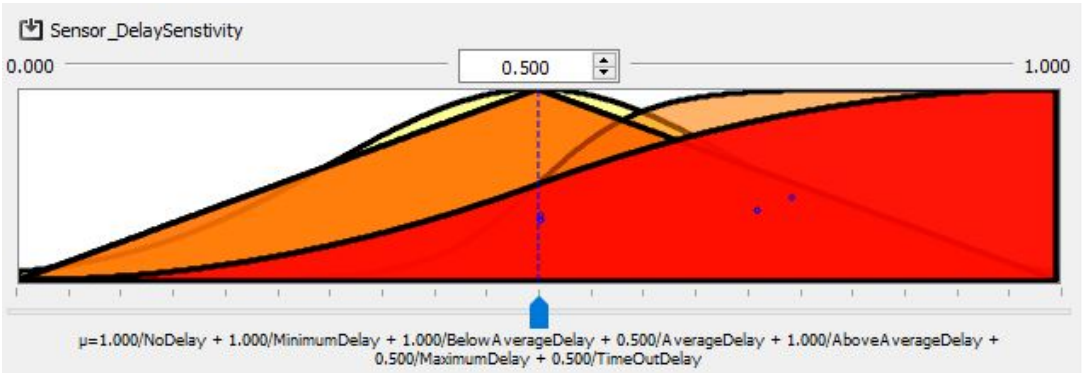


Figure 3.13: Delay Sensitivity considering Medium Linguistic Term Value in 7-Level Design

Energy consumption term computation equation (EC_{7S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is given through equation 3.12 and graphically it is depicted in Figure 3.14.

$$\begin{aligned} EC_{7S} = 0.001/VeryMinuteConsumption + 0.445/VeryLowConsumption \\ +0.445/LowConsumption + 0.445/BelowAverageConsumption \\ +0.445/AverageConsumption + 0.478/AboveAverageConsumption \\ +0.478/PeakConsumption. \end{aligned} \tag{3.12}$$

Link saturation term computation equation (LS_{7S}) at any particular instance (i.e. HIGH eg. 0.990) of linguistic term value is presented in equation 3.13 and graphically it is

Table 3.13: Input Variable: Data Boundaries for Energy Consumption in 7-Level Descriptor

Type of Traffic	Energy Consumption						
	VeryMinuteConsumption	VeryLowConsumption	LowConsumption	BelowAverageConsumption	AverageConsumption	AboveAverageConsumption	PeakConsumption
Sensor, Video and Web Data: Membership Function	Constant (0.001)	Sigmoid (0.510, 0.20408)	Sigmoid (0.510, 0.20408)	Sigmoid (0.510, 0.20408)	Sigmoid (0.510, 0.20408)	SShape (0.020, 1.000)	SShape (0.020, 1.000)

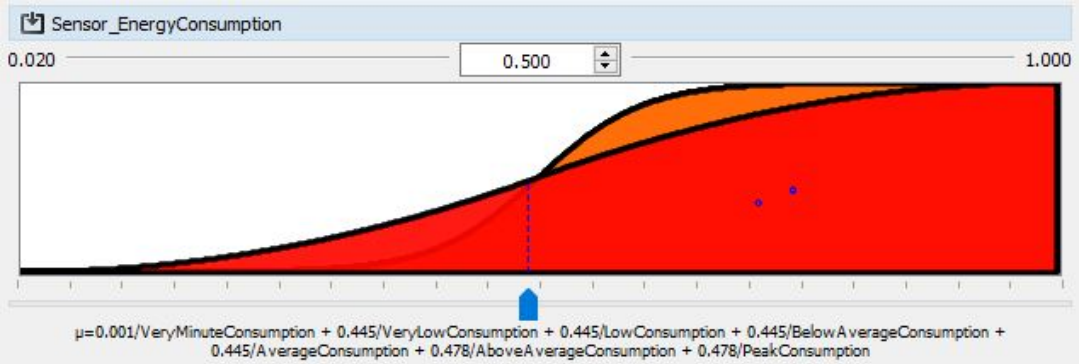


Figure 3.14: Energy Consumption considering Medium Linguistic Term Value in 7-Level Design

depicted in Figure 3.15.

$$\begin{aligned} LS_{7S} = {} & 0.020/VeryMinuteSaturation + 0.020/VeryLessSaturation \\ & + 0.990/LowSaturation + 0.020/BelowAverageSaturation \\ & + 0.990/AverageSaturation + 0.020/AboveAverageSaturation \\ & + 1.000/FullSaturation. \end{aligned} \tag{3.13}$$

The link health term computation equation (LH_{3S}) at any particular instance (i.e. Medium eg. 0.500) of linguistic term value is represented by equation 3.14. Graphically, this is shown in Figure 3.16.

$$LH_{3S} = 0.607/LowQuality + 0.607/MediumQuality + 0.607/HighQuality. \tag{3.14}$$

Here, membership functions for all scenarios have been discussed. Next section discusses

Table 3.14: Input Variable: Data Boundaries for Link Saturation in 7-Level Descriptor

Type of Traffic	Link Saturation						
	VeryMinuteSaturation	VeryLowSaturation	LowSaturation	BelowAverageSaturation	AverageSaturation	AboveAverageSaturation	FullSaturation
Sensor, Video and Web Data: Membership Function	Triangle (0.000, 0.500, 1.000)	Triangle (0.000, 0.500, 1.000)	Ramp (0.000, 1.000)	Triangle (0.000, 0.500, 1.000)	Ramp (0.000, 1.000)	Triangle (0.000, 0.500, 1.000)	SShape (0.000, 1.000)

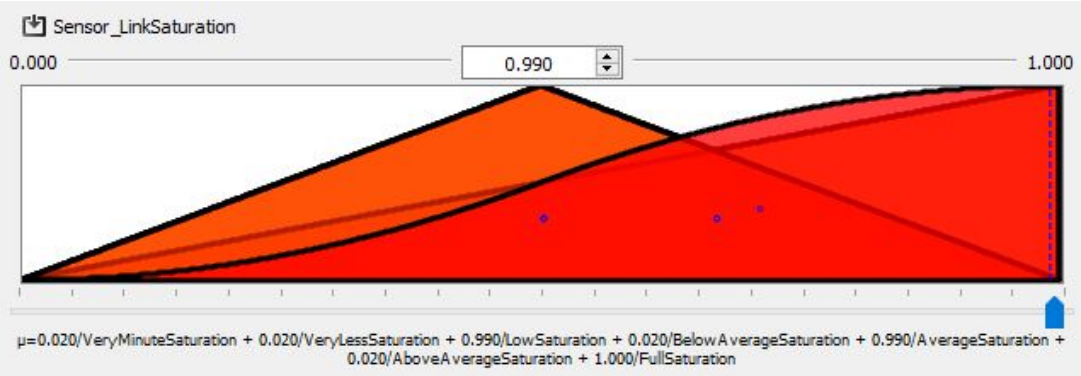


Figure 3.15: Link Saturation considering High Linguistic Term Value in 7-Level Design

the selection of fuzzy membership functions and inference controller.

3.4.2 Selection of Fuzzy Membership Functions and Inference Controller

The selection of fuzzy membership functions are computed on the basis of sample distribution of the influencing variables. The variables are mapped using two methods. In the first method, knowledge from the experts is taken and in the second method, knowledge is gathered from the empirical analysis of simulations/experimentation. The simulation

Table 3.15: Output Variable: Data Boundaries for Link Health

Type of Traffic	Link Health		
	LowQuality	MediumQuality	HighQuality
Membership Function	Ramp (0.000, 1.000)	Ramp (0.000, 1.000)	Ramp (0.000, 1.000)

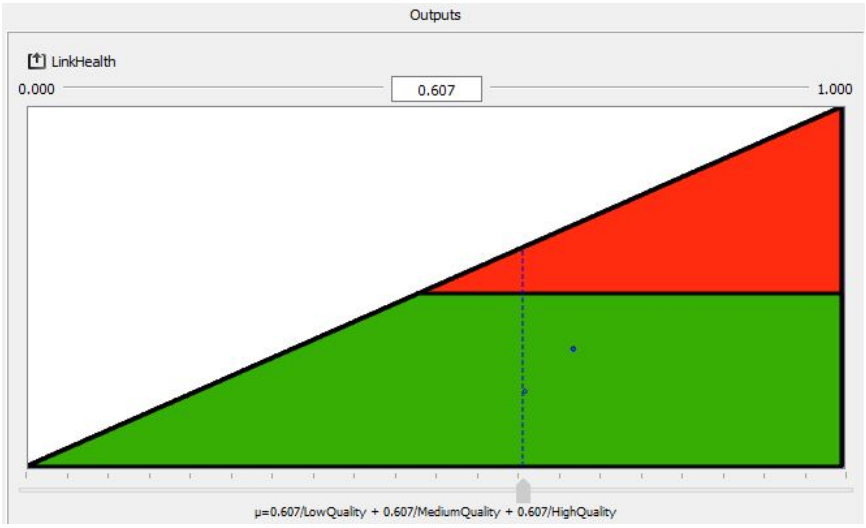


Figure 3.16: Link Health considering Medium Linguistic Term Value in 3-Level Design

and empirical data show that traffic load factor follows a distribution model of Gaussian and the intensity of the load that can be modeled using its product. Typically, in a day the traffic gradually peaks and then starts downward trend. Similar is the case of delay sensitivity, as it also follows the similar trend. In case of energy consumption, the consumption will be highest when the load factor is highest or there is Gaussian Peak. The link saturation is divided into four intervals due to fact that the utilization of bandwidth behaves in manner depicted like the intervals of Trapezoidal shape distribution. As the link saturation increases, the network gets slow but the moment traffic load decreases, the link saturation decreases.

The most appropriate inference engine in context of our problem is Sugeno as it helps to give output using weighted mathematical equations along with constants. This can be used to infer health of the links in the fog zone of the cloud network. It is preferred to Mamdani type fuzzy engine as it computes fuzzy set output. e.g If M is X_1, and N is X_2, then O is X_3 (X_1, X_2, X_3 are fuzzy sets), whereas in case of Sugeno controller, the values are given as linear expression (If M is X_1 and N is X_2 then $O = aX_1 + bX_2 + c$ (linear expression) (where a, b, and c are constants).

The above section describes the selection of fuzzy membership functions as well as inference controller. Next section discusses the fuzzy control rules using these membership functions.

3.5 Fuzzy Control Rules

Fuzzy control rules depend on the analytical models based on the experimental/ empirical/experienced behaviour of the system as compared to the knowledge of system variables. Hence, the rules are also fuzzy in nature. The design of fuzzy rules is entirely de-

pendent on the number of fuzzy inputs and their outcomes. The possible non-conflicting rules, that can be constructed to model their behaviour of load balancing system are required. Since, these rules work with the imprecise values, the rules must follow the fuzzy rule set and De Morgan's Law and theory. The data involved in this simulation is a universal set of continuous and finite values. All the fuzzy set operations can be applied to form sets using these rules. These rules maintain the transitive, idempotence, commutative, associative and distributive properties. Due to this, candidate rules need to be carefully analyzed before their actual application. Identical rules or the rules that have same outcome need to be eliminated and the only relevant rules are allowed in the system. To maintain brevity and readability of the rules, Tables 3.16 to 3.24 give a partial list of the rules being applied to the aforesaid problem of load balancing. There are four factors and each factor has minimum of three degrees or the terms (Low, Medium and High). The mathematical relationship is derived from the data by building linear equation model between the interacting variables. Since, the factors are influencing and competing with others, the relationship acquire certain degree of impreciseness or fuzziness but, they still follow the mathematics of non-classical set theory.

Table 3.16: 3-Level Fuzzy Rules for Sensor Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
Min_{load}	No_{delay}	$Low_{consumption}$	$Low_{saturation}$	$High_{quality}$
Min_{load}	No_{delay}	$Peak_{consumption}$	$Full_{saturation}$	$Low_{quality}$
Min_{load}	TO_{delay}	$Peak_{consumption}$	$Avg_{saturation}$	$Low_{quality}$
Avg_{load}	Avg_{delay}	$Low_{consumption}$	$Low_{saturation}$	$Med_{quality}$
Avg_{load}	Avg_{delay}	$Avg_{consumption}$	$Avg_{saturation}$	$Low_{quality}$
$Peak_{load}$	No_{Delay}	$Avg_{consumption}$	$Avg_{saturation}$	$Low_{quality}$
$Peak_{load}$	TO_{delay}	$Peak_{Consumption}$	$Full_{saturation}$	$Low_{quality}$

Table 3.17: 3-Level Fuzzy Rules for Video Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
Min_{load}	No_{delay}	$Avg_{consumption}$	$Full_{saturation}$	$Low_{quality}$
Min_{load}	No_{delay}	$Peak_{consumption}$	$Low_{saturation}$	$High_{quality}$
Avg_{load}	No_{delay}	$Avg_{consumption}$	$Avg_{saturation}$	$Med_{quality}$
Avg_{load}	No_{delay}	$Avg_{consumption}$	$Full_{saturation}$	$Low_{quality}$

Continued on next page

Table 3.17 – *Continued from previous page*

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
$Peak_{load}$	Avg_{delay}	$Peak_{consumption}$	$Full_{saturation}$	$Low_{quality}$
$Peak_{load}$	TO_{delay}	$Peak_{consumption}$	$Avg_{saturation}$	$Low_{quality}$

Table 3.18: 3-Level Fuzzy Rules for Web Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
Min_{load}	No_{delay}	$Low_{consumption}$	$Full_{saturation}$	$Med_{quality}$
Min_{load}	No_{delay}	$Avg_{consumption}$	$Low_{saturation}$	$High_{quality}$
Min_{load}	Avg_{delay}	$Avg_{consumption}$	$Low_{saturation}$	$Low_{quality}$
Min_{load}	Avg_{delay}	$Avg_{consumption}$	$Avg_{saturation}$	$Med_{quality}$
Avg_{load}	TO_{delay}	$Low_{consumption}$	$Full_{saturation}$	$Low_{quality}$
$Peak_{load}$	TO_{delay}	$Avg_{consumption}$	$Low_{saturation}$	$Low_{quality}$

Table 3.19: 5-Level Fuzzy Rules for Sensor Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
Min_{load}	No_{delay}	$Low_{consumption}$	$Low_{saturation}$	$High_{quality}$
Min_{load}	$BAvg_{delay}$	$BAvg_{consumption}$	$Avg_{saturation}$	$Med_{quality}$
Min_{load}	Avg_{delay}	$BAvg_{consumption}$	$Avg_{saturation}$	$Med_{quality}$
Min_{load}	$AAvg_{delay}$	$Avg_{consumption}$	$Full_{saturation}$	$Low_{quality}$
Min_{load}	TO_{delay}	$BAvg_{consumption}$	$Low_{saturation}$	$Low_{quality}$
$Peak_{load}$	$BAvg_{delay}$	$Low_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$

Table 3.20: 5-Level Fuzzy Rules for Video Data Boundaries

Traffic Load	Delay Senstivity	Energy Consumption	Link Saturation	Link Health
Min_{load}	No_{delay}	$Low_{consumption}$	$BAvg_{saturation}$	$High_{quality}$
$BAvg_{load}$	$BAvg_{delay}$	$BAvg_{consumption}$	$BAvg_{saturation}$	$Med_{quality}$
Avg_{load}	No_{delay}	$Avg_{consumption}$	$Low_{saturation}$	$High_{quality}$
$AAvg_{load}$	$BAvg_{delay}$	$AAvg_{consumption}$	$BAvg_{saturation}$	$Low_{quality}$
$Peak_{load}$	No_{delay}	$AAvg_{consumption}$	$Low_{saturation}$	$Low_{quality}$
$Peak_{load}$	TO_{delay}	$Avg_{consumption}$	$BAvg_{saturation}$	$Low_{quality}$

Table 3.21: 5-Level Fuzzy Rules for Web Data Boundaries

Traffic Load	Delay Senstivity	Energy Consumption	Link Saturation	Link Health
Min_{load}	Avg_{delay}	$Low_{consumption}$	$Low_{saturation}$	$Med_{quality}$
Min_{load}	TO_{delay}	$AAvg_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$
$BAvg_{load}$	$BAvg_{delay}$	$AAvg_{consumption}$	$Avg_{saturation}$	$Med_{quality}$
$AAvg_{load}$	No_{delay}	$BAvg_{consumption}$	$BAvg_{saturation}$	$Med_{quality}$
$Peak_{load}$	No_{delay}	$Avg_{consumption}$	$Full_{saturation}$	$Low_{quality}$
$Peak_{load}$	TO_{delay}	$Peak_{consumption}$	$Full_{saturation}$	$Low_{quality}$

Table 3.22: 7-Level Fuzzy Rules for Sensor Data Boundaries

Traffic Load	Delay Senstivity	Energy Consumption	Link Saturation	Link Health
No_{load}	No_{delay}	$Low_{consumption}$	$VMinute_{saturation}$	$High_{quality}$
No_{load}	Min_{delay}	$VMinute_{consumption}$	$Low_{saturation}$	$High_{quality}$
Min_{load}	Min_{delay}	$Peak_{consumption}$	$VMinute_{saturation}$	$Low_{quality}$
Min_{load}	TO_{delay}	$Avg_{consumption}$	$Full_{saturation}$	$Low_{quality}$
$BAvg_{load}$	Min_{delay}	$Peak_{consumption}$	$BAvg_{saturation}$	$Low_{quality}$
$AAvg_{load}$	No_{delay}	$Avg_{consumption}$	$VLess_{saturation}$	$Med_{quality}$

Table 3.23: 7-Level Fuzzy Rules for Video Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
No_{load}	No_{delay}	$VMinute_{consumption}$	$Low_{saturation}$	$Med_{quality}$
Min_{load}	No_{delay}	$BAvg_{consumption}$	$Low_{saturation}$	$High_{quality}$
$BAvg_{load}$	Avg_{delay}	$VLow_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$
Avg_{load}	Avg_{delay}	$VMinute_{consumption}$	$Full_{saturation}$	$Med_{quality}$
$AAvg_{load}$	Avg_{delay}	$VMinute_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$
Max_{load}	$AAvg_{delay}$	$VMinute_{consumption}$	$VMinute_{saturation}$	$Med_{quality}$

Table 3.24: 7-Level Fuzzy Rules for Web Data Boundaries

Traffic Load	**Delay Senstivity**	**Energy Consumption**	**Link Saturation**	**Link Health**
No_{load}	No_{delay}	$VMinute_{consumption}$	$Avg_{saturation}$	$High_{quality}$
$BAvg_{load}$	Avg_{delay}	$AAvg_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$
Avg_{load}	$AAvg_{delay}$	$AAvg_{consumption}$	$AAvg_{saturation}$	$Low_{quality}$
$AAvg_{load}$	TO_{delay}	$VLow_{consumption}$	$VLess_{saturation}$	$Med_{quality}$
Max_{load}	Max_{delay}	$VMinute_{consumption}$	$VMinute_{saturation}$	$Med_{quality}$
$Peak_{load}$	TO_{delay}	$Peak_{consumption}$	$Avg_{saturation}$	$Low_{quality}$

Above section discussed about fuzzy control rules and their formulation. Next subsections deals with crisp values that are obtained as the output from fuzzy process. The defuzzification process and strength of rules are discussed below.

3.5.1 Defuzzification

The defuzzification process gives a way to interpret numerical values corresponding to inputs processed. It is a process of getting crisp (numerical value) of the fuzzy algorithm. This is the output value with which the results are computed. Since, fuzzification means converting numerical values to human readable values, defuzzification can be understood as a process in reverse i.e. from human terms to mathematical values. There are multiple ways to arrive at the output value. This process includes methods such as Center of Gravity (COG), Center of Area (COA) and Center of Sums Method (COS). For this research work, COG has been used. It can be computed using the equation 3.15.

$$COG = \frac{\sum_{i=1}^{k} H_i \times \bar{G}_i}{\sum_{i=1}^{k} H_i}, \tag{3.15}$$

where H_i refers to rule strength of i^{th} rule and k refers to number of rules fired. $\bar{G}_i$ represents the center of fuzzy area.

This can be easily understood by taking an example having two fuzzy sets F_1 and F_2, as depicted in Figure 3.17. Let these two fuzzy sets have area as $Area_1$ and $Area_2$.

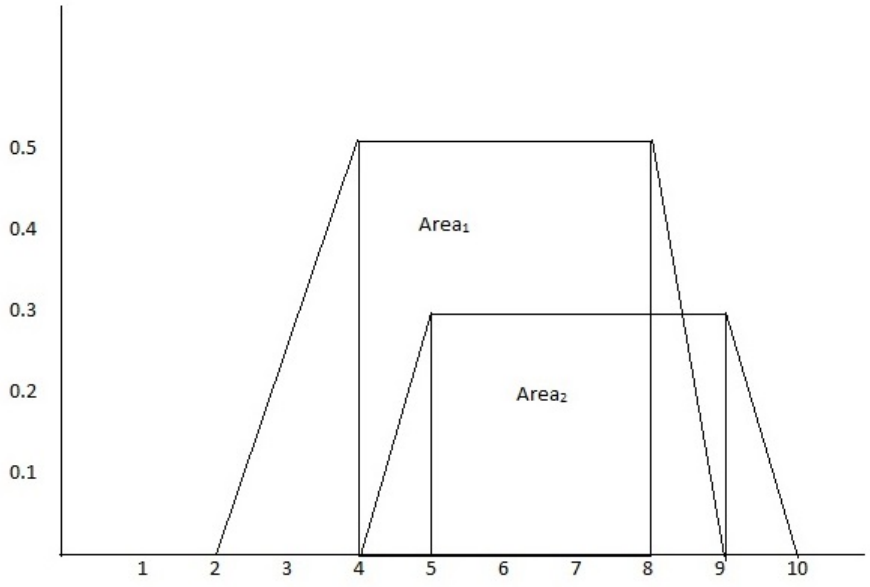


Figure 3.17: Example of Defuzzification

$Area_1 = \frac{1}{2} \times [(9-2)+(8-4)] \times 0.5 = 5.5/2 = 2.75$

$Area_2 = \frac{1}{2} \times [(10-4)+(9-5)] \times 0.3 = 3/2 = 1.5$

Center of Area for Fuzzy set F_1, say $\bar{X}_1 = (8+4)/2) = 6$

Center of Area for Fuzzy set F_2, say $\bar{X}_2 = (9+5)/2) = 7$

Hence, the defuzzification value, COG = $\frac{(Area_1\bar{X}_1+Area_2\bar{X}_2)}{Area_1+Area_2} = \frac{(2.75\times5+1.5\times6)}{2.75+1.5} = 22.75/4.25 = 5.35$

3.5.2 Strength of Rules

The execution of fuzzy rules follows the procedure that is based on their strength. This metric is computed to determine the fulfillment of the fuzzy terms being used.

In fact, the pre-step is to aggregate the quantified consequents to crisp output (defuzzification process), then aggregate all these crisp numerical outputs to establish a rule strength. In this way, consequent is found for each rule. This process is called implication. This helps to determine the relevancy of fuzzy rules.

3.6 Analysis of Various Design Levels

The selection of appropriate design of the fuzzy based fog load balancer depends on multiple factors as given below.

(i) Total number of rules: These are the total number of rules that are required to run the fuzzy traffic management algorithm. This count will vary as per the design selected. Higher levels have more number of rules than in case of lower levels.

(ii) Number of relevant and irrelevant rules: Higher number of redundant rules means waste of resources and large overheads. Hence, there is a need for more fine level tuning of the algorithm. This can be known by mapping the value of rule strength with respect to number of rules. The design of rules and fuzzy boundaries depend on the type of data stream i.e. sensor, web and video. In each data stream, different rules are followed to construct parameters with logical conditions and 'AND' logical operator.

- In *sensor data*, the amount/volume of data flow is low as compared to web and video data, and it is highly energy efficient. Therefore, the construction of if-then-else conditions for high quality are based on low and minimum energy consumption.
- In case of *web data*, the if-then-else fuzzy conditions are guided by the peak load and energy consumption scenarios. Higher peak load implies high energy usage. Taking this as guiding principle, rules are constructed and such condition is mapped as low quality link.
- *Video Data:* Since video data is highly sensitive to the delay and no one would like to have some time wastage in buffering, the designing of rules is done in such a way that the fuzzy condition of no delay or minimum delay is mapped as high quality otherwise all cases are either medium or low quality.

To evaluate various design levels (3-level, 5-level and 7-level), different test cases and comparative analysis have been carried out in subsequent sections.

3.6.1 Test Cases for Evaluation of 3-level, 5-level and 7-level Scenarios

To evaluate the levels of fuzzy controller (3-level, 5-level, 7-level), 'worst case analysis' approach has been followed. This method helps to identify the level at which fuzzy controller should be designed for each traffic type (sensor, web and video). Also, this helps us to identify rules that are irrelevant and not required in the context.

3.6.1.1 3-Level Design of Fog Load Balancer

3-level design of fog load balancer is considered for sensor data, video data and web data. These three types of data forms different test cases, that are described below.

(i) For Sensor Data: Table 3.25 shows 3-level descriptor test cases for sensor data.

Table 3.25: 3-Level Descriptors Case for analysis of Sensor Data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 1	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 2	Low (0.1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 3	Low (0.1)	High (1)	Low (0.02)	Low (0)	0	0
Case 4	Low (0.1)	Low (0)	High (1)	Low (0)	0	0
Case 5	Low (0.1)	Low (0)	Low (0.02)	High (1)	0	0
Case 6	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 7	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 8	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 9	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 10	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 11	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 12	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 13	Low (0.1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 14	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 15	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 16	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 17	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 18	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 19	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 20	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 21	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 22	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 23	High (1)	High (1)	High (1)	Low (0)	0	0
Case 24	High (1)	High (1)	Low (0.02)	High (1)	0	0
Case 25	High (1)	High (1)	Medium (0.5)	Medium (0.5)	27	0.044
Case 26	High (1)	High (1)	Medium (0.5)	High (1)	27	0.044
Case 27	High (1)	High (1)	High (1)	Medium (0.5)	27	0.044

Continued on next page

Table 3.25 – *Continued from previous page*

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 28	High (1)	High (1)	High (1)	High (1)	27	0.044
Case 29	Low (0.1)	High (1)	Medium (0.5)	Medium (0.5)	27	0.0895
Case 30	Low (0.1)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.135
Case 31	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	27	0.4645
Case 32	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	27	0.4645
Case 33	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.4645

It can be seen that in case 1 to case 24, the strength of rules remains zero, because of which no rule is fired. From this, it can be inferred that these rules are unnecessary and can be eliminated from the load balancing algorithm to optimize the algorithm. This behaviour of rules can be attributed to the fact that traffic load in cases (case 2 to case 5) remain low. Even, in cases when the traffic load value is between 0 (low) and 0.5 (medium) load, other factors remaining the same (as low), the number of rules fired remains zero.

In case 3 and case 7, it can be noted that even when the delay value reaches maximum value (1) with any magnitude of traffic load, there are no rules fired. This may be simply because logical conditions of low traffic, medium traffic and low energy keeps the rules dominant.

In case 24, when the value of energy consumption is low, and the value of traffic load, delay sensitivity and link saturation are high, the fuzzy logic rules still do not get excited. This shows that in such conditions the traffic remains self managed and does not require intervention from the fuzzy controls, because the health of the link remains 0.5 (medium).

It can be seen from the tabular data 3.25 and Figure 3.18 that the rules start getting fired when the value of traffic load and delay sensitivity are high (i.e. 1), energy consumption and link saturation is medium (i.e. 0.5) or high (i.e. 1). This is represented in case 25 to case 28. These observations are summarized using the pie graph as shown in Figure 3.19 and Figure 3.20.

It is apparent from the Figure 3.18 that when the magnitude of strength increases, the number of rules affected also increases and the relationship is almost linear in nature (with constant slope of 0.0016).

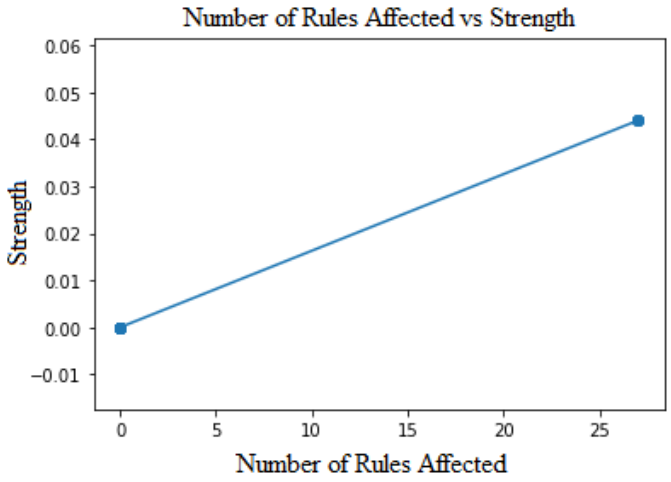


Figure 3.18: Number of Rules Affected vs Strength in 3-Level Design

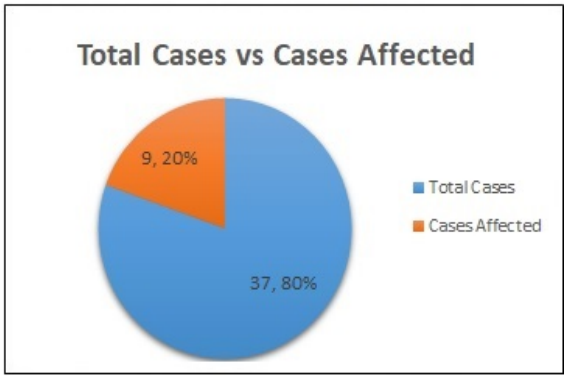


Figure 3.19: Total Cases vs Cases Affected

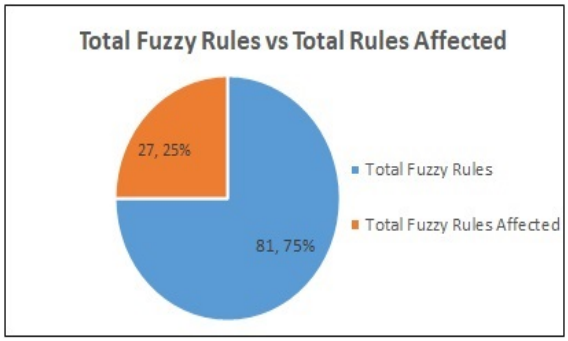


Figure 3.20: Total Fuzzy Rules vs Total Fuzzy Rules Affected

(ii) For Video Data: Table 3.26 shows 3-level descriptor test cases for video data.

Table 3.26: 3-Level Descriptors Case for analysis of Video Data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 34	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 35	Low (0.33)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 36	Low (0.33)	High (1)	Low (0.02)	Low (0)	0	0
Case 37	Low (0.33)	Low (0)	High (1)	Low (0)	0	0
Case 38	Low (0.33)	Low (0)	Low (0.02)	High (1)	0	0
Case 39	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 40	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 41	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 42	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0

Continued on next page

Table 3.26 – *Continued from previous page*

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 43	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 44	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 45	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 46	Low (0.33)	Low (0)	Low (0.02)	Low (0)	0	0
Case 47	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 48	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 49	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 50	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 51	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 52	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 53	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 54	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 55	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 56	High (1)	High (1)	High (1)	Low (0)	0	0
Case 57	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 58	High (1)	High (1)	Medium (0.5)	Medium (0.5)	27	0.044
Case 59	High (1)	High (1)	Medium (0.5)	High (1)	27	0.044
Case 60	High (1)	High (1)	High (1)	Medium (0.5)	27	0.044
Case 61	High (1)	High (1)	High (1)	High (1)	27	0.044
Case 62	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	27	0.3243
Case 63	Low (0.33)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.4645
Case 64	Low (0.33)	High (1)	Medium (0.5)	Medium (0.5)	18	0.4645
Case 65	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	27	0.4645
Case 66	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.4645

The analysis of 3-level fuzzy control load balancer, in case of video data (from Table 3.26), shows that the low states of traffic load, delay sensitivity, energy consumption and link saturation does not invoke any rule. This can be validated by checking the value of strength metric. Another observation that can be made is when the values of traffic load, delay sensitivity and link saturation are of the range 0 to 0.5, the strength remains zero and there is no activation of any rule. As we check the tabular data 3.26 from case 58 to case 66, it is observed that the rules are getting fired. It

can clearly be seen that major influence in case of video data is the traffic load. The cases 62-66 show that low and medium values are putting impact on the strength metric and so is the high value of traffic load. In fact low, medium and high traffic load conditions when joined with medium delay sensitivity, energy consumption and link saturation also lead to activation of about 18 to 27 number of rules. From this, it can be inferred that the role of fuzzy balance controller becomes important when there is low to high traffic load and delay sensitivity, energy consumption and link saturation is between medium (0.5) to high (1).

(iii) For Web Data: Table 3.27 shows 3-level descriptor test cases for web data.

Table 3.27: 3-Level Descriptors Case for analysis of Web Data

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 67	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 68	Low (0.22)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 69	Low (0.22)	High (1)	Low (0.02)	Low (0)	0	0
Case 70	Low (0.22)	Low (0)	High (1)	Low (0)	0	0
Case 71	Low (0.22)	Low (0)	Low (0.02)	High (1)	0	0
Case 72	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 73	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 74	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 75	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 76	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 77	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 78	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 79	Low (0.22)	Low (0)	Low (0.02)	Low (0)	0	0
Case 80	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 81	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 82	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 83	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 84	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 85	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 86	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 87	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 88	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 89	High (1)	High (1)	Medium (0.5)	Low (0)	0	0

Continued on next page

Table 3.27 – *Continued from previous page*

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 90	High (1)	High (1)	High (1)	Low (0)	0	0
Case 91	High (1)	High (1)	Medium (0.5)	Medium (0.5)	27	0.044
Case 92	High (1)	High (1)	Medium (0.5)	High (1)	27	0.044
Case 93	High (1)	High (1)	High (1)	Medium (0.5)	27	0.044
Case 94	High (1)	High (1)	High (1)	High (1)	27	0.044
Case 95	Low (0.22)	High (1)	Medium (0.5)	Medium (0.5)	27	0.2095
Case 96	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	27	0.3243
Case 97	Low (0.22)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.375
Case 98	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	27	0.4645
Case 99	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	27	0.4645

Table 3.27 represents the descriptor cases used for the analysis of web data. To understand the behaviour of fuzzy load balancer (when it captures the web data), 33 cases (i.e. case 67 to case 99) are selected. These 33 cases gave the maximum coverage to check different levels at which the fuzzy load balancer needs to operate. Here the three levels can have value as low as 0.1 and as high as 1. It can be observed from the Table 3.27 that there are cases that covers the scenarios when in most cases the traffic load is low, delay sensitivity is also low, energy consumption is also low and link saturation is also low. The table further gives information on the cases where the traffic load is of medium level and so are the other input variables. Such cases are numbered from case 72 to case 77. The Table 3.27 show greater impact of energy consumption and link saturation. It can also be observed that from case 91, the rules get activated and invoked, and the process continues in almost all the cases till end.

3.6.1.2 5-Level Design of Fog Load Balancer

5-level design of fog load balancer is considered for sensor data, video data and web data. This leads to creation of 625 rules in case of 5-level fuzzification. These three types of data forms different test cases, that are described below.

(i) For Sensor Data: Table 3.28 shows 5-level descriptor test cases for sensor data.

Table 3.28: 5-Level Descriptors Case for analysis of Sensor data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 100	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 101	Low (0.1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 102	Low (0.1)	High (1)	Low (0.02)	Low (0)	0	0
Case 103	Low (0.1)	Low (0)	High (1)	Low (0)	0	0
Case 104	Low (0.1)	Low (0)	Low (0.02)	High (1)	0	0
Case 105	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 106	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 107	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 108	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 109	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 110	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 111	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 112	Low (0.1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 113	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 114	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 115	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 116	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 117	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 118	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 119	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 120	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 121	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 122	High (1)	High (1)	High (1)	Low (0)	0	0
Case 123	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 124	High (1)	High (1)	Medium (0.5)	Medium (0.5)	25	0.044
Case 125	High (1)	High (1)	Medium (0.5)	High (1)	45	0.044
Case 126	High (1)	High (1)	High (1)	Medium (0.5)	45	0.044
Case 127	High (1)	High (1)	High (1)	High (1)	27	0.044
Case 128	Low (0.1)	High (1)	Medium (0.5)	Medium (0.5)	200	0.1263
Case 129	Low (0.1)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.1675
Case 130	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	200	0.4906
Case 131	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.6022
Case 132	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	200	0.6363

From Table 3.28, it can be inferred that this arrangement is an advancement of the 3-level fuzzy logic design by adding more fuzzy rule levels. It was expected that the traffic management would become more precise due to tighter boundaries. But, experimental observations show otherwise. Following are the observations of 5-level design of fog load balancer for sensor data.

- There are four logical 'AND' operator conditions in most cases of fuzzy rules. By empirical observation and experimentation, it was found that in most cases these conditions are not fulfilled and outcome is 'false'. This shows that adding more rules and fuzzy levels has no advantage.
- There are few chances that the values of traffic load and delay sensitivity would remain zero or low state.
- It was also found that even when the fuzzy rule level is increased to 5-level, there is no additional activation of rules. In fact it adds lot of overhead in terms of number of rules to be maintained and executed.
- The activation of 5-level fuzzy rule starts happening after the fuzzy states of the factors overlap each other's boundaries. Cases 124 to 132 show influence of medium and high states of energy consumption, delay sensitivity and link saturation variables with respect to varying traffic load.
- The rules get invoked when the strength metric value is between 0.044 to 0.6363. For positive strength value, the number of rules get invoked. In fact the number of rules fired increases at faster rate. At the same time it can also be observed that the redundant rules are quite high in 5-level design as compared to 3-level design.

(ii) For Video Data: Table 3.29 shows 5-level descriptor test cases for video data.

Table 3.29: 5-Level Descriptors Case for analysis of Video data

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 133	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 134	Low (0.33)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 135	Low (0.33)	High (1)	Low (0.02)	Low (0)	0	0
Case 136	Low (0.33)	Low (0)	High (1)	Low (0)	0	0
Case 137	Low (0.33)	Low (0)	Low (0.02)	High (1)	0	0

Continued on next page

Table 3.29 – *Continued from previous page*

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 138	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 139	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 140	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 141	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 142	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 143	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 144	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 145	Low (0.33)	Low (0)	Low (0.02)	Low (0)	0	0
Case 146	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 147	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 148	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 149	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 150	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 151	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 152	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 153	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 154	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 155	High (1)	High (1)	High (1)	Low (0)	0	0
Case 156	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 157	High (1)	High (1)	Medium (0.5)	Medium (0.5)	75	0.044
Case 158	High (1)	High (1)	Medium (0.5)	High (1)	45	0.044
Case 159	High (1)	High (1)	High (1)	Medium (0.5)	45	0.044
Case 160	High (1)	High (1)	High (1)	High (1)	27	0.044
Case 161	Low (0.33)	High (1)	Medium (0.5)	Medium (0.5)	225	0.4132
Case 162	Low (0.33)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.421
Case 163	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	225	0.4932
Case 164	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	225	0.6022
Case 165	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.6022

Case 150 and case 151 (in Table 3.29), when there is low link saturation and the values of traffic load, delay sensitivity, energy consumption are in medium states, the flow of network remains non-congested. In such cases, the fuzzy load balancer does not intervene to fire any rule.

It can be seen in cases 154-156, higher traffic states (medium and high) along-with no saturation does not require invoking of any rule, as the overall quality of link is good and hence, fog network remains in self-managed state.

The cases 157 onwards are only the cases where the traffic load, delay sensitivity, energy consumption, link saturation comes in states of high and medium, and the strength metric of fuzzy rules gains traction. This traction leads to firing of rules between the range of 27 to 375.

(iii) For Web Data: Table 3.30 shows 5-level descriptor test cases for web data.

Table 3.30: 5-Level Descriptors Case for analysis of Web data

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 166	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 167	Low (0.22)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 168	Low (0.22)	High (1)	Low (0.02)	Low (0)	0	0
Case 169	Low (0.22)	Low (0)	High (1)	Low (0)	0	0
Case 170	Low (0.22)	Low (0)	Low (0.02)	High (1)	0	0
Case 171	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 172	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 173	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 174	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 175	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 176	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 177	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 178	Low (0.22)	Low (0)	Low (0.02)	Low (0)	0	0
Case 179	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 180	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 181	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 182	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 183	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 184	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 185	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 186	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 187	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 188	High (1)	High (1)	High (1)	Low (0)	0	0

Continued on next page

Table 3.30 – *Continued from previous page*

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 189	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 190	High (1)	High (1)	Medium (0.5)	Medium (0.5)	50	0.044
Case 191	High (1)	High (1)	Medium (0.5)	High (1)	30	0.044
Case 192	High (1)	High (1)	High (1)	Medium (0.5)	30	0.044
Case 193	High (1)	High (1)	High (1)	High (1)	18	0.044
Case 194	Low (0.22)	High (1)	Medium (0.5)	Medium (0.5)	225	0.2863
Case 195	Low (0.22)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.4075
Case 196	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	225	0.4932
Case 197	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	225	0.6022
Case 198	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	375	0.6022

The behaviour of fuzzy load balancer (with web data) is similar to the way it performed when it captures video data. The packet size is usually smaller in case of web as compared to video. Clearly, it can be observed that fuzzy load balancer does not get activated till case 189. This may be attributed to the fact that strength of the fuzzy rules is not enough nor the fuzzy algorithm computes separate position of the centroid. In other words, this can be said that the computations of fuzzy rules are getting overlapped. Further, from cases 190 to 193, it can be observed that when at-least two of the parameters are high and other two are at medium level, the rules get activated and invoked.

3.6.1.3 7-Level Design of Fog Load Balancer

7-level design of fog load balancer is considered for sensor data, video data and web data. The idea behind creating more levels of fuzzy boundaries is to have tighter control over the traffic load management. This leads to creation of 2401 rules in case of 7-level fuzzification. These three types of data forms different test cases, that are described below.

(i) For Sensor Data: Table 3.31 shows 7-level descriptor test cases for sensor data.

Table 3.31: 7-Level Descriptors Case for analysis of Sensor data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 199	Low (0)	Low (0)	Low (0.02)	Low (0)	0	0
Case 200	Medium (0.5)	Low (0)	Low (0.02)	Low (0)	0	0
Case 201	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 202	Low (0.1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 203	Low (0.1)	High (1)	Low (0.02)	Low (0)	0	0
Case 204	Low (0.1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 205	Low (0.1)	Low (0)	High (1)	Low (0)	0	0
Case 206	Low (0.1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 207	Low (0.1)	Low (0)	Low (0.02)	High (1)	0	0
Case 208	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 209	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 210	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 211	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 212	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 213	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 214	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 215	Low (0.1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 216	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 217	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 218	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 219	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 220	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 221	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 222	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 223	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 224	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 225	High (1)	High (1)	High (1)	Low (0)	0	0
Case 226	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 227	High (1)	High (1)	Medium (0.5)	Medium (0.5)	196	0.0225
Case 228	High (1)	High (1)	Medium (0.5)	High (1)	88	0.0225
Case 229	High (1)	High (1)	High (1)	Medium (0.5)	196	0.0225
Case 230	High (1)	High (1)	High (1)	High (1)	88	0.0225
Case 231	Low (0.1)	High (1)	Medium (0.5)	Medium (0.5)	196	0.06

Continued on next page

Table 3.31 – *Continued from previous page*

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 232	Low (0.1)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.068
Case 233	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	196	0.31
Case 234	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	147	0.31
Case 235	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.31

Following points have been observed for understanding the feasibility of 7-level design using Table 3.31.

- Tighter fuzzy boundaries with multi-values does not really help to split the traffic due to overlapping of fuzzy values with each other.
- Here also, same four factors (traffic load, delay sensitivity, energy consumption and link saturation) have been considered. Each factor is a part of the fuzzy rule equation that leads to activation of fuzzy rule to balance the incoming traffic data.
- It was also found that even when the fuzzy rule level is increased to 7-level, there is no additional activation of rules. In fact it adds lot of overhead in terms of number of rules to be maintained and executed.
- The activation of 7-level fuzzy rule starts happening, after the fuzzy states of the factors overlap each other's boundaries. This happens, in almost all cases from 227 to 235, when low, medium and high traffic load states influence the medium and high states of energy consumption, delay sensitivity and link saturation.
- It clearly shows that tighter fuzzy boundaries and level in lower states (cases having low and very-low values) do not need any intervention from fuzzy controller, and rules in such cases can be ignored to reduce overhead.
- The rules get invoked when the strength metric value is between 0.0225 to 0.31. Depending upon the strength value, the number of rules get invoked. In fact the number of rules fired increases at fast rate. At the same time it can also be observed that the redundant rules are quite high in 7-level design as compared to 3-level design and almost similar to 5-level design.

(ii) For Video Data: Table 3.32 shows 7-level descriptor test cases for video data.

Table 3.32: 7-Level Descriptors Case for analysis of Video data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 236	Low (0.33)	Low (0)	Low (0.02)	Low (0)	0	0
Case 237	Medium (0.5)	Low (0)	Low (0.02)	Low (0)	0	0
Case 238	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 239	Low (0.33)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 240	Low (0.33)	High (1)	Low (0.02)	Low (0)	0	0
Case 241	Low (0.33)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 242	Low (0.33)	Low (0)	High (1)	Low (0)	0	0
Case 243	Low (0.33)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 244	Low (0.33)	Low (0)	Low (0.02)	High (1)	0	0
Case 245	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 246	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 247	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 248	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 249	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 250	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 251	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 252	Low (0.33)	Low (0)	Low (0.02)	Low (0)	0	0
Case 253	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 254	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 255	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 256	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 257	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 258	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 259	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 260	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 261	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 262	High (1)	High (1)	High (1)	Low (0)	0	0
Case 263	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 264	High (1)	High (1)	Medium (0.5)	Medium (0.5)	196	0.0225
Case 265	High (1)	High (1)	Medium (0.5)	High (1)	84	0.0225
Case 266	High (1)	High (1)	High (1)	Medium (0.5)	196	0.0225
Case 267	High (1)	High (1)	High (1)	High (1)	88	0.0225
Case 268	Low (0.33)	High (1)	Medium (0.5)	Medium (0.5)	196	0.0608

Continued on next page

Table 3.32 – *Continued from previous page*

Case No.	**Traffic Load**	**Delay Sensitivity**	**Energy Consumption**	**Link Saturation**	**Number of Rules Fired**	**Strength**
Case 269	Low (0.33)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.31
Case 270	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	196	0.31
Case 271	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	147	0.31
Case 272	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.31

From Table 3.32, following points have been inferred.

- The total number of rules that are constructed initially were based on all possible permutation of four factors (traffic load, delay sensitivity, energy consumption and link saturation). But these all requires review of the rules due to high overhead in running all the permutations. The reviewing and multi-evaluations of rules helped to eliminate about 2058 rules, as it can be seen that lower fuzzy states needs no interference from fuzzy controller.
- The behaviour of fuzzy load balancer is consistent and similar to the 3-level and 5-level fuzzy design in cases where the traffic load is low, medium or high (cases 34-66 and cases 133-165), the delay sensitivity is low and medium (cases 34-35, 37-39, 41-45, 47-54, 62, 65-66, 133-134, 136-138, 140-144, 146-153, 161 and 164-165), the energy consumption is low (cases 40, 43-46, 49-50, 53-54, 57, 133-135, 137-139, 142-145, 148-149, 152-153 and 156), and link saturation is less than average value (cases 40-42, 45-48, 51-52, 55-57, 133-136, 138-141, 144-147, 150-151 and 154-156).
- It can be observed from the Table 3.32 that the maximum number of rules that are fired are from the cases when three variables are in the medium level and one of the variable is in low or high level. In such cases, the number of rules fired may varies from 147 to 343 (case 268, 270 to 272).

(iii) For Web Data: Table 3.33 shows 7-level descriptor test cases for web data.

Table 3.33: 7-Level Descriptors Case for analysis of Web data

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 273	Low (0.22)	Low (0)	Low (0.02)	Low (0)	0	0
Case 274	Medium (0.5)	Low (0)	Low (0.02)	Low (0)	0	0
Case 275	High (1)	Low (0)	Low (0.02)	Low (0)	0	0
Case 276	Low (0.22)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 277	Low (0.22)	High (1)	Low (0.02)	Low (0)	0	0
Case 278	Low (0.22)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 279	Low (0.22)	Low (0)	High (1)	Low (0)	0	0
Case 280	Low (0.22)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 281	Low (0.22)	Low (0)	Low (0.02)	High (1)	0	0
Case 282	Medium (0.5)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 283	Medium (0.5)	High (1)	Low (0.02)	Low (0)	0	0
Case 284	Medium (0.5)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 285	Medium (0.5)	Low (0)	High (1)	Low (0)	0	0
Case 286	Medium (0.5)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 287	Medium (0.5)	Low (0)	Low (0.02)	High (1)	0	0
Case 288	High (1)	Medium (0.5)	Low (0.02)	Low (0)	0	0
Case 289	Low (0.22)	Low (0)	Low (0.02)	Low (0)	0	0
Case 290	High (1)	Low (0)	Medium (0.5)	Low (0)	0	0
Case 291	High (1)	Low (0)	High (1)	Low (0)	0	0
Case 292	High (1)	Low (0)	Low (0.02)	Medium (0.5)	0	0
Case 293	High (1)	Low (0)	Low (0.02)	High (1)	0	0
Case 294	Medium (0.5)	Medium (0.5)	Medium (0.5)	Low (0)	0	0
Case 295	Medium (0.5)	Medium (0.5)	High (1)	Low (0)	0	0
Case 296	Medium (0.5)	Medium (0.5)	Low (0.02)	Medium (0.5)	0	0
Case 297	Medium (0.5)	Medium (0.5)	Low (0.02)	High (1)	0	0
Case 298	High (1)	High (1)	Medium (0.5)	Low (0)	0	0
Case 299	High (1)	High (1)	High (1)	Low (0)	0	0
Case 300	High (1)	High (1)	Low (0.02)	Low (0)	0	0
Case 301	High (1)	High (1)	Medium (0.5)	Medium (0.5)	196	0.0225
Case 302	High (1)	High (1)	Medium (0.5)	High (1)	84	0.0225
Case 303	High (1)	High (1)	High (1)	Medium (0.5)	196	0.0225
Case 304	High (1)	High (1)	High (1)	High (1)	84	0.0225
Case 305	Low (0.22)	High (1)	Medium (0.5)	Medium (0.5)	196	0.14

Continued on next page

Table 3.33 – *Continued from previous page*

Case No.	Traffic Load	Delay Sensitivity	Energy Consumption	Link Saturation	Number of Rules Fired	Strength
Case 306	Low (0.22)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.188
Case 307	Medium (0.5)	High (1)	Medium (0.5)	Medium (0.5)	196	0.31
Case 308	Medium (0.5)	Medium (0.5)	Medium (0.5)	High (1)	147	0.31
Case 309	Medium (0.5)	Medium (0.5)	Medium (0.5)	Medium (0.5)	343	0.31

Following points have been observed from the Table 3.33.

- The cases 301-309 show the fuzzy rules against strength when the fuzzy states gets more pronounced in terms of magnitude. Higher fuzzy states need more intervention from the fuzzy load balancer. This is happening in all these nine cases of web data.
- From this, it can be inferred that many rules are not useful and are definitely redundant.
- It can be observed from the Table 3.33 that the maximum strength which the fuzzy rules may acquire is 0.31 and this typically happens when at-least three variables are at medium level (case 305, 307 to 309).

3.6.2 Comparative Analysis of 3-level, 5-level and 7-level Design Scenarios

Number of rules would suffice to make an efficient level design of load balancer is a question that needs to be addressed before finally implementing the fog load balancer. Figure 3.21 gives a comparative analysis of designs with respect to their relevance and activation for traffic management.

A fuzzy controller works well when classical mathematics cannot consider to imprecise the nature of process. Also, when the computation of threshold or decision points are influenced by multiple factors, it is hard to define precise 'if-then-else' rules, especially if the system is dynamic and have erratic influxes of factor values (traffic load, delay sensitivity, energy consumption and link saturation).

It can be seen from the histogram, represented in Figure 3.21 as well as Tables 3.25 to 3.33 that adding more rules with 'AND' logical operator conditions and levels have no advantage. Thus, 3-level design is the optimal design out of the three designs. There are three reasons for this argument.

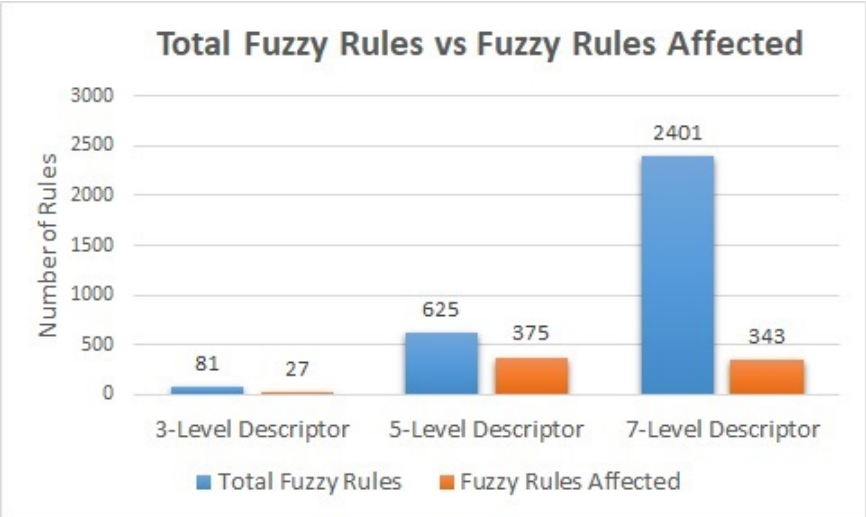


Figure 3.21: Comparative Analysis of 3-level, 5-level and 7-level Designs

- It has minimum rules and has less number of redundant rules.
- This design takes less time as it involves less number of levels and rules.
- It has minimum overheads.

After evaluation, validation has been carried out with comparative analysis.

3.7 Validation of Design Scenarios of Fog Load Balancer

In this subsection, the design of fuzzy load balancer is validated (using 5K-fold cross validation method) on the basis of four metrics i.e. traffic load, delay sensitivity, energy consumption and link saturation. In this, redundant rules are those rules that have zero strength value and consequently number of rules affected are also zero. The section gives a comparative analysis and insights from state of the art work done in this context.
There are three kinds of traffic services (data, video and web) that have been evaluated and 3-level, 5-level and 7-level designs are simulated. This section gives a comparative analysis of the three designs.

Table 3.34: Sensor Data

Fuzzy Design level	Total number of rules	Number of rules Affected	Strength Value
3-level	33	27	0.199
5-level	33	165	0.2443
7-level	37	200	0.1275

It can be observed from the Table 3.34 that number of rules that are affected in case of 3-level, 5-level and 7-level are 27, 165 and 200. From this, it can be established that most

of the rules are overlapping in each case. By increasing the number of fuzzy intervals, on which the Fuzzy Rule engine makes decisions, sensitivity does not get better. It seems that large number of rules are redundant as well.

Table 3.35: Video Data

Design level	Total number of rules	Number of rules Affected	Strength Value
3-level	33	26	0.2620
5-level	33	180	0.3009
7-level	37	199	0.1545

It can be seen from Table 3.35 that as design level increases, number of rules affected also increases.

Table 3.36: Web Data

Design level	Total number of rules	Number of rules Affected	Strength Value
3-level	33	27	0.2238
5-level	33	172	0.2853
7-level	37	198	0.1497

The strength value is lowest in case of 7-level (in Table 3.36) which means there is a weak argument for having more rules and fuzzy levels. The overlap is also high in case of 7-level as well as overhead is going to be the highest, given the number of rules been affected.

3.7.1 Validation of 3-Level Design of Fog Load Balancer

Validation of 3-level design of fog load balancer is considered for sensor data, video data and web data. These three types of data are validated and their analysis is discussed below.

(i) For Sensor Data:

Table 3.37 shows that the number of cases that have been validated using 3-level design for sensor data. The table shows that there are three classes i.e. Overload (OL), Normal-load (NL) and Under-load (UL), that have 7, 18 and 8 cases respectively, thus total of 33 cases were evaluated and validated. The outcome of all these cases is mentioned in Table 3.38.

Following are the observation from the Table 3.38.

Table 3.37: 3-level Cases for Sensor Data

Abbreviation	Meaning	Number of cases
OL	Overload	7
NL	Normal-load	18
UL	Under-load	8
Total Number of Cases		33

- The maximum numbers (3) of incorrect classifications occurring in NL class. Lowest number (2) of incorrect classifications is happening in OL and UL class. Instead of classifying instances as NL, the 1 instance is classified as OL and 2 instances are classified as UL. From this it can be observed that NL's micro recall and precision are least.
- The recall value in the case of NL is lowest and the recall and precision values of most the classes are above 0.75. This is a case of high precision and high recall. It means that there is less percentage of false positives.
- The accuracy of the system is close to 0.7879. This means that the maximum possible accuracy is 78.79%, that can be observed after maximum possible optimization of fuzzy load balancer.
- The average recall should be atleast 0.80 for the algorithm to make least number of mistakes and be sensitive to changes in the patterns of traffic. The value is just 0.2% less.

(ii) For Web Data:

Table 3.39 shows that the number of cases that have been validated using 3-level design for web data. The table shows that there are three classes i.e. Overload (OL), Normal-load (NL) and Under-load (UL), that have 15, 12 and 6 cases respectively, thus forming 33 cases in total. The outcome of all these cases is mentioned in Table 3.40.

Following are the observation from the Table 3.40.

- Maximum numbers (3) of incorrect are happening in NL and OL classes. Lowest number (2) of incorrect classifications is happening in OL and UL classes. Instead of classifying instances as NL, the 1 instance is classified as OL and 2 instances are classified as UL. From this, it can be observed that NL's micro recall and precision is least.
- The recall and precision values of most the classes are above 0.75. High precision and high recall values mean that there is less percentage of false positives.

Table 3.38: 3-level Design for Sensor Data

	OL	**NL**	**UL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	5	2	0	TP=5; FN=2	2	0.714	0.833	0.7908	0.765	0.777	0.7879
NL	1	15	2	TP=1; FN=3	3	0.833	0.7895				
UL	0	2	6	TP=6; FN=2	2	0.75	0.75				
	TN=5; FP=1	TN=15; FP=4	TN=6; FP=2								

Table 3.39: 3-level Cases for Web Data

Abbreviation	Meaning	Number of cases
OL	Overload	15
NL	Normal-load	12
UL	Under-load	6
Total Number of Cases		33

- The accuracy of the system is close to 0.7575. This means that when the fuzzy design level of load balancer is implemented, the maximum possible accuracy is 75.75%.
- The value of F1-score helps to analyze false negatives and false positives proportion. This metric is helpful in current work as this work do not have equal number of cases for each class (OL, NL and UL). The value is quite less (0.26) from the best possible value.

(iii) For Video Data:

Table 3.41 shows that the number of cases that have been validated using 3-level design for video data. Here also we have three classes i.e. Overload (OL), Normal-load (NL) and Under-load (UL), that have 20, 8 and 5 cases respectively, thus forming 33 cases in total. The outcome of all these cases is mentioned in Table 3.42.

Following are the observation from the Table 3.42.

- The maximum numbers (4) of incorrect are occurring in OL class. Lowest number (1) of incorrect classifications is happening in UL class. Instead of classifying instances as NL, the 1 instance is classified as OL and UL each. From this it can be inferred that NL's micro recall and precision is going to least.
- It can be observed that the recall value in the case of NL is lowest (0.75) and the recall and precision values of most the classes are above 0.75.
- The accuracy of the system is close to 0.787878. This accuracy was computed on the basis of 33 cases, as shown in Table 3.41.
- The value of F1-score is about 0.23 less from the best possible value. Comparing all fuzzy design levels considered in our research work, maximum trade-off is achieved by this design.

Table 3.40: 3-level Design for Web Data

	OL	**NL**	**UL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	12	3	0	TP=12; FN=3	3	0.80	0.92	0.7433	0.74	0.7416	0.7575
NL	1	9	2	TP=9; FN=3	3	0.75	0.64				
UL	0	2	4	TP=4; FN=2	2	0.67	0.67				
	TN=12; FP=1	TN=9; FP=5	TN=4; FP=2								

Table 3.41: 3-level Cases for Video Data

Abbreviation	Meaning	Number of cases
OL	Overload	20
NL	Normal-load	8
UL	Under-load	5
Total Number of Cases		33

3.7.2 Validation of 5-Level Design of Fog Load Balancer

Validation of 5-level design of fog load balancer is considered for sensor data, video data and web data. These three types of data are validated and their analysis is discussed below.

(i) For Sensor Data:

Table 3.43 shows that the number of cases that have been validated using 5-level design for sensor data. The table shows that there are five classes i.e. Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 3, 4, 21, 2 and 3 cases respectively, thus forming 33 cases in total. The outcome of all these cases is mentioned in Table 3.44.

Following are the observation from the Table 3.44.

- The maximum numbers (6) of incorrect are occurring in NL class. Lowest number (0) of incorrect classifications is happening in NoL class. This means system is more accurate for NoL class as it has not performed any mistake in finding its optimal results. Instead of classifying instances as UL, PL and OL, the 1 instance is classified as UL and 2 instances are classified as PL and OL. From this it can be observed that UL's micro recall and precision are least.
- It can be observed that the recall value in the case of OL is lowest (0.33) whereas the recall and precision values for most of the classes are above 0.33, which is not as expected.
- The accuracy of the system is close to 0.66667. This value is not a good value in terms of performance.
- The mean recall (0.608) is quite below than the expected recall value.
- The value of F1-score (0.5617) is quite lower as compared with 3-level design evaluations. Thus, this design should not be considered for the task of load balancing.

(ii) For Web Data:

Table 3.42: 3-level Design for Video Data

	OL	**NL**	**UL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	16	4	0	TP=16; FN=4	4	0.80	0.94	0.76	0.7833	0.7715	0.7878
NL	1	6	1	TP=6; FN=2	2	0.75	0.54				
UL	0	1	4	TP=4; FN=1	1	0.80	0.80				
	TN=16; FP=1	TN=6; FP=5	TN=4; FP=1								

Table 3.43: 5-level Cases for Sensor Data

Abbreviation	Meaning	Number of cases
OL	Overload	3
PL	Peak-load	4
NL	Normal-load	21
UL	Under-load	2
NoL	No-load	3
Total Number of Cases		33

Table 3.45 shows that the number of cases that have been validated using 5-level design for web data. The table shows that there are five classes i.e. Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 11, 10, 6, 4 and 2 cases respectively, thus forming 33 cases in total. The outcome of all these cases is mentioned in Table 3.46.

Following are the observation from the Table 3.46.

- The maximum numbers (3) of incorrect are occurring in PL class. Lowest number (0) of incorrect classifications is happening in NoL class.
- The recall value in the case of NL is lowest and the recall and precision values of most the classes are above 0.68. High precision and high recall values mean that there is less percentage of false positives.
- The accuracy of the system is close to 0.75. This value is somewhat not satisfactory, as higher values were expected.
- The average recall is not above 0.90, which means there are fair chances that the algorithm will make mistakes in classifications as it is 0.78. A value of 0.90 would have been better in context of the problem undertaken.

(iii) For Video Data:

Table 3.47 shows that the number of cases that have been validated using 5-level design for video data. The table shows that there are five classes i.e. Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 12, 12, 5, 2 and 2 cases respectively, thus forming 33 cases in total. The outcome of all these cases is mentioned in Table 3.48.

Following are the observation from the Table 3.48.

- The maximum numbers (3) of incorrect are occurring in PL class. Lowest number (0) of incorrect classifications is happening in NoL class.

Table 3.44: 5-level Design for Sensor Data

	OL	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	1	2	0	0	0	TP=1; FN=2	2	0.33	0.33	0.522	0.608	0.5617	0.66667
PL	2	2	0	0	0	TP=2; FN=2	2	0.50	0.33				
NL	0	2	15	4	0	TP=15; FN=6	6	0.71	1.00				
UL	0	0	0	1	1	TP=1; FN=1	1	0.5	0.2				
NoL	0	0	0	0	3	TP=3; FN=0	0	1.00	0.75				
	TN=1; FP=2	TN=2; FP=4	TN=15; FP=0	TN=1; FP=4	TN=3; FP=1								

Table 3.45: 5-level Cases for Web Data

Abbreviation	Meaning	Number of cases
OL	Overload	11
PL	Peak-load	10
NL	Normal-load	6
UL	Under-load	4
NoL	No-load	2
Total Number of Cases		33

- The UL class has 1 incorrect classification with average levels of recall and precision.
- The recall value in the case of UL is lowest and the recall and precision values of most the classes are above 0.33. High precision and high recall values mean that there is less percentage of false positives.
- The accuracy of the system is close to 0.7575. This value is not encouraging for evaluator to take decision for this design.
- The OL class has the maximum micro-precision and recall (0.83). It is always desired in context of current research problem that machine learning models must provide high levels of micro-recall. This is because it is always better to classify the incoming traffic unit as congestion or traffic jam causing element than as a packet that does not cause traffic problems. The advantage is high level of sensitivity and accuracy to handle the traffic management issues.
- F1-score is the weighted average that is computed using harmonic mean. The best value should be 1, whereas in this case, the value is 0.7175.

3.7.3 Validation of 7-Level Design of Fog Load Balancer

The validation of 7-level design has been computed for three types of data: sensor data, video data and web data. These three types of data are validated and their analysis is discussed below.

(i) For Sensor Data:

Table 3.49 gives information about number of cases validated using 7-level design for sensor data. The table shows that there are seven classes i.e. Traffic Jam (TJ), Congestion (CL), Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 2, 3, 9, 8, 8, 5 and 2 cases respectively, thus

Table 3.46: 5-level Design for Web Data

	OL	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	9	2	0	0	0	TP=9; FN=2	2	0.81	0.81				
PL	2	7	1	0	0	TP=7; FN=3	3	0.70	0.70	0.746	0.786	0.7654	0.7575
NL	0	1	4	1	0	TP=4; FN=2	2	0.67	0.80				
UL	0	0	0	3	1	TP=3; FN=1	1	0.75	0.75				
NoL	0	0	0	0	2	TP=3; FN=0	0	1.00	0.67				
	TN=9; FP=2	TN=7; FP=3	TN=4; FP=1	TN=3; FP=1	TN=2; FP=1								

Table 3.47: 5-level Cases for Video Data

Abbreviation	Meaning	Number of cases
OL	Overload	12
PL	Peak-load	12
NL	Normal-load	5
UL	Under-load	2
NoL	No-load	2
Total Number of Cases		33

forming 37 cases in total. The outcome of all these cases is mentioned in Table 3.50.

Following are the observation from the Table 3.50.

- The maximum numbers (9) of incorrect are occurring in NL class. Lowest number (0) of incorrect classifications is happening in NoL class.
- The recall value in the case of CL is lowest (0.33) whereas the recall and precision values of most the classes are above 0.33. This is not satisfactory.
- The accuracy of the system is 0.6216. This means that when the fuzzy design level of load balancer is implemented the maximum possible accuracy is 62.16%. These observations are based on 37 cases, as mentioned in Table 3.49.
- Only in case of NoL, the system has maximum level of recall whereas in most cases, its value is below 0.68.
- At the same time, it is observed that PL has the maximum precision value. It seems the algorithm is unable to classify the class data point with sufficient amount of accuracy and precision. Almost all class data points have incorrect classifications.

(ii) For Web Data:

Table 3.51 shows that the number of cases that have been validated using 7-level design for web data. The table shows that there are seven classes i.e. Traffic Jam (TJ), Congestion (CL), Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 2, 3, 11, 10, 8, 2 and 1 cases respectively, thus forming 37 cases in total. The outcome of all these cases is mentioned in Table 3.52.

Following are the observation from the Table 3.52.

- The maximum numbers (3) of incorrect are occurring in OL and PL classes. Lowest number (0) of incorrect classifications is happening in NoL class.

Table 3.48: 5-level Design for Video Data

	OL	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifications**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
OL	10	2	0	0	0	TP=10; FN=2	2	0.83	0.83	0.70	0.736	0.7175	0.7575
PL	2	9	1	0	0	TP=9; FN=3	3	0.75	0.75				
NL	0	1	3	1	0	TP=3; FN=2	2	0.60	0.75				
UL	0	0	0	1	1	TP=1; FN=1	1	0.50	0.50				
NoL	0	0	0	0	2	TP=2; FN=0	0	1.00	0.67				
	TN=10; FP=2	TN=9; FP=3	TN=3; FP=1	TN=1; FP=1	TN=2; FP=1								

Table 3.49: 7-level Cases for Sensor Data

Abbreviation	Meaning	Number of cases
TJ	Traffic Jam	2
CL	Congestion	3
OL	Overload	9
PL	Peak-load	8
NL	Normal-load	8
UL	Under-load	5
NoL	No-load	2
Total Number of Cases		37

- The recall value in the case of CL is lowest (0.33) whereas the recall and precision values of most the cases are above 0.33.
- The accuracy of the system is 0.6757.
- Moderate levels of recall and precision can be observed in case of TJ and UL classes and as expected NoL has high recall values because fuzzy loader has ease in computing, when no or limited data is there.
- Low recall (0.33) and low precision (0.25) levels can be observed in case of CL class. From this, it can be inferred that positives predictions are low for the class as well as for such trained data instances.
- High recall is always desirable in context of our research work because all we do not want. It is better to mark the class data point to their respective class and also to a class that may not lead to traffic congestion or jam. Hence, it is better to classify a data point as a 'traffic packet' that creates higher levels of load i.e. overload, peak-load etc., so that action can be taken to decongestant with higher levels of sensitivity. It can be observed that in case of NoL and NL, there high recall value, but OL class have higher level of precision 0.80 correspondingly.
- F1-score value is lower than 3-level design and is slightly higher than 5-level design. Thus, this is too far from the best possible value.

(iii) For Video Data:

Table 3.53 shows that the number of cases that have been validated using 7-level design for video data. The table shows that there are seven classes i.e. Traffic Jam (TJ), Congestion (CL), Overload (OL), Peak-load (PL), Normal-load (NL), Under-load (UL) and No-load (NoL), that have 3, 2, 11, 9, 8, 2 and 2 cases respectively, thus forming 37 cases in total. The outcome of all these cases is mentioned in Table 3.54.

Table 3.50: 7-level Design for Sensor Data

	TJ	**CL**	**OL**	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifi-cations**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
TJ	1	1	0	0	0	0	0	TP=1; FN=1	1	0.50	0.50	0.5857	0.6228	0.6037	0.6216
CL	1	1	1	0	0	0	0	TP=1; FN=2	2	0.33	0.25				
OL	0	2	6	1	0	0	0	TP=6; FN=3	3	0.67	0.75				
PL	0	0	1	5	2	0	0	TP=5; FN=3	3	0.63	0.71				
NL	0	0	0	1	5	2	0	TP=5; FN=3	3	0.63	0.63				
UL	0	0	0	0	1	3	1	TP=3; FN=2	2	0.60	0.60				
NoL	0	0	0	0	0	0	2	TP=2; FN=0	0	1.00	0.66				
	TN=1; FP=1	TN=1; FP=3	TN=6; FP=2	TN=5; FP=2	TN=5; FP=3	TN=3; FP=2	TN=2; FP=1								

Table 3.51: 7-level Cases for Web Data

Abbreviation	Meaning	Number of cases
TJ	Traffic Jam	2
CL	Congestion	3
OL	Overload	11
PL	Peak-load	10
NL	Normal-load	8
UL	Under-load	2
NoL	No-load	1
Total Number of Cases		37

Following are the observation from the Table 3.54.

- From Table 3.54, it is apparent that fuzzy algorithm is making mistakes in classification in almost every case except when there is no traffic load. The maximum number of incorrect classifications are in OL and NL.
- Low precision and low recall means that CL and UL class data point (positive values) has never predicted in majority and at the same time, few of them predicted correctly.
- The micro-recall of PL is maximum (0.78) besides NoL (1.00) among all other classes.
- The CL and UL classes have similar value of micro-recall and at the same time, precests are also low. In both cases, low precision and low recall value can be observed. For this reason, this design should not be considered for real-time applications.
- The TJ and NL classes have 0.67 and 0.63 micro-recall. However, the OL (0.72) and PL classes have higher levels of micro-precision.
- In case of TJ, moderate level of recall and high precision can be observed. This means that significant proportion of the data points that are positives and have been predicted correctly.
- The F1-score values are harmonic mean of the recall and precision values. This helps to find the trade-off between true-positives with respect to total instances. It can be observed that value of F1-score (0.67) is not fair enough.

Table 3.52: 7-level Design for Web Data

	TJ	**CL**	**OL**	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifi-cations**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
TJ	1	1	0	0	0	0	0	TP=1; FN=1	1	0.50	0.50	0.5828	0.6429	0.6114	0.6757
CL	1	1	1	0	0	0	0	TP=1; FN=2	2	0.33	0.25				
OL	0	2	8	1	0	0	0	TP=8; FN=3	3	0.72	0.80				
PL	0	0	1	7	2	0	0	TP=7; FN=3	3	0.70	0.78				
NL	0	0	0	1	6	1	0	TP=6; FN=2	2	0.75	0.75				
UL	0	0	0	0	0	1	1	TP=1; FN=1	1	0.50	0.50				
NoL	0	0	0	0	0	0	1	TP=1; FN=0	0	1.00	0.50				
	TN=1; FP=1	TN=1; FP=3	TN=8; FP=2	TN=7; FP=2	TN=6; FP=2	TN=1; FP=1	TN=1; FP=1								

Table 3.53: 7-level Cases for Video Data

Abbreviation	Meaning	Number of cases
TJ	Traffic Jam	3
CL	Congestion	2
OL	Overload	11
PL	Peak-load	9
NL	Normal-load	8
UL	Under-load	2
NoL	No-load	2
Total Number of Cases		37

3.7.4 Comparative Analysis of 3-level, 5-level and 7-level Fuzzy Load Balancer Designs

In traffic management, security and antivirus related studies, the preferred evaluation method/metrics must be based on full database instances scan. It is always desired that an instance be put into a suspicious class by classifier in case the data point lies on the fence. In simple words, data point must be classified too closer to the class that helps to keep congestion away in the fog network.

Because of this, it is desired that the design should have atleast 0.80 to 0.90 recall value. But it can be observed from Table 3.55 this value is not obtainable. Only 3-level design is able to obtain value closer to 0.80 that to in case of video. But a similar performance can be seen in 5-level design (0.78) for web data. In case of 3-level sensor and web data, the recall value is highest among all designs.

It can also be observed that the precision values in case of 3-level design (sensor, web and video) are highest among all other designs. Consequently, the ratio of recall and precision impacts the F1-score. This value is also higher in all cases of 3-level design.

Table 3.55 clearly shows that accuracy of 3-level design is again highest. This is consistent performance when we observe the values of recall, precision and F1-score.

3.8 Summary and Conclusion

The main aim of this work is to produce software defined traffic splitting for load balancing in fog networks. Fuzzy based approaches have been evaluated using different design levels (3-level, 5-level and 7-level) that are modeled in fog environment, estimating the capabilities of fuzzy controller. Analysis is carried out while focusing on the network strategies and parameters for their traffic classes (sensor, web and video).

Uncertainties in the traffic flow and imprecision in determining the current status of multiple routing paths are the challenges that need to be addressed. Many times the fuzzy

Table 3.54: 7-level Design for Video Data

	TJ	**CL**	**OL**	**PL**	**NL**	**UL**	**NoL**		**Incorrect classifi-cations**	**Micro-R**	**P**	**Average P**	**Average R**	**F**	**A**
TJ	2	1	0	0	0	0	0	TP=2; FN=1	1	0.67	1.00	0.6657	0.6857	0.6756	0.7027
CL	0	1	1	0	0	0	0	TP=1; FN=1	1	0.50	0.25				
OL	0	2	8	1	0	0	0	TP=8; FN=3	3	0.72	0.80				
PL	0	0	1	7	1	0	0	TP=7; FN=2	2	0.78	0.78				
NL	0	0	0	1	5	2	0	TP=5; FN=3	3	0.63	0.83				
UL	0	0	0	0	0	1	1	TP=1; FN=1	1	0.50	0.33				
NoL	0	0	0	0	0	0	2	TP=2; FN=0	0	1.00	0.67				
	TN=2; FP=0	TN=1; FP=3	TN=8; FP=2	TN=7; FP=2	TN=5; FP=1	TN=1; FP=2	TN=2; FP=1								

Table 3.55: Comparative Analysis of 3-level, 5-level and 7-level Fuzzy Load Balancer Designs

Design level	Type of Traffic	Precision	Recall	F1-Score	Accuracy
3-level	Sensor Data	0.7908	0.765	0.777	0.7878
	Web Data	0.7433	0.74	0.7416	0.7575
	Video Data	0.76	0.7833	0.7715	0.7878
5-level	Sensor Data	0.522	0.608	0.5617	0.6666
	Web Data	0.746	0.786	0.7654	0.7575
	Video Data	0.70	0.736	0.7175	0.7575
7-level	Sensor Data	0.5857	0.6228	0.6037	0.6216
	Web Data	0.5828	0.6429	0.6114	0.6757
	Video Data	0.6657	0.6857	0.6756	0.7027

system is overwhelmed with overlapping and inconsistent fuzzy rules. These conditions need careful evaluation and selection of fuzzy controller for load balancing, that too with a feedback loop. Fuzzy controllers can be designed using different levels. The quest in this research work has been to find the most optimal design that has minimum overheads in terms of rule-base, for controlling the traffic in fog zone. The findings of this work show that the 3-level design works better as compared to 5-level and 7-level fuzzy traffic controller. This can be attributed to the fact that more fuzzy levels lead to overlapping and irrelevant rules. The advantage using 3-level model is that engineers/designers do not need to spend time in design and selection of rules from large permutation of rules. Research in this chapter shows that as the fuzzy levels and boundaries increase, there is a need to design more complex 'if-then-else conditions' and lot of unnecessary rules that gets added to the fuzzy rule base. The 3-level fuzzy logic system is easy and faster to implement.

These research outcomes suggest that the energy efficient load balancing at the edge can be done with the help of intelligent traffic management algorithms. These algorithms can help in distribution of tasks as well as execution of tasks. The approach of building 'fog controller', 'cloud controller' and 'mobile data controller' is similar to the act of 'human decision making'. As humans understand the things in terms of linguistic expressions, these controllers understand the current conditions using underlying fuzzy mathematical functions. Hence, these software defined controllers are able to consider both aspects of the payload management: the conformance aspect (where they need to conform to the green standards, power budgeting etc.) and the demand aspect (computational and data intensive service demand). Finally, we can say that bifurcation of payload management has helped in increasing the efficiency of full network.

Chapter 4

Optimization of Fog Load Balancer Using Energy-Efficient Algorithms

In this chapter, optimization algorithms namely Teaching Learning Based Optimization (TLBO), Stimulated Annealing (SA) and proposed algorithm Modified Teaching Learning Based Optimization (MTLBO) have been discussed. Various test cases have been considered with quality of service parameters for implementation and comparison of these three algorithms. Evaluation has also been carried out for the successful testing of these algorithms for fog load balancer. The MTLBO has been used for optimization of traffic management in fog load balancer, discussed in the previous chapter.
This chapter consists of seven sections. Section 4.1 discusses TLBO followed by Section 4.2 that deals with SA. Section 4.3 describes proposed algorithm MTLBO, and its implementation is discussed in Section 4.4. Analysis of proposed algorithm and comparative study of these three algorithms are given in subsequent Sections 4.5 and 4.6. The Section 4.7 gives the summary and conclusion of the chapter.

4.1 Teaching Learning Based Optimization (TLBO)

Teaching Learning Based Optimization (TLBO) is an optimization algorithm that works in two phases to arrive at optimal solution. The first phase is the 'Teacher Phase' and second one is the 'Learner Phase' [196, 197]. To initiate with the optimization process, the design variables are defined with their ranges. Formally, this approach can be understood in terms of a combinatorial problems of optimization as follows:

(i) Let I be set of Instances that define the problem.

(ii) $f(x)$ is set of feasible solutions that need to be evaluated for given values of x.

(iii) y denotes the solution or a measure $m(x,y)$ of x found from feasible solution.

(iv) g is the objective function of the optimization that defines the aim of the optimization, which may be maximization or minimization.

(v) The optimization problem A is a set of four parts that is called a Quadruple, $A = (I, f, m, g)$, that need to be solved in two phases according to the teaching learning analogy.

(vi) Let m_r be the mean result, that teacher wants to influence out of the class.

(vii) Let s be the number of subjects to be studied by a student.

(viii) Let i be the number of iterations between the teacher-student to maximize the average marks avg_m of the class.

(ix) Let X be the number of learners (population).

4.1.1 Teacher Phase

(i) Teacher is the entity that learns the maximum and is the best person X_{best} out of the population.

(ii) Decision variable 'Teaching Factor', denoted by T_f, decides whether there is a need to change learning rate or moderation rate. The values of T_f is randomly generated and selected to find optimal conditions for finding the feasible solution. Here, $T_f = \{1 \mid 2\}$.

(iii) Based on the existing solution, next best solution is searched, and arrive at decision to terminate in i number of iterations before starting the Learner Phase.

4.1.2 Learner Phase

(i) In this phase, interaction is handled among themselves. The algorithm randomly select two learners P and Q.

(ii) Evaluation of P and Q is carried out as per the goal or objective function.

(iii) The solution is accepted based on minimize or maximize goal, if it gives better function value. Equation 4.1 and 4.2 are used for minimization, equation 4.3 and 4.4 are used for maximization.

$$X^{"}_{j,P,i} = X^{'}_{j,P,i} + r_i(X^{'}_{j,P,i} - X^{'}_{j,Q,i}), \qquad if X^{'}_{total-P,i} < X^{'}_{total-Q,i}\,, \tag{4.1}$$

$$X^{"}_{j,P,i} = X^{'}_{j,P,i} + r_i(X^{'}_{j,Q,i} - X^{'}_{j,P,i}), \qquad if X^{'}_{total-Q,i} < X^{'}_{total-P,i}\,, \tag{4.2}$$

$$X^{"}_{j,P,i} = X^{'}_{j,P,i} + r_i(X^{'}_{j,P,i} - X^{'}_{j,Q,i}), \qquad if X^{'}_{total-Q,i} < X^{'}_{total-P,i}\,, \tag{4.3}$$

$$X^{"}_{j,P,i} = X^{'}_{j,P,i} + r_i(X^{'}_{j,Q,i} - X^{'}_{j,P,i}), \qquad if X^{'}_{total-P,i} < X^{'}_{total-Q,i}\,, \tag{4.4}$$

where P and Q are randomly selected learners from the class, r_i is the teaching factor in the range $[0,1]$ and $X^{'}$ represents the mean of respective randomly selected learner.

Algorithm 3 gives the steps for TLBO algorithm.

Algorithm 3 Teaching Learning Based Optimization (TLBO)

Input: I,g,m,n,s,i,m_r, D(number of dimensions)
Output: y ▷ An Optimal Solution
Objective Function: f(x) $\rightarrow$ g ▷ Minimization or Maximization

initialize n, s and D
initialize learners X and evaluate all learners X
assign the best learner X_{best} as Teacher and the mean of all learners as Mean
while stopping condition not met **do**
    for each learner X_i of the class **do** ▷ Teacher Phase
        modify learner X_i according to equations 4.1, 4.2, 4.3 or 4.4
        accept $newX_i$ if f($newX_i$) is better than f(X_i)
    end for
    for each learner X_i of the class **do** ▷ Learner Phase
        randomly select one learner X_i, such that $i \neq k$
        modify learner X_i according to equations 4.1, 4.2, 4.3 or 4.4
        accept $newX_i$ if f($newX_i$) is better than f(X_i)
    end for
    update assigned Teacher and Mean
end while

4.2 Simulated Annealing (SA) Algorithm

Simulated Annealing (SA) algorithm has been used for solving many optimization problems. The SA algorithm needs to define the number of design variable(s) and generates random population of the composite metrics that would define/feed the input data to the optimization algorithm. In the next step, the SA algorithm randomly initiates its first move and continues to iterate until the state, based on the value of temperature t and energy e, is achieved. This step is analogous to the process that happens in metallurgy while making the alloy strong by varying the temperature of the alloy material. Typically, in metallurgy process the temperature ranges from 260°C to 1400°C. But in context of simulation of this metallurgy process, the values of temperature are configured to any positive range.

The SA algorithm checks and updates the values of temperature to analyze the health of each Nano-Cache (in terms of energy) in each iteration *i*. It checks the new state and then decides whether to involve or ignore the current solution based on the current temperature. The process is repeated until sufficient coverage of the search space is done and acceptable solution is found. For acceptance of the solution, the value of tolerance and temperature must be within the acceptable limits. During the iteration processes, if the state energy decreases, it means the value of objective function has improved and if the state energy increases then the solution ships to a slightly worse solution. This is attributed to the fact that the temperature decreases exponentially as the SA algorithm progresses.
In this work, SA algorithm is applied to find the difference in performance with respect to the MTLBO algorithm. The pseudo code of SA is shown in Algorithm 4.

Algorithm 4 Simulated Annealing Algorithm Pseudo Code

for i = 0 to N_c **do**
    initialize minbw, maxbw, N_c, ISC, MSC, LR, t_1, t_2, F_c
    Compute random number randomNum
    Calculate bw = minbw + randomNum.nextInt(maxbw)
    Initialize bw = F_c
    HealthMatrix (H_m) = addCache(N_{ci})
end for
for i = 0 to i < TeacherStudentEvaluator.numberOfFogCacheNanoServers() **do**
    Initialize session.add to NULL
end for
Create a random individual
Loop through all our destination class_score_quality (minScoreofClass i.e. $C_{minscore}$) and add them to our session time line
Randomly reorder the session
Set as current best (bestScoreofClass i.e. $C_{bestscore}$)
Loop until system has cooled ($t_2 = t_1$)
while $C_{bestscore} > C_{minscore}$ **do**
    acceptanceFunction(currentScore, peerScore, $C_{bestscore}$) > Math.random()
    Keep track of the best solution found
    Compute $C_{bestscore}$ *= 1 - learningRate (temperatureRate)
end while

4.3 Proposed Modified Teaching Learning Based Optimization (MTLBO) Algorithm

The objective of MTLBO is to work like a unicast routing algorithm for delivering content at the remote/edge locations based on the current capacity of 'cache server' and interconnect links for data forwarding. Other than these characteristics, the architecture (comprises of Nano-Cache having proposed algorithm) will be distributed that is asyn-

chronous in nature. It will maintain the link state(s) at the nodes and internodes between the fog and cloud zones which impact the capacity of cache link utmost. Hence, the main objective is to minimize 'content retrieval time' using caching and optimization approach.

4.4 Implementation for MTLBO

This part elaborates how the MTLBO helps in improving the content retrieval time in fog zone of cloud. This section consists of five sub-sections. First subsection 4.4.1 describes terminology and definitions used for MTLBO. Assumed fog zone and system model is elaborated and discussed in second sub-section 4.4.2. The third sub-section 4.4.3 explains the construction of composite metric and the fourth sub-section 4.4.4 describes the process of normalization. The last sub-section 4.4.5 implements the proposed MTLBO algorithm.

4.4.1 Terminology and Definitions used for MTLBO

The terminology for the design and implementation of MTLBO are as follows:

(i) Access Time: This is the time taken in nanoseconds to access x number of bytes from the cache storage. This time may increase due to latency issues in fog network.

(ii) Transmission Delay: Transmission delay is the time taken by a packet from source to destination. Higher the size of packet, more is the transmission delay. It is computed in milliseconds.

(iii) Render Time: It is the total time taken by the device to render full webpage once the packet arrives from the content server, and is computed in milliseconds.

(iv) Bandwidth: This is the rate at which the data is transferred. It is computed in bits per seconds (bps). Bandwidth can be categorized as 'Available Bandwidth' and 'Used Bandwidth'. The term 'Available Bandwidth' signifies the unutilized bandwidth as a resource. The values of available bandwidth are represented as moving averages of last 10 intervals and current value is computed using equation 4.5 [198, 199].

$$T_{B_w} = \frac{1}{m}\left(\sum_{j=-k}^{k} [yt + j]\right), \tag{4.5}$$

where m = 2k+1, that is the estimate of the trend cycle at time 't' is obtained by averaging values of the time series within 'k' period of t. T_{B_w} is the total bandwidth and y is the original time series.

The term 'Used Bandwidth' is the amount of consumed bandwidth that is in use. This can also be calculated as the difference of total bandwidth and the available bandwidth.

(v) Total cycle time: Total time consumed in accessing, transmitting and rendering particular set of bytes (content) from one device to another device is considered as total cycle time. Numerically, it is a sum of access time, transmission delay and render time. It is computed in milliseconds. A cycle comprising of accessing the cache, rendering the data content and then transmitting the content to the requester. The total time taken by all such cycles forms the total cycle time.

$$T_c = (A_t + T_d + R_t), \tag{4.6}$$

where T_c is the total cycle time, A_t is the access time, T_d is the transmission delay and R_t is the render time, in milliseconds.

(vi) Total cycles: It is the total number of bytes (data) that are involved per cycle. A cycle is a total period through which the bytes are send and received to complete one transaction. Mathematically, it is computed as a ratio of total number of data bytes divided by the total cycle time. It is calculated using the equation 4.7.

$$T_{cyc} = \frac{N_{cb}}{T_c}, \tag{4.7}$$

where T_{cyc} is the total cycles, N_{cb} is the number of content bytes and T_c is the total cycle time as calculated in equation 4.6.

(vii) Average Content Retrieval Time: 'Content Retrieval Time' is the time between the user, requests for a particular content till the content is finally available to the user. Average content retrieval time is the number of times the content retrieval process is computed in a particular time frame. This is also termed as time elapsed or execution time. It is computed in milliseconds. It is calculated using the equation 4.8.

$$A_{crt} = \frac{T_{cyc}}{N_t}, \tag{4.8}$$

where A_{crt} is the average content retrieval time, T_{cyc} is the total cycles calculated in equation 4.7 and N_t is the time frame.

(viii) Distance: It is the euclidean distance between the source cache and the receiving device. The positions of source and receiving device are measured in terms of longitude and latitude difference. It is computed as:

$$E_{dist} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}, \tag{4.9}$$

where (x_1,y_1) and (x_2,y_2) represent the position of Nano-Cache server in the vector space model.

(ix) Energy consumed: This is the minimum energy that is consumed when one byte of content is processed for operations such as accessing and rendering. It is computed in Joules (J).

(x) Access energy consumed: Energy consumed in accessing data bytes in a given interval and is represented by E_a.

(xi) Transmission energy consumed: Energy consumed in transmitting data bytes in a given interval and is denoted by E_t.

(xii) Render energy consumed: Energy consumed in rendering data bytes in a given interval and is denoted by E_r.

The above mentioned parameters are used to design the fog zone's system model, that is used for implementation of MTLBO algorithm.

4.4.2 Fog Zone's System Model

Figure 4.1 depicts the components and system model for MTLBO. The system model comprises of three layers namely, Data Center Layer, Fog Zone Layer and Client Layer, as described below.

- *Data Center Layer:* This layer consists of Data Centers, Cores and Data Farms. The core of the cloud infrastructure is made up of data farms. These data farms consists of servers that holds the content, needs to be delivered at various points.

- *Fog Zone Layer:* This layer consisting of fog devices like switches, hub, routers etc. The devices have small data farms, that may further have network of cache servers that are geographically distributed to form content delivery network (CDN). The geo-distribution of these cache servers is done strategically so that the advantage of proximity for particular user base may be taken. For this, the various Internet Service Providers (ISPs) may even deploy their own cache servers for catering to their own subscribers.

- *Client Layer:* This layer consists of diverse devices which make the mix of user base, sensors, data aggregators, hotspots, mobile devices and televisions that are andriod enabled. This layer primarily consists of 'data producers', 'data aggregators' and 'data transmitters'. These devices may be spatially distributed or may be deployed with 'client-server' architecture or 'n-tier' architecture or may have simply a mesh topology. These devices being far away from the core of the cloud, may have larger content retrieval time. To overcome this challenge, the network layer at this end may be augmented with Nano-Cache servers that may be privately owned or may be part of public infrastructure [200, 201].

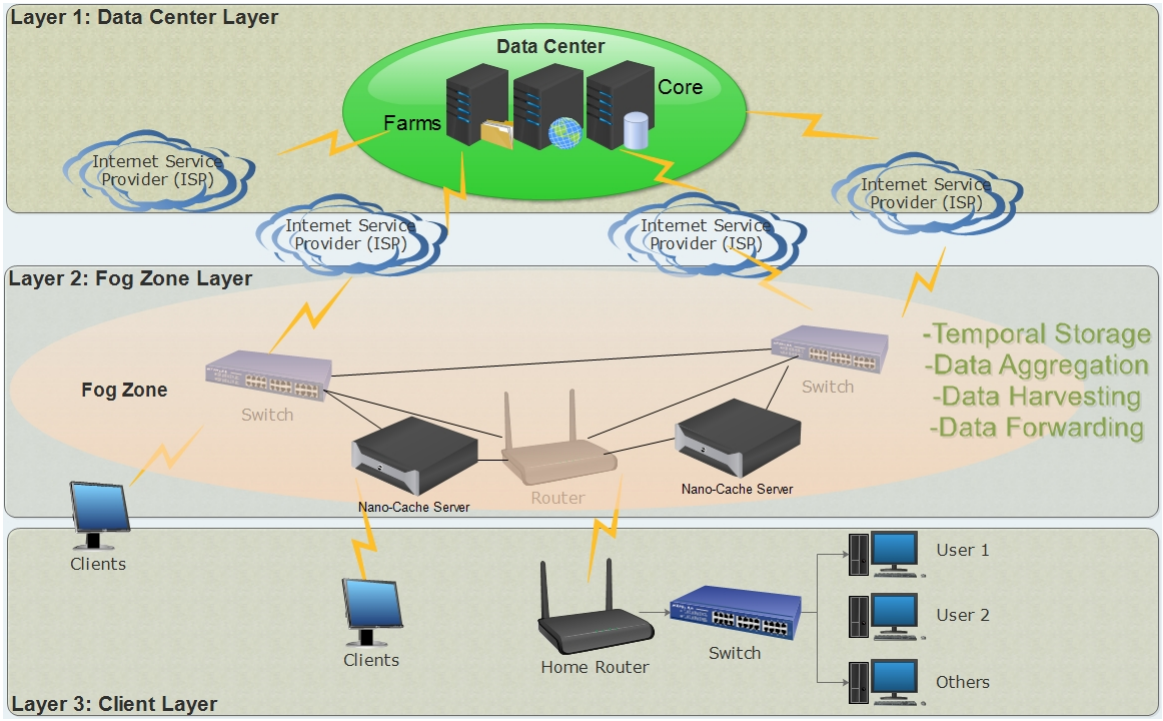


Figure 4.1: Assumed Fog Zone and Components of System Model

4.4.3 Construction of Composite Metric for Fog Zone's System Model

The objective of this work is to improve content retrieval time of the fog zone and this can be achieved by using the concept of Nano-Cache. But, this concept is needed only if there are specialized software defined components that work along with the routing process for selecting highly responsive Nano-Cache machine and best path to reach this machine. This section also explains how the capacity of a Nano-Cache server is measured using "derived or composite" metric. The advantage of using composite metric is to use multiple parameters that are combined to form a single metric and is easy to interpret on fog zone network. In this work, the value of composite metric at the point (x,y) can be calculated as:

$$f(x,y) = \alpha(A_{crt}) + \beta(P_r) + \gamma(R_u) + \delta(E_c), \tag{4.10}$$

where A_{crt} is the average content retrieval time between the source cache and destination, proximity 'P_r' is defined as euclidean distance between the source cache and destination, I/O resource utilization 'R_u' is defined as the available bandwidth on the link, energy consumed 'E_c' is defined as the consumption of total energy i.e. access energy consumed E_a, transmission energy consumed E_t and render energy consumed E_r in last 't' number of intervals as represented in equation 4.11. Section 4.4.1 describes the detailed explaination of these terms. Thus,

$$E_c = (E_a + E_t + E_r). \tag{4.11}$$

α,β,γ, δ are constants.

Next sub-section discusses about the normalization of data used in MTLBO.

4.4.4 Normalization of Data

All these values undergo the process of normalization, before they are used by the optimization algorithm. The process of data normalization induces smoothness in the data by removing abnormalities, diverseness and skewness to make data invariant in nature.
Let dp_1 be the variable representing the composite metric vector having all the characteristics of A_{crt},P_r,R_u,E_c and is denoted in equation 4.12.

$$dp_1 = ((A_{crt_1}, P_{r_1}, R_{u_1}, E_{c_1}), (A_{crt_2}, P_{r_2}, R_{u_2}, E_{c_2}), \ldots, (A_{crt_i}, P_{r_i}, R_{u_i}, E_{c_i})). \tag{4.12}$$

The mean of these data points are:

$$\bar{A_{crt}} = \frac{A_{crt_1} + A_{crt_2} + \ldots + A_{crt_i}}{n}, \tag{4.13}$$

$$\bar{P_r} = \frac{P_{r_1} + P_{r_2} + \ldots + P_{r_i}}{n}, \tag{4.14}$$

$$\bar{R_u} = \frac{R_{u_1} + R_{u_2} + \ldots + R_{u_i}}{n}, \tag{4.15}$$

$$\bar{E_c} = \frac{E_{c_1} + E_{c_2} + \ldots + E_{c_i}}{n}, \tag{4.16}$$

where n is the number of tuples. Hence,

$$dp_1 \rightarrow (A_{crt_1} - \bar{A_{crt}}, P_{r_1} - \bar{P_r}, R_{u_1} - \bar{R_u}, E_{c_1} - \bar{E_c}), \ldots. \tag{4.17}$$

Let dp_2 be the variable representing the difference of root mean square distance between each point. This can be computed using the formula mentioned in equation 4.18.

$$dp_2 = \sqrt{\frac{(A_{crt_1} + \bar{A_{crt}})^2 + (P_{r_1} + \bar{P_r})^2 + (R_{u_1} + \bar{R_u})^2 + (E_{c_1} + \bar{E_c})^2 \ldots}{n}}. \tag{4.18}$$

$$C_m = \frac{1}{E_{rr}}(|(A_{crt}, P_r, R_u, E_c)|). \tag{4.19}$$

Let 'T_m' be the minimum threshold matrix required for using the sampled valued in search algorithm. Therefore, if $|(C_m - T_m)| > 0$ then find distance, $C_{mo} = |Q_m - C_m|$ else ignore. Mathematically this is represented using equations 4.19, 4.20 and 4.21.

$$T_m = (A_{crt_{tm}}, P_{r_{tm}}, R_{u_{tm}}, E_{c_{tm}}). \tag{4.20}$$

Assumptions on T_m are:

(i) Value of p_{tm} must be greater than 0.

(ii) Minimum of 0.5J energy (e_{tm}) is dissipated in transmitting or rendering or accessing the content [202].

$$Q_m = (A_{crt_{qm}}, P_{r_{qm}}, R_{u_{qm}}, E_{c_{qm}}). \tag{4.21}$$

The value of Q_m matrix depends on the agreement between customer and the content service provider. The next step checks the difference between the current and historical data. Mean deviation is computed for the current set of observation for the parameters that influence the quality of link. If the distance between the mean and Q_m matrix is above the threshold, set the Q_m matrix readings to the mean deviation of current readings. If the distance between the mean and Q_m matrix is less then threshold, then the Nano-Cache machine is dropped from the current table as available routes.

The key objective of MTLBO is to optimize the total cost of all the edges for delivering/forwarding the content. The content is received from cloud group 'C_g' and it reaches the fog zone group 'F_g' having group with its own set of Nano-Caches and unicast content delivery Nano services. The following equation 4.22 and 4.23 are used for cloud group and fog group, respectively.

$$C_g =< V_c, E_c > . \tag{4.22}$$

$$F_g =< V_f, E_f > . \tag{4.23}$$

V_c, V_f are either routers or data aggregators from which the content is to be delivered to far end-nodes.

The computation parameters shown in Table 4.1 are computed as per the terminology discussed in Section 4.4.1.

In this, **X** and **Y** represent the X-coordinate and Y-coordinate, respectively, of each cache server deployed in the fog zone. Table 4.1 represents the partial dataset used for this algorithm.

In this, A_t is the Access Time, T_d is the Transmission Delay, R_t is the Render Time and B_w is the bandwidth. $\mathbf{T_c}$ is the total cycle time which is computed using A_t, T_d and R_t as represented in equation 4.6. T_{cyc} is the total number of cycles that are computed using the equation 4.7. A_{crt} is the average content retrieval time and is computed using equation 4.8.

Considering the first cache as the destination cache server and the last cache, shown in the Table 4.1, as the source from where the request is initiated at any particular instance, euclidean distance 'P_r' between the source and destination cache is calculated using the equation 4.9. A_{B_w} is the available bandwidth which is the difference of used bandwidth 'U_{B_w}' from the maximum bandwidth. It is assumed that the bandwidth consumed is 2GB

per day i.e. 87381.33 Kbps per hour. E_c is the total energy consumed and is computed using the equation 4.24.

$$E_c = \frac{N_{bytes} \cdot E_o}{A_t}, \tag{4.24}$$

where N_{bytes} is the request by source cache in bytes and it is assumed to be 64 Kbps. $\mathbf{E_o}$ is constant and the value of $\mathbf{E_o}$ is taken as 0.5 Joules [202].
Table 4.2 represents the input parameters that are used for simulation. At any instance, the value of optimization parameters are considered from Table 4.1.
To conduct the experiment, various parameters have been considered as mentioned in Table 4.3.

4.4.5 Implementation of Modified Teaching Learning Based Optimization Algorithm (MTLBO)

MTLBO algorithm find 'best solution' with respect to the 'previous' solution as calculated using equation 4.23.
The implementation of MTLBO has been carried out in three steps as described below:

Step I. This step is used to set up the initial design parameters and initial marks that generate initial output randomly. Proximity from the request 'P_r', Average Content Retrieval Time between the source and destination 'A_{crt}', energy consumed 'E_c' and bandwidth 'B_w' are the key design parameters that can help in identification of the optimal conditions for forwarding the cached data to the destination.

Step II. In this step iteration begins, and it works until the solution is met. All the options are exhausted to get the optimized functional value, and the best solution is achieved. This step consists of two phases, namely *Teacher Phase* and *Learner Phase*, described below.

- *Teacher phase:* It is the first phase, analogous to learning from the teacher. With the efforts of the teacher, the class mean score/marks increases, as shown in Algorithm 5.
- *Learner phase:* This phase is analogous to the learning by the students for improvement.

In this phase, the acceptance function provides the criteria for accepting the particular route for the current request for content. To fulfill the request, the cached contents are forwarded. The acceptance function accepts or rejects the solution based on the allowable tolerances and optimization criteria. The acceptance function will accept solution if the condition is fulfilled, as shown in Algorithm 6.

Table 4.1: Computation Parameters used for Experimental Work

X	Y	A_t	T_d	R_t	B_w	T_c	T_{cyc}	A_{crt}	P_r	A_{B_w} = 87381.33-B_w	E_c
41.68	19.81	0.65	0.11	0.97	434.49	1.72	5.80	0.58	35.26	86946.84	49569.99
65.69	48.97	0.48	0.11	0.19	276.76	0.78	12.80	1.28	29.03	87104.57	66741.30
62.80	33.95	0.64	0.06	0.14	201.78	0.84	11.88	1.19	32.97	87179.55	50053.42
29.20	95.16	0.54	0.40	0.70	841.44	1.65	6.08	0.61	41.17	86539.89	58746.20
43.17	92.03	0.65	0.45	0.09	440.24	1.19	8.41	0.84	37.71	86941.09	49435.24
1.55	5.27	0.54	0.37	0.53	909.08	1.44	6.97	0.70	61.04	86472.25	58835.87
98.41	73.79	0.72	0.76	0.53	400.57	2.01	4.96	0.50	64.05	86980.76	44379.93
16.72	26.91	0.52	0.63	0.86	787.57	2.01	4.97	0.50	34.62	86593.76	61244.57
10.62	42.28	0.99	0.77	0.48	406.31	2.25	4.44	0.44	29.41	86975.02	32202.73
37.24	54.79	0.22	0.93	0.39	827.92	1.54	6.47	0.65	0.00	87003.41	146334.79

Table 4.2: Input Parameters for MTLBO

Parameter's Name	Notation	Value(s)
Number of Cache Servers	No_{cache}	10, 15, 20, 25, 30, 35, 40, 45, 50, 55
Learning/Moderation Rate	Mod_{Rate}	0.0116, 0.0125, 0.0258, 0.0379, 0.0473, 0.0582, 0.0667, 0.0781, 0.0873, 0.0999
Minimum Bandwidth	$MinB_w$	201.78 bits per second
Maximum Bandwidth	$MaxB_w$	87381.33 bits per second
Minimum Energy	$MinE_c$	1.5 Joules (for page of 64kb)
Maximum Energy	$MaxE_c$	5000 Joules
Minimum Average Content Retrieval Time	$MinA_{crt}$	2000
Maximum Average Content Retrieval Time	$MaxA_{crt}$	4000
Initial Score of Class	$InitS_c$	5000
Minimum Score of Class	$MinS_c$	1000
Optimization Parameters		
Bandwidth	B_w	434.49 bits per second
Energy consumed	E_c	49569.99 Joules
Average Content Retrieval Time	A_{crt}	0.58

Step III. In each iteration, we select a neighbour by making a small rate of change to current output value. The next neighbour is the next 'cache server' in the fog network.

Algorithm 5 Teacher Phase Pseudo Code of MTLBO

Input: Deploy Nodes in the Vector Space model, fog_edges_id
Initializations: Default Cost Score, A_{crt}, P_r, B_w, E_c, Solution Population
Output: Composite Matrix ▷ Returns the best cache server

```
initialization                                  ▷ Initialization of all values
for each subnet in the fog network do
    for each 'cache server' in network do
        compute composite metric score
        assign cost score to fog_edge_id
    end for
end for
```

Then, we decide to move the tuples of the next neighbour solution (fog cache), and finally, we continue to decrease the 'maximum marks', so that maximum number of intermediate solution can enter the search space and finally compare the solution with the solution found in previous iterations.

The best solution is analogous to finding best team of Teacher-Student, who get maximum score or marks. The idea is to find the best team having cumulated score 'C_s'. The cumulated score is sum of all the scores/marks of students, and can be expressed in equation

Table 4.3: Experimental Configuration Parameters for MTLBO

Experiment Parameter	Explanation
Name	Name of the experiment
Run	Name of the algorithm for evaluation
Stopping criteria	Stop criteria
Configuration file	Name of the configuration file
Mode	Single or parallel run of algorithm
Repeat	Run the algorithm in repetitive mode
Input directories	Configuration file
Output directories	Result/output of the experiments
Max failure	Maximum number of times the experiment is allowed to fail
Max time	Maximum allowed time to run the experiment

Algorithm 6 Learner Phase Pseudo Code of MTLBO

Input: Deploy Nodes in the Vector Space model
Initializations: Default Cost Score, A_{crt}, P_r, B_w, E_c, Solution Population
Output: Composite Matrix

```
initialize the fog resource monitor
register fog node statistics
for each 'cache' in the fog network do
    compute the difference between the previous and current performance statistics
    execute acceptance function
    if (acceptanceFlag==true) then
        if (check termination condition == true) then
            accept value, cache server IP
        else
            continue
        end if
        continue
    end if
end for
```

4.25.

$$C_s = \frac{exp(C_{score} - P_{score})}{T_{score}}, \tag{4.25}$$

where $\mathbf{C}_{score}$ is the current score, $\mathbf{P}_{score}$ is the previous score and T_{score} is the total score. In our context, the marks of the group can be considered on the basis of the aforesaid parameters. Smaller the rate of change in the marks or solution output, higher will be the marks and more likely for the algorithm to accept the final output as the best solution.

4.5 Analysis of Proposed MTLBO Algorithm

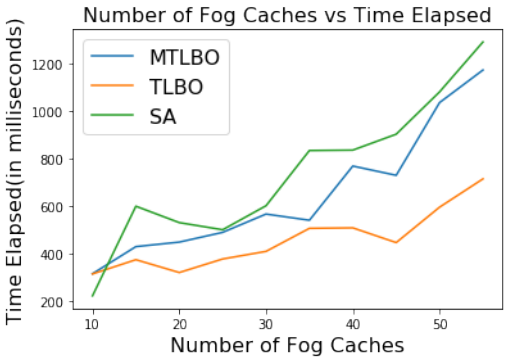
The efficiency of the algorithms is normally evaluated on the basis of running time and memory space used. But due to asynchronous model and distributed nature of the implemented algorithms, computation running time and memory space is not properly quantifiable. Therefore, the analysis of MTLBO has been done on the basis of following parameters:

(i) Computational rounds: This analysis helps to understand the overhead in terms of number of iterations. Higher number of iterations means higher degree of overhead and reduced performance.

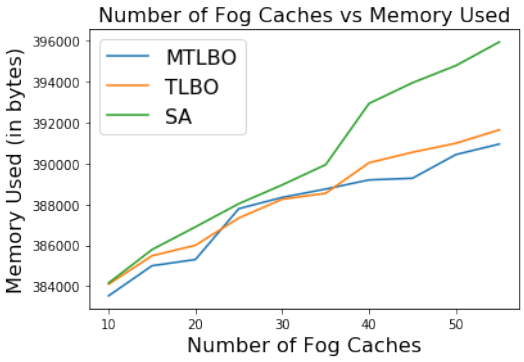
(ii) Cost analysis: Each round of forwarding data can be formed by following multiple routes. Hence, cost analysis of path is important. The cost of each path is represented in the form of 2D vector. All the parameters are transformed into 2D vectors and then the difference between each machine performance is found using distance metric.

(iii) Time analysis: Higher the number of iterations and multiple path options, more will be the time to identify the optimal Nano-Cache. The time analysis thus would help to identify which path to follow for transfer of packets in terms of time efficiency.

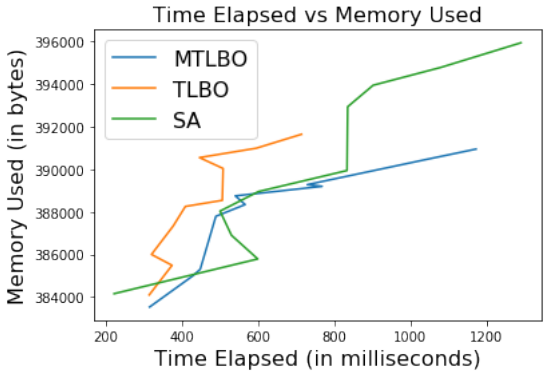
For the analysis, three different test cases (Test Case I - Test Case III) are considered that are based on varying number of fog caches, and four test cases (Test Case IV - Test Case VII) are based on varying the moderation rate. These test cases are elaborated below:
Test Case I: Number of Fog Caches vs Time Elapsed
In this test case, Figure 4.2 shows the total execution time in milliseconds taken by MTLBO, SA and TLBO to execute with respect to size of the network. The network size is calculated by counting the number of Nano-Caches in the fog network, although the fog network may have other devices also. It is clear from the Figure 4.2(a) that SA is taking more time in finding feasible solutions and finally to identify the optimal solution.



(a) Case I



(b) Case II



(c) Case III

Figure 4.2: Varying the Number of Fog Caches

This can be attributed to the fact that algorithm needs to do more computational rounds as compared to MTLBO and TLBO.

Figure 4.2(a) shows that as the fog zone size increases, time taken for routing as well as algorithms execution also increases. The execution time is almost linearly and the slope is approximately 17.1769, in case of MTLBO. This reflects that TLBO undergoes slow changes with equal number of fog caches, and it takes less time in execution. Since, the values of slope is positive, the rise is from left to right, which clearly shows, that rate at which the time increases, is not non-linear in nature. Hence, the performance of TLBO algorithm is better than MTLBO and performance of MTLBO algorithm is better than SA. The equation 4.26 gives a mathematical representation of the relationship between number of fog caches and execution time in case of MTLBO. This relationship is derived using curve fitting technique [203, 204].

$$f(x_1) = C_{e_1} \cdot x_1^3 + C_{e_2} \cdot x_1^2 + C_{e_3} \cdot x_1 + C, \tag{4.26}$$

where x_1 is number of fog caches. It is normalized by *mean* 32.5 and *standard deviation* 15.14 coefficients (with 95% confidence bounds): C_{e_1} = 50.34, C_{e_2} = 86.04, C_{e_3} = 179.6 and C = 571.7.

Test Case II: Number of Fog Caches vs Memory Used

In this test case, the efficiency of MTLBO is dependent on the number of computational resources used by it. Time and memory are two main parameters that contribute towards the efficiency and are considered for MTLBO. It is abundantly clear from the Figure 4.2(b) that when the algorithm was run, memory consumed by the process is increasing linearly and the rate (average slope = 156.858) at which it is increasing is not that fast even when the network size (including Nano-Caches) increases to 40 or 50. Memory consumption in case of SA is highest due to the fact that it is consuming more time and is performing more computation iterations to acquire optimal solution. Mathematically, the relation for MTLBO can be understood using the equation 4.27.

$$f(x_2) = C_{e_1} \cdot x_2^3 + C_{e_2} \cdot x_2^2 + C_{e_3} \cdot x_2 + C, \tag{4.27}$$

where number of fog caches is denoted by x_2 and is normalized by *mean* 32.5 and *standard deviation* 15.14 coefficients (with 95% confidence bounds): C_{e_1} = 258.5, C_{e_2} = -561.9, C_{e_3} = 1961 and C = 3.884e+05.

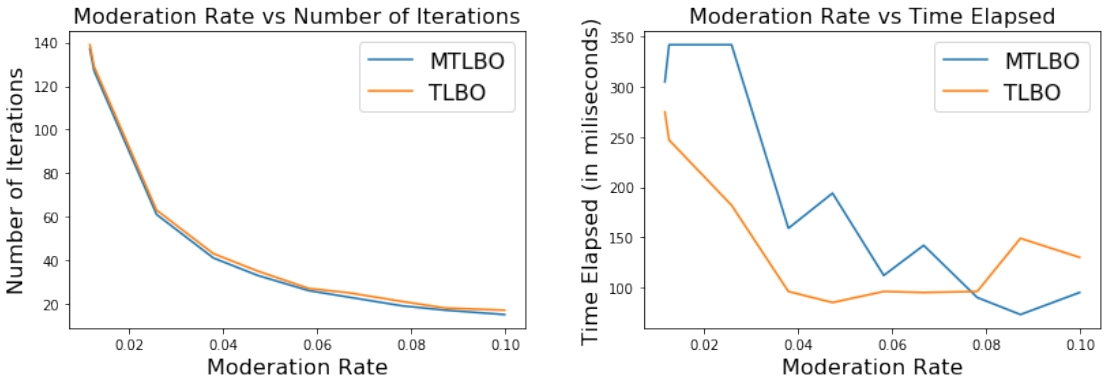
The Figure 4.2(b) clearly shows that MTLBO is consuming less memory as compared to TLBO.

Test Case III: Time Elapsed vs Memory Used

An algorithm is efficient, if the number of computational resources taken by it is less as compared to the output generated. It is clear from the Figure 4.2(c), that the MTLBO algorithm is taking less memory as compared to TLBO. But, theoretically, the estimate on their complexity is almost same as both are heuristic random search techniques. From the Figure 4.2(c), it can also be interpreted that if the algorithm consumes more memory, it also takes more time to execute the tasks, indicating that overhead is increasing at estimated rate of 7.777. The quality of optimization algorithm depends on, firstly, how fast it is able to search optimal solution and secondly, on how much overhead (memory) it takes to acquire optimality. It is clear from the Figure 4.2(c) that SA is taking more time in execution and consequently taking more memory compared to TLBO and MTLBO. The derived relationship from the graph's curve in case of MTLBO is shown in equation 4.28.

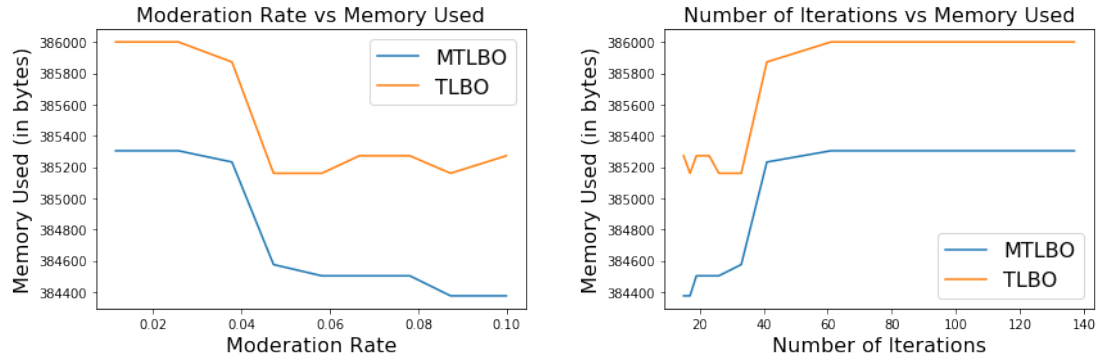
$$f(x_3) = C_{e_1} \cdot x_3^3 + C_{e_2} \cdot x_3^2 + C_{e_3} \cdot x_3 + C, \tag{4.28}$$

where x_3 is time elapsed in milliseconds. It is normalized by *mean* 649.1 and *standard deviation* 276.4 coefficients (with 95% confidence bounds): C_{e_1} = 537.6, C_{e_2} = -1570, C_{e_3} = 2140 and C = 3.889e+05.



(a) Case IV

(b) Case V



(c) Case VI

(d) Case VII

Figure 4.3: Varying the Moderation Rate

Test Case IV: Moderation Rate vs Number of Iterations

It is apparent from the Figure 4.3(a) that the moderation rate of TLBO, SA and MTLBO allows the algorithm to learn and evaluate the population (solution) at different rates. The moderation rate gives a chance to the algorithm to fine tune its accuracy and number of iterations execution. Less iterations would mean faster output and less memory overhead. Figure 4.3(a) shows that as rate of moderation increases, the number of iterations decreases. Initially, the slope is -2975.9489 which means the drop (number of iterations) is very high and later on it becomes steady. The negative slope is an indicator that the number of iterations are decreased drastically as such overheads were also reduced.

Mathematically, the relation for MTLBO can be understood using the equation 4.29.

$$f(x_4) = C_{e_1} \cdot x_4^3 + C_{e_2} \cdot x_4^2 + C_{e_3} \cdot x_4 + C, \tag{4.29}$$

where moderation rate is represented by x_4 and is normalized by *mean* 0.05253 and *standard deviation* 0.03086 coefficients (with 95% confidence bounds) and C_{e_1} = -15.49, C_{e_2} = 28.8, C_{e_3} = -15.95 and C = 24.67.

The Figure 4.3(a) shows that the number of iterations are almost same. But MTLBO takes few iterations less as compared to TLBO.

Test Case V: Moderation Rate vs Time Elapsed

The change in moderation metric impacts the execution time as it impacts the iteration rounds. The trend is almost similar to Figure 4.2(a), but there are more ups and downs in Figure 4.3(b). It indicates when the moderation rate is between 0.2 to 0.3, the time elapse is also the same (300-350 milliseconds). But, as the moderation rate increases, the execution time further decreases and it can be attributed to the fact that it takes less number of iterations. When the moderation rate is between 0.075 to 0.10, the execution time drops from 300-350 milliseconds at initial stage and becomes 150-100 milliseconds. Mathematically, it can be represented by equation 4.30 in case of MTLBO.

$$f(x_5) = C_{e_1} \cdot x_5^3 + C_{e_2} \cdot x_5^2 + C_{e_3} \cdot x_5 + C, \tag{4.30}$$

where x_5 is the moderation rate. It is normalized by *mean* 0.05253 and *standard deviation* 0.03086 coefficients (with 95% confidence bounds): C_{e_1} = 16.58, C_{e_2} = 28.34, C_{e_3} = -123.6 and C = 159.2.

In this case, the overall performance of MTLBO is poor as compared to the TLBO till the moderation is in the bracket of 0.02 - 0.075, but after that TLBO starts taking more time for the computation.

Test Case VI: Moderation Rate vs Memory Used

The trend of Figure 4.3(c) is consistent with Figure 4.3(b). As the moderation rate increases, the number of rounds decreases and at the same time there is decrease in execution time, consequently the memory overhead is also decreasing with increase in moderation rate. From this, it can be safely said that moderation rate is inversely proportion to the memory consumption, time and rounds/iterations of the algorithm. Mathematically, this can be represented as:

$$m \propto (c_1 \cdot \frac{1}{Time} + C_{onst_1}), (c_2 \cdot \frac{1}{Memory} + C_{onst_2}), (c_3 \cdot \frac{1}{Iterations} + C_{onst_3}). \tag{4.31}$$

The relation, in case of MTLBO, can be well understood by the equation 4.32.

$$f(x_6) = C_{e_1} \cdot x_6^3 + C_{e_2} \cdot x_6^2 + C_{e_3} \cdot x_6 + C, \tag{4.32}$$

where moderation rate is denoted by x_6 and is normalized by *mean* 0.05253 and *standard deviation* 0.03086 coefficients (with 95% confidence bounds): C_{e_1} = 119.8, C_{e_2} = 67.75, C_{e_3} = -580.2 and C = 3.847e+05.

The Figure 4.3(c) curves clearly show MTLBO is better in terms of moderation rate for memory used.

Test Case VII: Number of Iterations vs Memory Used

It is clear from the Figure 4.3(d) that as the number of iterations increases initially, the consumption of the memory increases exponentially, and then later on, the graph line becomes horizontal showing no change in slope and constant memory consumption with the increase in number of iterations. It can be inferred that memory consumption is proportional to number of iterations. Equation 4.33 represents this mathematically.

$$MemoryConsumption \propto Numberof Iterations \tag{4.33}$$

In case of MTLBO, the relation can be understood by the equation 4.34.

$$f(x_7) = C_{e_1} \cdot x_7^3 + C_{e_2} \cdot x_7^2 + C_{e_3} \cdot x_7 + C, \tag{4.34}$$

where x_7 is number of iterations. It is normalized by *mean* 49.9 and *standard deviation* 45.42 coefficients (with 95% confidence bounds): C_{e_1} = 27.78, C_{e_2} = -413.5, C_{e_3} = 752.5 and C = 3.851e+05.

The gap of values between the TLBO and MTLBO shows MTLBO is out performing TLBO in context of memory as well.

4.6 Comparison of MTLBO with TLBO and SA

TLBO algorithm was developed by R. Venkata Rao [205] and has been widely accepted as an excellent optimization algorithm for number of problems such as optimal design of spur-gears, composite test functions, multi-objective unconstrained and constrained test functions, design of pressure vessel, compression/tension spring design, speed reducer design etc [141, 206, 207, 208, 209, 210]. This algorithm has also been successfully employed on unconstrained problems as well [211, 212].

The problem of Pareto Front (PF) has been solved by implementing modified version of TLBO and it is known as elitist TLBO (ETLBO) [211, 213]. This algorithm has also been used to multi-objective optimization problems such as robot gripper, wind-thermal emission etc [214, 215]. After extensive study of the TLBO and its variants, it was found that this optimization technique is suitable for load balancing in fog infrastructure also. Therefore, its previous version and modified version were applied with suitable parameter tuning to obtain the best results. To benchmark the performance of both the algorithms, a comparative study is carried here.

The main difference between the previous implementation (TLBO) and new implementation (MTLBO) depends upon how the learning rate is modified. In previous algorithm, learning rate was linear in nature whereas in new algorithm it is computed exponentially (Ref. Eq. 4.25). The second difference is that the algorithm covers up the search space

faster as compared to the previous work, consequently takes less time, memory and energy for execution.
The performance of MTLBO is good in context of the problem undertaken because it is consuming less memory and is taking less time for identifying optimal solution as compared with SA. But it performs farely well when it is compared with TLBO because it is taking 0.2% more memory (considering moderation rate) in execution.

Table 4.4: Comparison of MTLBO with TLBO and SA

Algorithm	Overheads in terms of Network Size	Overheads in terms of Memory	Polynomial Time in Execution	Scalability as function of Network Size	Optimization Strategies	Type of Optimization Problem	Learning Rate
SA	More	More	More	Average	Search Model, Stochastic	Convex	Not Present
TLBO	Less	Less	Less	Average	Heuristic/ Meta Heuristic	Convex	Present
MTLBO	Least	Least	Least	Best	Heuristic/ Meta Heuristic	Concave	Present

In case of SA, there is no concept of multiple phases and moderation rate. It has 'temperature' parameter to adjust and to search for optimal solution. Experimental results show that the temperature value does not impact much on optimal value iterately due to which the SA algorithm conducts more iterations.
The comparison of existing algorithms (TLBO and SA) with the proposed approach (MTLBO) is shown in Table 4.4.

4.7 Summary and Conclusion

In this work, an optimized algorithm 'Modified Teaching Learning Based Optimization (MTLBO)' is devised for conducing route analysis for forwarding content using cache in the fog computing network. The optimization algorithm is able to optimally reflect the conditions of the system. In this chapter, MTLBO took less number of iterations to reach the optimal coverage from which we can infer that the algorithm was performing with minimum overhead (computation and other resources). This was evident from the fact that more iterations did not yield any optimal value. The fog network became efficient and the traffic was routed to the most optimal path.

The MTLBO has been compared with existing TLBO and SA on the basis of metrics that impact the performance of the routing process between the Nano-Caches. The analysis of the series of simulation runs show that MTLBO is better than TLBO as it has less overheads in terms of memory (considering number of fog caches) and network size for delivering contents at remote areas. MTLBO performs better than SA in terms of execution time, overhead in terms of memory, and scalability as function of network size. When the memory was plotted against number of iterations, it was found that the average memory of MTLBO is 0.3847MB and it is 0.186% less than TLBO. But in case of execution time, the TLBO is performing slightly better. It can also be concluded that 'moderation rate' is main factor in increasing or decreasing the cost of memory and time. The relationship between the moderation rate and execution time as well as moderation rate and memory is inversely proportional and the nature of relationship can be understood by using cubic equation model. The concept of content caching gives power to the service providers for providing popular content to the intermediate storing points. This results in reduced traffic load on long-distance cloud networks and Internet access points.

Chapter 5

Applications of Fog Computing Using Fog Load Balancer

India is most populated country and many people living on the edges of society, all these need to be included into the formal economy, whether they are poverty-stricken people or climate refugees. By using inclusive technologies, improving the chances of absorbing surplus rural labour can be increased many folds. Such technologies can be helpful for differently abled people as well. The gap between the more developed countries, states and areas can also be reduced to some extent. Case studies from many parts of world show that mobile or smartphone as inclusive technology has brought sea changes in the way street markets and other informal or gray market work and behave. The smartphones have empowered governments and corporate houses to develop new models of trust that are other than the economic model of trust, to include large section of people.

This chapter discusses the application of fog computing in social economy where participants are examined on the basis of certain qualification criteria. For this application of social economy, proposed fog load balancer has been used. In Section 5.1, a Fog Based Communication Model for Inclusiveness of Informal Economy (FBCMI2E) for examining the participants of informal economy, considering some assumptions, has been devised. Section 5.2 discusses the mathematical model for FBCMI2E. Inclusiveness qualification computations of FBCMI2E has been computed in Section 5.3. In Section 5.4, FBCMI2E is automated using machine learning algorithms like Naive Bayes (NB) algorithm, K-Nearest Neighbors (KNN), Decision Tree (DT) and Linear Discriminant Analysis (LDA). The outcomes of this automation is the qualification criteria based upon parameters: fitness index, call analysis, geospatial analysis and modularity of the participant. This qualification criteria will help in determining whether to include the participant into formal economy or not. In the last Section 5.5, theoretical energy analysis for FBCMI2E is presented.

5.1 Fog Based Communication Model for Inclusiveness of Informal Economy (FBCMI2E)

In this chapter, an approach to solve the problem of people living on the margins of economy has been proposed. A case study with simulated data is also presented here. The purpose of this model is to compute four parameters (i.e. fitness index of work(FIW), call log analysis (CA), geospatial analysis (GA) and modularity (M)) for selecting participants. The participants are those people who are registered, by which they can qualify and become part of the main economy. Since, this is a simulation based case study, Figure 5.2 gives details of the assumed network model and other assumptions taken to simulate the application of fog computing.

Following are the assumptions to design and development of inclusive platform using fog and cloud system. It is known by the name 'Fog Based Communication Model for Inclusiveness of Informal Economy' (FBCMI2E).

Assumption 1: The program (mobile application, web application etc.) identify the participants who need help and are considered right candidates to become beneficiary of the inclusiveness program. Identification and inclusion do not mean the person will get guaranteed benefits of the inclusive program. The individual needs to get qualified over a particular span of time.

Assumption 2: Each identified person has to registered voluntarily and agrees to wear and share its location and other medical data for 24x7 and can withdraw from the inclusive qualifying program any time. It is assumed that the participant will either voluntarily or in an organized manner will get registered with some government or non-government agency, running this program. The registration process will consists of registering the mobile phone number along with the mobile operator with whom the government or non-government agency has legal understanding.

Assumption 3: The individuals are monitored for minimum of six months for qualification, and the system must collect at least hundred instances of their data per hour.

Assumption 4: The devices follow Health Insurance Portability and Accountability Act of 1996 (HIPAA) standards to maintain integrity and confidentiality.

Figure 5.1 gives a block diagram of the process to identify and construct the trust model for qualifying the person in the inclusiveness program.

5.1.1 Dataset for FBCMI2E

The initial dataset for this work has been created using multiple methods and sources. The initial set of subjects who are included in the said program were generated using Erdos Renyi (ER) algorithm. The ER algorithm generates the data of the person in the form of graph, as shown in Figure 5.8.

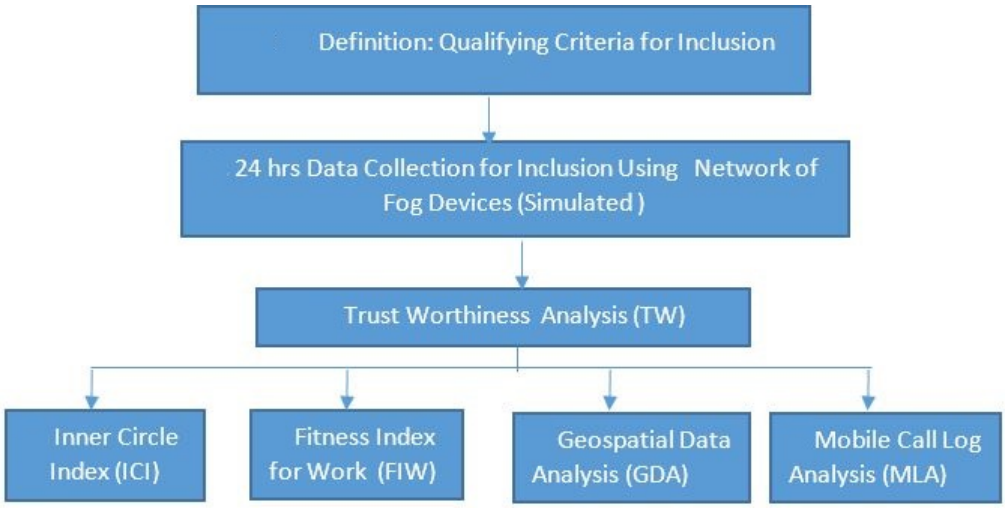


Figure 5.1: Inclusiveness Program for Qualifying Process in FBCMI2E

This graph dataset and the fitness dataset [2] are linked as per the key attribute RegistrationID (in Figure 5.9 and Figure 5.11). The mobile data [216] and geospatial data [217] are also linked as per the identity keys in the graph.

The first step will consists of setting up of infrastructure for communication and observing the subjects. The concept is similar to the mobile computing paradigm, where useful activity data is accessed from the user's smartphone or wearable medical sensors. It is assumed that no data is stored on the smartphone nor any changes are allowed locally during accumulation of sensor data stream but the data is accessible through cloud storage interface. This is done with the help of integrity protocols such as blockchain, Health Level 7 (HL7) etc. as shown in Figure 5.2.

FBCMI2E consists of five layers and the explanation of each layer is discussed below.

(i) Fog Device Layer: This layer consists of devices such as sensors, smart watches, smartphones etc. There is a machine-to-machine communication (fog-to-fog device communication) in case there is need to aggregate and replicate data temporary before sending it to the main cloud server. Hence, there is a need for fog machine to fog machine authentication (machine-to-machine) in this layer, especially when the services such as backup or update needs to be done.

Table 5.1 gives information on the possible use of various medical sensors, devices and applications that can be used for bringing a subject into the formal economy [218]. These devices would act as fog/edge devices to send medical grid information to the cloud server for further analysis. The analysis would detect whether the person is on the threshold of getting inclusive program or not. The purpose of inclusive program is to give some degree of credit support so that the person cannot only survive but thrive in the formal economy. The next section gives the detail of the dataset used for conducting this study. The dataset has been created by collecting nine health parameters with the help of body composition analysing machine by

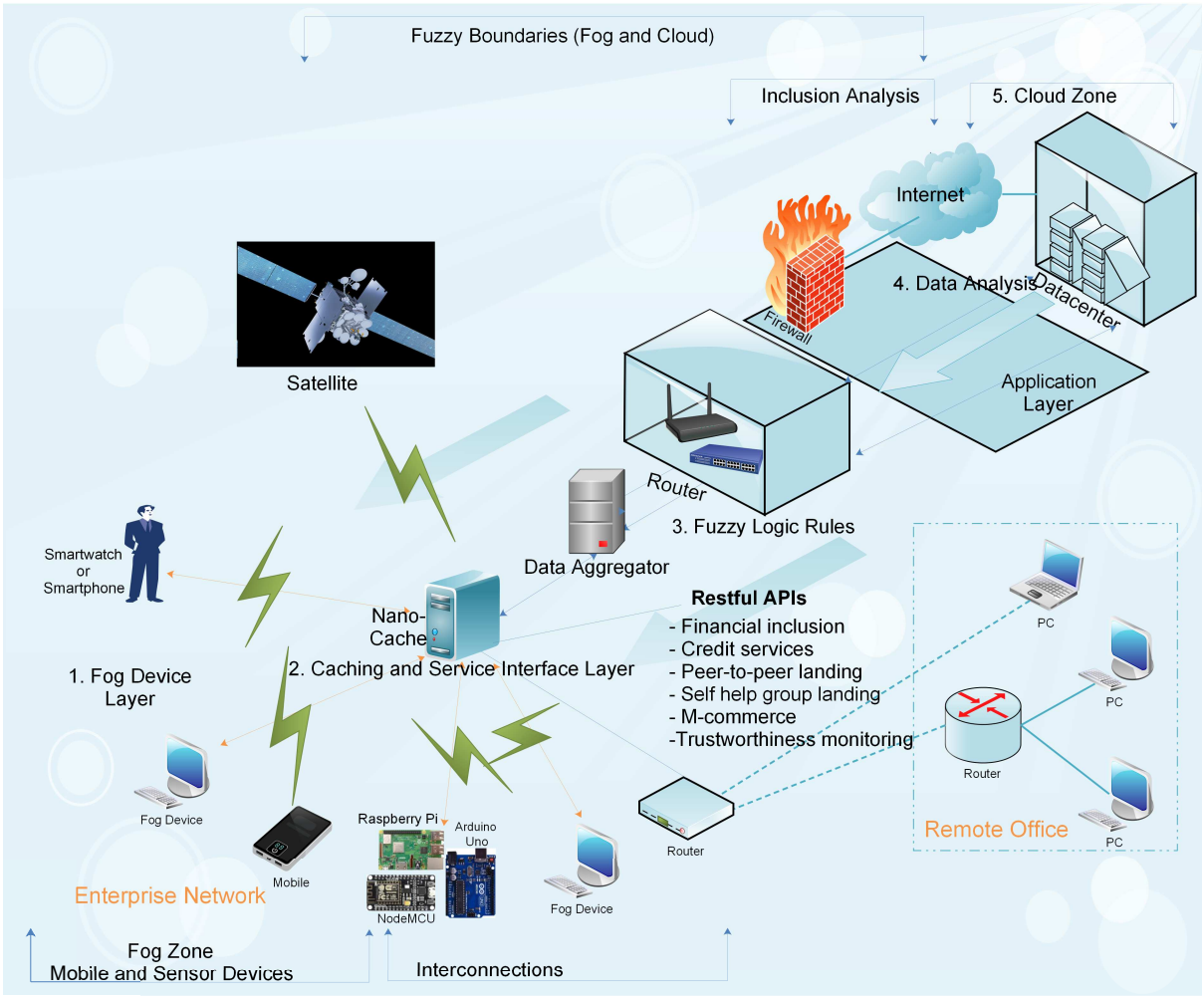


Figure 5.2: System Model of FBCMI2E

Omron Healthcare [2]. Using this as a base dataset, huge volume of dataset is generated. This is done to simulate the conditions in which such inclusive technology can be used. For generating the higher volume of dataset, mathematical analysis is carried out on nine parameters.

(ii) Caching and Service Interface Layer: All the devices from the layer 1 (smartphones, smart watches and other wearables) need to register with this Service Interface, as shown in Figure 5.2. The service interface is inter-operable component of the system that allows heterogeneous network of fog devices to work together with the cloud. It consists of self-care user interface for all the subjects registered with the inclusiveness program. It is a place, which allows temporary caching and aggregation of data (layer 2 in Figure 5.3) of the frequently called objects. Some devices in this layer may work as Nano-Caches to support high degree of responsiveness in edge zone. Algorithms such as Simulated Annealing (SA) reside here for reducing latency as mentioned in chapter 3 of the thesis. Nano-Caches may have algorithms such as Simulated Annealing (SA), Modified Teaching Learning Based

Table 5.1: Fog/IoT/Wireless Medical Sensors and Devices Used as Analysers in Inclusive Technology

Types of Fog Devices	Use as Inclusive Technology
Fitness/Muscle Analyser	There are innumerable variable devices available in the market that can be used for tracking fitness level of the subject. Fitbit, Misfit, Apple, Garmin and Polar are some devices that are popular for tracking fitness level. Such devices are also registered in clinical studies. In context of our research work, such devices can effectively be used as inclusive technology in infrastructure.
Sleep Pattern Analyser (Polysomnography)	The variable devices are now equipped with multi-sensor and multi-objective monitoring algorithms that can record sleep patterns to track the fitness level of a person. For a person in context of the problem undertaken, sleep analysis can track location as well as health details as a criteria to trust and include in the formal economy.
Blood Pressure and Pulse	Typically, wrist worn devices can track heart rate, blood pressure, electrothermal activity of a person. The blood pressure and heart rate are key parameters that can be used for risk analysis of a person for inclusiveness. The readings of pulse can tell about the health of the heart of the subject. If there is a consistent higher order of pulse rate for substantial period, it shows that the subject is having poor heart health.
Body Temperature and Skin	The information from the pattern of the temperature of a person can help to know the health of a person in terms of any vector disease such as fever, viral, dengue, malaria etc. Sudden changes in temperature patterns of a subject from informal economy reflects that the person is unwell and the "inclusiveness factor" needs to be revised. For tracking temperature, skin patch based technology is already in use. This algorithm can also be used as "inclusive technology". Such patches can be embedded in the clothing for tracking heart variability rate and to do actigraphy (study of motor activity).
Oxygen	Finger worn devices can be used to track heart rate and to monitor oxygen level of a person. The level of oxygen is an indicator of respiratory health of a person. From the patterns, it can be inferred whether the person is smoking or is suffering from some respiratory disorder. These days clothing with embedded sensors can track the breathing patterns also. Such clothing can also be given to the person for including him in the main stream economy.

Optimization (MTLBO), Teaching Learning Based Optimization (TLBO) to optimize the traffic load balancing in the edge zone of the network [18].

(iii) Business Rules: The incoming data need to be managed as per the network load conditions. Algorithms based on fuzzy logic (layer 3 in Figure 5.2) may use to manage efficient routing paths aforementioned system. The firewall and all rules related to the security will also apply here as they impact the traffic management. Figure 5.4 depicts how to manage traffic and payloads in fog zone.

Initial deployments in industry have shown that edge computing has helped in fine

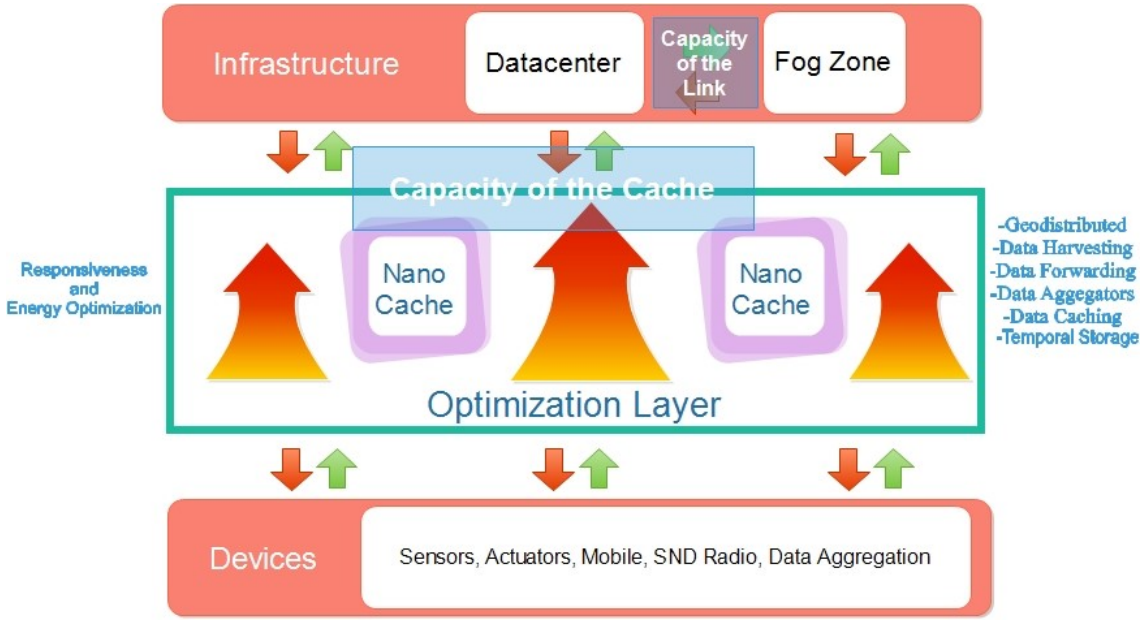


Figure 5.3: Caching and Service Interface Layer

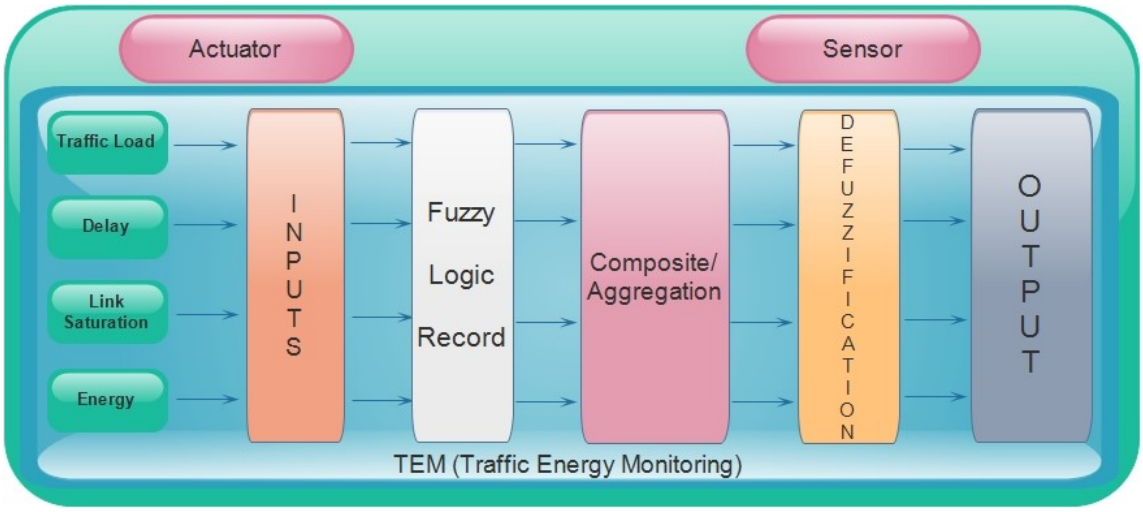


Figure 5.4: Fuzzy Logic Business Rules

grain customization of services and have produced better experiences for the users. However, at the same time, it also invites challenges such as overload, under-load, and imbalance in the utilization of resources such as bandwidth, timely responses, throughput etc. The main reason is highly constrained environment and restricted hardware capabilities of fog devices. The conventional algorithms that work on finding the best routing paths for distributing and processing tasks may not suffice with respect to quality of service (QoS) parameters of fog zone as their computations become unstable. An alternative approach is adding software-defined components, that can help in load balancing and managing the payloads using link analysis at the inter-nodes of cloud and fog zone.

(iv) FBCMI2E Data Analysis: This layer consists of data that is collected at specific pe-

riod/timeline. Hence, this interface will consist of module called inclusiveness data analysis. In this module, the fitness data such as height, gender, body fat, visceral fat, resting metabolizer, basic metabolizer rate, pulse, sugar level during fasting and sugar level after meals are collected and analyzed. The analysis of these parameters helps in construction of mathematical trust model with which a decision can be made to include the person as beneficiary of the inclusionary program of informal economy. The longitudinal data will be accessed for computing health/fitness index, geospatial index and call data index. This module will consist of statistical and machine learning functions for computing trust. But the computations of the machine learning based modeling will require selection of appropriate parameters. Hence, the first step is to compute the correlation between the fitness parameters.

(v) Cloud Zone: In this layer, all the transactions of the system will be stored on the cloud servers permanently. Cloud zone also contains huge data along with the computational capabilities like network, storage, memory resources etc.

The next section explains the mathematical trust model on which the selection criteria for inclusiveness has been considered.

5.2 Mathematical Trust Model of FBCMI2E

For understanding the nature of FBCMI2E dataset, Pearson correlation have been used. It is computed with the help of equation 5.1.

$$P_{corr} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{5.1}$$

where P_{corr} is the Pearson correlation, n is the sample size, x_i,y_i are individual points of sample having index i, $\bar{x}$ is the sample mean and is computed as $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, sample mean $\bar{y}$ is computed in the same manner as $\bar{x}$.
In this section, we conduct pair wise analysis of the fitness data [2]. The purpose is to find variables in pairs that vary together and eliminate them as they characterize collinearity (a condition that lead to unstable outcomes) in machine learning. The second objective is to make pool of variables that can help to construct the mathematical equation model of all the important variables that would impact the response/category variable (qualified or not-qualified).

5.2.1 Correlation Matrix

The variables (H, G), (H, BF), (H, BMR), (G, VF), (G, BMR), (G, RM), (G, SF) have negative correlation (negative correlation score), which means that as the first variable

Birth Age(yrs)	Height(cms)	Gender	Weight(kg)	Body Fat	Visceral Fat	Skeleton Muscle(%age)	Body Age(yrs)	RM(Kcal)	BMI	Systolic BP	Diasttolic BP	Pulse	Suger Fasting	Sugar PP	Waist(cms)
43	155	1000	51.7	20.5	5	35.8	30	1327	21.5	140	88	68	120	125	79
27	143	2000	38	21.2	1	30.5	18	962	18.6	132	70	72	100	120	100
33	165	1000	67.1	23.8	8	32.4	39	1556	24.6	140	85	75	125	150	85
26	163	2000	72	35.9	10	34.4	45	1460	27.1	170	100	95	140	180	90
35	159	2000	66.7	33.5	17	25.5	44	1456	26.4	180	90	98	140	180	100
40	155	2000	45	30.2	6	27.5	50	1645	18.7	125	82	67	190	200	105
50	160	1000	62.6	32.1	3	28.6	61	1632	24.5	130	70	68	120	160	100
29	173	1000	67	29.5	8	38.4	25	1455	22.4	190	100	78	90	124	105
27	168	1000	60.3	27	7	38.1	60	1390	21.4	200	110	79	110	115	95
33	163	1000	76.5	26.8	11	39.5	45	1423	28.8	190	90	94	120	130	85
46	150	2000	68.2	40	14	22.6	61	1353	30.3	150	96	92	200	210	97
58	170	1000	96.9	31.6	26	27.6	71	1970	33.5	160	85	98	180	200	114
45	174	1000	82.5	29.3	15	29.7	56	1769	27.2	150	92	91	200	220	107
28	160	2000	65.6	35.5	8	24.8	43	1339	25.6	118	85	76	170	200	87
43	165	1000	78	30	15	29.2	55	1692	28.7	165	95	93	145	170	102
42	160	2000	92.7	40.3	23	21.2	67	1703	36.2	148	72	94	160	185	119
43	165	1000	86.6	32.2	20	28	60	1810	31.8	180	90	115	200	220	110

Figure 5.5: FBCMI2E (Fitness Dataset) [2]

increases, the second variable decreases.

The pairs (H, VF), (H, P), (H, SF), (H, SP), (G, BF), (G, P), (G, SP), (BF, RM), (BF, P), (VF, P), (BMR, P), (RM, P), (P, SF), (P, SP) have partial correlation (correlation values between 0 to 0.5), which means these pairs do not hold the strong association with each other.

	H	G	BF	VF	BMR	RM	P	SF	SP
H	1.0000	-0.4737	-0.2408	0.0576	-0.0331	0.5131	0.0005	0.0006	0.0413
G	-0.4737	1.0000	0.4399	-0.1532	-0.0604	-0.5321	0.1573	-0.0278	0.0039
BF	-0.2408	0.4399	1.0000	0.5610	0.6705	0.1996	0.2578	0.5774	0.5253
VF	0.0576	-0.1532	0.5610	1.0000	0.8252	0.6205	0.2494	0.7709	0.6978
BMR	-0.0331	-0.0604	0.6705	0.8252	1.0000	0.6687	0.3027	0.7003	0.6627
RM	0.5131	-0.5321	0.1996	0.6205	0.6687	1.0000	0.1104	0.5208	0.5025
P	0.0005	0.1573	0.2578	0.2494	0.3027	0.1104	1.0000	0.2506	0.2921
SF	0.0006	-0.0278	0.5774	0.7709	0.7003	0.5208	0.2506	1.0000	0.8081
SP	0.0413	0.0039	0.5253	0.6978	0.6627	0.5025	0.2921	0.8081	1.0000

Figure 5.6: Correlation Matrix of FBCMI2E

The pairs (H, RM), (BF, VF), (BF, BMR), (BF, SF), (BF, SP), (VF, RM), (VF, SF), (VF, SP), (BMR, RM), (BMR, SF), (BMR, SP), (RM, SF), (RM, SP) have medium influence (correlation values between 0.5 to 0.8) on each other.

Based on the correlation matrix, the variables (H, H), (G, G), (BF, BF), (VF, VF), (BMR, BMR), (RM, RM), (P, P), (SF, SF), (SP, SP), (VF, BMR), (SP, SF) (correlation score $>$ 0.8) can be considered group of variables that influence the fitness index together. The correlation allows us to find strong associations between the variables. A group of all these variables can be considered as function (f(y), where y is FIW or fitness index of work). However, the correlation analysis does not give information about the cause and effect between the variables. Hence, to know which variable or set of variables influence the fitness index, there is a need to do regression analysis. Thus, the problem undertaken here is about classifying a fitness data into two classes (qualified and not-qualified.). Hence, four classifier algorithms (LDA, KNN, DT and NB) have been evaluated after annotation of the each feature using medical limits and ranges (normal and abnormal).

Linear Discriminant Analysis (LDA) algorithm is closely related to the factor analysis, Principal Component Analysis (PCA) and regression. In regression, the variables are continuous dependent variables, but in LDA, the variables are continuous independent. In LDA algorithm, the classes are based on dissimilarity between them whereas in PCA, they does not care about the difference. The factor analysis is a method that makes combinations of factors to find which factor is function of other. In LDA, such kind of interdependence between variables is not sought. LDA works well in cases where there are known (priori) clusters (classes) of data and each class has a score or criteria as the measure [219]. Due to this fact, LDA algorithm was selected for evaluation. K-Nearest Neighbors (KNN) supports non-parametric statistics that can be used for classification as well as for regression. Inputs to the KNN are the parameters of training data in the feature space and output depends upon how KNN is being used (for classification or regression). In this research work, KNN classifier with distance based approach has been used. Decision Tree (DT) is a support tool that uses tree like structures to take decisions. This

technique may be used as one of the way to represent an algorithm that contains conditional statements. With the help of decision tree, one can reach to the target value starting from the observations. Thus, this is one of the predictive modeling approaches that are used in statistics. Decision trees, in which discrete set of values are accepted by the target variable, are called classification trees. Naive Bayes (NB) is a classifier that belongs to the category of probabilistic classifiers.

Table 5.2: Performance of Machine Learning Models on Fitness data

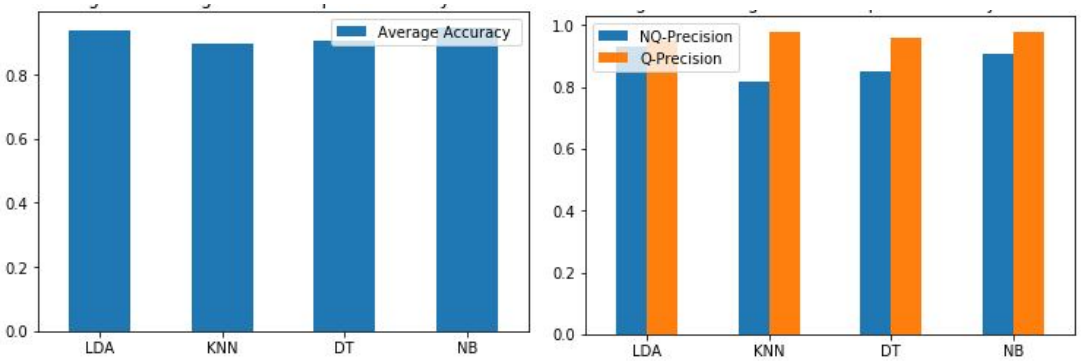
Model Name	Accuracy	Precision		Recall		F1-Score		Support	
		NQ	Q	NQ	Q	NQ	Q	NQ	Q
LDA	0.938144	0.93	0.95	0.93	0.95	0.93	0.95	41	56
KNN	0.896907	0.82	0.98	0.98	0.84	0.89	0.90	41	56
DT	0.907216	0.85	0.96	0.95	0.88	0.90	0.92	41	56
NB	0.948453	0.91	0.98	0.98	0.93	0.94	0.95	41	56

Table 5.2 shows the performance of machine learning models on fitness data. Following are the observations from this table.

(i) The NB algorithm has the maximum accuracy. It seems that probability based calculation of the outcome works well with this dataset. The LDA algorithm is second in producing accuracy. This may also be attributed to the fact that it is able to develop generative probabilistic model accurately on this dataset. Clearly, the probability computations work good for both these algorithms. The DT algorithm works in the third position in terms of its accuracy. This may be attributed to the fact that non-linearity does not become an issue for computing target label.

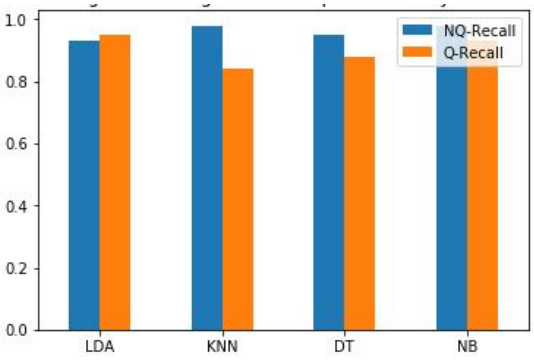
(ii) The NB algorithm has the maximum precision values. Clearly, the proposition of true positive is maximum in case of NB, followed by KNN. LDA has the lowest precision, which means the proposition of positive identification is lowest. It is always desired that the values of recall and precision for each class must be comparable. The algorithm should not be more accurate for a particular class; it reflects bias in learning. However, in our case, it is clear from Figure 5.7(b) that precision values are almost equal (precision values of Q = 0.95, 0.98, 0.96, 0.98; precision values of NQ = 0.93, 0.82, 0.85, 0.91) in both the cases. This means that for given sample of instances, NB algorithm is able to find good level of instance memberships correctly for respective class. It can also be noted that in other algorithms, NQ precision is less as compared to Q precision values. For example, if we check the difference between KNN and NB, precision value of NQ class difference comes out to be 9.89%.

(iii) The NB has the maximum recall value followed by KNN and DT. The trade-off between the recall and precision is playing around in other cases as well. The LDA

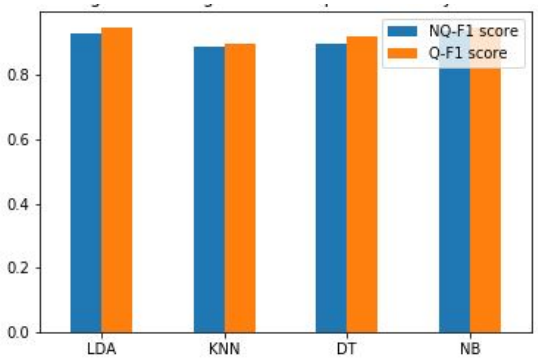


(a) Accuracy of different models

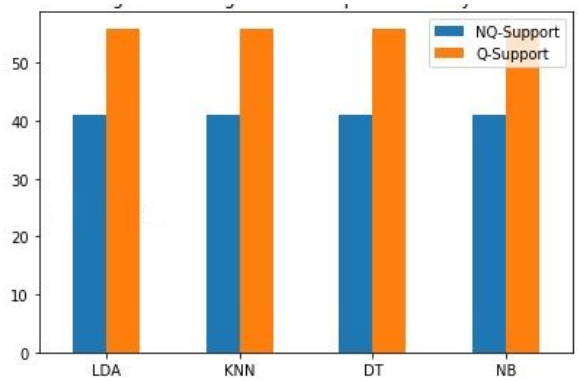
(b) Precision of different models



(c) Recall of different models



(d) F1-score of different models



(e) Support of different models

Figure 5.7: Comparison of Machine Learning Models on Fitness Data

has the lowest recall value among these algorithms. It is always desired that balance between the recall and precision should be there so that false alarms is low. Recall value expresses the ability of the machine learning model to find all relevant feature rows in the feature dataset, which gets classified on the basis of full dataset. It can be observed from the recall graph (Figure 5.7(c)) that NB has the maximum number of instances correctly classified based on full database scan. The observations hold true for both cases individually. LDA algorithm has also comparable recall values due to which it became difficult to choose best performance.

(iv) It is amply clear that the structure of the dataset is such that in almost all cases the support value of Q (qualified class) is higher than the support value of NQ (not-qualified class). This means all algorithms were able to find evidence with respect to membership of particular class.

(v) It can be observed that all four algorithms are competing fairly with each other and their performance is comparable. Due to this fact, another evaluation metric called F1-score has been used to finally select the best performing algorithm. It is always desired that the value of F1-score should be close to 1. Hence, it can be noted that LDA is the runner-up and NB is the best.

The dataset has been labeled and grouped based on the medical ranges, as mentioned in Table 5.3. Thus, this results in two-class problem (qualified or not-qualified) can further be used for automating the process of calculating the fitness index (FIW).

Table 5.3: Medical Ranges of parameters

Notation	**Medical Ranges**	
BF (%age)	**Gender**	**Range**
	Male	10.0 – 19.9 [220, 221]
	Female	20.9 – 29.9 [222, 223]
VF (%age)	1 – 9% [224]	
BMR	18.5 – 24.9 [225, 226]	
RM	Male: 1600 calories per day [227]	
	Female: 1400 calories per day [228]	
P_s	<60 - 100 [229, 230]	
Sugar (mg/dL)	**Fasting**	**After Meal (PP)**
	80 – 120 [231, 232, 233]	80 – 140 [234]

The dataset is almost balanced as it has 5 number of instances of Q and 100 number of featured rows of NQ. In such cases, the evaluation metric precision and recall at micro-level and macro-level are important.
The overall accuracy of the process shows that NB is the best performer (94.8%) among all algorithms. Performance of LDA (93.8%) and DT (90.7%) lag with small difference.

5.3 Inclusiveness Qualification Computations for FBCMI2E

The qualification of the participants will be based on four types of scores and for each score there is a qualifying criteria discussed in each sub-section below. The qualification will be represented with equation 5.2 (dataset [235]).

$$Q_{qualified/not\_qualified} = \{FIW, ICA, CLA, GSA\}, \tag{5.2}$$

where, FIW is the fitness index of work (computed in Section 5.2), ICA is the inner circle analysis (Section 5.3.1), CLA is the mobile call log analysis (Section 5.3.2), GSA is the geospatial analysis (Section 5.3.3).

The next section explains how the data is simulated so that FBCMI2E can be constructed. Section 5.3.1 gives explanation of how communities for a particular entity are identified so that calculations related to modularity can be done, which finally contribute to qualification criteria.

5.3.1 Inner Circle Analysis

The second most important factor for qualification of the inclusive program is inner circle analysis. The computation of the qualifying score i.e inner circle trust score is also calculated based on the analysis of the calls done by the participant every week. The purpose of all call log analysis is to find the inner sociological circle of the participant. If the participant has a healthy social circle, then the participant seems to be socially, mentally as well as physically healthy. The credit and financial analysis of the person can come later to maintainability of the status quo in the scheme for inclusiveness. For the call analysis the problem can be considered as construction of graph. A graph G_1 is a set V of vertices and a collection E of pairs of vertices from V, called edges. This way, the network of related relationships between different pair of entities from set V are formed [236].

An edge (u,v) is said to be directed from u to v if the pair (u,v) is ordered with u preceding v. An edge (u,v) is said to be undirected if the pair (u,v) is not ordered. Undirected edges are denoted with set notation as $\{u,v\}$, but for ease we use the pair notation (u,v). Thus, in the undirected case (u,v) is the same as (v,u). Edges in graph are either directed or undirected [237].

Figure 5.8(a) and Figure 5.8(b) visualize the collaborations of participants involved in the program. How they interact with their family, friends, relatives, acquaintances, and other circles of their life. The Louvain algorithm partitions the initial community graph into its inner circle. The graph of inner circle represents the modularity of her/his inner community.

5.3.1.1 Simulating Communities Using Erdos Renyi Pseudo Logic

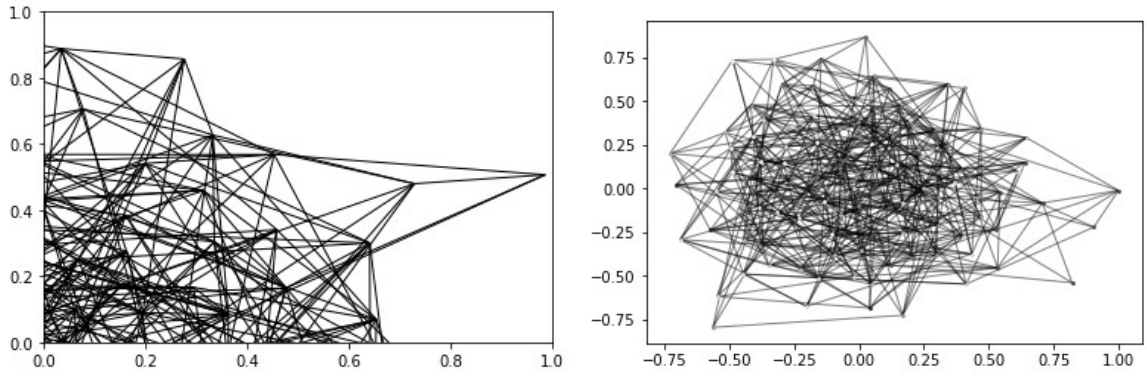
This algorithm is used for simulating the socio-economic circle/community which needs to be analyzed for inclusiveness program. The pseudo logic of Erdos Renyi algorithm is shown in Algorithm 7.

Algorithm 7 Erdos Renyi Pseudo Logic

1: *Let n be the number of nodes*
2: *Let p be the probability of edge creation*;
$$p \in \{0,1\};$$
where 1 *is maximum probability of node creation*
3: *Let t be the type of graph*; $t \in \{directed, undirected\}$
4: *Let* $G(n,p)$ *be a graph constructed by connected random events*
5: *In each iteration,* $G(n,p)$ *selects all possible edges having same probability p computed using equation* 5.3

$$p^{M}(1-p)^{\frac{n}{2}-M_e}, \tag{5.3}$$

where n represents nodes and M_e *are the edges.*



(a) Full Social Circle (b) Inner Circle Detection

Figure 5.8: Community Detection of FBCMI2E

5.3.1.2 Community Detection Louvain Algorithm

The working of Louvain algorithm can be explained in three steps. In the first step, the algorithm gives greedy assignments of the nodes to the community, in which it tries to optimize local community based on the value of modularity. In the second step, the definition of the new network are replaced with new communities and the whole network is updated. The third step implements the termination condition based on which it stops iteration as it cannot assign better degree of modularity for all the communities.

Figure 5.8(a) shows the interaction of the person in society based on the calls he or she makes. It is apparent that the call made by person will either personal or will be made for his/her profession. In context of the problem, there is a need to identify the inner social circle with whom the incumbent is more or less in contact on everyday basis. Figure 5.8(b) shows the inner circle of the persons, computed using Louvain algorithm. The detection of the inner circle shows the modularity of 0.25 (modularity(M) = 0.25924783471234336), from which it can be inferred that the incumbent fair degree of association with his or her inner circle on daily basis. From this, trust can be quantified and used for qualification purpose for the inclusiveness program.

5.3.2 Mobile Call Analysis for FBCMI2E

The mobile call analysis helps us to compute the stability of the social life and level of activeness the person is having. The call log data table (Figure 5.9), gives clear information about the most frequent calls that are initiated by the participant having mobile number 8225612216 with 26.0 support value per day (Figure 5.10). This set of calls is used for grouping up the participants using FP-Growth algorithm and it shows that participant communicates most frequently with these people. This process is continued for over six months or so. It can be inferred that the participant has good level of stable socio-economic relationship with people.

RegistrationID	ID	Date	CallEvent	M	M1	M2	M3	M4
1234	189399694	1122018	6793910055	2235493489	3487725493	5352174393	5622186604	4204686376
1234	167585077	2122018	3295759919	9329945805	7674273526	5247382724	6259222973	9292364001
1234	392963298	3122018	1401170901	7308937431	9139515708	3288433262	9668785533	8312605135
1234	470516402	4122018	4298991317	7566584939	5537798820	3328371616	4882734312	4506829259
1234	757582261	5122018	9216219338	3607309661	6683058257	8839582825	9462858638	9278378762
1234	468055209	6122018	3071017970	6658791482	7876359214	7447813541	6492224825	4868571238
1234	506457844	7122018	2229691249	4627519135	8565604216	6197788959	5373741090	7194457024
1234	94322301	8122018	8667931329	4566393891	7043424299	3008622454	5633641861	3556901265
1234	779766902	9122018	7890161114	7088238935	2859732089	3419967174	4853684518	8225051844
1234	527330834	10122018	6777016943	9452874510	6524645155	8115972634	9837272436	4603724249
1234	631059504	11122018	9667230055	8384086261	4426374396	8283122552	5822153306	8102556058
1234	683276281	12122018	171610006	2703206514	3896472263	4564727910	5395191253	9689925550
1234	590517638	13122018	3378198299	8012653663	5842745103	4489963564	8232897574	7444406702
1234	656672008	14122018	9451129756	7789631257	7707418138	3162704496	3289413673	3197548230
1234	420357611	15122018	8920463609	9657997531	8426482631	5062518961	5748302628	6466436205
1234	54165874	16122018	748732134	3849331730	8292604584	7124567690	2496326698	7599109183
1234	405673236	17122018	881629989	2497227479	6783728777	3535944291	2864789622	8192311565
1234	371164425	18122018	2591208162	5694819948	8814049892	5838474002	4075086109	6146891912
1234	979026432	19122018	6863634075	4262702415	8215336231	3127039167	7775778357	4304367884
1234	467812522	20122018	6280735612	3127288248	7166127375	5763722018	9574904545	7674465701
1234	689514488	21122018	6651610090	4606816573	3884387844	3448256879	8623591087	4823181124
1234	525015022	22122018	4034603668	2798844050	6139708663	2179247181	3178727006	6747094101

Figure 5.9: Call Analysis of FBCMI2E Dataset

The algorithm (Algorithm 8) helps to identify the most frequent patterns of calls of an incumbent and it tells that how many times the person calls other person. The value of support is an indicator of how frequently the call numbers appear in his call log. This way we are able to identify how stable is the relationship of person with other person in general. Stable social relationship means that the person is socially engaged and useful to the society and has stable family and business circle. The score of social stability can be computed with the help of support value.

The purpose of considering these factors and running this algorithm is to identify most frequent patterns of her/his calls. It would help to identify n number of sub-groups. Sub-groups are the group of individuals with whom the incumbent has stable relationship, hence talks to them in routine. This helps to identify only those individuals and groups

Algorithm 8 FP-Growth Algorithm for Mobile Call Data Analysis

Input: call data(RegistrationID, SubscriberID, Date, CallEvent, sourceMobile, destinationMobiles), minimum support value

Output: set of frequent patterns showing how frequently a person calls to different mobile numbers

Method: call FP-calldata(FP-calldatatree)

```
identify prefixes
iterate unique mobile number with prefix path
for each path in path do
    generate call data patterns
    collect support score of call patterns
    if support_patterns ≥ minsupport then
        keep(pattern)
    else
        ignore(pattern)
        update FP-calldatatree
        return {call pattern, support score}
    end if
end for
```

that are statistically significant and have high degree of positive statistical dependence. Higher support value refers here is the group of mobile numbers that are called in routine by the participant of the program.

5.3.3 Geospatial Analysis for FBCMI2E

The next step is compute the mobility factor. The innate nature of formal economy is temporal high degree of mobility, migration and immigration. This step consists of conducting a geospatial analysis of the GPRS data (Figure 5.11) and then visualization of the routes taken by the subject. The person itself is considered as a sensor; the physical location of a person can be captured in three ways for achieving this aforementioned goal. First from the mobile sensor that takes GPRS data and second from the wearable medical device. In both the cases a frequency analysis can be done for spotting the route which participant follows and the spot where she or he spends maximum time such as home or temporary work space. The algorithm (Algorithm 9) helps to identify the most frequent GPRS patterns of the participant. The GPRS coordinates will tell about the stability of physical presence (support value of 14.0). This way the actual score of the person can be compared with qualifying score matrix to arrive at a decision of including the person in main stream industry, and later on benefits of formal economy can be given to her/him.
Figure 5.12 shows the person that is most likely to be stationed or visits the place at (50.67856,3.138032) with support value of 14.0 in routine. Apparently, the place may be his home or office. From this, it can be inferred that the person has a stable place to stay or is doing some professional work at that place.

7674273526	5247382724	6259222973	9292364001	2235493489
7043424299	9815816740	2235493489	2235493489	4784703386
9815816740	8384086261	8384086261	8384086261	4784703386
8384086261	3535944291	2864789622	8192311565	7447813541
8215336231	3127039167	7775778357	4304367884	7447813541
4318215135	4327518630	7633366673	5032036655	9693171297
6197806142	4252773443	9864545117	7404894473	4383634313
3648421905	5853489322	4252773443	8383864796	8232897574
4252773443	6836852808	6226088384	8655037203	4257799874
2112509189	4577914083	9213365657	5189043191	8232897574
3424126660	2186505508	8396352315	6718044623	7447813541
6349912762	2356854930	4094183388	2602609156	5838474002
2103728789	6524005690	3706585502	3407109373	5838474002
5383627333	4257799874	6425535714	5023619833	4257799874
2393192253	6374386705	9607324856	5643601723	5838474002
5338636116	6135404861	5544949052	6096615257	2235493489
9268266769	8179075234	3003177401	3834917300	4784703386
2608242029	9429509031	5338636116	4069948015	4257799874
7009375225	7393002994	6737659431	2303783991	2728712360
4465969241	4167319397	3017204467	5184379767	8232897574
3822231927	5769398298	7513991187	7513991187	4257799874
4309265261	2676508193	2778184194	6462225380	4784703386
2343767888	7765739528	9476293991	9727446198	7175455565
3358388764	3558754110	9059764126	7298062924	4257799874
9059764126	2297441943	3436218386	3884338101	9089719225
3094841975	2564255217	5762535520	5802925253	9089719225

```
(7766213267,)  support = 100.0
(6926, 7766213267)  support = 100.0
(9814, 7766213267)  support = 100.0
(2122018, 7766213267)  support = 26.0
(6926, 9814, 7766213267)  support = 100.0
(6926, 2122018, 7766213267)  support = 26.0
(9814, 2122018, 7766213267)  support = 26.0
(6926, 9814, 2122018, 7766213267)  support = 26.0
(5583,)  support = 100.0
(5583, 2122018)  support = 26.0
(4113,)  support = 100.0
(4113, 5583)  support = 100.0
(4113, 2122018)  support = 26.0
(4113, 5583, 2122018)  support = 26.0
(8225612216,)  support = 100.0
(4113, 8225612216)  support = 100.0
(5583, 8225612216)  support = 100.0
(2122018, 8225612216)  support = 26.0
(4113, 5583, 8225612216)  support = 100.0
(4113, 2122018, 8225612216)  support = 26.0
(5583, 2122018, 8225612216)  support = 26.0
(4113, 5583, 2122018, 8225612216)  support = 26.0
(3122018,)  support = 110.0
(3122018, 9815816740)  support = 45.0
(2122018,)  support = 130.0
(2122018, 9815816740)  support = 41.0
(9815816740,)  support = 208.0
(9815816740, 9815816740)  support = 72.0
```

Figure 5.10: Result of Call Analysis Using FP-Growth

Algorithm 9 FP-Growth Algorithm for Geospatial Data

Input: geospatial data(RegistrationID,SubscriberID,date,latitude,longitude), minimum support value

Output: set of frequent patterns showing how frequently a person is physically available at different latitude and longitude positions

Method: call FP-geospatialdata(FP-geospatialtree)

```
1: identify unique prefixes of geospatial data(longitude,latitude)
2: iterate Tree with prefix path
3: for each path in path do
4:     generate geospatial patterns
5:     collect support of geospatial patterns
6:     if support_patterns ≥ minsupport then
7:         m_i = m + 1
8:         add(pattern)
9:     else
10:        minimumsupport = 0
11:        ignore
12:        update FP-geospatialtree
13:        return {geospatial pattern, support}
14:    end if
15: end for
```

This section computes the values for four parameters on which qualification criteria depend (equation 5.2). Next section uses these values as inputs and computes qualification criteria for each parameter. Qualification cutt-offs for geospatial events (Section 5.4.1), mobile call events (Section 5.4.2) and measure of modularity (Section 5.4.3) has been computed. On the basis of qualification criteria, every participant will be judged for her/his qualification for inclusiveness program.

5.4 Automating FBCMI2E Using Machine Learning

In this section, machine learning algorithms have been compared to develop a fully automated process of identification of the subjects that have successfully completed the inclusive program. The section begins with explanation of the method by which threshold or cut-off is calculated for converting the problem into two class problem (qualified or not-qualified). The method employed is identifying the mean of groups and comparing the mean of each qualification variable with the mean of the individual. If the mean of the individual is below the global mean, then the person is considered not suitable for the program.

RegistrationID	PersonID	Date	Latitude	Longitude
9872	61762798	6092018	44.08179	6.001625
9872	11522199	6092018	44.06157	5.997373
9872	9295871	6092018	44.06386	6.011248
9872	62518989	6092018	44.35	6.35
9872	40660458	7092018	44.08871	6.222982
9872	46772404	7092018	43.14494	6.068766
9872	60392074	7092018	43.70385	-0.258269
9872	59554260	7092018	43.65813	7.14864
9872	15224359	7092018	43.77649	7.504349
9872	49198516	7092018	43.77444	7.4933
9872	20730811	8092018	43.7852	7.517297
9872	21077785	8092018	44.11256	7.562798
9872	96597121	8092018	43.5239	6.941313
9872	91100393	8092018	44.55	2.25
9872	26383065	8092018	44.56143	2.255896
9872	35906490	9092018	44.55557	2.218258
9872	65228191	9092018	44.55564	2.263741
9872	8329715	10092018	48.83333	-1.033333
9872	69535160	10092018	48.83952	-1.049283
9872	41613241	10092018	48.84361	-1.051698
9872	98027048	10092018	48.83897	-0.88928
9872	34825729	10092018	48.83764	-0.880538

Figure 5.11: Geospatial Analysis of FBCMI2E Dataset

5.4.1 Qualification Cut-Off for Geospatial Events

The summarization of the frequencies of events (Figure 5.13) shows that a total of 21812 events have been fired by five participants in a week. Which means that on an average about 4362 events were triggered per week per participant. The average location events recorded per day will be $4362/7 = 623$ per day per participant.

From computing this cut-off, this can be found that how active the person is. If the number of geo-spatial events recorded by the person is far more than the global mean, then this means the person has qualified and is actively moving around doing daily tasks. Similarly, if the participants mean comes out below the global mean, it infers that he or she is active below average among that segment of participants.

Figure 5.12: Result of Geospatial Analysis

5.4.2 Qualification Cut-Off for Mobile Calls events

Higher number of calls done by the participants would mean that the person is at least active with the socio-economic circle. And the geo-spatial will give indication how the

Sum of G	Column Labels					
Row Labels	7766213267	8225612216	9772855482	9776632312	9815816740	Grand Total
1122018	430	430	358	366	399	1983
2122018	1268	1342	1023	884	920	5437
3122018	1016	1164	1061	775	804	4820
4122018	408	556	502	354	369	2189
5122018	612	908	716	494	555	3285
6122018	408	482	442	423	364	2119
7122018	382	530	403	328	336	1979
Grand Total	**4524**	**5412**	**4505**	**3624**	**3747**	**21812**

Figure 5.13: Geospatial Events Cut-Off

Sum of C	Column Labels					
Row Labels	7766213267	8225612216	9772855482	9776632312	9815816740	Grand Total
1122018	504	652	292	301	304	2053
2122018	1712	2008	1131	1096	946	6893
3122018	1312	1608	1022	909	826	5677
4122018	630	778	475	496	356	2735
5122018	760	1130	749	704	556	3899
6122018	482	778	475	427	397	2559
7122018	456	678	303	327	360	2124
Grand Total	**5856**	**7632**	**4447**	**4260**	**3745**	**25940**

Figure 5.14: Qualification Cut-Off of Mobile Calls Events

person is active physically. If the person is walking or doing some physical activity such as running or is moving in a car, the geo-spatial analysis will give a high average values. However, in case of calls, it is assumed that in a day the person is at least active for 12 hours and has a contact with the outside world through the mobile phone at least 6 times. This value can be considers as axiomatic value. Figure 5.14 shows the computations for qualification cutt-off of mobile calls events.

Sum of M	Column Labels					
Row Labels	7766213267	8225612216	9772855482	9776632312	9815816740	Grand Total
1122018	3.2497532	2.64717795	3.051524199	2.63378479	3.119051919	14.70129206
2122018	9.273598502	8.106691701	9.18382719	9.697990765	8.668555099	44.93066326
3122018	7.920237549	7.182698997	8.294693425	7.05218649	7.751553485	38.20136995
4122018	3.37795016	3.358654198	3.32337298	3.492512438	3.362350263	16.91484004
5122018	5.124468587	4.64600145	5.198079353	5.500371335	4.886086352	25.35500708
6122018	3.795597259	3.268436302	3.67038215	3.301428826	3.639996479	17.67584102
7122018	3.10919634	2.734789089	3.2987829	3.140555983	3.12978807	15.41311238
Grand Total	**35.8508016**	**31.94444969**	**36.0206622**	**34.81883063**	**34.55738167**	**173.1921258**

Figure 5.15: Measure of Modularity

5.4.3 Measure of Modularity

The measure of modularity gives information how closely knitted the person is in the society or inner circle (Figure 5.15). It gives strength of the association and degree of socio-economic bond the person may hold with given set of socio-economic connections. Higher values of modularity are always desired for a participant of the inclusive program. The values of modularity move in a bracket of $[-1, 1]$. The value of 0 shows that the person's inner circle size is too small and as the value moves towards -1 then it can stated that he or she is not socially active. As the value moves towards 1, it shows that the person is active and has large socio-economic network. In this case, a value of 0.20 is considered as a cut-off based on which the qualification maybe decided. Thus, this information about qualifying/not-qualifying criteria can be summarized in Table 5.4.

Table 5.4: Qualifying/Not-Qualifying Criteria

Parameter	**Qualifying Criteria**	**Not-Qualifying Criteria**
Geospatial	$\leq$623	$>$623
Call	$\leq$6	$>$6
Modularity	$\leq$0.20	$>$0.20

Table 5.5: Performance of Machine Learning Models on Fitness data

Model Name	Accuracy	Precision		Recall		F1-Score		Support	
		NQ	Q	NQ	Q	NQ	Q	NQ	Q
LR	0.632	0.63	0.64	0.88	0.30	0.73	0.41	72	53
LDA	0.752	0.85	0.67	0.69	0.83	0.76	0.74	72	53
KNN	0.928	1.00	0.85	0.88	1.00	0.93	0.92	72	53
SVC	0.424	0.00	0.42	0.00	1.00	0.00	0.60	72	53
NB	0.544	0.59	0.45	0.67	0.38	0.63	0.41	72	53
NN	0.632	0.62	0.71	0.93	0.23	0.74	0.34	72	53

NAN values may happen due to malfunction of instruments collecting data (eg. sensors). The performance metrics in this research work are computed on a dataset, which do not have any missing NAN values. The ratio of qualified and not-qualified instances is almost equal. This means the database does not suffer from any kind of imbalance of the class. The dataset was treated for all these qualities. The validation ratio has kept at 0.20. This has been taken because this covers atleast 20% of the dataset for validating the performance of the dataset.

It can be observed from the Table 5.5 that KNN is performing best in terms of accuracy. Consequently, recall and precision is also good as compared to other algorithms. The experimental evaluation of all these algorithms show that KNN is easy to implement and

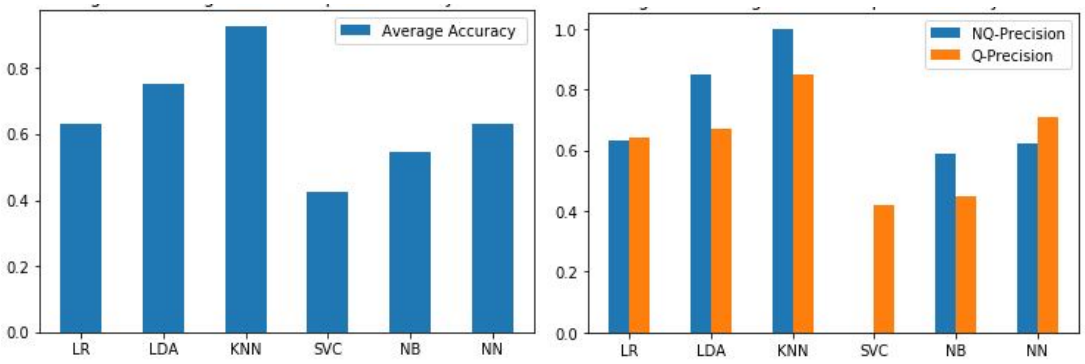
have good value of predictive power. It can also be seen that KNN algorithm with the value of $K = 5$ is able to differentiate boundaries of the two classes in better way as compared to other algorithms. The LDA and NN are runner up in term of accuracy. The precision method computed for each class also shows that KNN performs better individually (at micro-level). However, in case of SVC, the recall values is high at micro-level for finding the balance between the recall and precision. The F1-score is computed and it is found that for both the classes the metric (F1-score) is able to perform better.
Thus, KNN algorithm performs better due to the fact that it is able to identify similar neighbor(s) using distance approach even on the non-linear data. Secondly, the results are stable even when the data has some amount of noise as it computes distance on the basis of inverse square of weighted distance.

5.5 Energy Analysis for FBCMI2E

The success of this model is inevitable as the level of penetration of mobiles, sensors and apps in many demographic regions has increased. The case presented here gives us chance to think about the problems and issues that may be faced by the implementers of the proposed system and algorithms. Energy overhead of devices and other infrastructure may be one of the impeding factor of slow adoption. The power/energy requirements for monitoring 24x7 may require special quality of batteries for system. The financial cost may also be the barrier that may impede the growth of such initiatives. This section discusses all these parameters and check the feasibility of such initiatives according to the current market and technological socio-economic conditions, in context of energy requirements. Following are the assumptions.

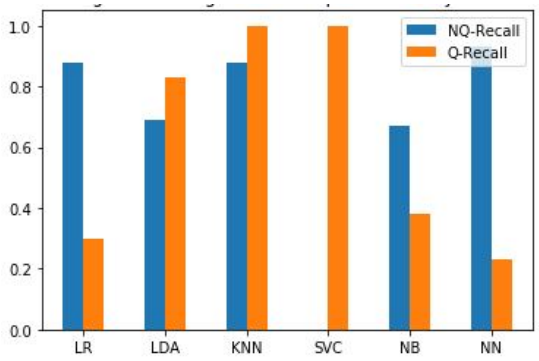
(i) Power consumption: It is assumed that ultra-low power consuming devices will be employed to collect data. These devices will be attachable to the body of the subject and will consist of Wi-Fi and sensor model that send information to the data aggregators. These wearable devices will battery powered and will have water proofing casing as well. Table 5.6 shows the parameters' requirements for each component.

(ii) Network requirements: The network required for the deployment of such inclusive program will have nodes of various hardware sizes, software configurations and multiple communication models eg. Wi-Fi, Bluetooth etc. It can effectively be called as heterogeneous networks. All the devices will work in a asynchronized manner and multi-devices would collect and forward data from fog zone to cloud zone. Assuming that each sensor communication and sensing model (sense-process-actuate model) are almost same.

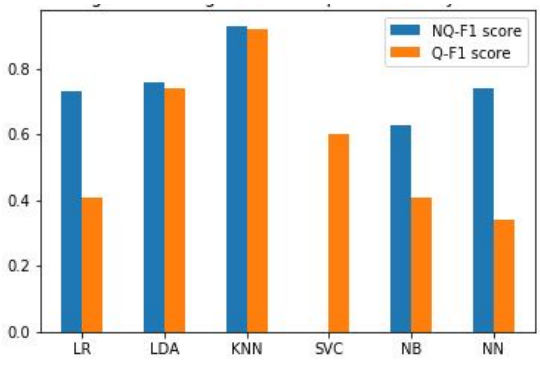


(a) Accuracy of different models

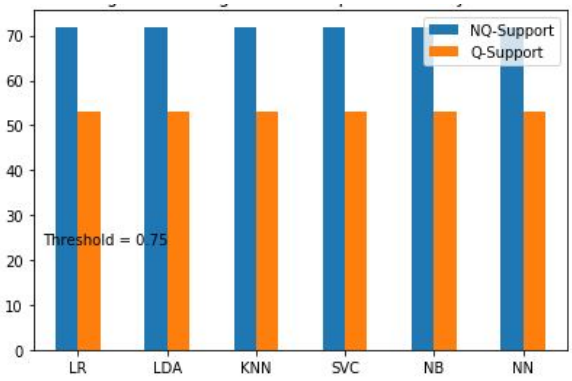
(b) Precision of different models



(c) Recall of different models



(d) F1-score of different models



(e) Support of different models

Figure 5.16: Comparison of Machine Learning Models

(iii) Pattern of Events: Theoretically, the energy consumption of a single sensor node and the full network can be modeled mathematically based on multiple conditions and scenarios.

Taking the case of inclusive program mentioned in this research work, mathematical equations for energy computation of heterogeneous devices (Section 5.5.1), Nano-Cache nodes (Section 5.5.2), computational intensive nodes (Section 5.5.3) and total energy expenditure (Section 5.5.4) can be used to model the case of inclusive program. The data events may be temporal in nature and the event may follow correlated or uncorrelated patterns due to which the overhead in terms of energy

Table 5.6: Parameters Requirements for FBCMI2E

Parameter	Definition	Sensor Required
BF	Body fat percentage refers to the amount of body fat mass in regards to the total body weight expressed as a percentage	RENPHO Bluetooth Body Fat Scale
VF	Visceral fat refers to the fat stored in abdomen and surrounding vital organs	Omron HBF385
BMR	BMR is the rate of burning calories at rest	RENPHO Bluetooth Body Fat Scale
RM	RM is the number of calories burnt in maintaining vital body processes in a resting state	Omron HBF385
Ps	Pulse is defined as the rhythmic beating in the arteries caused by beating of the heart	Omron HBF385
SugarPP	SugarPP refers to level of glucose (sugar) in blood sample after meal	Omron HBF385
SugarFF	SugarFF means level of glucose (sugar) in a blood sample after overnight (8 hours) fasting	Omron HBF385
Other Requirements		
Program/ Parameter	**Equipment Required**	
Call Analysis	Mobile Phones	
Geospatial Analysis	Mobile Phones with GPS support	

gets impacted.

It is assumed that the correlated events will be based on uniform distribution and uncorrelated events will be based on Gaussian distribution. Secondly, the correlated events are distributed with standard time period.

5.5.1 Energy Model of FBCMI2E for Heterogeneous Device

Typical, the model required for computing the energy in context of the inclusive program may require 2-level (equation 5.4), 3 level (equations 5.5 and 5.6) or n-level (equations 5.7 and 5.8) of rank and hierarchy to manage the inclusive program network. There will be n levels of fog devices working under a mesh topology where a device may work as server and a client at the same time. Thus, impacting energy consumption pattern. The equations 5.7 and 5.8 give n-level equations. The network will have mix of source initiated and destination initiated fog/cloud devices. These equations computes the energy in such situations. Table 5.7 shows the heterogeneity parameters involved in FBCMI2E.

$$E_{total} = N * (1-m) * E_0 + N * m * E_0 * (1+a). \tag{5.4}$$

Table 5.7: Heterogeneity Parameters for FBCMI2E

Variable	Notation
Number of fog devices	N
Fraction of computation intensive nodes	m_0
Fraction of Nano-cache nodes	m
Fraction of normal Nano-cache nodes	$1-m$
Total number of computation intensive nodes	$N*m*m_0$
Total number of Nano-cache nodes	$N*m*(1-m_0)$
Initial energy of the normal Nano-cache nodes	E_0
Extra energy in the Nano-cache nodes	a
Extra energy in the computation intensive nodes	b
Energy of each computation intensive node	$E_0(1+b)$
Energy of each Nano-cache node	$E_0(1+a)$

*All Energy computations are in Joules (J).

$$E_{total} = N*(1-m)*E_0 + N*m*(1+m_0)*E_0*(1+a) + N*m*m_0*E_0(1+\beta). \quad (5.5)$$

$$E_{total} = N*E_0*(1+m*(a+m_0*b)). \quad (5.6)$$

$$E_{total} = \sum_{i=1}^{n} E_0*(1+a_i). \quad (5.7)$$

$$E_{total} = E_0*\sum_{i=1}^{n}(1+a_i). \quad (5.8)$$

5.5.2 Energy Model of FBCMI2E for Nano-Cache Nodes

Cache devices will have heavy duty cycles hence will consume more energy as compared to the normal nodes in working. This energy will be directly proportional to hit ratio of Nano-Cache server, as shown in equation 5.9.

$$E_{ch} = (m)x + W, \quad (5.9)$$

where m is current hit rate, x is the unit of energy and W is weight. All energy computations are in Joules(J).

5.5.3 Energy Model of FBCMI2E for Computational Intensive Nodes

The computation capabilities make difference in terms of overhead and energy consumption cost. This computation energy depends upon number of devices, number of CPUs and memory, as shown in equation 5.10.

$$E_{compute} = (m_1 + m_2 + m_3) \cdot (x_1 + x_2 + x_3) + (W_1 + W_2 + W_3), \tag{5.10}$$

where $E_{compute}$ is the computation energy, m_1, m_2, m_3 are memory (in kb), x_1, x_2, x_3 are CPUs and W_1, W_2, W_3 are weights.

5.5.4 Computation of Total Energy for FBCMI2E

The total energy expenditure of the inclusive program will be sum of all the energies calculated in the equations 5.4 to 5.10. In terms of the mathematical relationship, the total energy spent will be directly proportional to the number of subscribers and devices involved in the program. This energy depends upon number of subscribers, number of mobile devices and number of fitness sensors, as shown in equation 5.11.

$$T_{eexp} = \sum (aE_{ch} + (E_{compute} \cdot s \cdot n_m \cdot f_n)), \tag{5.11}$$

where T_{eexp} is the total energy expenditure, a is weight and $a \neq 0$, s is the number of subscribers, n_m is number of mobile devices and f_n is number of fitness sensors.

5.6 Summary and Conclusion

This chapter discusses the application of fog load balancer by giving a technological solution on how segregation of people can be done easily with the help of FBCMI2E. Using this application, the people who do not belong to main stream economics can be made inclusive into the society. The proposed model is based on the trust and credibility. Although, these concepts are intangible but they can be quantified by using statistical and data mining methods.

FBCMI2E identifies the frequency between the most visited locations using GPRS coordinates. Using similar logic, the program computes the social network score based on the frequency of the calls to inner circle and outer circle of the participant (who needs to be given the facility). The location histories of a person are basically a spatial-temporal data. The record of such real-world location histories implies that we can track individuals to a great extent and identify the mobile users' home location as well. Thus, by using both spatial and temporal aspects, trust matrix for inclusion program is constructed. This is done by computing the activity points and then grouping them to determine the place visited maximum number of times within a particular time period which reflects the stability

of a person.

This chapter solves the classic problem of finding right candidates for inclusion program, that too, by using non-financial characteristics. This research work demonstrates this with simple steps. After the registration of the person into the inclusive program, the program calculates four aspects (i.e. fitness, inner social-economic circle, geospatial and call analysis) to finally arrive at qualification criteria. After collecting information on all these aspects, the process of quantification for inclusiveness begins, using qualification criteria.

The process is automated using machine learning model. After multiple series of experimentation, it was found that NN gives the best accuracy. The computation of fitness index dataset [2] was also automated and subjected to machine learning modeling. It was found that NB algorithm works best to classify the person on the basis of nine fitness factors of the participants. For geospatial and call analysis, FP-Growth data mining method is used that helps to identify the patterns about behavior of the person. The inner circle detection and modularity computation are computed using Louvain algorithm.

The outcomes of such analysis help us foresee the possible way by which the inclusive growth may happen in context of inclusive banking, inclusive finance policies, inclusive health care, insurance and inclusive overall development of informal economy.

Chapter 6

Conclusion and Future Directions

This chapter presents concluding remarks on the thesis by highlighting the principal contributions of this research study. Energy efficient load balancing is a challenging task in fog computing due to potential involvement of extremely large networks of heterogeneous devices, heterogeneous resources, heterogeneity and fluctuations in the applications' workload. This research work set out to propose energy-efficient load balancing algorithms while respecting QoS requirements of the end users. This thesis efficaciously addresses the energy efficient load balancing algorithms in fog computing through the proposed design level and optimization algorithm of fog load balancer. The proposed fog load balancer considers energy efficient parameters at design level; the optimization algorithm MTLBO also uses energy efficient parameters. Further, the devised fog load balancer is used to propose the case study on developing qualification criteria using data mining (FP-Growth algorithm) and machine learning algorithms for social well-being. This qualification criteria helps people living at fringes of the society to be included into the main fold economy. The chapter details the outcome of each chapter and later highlights the contributions of proposed energy efficient load balancing algorithms in fog computing. Subsequently, the last section of this chapter covers the ways and methods by which this research work can be further extended and meaningful contributions to the society can be made.

6.1 Conclusion

The aim of this research work has been to design and develop energy-efficient load balancing algorithm for fog computing environment, which has been addressed by the proposed fog load balancer, comprises of proposed design level and devised optimization algorithm.

The efficiency of the fog networks depends upon the type of design adopted and the effectiveness of optimization algorithm used to manage traffic load. The objectives set out for this thesis are achieved in phases. The brief summary of each phase is as follows:

(i) In the initial phase, in depth review and analysis of existing literature has been carried out to identify the gaps existing in the area of traffic load management, energy-efficient and load balancing algorithms in fog computing (Chapter 2). The classification of related work on various factors has been done. More specifically, comparative study of designs as well as optimization algorithms for load balancing have been carried out. The existing literature analysis helped to identify research gaps and challenges in the area of energy efficient load balancing algorithms in fog computing.

(ii) To address the design issues of energy efficient fog load balancer, fuzzy logic based design levels (3-level, 5-level and 7-level) are evaluated and validated with different traffic types (sensor data, video data and web data), as discussed in Chapter 3. The design level of devised fog load balancer is constructed on the basis of fuzzy distributions, traffic analysis, rule analysis and rule generation mechanisms, and the outcome clearly shows that deeper levels of fuzzy control is not good. It is concluded that most appropriate and accurate way of managing the traffic load in fog zone is 3-level fuzzy design. This is because the 5-level and 7-level fuzzy designs have lot of overlapping and redundant candidate rules that leads to reduce energy efficiency and level of accuracy.

(iii) To address energy efficient optimization algorithm for fog load balancer, an algorithm namely Modified Teaching Learning Based Optimization (MTLBO) has been contrived and employed in fog zone (Chapter 4), that finds an efficient route for forwarding contents using Nano-Caches and subsequently improves the content retrieval time. MTLBO is compared with existing algorithms, namely, Teaching Learning Based Optimization (TLBO) algorithm and Simulated Annealing (SA) algorithm for identification of better energy-efficient load balancing algorithm, and MTLBO is found to be the best optimizer for energy and memory overheads in fog networks. The results show that MTLBO is better than TLBO as it has less overheads in terms of memory (considering number of fog caches) and network size for delivering contents at remote areas. In comparison to SA algorithm, MTLBO performs better in terms of execution time, overhead in terms of memory, and scalability as function of network size. The MTLBO has faster learning rate as well. Hence, this algorithm is able to optimize and cover the solution space faster with higher level of efficiency. The result also shows that adding Nano-Caches is a feasible solution in context of solving latency issues at edges of the network, and it

requires Nano-Cache management software component (optimization algorithm) to perform better than legacy solutions.

(iv) The design level as well as optimization algorithm, presented in Chapter 3 and Chapter 4, formulate an energy efficient fog load balancer. The research work uses this fog load balancer to present an application of fog computing (Chapter 5) that formulates a system for providing a base model for socio-engineering work. The key contribution is to design the system, that can help to bring a person into formal economy, without the usage of economic instrument such as credit score or rating. A unique system of computing inclusiveness score has been introduced and implemented using fog computing. The application of fog computing considering social inclusiveness demonstrates whether respective participant is eligible to be considered for formal economy or not. This was based on qualification criteria that comprises of four parameters i.e. fitness index of work (FIW), call analysis (CA), geospatial analysis (GA) and modularity (M). Machine learning algorithms have been used to predict the accuracy of the results. In this research work, various types of sensor data have been used to construct an algorithm for identifying beneficiaries of government schemes. This work will help in justifiable and inclusive growth of all people, regardless of sex, age, race, background, and persons with infirmities, refugees, aboriginals, children and youth, particularly those in vulnerable circumstances. These people will have access towards opportunity to grow in high growth sectors of economy. Such initiatives will promote both physical and psychological health which subsequently increases the life expectancy for all.

6.2 Future Directions

The research work presents a way of doing link analysis at interconnects of mobile, fog and fuzzy boundaries. It is analogous to routing updates between the various components of the fog, cloud and mobile networks. For future directions, the work can be extended to handle routing failures in more robust manner and network convergence issues in case load balancing/traffic handling suffers from any issue.

It is also suggested that this work may be extended to construct solutions based on 'transparent caching technology'. This concept allows the deployment of similar scenarios as presented in the current work. This can be achieved in cases where the subscriber-publisher model is not relevant and the content material is not premium or costly in nature as compared to content retrieval time. The optimization algorithm can be applied to the edge nodes at the outer zone of the cloud networks to maintain high quality of delivery.

Not only this, in coming days there will an explosion of data volume related to fitness and health insurance. The impact of big data and artificial intelligence will become pro-

found as such there will be fierce competition among the emerging technologies. The risk profiles of each individual will be done using highly responsive algorithms. Advance analytics will make the inclusiveness more transparent, profound and easy to implement. Hence, this work can further be enriched with knowledge and multi-dimension rule discovery algorithms that can help to identify more people that are on the edges of the society.

Appendices

```
Activities  Terminal  Tue 11:37
root@5fa6e6549296: /home/ns3/ns-3.30.1/src/fuzzylite/code
File Edit View Search Terminal Help
// Simar Preet Singh
// fuzzylite simulation using ns3

#include "fl/Headers.h"
#include "ns3/core-module.h"
#include "ns3/fuzzylite.h"

using namespace ns3;
using namespace fl;

int main (int argc, char *argv[])
{
    // fuzzylite code

    Engine* engine = new Engine;
    engine->setName("3-levelDesignSimulation");
    engine->setDescription("");

    InputVariable* TrafficLoad = new InputVariable;
    TrafficLoad->setName("TrafficLoad");
    TrafficLoad->setDescription("");
    TrafficLoad->setEnabled(true);
    TrafficLoad->setRange(0.000, 1.000);
    TrafficLoad->setLockValueInRange(false);
    TrafficLoad->addTerm(new Trapezoid("MinimumLoad", 0.00, 0.08,0.24,0.33));
    TrafficLoad->addTerm(new Trapezoid("AverageLoad", 0.24, 0.33, 0.57, 0.66));
    TrafficLoad->addTerm(new Gaussian("PeakLoad",0.500,0.200));
                                                    7,0-1         Top
```

Figure I: Coding (View 1) involved in Simulation Process

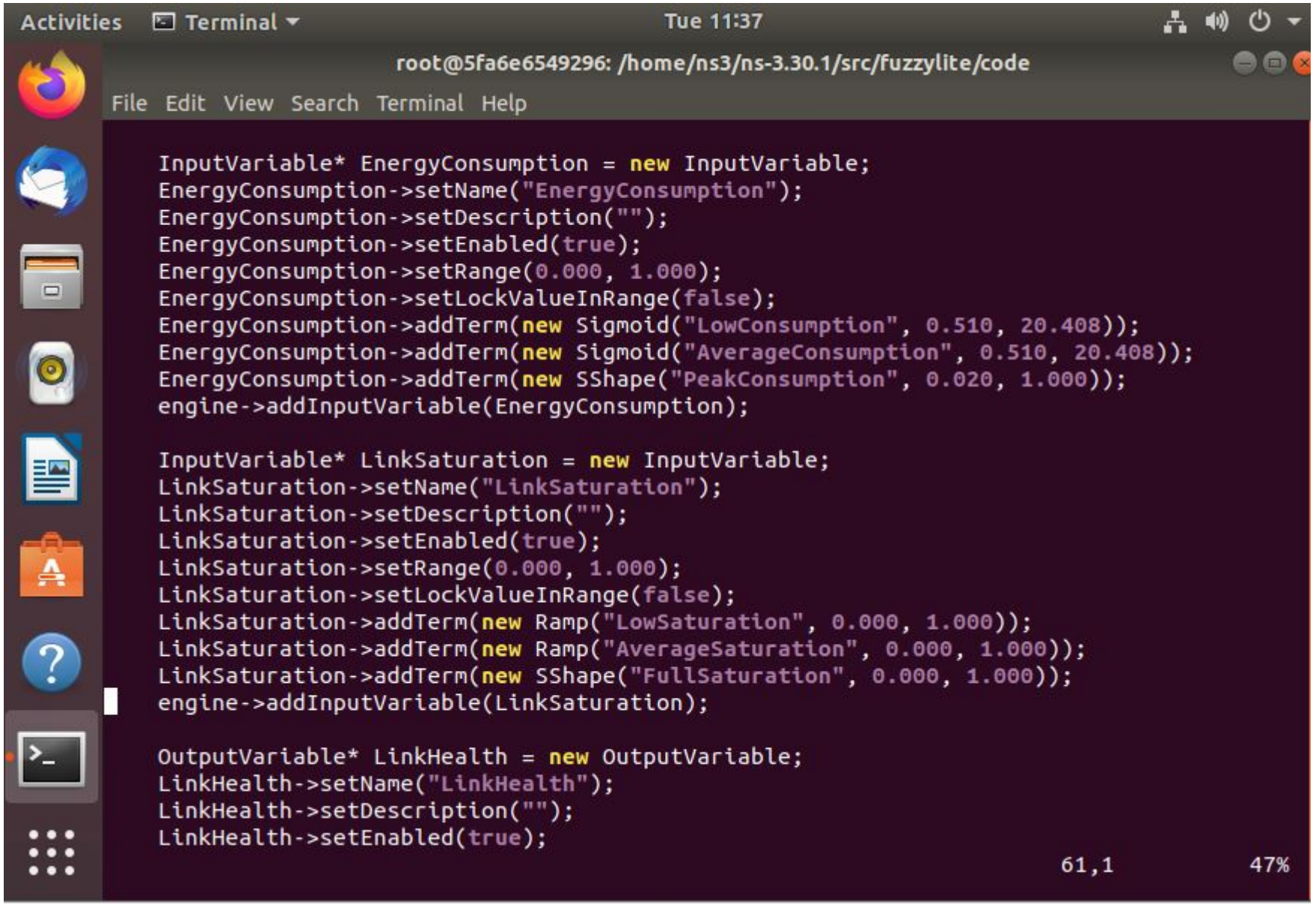


Figure II: Coding (View 2) involved in Simulation Process

Figure III: Results of Simulation Process

Bibliography

[1] S. P. Singh, A. Nayyar, R. Kumar, and A. Sharma, "Fog computing: from architecture to edge computing and big data processing," *The Journal of Supercomputing*, vol. 75, no. 4, pp. 2070–2105, 2019. [Online]. Available: https://link.springer.com/article/10.1007/s11227-018-2701-2

[2] A. Singh, "Healthriskdata-ver-1," *Mendeley Data, v2*, 2018. [Online]. Available: http://dx.doi.org/10.17632/8y7628b96z.1

[3] CISCO, "Cisco Fog Computing Solutions: Unleash the Power of the Internet of Things," p. 6, 2015. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-solutions.pdf

[4] M. Kumar, H. Kumar Verma, and G. Sikka, "A secure data transmission protocol for cloud-assisted edge-internet of things environment," *Transactions on Emerging Telecommunications Technologies*, p. e3883, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3883

[5] M. M. Hassan, J. Abawajy, M. Chen, M. Qiu, and S. Chen, "Special section on cloud-of-things and edge computing: Recent advances and future trends," 2019. [Online]. Available: https://eprints.soton.ac.uk/436568/

[6] I. Stojmenovic and S. Wen, "The Fog Computing Paradigm: Scenarios and Security Issues," Sep. 2014, pp. 1–8. [Online]. Available: https://fedcsis.org/proceedings/2014/drp/503.html

[7] E. Baccarelli, P. G. V. Naranjo, M. Scarpiniti, M. Shojafar, and J. H. Abawajy, "Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study," *IEEE access*, vol. 5, pp. 9882–9910, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7921687

[8] A. Dastjerdi, H. Gupta, R. Calheiros, S. Ghosh, and R. Buyya, "Fog Computing: principles, architectures, and applications," in *Internet of Things*. Elsevier, 2016, pp. 61–75. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/B9780128053959000046

[9] U. A. Deshmukh and S. A. More, "Fog computing: New approach in the world of cloud computing," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 9, pp. 16 310–16 316, 2016. [Online]. Available: http://itdl.org/Journal/Sep_16/Sep16.pdf#page=53

[10] M. Gohar, S. H. Ahmed, M. Khan, N. Guizani, A. Ahmed, and A. U. Rahman, "A big data analytics architecture for the internet of small things," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 128–133, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8291127

[11] E. M. Tordera, X. Masip-Bruin, J. García-Almiñana, A. Jukan, G.-J. Ren, and J. Zhu, "Do we all really know what a fog node is? current trends towards an open definition," *Computer Communications*, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0140366416307113?via=ihub

[12] M. R. Anawar, S. Wang, M. Azam Zia, A. K. Jadoon, U. Akram, and S. Raza, "Fog computing: An overview of big iot data analytics," *Wireless Communications and Mobile Computing*, vol. 2018, 2018. [Online]. Available: https://www.hindawi.com/journals/wcmc/2018/7157192/abs/

[13] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762118306349

[14] M. De Donno, K. Tange, and N. Dragoni, "Foundations and evolution of modern computing paradigms: Cloud, iot, edge, and fog," *IEEE Access*, vol. 7, pp. 150 936–150 948, 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8869772

[15] I. Jabri, T. Mekki, A. Rachedi, and M. B. Jemaa, "Vehicular fog gateways selection on the internet of vehicles: A fuzzy logic with ant colony optimization based approach," *Ad Hoc Networks*, vol. 91, p. 101879, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1570870518308096

[16] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments," *Neural Computing and Applications*, pp. 1–11, 2019. [Online]. Available: https://link.springer.com/article/10.1007/s00521-019-04119-7

[17] P. Bhargavi and S. Jyothi, "Object detection in fog computing using machine learning algorithms," in *Architecture and Security Issues in Fog Computing Applications*. IGI Global, 2020, pp. 90–107. [Online]. Available: https://www.igi-global.com/chapter/object-detection-in-fog-computing-using-machine-learning-algorithms/236443

[18] S. P. Singh, R. Kumar, and A. Sharma, "Efficient content retrieval in fog zone using nano-caches," *Concurrency and Computation: Practice and Experience*. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5438

[19] A. Mulyukin and I. Perl, "Adaptation of system dynamics model execution algorithms for cloud-based environment," in *2018 22nd Conference of Open Innovations Association (FRUCT)*. IEEE, 2018, pp. 179–189. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8468291

[20] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1728–1739, 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7439752

[21] L. N. Hersey and G. Schaefer, "New communication technology integration: Recommendations for public sector change," in *Returning to Interpersonal Dialogue and Understanding Human Communication in the Digital Age*. IGI Global, 2019, pp. 186–203. [Online]. Available: https://www.igi-global.com/chapter/new-communication-technology-integration/208233

[22] G. Zhao, M. A. Imran, Z. Pang, Z. Chen, and L. Li, "Toward real-time control in future wireless networks: communication-control co-design," *IEEE Communications Magazine*, vol. 57, no. 2, pp. 138–144, 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8558500

[23] T. Enokido, D. Doulikun, and M. Takizawa, "An energy-aware load balancing algorithm to perform computation type application processes in a cluster of servers," *International Journal of Web and Grid Services*, vol. 13, no. 2, p. 145, 2017. [Online]. Available: http://www.inderscience.com/link.php?id=10004125

[24] Z. Liu, J. Li, Y. Wang, X. Li, and S. Chen, "HGL: A hybrid global-local load balancing routing scheme for the Internet of Things through satellite networks," *International Journal of Distributed Sensor Networks*, vol. 13, no. 3, p. 155014771769258, Mar. 2017. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1550147717692586

[25] Cisco estimation report. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html#_Toc503317525

[26] F. Hussain and A. Alkarkhi, "Big data and fog computing," pp. 27–44, 03 2017. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-55405-1_3

[27] P. Munoz, R. Barco, I. de la Bandera, M. Toril, and S. Luna-Ramirez, "Optimization of a fuzzy logic controller for handover-based load balancing," in *Vehicular technology conference (VTC Spring), 2011 IEEE 73rd.* IEEE, 2011, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5956148/

[28] G. Salgueiro, M. D. Hanes, J. M. Clarke, and C. C. Byers, "Cognitive profiling and sharing of sensor data across iot networks," Nov. 1 2018, uS Patent App. 15/582,642. [Online]. Available: https://patents.google.com/patent/US20180316555A1/en

[29] S. Sethi, A. Sahu, and S. K. Jena, "Efficient load balancing in cloud computing using fuzzy logic," *IOSR Journal of Engineering*, vol. 2, no. 7, pp. 65–71, 2012. [Online]. Available: https://pdfs.semanticscholar.org/ff3d/93a49f075e0281e687d8bd3c7dcd26a7a3e3.pdf

[30] S. Mani, S. Yelamanchi, A. Musku, S. H. K. Masilamani, and D. P. Singh, "Secure and policy-driven computing for fog node applications," Nov. 1 2018, uS Patent App. 15/581,455. [Online]. Available: https://patents.google.com/patent/US20180316725A1/en

[31] Y. Jiang, Z. Huang, and D. H. Tsang, "Challenges and solutions in fog computing orchestration," *IEEE Network*, vol. 32, no. 3, pp. 122–129, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8121864/

[32] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of internet of things," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 46–59, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7286781/

[33] J. Fu, Y. Liu, H.-C. Chao, B. Bhargava, and Z. Zhang, "Secure data storage and searching for industrial iot by integrating fog computing and cloud computing," *IEEE Transactions on Industrial Informatics*, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8259028/

[34] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of everything*. Springer, 2018, pp. 103–130. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-10-5861-5_5

[35] Y.-S. Chen, C.-S. Hsu, and Y.-G. Siao, "Linear regression-based delay-bounded routing protocols for vanets," *Wireless Communications and Mobile Computing*, vol. 14, no. 2, pp. 186–199, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/wcm.1243

[36] P.-H. Kuo, A. Mourad, C. Lu, M. Berg, S. Duquennoy, Y.-Y. Chen, Y.-H. Hsu, A. Zabala, R. Ferrari, S. Gonzalez *et al.*, "An integrated edge and fog system for future communication networks," in *Wireless Communications and Networking Conference Workshops (WCNCW), 2018 IEEE*. IEEE, 2018, pp. 338–343. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8369023/

[37] P. G. V. Naranjo, E. Baccarelli, and M. Scarpiniti, "Design and energy-efficient resource management of virtualized networked fog architectures for the real-time support of iot applications," *The Journal of Supercomputing*, vol. 74, no. 6, pp. 2470–2507, 2018. [Online]. Available: https://link.springer.com/article/10.1007/s11227-018-2274-0

[38] A. K. Moghe and D. A. Maluf, "Interfacing a vehicle with a fog device during fueling," Nov. 1 2018, uS Patent App. 15/498,095. [Online]. Available: https://patents.google.com/patent/US20180315258A1/en

[39] G. L. Stavrinides and H. D. Karatza, "Scheduling different types of applications in a saas cloud." [Online]. Available: https://www.researchgate.net/profile/Georgios_Stavrinides/publication/300301927_Scheduling_Different_Types_of_Applications_in_a_SaaS_Cloud/links/57c2a02b08aeda1ec38f4cfa/Scheduling-Different-Types-of-Applications-in-a-SaaS-Cloud.pdf

[40] I. A. Moschakis and H. D. Karatza, "A meta-heuristic optimization approach to the scheduling of bag-of-tasks applications on heterogeneous clouds with multi-level arrivals and critical jobs," *Simulation Modelling Practice and Theory*, vol. 57, pp. 1–25, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X15000787

[41] I. A. Moschakis and H. D. Karatza, "Towards scheduling for internet-of-things applications on clouds: a simulated annealing approach," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 8, pp. 1886–1899, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.3105

[42] S. Kitanov and T. Janevski, "State of the art: Fog computing for 5g networks," in *24th Telecommunications Forum (TELFOR), 2016*. IEEE, 2016, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7818728/

[43] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5g systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. pp. 131–139, Feb. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6736753/

[44] A. González, C. Aliagas, and M. Valero, "A data cache with multiple caching strategies tuned to different types of locality." *ICS '95 Proceedings of the 9th International Conference on Supercomputing, Barcelona, Spain,*, pp. pp. 338–347, July, 1995. [Online]. Available: https://dl.acm.org/citation.cfm?id=224622

[45] X. Chen, L.-W. Chang, C. I. Rodrigues, J. Lv, Z. Wang, and W.-M. Hwu, "Adaptive Cache Management for Energy-Efficient GPU Computing," in *Proceedings of the 47th annual IEEE/ACM international symposium on microarchitecture*. IEEE, Dec. 2014, pp. 343–355. [Online]. Available: http://ieeexplore.ieee.org/document/7011400/

[46] M. Eckert, D. Meyer, B. Klauer, and J. Haase, "Comparison and evaluation of cache parameters for softcores on FPGAs," in *2017 International Conference on FPGA Reconfiguration for General-Purpose Computing (FPGA4GPC)*. IEEE, May 2017, pp. 19–24. [Online]. Available: http://ieeexplore.ieee.org/document/8008961/

[47] B. Liu, H. Zhang, H. Ji, X. Li, and K. Wang, "High quality guarantee for video streaming in massive MIMO relay networks with caching." IEEE, Sep. 2016, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/7794944/

[48] W. Chen, Z. Gao, X. Shao, Y. Gong, X. Xu, and L. Han, "A Novel Cache Replacement Algorithm of EPCIS," in *International Conference on Logistics Engineering, Management and Computer Science (LEMCS 2015)*. Atlantis Press, 2015. [Online]. Available: http://www.atlantis-press.com/php/paper-details.php?id=25838308

[49] X. Sun and Z. Wang, "An Optimized Cache Replacement Algorithm for Information-centric Networks," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE, Dec. 2015, pp. 683–688. [Online]. Available: http://ieeexplore.ieee.org/document/7463802/

[50] N. Kumar and S. Zeadally, "QoS-Aware Hierarchical Web Caching Scheme for Online Video Streaming Applications in Internet-Based Vehicular Ad Hoc Networks," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. pp. 7892–7900, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7091894

[51] B. Liu, H. Zhang, H. Ji, and X. Li, "A novel joint transmission and caching optimizing scheme in multirelay networks: Video service quality assurance scheme," *International Journal of Communication Systems*, vol. 30, no. 14, p. e3284, Sep. 2017. [Online]. Available: http://doi.wiley.com/10.1002/dac.3284

[52] D. Kim, S.-W. Lee, Y.-B. Ko, and J.-H. Kim, "Cache capacity-aware content centric networking under flash crowds," *Journal of Network and Computer Applications*, vol. 50, pp. pp. 101–113, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804514001416

[53] C. Amza, G. Soundararajan, and E. Cecchet, "Transparent caching with strong consistency in dynamic content web sites," in *Proceedings of the 19th Annual International Conference on Supercomputing*, ser. ICS '05. New York, NY, USA: ACM, 2005, pp. 264–273. [Online]. Available: http://doi.acm.org/10.1145/1088149.1088185

[54] M. Usman, M. R. Asghar, I. S. Ansari, F. Granelli, Q. H. Abbasi, and K. Qaraqe, "A marketplace for efficient and secure caching for iot applications in 5g networks," in *Wireless Communications and Networking Conference (WCNC)*. IEEE, 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8377254

[55] B. Varghese, N. Wang, D. S. Nikolopoulos, and R. Buyya, "Feasibility of fog computing," *arXiv preprint arXiv:1701.05451*, 2017. [Online]. Available: https://arxiv.org/abs/1701.05451

[56] S. Ningning, G. Chao, A. Xingshuo, and Z. Qiang, "Fog computing dynamic load balancing mechanism based on graph repartitioning," *China Communications*, vol. 13, no. 3, pp. 156–164, 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7445510

[57] Y.-C. Tsao, V.-V. Thanh, and J.-C. Lu, "Multiobjective robust fuzzy stochastic approach for sustainable smart grid design," *Energy*, vol. 176, pp. 929–939, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0360544219306711

[58] N. Apthorpe, D. Y. Huang, D. Reisman, A. Narayanan, and N. Feamster, "Keeping the smart home private with smart (er) iot traffic shaping," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 3, pp. 128–148, 2019. [Online]. Available: https://content.sciendo.com/view/journals/popets/2019/3/article-p128.xml

[59] R. A. Rashid, L. Chin, M. Sarijari, R. Sudirman, and T. Ide, "Machine learning for smart energy monitoring of home appliances using iot," in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2019, pp. 66–71. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8806026

[60] J. Wu, M. Dong, K. Ota, J. Li, and Z. Guan, "Fcss: Fog computing based content-aware filtering for security services in information centric social networks," *IEEE Transactions on Emerging Topics in computing*, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8036242

[61] M. A. Al Faruque and K. Vatanparvar, "Energy management-as-a-service over fog computing platform," *IEEE internet of things journal*, vol. 3, no. 2, pp. 161–169, 2015. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7217791

[62] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7498684

[63] M. Gorlatova, H. Inaltekin, and M. Chiang, "Characterizing task completion latencies in fog computing," *arXiv preprint arXiv:1811.02638*, 2018. [Online]. Available: https://arxiv.org/abs/1811.02638

[64] R. Mahmud, K. Ramamohanarao, and R. Buyya, "Latency-aware application module management for fog computing environments," *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 1, p. 9, 2019. [Online]. Available: https://dl.acm.org/citation.cfm?id=3186592

[65] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 workshop on mobile big data*. ACM, 2015, pp. 37–42, 00437. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2757384.2757397

[66] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*. IEEE, doi: 10.1109/HotWeb.2015.22, pp. 73–78, Nov 12-13, 2015. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7372286/

[67] A. Rayes and S. Salam, "Fog computing," in *Internet of Things From Hype to Reality*. Springer, doi: 10.1007/978-3-319-44860-2, 2017, pp. 139–164. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-99516-8_6

[68] P. Kukreja and D. D. Sharma, "A detail review on cloud, fog and dew computing," *International Journal of Science, Engineering and Technology Research (IJSETR)*, ISSN: 2278-7798, vol. 5, no. 5, pp. 1412-1420, May 2016. [Online]. Available: http://ijsetr.org/wp-content/uploads/2016/05/IJSETR-VOL-5-ISSUE-5-1412-1420.pdf

[69] P. More, "Review of implementing fog computing," *International Journal of Research in Engineering and Technology*, ISSN: 2319-1163, vol. 4, no. 6, pp. 335-338, June 2015. [Online]. Available: http://www.academia.edu/download/41935041/REVIEW_OF_IMPLEMENTING_FOG_COMPUTING.pdf

[70] K. P.Saharan and A. Kumar, "Fog in Comparison to Cloud: A Survey," *International Journal of Computer Applications*, vol. 122, no. 3, pp. 10–12, Jul. 2015. [Online]. Available: http://research.ijcaonline.org/volume122/number3/pxc3904773.pdf

[71] M. Katyal and A. Mishra, "A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment," vol. 1, no. 2, p. 10, 2013. [Online]. Available: https://arxiv.org/pdf/1403.6918

[72] M. Verma, N. Bhardawaj, and A. K. Yadav, "An architecture for Load Balancing Techniques for Fog Computing Environment," *International Journal of Computer Science & Communication (IJCSC)*, vol. 6, no. 2, pp. 269–274, 2015. [Online]. Available: https://pdfs.semanticscholar.org/24e5/022d0b6775f460159dd1a96d63ec4fbb4f68.pdf

[73] Y. Lin and H. Shen, "Cloud Fog: Towards High Quality of Experience in Cloud Gaming." IEEE, Sep. 2015, pp. 500–509. [Online]. Available: http://ieeexplore.ieee.org/document/7349605/

[74] S. Verma, A. K. Yadav, D. Motwani, R. Raw, and H. K. Singh, "An efficient data replication and load balancing technique for fog computing environment," in *3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, March 16-18, 2016, pp. 2888–2895. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7724792/

[75] M. Verma, N. Bhardwaj, and A. K. Yadav, "Real time efficient scheduling algorithm for load balancing in fog computing environment," *International Journal of Information Technology and Computer Science (IJITCS), doi: 10.5815/ijitcs.2016.04.01*, vol. 8, no. 4, pp. 1–10, 2016. [Online]. Available: http://mecs-press.org/ijitcs/ijitcs-v8-n4/IJITCS-V8-N4-1.pdf

[76] C. A. Brennand, F. D. da Cunha, G. Maia, E. Cerqueira, A. A. Loureiro, and L. A. Villas, "Fox: A traffic management system of computer-based vehicles fog," in *IEEE Symposium on Computers and Communication (ISCC)*. IEEE, June 27-30, 2016, pp. 982–987. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7543864/

[77] A. Paya and D. Marinescu, "Energy-aware load balancing and application scaling for the cloud ecosystem," *IEEE Transactions on Cloud Computing*, doi: 10.1109/TCC.2015.2396059, vol. 5, no. 1, pp. 15-27, Jan-March 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7018917

[78] M. Kalra and S. Singh, "A review of metaheuristic scheduling techniques in cloud computing," *Egyptian informatics journal*, vol. 16, no. 3, pp. 275–295, Nov. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866515000353?via=ihub

[79] Vijindra, S. Shenai *et al.*, "Survey on scheduling issues in cloud computing," *Procedia Engineering*, Elsevier, vol. 38, pp. 2881-2888, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705812022503?via=ihub

[80] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers: ENERGY AND PERFORMANCE EFFICIENT DYNAMIC CONSOLIDATION OF VIRTUAL MACHINES," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012. [Online]. Available: http://doi.wiley.com/10.1002/cpe.1867

[81] C. Vijaya and D. Srinivasan, "A survey on resource scheduling in cloud computing," *Int. J. Pharm. Technol*, vol. 8, no. 4, pp. 26 142–26 162, 2016. [Online]. Available: http://www.ijptonline.com/wp-content/uploads/2017/01/26142-26162.pdf

[82] Y. Sun and N. Zhang, "A resource-sharing model based on a repeated game in fog computing," *Saudi Journal of Biological Sciences*, pp. 687–694, Elsevier, http://dx.doi.org/10.1016/j.sjbs.2017.01.043, pp. 687-694, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1319562X17300529

[83] S. Agarwal, S. Yadav, and A. K. Yadav, "An architecture for elastic resource allocation in fog computing," *International Journal of Computer Science and Communication*, vol. 6, no. 2, pp. 201–207, 2015. [Online]. Available: http://csjournals.com/IJCSC/PDF6-2/31.%20Swati.pdf

[84] A.-p. Xiong and C.-x. Xu, "Energy Efficient Multiresource Allocation of Virtual Machine Based on PSO in Cloud Data Center," *Mathematical Problems in Engineering*, vol. 2014, pp. 1–8, 2014. [Online]. Available: http://www.hindawi.com/journals/mpe/2014/816518/

[85] A. Saraswathi, Y. Kalaashri, and S. Padmavathi, "Dynamic Resource Allocation Scheme in Cloud Computing," *Procedia Computer Science*, vol. 47, pp. 30–36, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877050915004482

[86] P. G. Jeba Leelipushpam and J. Sharmila, "Live vm migration techniques in cloud environment – a survey." IEEE, Apr. 2013, pp. 408–413. [Online]. Available: http://ieeexplore.ieee.org/document/6558130/

[87] A. Gupta and M. K. Mukhija, "A resourceful technique for virtual machine migration in fog computing," *International Journal of Emerging Technology and Advanced Engineering*, ISSN: 2250-2459, ISO 9001:2008 Certified Journal, vol. 6, no. 6, pp. 167-170, 2016. [Online]. Available: http://www.ijetae.com/files/Volume6Issue6/IJETAE_0616_25.pdf

[88] K. Wang, M. Shen, J. Cho, A. Banerjee, J. Van der Merwe, and K. Webb, "Mobiscud: A fast moving personal cloud in the mobile network," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. ACM, August 17-21, 2015, London, UK, pp. 19–24, http://dx.doi.org/10.1145/2 785 971.2 785 979. [Online]. Available: http://www.cs.utah.edu/~jmanbal/paper/mobiscud.pdf

[89] L. F. Bittencourt, M. M. Lopes, I. Petri, and O. F. Rana, "Towards virtual machine migration in fog computing," in *10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*. IEEE, 4-6 Nov. 2015, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/document/7424534/

[90] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu, "Live migration of virtual machine based on full system trace and replay," in *Proceedings of the 18th ACM international symposium on High performance distributed computing*. ACM, 2009, pp. 101–110. [Online]. Available: https://dl.acm.org/doi/10.1145/1551609.1551630

[91] K. Sabiri, F. Benabbou, H. Mustapha, H. Moutachaouik, and K. Akodadi, "A survey of cloud migration methods: A comparison and proposition," *International Journal of Advanced Computer Science & Applications*, vol. 1, no. 7, pp. 598–604, 2016. [Online]. Available: https://thesai.org/Publications/ViewPaper?Volume=7&Issue=5&Code=ijacsa&SerialNo=79

[92] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 2005, pp. 273–286. [Online]. Available: https://www.usenix.org/legacy/event/nsdi05/tech/full_papers/clark/clark_html/

[93] Y. Zhou, W. Shi, and F. Song, "A smart collaborative policy for mobile fog computing in rural vitalization," *Wireless Communications and Mobile Computing*, vol. 2018, 2018. [Online]. Available: https://doi.org/10.1155/2018/2643653

[94] G. Li, H. Zhou, B. Feng, G. Li, T. Li, Q. Xu, and W. Quan, "Fuzzy theory based security service chaining for sustainable mobile-edge computing," *Mobile Information Systems*, vol. 2017, 2017. [Online]. Available: https://doi.org/10.1155/2017/8098394

[95] M. Wazid, A. K. Das, N. Kumar, and A. V. Vasilakos, "Design of secure key management and user authentication scheme for fog computing services," *Future Generation Computer Systems*, vol. 91, pp. 475–492, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18303959

[96] A. Z. Nasrollahi and A. A. P. Kazem, "Resource discovery in grid computing using fuzzy logic and tabu table," *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 16, no. 9, p. 61, Sep. 2016. [Online]. Available: http://paper.ijcsns.org/07_book/201609/20160910.pdf

[97] S. A. Soleymani, A. H. Abdullah, M. Zareei, M. H. Anisi, C. Vargas-Rosales, M. K. Khan, and S. Goudarzi, "A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing," *IEEE Access*, vol. 5, pp. 15 619–15 629, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7995031

[98] F. H. Rahman, T.-W. Au, S. S. Newaz, W. S. Suhaili, and G. M. Lee, "Find my trustworthy fogs: A fuzzy-based trust evaluation framework," *Future Generation Computer Systems*, Jun. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17322367

[99] S. Bitam, S. Zeadally, and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm," *Enterprise Information Systems*, vol. vol. 12,, no. 4, pp. pp. 373–397, Apr. 2018. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/17517575.2017.1304579

[100] Y. Shu and F. Zhu, "An edge computing offloading mechanism for mobile peer sensing and network load weak balancing in 5g network," *Journal of Ambient Intelligence and Humanized Computing*, Aug. 2018. [Online]. Available: http://link.springer.com/10.1007/s12652-018-0970-5

[101] M. B. Kamal, N. Javaid, S. A. A. Naqvi, H. Butt, T. Saif, and M. D. Kamal, "Heuristic Min-conflicts Optimizing Technique for Load Balancing on Fog Computing," in *Advances in Intelligent Networking and Collaborative Systems*, F. Xhafa, L. Barolli, and M. Greguš, Eds. Cham: Springer International Publishing, 2019, vol. 23, pp. 207–219. [Online]. Available: http://link.springer.com/10.1007/978-3-319-98557-2_19

[102] F.-H. Tseng, M.-S. Tsai, C.-W. Tseng, Y.-T. Yang, C.-C. Liu, and L.-D. Chou, "A Lightweight Autoscaling Mechanism for Fog Computing in Industrial Applications," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4529–4537, Oct. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8272512/

[103] X. Xu, S. Fu, Q. Cai, W. Tian, W. Liu, W. Dou, X. Sun, and A. X. Liu, "Dynamic Resource Allocation for Load Balancing in Fog Environment," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–15, 2018. [Online]. Available: https://www.hindawi.com/journals/wcmc/2018/6421607/

[104] J. Oueis, E. C. Strinati, and S. Barbarossa, "The fog balancing: Load distribution for small cell cloud computing," in *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*. IEEE, 2015, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7146129/

[105] S. Singh and R. K. Jha, "A survey on software defined networking: Architecture for next generation network," *Journal of Network and Systems Management*, vol. 25, no. 2, pp. 321–374, 2017. [Online]. Available: https://link.springer.com/article/10.1007/s10922-016-9393-9

[106] D. Chen and V. Kuehn, "Adaptive radio unit selection and load balancing in the downlink of fog radio access network," in *Global Communications Conference (GLOBECOM), 2016 IEEE*. IEEE, 2016, pp. 1–wwww–7. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7841568/

[107] F. Al-Turjman, "Cognitive caching for the future sensors in fog networking," *Pervasive and Mobile Computing*, vol. vol. 42, pp. pp. 317–334, Dec. 2017. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1574119216303728

[108] S. M. Azimi, O. Simeone, A. Sengupta, and R. Tandon, "Online Edge Caching and Wireless Delivery in Fog-Aided Networks with Dynamic Content Popularity," *arXiv:1711.10430 [cs, math]*, Nov. 2017, arXiv: 1711.10430. [Online]. Available: http://arxiv.org/abs/1711.10430

[109] M. Zhanikeev, "Fog Cloud Caching at Network Edge via Local Hardware Awareness Spaces," in *2016 IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. Nara, Japan: IEEE, Jun. 2016, pp. 184–188. [Online]. Available: http://ieeexplore.ieee.org/document/7756228/

[110] M. Enguehard, G. Carofiglio, and D. Rossi, "A popularity-based approach for effective cloud offload in fog clusters," *30th International Teletraffic Congress (ITC 30),*, vol. vol. 1, pp. pp. 55–63, 2018. [Online]. Available: https://perso.telecom-paristech.fr/drossi/paper/rossi18itc.pdf

[111] F. Song, Z.-Y. Ai, J.-J. Li, G. Pau, M. Collotta, I. You, and H.-K. Zhang, "Smart Collaborative Caching for Information-Centric IoT in Fog Computing," *Sensors*, vol. 17, no. 11, p. 2512, Nov. 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/11/2512

[112] Y. Hua, L. Guan, and K. Kyriakopoulos, "Semi-Edge: From Edge Caching to Hierarchical Caching in Network Fog," in *Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking - EdgeSys'18*. Munich, Germany: ACM Press, 2018, pp. 43–48. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3213344.3213352

[113] F. Malandrino, C. Chiasserini, and S. Kirkpatrick, "The price of fog: a data-driven study on caching architectures in vehicular networks," in *Proceedings of the First International Workshop on Internet of Vehicles and Vehicles of Internet - IoV-VoI '16*. Paderborn, Germany: ACM Press, 2016, pp. 37–42. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2938681.2938682

[114] H. Zipper, M. Meier, E. Hintze, and C. Diedrich, "Implementing state machines in distributed event-based systems," in *3rd International Conference on Event-Based Control, Communication and Signal Processing (EBCCSP)*. IEEE, 2017, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/document/8022809

[115] X. Song, Y. Yang, and X. Li, "Simd-based multiple sets intersection with dual-scale search algorithm," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 2311–2314. [Online]. Available: https://dl.acm.org/citation.cfm?id=3133082

[116] M. Z. Chowdhury, T. Fujii, G.-M. Muntean, J.-W. Choi, and G. Araniti, "Emerging small cell wireless technologies for 5g: Architectures and applications," *Wireless Communications and Mobile Computing*, vol. vol. 2018,, 2018. [Online]. Available: https://www.hindawi.com/journals/wcmc/2018/8969264/

[117] P. Liu, X. Han, Z. Liu, and Z. Ji, "A research on unified storage management and access technology applied in power network dispatch and control big data," in *3rd Information Technology and Mechatronics Engineering Conference (ITOEC)*. IEEE, 2017, pp. 1035–1039. [Online]. Available: https://ieeexplore.ieee.org/document/8122511

[118] P. Rosenberger, M. Holder, M. Zirulnik, and H. Winner, "Analysis of real world sensor behavior for rising fidelity of physically based lidar sensor models," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 611–616. [Online]. Available: https://ieeexplore.ieee.org/document/8500511

[119] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *arXiv preprint arXiv:1803.04311*, 2018. [Online]. Available: https://arxiv.org/abs/1803.04311

[120] K. Hwang, J. Dongarra, and G. C. Fox, *Distributed and cloud computing: from parallel processing to the internet of things*. Morgan Kaufmann, 2013. [Online]. Available: https://www.elsevier.com/books/distributed-and-cloud-computing/hwang/978-0-12-385880-1

[121] K. Xia, L. Gao, W. Li, and K.-M. Chao, "Disassembly sequence planning using a simplified teaching-learning-based optimization algorithm," in *Sustainable Manufacturing and Remanufacturing Management*. Springer, 2019, pp. 319–343. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-73488-0_13

[122] J. Ekanayake and G. Fox, "High performance parallel computing with clouds and cloud technologies," in *International Conference on Cloud Computing*. Springer, 2009, pp. 20–38. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-12636-9_2

[123] D. O. Mau, T. Taleb, and M. Chen, "MM3c: Multi-Source Mobile Streaming in Cache-enabled Content-Centric Networks," *IEEE Global Communications Conference (GLOBECOM)*, pp. pp. 1–6, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7417045

[124] Junaid Ahmed Khan, Cedric Westphal, and Yacine Ghamri-Doudane, "Offloading Content with Self-organizing Mobile Fogs," 2017, arXiv:1707.06285v1 [cs.NI]. [Online]. Available: https://arxiv.org/abs/1707.06285

[125] G.-q. Wang, T. Huang, J. Liu, R.-c. Xie, and Y.-j. Liu, "In-network caching for energy efficiency in content-centric networking," *The Journal of China Universities of Posts and Telecommunications*, vol. 21, no. 4, pp. pp. 25–31, Aug. 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1005888514603125

[126] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas, "Distributed Cache Management in Information-Centric Networks," *IEEE Transactions on Network and Service Management*, vol. 10, no. 3, pp. pp. 286–299, Sep. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6522566/

[127] A. A. A. Mohammed and K. Okamura, "Distributed GA for popularity based partial cache management in ICN." ACM Press, 2014, pp. 1–2. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2619287.2619305

[128] F.-C. Chang and H.-C. Huang, "A refactoring method for cache-efficient swarm intelligence algorithms," *Information Sciences*, vol. 192, pp. pp. 39–49, Jun. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0020025510001039

[129] S. Garg, A. Singh, S. Batra, N. Kumar, and L. T. Yang, "Uav-empowered edge computing environment for cyber-threat detection in smart vehicles," *IEEE Network*, vol. vol. 32, no. 3, pp. pp. 42–51, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8370877

[130] D. Rahbari and M. Nickray, "Task offloading in mobile fog computing by classification and regression tree," *Peer-to-Peer Networking and Applications*, pp. pp. 1–19, 2019. [Online]. Available: https://link.springer.com/article/10.1007/s12083-019-00721-7

[131] X. Lin, J. Xia, and Z. Wang, "Probabilistic caching placement in uav-assisted heterogeneous wireless networks," *Physical Communication*, vol. vol. 33, Apr. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1874490718304476

[132] J. Ma, J. Wang, G. Liu, and P. Fan, "Low latency caching placement policy for cloud-based vanet with both vehicle caches and rsu caches," in *Globecom Workshops (GC Wkshps)*. IEEE, 2017, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8269203

[133] S. Schlag, C. Schulz, D. Seemaier, and D. Strash, "Scalable Edge Partitioning," pp. arXiv:1808.06 411v2 [cs.DS], pp. 1–15, 2019. [Online]. Available: https://arxiv.org/abs/1808.06411

[134] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Communications Surveys & Tutorials*, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8556457

[135] J. Zhan, Y. Zhang, W. Jiang, J. Yang, L. Li, and Y. Li, "Energy-aware page replacement and consistency guarantee for hybrid nvm–dram memory systems," *Journal of Systems Architecture*, vol. vol. 89, pp. pp. 60–72, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762118300596

[136] Y. Huang, Y. Zhu, X. Fan, X. Ma, F. Wang, J. Liu, Z. Wang, and Y. Cui, "Task scheduling with optimized transmission time in collaborative cloud-edge learning," in *27th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2018, pp. 1–9. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8487352

[137] D. Bhamare, A. Erbad, R. Jain, M. Zolanvari, and M. Samaka, "Efficient virtual network function placement strategies for cloud radio access networks," *Computer Communications*, vol. 127, pp. pp. 50–60, Sep. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366418301142

[138] M. Scarpiniti, E. Baccarelli, P. G. V. Naranjo, and A. Uncini, "Energy performance of heuristics and meta-heuristics for real-time joint resource scaling and consolidation in virtualized networked data centers," *The Journal of Supercomputing*, vol. vol. 74, no. 5, pp. pp. 2161–2198, 2018. [Online]. Available: https://link.springer.com/article/10.1007/s11227-018-2244-6

[139] H. Wu, S. Deng, W. Li, S. U. Khan, J. Yin, and A. Y. Zomaya, "Request dispatching for minimizing service response time in edge cloud systems," in *27th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2018, pp. 1–9. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8487354

[140] D. Joshi, M. Mittal, and M. Kumar, "A teaching–learning-based optimization algorithm for the resource-constrained project scheduling problem," in *Harmony Search and Nature Inspired Optimization Algorithms*. Springer, 2019, pp. 1101–1109. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-13-0761-4_103

[141] E. V. Altay and B. Alatas, "Performance comparisons of socially inspired metaheuristic algorithms on unconstrained global optimization," in *Advances in Computer Communication and Computational Sciences*. Springer, 2019, pp. 163–175. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-13-0341-8_15

[142] J. Nayak, B. Naik, G. Chandrasekhar, and H. Behera, "A survey on teaching–learning-based optimization algorithm: Short journey from 2011 to 2017," in *Computational Intelligence in Data Mining*. Springer, 2019, pp. 739–758. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-10-8055-5_66

[143] J. Kaur, S. S. Chauhan, and P. Singh, "Nsgltlbole: A modified non-dominated sorting tlbo technique using group learning and learning experience of others for multi-objective test problems," in *Soft Computing and Signal Processing*. Springer, Jan. 2019, pp. 243–251. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-13-3600-3_23

[144] M. K. Hussein and M. A. Mohammed, "Efficient and accuracy of retrieval files in e-learning system based on click method," in *1st International Scientific Conference of Engineering Sciences-3rd Scientific Conference of Engineering Science (ISCES)*. IEEE, 2018, pp. 34–38. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8340524

[145] D. Chen, F. Zou, J. Wang, and W. Yuan, "Samcctlbo: a multi-class cooperative teaching–learning-based optimization algorithm with simulated annealing," *Soft Computing*, vol. vol. 20, no. 5, pp. pp. 1921–1943, 2016. [Online]. Available: https://link.springer.com/article/10.1007/s00500-015-1613-9

[146] R. Bellio, S. Ceschia, L. Di Gaspero, A. Schaerf, and T. Urli, "Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem," *Computers & Operations Research*, vol. vol. 65, pp. pp. 83–92, Jan. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054815001690

[147] Q. Zhang, G. Yu, and H. Song, "A hybrid bird mating optimizer algorithm with teaching-learning-based optimization for global numerical optimization," *Statistics, Optimization & Information Computing*, vol. vol. 3, no. 1, pp. pp. 54–65, 2015. [Online]. Available: http://iapress.org/index.php/soic/article/download/20150305/183

[148] R. V. Rao and V. Kalyankar, "Multi-pass turning process parameter optimization using teaching–learning-based optimization algorithm," *Scientia Iranica*, vol. vol. 20, no. 3, pp. pp. 967–974, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1026309813000035

[149] B. Amiri, "Application of teaching-learning-based optimization algorithm on cluster analysis," *Journal of Basic and Applied Scientific Research*, vol. vol. 2, no. 11, pp. pp. 11 795–11 802, 2012. [Online]. Available: https://pdfs.semanticscholar.org/c7b6/f461cec909f07b4f95f8a1f8f45b66e05171.pdf

[150] T. Goyal, A. Singh, and A. Agrawal, "Cloudsim: Simulator for cloud computing infrastructure and modeling," *Procedia Engineering*, vol. 38, pp. 3566–3572, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705812023259?via=ihub

[151] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments," *arXiv preprint arXiv:1606.02007*, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2509

[152] R. Mayer, L. Graser, H. Gupta, E. Saurez, and U. Ramachandran, "EmuFog: Extensible and Scalable Emulation of Large-Scale Fog Computing Infrastructures," p. 6. [Online]. Available: https://ieeexplore.ieee.org/document/8368525

[153] C. Cicconetti, M. Conti, and A. Passarella, "An architectural framework for serverless edge computing: design and emulation tools," in *2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2018, pp. 48–55. [Online]. Available: https://ieeexplore.ieee.org/document/8590993

[154] J. Hasenburg, M. Grambow, E. Grünewald, S. Huk, and D. Bermbach, "Mockfog: Emulating fog computing infrastructure in the cloud," in *2019 IEEE International Conference on Fog Computing (ICFC)*. IEEE, 2019, pp. 144–152. [Online]. Available: https://ieeexplore.ieee.org/document/8821958

[155] I. Lera, C. Guerrero, and C. Juiz, "Yafs: A simulator for iot scenarios in fog computing," *IEEE Access*, vol. 7, pp. 91 745–91 758, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8758823

[156] Y. Zeng, M. Chao, and R. Stoleru, "Emuedge: A hybrid emulator for reproducible and realistic edge computing experiments," in *2019 IEEE International Conference on Fog Computing (ICFC)*. IEEE, 2019, pp. 153–164. [Online]. Available: https://ieeexplore.ieee.org/document/8822062

[157] M. Ashouri, F. Lorig, P. Davidsson, and R. Spalazzese, "Edge computing simulators for iot system design: An analysis of qualities and metrics," *Future Internet*, vol. 11, no. 11, p. 235, 2019. [Online]. Available: https://res.mdpi.com/d_attachment/futureinternet/futureinternet-11-00235/article_deploy/futureinternet-11-00235.pdf

[158] T. Qayyum, A. W. Malik, M. A. K. Khattak, O. Khalid, and S. U. Khan, "Fognetsim++: A toolkit for modeling and simulation of distributed fog environment," *IEEE Access*, vol. 6, pp. 63 570–63 583, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8502760

[159] M. H. Miraz and M. Ali, "Applications of blockchain technology beyond cryptocurrency," *arXiv preprint arXiv:1801.03528*, 2018. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1801/1801.03528.pdf

[160] P. Mader, "Contesting financial inclusion," *Development and Change*, vol. 49, no. 2, pp. 461–483, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1111/dech.12368

[161] S. Chai, Y. Chen, B. Huang, and D. Ye, "Social networks and informal financial inclusion in the people's republic of china," 2018. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3140099

[162] J. Jagtiani and C. Lemieux, "The roles of alternative data and machine learning in fintech lending: evidence from the lendingclub consumer platform," 2018. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3178461

[163] R. Agrawal, "Measures adopted for promoting inclusive and sustainable growth: An empirical study of india," *Paradigm*, vol. 22, no. 2, pp. 143–159, 2018. [Online]. Available: https://journals.sagepub.com/doi/abs/10.1177/0971890718787904

[164] A. D. Rothenberg, A. Gaduh, N. E. Burger, C. Chazali, I. Tjandraningsih, R. Radikun, C. Sutera, and S. Weilant, "Rethinking indonesia's informal sector," *World Development*, vol. 80, pp. 96–113, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305750X15300978

[165] C. W. Larsson and J. Svensson, "Mobile phones in the transformation of the informal economy: stories from market women in kampala, uganda," *Journal of Eastern African Studies*, vol. 12, no. 3, pp. 533–551, 2018. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/17531055.2018.1436247

[166] J. O. Ejemeyovwi and E. S. Osabuohien, "Investigating the relevance of mobile technology adoption on inclusive growth in west africa," *Contemporary Social Science*, pp. 1–14, 2018. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/21582041.2018.1503320

[167] A. A. Kemal, "Mobile banking in the government-to-person payment sector for financial inclusion in pakistan," *Information Technology for Development*, pp. 1–28, 2018. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/02681102.2017.1422105

[168] C. C. Williams, A. Martinez-Perez, and A. M. Kedir, "Informal entrepreneurship in developing economies: The impacts of starting up unregistered on firm performance," *Entrepreneurship Theory and Practice*, vol. 41, no. 5, pp. 773–799, 2017. [Online]. Available: https://journals.sagepub.com/doi/abs/10.1111/etap.12238

[169] C. C. Williams and M. S. Shahid, "Informal entrepreneurship and institutional theory: Explaining the varying degrees of (in) formalization of entrepreneurs in pakistan," *Entrepreneurship & Regional Development*, vol. 28, no. 1-2, pp. 1–25, 2016. [Online]. Available: https://rsa.tandfonline.com/doi/abs/10.1080/08985626.2014.963889#.XMr7bugzbIU

[170] J. Fairhead, "Technology, inclusivity and the rogue: bats and the war against the 'invisible enemy'," *Conservation and Society*, vol. 16, no. 2, pp. 170–180, 2018. [Online]. Available: http://sro.sussex.ac.uk/id/eprint/70018/

[171] E. S. Izmailova, J. A. Wagner, and E. D. Perakslis, "Wearable devices in clinical trials: hype and hypothesis," *Clinical Pharmacology & Therapeutics*, vol. 104, no. 1, pp. 42–52, 2018. [Online]. Available: https://ascpt.onlinelibrary.wiley.com/doi/full/10.1002/cpt.966

[172] S. Kato, M. Ando, T. Kondo, Y. Yoshida, H. Honda, and S. Maruyama, "Lifestyle intervention using internet of things (iot) for the elderly: A study protocol for a randomized control trial (the best-life study)," *Nagoya journal of medical science*, vol. 80, no. 2, p. 175, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5995741/

[173] P. H. Vilela, J. J. Rodrigues, P. Solic, K. Saleem, and V. Furtado, "Performance evaluation of a fog-assisted iot solution for e-health applications," *Future Generation Computer Systems*, vol. 97, pp. 379–386, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18323458

[174] C. Bravo, C. Sarraute, B. Baesens, J. Vanthienen *et al.*, "Credit scoring for good: Enhancing financial inclusion with smartphone-based microlending," 2018. [Online]. Available: https://aisel.aisnet.org/icis2018/implement/Presentations/6/

[175] N. Kumar, K. Raghunathan, A. Arrieta, A. Jilani, S. Chakrabarti, P. Menon, and A. R. Quisumbing, "Social networks, mobility, and political participation: The potential for women's self-help groups to improve access and use of public entitlement schemes in india," *World Development*, vol. 114, pp. 28–41, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305750X18303553

[176] R. Suryawanshi and G. Mandlik, "Focusing on mobile users at edge and internet of things using fog computing," *Int. J Sci. Eng. Tech. Res*, vol. 4, no. 17, pp. 3225–3231, 2015. [Online]. Available: http://ijsetr.com/uploads/412365IJSETR4999-598.pdf

[177] D. Das, "A fuzzy multiobjective approach for network reconfiguration of distribution systems," *IEEE transactions on power delivery*, vol. 21, no. 1, pp. 202–209, 2006. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1564201/

[178] S. Rathore, P. K. Sharma, A. K. Sangaiah, and J. H. Park, "A hesitant fuzzy based security approach for fog and mobile-edge computing," *IEEE Access*, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8114191/

[179] D. Bendouda, A. Rachedi, and H. Haffaf, "Programmable architecture based on software defined network for internet of things: Connected dominated sets approach," *Future Generation Computer Systems*, vol. 80, pp. 188–197, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17314590

[180] B. Khalifa, A. Khedr, Z. Al Aghbari, and J. Abawajy, "Fuzzy logic approach to repair coverage holes in internet of things monitoring applications," *IET Wireless Sensor Systems*, vol. 9, no. 4, pp. 227–235. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-wss.2018.5174

[181] X. He, Z. Ren, C. Shi, and J. Fang, "A novel load balancing strategy of software-defined cloud/fog networking in the internet of vehicles," *China Communications*, vol. 13, no. 2, pp. 140–149, 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7405730/

[182] T. J. Ross, *Fuzzy logic with engineering applications*. John Wiley & Sons, 2009. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=3zcgIKPl8L0C&oi=fnd&pg=PR7&dq=Fuzzy+logic+with+engineering+applications&ots=-AyLRUSJ4G&sig=oybmwMe9QagSuwCwVY6Isd2lmkY

[183] R. N. Yadav, R. Misra, and S. Bhagat, "Spectrum access in cognitive smart-grid communication system with prioritized traffic," *Ad Hoc Networks*, vol. 65, pp. 38–54, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1570870517301361

[184] A. Lewandowski, M. Putzke, V. Koster, and C. Wietfeld, "Coexistence of 802.11 b and 802.15. 4a-css: Measurements, analytical model and simulation," in *2010 IEEE 71st Vehicular Technology Conference*. IEEE, 2010, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5493881

[185] O. Fratu, E. C. Popovici, A. Vulpe, and S. V. Halunga, "Heterogeneous wireless access networks analysis from simulation to implementation," in *2011 10th International Conference on Telecommunication in Modern Satellite Cable and Broadcasting Services (TELSIKS)*, vol. 2. IEEE, 2011, pp. 481–488. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6143248

[186] S. S. Shihab and I. J. Mohammed, "Pim-sm based multicast comparison for ipv4 verses ipv6 using gns3 and jperf," *Iraqi Journal of Science*, vol. 58, no. 1A, pp. 140–151, 2017. [Online]. Available: https://www.iasj.net/iasj?func=article&aId=123230

[187] R. D. Hegazy, O. A. Nasr, and H. A. Kamal, "Optimization of user behavior based handover using fuzzy q-learning for lte networks," *Wireless Networks*, vol. 24, no. 2, pp. 481–495, 2018. [Online]. Available: https://link.springer.com/article/10.1007/s11276-016-1348-2

[188] A. A. Yayah, A. S. Ismail, T. Al-Hadhrami, and Y. Coulibaly, "Intelligent offset time algorithm for optical burst switching networks," in *International Conference of Reliable Information and Communication Technology*. Springer, 2018, pp. 427–439. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-99007-1_41

[189] S. P. Singh, "Fuzzylite with ns3 simulator," 2020. [Online]. Available: https://github.com/SimarPreetSingh/FuzzyLite

[190] P. Wang, X. Chen, and Z. Sun, "Performance modeling and suitability assessment of data center based on fog computing in smart systems," *IEEE Access*, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8368197/

[191] W.-S. Kim and S.-H. Chung, "User-participatory fog computing architecture and its management schemes for improving feasibility," *IEEE Access*, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8319964/

[192] C. Thota, R. Sundarasekar, G. Manogaran, R. Varatharajan, and M. Priyan, "Centralized fog computing security platform for iot and cloud in healthcare system," in *Exploring the convergence of big data and the internet of things*. IGI Global, 2018, pp. 141–154. [Online]. Available: https://www.igi-global.com/chapter/centralized-fog-computing-security-platform-for-iot-and-cloud-in-healthcare-system/205985

[193] P. Galiotos, T. Dagiuklas, and S. Kotsopoulos, "Call-level voip traffic modelling based on data from a real-life voip service provider," in *Globecom Workshops (GC Wkshps), 2015 IEEE*. IEEE, 2015, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7414101/

[194] L. Gurel and O. Ergul, "Design and simulation of circular arrays of trapezoidal-tooth log-periodic antennas via genetic optimization," *Progress In Electromagnetics Research*, vol. 85, pp. 243–260, 2008. [Online]. Available: http://www.jpier.org/PIER/pier.php?paper=08081809

[195] Ö. Ergül and L. Gürel, "Nonplanar trapezoidal tooth log-periodic antennas: Design and electromagnetic modeling," *Radio Science*, vol. 40, no. 5, 2005. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2004RS003215

[196] F. Zou, D. Chen, R. Lu, and P. Wang, "Hierarchical multi-swarm cooperative teaching–learning-based optimization for global optimization," *Soft Computing*, vol. vol. 21, no. 23, pp. pp. 6983–7004, Dec. 2017. [Online]. Available: https://link.springer.com/article/10.1007/s00500-016-2237-4

[197] J. Nayak, B. Naik, D. Kanungo, and H. Behera, "A hybrid elicit teaching learning based optimization with fuzzy c-means (etlbo-fcm) algorithm for data clustering," *Ain Shams Engineering Journal*, vol. vol. 9, no. 3, pp. pp. 379–393, Sep. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090447916000289

[198] R. G. Pierse, "Economic Forecasting Lecture 9: Smoothing Methods," pp. pp. 1–9. [Online]. Available: http://rpierse.esy.es/rpierse/files/bf9.pdf

[199] R. J. Hyndman, "Moving averages," pp. pp. 1–5, Nov. 2009. [Online]. Available: https://robjhyndman.com/papers/movingaverage.pdf

[200] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Computer Networks*, vol. vol. 57, no. 16, pp. pp. 3128–3141, Nov. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1389128613002235

[201] D. Das and R. Misra, "Caching algorithm for fast handoff using ap graph with multiple vehicles for vanets," *International Journal of Communication Networks and Distributed Systems*, vol. 14, no. 3, pp. 219–236, 2015. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJCNDS.2015.068662

[202] N. Sharma and A. K. Sharma, "Cost analysis of hybrid adaptive routing protocol for heterogeneous wireless sensor network," *Sadhana*, vol. vol. 41, no. 3, pp. pp. 283–288, Mar. 2016. [Online]. Available: http://link.springer.com/10.1007/s12046-016-0469-8

[203] B. W. Silverman, *Density estimation for statistics and data analysis*. Routledge, 2018, vol. 1st Edition. [Online]. Available: https://www.taylorfrancis.com/books/9781351456173

[204] Y. Xiang, J. Donley, E. Seletskaia, S. Shingare, J. Kamerud, and B. Gorovits, "A simple approach to determine a curve fitting model with a correct weighting function for calibration curves in quantitative ligand binding assays," *The AAPS Journal*, vol. vol. 20, no. 3, pp. pp. 20–45, Mar 2018. [Online]. Available: https://link.springer.com/article/10.1208%2Fs12248-018-0208-7

[205] R. V. Rao, V. J. Savsani, and D. Vakharia, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0010448510002484

[206] S. Suganthi, D. Devaraj, S. H. Thilagar, and K. Ramar, "Optimal generator rescheduling with distributed slack bus model for congestion management using improved teaching learning based optimization algorithm," *Sādhanā*, vol. vol. 43, no. 11, p. 181, Nov. 2018. [Online]. Available: https://link.springer.com/article/10.1007/s12046-018-0941-8

[207] K. Yu, L. While, M. Reynolds, X. Wang, J. Liang, L. Zhao, and Z. Wang, "Multiobjective optimization of ethylene cracking furnace system using self-adaptive multiobjective teaching-learning-based optimization," *Energy*, vol. vol. 148, pp. pp. 469–481, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0360544218301877

[208] D. Yu, J. Hong, J. Zhang, and Q. Niu, "Multi-objective individualized-instruction teaching-learning-based optimization algorithm," *Applied Soft Computing*, vol. vol. 62, pp. pp. 288–314, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494617305380

[209] D. Lei, L. Gao, and Y. Zheng, "A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop," *IEEE Transactions on Engineering Management*, vol. vol. 65, no. 2, pp. pp. 330–340, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8194867

[210] A. Birashk, J. K. Kordestani, and M. R. Meybodi, "Cellular teaching-learning-based optimization approach for dynamic multi-objective problems," *Knowledge-Based Systems*, vol. vol. 141, pp. pp. 148–177, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705117305385

[211] R. V. Rao, *Teaching learning based optimization algorithms: and its engineering applications*. Cham Heidelberg New York Dordrecht London: Springer, 2016, oCLC: 913557274. [Online]. Available: https://www.springer.com/in/book/9783319227313

[212] S. P. Das and S. Padhy, "A novel hybrid model using teaching–learning-based optimization and a support vector machine for commodity futures index forecasting," *International Journal of Machine Learning and Cybernetics*, vol. vol. 9, no. 1, pp. pp. 97–111, Jan. 2018. [Online]. Available: https://link.springer.com/article/10.1007/s13042-015-0359-0

[213] J. Nayak, B. Naik, H. Behera, and A. Abraham, "Elitist teaching–learning-based optimization (etlbo) with higher-order jordan pi-sigma neural network: a comparative performance analysis," *Neural Computing and Applications*, vol. vol. 30, no. 5, pp. pp. 1445–1468, Sep. 2018. [Online]. Available: https://link.springer.com/article/10.1007/s00521-016-2738-1

[214] M. Maiti, S. Mukherjee, and P. K. G. Thakurta, "An energy efficient teaching learning based optimization approach for common content distribution in mobile ad hoc networks," *Computers & Electrical Engineering*, vol. vol. 72, pp. pp. 296–306, Nov. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790617303361

[215] E. Natarajan, V. Kaviarasan, W. H. Lim, S. S. Tiang, and T. H. Tan, "Enhanced multi-objective teaching-learning-based optimization for machining of delrin," *IEEE Access*, vol. vol. 6, pp. pp. 51 528–51 546, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8466876

[216] S. P. Singh, "Mobile call frequency analysis dataset for fog-based inclusion analysis," *Mendeley Data, v1*, 2019. [Online]. Available: http://dx.doi.org/10.17632/mrr6c6m3f5.1

[217] S. P. Singh, "Geospatial frequency analysis dataset for fog computing based heterogeneous networks," *Mendeley Data, v1*, 2019. [Online]. Available: https://data.mendeley.com/datasets/cdc925rps3/1

[218] R. Misra and C. Mandal, "Efficient clusterhead rotation via domatic partition in self-organizing sensor networks," *Wireless Communications and Mobile Computing*, vol. 9, no. 8, pp. 1040–1058, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/wcm.662

[219] P. Goyal, N. Mehala, D. Bhatia, and N. Goyal, "Topical document clustering: two-stage post processing technique," *International Journal of Data Mining, Modelling and Management*, vol. 10, no. 2, pp. 127–170, 2018. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJDMMM.2018.092536

[220] M. Wilkes, J. Thornton, M. Horlick, A. Sopher, J. Wang, E. Widen, R. Pierson, and D. Gallagher, "Relationship of bmi z score to fat percent and fat mass in multiethnic prepubertal children," *Pediatric obesity*, vol. 14, no. 1, p. e12463, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/ijpo.12463

[221] A. J. Crean and A. M. Senior, "High-fat diets reduce male reproductive success in animal models: A systematic review and meta-analysis," *Obesity Reviews*, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/obr.12827

[222] S. Stark, M. Weatherborn, and D. Landau, "Normal reference values for percent body fat at term [30c]," *Obstetrics & Gynecology*, vol. 133, 2019. [Online]. Available: https://journals.lww.com/greenjournal/Abstract/2019/05001/Normal_Reference_Values_for_Percent_Body_Fat_at.131.aspx

[223] D. J. Tomlinson, R. M. Erskine, C. I. Morse, and G. L. Onambélé, "Body fat percentage, body mass index, fat mass index and the ageing bone: Their singular and combined roles linked to physical activity and diet," *Nutrients*, vol. 11, no. 1, p. 195, 2019. [Online]. Available: https://www.mdpi.com/2072-6643/11/1/195

[224] S. Noronha, P. Lima, G. Campos, M. Chírico, A. Abreu, A. Figueiredo, F. Silva, D. Chianca Jr, C. Lowry, and R. Menezes, "Association of high-fat diet with neuroinflammation, anxiety-like defensive behavioral responses, and altered thermoregulatory responses in male rats," *Brain, behavior, and immunity*, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0889159119301370?via=ihub

[225] M. Szucs, P. Osvath, A. Jakab, D. Varga, B. Varga, and B. Juhasz, "Hyaluronan bound mature sperm count (hb-masc) is a more informative indicator of fertility than conventional sperm parameters: Correlations with body mass index (bmi)," *Reproductive biology*, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1642431X18301839?via=ihub

[226] W. C. Willett, J. E. Manson, M. J. Stampfer, G. A. Colditz, B. Rosner, F. E. Speizer, and C. H. Hennekens, "Weight, weight change, and coronary heart disease in women: risk within the'normal'weight range," *Jama*, vol. 273, no. 6, pp. 461–465, 1995. [Online]. Available: https://jamanetwork.com/journals/jama/article-abstract/386879

[227] B. I. Campbell, R. J. Colquhoun, G. Zito, N. Martinez, K. Kendall, L. Buchanan, M. Lehn, M. Johnson, C. S. Louis, Y. Smith *et al.*, "The effects of a fat loss supplement on resting metabolic rate and hemodynamic variables in resistance trained males: a randomized, double-blind, placebo-controlled, cross-over trial," *Journal of the International Society of Sports Nutrition*, vol. 13, no. 1, p. 14, 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4818444/

[228] B. I. Campbell, G. Zito, R. Colquhoun, N. Martinez, K. Kendall, L. Buchanan, M. Lehn, M. Johnson, C. S. Louis, Y. Smith *et al.*, "The effects of a single-dose thermogenic supplement on resting metabolic rate and hemodynamic variables in healthy females-a randomized, double-blind, placebo-controlled, cross-over trial," *Journal of the International Society of Sports Nutrition*, vol. 13, no. 1, p. 13, 2016. [Online]. Available: https://jissn.biomedcentral.com/track/pdf/10.1186/s12970-016-0123-1

[229] J. Thomas and R. T. Princy, "Human heart disease prediction system using data mining techniques," in *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*. IEEE, 2016, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/7530265

[230] A. Sikdar, S. K. Behera, and D. P. Dogra, "Computer-vision-guided human pulse rate estimation: a review," *IEEE reviews in biomedical engineering*, vol. 9, pp. 91–105, 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7448856

[231] O. Odigie, A. Naiho, E. Nwangwa, E. Aisuodione, and J. Igweh, "Correlation of human fasting blood sugar with grip muscle strength and reflex response time," *British Journal of Medicine & Medical Research*, vol. 14, no. 6, pp. 1–10, 2016. [Online]. Available: https://www.journaljammr.com/index.php/JAMMR/article/view/14692/27065

[232] F. Bertin, S. D. Taylor, A. Bianco, and J. Sojka-Kritchevsky, "The effect of fasting duration on baseline blood glucose concentration, blood insulin concentration, glucose/insulin ratio, oral sugar test, and insulin response test results in horses," *Journal of veterinary internal medicine*, vol. 30, no. 5, pp. 1726–1731, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1111/jvim.14529

[233] I. Prokopenko, C. Langenberg, J. C. Florez, R. Saxena, N. Soranzo, G. Thorleifsson, R. J. Loos, A. K. Manning, A. U. Jackson, Y. Aulchenko *et al.*, "Variants in mtnr1b influence fasting glucose levels," *Nature genetics*, vol. 41, no. 1, p. 77, 2009. [Online]. Available: https://www.nature.com/articles/ng.290

[234] S. Søberg, C. H. Sandholt, N. Z. Jespersen, U. Toft, A. L. Madsen, S. von Holstein-Rathlou, T. J. Grevengoed, K. B. Christensen, W. L. Bredie, M. J. Potthoff *et al.*, "Fgf21 is a sugar-induced hormone associated with sweet intake and preference in humans," *Cell metabolism*, vol. 25, no. 5, pp. 1045–1053, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1550413117302140

[235] S. P. Singh, "Machine learning based fitness dataset for fog-based inclusion analysis," *Mendeley Data, v1*, 2019. [Online]. Available: http://dx.doi.org/10.17632/pgjdr7s9ky.1

[236] M. C. Golumbic, *Algorithmic graph theory and perfect graphs*. Elsevier, 2004. [Online]. Available: https://www.sciencedirect.com/book/9780122892608/algorithmic-graph-theory-and-perfect-graphs

[237] D. B. West *et al.*, *Introduction to graph theory*. Prentice hall Upper Saddle River, 2001, vol. 2. [Online]. Available: http://docshare01.docshare.tips/files/26167/261678089.pdf