OPTIMIZED VOID-AWARE ROUTING PROTOCOL FOR UNDERWATER ACOUSTIC SENSOR NETWORKS

Thesis

Submitted in partial fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

PRADEEP NAZARETH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575 025

June, 2023

DECLARATION

by the Ph.D. Research Scholar

I hereby declare that the Research Thesis entitled **Optimized Void-Aware Routing Protocol for Underwater Acoustic Sensor Networks** which is being submitted to the **National Institute of Technology Karnataka, Surathkal** in partial fulfilment of the requirements for the award of the Degree of **Doctor of Philosophy** in Department of Computer Science and Engineering is a bonafide report of the research work carried out by me. The material contained in this Research Thesis has not been submitted to any University or Institution for the award of any degree.

Pradeep Nazareth, 177028CO003
Department of Computer Science and Engineering

Place: NITK, Surathkal.
Date: June 27, 2023

CERTIFICATE

This is to certify that the Research Thesis entitled **Optimized Void-Aware Routing Protocol for Underwater Acoustic Sensor Networks** submitted by **Pradeep Nazareth** (Register Number: 177028CO003) as the record of the research work carried out by him, is accepted as the Research Thesis submission in partial fulfilment of the requirements for the award of degree of **Doctor of Philosophy**.

03.07.23

Dr. B. R. Chandavarkar
Research Guide

03/07/23

Chairman - DRPC
(Signature with Date and Seal)

Chairman
~~DUGC~~ / ~~DPGC~~ / DRPC
Dept. of Computer Engg.
NITK - Surathkal
Srinivasnagar - 575 025

ACKNOWLEDGEMENTS

I am pleased to thank the people who have supported and helped me throughout my research. First and foremost, I would like to express my sincere gratitude to my research supervisor Dr. B. R. Chandavarkar, Asst. Professor, Department of Computer Science and Engineering, for his constant support and encouragement. He has been a continuous source of inspiration and guiding light throughout my research. I have learned a lot from him, both from a technical standpoint and in the form of life lessons, which will be of use throughout my life.

I express my sincere gratitude to my Research Progress Assessment Committee (RPAC) members, Prof. Neelawar Shekar Vittal Shet, Professor and Head, Department of Electronics and Communication Engineering, and Dr. Shashidhar G Koolagudi, Associate Professor, Department of Computer Science and Engineering. They have been a continuous source of inspiration with their valuable suggestions to assist me during my Ph.D. journey. I would like to thank my instructor during course work, Dr. Mohit P. Tahiliani, Asst. Professor, Department of Computer Science and Engineering, for his constant support and encouragement. I would like to express my heart-filled gratitude to the head of the department Dr. Manu Basavaraju, all previous heads of the department, and all the professors of the department for helping me at various stages of my Ph.D studies. Further, I thank the National Institute of Technology Karnataka, Surathkal for providing a suitable environment for the research work.

I sincerely thank all teaching, technical, and administrative staff of the Computer Science and Engineering Department for their help during my research.

I would like to thank Mr. Sandeep M., Ms. Sneha Kamble, Mr. Akhilraj Gadagkar, and Mr. Nitesh Naik for their support and encouragement during my research.

I thank my parents and family for their support and encouragement. I want to thank my wife, Jevitha, and son Johan; without them, this research work would not have been possible. I would like to thank the almighty for granting me the health, strength, and wisdom for carrying out my research work.

Pradeep Nazareth

ABSTRACT

Underwater Acoustic Sensor Networks (UASNs) are the technologies used to explore underwater resources. UASNs have been used in numerous applications such as environmental monitoring, underwater surveillance, underwater exploration, detection of resources and disasters, etc. However, UASNs face several fundamental issues like low bandwidth due to the environmental noise, high bit error rate as a result of fading and multipath propagation, energy constraints on nodes, security as they are vulnerable to active and passive attacks, complicated routing due to dynamic network topology and variation of the link quality between nodes.

The UASNs architecture consists of sensor nodes deployed underwater for sensing the events and forwarding or routing the data, in one or multiple hops, to the sink nodes deployed at the water's surface. Wireless routing has three significant categories, that are proactive, reactive, and geographic routing. However, proactive routing requires transmission of more number control packets, thus increasing energy consumption and overhead on the network. The reactive routing results in increased end-to-end delay due to high propagation delay. Geographic routing forwards the data, using the position information of the neighbors and the sink. It uses greedy forwarding, and every node determines only its next hop, rather than the end-to-end path.

Geographic routing is the most suitable protocol to forward data from the source to the sink node in UASNs. However, communication void or void node is one of the major challenges in UASNs to deliver the data to the sink reliably. The non-availability of a neighbor, in the positive progress to a source or forwarding node, results in a communication void. Communication void impacts the performance of the UASNs in terms of packet loss, high end-to-end delay, waste of energy, etc. Primary reasons for communication void are, node movement due to water current, ship movement, or a drop in Signal-to-Noise Ratio (SNR) between nodes. Many methods are proposed in the literature to deal with the communication void, such as backward forwarding, topology

adjustment, transmission power adjustment, etc. The major drawbacks of these methods are, void nodes as a part of routing, loops, unreachable data to the sink, more duplicate packets, hidden-node issues, and more energy consumption.

This thesis mainly addresses the issues of the communication void in underwater routing. In existing void-aware routing protocols, the source/forwarder node decides the next hop using multiple attributes, such as hop count, residual energy, distance with the neighbor, depth, Packet Delivery Probability (PDP), status (void or normal), etc., of the neighboring nodes. However, the priorities of the individual attributes are not considered in determining the next hop(s). Hence, this thesis presents the selection or identification of an appropriate combination of attributes of neighboring nodes. Accordingly, this thesis proposes the Enhanced-Void-Aware Routing (E-VAR) protocol, which uses a combination of the neighbor's status and Euclidean distance between the neighbor to the sink attributes to decide the next hop for delay-sensitive applications. Further, Link Quality-based Routing Protocol (LQRP) proposed in this thesis uses link quality between source to neighbors and neighbor to its best hop as attributes. Additionally, applying appropriate weights, a suitable neighbor is selected as its next hop. The LQRP protocol achieves better reliability than the state-of-the-art protocol. The Location-Free Void Avoidance Routing (LFVAR) protocol proposed in this thesis uses status, hop count, and depth of neighbor as attributes. Further, by computing the cost of neighbors, one of them is selected as the next hop. The Link and Void-Aware Routing (LVAR) protocol proposed in this thesis uses status, PDP, and hop count of neighbor as attributes to select the next hop.

The state-of-the-art routing protocols proposed in the literature do not consider Multiple Attribute Decision Making (MADM) techniques to evaluate the neighboring nodes using identified attributes. Hence, this study proposes a Cluster-based Multi-Attribute Routing (CMAR) protocol. CMAR is a sender-based, opportunistic routing protocol. The source/forwarder node evaluates its neighbors using the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) method. Additionally, it forms the cluster(s) of neighboring nodes, consisting of a threshold number of nodes in the vicinity of each other. The source/forwarder node forwards the data to the cluster using

opportunistic routing.

The protocols (E-VAR, LQRP, LFVAR, LVAR, and CMAR) designed, as a part of this research work, are simulated and evaluated in industry-standard simulators such as MATLAB and UnetStack. UnetStack is an agent-based simulator used to develop and evaluate underwater protocols. Further, E-VAR, LQRP, LFVAR, LVAR, and CMAR are evaluated in terms of various metrics such as the number of nodes reachable to the sink, number of nodes not reachable to the sink due to loops, packet delivery ratio, hop count, propagation distance from source to the sink, throughput, number of the clusters formed, number of times a void node is part of the routing, etc.

In conclusion, the major contribution of this thesis, focuses on identifying the most suitable combination of attributes of neighbors to select the next hop(s) with E-VAR, LQRP, LFVAR, and LVAR, further evaluating neighboring nodes, using the MADM approach with CMAR protocol. Additionally, designed protocols are evaluated using MATLAB or UnetStack and are compared with state-of-the-art routing protocols.

Keywords: Underwater routing, Void node, Communication void, Multiple Attribute Decision Making, Next hop, Hop count.

CONTENTS

List of Figures x

List of Tables xii

List of Algorithms xiii

List of Abbreviations and Acronyms xv

1 Introduction 1

1.1 Challenges of UASNs 4
1.2 Underwater routing strategies 6
1.3 Void node in UASNs 7
1.4 Void-handling in UASNs 9
1.5 Multi-Attribute Decision Making 10
1.6 Motivation 10
1.7 Contributions of this thesis 11
1.8 Organization of the thesis 13
1.9 Summary 14

2 Literature Survey 15

2.1 Void-handling routing protocols 15
2.2 Location-based routing protocols 17
2.3 Depth-based routing protocols 22
2.4 Comparative analysis of void-handling routing protocols 28
2.5 Summary 30

3 Problem statement and objectives 31

3.1 Problem statement 31

3.2 Research objectives 32

3.3 Summary 33

4 Design and evaluation of Enhanced-Void-Aware Routing (E-VAR) protocol 35

4.1 Design of the E-VAR protocol 35

4.1.1 Phase-I: Status (void or normal) identification by a node 37

4.1.2 Phase-II: Identifying a next hop 38

4.2 Simulation and evaluation of the E-VAR protocol 39

4.2.1 Case-I: Manual deployment 40

4.2.2 Case-II: Random deployment 46

4.3 summary 49

5 Design and evaluation of Link Quality-based Routing Protocol (LQRP) 51

5.1 Design of the LQRP protocol 51

5.1.1 Computation of SNR 52

5.1.2 Identifying a next hop in LQRP 55

5.2 Simulation and evaluation of the LQRP 56

5.2.1 Simulation setup 57

5.2.2 Results and analysis 58

5.3 Summary 60

6 Design and evaluation of Location-Free Void Avoidance Routing (LFVAR) protocol 61

6.1 Design of the LFVAR protocol 61

6.1.1 Phase-I: Initialization of hop count, status, and depth followed by the broadcasting of beacon, by the sink 62

6.1.2 Phase-II: Identifying the next hop 63

6.2 Implementations of LFVAR 65

6.2.1 UnetStack 66

6.2.2 The *SinkAgent* 67

6.2.3 The *NodeAgent* 68
6.3 Simulation and evaluation of the LFVAR protocol 69
6.3.1 Next hop selection in Intar and LFVAR 69
6.3.2 Performance comparison between LFVAR and Intar 72
6.4 Summary 75

7 Design and evaluation of Link and Void-Aware Routing (LVAR) Protocol 77
7.1 Design of the LVAR protocol 77
7.1.1 Computation of packet delivery probability 78
7.2 Simulation and evaluation of the LVAR protocol 79
7.2.1 Next hop selection in Intar and LVAR 80
7.2.2 Simulation results 82
7.3 Summary 84

8 Design and evaluation of Cluster-based Multi-Attribute Routing (CMAR) Protocol 85
8.1 Design of the CMAR protocol 85
8.1.1 Phase-I: Score computation and clustering 86
8.1.2 Phase-II: Forwarding received data 94
8.2 Simulation and evaluation of the CMAR protocol 98
8.2.1 Simulation setup 99
8.2.2 Simulation results 99
8.3 summary 104

9 Conclusions and Future Scope 105
9.1 Summary of contributions 105
9.2 Future scope 107

Bibliography 108

Publications 121

Bio-data 123

LIST OF FIGURES

1.1 UASNs architecture. 2
1.2 Types of the wireless routing protocols. 6
1.3 Void node scenario in underwater communication. 7

2.1 Categorization of void-handling underwater routing protocols. 16
2.2 Void-handling in VBVA (a) Vector-shift mechanism (b) Back-pressure mechanism. 18
2.3 Packet forwarding in DFR protocol. 19
2.4 Candidate node selection in the IVAR. 25

4.1 Phase-I: Status (void or normal) identification by a node in E-VAR. . . . 36
4.2 Status (void or normal) detection by node 3, (a) $V_3 = 0$ (b) $V_3 = 1$. . . . 38
4.3 Phase-II: Identifying the next hop in E-VAR. 39
4.4 Looping scenario in Backward-forwarding. 41
4.5 Looping scenario in Intar. 42
4.6 Scenario showing avoidance of looping in E-VAR. 43
4.7 Hop count (left) and Distance to reach sink (right), in Backward-forwarding, Intar, and E-VAR with reference to Figure 4.4 through 4.6. 44
4.8 Backtracking scenario in Backward-forwarding and Intar. 44
4.9 Avoidance of backtracking scenario in E-VAR. 45
4.10 Hop count (left) and distance to reach sink (right) in Backward–forwarding, Intar, and E-VAR with reference to Figure 4.8 and 4.9. 46

5.1 Identifying the next hop in LQRP. 55
5.2 Simulation topology to evaluate LQRP with LQER. 56
5.3 SNR-PDR profile (Tan et al. 2012). 58

5.4 Computation of overall PDR. 58
5.5 Overall PDR between a source to the sink when (a) $W = 1$ (b) $W = 0.75$ (c) $W = 0.5$ (d) $W = 0.25$ (e) $W = 0$. 59

6.1 Phase-II: Determining the next hop at node i (except the sink). 63
6.2 Agents used in the implementation of LFVAR protocol in UnetStack. . . 65
6.3 Organization of *SinkAgent*. 67
6.4 Organization of *NodeAgent*. 68
6.5 Next hop selection in the Intar. 70
6.6 Next hop selection in the LFVAR. 71
6.7 Average hop count in LFVAR and Intar. 72
6.8 Average end-to-end delay in LFVAR and Intar. 73
6.9 PDR of LFVAR and Intar. 74
6.10 Throughput of LFVAR and Intar. 74
6.11 Energy consumption in LFVAR and Intar. 75

7.1 Next hop selection in the Intar. 80
7.2 Next hop selection in the LVAR. 81
7.3 Hop count in LVAR and Intar. 82
7.4 Average end-to-end delay in LVAR and Intar. 83
7.5 Throughput in LVAR and Intar. 83

8.1 Example of the cluster formation of node i in the CMAR with Th = 4. . 92
8.2 Hold time computation for node $j_1(P_{j_1} = 1)$, $j_3(P_{j_3} = 2)$, $j_5(P_{j_5} = 3)$. . 95
8.3 Packet reception and forwarding by node j in CMAR. 97
8.4 Comparison of CMAR and HydroCast w.r.t. chances of forwarding packets. 101
8.5 Average EPA of CMAR and HydroCast. 102
8.6 Percentage (%) input combinations, Nodes at (a) Priority 1 (left) (b) Priority 2 (right). 103
8.7 Transmission reliability vs percentage of input combinations. 104

LIST OF TABLES

2.1 Comparison between location and depth-based underwater routing protocols. 17
2.2 Analysis of void-handling routing protocols. 28

4.1 Advancement of neighbors of node 2 and 5 in Figure 4.4. 41
4.2 Cost of PFNs of node 2. 43
4.3 Simulation parameters used in random deployment. 46
4.4 Performance metrics for Backward-forwarding, Intar, and E-VAR- Nodes reachable to the sink, Nodes unreachable to the sink due to looping. . . 47
4.5 Performance metrics for Backward-forwarding, Intar, and E-VAR- Average (Avg.) hop count, Avg. distance to reach the sink. 48

5.1 Simulation parameters used in LQRP and LQER. 57
5.2 Average PDR in LQRP for various weight W. 60

6.1 Cost of PFN of node 5. 70
6.2 Depth difference with neighbors of nodes 6 and 1. 71
6.3 Simulation parameters used in the evaluation of the LFVAR. 72

7.1 Cost of PFNs of node 5 & 3. 80
7.2 Simulation parameters used in LVAR and Intar. 82

8.1 Hold time calculation of nodes with reference to Figure 8.2 with T_{TRANS} = 0.5 sec, T_{PROC} = 0.25 sec, and V = 100 m/s. 96
8.2 Simulation parameters used in the evaluation of the CMAR. 99
8.3 Number of nodes in the selected cluster(s) in the CMAR and HydroCast. 99
8.4 Various scenarios of cluster formation in CMAR with $Th = 3$. 100

LIST OF ALGORITHMS

1 Generation of input combination of SNR between nodes. 57

2 Updating PRN_i at the node i. 64

3 Cluster formation in CMAR at the node i. 91

LIST OF ABBREVIATIONS AND ACRONYMS

Abbreviations	**Expansion**
AHH-VBF	Adaptive Hop-by-Hop Vector-Based Forwarding
AUVs	Autonomous Underwater Vehicles
AWGN	Additive White Gaussian Noise
BP	Back- Pressure
CARP	Channel-Aware Routing Protocol
CF	Cost Function
CMAR	Cluster-based Multi-Attribute Routing
DFR	Directional Flooding-based geographic Routing
DVOR	Distance-Vector-based Opportunistic Routing
E-VAR	Enhanced-Void-Aware Routing
EPA	Effective Packet Advancement
ETX	Expected Transmission Count
EVAGR	Energy-efficient Void Avoidance Geographic Routing
EWMA	Exponentially Weighted Moving Average
GEDAR	GEographic and opportunistic routing with Depth Adjustment-based topology control for communication Recovery
Intar	Interference-aware Routing
IVAR	Inherently Void Avoidance Routing
LFVAR	Location-Free Void Avoidance Routing
LLSR	Location-free Link State Routing
LQER	Link Quality Estimation based Routing
LQRP	Link Quality-based Routing Protocol

LVAR	Link and Void Aware Routing
MADM	Multi-Attribute Decision Making
NADV	Normalized Advancement
OVAR	Opportunistic Void Avoidance Routing
PFN	Potential Forwarding Nodes
PDP	Packet Delivery Probability
PDR	Packet Delivery Ratio
PRNs	Potential Relay Node(s)
SAW	Simple Additive Weighting
SNR	Signal-to-Noise Ratio
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
UASNs	Underwater Acoustic Sensor Networks
UWSNs	Underwater Wireless Sensor Networks
VAPR	Void-Aware Pressure Routing
VBF	Vector-Based Forwarding
VBVA	Vector-Based Void Avoidance

Acronyms	**Meaning**
$\alpha(f)$	Absorption coefficient with frequency f
A^{i+}	Set of positive ideal solution of all attributes at node i
A^{i-}	Set of negative ideal solution of all attributes at node i
A_1	Boric acid level
A_2	Magnesium sulphate level in sea water
A_3	Pure water component
$ADV_{i,j}$	Advancement of node j with respect to i
$A(d, f)$	Path loss over a distance d with frequency f
$Cluster_{i,k}$	k^{th} cluster at node i
C_i	Candidate forwarding set of node i
$C_{i,j}$	Cost of node j with reference to i
CA_b	Current angle of node b
r	Distance between nodes
$depth\_difference_{i,j}$	Depth difference with node j with reference to i
D_i	Depth at which node i is deployed
$D_{i,S}$	Euclidean distance between node i and S
DI	Directivity index
$(DM^i[x][y])_{(m \times n)}$	Decision matrix at node i consists of m rows and n columns
E_b	Average transmission energy per bit
$(ED^i[x]^+)_(m \times 1)$	Euclidean distance from positive ideal solution, consist of m rows and 1 column
$(ED^i[x]^+)_(m \times 1)$	Euclidean distance from positive ideal solution, consist of m rows and 1 column
$E^i[y]$	Entropy of y^{th} attribute at node i
$EPA(S_l)$	Expected packet advancement of cluster S_l
f	Frequency of transmission
f_1	Relaxation frequency of Boric acid
f_2	Relaxation frequency of Magnesium sulphate

f_3	Pure water component in sea water
G	Number of source nodes
H	Mixed layer depth
H_j	Hop count of node j
(i_x, i_y, i_z)	x,y, and z-coordinates of node i
$j_{MAX(C_{i,j})}$	Neighboring node j with maximum cost
$j_{MIN(depth\_difference_{i,j})}$	Node j having minimum depth difference with reference to i)
k	Spreading factor
L_i	List of location information of neighbors of node i
$MIN(D_{j,S})$	Minimum distance among neighbors j to S
$next\_hop$	Next hop
$N(f)$	Noise level
N_0	Noise power density in nonfading AWGN
N_i	Neighbor list of node i
$NADV_{i,j}$	Normalized advancement of j with reference to i
$NT^i_{p\times q}$	Neighbor Table with p rows representing neighbors and q columns representing attributes
$p_e(d)$	Bit error Probability over a distance d
P_1	Depth pressure of Boric acid
P_2	Depth pressure of Magnesium sulphate
P_3	Depth pressure of pure water
$P_{i,j}$	Packet delivery probability between node i and j
P_i	List of all normal neighbors of node i
PRN_i	Potential Relay Nodes of node i
$Range$	Communication range
$(R^i[x][y])_{(m\times n)}$	Normalized matrix of node i consists of m rows and n columns
RA_a	Reference angle of node a
S_i	List contain, status of neighbor of node i

SL	Source level
SNR	Signal-To-Noise ratio
$Score^i[x]_{(m\times 1}$	Score at node i
$SNR(d)$	SNR over a distance d
SNR_{fh}	SNR between node with its neighbor or first hop
$SNR_{fh}(j)$	SNR between source/forwarding node to its neighbor j
SNR_{sh}	SNR between neighbor with its best hop
$SNR_{sh}(j)$	SNR between neighboring node j with its best hop
SS	Sea state
SY	Salinity
T^j_{HOLD}	Hold time of the node j
T^j_{PAUSE}	Pause period of the node j
$T^j_{OVERHEAR}$	Total overhearing wait period correspond to node j
T^j_{TRANS}	Total transmission wait period correspond to node j
T^j_{PROC}	Total processing wait period correspond to node j
Th	Threshold
TL	Transmission Loss
V_i	Status (void or normal) of node i
V	Propagation speed of acoustic signal underwater
W	Weight
W_i	List of all void neighbors of node i
$(W^i[y])_{(1\times n)}$	Matrix contains weight of n attributes
$W^i[y]$	Weight of y^{th} attribute at node i
$(WC^i[x][y])_{(m\times n)}$	Weight-cost matrix at node i consists of m rows and n columns

CHAPTER 1

INTRODUCTION

More than 70% of the earth's surface is covered by water. The depth of water bodies are reserve of minerals, petroleum, biological resources, etc. However, underwater depths are explored only to a small extent (Coutinho et al. 2018). Communication technology plays a major role in exploring the underwater and has attained a major focus among researchers and the business community. Moreover, oceans being the primary source of water, a cabled observation system is one of the initial solutions to monitor the oceans. However, they are not economically feasible to deploy and maintain in an extensive monitoring area (Chitre 2022b). To overcome the issues of cabled observation systems, the field of Underwater Wireless Sensor Networks (UWSNs) has emerged as a result of advances in wireless technology.

UWSNs can use three types of possible wireless technology for underwater communication: radio, optical and acoustic. However, in UWSNs, radio waves are absorbed by the water and transmit longer distances only at lower frequencies, resulting in the need for more transmission power. On the other hand, optical waves suffer from scattering. Thus, radio and optical waves are suitable only for short-range underwater communication (Chitre et al. 2014b; Akyildiz et al. 2005). The physical properties of acoustic waves perform better and are more suitable for long-range underwater communication (Zheng et al. 2019). The technology that uses acoustic waves for underwater communication is called Underwater Acoustic Sensor Networks (UASNs).

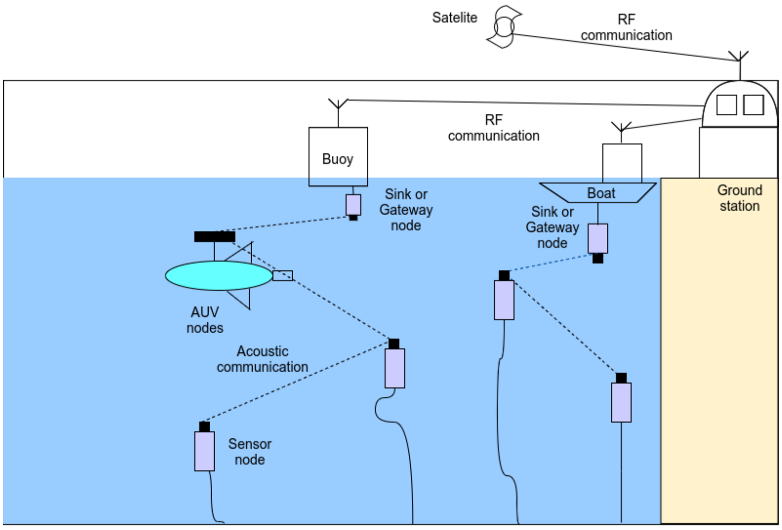

Figure 1.1: UASNs architecture.

Figure 1.1 presents the simple architecture of the UASNs. It consists of sensor nodes, Autonomous Underwater Vehicles (AUVs), a gateway node or sink, and a ground station. These are elaborated as follows:

- **Sensor node**

  The sensor nodes detect a particular event in underwater. One sensor node is capable of communicating with others, using an acoustic link. Further, the sensed event is transmitted to the gateway, or the sink through single or multi-hop communication, through intermediate sensor nodes.

- **Autonomous Underwater Vehicles (AUVs)**

  AUVs are vehicles which dive into the water and move in the monitoring area. Further, sense or relay the data and send it to the gateway or sink node.

- **Gateway node or sink**

  A gateway node or sink is deployed at the water surface. It has the capability of both acoustic and radio communication. The sink collects sensed data from underwater sensor nodes using acoustic links. Collected information at the sink is sent to the ground station using a radio link.

- **Ground station**

The ground station receives the data from the sink nodes deployed at the water surface. The ground station further processes the data received from the sink and sends it to external processing sources like satellites or remote stations.

UASNs have become a prominent field in communication due to their wide range of applications. UASNs have applications in disaster detection, resource detection, environmental monitoring, underwater surveillance, underwater exploration, etc. (Ismail et al. 2022a; Karim et al. 2021; Felemban et al. 2015; Akyildiz et al. 2004, 2006). These applications of UASNs are further elaborated as follows:

- **Disaster detection**
  Sensors deployed at nodes of UASNs measure the seismic activity and generate early warnings in case of a tsunami. For example, tsunami detection and warning system has been developed (Kumar et al. 2012). In another example, a Tsunami alert system is designed in which underwater nodes collect information on the magnetic field, wave gradient, and heat energy. Further, using this information and intelligent data retrieval technique, Tsunami is detected (Virmani and Jain 2016).

- **Resource detection**
  UASNs are used to explore various ocean mining and petroleum resources.

- **Environment monitoring**
  UASNs have a wide range of underwater environmental monitoring applications. To obtain information about the water quality in the ocean, river, and lake, UASNs can be used. Further, to study the variation of the temperature and salinity and to detect oil spills, UASNs play a significant role. For example, the aqueous sensor network system (Yang et al. 2002) is designed to monitor water quality in drinking water reservoirs. The system has a number of nodes. Nodes consist of sensors capable of measuring pH value. The node can also forward sensed data to the sink through multi-hop communication. An underwater network system is designed to monitor gradient in temperature (Zhang et al. 2004). Further, the event must notified if the gradient is beyond the threshold.

- **Underwater surveillance**

  UASNs is used in surveillance application where the intrusion of foreign objects is detected. For example, an underwater surveillance system is developed to detect submarines, small delivery vehicles, miners, and divers (Cayirci et al. 2006).

- **Undersea exploration**

  UASNs are used in determining the routes to the undersea cables and pipelines. They are also deployed to monitor the underwater pipeline infrastructure and notify the reliability issues of it (Mohamed et al. 2010).

Despite several applications of UASNs, it faces a number of challenges, and they are elaborated on, in the following section.

1.1 CHALLENGES OF UASNs

Major challenges of UASNs are their low bandwidth, large propagation delay, high bit error rate, energy constraint, node failure, and routing in dynamic underwater environment (Ismail et al. 2022a; Luo et al. 2021; Karim et al. 2021; Jiang 2018; Felemban et al. 2015; Han et al. 2015; Erol-Kantarci et al. 2011a).

- **Low bandwidth**

  UASNs have bandwidth of a few thousand bits per second, resulting in much lower data rates. The major reasons for limited bandwidth are environmental noise and underwater characteristics. Further, bandwidth also depends on transmission or coverage range and frequency used by underwater nodes for acoustic communication (Ghoreyshi et al. 2017). However, data rates can be increased using short-range communication but require more intermediate nodes to deliver the data to the sink.

- **Large propagation delay**

  The speed of the acoustic waves in underwater communication is around 1500 m/s, resulting in a high propagation delay. Apart from depth, underwater propagation delay depends on chemical composition, such as salinity and temperature of water (Burrowes and Khan 2010).

- **High bit error rate**

  The quality of acoustic waves in underwater communication is not predictable and varies frequently. Underwater acoustic communication causes a high bit error rate due to fading and multipath propagation (Burrowes and Khan 2010). The multipath propagation is caused due to the reflection and refraction is caused due to underwater objects (Wati et al. 2019).

- **Energy constraint**

  Energy consumption is another major challenge in UASNs, as replacing or recharging the battery is hard to accomplish. Further, energy consumed in UASNs is more than that of terrestrial sensor networks (Ismail et al. 2022a).

- **Nodes are prone to failures**

  Nodes in the UASNs are prone to failure due to the underwater environment in which it is deployed.

- **Routing**

  Dynamic network topology due to water current and variable link quality between nodes, result in increased complexity of underwater routing (Ghoreyshi et al. 2017).

As shown in Figure 1.1, to forward the data, the routing protocol determines the path from the source to the sink. Underwater routing is challenging as it has to fulfill the requirements, such as end-to-end delay, reliability, etc., with minimum resource consumption to route packets from the source to the sink node. Thus, the following section presents various routing strategies and their pros and cons.

1.2 UNDERWATER ROUTING STRATEGIES

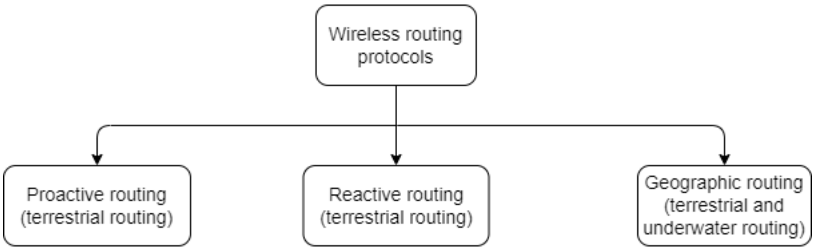
Wireless routing protocols
Proactive routing (terrestrial routing)
Reactive routing (terrestrial routing)
Geographic routing (terrestrial and underwater routing)

Figure 1.2: Types of the wireless routing protocols.

As shown in Figure 1.2, routing in wireless networks is mainly classified into three types: proactive, reactive, and geographic routing. In proactive routing, every node maintains an up-to-date routing path, to every other node in the network, by periodically sending control packets to its neighbors. However, proactive routing requires more control packets to determine the path for the first time. Further, when topology changes due to node failure or mobility, the transmission of control packets is required to maintain an up-to-date path to every node in the network. However, underwater routing does not need to establish the path to every other node. Due to frequent changes in the topology, proactive routing results in transmission of more control packets which causes fast energy dissipation of the node energy. Thus, proactive routing approach may not be suitable for underwater routing (Pompili and Akyildiz 2009; Ghoreyshi et al. 2017).

The next category of routing protocol is reactive routing; the reactive routing protocol is also known as the ‘on-demand’ routing protocol, in which the path to the sink is discovered only when it is needed. The path to the sink is initiated by broadcasting the route request packets in the network. Even though reactive routing is better for dynamic environments, finding the path to other nodes takes a longer time in underwater, due to the large propagation delay of acoustic signals. As a result, reactive routing has a high end-to-end delay and is unsuitable for underwater routing (Akyildiz et al. 2005; Ghoreyshi et al. 2017).

The final category of routing protocols is geographic routing, in which a node selects the next hop based on the position of the neighbors and the sink node. Geographic routing uses the greedy forwarding technique, which is different from end-to-end routing

(Ghoreyshi et al. 2017). In end-to-end routing, the path from source to sink is discovered during path discovery. However, in greedy forwarding, a node only determines the next hop at every node (Kheirabadi and Mohamad 2013). In greedy forwarding, the forwarding node transfers packets to a node that is closer to the sink than itself (Pompili and Melodia 2005; Tzu-Chiang et al. 2013).

A few significant advantages of greedy forwarding are scalability and simplicity. Greedy forwarding provides a relatively better result in comparison to proactive and reactive protocols. A major challenge concerning the greedy forwarding technique is its failure to forward the packets in the event of communication void or local maxima or void node (Tzu-Chiang et al. 2013). In the void node or communication void scenario, the forwarding node cannot find any neighbor with positive advancement toward the sink, even though there exists a valid path from the source/forwarder to the sink (Ghoreyshi et al. 2017). Hence, this thesis focuses on addressing the void node problem in underwater routing. The following section presents the void node and its impact on the performance of the UASNs.

1.3 VOID NODE IN UASNs

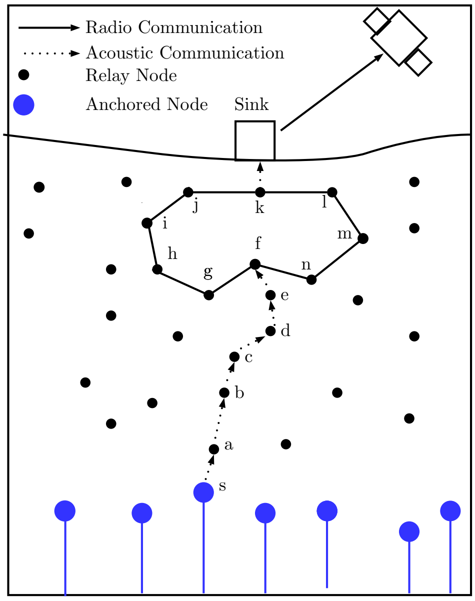

Figure 1.3: Void node scenario in underwater communication.

Figure 1.3 presents typical void node scenario in UASNs. In greedy forwarding, each node transmits packets to a neighboring node with a distance to sink closer than itself. If assumed that a node forwards a packet to a neighboring node that has the highest positive advancement, there may be a situation where a node cannot find a neighboring node with positive advancement. Such nodes are termed void nodes or stuck nodes, or local maxima nodes (Ghoreyshi et al. 2017; Kheirabadi and Mohamad 2013; Tzu-Chiang et al. 2013). The other consequence of a void node is a trap node in the routing. A trap node is a node in which greedy forwarding leads to a void node (Noh et al. 2012). In Figure 1.3, node f is a void node as it does not have any neighbors with positive advancement. Further, nodes s, a, b, c, d, and e are trap nodes because, packets are forwarded by all these nodes leading to a void node f and are stuck at that node. The area covered by f-g-h-i-j-k-l-m-n-f (as shown in Figure 1.3) is called communication void region.

The advancement of the neighbor is calculated as follows. Let i be a node with a packet to forward, j be a neighboring node (j is in the transmission or communication range of i), and the sink S. The advancement of a node j with respect to i, towards the sink S, can be calculated as per Eq. (1.1) (Coutinho et al. 2015a)

$$ADV_{i,j} = D_{i,S} - D_{j,S} \tag{1.1}$$

where $D_{i,S}$ is the distance from node i to the sink S. $ADV_{i,j}$ is the advancement of node j with respect to i towards sink S. In location-based routing, $D_{i,S}$ is the Euclidean distance between node i and S. Suppose $i(i_x, i_y, i_z)$ are the 3D coordinates of i and $S(S_x, S_y, S_z)$ are the 3D coordinates of S. Then Euclidean distance between i and S can be calculated as follows:

$$D_{i,S} = \sqrt{(S_x - i_x)^2 + (S_y - i_y)^2 + (S_z - i_z)^2} \tag{1.2}$$

In depth-based routing (pressure-based routing), $D_{i,S}$ is the depth of node i or the minimal distance between node i and the water surface. One of the nodes with the highest advancement in the candidate forwarding set is selected as the next hop. Candidate forwarding set of node i is calculated as (Noh et al. 2012):

$$C_i = \{j \in N_i : ADV_{i,j} > 0\} \tag{1.3}$$

where N_i is the set of neighboring nodes of i. An empty C_i indicates that node i does not have any neighbor with positive advancement. Thus, node i is referred to as a void node. The distance between nodes is not the only reason for the void node situation, a number of other factors, alone or in combination, are responsible for it. Some reasons for the void node are ship movement between/over the nodes, sparse deployment of nodes, and noise. Packets received by a void node is not forwarded further towards the sink and dropped by the void node if void-handling/recovery/processing mechanisms are not defined. If the void node scenario is not handled properly, the performance of UASNs is affected. Some consequences of the mishandling of void problems are:

- Increased end-to-end delay
- Packet loss
- Increased hop count
- Looping during routing

1.4 VOID-HANDLING IN UASNs

There is a need for a void-handling strategy to overcome the impact of the void node in UASNs. The void-handling strategy needs to be defined in the routing protocol. Routing protocols dealing with void nodes are called void-aware or void-handling routing protocols. Nodes in the void-handling routing protocols can detect their status (void or normal). The source/forwarder node in void-handling routing protocol uses multiple attributes of its neighbor to decide the next hop. Some of the attributes of neighbors that the source/forwarder node uses in determining the next hop are hop count (Nazareth and Chandavarkar 2021b; Guan et al. 2019), residual energy, distance with the neighbor (Javaid et al. 2018), depth (Ghoreyshi et al. 2015), status, and Packet Delivery Probability (PDP) (Alasarpanahi et al. 2020). However, choosing a suitable combination of attributes is a major challenge, since it affects the performance of UASNs directly. Additionally, all the attributes of nodes do not have a similar priority. However, in existing methods, the source/forwarder node, without considering the priority of attributes of neighbors, decides their next hop. To overcome this issue, this research focuses on using Multi-Attribute Decision Making (MADM) to decide the next hop. MADM considers the priority of the attributes and combines the multiple conflicting attributes of neighboring nodes. Further, it computes the suitability of neighboring nodes to become the next hop(s).

1.5 MULTI-ATTRIBUTE DECISION MAKING

Multi-Attribute Decision Making (MADM) is a field of operations research that is used in decision-making where a number of finite alternatives are present (Vinogradova 2019). In the case of the routing in UASNs, the alternatives are the neighboring nodes. The method used in MADM is able to assess multiple conflicting attributes of neighboring nodes and computes their score or rank. The outcome of MADM depends on the attributes of the neighboring nodes and the importance or impact of each attribute in decision-making. Their weight usually determines the importance or impact of attributes. MADM has many methods to compute the score or rank of the neighboring nodes; some of them are the Simple Additive Weighting (SAW) method (Rao 2007), ELECTRE method (Figueira et al. 2013), and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) (Panda and Jagadev 2018; Behzadian et al. 2012).

1.6 MOTIVATION

UASNs are used in various underwater applications like disaster detection, resource detection, surveillance, and underwater exploration. As mentioned in Section 1.3, if void nodes are not handled appropriately, it causes increased end-to-end delay or packet loss or consumes more resources. Especially in the case of some critical applications, like disaster detection and underwater surveillance, void nodes during routing, which compromise the reliability of the application. Thus, it is essential to design void-handling routing protocols that aim at reducing end-to-end delay, high packet delivery ratio, and efficient consumption of resources during routing. Hence, this research focuses on designing a void-aware underwater routing protocol that optimally selects the next hop(s).

1.7 CONTRIBUTIONS OF THIS THESIS

This section highlights the contributions of the thesis.

- **Enhanced-Void-Aware Routing (E-VAR) protocol for UASNs**
  E-VAR is a void avoidance underwater routing protocol. It uses neighboring node's status as primary criterion. Further, Euclidean distance from the neighbors to the sink is a secondary criterion. The E-VAR protocol is compared with the state-of-the-art Interference-aware routing (Intar) (Javaid et al. 2018) and Backward-forwarding (Ghoreyshi et al. 2017), using MATLAB. The results show that the E-VAR protocol avoids the looping during data forwarding and results in a lower hop count to reach the sink.

- **Link Quality-based Routing Protocol (LQRP)**
  LQRP is an underwater routing protocol. It takes the routing decision based on two-hop link quality. Appropriate weights are assigned to the source/forwarder node, to its neighbor and between the neighbor, to its next hop. Further, the cost of the neighbors is determined. Based on cost, the next hop is selected. Using MATLAB, the LQRP is compared with the state-of-the-art Link Quality Estimation based Routing (LQER) (Jiang et al. 2006) protocol. The results show that LQRP outperforms the LQER in terms of Packet Delivery Ratio (PDR).

- **Location-Free Void Avoidance Routing (LFVAR) protocol**
  LFVAR is a void avoidance underwater routing protocol. It uses the status of neighboring nodes as a primary attribute. Additionally, the source/forwarder node

determines the cost of every normal neighboring node, using depth and hop count. Based on the cost of neighbors, the source/forwarder node selects the next hop. LFVAR ensures that void nodes are not a part of the routing path from the source to the sink. The void node finds the recovery path in the higher depth, through a node with the lowest depth difference. The LFVAR protocol is compared with the state-of-the-art Intar using the UnetStack (Chitre 2022b) simulator. The results show that the LFVAR protocol outperforms Intar in terms of average hop count, end-to-end delay, PDR, throughput, and energy consumption.

- **Link and Void Aware Routing (LVAR) protocol**

  LVAR is a void avoidance underwater routing protocol. It uses the status of neighboring nodes as a primary attribute. Additionally, using PDP and the hop count of neighboring nodes, the source/forwarder node determines the next hop. LVAR ensures that the void nodes are not involved during the packet forwarding. The void node finds the recovery path in the higher depth, through a node with the lowest depth difference, with reference to the source. The LVAR protocol is compared with the state-of-the-art Intar, using the UnetStack simulator. The results show that the LVAR protocol outperforms Intar in terms of hop count, end-to-end, and throughput.

- **Cluster-based Multi-Attribute Routing (CMAR) protocol**

  CMAR is a sender-based, opportunistic underwater routing protocol. It uses the status of the normal neighboring nodes, deployed at a lower depth than the source/forwarder node, as the primary attribute. Additionally, CMAR uses advancement and PDP of normal nodes present at a lower depth. CMAR uses Multi-Attribute Decision Making (MADM) approach to prioritize the nodes. Further, using the priority of the nodes and the threshold (Th) number of nodes to be clustered, candidate forwarding set(s) is/are obtained. The source/forwarder node sends its data to its cluster. Upon receiving the data, nodes that are part of the cluster, coordinate the further forwarding, by computing hold time. The CMAR protocol is compared with the state-of-the-art HydroCast (Noh et al. 2015) using MATLAB. The results show that CMAR protocol outperforms HydroCast in terms of the number of forwarding nodes, number of clusters formed, expected packet advancement, number of times void nodes are selected as part of the forwarding set, and transmission reliability.

1.8 ORGANIZATION OF THE THESIS

The remaining part of the thesis is organized as follows:

Chapter 2 elaborates on the classification of existing state-of-the-art void-handling underwater routing protocols, attributes used in determining the next hop, and their issues. Further, discusses open issues associated with void-handling routing protocols.

Chapter 3 elaborates on the research problem and its description. Further, research objectives are presented.

Chapter 4 presents the proposed E-VAR protocol's design, simulation, and evaluation. The E-VAR determines the next hop by using its neighbor's status and Euclidean distance to the sink. E-VAR ensures that void nodes do not participate in data forwarding.

Chapter 5 elaborates on the design, simulation, and evaluation of the proposed LQRP. LQRP uses a two-hop link quality in deciding the next hop.

Chapter 6 presents the detailed design, simulation, and evaluation of the LFVAR protocol. LFVAR generates awareness of the status, depth, and hop count information of nodes in the network. Using this information source/forwarder node determines the next hop, which avoids the void node.

Chapter 7 presents the detailed design, simulation, and evaluation of the LVAR protocol. It uses status, hop count, depth, and PDP as attributes in deciding the next hop. LVAR ensures that the void node does not participate in packet forwarding.

Chapter 8 elaborates on the detailed design, simulation, and evaluation of design and evaluation of the CMAR protocol. CMAR determines Th of nodes in the candidate forwarding set, by applying the MADM approach and novel clustering technique. Additionally, using holding time computation, nodes forward the received data.

Finally, **Chapter 9** summarizes the research contributions of the thesis and directions for future improvements in this area.

1.9 SUMMARY

This chapter presented the UASNs, their applications, and some of their significant challenges. Additionally, this chapter elaborated on various wireless routing strategies for UASNs; void node issues arising during underwater routing and their impact are also explained. Further elaborated, is the need to design a void-aware routing protocol using MADM technique. Finally, this chapter concludes contribution and organization of this research.

The next chapter presents the state-of-the-art void-aware routing protocols that exists in the literature.

CHAPTER 2

LITERATURE SURVEY

Underwater routing is one of the challenging fields in the UASNs due to dynamic network topology. Void nodes in UASNs further increase the complexity of the routing. Based on the capability to handle void nodes, routing protocols are classified into void-ignorance and void-handling or void-aware routing protocols. The void-ignorance routing protocols do not have any mechanism to address void node issues. However, void-handling protocols offer techniques for dealing with void nodes, such as determining a recovery path or avoiding them during routing. This research focuses on designing void-handling routing protocols and this chapter elaborates only on void-handling routing protocols. Void-handling routing protocols are classified into location-based and depth-based underwater routing protocols. The classification of void-handling routing protocols followed by state-of-the-art location-based and depth-based routing protocols are elaborated in Section 2.1, 2.2, and 2.3 respectively. Further, Section 2.4 summarizes and analyzes existing state-of-the-art routing protocols.

2.1 VOID-HANDLING ROUTING PROTOCOLS

This section elaborates on the classification of void-handling underwater routing protocols. The detailed classification of void-handling underwater routing protocols is shown in Figure 2.1. Void-handling routing protocols are mainly classified as follows:

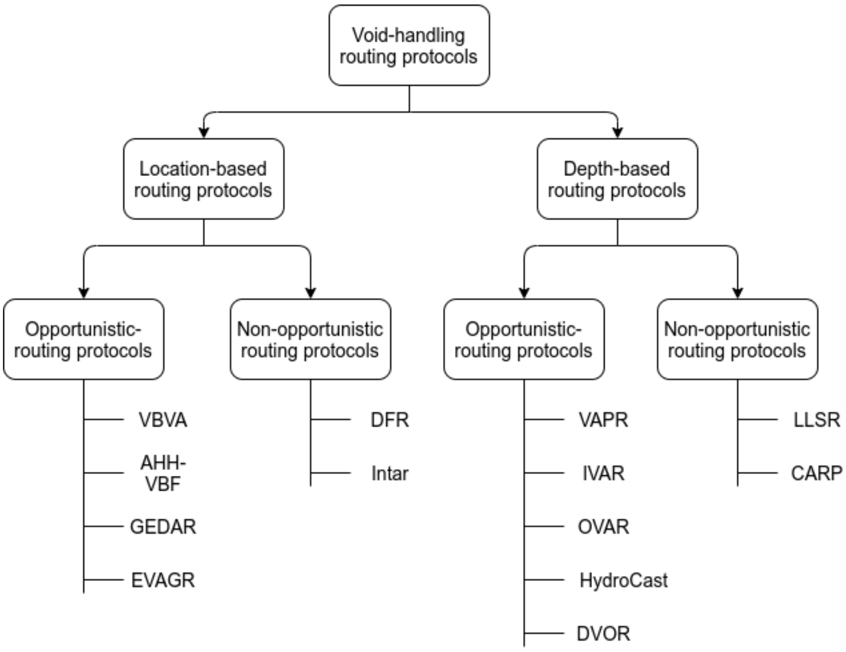
Void-handling routing protocols
Location-based routing protocols
Depth-based routing protocols
Opportunistic- routing protocols
Non-opportunistic routing protocols
Opportunistic- routing protocols
Non-opportunistic routing protocols
VBVA
AHH-VBF
GEDAR
EVAGR
DFR
Intar
VAPR
IVAR
OVAR
HydroCast
DVOR
LLSR
CARP

Figure 2.1: Categorization of void-handling underwater routing protocols.

- **Location-based routing protocols**

  All nodes must know their 3D coordinate (x, y, and z) information. It is achieved using underwater localization algorithms (Ismail et al. 2022b; Su et al. 2020; Erol-Kantarci et al. 2011b, 2010).

- **Depth-based routing protocols**

  These routing methods rely solely on the depth information of nodes to route the data. A pressure gauge is placed on each node to determine its depth in the underwater (Coutinho et al. 2013).

The difference between location and depth-based underwater routing protocols is shown in Table 2.1. The location-based and depth-based routing protocols are further classified as opportunistic routing and non-opportunistic routing protocols, and they are elaborated as follows:

- **Opportunistic routing protocol**

  Opportunistic routing broadcasts data to a set of potential neighbors, referred to as the candidate forwarding set. Nodes that successfully receive the data in the candidate forwarding set, further forward them, based on the priority and

Table 2.1: Comparison between location and depth-based underwater routing protocols.

Location-based routing	**Depth-based routing**
All nodes must know their three-dimensional location information.	All nodes must know their depth information.
Requires to execute localization algorithms to obtain their location.	Depth information is obtained by pressure gauge.
The location of the sink node should be known to all nodes.	No need to know the sink node location. It is assumed that it is present on the water surface.
In some routing protocols, the node must know its neighbor's locations.	In some routing protocols, the node maintains the depth of its neighbors.

coordination among them. Additionally, opportunistic routing achieves better performance due to reduced retransmissions and consumes minimum network resources (Chakchouk 2015; Boukerche and Darehshoorzadeh 2014).

- **Non-opportunistic routing protocol**

  In the non-opportunistic or traditional form of routing in which, the source/forwarder node selects a single next hop as its next hop and transmits the data to it. Each forwarder node continues the same process until data reaches the sink node. The non-opportunistic routing protocol requires more retransmissions, thereby consuming more network resources (Boukerche and Darehshoorzadeh 2014).

2.2 LOCATION-BASED ROUTING PROTOCOLS

This section elaborates on the location-based void-handling routing protocols. One of the first void-handling underwater routing protocols is Vector-Based Void Avoidance (VBVA) (Xie et al. 2009). It is a receiver-based, opportunistic routing protocol and is an extended version of Vector-Based Forwarding (VBF) (Xie et al. 2006). VBVA includes void-handling features. In the absence of the void node, routing in VBVA is the same as in VBF. The data following a path from the source to the sink is referred to as the forwarding vector. The pre-defined threshold distance, surrounding the forwarding vector, is referred to as a pipe. Only nodes within the pipe are eligible to forward the received data packets. When a node receives the data from the source/forwarder, it checks whether it is inside the pipe or not, using the location information of itself, source, and sink. If the node is inside the pipe, it has to wait for a certain period before further forwarding the received packet, referred to as hold time. The hold time is a function of the distance between the node to the forwarding vector. The lower distance between the node and forwarding vector results in a lower hold time. A node that has a lower hold time further forwards the packet first.

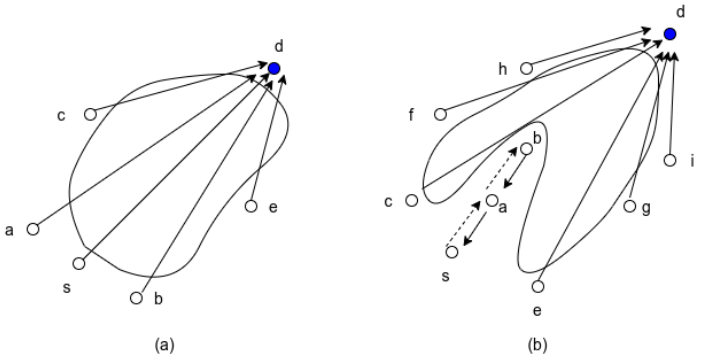

Figure 2.2: Void-handling in VBVA (a) Vector-shift mechanism (b) Back-pressure mechanism.

When a node forwards the packet and does not overhear further forwarding by any other nodes, the node concludes itself as a void node. In such cases, it broadcasts a vector-shift control packet. Upon receiving the packet, nodes outside the pipe define a new forwarding vector between themselves, to the sink and forward the received data packet. As shown in Figure 2.2(a) node s and d are source and the sink node respectively. Node s forwards the packet through the $\overrightarrow{sd}$. Node s waits to overhear further forwarding of the packet, by the nodes present within its pipe, until a pre-defined time expires. If it does not overhear the transmission of the packet within a pre-defined time, it concludes that it is a void node. To recover from the void, it broadcasts a vector-shift control packet. The neighbor nodes, a and b, receive the packet. Further, they change the current forwarding vector to $\overrightarrow{ad}$ and $\overrightarrow{bd}$, respectively. Neighbors a and b forward the data packet received in the vector-shift through their corresponding pipe.

There are situations where, even after vector-shift, the current forwarding node does not overhear further packet forwarding. In such situations, the node broadcasts the Back- Pressure (BP) packet, and upon receiving the packet, nodes shift the forwarding vector, if it has not shifted the vector, for the same packet. This process is repeated until a node forwards the corresponding packet through vector-shift. As shown in Figure 2.2(b), source node s forwards packet through $\overrightarrow{sd}$ to b. Node b does not overhear the further forwarding, and it uses the vector-shift mechanism first. Despite vector-shift, node b does not overhear further forwarding and broadcasts BP packet. Upon receiving the BP packet, node a broadcasts vector-shift and fails to overhear the packet. Thus, node a broadcasts BP packet and is received by node s. The node s performs vector-shift and through the $\overrightarrow{cd}$ and $\overrightarrow{ed}$ packet is further forwarded.

VBVA performs routing of data without holding neighbor node information. Thus VBVA is robust and scalable even in a dynamic environment. However, the void-recovery technique used in the VBVA is complex and time-consuming. Further, duplicate packet transmission and hidden-node terminal are other issues associated with the VBVA. Due to duplicate packet transmission and complex void-recovery technique, the energy efficiency of VBVA is low.

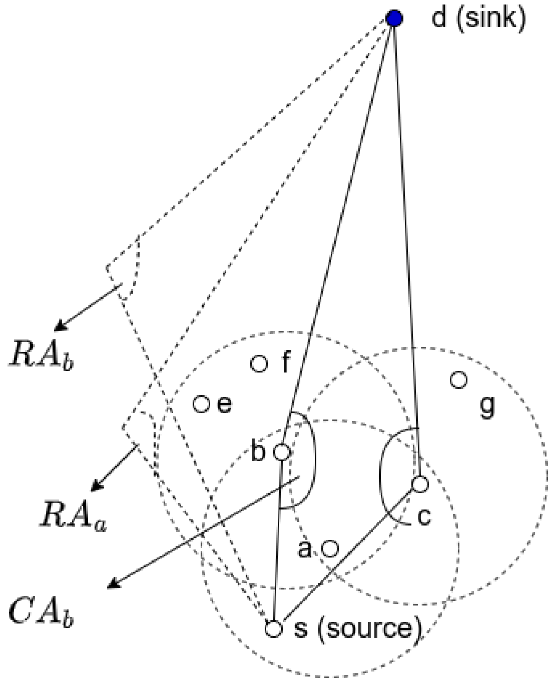

Figure 2.3: Packet forwarding in DFR protocol.

Hidden-node and duplicate packet transmission issues of VBVA are mitigated by Directional Flooding-based Routing (DFR) (Shin et al. 2012). In DFR, forwarding nodes reduce their flooding zone in the case of good link quality with their neighbor. Thereby mitigating hidden-node and duplicate packet transmission issues.

In the DFR protocol, every node knows its location, neighbors, and sink node. DFR uses controlled flooding and a receiver-based routing approach. The flooding zone is defined based on link quality, calculated using the Expected Transmission Count (ETX). DFR uses two types of angles- reference angle and current angle. The reference angle is set by the previous forwarding node a and represented by RA_a as shown in Figure 2.3. CA_b represents the current angle of the node b. The CA_b is determined by the angle formed by the sink, node b, and source. The lines db and bs form the CA_b. When node b receives the packet from a, node b compares its current angle CA_b with the reference angle of a (RA_a), which is specified in the packet. In the case of the CA_b being equal to or greater than the RA_a, the node b concludes itself as eligible to forward the packet. Then node b updates RA_b based on link quality with its neighbors and further, rebroadcasts the packet with updated RA_b.

The void node in DFR occurs in two cases- first: no forwarding nodes in the flooding zone, and second: no neighbors with positive progress towards the sink with respect to the forwarding node. The void node, in the first scenario, occurs due to the continuously increasing reference angle based on good link quality. Further, no node, with a current angle greater than the reference angle, creates a void node. This issue is resolved by defining a threshold limit to the reference angle; it does not increase after a certain reference angle. The void node occurring in the second scenario is because no neighbor is closer to the sink than itself. In such cases, the forwarding node finds a detour path by sending the packet to the neighbor having the smallest current angle. Further, the node acts as the source and forwards the packet.

The continuously increasing link quality in the DFR results in decreased flooding zone and creates a void node. This issue has been resolved by Adaptive Hop by Hop Vector-Based Forwarding (AHH-VBF) protocol (Yu et al. 2015). Every node in AHH-VBF knows the location information of the sink, the sender, the neighbor, and itself. Unlike DFR and VBVA, which use fixed transmission power, AHH-VBF uses adaptive transmission power. Additionally, AHH-VBF changes the direction of the forwarding vector on a hop by a hop basis. Further, in the case of a void scenario, it increases the width of the pipe beyond its pre-defined threshold finding more candidate relay nodes in the sparse region. Additionally, the AHH-VBF is capable of transmission power adaption according to the density of nodes. In the sparse region, transmission power is increased to cover more nodes and decreased to save power. Control packets are exchanged to deal with the change in neighbors and their locations at regular time intervals. The time interval is defined based on the environment in which AHH-VBF is deployed. AHH-VBF does not always guarantee the data delivery to the sink node. AHH-VBF can only handle a small void region in the network (Ghoreyshi et al. 2017).

The duplicate packet transmission and hidden-node terminal issues of VBVA, DFR, and AHH-VBF are overcome by the GEographic and opportunistic routing with Depth Adjustment-based topology control for communication Recovery (GEDAR), is an opportunistic, sender-based routing protocol (Coutinho et al. 2015b). It operates in two phases- enhance beaconing and next node candidate set selection. In the enhanced beaconing phase, regular beacons are broadcasted by the sink and propagated by the nodes in the network. Beacons consist of sequence numbers and coordinates of the node. Upon receiving beacons, nodes update the list of sink nodes they can reach and the coordinates of the neighbors. In the next hop candidate selection phase, the Normalized Advancement (NADV) (Noh et al. 2015; Lee et al. 2005a) of neighbors, present at lower depth, are computed. Further, clusters are formed so that all the nodes in it can hear each other. Additionally, selecting a cluster that has the highest Effective Packet Advancement (EPA) (Zeng et al. 2007) is chosen as a candidate forwarding node. A node can detect its status (void or normal) using the information of the neighboring nodes. When a void node receives data, it handles it using topology control. In topology control, the void node finds a normal node; greedy forwarding is possible with minimal depth displacement.

Even though GEDAR avoids exchanging control messages to recover from the void, the recovery procedure takes time and energy. Thus, GEDAR is not appropriate for delay-sensitive applications.

The high energy cost and time-consuming void recovery in GEDAR are overcome by Interference-aware routing (Intar) protocol (Javaid et al. 2018). Interference-aware routing (Intar) is a sender-based, non-opportunistic routing protocol has two phases - network setup and data forwarding phase (Javaid et al. 2018). The network setup phase develops awareness of neighbor node's attributes such as hop count, number of neighbors, Euclidean distance to the sink, and depth through beacons. Further, using the depth information of the neighbor, it determines the Potential Forwarding Nodes (PFN). PFN are neighboring nodes at a lower depth than the source/forwarder node. During the data forwarding phase, the source/forwarder node computes the cost of each PFN. The PFN with the highest cost is selected as the next hop, and the source/forwarder node forwards the data to the selected next hop. However, in the void node situation, the source/forwarder node does not find any PFN. In such cases, the source/forwarder node finds a neighbor that has the lowest Euclidean distance to the sink in the higher depth. The major issue of the Intar routing protocol is that, it uses a void-recovery mechanism that results in increased end-to-end delay.

Energy-efficient Void Avoidance Geographic Routing (EVAGR) is an opportunistic, sender-based routing protocol(Alasarpanahi et al. 2020). The source/forwarder nodes in EVAGR compute the NADV and weight of neighbors that have positive progress. The attributes used in the weight computation are consumed energy, depth, and link quality of nodes. Further, using NADV and two-hop neighbor information (similar to GEDAR) source/forwarder node initiates the cluster formation process. Additionally, the source/forwarder node computes every cluster's EPA and selects a cluster with the highest EPA. Further, all void nodes in the cluster are removed. Only half of the nodes with the highest weight in the selected cluster are chosen as candidate forwarding nodes.

2.3 DEPTH-BASED ROUTING PROTOCOLS

In depth-based routing protocols, route is determined using the depth (z-coordinate) information of the node itself and the neighboring nodes, without knowing it's full geographical coordinates information. It eliminates the need for costly localization & consumes less energy. This section discusses the depth-based, void-handling protocols.

Void-Aware Pressure Routing (VAPR) is an opportunistic protocol (Noh et al. 2012). The VAPR uses a void-avoidance mechanism to keep data packets away from void nodes during routing. It operates in two phases - enhanced beaconing and opportunistic directional forwarding. The enhanced beaconing involves developing direction trails. The beacons are periodically generated from the sink nodes and comprise attributes like hop count, packet sequence number, forwarding direction, and depth of the node. The sequence number and the hop count information set the node's data forwarding direction. Further, by comparing the direction (up or down) from which it receives the beacon and by extracting the sender's data forwarding direction, it can differentiate normal/void and trap neighbor, thereby avoiding void and trap nodes. However, VAPR avoids the void nodes by holding information up to two hops, resulting in high overhead (Ghoreyshi et al. 2017).

The issue of high overhead in the VAPR is overcome by the Location-free Link State Routing (LLSR) protocol (Barbeau et al. 2015). A node chooses the next hop based on attributes like hop count, path quality, and pressure attributes of neighbors. The source/forwarder node considers the neighbor that has a minimum hop count as its next hop. In the case of a tie, the source/forwarder considers the path quality of the nodes among those neighbors and selects a neighbor that has the highest path quality. If the tie persists, the node chooses a neighbor at the lowest depth as the next hop.

The attributes in LLSR are distributed across the nodes in the network through the periodic beacons. The beacons are originated from the sink and subsequently propagated by the other nodes in the network. However, only one node is selected as the next hop, which may result in a lower successful packet delivery ratio (Ghoreyshi et al. 2017). Further, the hop count value depends on the beacon period. The hop count value may not reflect the up-to-date network topology if the beacon period is too large. If the beacon period is too small, the network is loaded with beacons and consumes more energy. LLSR does not consider the link quality between the source/forwarder and the neighboring node when the next hop is selected. There is a possibility of poor link quality between the source/forwarder and the selected next hop.

Channel-Aware Routing Protocol (CARP) addresses issues of LLSR by selecting the next hop based on goodness and hop count. Goodness is the product of the two-hop link quality of the source/forwarder and the link quality between nodes is measured using Exponentially Weighted Moving Average (EWMA) (Basagni et al. 2015). The hop count in CARP is more sensitive to the link quality than the actual hop distance from the sink. The hop count is computed by maintaining a variable referred to as L_{ratio_h}, where h is the hop count. For every hop count, h maintains the ratio of the number of packets acknowledged to the total number of packets sent by the node. Further, the node considers the minimum hop count such that L_{ratio_h} is above the threshold value. However, CARP is the non-opportunistic routing protocol that chooses only a neighbor as its next hop.

Inherently Void-Avoidance Routing (IVAR) is a receiver-based, opportunistic underwater routing protocol (Ghoreyshi et al. 2015). Only the nodes with a lower hop count than the source/forwarder are eligible to forward the data further; such nodes are referred to as candidate forwarding nodes. Among the candidate forwarding nodes, using hold time computation, coordination between them is achieved. IVAR has two phases - periodic beaconing and routing. Periodic beaconing aims to update the hop count of the nodes in the network. The sink node triggers the beacon and progresses toward the leaf nodes.

The source/forwarder node broadcasts the data packet during the routing phase. The data packet consists of data, depth, node id, and hop count of the source/forwarder node, and packet sequence number. When neighboring nodes receive the data packet, they check whether it is a duplicate or a new packet. In the case of a duplicate packet, the neighboring node discards the packet. Otherwise, the hop count of the neighboring node is compared with that of the source/forwarder. If the hop count of the neighboring node is smaller than that of the source/forwarder, it is considered as part of candidate forwarding nodes, to forward the data packet further. Moreover, all candidate forwarding nodes compute their hold time.

A candidate node with the lowest hold time forwards the data packet first. Other nodes in its proximity overhear the transmission and suppress the transmission. Hold time is computed using attributes such as hop count, depth difference and relative distance between source/forwarder and candidate node.

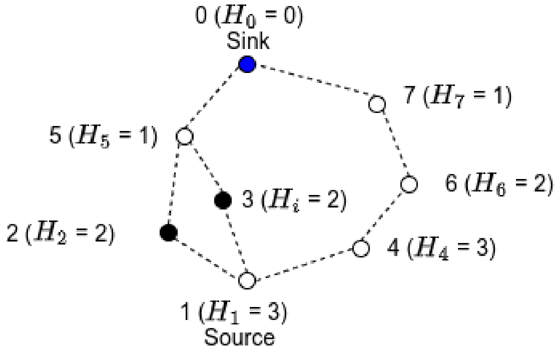

Figure 2.4: Candidate node selection in the IVAR.

Figure 2.4 shows the operation of the IVAR protocol in detail. Node with id 1 and 0 are the source and the sink, respectively. In the beaconing phase, beacons are initialized by the sink node. Also, beacon is received by the nodes 5 and 7, which accordingly update their hop count H_5 = 1 & H_7 =1, respectively. In addition, nodes 5 and 7 propagate the beacon, and the process continues till the beacons are propagated to the leaf nodes. Node 1 is the source node that has data to forward. It broadcasts the data packet consisting of the data, its depth, node id, hop count, and sequence number. Neighboring nodes 2, 3, and 4 receive the data packet. Nodes 2 and 3 are candidate forwarding nodes, since their hop count (H_2 = 2 & H_3 = 2) is less than that of source node 1 (H_1 = 3). Moreover, they compute their hold time. The node with the lowest hold time, whose timer expires first, forwards the packet. However, node 4 is not a candidate forwarding node, since its hop count (H_4 = 3) is not less than that of the source(H_1 = 3). Thus, node 4 drops the packet.

Even though IVAR improves transmission reliability by using an opportunistic routing technique, there are chances of duplicate packet transmission and hidden-node problems in case the candidate nodes are out of communication range of each other (Ghoreyshi et al. 2017).

The issues of IVAR are overcome by the HydroCast (Noh et al. 2015) selecting candidate forwarding nodes in the vicinity of each other. It is a sender-based, opportunistic underwater routing protocol. HydroCast uses multiple attributes in selecting candidate forwarding nodes. The attributes used in the HydroCast protocol are advancement and the Packet Delivery probability (PDP) of neighboring nodes. The neighboring node information is obtained by the beaconing process. The HydroCast is an opportunistic routing protocol; it has to determine its candidate forwarding set.

Due to channel fading, the longer the distance, the higher the chances of packet loss. Thus, Hydrocast considers normalization of advancement of neighboring nodes using Normalized Advancement (NADV) (Noh et al. 2015; Lee et al. 2005a). The source/-forwarder node i computes the NADV of each neighbor j, present at the lower depth. Further, source/forwarder node i begins the clustering process from the neighbor node j that has the highest NADV. The source/forwarder i expands the cluster by including common entries in its neighbor list and neighbors at a distance $(\beta \times Range)$ of the node that has the highest NADV. The value of β is selected as 0.5, hence all nodes clustered by node i can hear each other; thus, duplicate transmissions and hidden-node problems are mitigated.

The clustering process continues with the highest NADV among the remaining neighbors of i, which are not included in any cluster. The clustering process continues until all neighbors node i is included in one of the clusters. Thus, the number of clusters is generated at the end of the clustering phase. Further, Effective Packet Advancement (EPA) (Zeng et al. 2007) of each cluster is computed. The cluster that has the highest EPA is the selected candidate forwarding set. When the source/forwarer node i has data to forward, it sends it to nodes of the selected cluster by including node id's in the data packet. Upon receiving the data, nodes compute their hold time.

The hold time is designed so that a node with the highest progress (or higher priority) with reference to the source/forwarder i has the lowest hold time. The lower hold time expires first; thus, a node with higher progress further forwards the packet, and other nodes in its vicinity overhear the transmission and suppress the transmission of the same data packet. The HydroCast protocol uses the void-recovery mechanism, resulting void node that can also be a part of the routing, resulting in a higher end-to-end delay.

Opportunistic Void Avoidance Routing (OVAR) is a sender-based opportunistic routing protocol that selects neighbors with a lower hop count than the source/forwarder to avoid void nodes (Ghoreyshi et al. 2016). It uses one-hop neighbor information to be aware of neighbor node information. The forwarding set is selected by computing EPA. The number of nodes in the forwarding set is adjusted to balance the trade-off between reliability and energy.

Distance-Vector-based Opportunistic Routing (DVOR) is a receiver-based, opportunistic routing protocol (Guan et al. 2019). DVOR uses hop count as a distance-vector, which is propagated through query packets from the sink toward the nodes deployed at higher depths in the underwater network. Every node (except the sink) in the network buffers the hop count with the corresponding query id, by which its distance-vector is updated. The query packets consist of hop count and query id, which represents the time stamp of that packet generation at the sink. When the query packet arrives at the node, it compares the query id stored in its buffer with the new query packet. If the query id stored in the buffer is smaller than an arrived packet, the node updates its hop count, corresponding to arrived query packet, and rebroadcasts the query packet with the updated hop count. At every node, arrived old query packets are eliminated. However, when query packets with similar id arrive at the node, the lowest hop count is considered, and the hop count is updated accordingly.

The Data packet in DVOR consists of the source id, the hop count of the previous forwarder, and data. When the data packet arrives at the nodes, only those nodes with a hop count lower than the last forwarder are eligible to forward the data further. Moreover, eligible forwarding nodes compute their hold time. The hold time depends on the random number between 0 and the back-off window. The major limitations of the DVOR are duplicate packet transmission and hidden-node problems.

As per this research, no underwater routing protocols, that use Multi-Attribute Decision Making (MADM) and weights of the individual attributes are proposed. Therefore, this research focuses on designing a void-aware routing protocol that uses suitable weight computation and the MADM approach to select the nodes in the candidate forwarding set.

2.4 COMPARATIVE ANALYSIS OF VOID-HANDLING ROUTING PROTOCOLS

Table 2.2: Analysis of void-handling routing protocols.

Sl. No.	Protocol	Attributes used	Void-handling strategy	End-to-end delay	Energy efficiency
1	VBVA	Vector between source to the sink, Fixed pipe width	Reactive	High	Low
2	DFR	Link quality, Reference/Current angle	Hybrid	Low	High
3	AHH-VBF	Vector between node with neighbor, Variable pipe width	Preventive	Low	High
4	GEDAR	Advancement, PDP	Preventive	Low	Low
5	Intar	Distance to PFN, hop count, and number of neighbors to PFNs	Reactive	High	Medium
6	EVAGR	Advancement, PDP, Status	Preventive	Low	High
7	VAPR	Two-hop forwarding directing	Preventive	Low	High
8	LLSR	Hop count, Path quality, Depth of neighbor	Preventive	Low	Medium
9	CARP	Two-hop link quality, Hop count (sensitive to link quality)	Preventive	Low	High
10	IVAR	Hop count, Advancement	Preventive	Low	Medium
11	OVAR	Hop count, Advancement, PDP, Energy consumption	Preventive	Low	High
12	HydroCast	Advancement, PDP	Proactive	Medium	Medium
13	DVOR	Hop count	Preventive	Low	High

This section compares various void-handling routing protocols in terms of parameters such as attributes used in route selection, void-handling strategy, end-to-end delay, and energy efficiency of routing in the presence of a void node. The routing protocol with reference to those parameters is highlighted in Table 2.2 and analyzed as follows:

A. Attributes used by the routing protocols

Attributes used in selecting the next hop influence the performance of the routing protocol. In location-based protocols, the forwarding node uses 3D coordinates information of itself. Additionally, VBVA and DFR protocols use the location information of the source and the sink node. GEDAR uses the location information of neighboring nodes and the sink. Intar uses 3D location information of the sink. Protocols such as EVAGR and HydroCast use advancement and PDP as their attributes. VAPR, IVAR, OVAR, CARP, and DVOR use hop count as one of the attributes to select the next hop(s).

B. The void-handling strategy used

Some routing protocols use a reactive void-handling strategy. When a packet is stuck at a void node, the protocol tries to find a recovery path so that forwarding in the normal mode of operation can resume. In the proactive void-handling technique, in advance, the routing protocol finds a void-recovery path. In preventive void-handling technique, care is taken that the packet does not get stuck at a void node; no need for recovery mechanisms.

C. End-to-end delay

It is the time taken by the packet to reach from the source to the sink, while in the case of void node. The VBVA protocol requires high end-to-end delay, due to the complex technique followed to recover from void nodes. However, DFR results in lower end-to-end delay, due to the preventive void-handling technique, and sometimes there may be more delay in recovering from the void nodes. AHH-VBF and GEDAR result in a low end-to-end delay as they use a preventive void-handling strategy. Intar has a higher end-to-end delay due to the usage of void-recovery technique. EVAGR, VAPR, LLSR, CARP, IVAR, OVAR, and DVOR result in lower end-to-end delay as they use the preventive void-handling technique. However, the HydroCast has medium end-to-end delay as it uses a proactive void-handling technique.

D. Energy efficiency

VBVA is not energy efficient due to the requirement of a large number of control packet exchanges in the scenarios of the void. The DFR protocol is energy efficient due to the controlled flooding of packets and simple void recovery mechanism. Due to adaptive pipe width and transmission power, AHH-VBF achieves better energy efficiency. However, GEDAR is not energy efficient due to the need for beacon transmission and depth adjustment of the void nodes. The protocols EVAGR, VAPR, and OVAR, achieve higher energy efficiency due to avoiding the hidden-node issue and duplicate packet transmissions. The LLSR achieves medium energy efficiency due to the high signaling overhead. Due to the backward transmission of packets in the case of void-handling scenario, HydroCast is medium efficient. IVAR uses opportunistic routing and does not prevent hidden-node and duplicate packets; it is medium energy efficient.

2.5 SUMMARY

This chapter has discussed the categories of void-handling underwater routing protocols. Moreover, in the void-handling protocols, various location-based and depth-based protocols with their features, advantages, and disadvantages are elaborated. The future direction of the research is also discussed.

The next chapter discusses the problem description and objectives of the research.

CHAPTER 3

PROBLEM STATEMENT AND OBJECTIVES

This chapter focuses on problem statement and objectives of this research work. The problem statement and objectives of this research are elaborated in section 3.1 and 3.2, respectively.

3.1 PROBLEM STATEMENT

There are many void-handling underwater routing protocols, such as VBVA, DFR, AHH-VBF, GEDAR, Intar, EVAGR, VAPR, IVAR, OVAR, HydroCast, LLSR, CARP, DVOR, etc. These routing protocols use one or more attributes of neighbors like location, the Euclidean distance between the neighbor to the sink, depth, hop count, link quality of neighboring nodes, status, etc. However, in the literature, the priorities of the individual attributes are not considered in any of the routing protocols. Therefore, it is necessary to select appropriate combinations of attributes of neighbors and combine them so that neighboring nodes can be evaluated to determine the next hop(s). It is also necessary to consider Multiple Attribute Decision Making (MADM) to select the next hop(s). In MADM, priorities for selected attributes are assigned, and the suitability of neighboring nodes is evaluated to select the next hop(s). Further, void nodes, during routing, pose a major impact on the performance of the routing. It is necessary to avoid void nodes during routing. Accordingly, the problem statement of this research is formulated as follows:

Design and Development of Multi-Attribute, Void-Aware Routing Protocol for Underwater Acoustic Sensor Networks.

3.2 RESEARCH OBJECTIVES

In the direction of the problem statement of this research, the following objectives are identified:

- **Objective-1: Identification of attributes for the selection of next hop(s) or forwarding node(s)**
  Attributes of neighboring nodes play a vital role in selecting the next hop(s) or forwarding node(s). As existing void-handling routing protocols use different combinations of attributes to choose the next hop(s), this objective aims to identify the more suitable combination of attributes for designing an optimum void-aware routing protocol.

- **Objective-2: Designing a void-aware underwater routing algorithm using MADM**
  The existing void-aware routing protocols use different combinations of attributes without fixing the priority to select the next hop(s). However, this objective aims at using the MADM approach, to evaluate neighbors using most suitable combination of attributes, to design an optimum void-aware routing algorithm for UASNs.

- **Objective-3: Deployment, simulation, and evaluation of the proposed algorithm in the industry-standard simulator**
  This objective aims at deploying and simulating a designed void-aware underwater routing protocol. The simulation has to be carried out using an industry-standard simulator, such as MATLAB and UnetStack. Further, the performance of the developed protocol has to be validated in terms of different performance metrics, such as the number of nodes reachable to the sink, number of nodes not reachable to the sink due to loops, PDR, hop count, propagation distance from

source to the sink, throughput, number of the clusters formed, number of times void node is part of the routing, etc., with the state-of-the-art routing protocols.

3.3 SUMMARY

This chapter elaborates on the problem statement of the research. Moreover, in line with the problem statement, the research objectives are discussed in detail.

The next chapter presents one of the research contributions of this research - the design and evaluation of the Enhanced-Void-Aware Routing (E-VAR) protocol.

CHAPTER 4

DESIGN AND EVALUATION OF ENHANCED-VOID-AWARE ROUTING (E-VAR) PROTOCOL

This chapter elaborates on the Enhanced-Void-Aware Routing (E-VAR) protocol for UASNs published in IET's Wireless Sensor Systems (Nazareth and Chandavarkar 2019). In E-VAR, every node (except the sink) uses its neighboring node's status (void or normal) as a primary attribute, followed by the distance between its neighbor to the sink as a secondary attribute in selecting its next hop. The detailed design and simulation of E-VAR, followed by its comparison with state-of-the-art Backward-forwarding (Ghoreyshi et al. 2017) and Intar (Javaid et al. 2018), in terms of the number of nodes reachable to the sink, the number of transmissions that failed due to looping, the average hop count and distance to reach the sink are presented in Section 4.1 and Section 4.2 respectively.

4.1 DESIGN OF THE E-VAR PROTOCOL

This section discusses the design of the E-VAR protocol in detail. The next hop selection in the E-VAR has the following two phases:

- **Phase-I: Status (void or normal) identification by a node**

  During this phase, every node (except the sink) identifies, by itself, whether it is void or normal and sends its status to all its neighbors through beacons, at regular intervals (Refer Section 4.1.1).

- **Phase-II: Identifying the next hop**

  During this phase, source/forwarder nodes identify the next hop by excluding void node(s) among its neighbors (Refer Section 4.1.2).

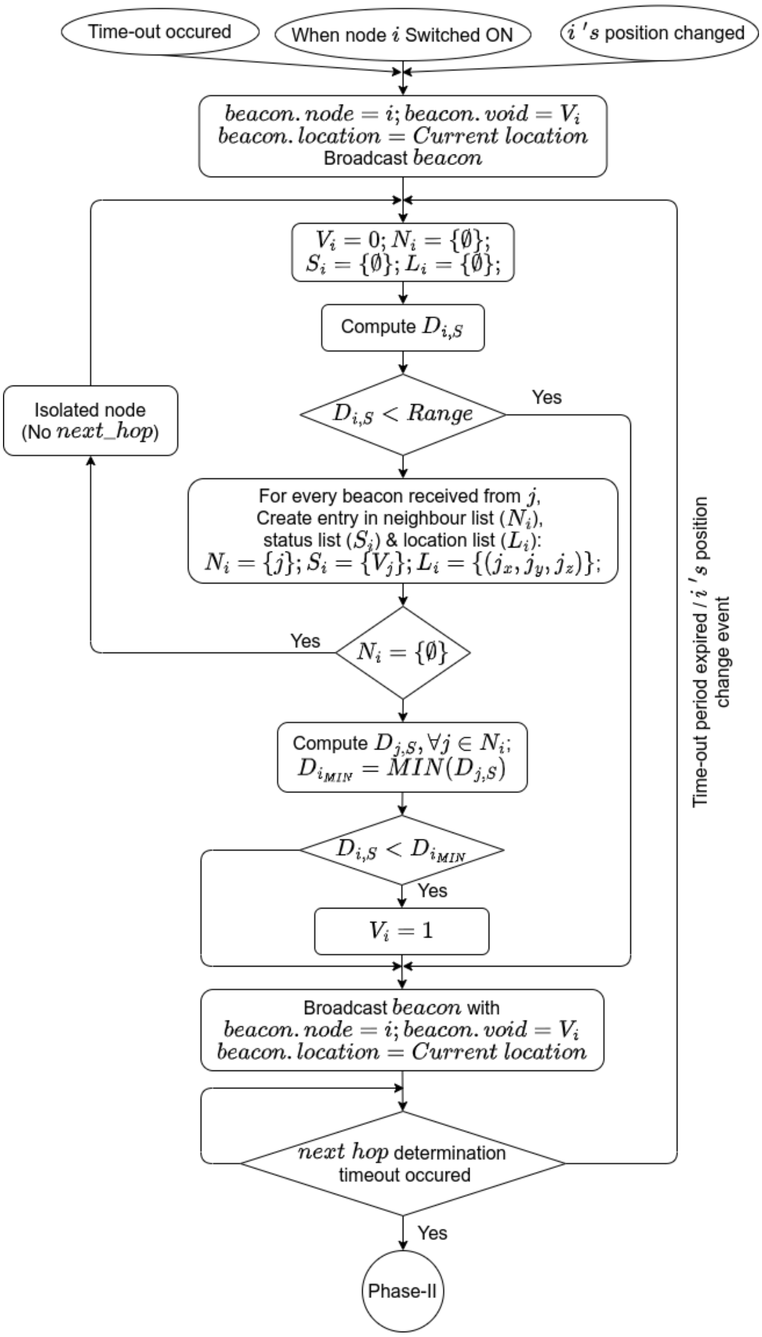
Time-out occured
When node i Switched ON
i 's position changed
$beacon.node = i; beacon.void = V_i$
$beacon.location = Current\ location$
Broadcast $beacon$
$V_i = 0; N_i = \{\emptyset\};$
$S_i = \{\emptyset\}; L_i = \{\emptyset\};$
Compute $D_{i,S}$
$D_{i,S} < Range$
Yes
Isolated node
(No $next\_hop$)
For every beacon received from j,
Create entry in neighbour list (N_i),
status list (S_j) & location list (L_i):
$N_i = \{j\}; S_i = \{V_j\}; L_i = \{(j_x, j_y, j_z)\};$
Yes
$N_i = \{\emptyset\}$
Compute $D_{j,S}, \forall j \in N_i;$
$D_{i_{MIN}} = MIN(D_{j,S})$
$D_{i,S} < D_{i_{MIN}}$
Yes
$V_i = 1$
Broadcast $beacon$ with
$beacon.node = i; beacon.void = V_i$
$beacon.location = Current\ location$
$next\ hop$ determination
timeout occured
Yes
Phase-II
Time-out period expired / i 's position
change event

Figure 4.1: Phase-I: Status (void or normal) identification by a node in E-VAR.

4.1.1 Phase-I: Status (void or normal) identification by a node

Figure 4.1 presents a flowchart for a node to identify itself as void or normal. Every deployed node (except the sink) executes this mechanism under scenarios such as switching on, changing position, or time-out (regular interval). As shown in Figure 4.1, every deployed node i initializes status (V_i) to 0, indicating it, as a normal, and neighbour list (N_i) to $\{\emptyset\}$ indicating that no neighbors exist. Further, i uses S_i to indicate the status of neighbors $j \in N_i$ and initializes to S_i = $\{\emptyset\}$. The L_i indicates the three-dimensional location of neighbors $j \in N_i$ and initializes to $L_i = \{\emptyset\}$ by the node i.

$$D_{i,S} = \sqrt{(i_x - S_x)^2 + (i_y - S_y)^2 + (i_z - S_z)^2} \tag{4.1}$$

As a first step, node i computes its own Euclidean distance $D_{i,S}$ with the sink S, using 3-dimensional positions (i_x, i_y, i_z) and (S_x, S_y, S_z) of itself and the sink respectively (Eq. (4.1)). In case $D_{i,S}$ is less than the coverage range ($Range$), it indicates that node i is directly reachable to the sink S through single-hop communication, and it is normal node ($V_i = 0$). In the event of empty neighbour list ($N_i = \{\emptyset\}$), node i concludes itself as an isolated node. However, in the case of a non-empty neighbor list, node i computes the Euclidean distance $D_{j,S}$ of all its neighbors j and the sink S using L_i. Further, it identifies a neighbor with minimum distance (MIN($D_{j,S}$)) to the sink. Also, comparing $D_{i,S}$ and MIN($D_{j,S}$) of a node, i concludes itself as a void (V_i = 1) if it is less, otherwise i remains a normal node (V_i = 0). Additionally, node i broadcasts its id, status, and location information using beacons $beacon.node$, $beacon.void$, and $beacon.location$ respectively. The above steps (Figure 4.1) are repeated by a node i whenever (i_x, i_y, i_z) changes or at time-out.

Figure 4.2 presents an example of status detection by node 3. As shown in Figure 4.2(a), node 3 has neighbors 1, 2, 4 and 5. Node 1 and 2 are at lower Euclidean distance to the sink ($D_{1,0} = 80$ & $D_{2,0} = 60$) than that of node 3 ($D_{3,0} = 120$); node 3 concludes itself as normal ($V_3 = 0$). However, in Figure 4.2(b) node 3 has two neighbor nodes 4 and 5, but at higher Euclidean distance to the sink ($D_{4,0} = 160$ & $D_{5,0} = 140$) than that of node 3 ($D_{3,0} = 120$). Thus, status of node 3 is void ($V_3 = 1$) (Refer Figure 4.1).

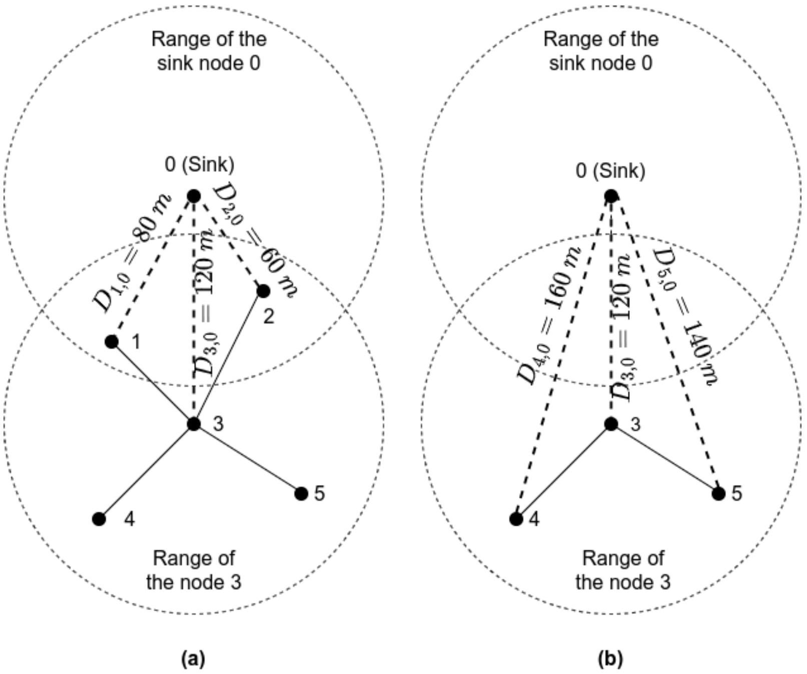

Figure 4.2: Status (void or normal) detection by node 3, (a) $V_3 = 0$ (b) $V_3 = 1$.

4.1.2 Phase-II: Identifying a next hop

Figure 4.3 presents the steps involved in Phase-II to determine the next hop. In Phase-II, a node can undergo different scenarios depending on the status of neighbors, which is collected through the beacons, and the selection of the next hop varies accordingly. As shown in Figure 4.3, list of normal nodes (P_i) and void nodes (W_i) are initialized to $\{\emptyset\}$. Further, the conditional statements present the different scenarios of a node in E-VAR. Selection of the next hop in E-VAR, for different scenarios of a node i, is explained as follows:

- **One of the neighbors is the sink node** ($Sink \in N_i$)

  The node i selects the sink as its next hop, irrespective of the advancement of other neighbors.

- **A node that has only void neighbors** ($P_i = \{\emptyset\}$ and $W_i \neq \{\emptyset\}$)

  The node i selects a neighbor that has the highest advancement towards the sink among W_i.

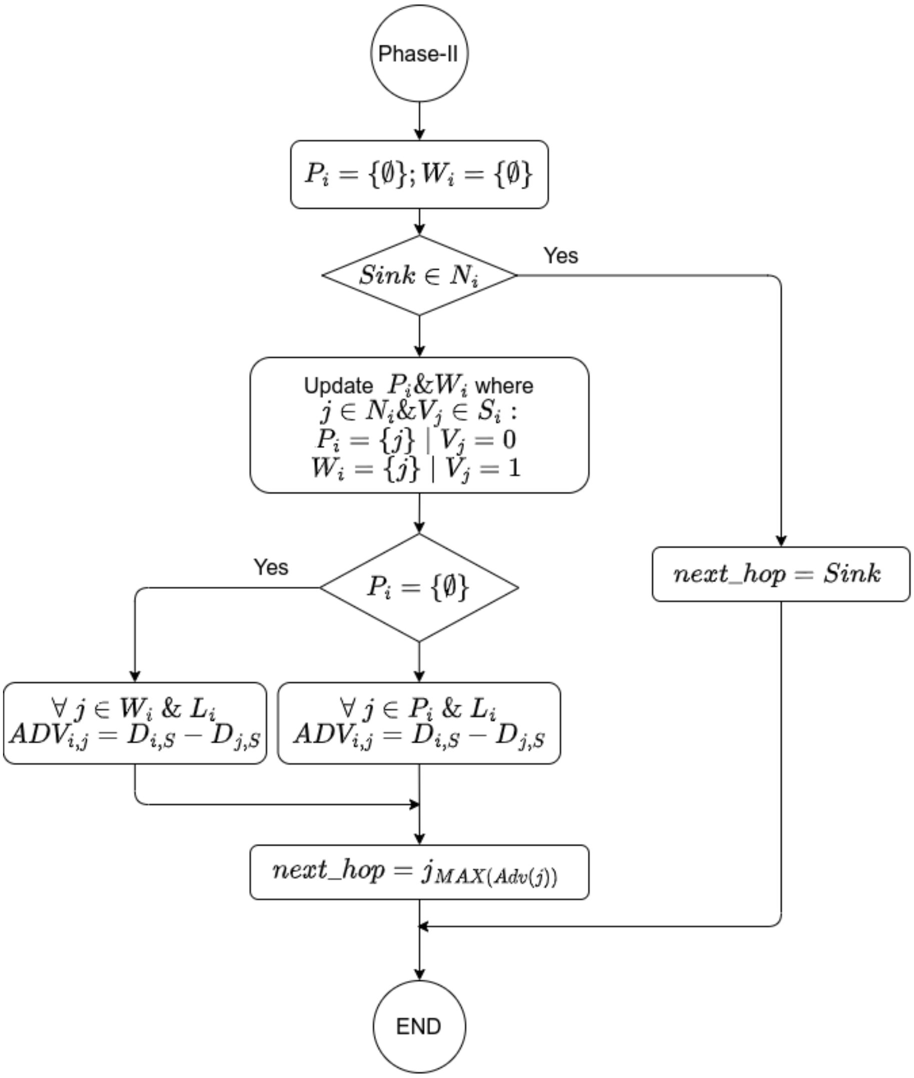
Phase-II
$P_i = \{\emptyset\}; W_i = \{\emptyset\}$
$Sink \in N_i$
Yes
Update $P_i \& W_i$ where $j \in N_i \& V_j \in S_i$: $P_i = \{j\} \mid V_j = 0$ $W_i = \{j\} \mid V_j = 1$
Yes
$P_i = \{\emptyset\}$
$next\_hop = Sink$
$\forall j \in W_i$ & L_i $ADV_{i,j} = D_{i,S} - D_{j,S}$
$\forall j \in P_i$ & L_i $ADV_{i,j} = D_{i,S} - D_{j,S}$
$next\_hop = j_{MAX(Adv(j))}$
END

Figure 4.3: Phase-II: Identifying the next hop in E-VAR.

- **A node that has both void and normal neighbors** ($P_i \neq \{\emptyset\}$ and $W_i \neq \{\emptyset\}$) The node i selects a neighbor that has the highest advancement towards the sink among P_i. This prevents the trap node and looping of the packets.

In conclusion, the E-VAR considers all the different possible scenarios of a node in UASNs and proposes an optimum path to the sink (except in an isolated node).

4.2 SIMULATION AND EVALUATION OF THE E-VAR PROTOCOL

This section discusses the simulation and evaluation of E-VAR in detail. Simulation of E-VAR is carried out using MATLAB. Further, its performance is compared with the state-of-the-art Backward-forwarding (Ghoreyshi et al. 2017) and Intar (Javaid et al. 2018) routing protocol. The performance evaluation of Backward-forwarding, Intar, and E-VAR are carried out for two different cases as follows:

- **Case-I: Manual deployment**

  The objective of this deployment is twofold; one is to present the different possible scenarios of a node in E-VAR, with the selection of the next hop, comparing it to the Backward-forwarding and Intar routing protocol. Second, to present the working of E-VAR in comparison with Backward-forwarding and Intar routing protocol at a smaller scale. The results and analysis of the manual deployment are explained in Section 4.2.1.

- **Case-II: Random deployment**

  The objective of this deployment is to perform an exhaustive simulation through the random deployment of nodes. In this simulation, Backward-forwarding, Intar, and E-VAR are compared in terms of the number of nodes reachable to the sink, the number of transmissions that failed due to looping, the average hop count, and distance. Here, the initial simulation is carried out for 10 randomly deployed nodes (node 0 as a sink) with the increment of 10 more nodes during the successive simulations, by retaining the locations of previous nodes intact. The other parameters of the MATLAB simulation in Case-II are same as in Case-I. The results and analysis of the random deployment are explained in Section 4.2.2.

4.2.1 Case-I: Manual deployment

In manual deployment, two scenarios of 10 nodes (node 0 as a sink) with 3-dimensional location (x, y, z) are manually fixed in 500 X 500 X 500 units area with the coverage range of nodes as 150 units. The first scenario presents the eradication of looping in E-VAR in comparison with Backward-forwarding and Intar. Moreover, to reach the sink in E-VAR, the second scenario presents a lower hop count than the other two.

A. Scenario-I: E-VAR eradicates looping during the selection of a next hop

The scenario 1 elaborates on the next hop selection, with looping in Backward-forwarding, Intar, and E-VAR, concerning a similar topology (Figure 4.4 through 4.6). In Figure 4.4 through 4.6, nodes 1 and 4 are void (V_i = 1) because either their distance to the sink ($D_{1,0}$ = 330 units and $D_{4,0}$ = 228 units) is higher than the coverage area (150 units) or their neighbor distance to the sink ($D_{2,0}$ = 359 units of node 1 and $D_{5,0}$ = 240 units of node 4) is higher than the nodes 1 or 4 itself.

A.1. Backward-forwarding

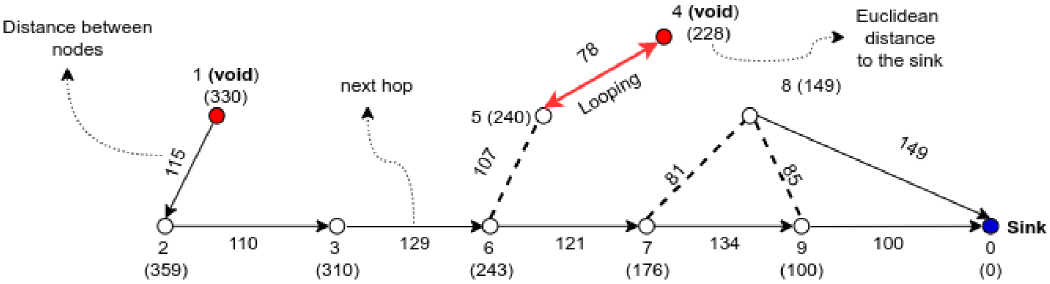

Figure 4.4: Looping scenario in Backward-forwarding.

In Backward-forwarding, a neighboring node that has the highest advancement (closer to the sink) is selected as the next hop. For example, consider the next hop selection at nodes 2 and 5 with the candidate next hops {1, 3} and {4, 6} respectively, as shown in Figure 4.4.

Table 4.1: Advancement of neighbors of node 2 and 5 in Figure 4.4.

Node	**Status of the node**	**Euclidean distance to the sink**	**Advancement**
Node 2 (Euclidean distance to the sink = 359)			
1	Void	330	359 - 330 = 29
3	Normal	310	359 - 310 = 49
Node 5 (Euclidean distance to the sink = 240)			
4	Void	228	240 - 228 = 12
6	Normal	243	240 - 243 = -3

Table 4.1 presents the calculation of advancement of neighbors of nodes 2 and 5. Accordingly, the node 3 and 4 (void) are selected as the next hop at nodes 2 and 5,

respectively. However, Backward-forwarding works in recovery mode, at void nodes 1 and 4 (Figure 4.4), to identify a recovery path (identifying negative advancement node) and reach the sink using its neighbors. Accordingly, node 5 is the only neighbor to void node 4, which results in node 5 as the next hop to 4. Similarly, node 2 becomes the next hop to node 1. In Figure 4.4, forwarding of packets to node 5 by 4 and vice versa, resulting in looping of packets in Backward-forwarding. The infinite looping of packets between nodes 4 and 5 eventually results in a drop of packets. Additionally, a significant amount of node energy is wasted because of looping, resulting in a reduced lifetime of a node.

A.2. Intar

In Intar, a source/forwarder initially identifies its neighbors present at a lower depth (closer to the sink node); such a neighbor is called a Potential Forwarder Nodes (PFN). One among the PFNs is selected as the next hop based on the cost ($C_{i,j}$) (Eq. (4.2)) (Javaid et al. 2018).

$$C_{i,j} = \frac{D_{i,j}}{H_j \times |N_j|} \tag{4.2}$$

where $C_{i,j} \rightarrow$ cost of the PFN j with reference to source/forwarder i, $D_{i,j} \rightarrow$ Euclidean distance, $H_j \rightarrow$ hop count of PFN j, $|N_j| \rightarrow$ the total number of neighbors to node j. A PFN that has the highest cost is selected as the next hop. In case of non-availability of PFN, Intar selects a neighbor with the minimum distance to the sink as the next hop, among the non-PFNs, without applying Eq. (4.2).

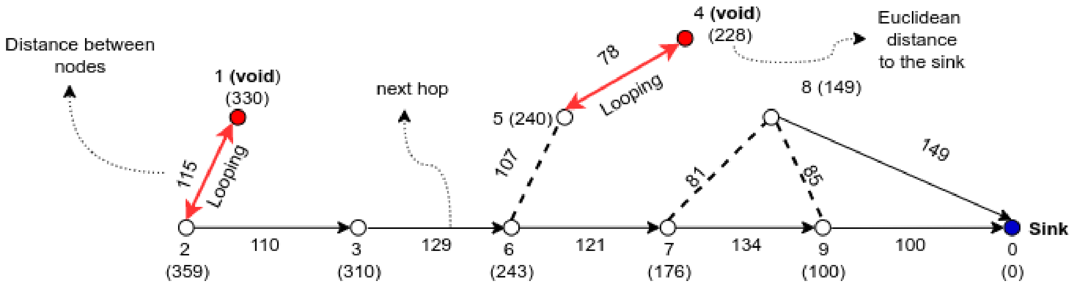

Figure 4.5: Looping scenario in Intar.

Table 4.2: Cost of PFNs of node 2.

Node (j)	**Number of neighbors** ($\lvert N_j \rvert$)	**Distance from node 2** ($D_{2,j}$)	**Hop distance from the sink** (H_j)	**Cost** ($C_{2,j}$)
1	1	115	6	$\frac{115}{1 \times 6} = 19.16$
3	2	110	4	$\frac{110}{2 \times 4} = 13.75$

As shown in Figure 4.5, nodes 1 and 3 are the candidate PFN at node 2. As shown in Table 4.2, $C_{2,1}$ and $C_{2,3}$ (Eq. (4.2)) are calculated, resulting in node 1 as a next hop. However, at node 1 (void node), due to the non-availability of PFN, node 2 (non-PFN) is selected as the next hop resulting in the looping of packets between nodes 1 and 2. A similar situation is observed between nodes 4 and 5. Additionally, looping between $\{1, 2\}$ and $\{4, 5\}$ in Intar results in nodes 1, 2, 4, and 5 being unreachable to the sink.

A.3. E-VAR

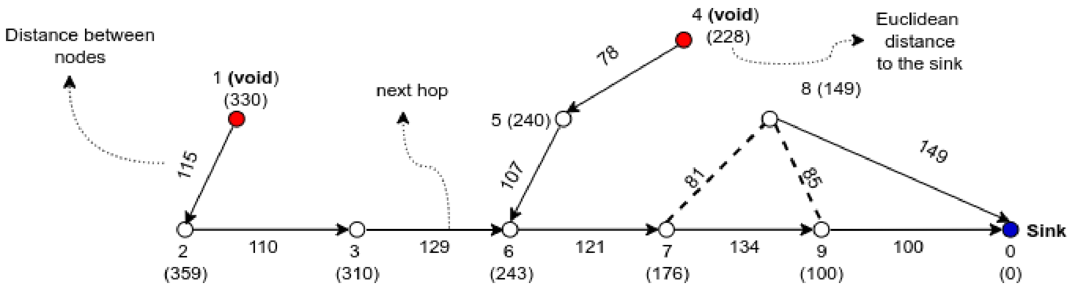

Figure 4.6: Scenario showing avoidance of looping in E-VAR.

The problem of looping and nodes unreachable to the sink shown in Figure 4.4 and 4.5, is resolved in E-VAR due to the void awareness among the other nodes as shown in Figure 4.6. E-VAR follows a similar approach of Backward-forwarding, shown in Table 4.1, in selecting the next hop and excluding all nodes with the status $V_i = 1$. E-VAR also follows the recovery mode of Backward-forwarding at the void nodes 1 and 4 to select the next hop.

Accordingly, the selection of next hop as 4 (void) in Backward-forwarding (Figure 4.4) and Intar (Figure 4.5) by the node 5 is resolved to node 6 (normal) in E-VAR

(Figure 4.6). Similarly, the selection of next hop as 1 in Intar (Figure 4.5) by the node 2 is resolved to node 3 (normal) in E-VAR (Figure 4.6).

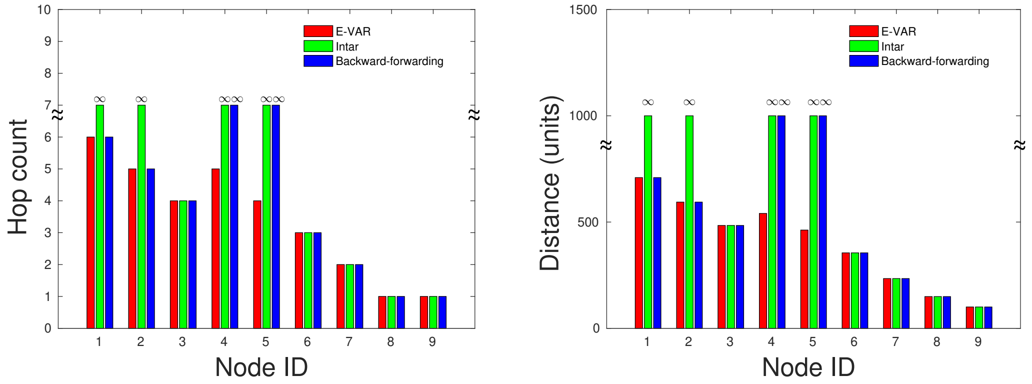

Figure 4.7: Hop count (left) and Distance to reach sink (right), in Backward-forwarding, Intar, and E-VAR with reference to Figure 4.4 through 4.6.

In summary, Figure 4.7 presents hop count and distance to reach the sink in Backward-forwarding, Intar, and E-VAR with reference to next hop selection in Figure 4.4, 4.5, and 4.6 respectively. As shown in Figure 4.7 hop count and distance to reach the sink is ∞ at nodes 4 and 5 in both Backward-forwarding and Intar. Moreover, in Intar, nodes 1 and 2 are also ∞ due to looping and are unreachable to the sink. However, the hop count and distance to reach the sink at nodes 3, 6, 7, 8, and 9 remain the same in Backward-forwarding, Intar, and E-VAR.

B. Scenario 2: E-VAR results in lesser hop count to reach the sink

The scenario 2 elaborates on the hop count to reach the sink in Backward-forwarding, Intar, and E-VAR. A slightly modified topology (no looping and one void node (node 6)) is considered in scenario 2 (Figure 4.8 and 4.9) rather than scenario 1.

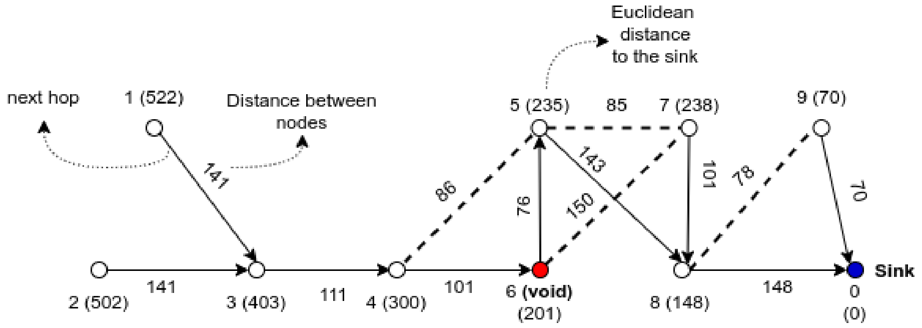

Figure 4.8: Backtracking scenario in Backward-forwarding and Intar.

B.1. Backward-forwarding and Intar

As shown in Figure 4.8, the next hop selection in Backward-forwarding and Intar is the same at every node. However, according to Backward-forwarding, at node 4, nodes 3, 5, and 6 are the neighbors with $D_{3,0}$ = 403, $D_{5,0}$ = 235 and $D_{6,0}$ = 201 units. Further, among the neighbors $\{3, 5, 6\}$ at node 4, node 6 (void) is selected as the next hop because of its higher advancement to the sink. Similarly, in Intar, at node 4, nodes 5 and 6 are the candidate PFN with $C_{4,5}$ = 10.75 and $C_{4,6}$ = 11.33 resulting in node 6 (void) as the next hop.

B.2. E-VAR

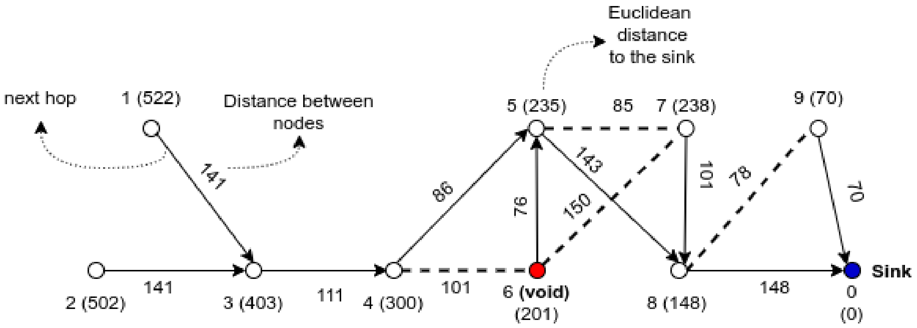

Figure 4.9: Avoidance of backtracking scenario in E-VAR.

Figure 4.9 presents the next hop selection in E-VAR with reference to the topology shown in Figure 4.8. However, in comparison with Backward-forwarding and Intar, E-VAR appropriately selects node 5 as the next hop at node 4 even though node 6 has higher advancement. As node 6 is void in status, E-VAR bypasses such nodes even though they are with higher advancement. Moreover, except at node 4, the next hop selection at the other nodes in E-VAR remains the same as Backward-forwarding and Intar, as shown Figure 4.8 and 4.9.

In short, Figure 4.10 compares hop count and distance, to reach the sink in Backward-forwarding, Intar, and E-VAR, with reference to the topology shown in Figure 4.8 and 4.9. As shown in Figure 4.8 and 4.9, hop count and distance at node 5, 6, 7, 8, and 9 are same in Backward-forwarding, Intar, and E-VAR. However, the variation in the

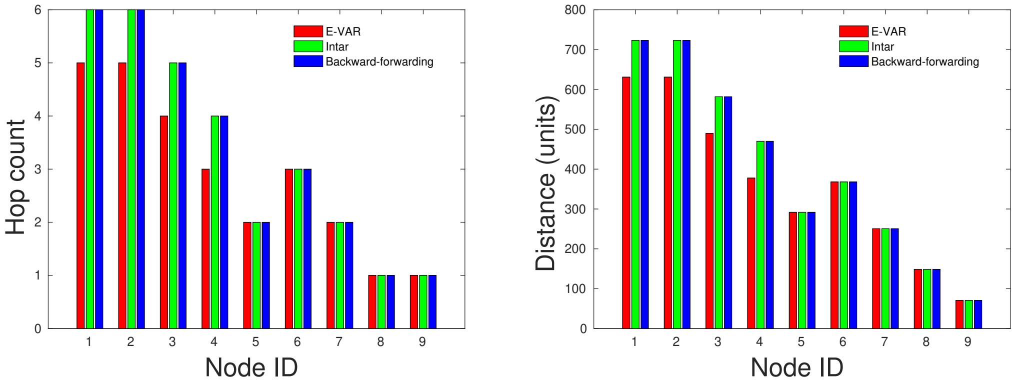

Figure 4.10: Hop count (left) and distance to reach sink (right) in Backward–forwarding, Intar, and E-VAR with reference to Figure 4.8 and 4.9.

selection of the next hop at node 4 results in increased hop count and distance, at nodes 1, 2, 3, and 4 in Backward-forwarding/Intar, in comparison with E-VAR.

4.2.2 Case-II: Random deployment

This case presents the performance evaluation of Backward-forwarding, Intar, and E-VAR in randomly deployed nodes. Further, Backward-forwarding, Intar, and E-VAR are compared in terms of the number of nodes reachable to the sink, the number of transmissions that failed due to looping, the average hop count, and distance. Higher and lower values of the number of nodes reachable and unreachable to the sink, respectively, are better in the utilization of deployed nodes and coverage area. Similarly, the lower average hop count and distance to reach the sink better is the lifetime of deployed nodes.

Table 4.3: Simulation parameters used in random deployment.

Parameters	**Values**
Area covered	$500 \times 500 \times 500$ units
Initial number of nodes	10
Additive increase of nodes in each simulation	10
Maximum number of nodes deployed	150
Coverage range of a sensor node	150

The MATLAB simulation is carried out initially for 10 randomly deployed nodes, with node 0 as a sink. In the subsequent simulation, additional 10 randomly deployed nodes are added, retaining the positions of the previous node intact. That is, during k^{th} simulation, node position of $(k-1)^{th}$ simulation is retained, and 10 additional randomly deployed nodes are added to the k^{th} simulation. In summary, simulation parameters are mentioned in Table 4.3.

Table 4.4: Performance metrics for Backward-forwarding, Intar, and E-VAR- Nodes reachable to the sink, Nodes unreachable to the sink due to looping.

Sl. No.	**No. of source nodes (G)**	**No. of isolated nodes**	**No. of void nodes**	**No. of reachable source nodes to the sink**			**No. of unreachable source nodes due to looping**		
				E-VAR	**Backward-forwarding**	**Intar**	**E-VAR**	**Backward-forwarding**	**Intar**
1	9	4	5	5	3	3	0	2	2
2	19	5	7	11	9	9	3	5	5
3	29	6	11	14	12	12	9	11	11
4	39	4	8	28	22	16	7	13	19
5	49	3	8	42	39	30	4	7	16
6	59	0	5	55	52	43	4	7	16
7	69	1	6	66	61	49	2	7	19
8	79	0	3	77	71	47	2	8	32
9	89	0	3	79	79	56	10	10	33
10	99	0	2	92	92	61	7	7	38
11	109	0	1	109	109	77	0	0	32
12	119	0	1	119	116	76	0	3	43
13	129	0	1	129	129	82	0	0	47
14	139	0	1	139	139	85	0	0	54
15	149	0	1	149	149	88	0	0	61

As shown in Table 4.4 and with reference to the explanation presented in Section 4.2.1, E-VAR outperforms 2.87% and 34% in comparison with Backward-forwarding and Intar, respectively, in terms of the number of reachable source nodes. E-VAR outperforms 40% and 88.78% in comparison with Backward-forwarding and Intar, respectively, in terms of the number of unreachable source nodes.

In this simulation, the average hop count and distance to reach the sink are computed as follows:

$$\text{Average hop count} = \frac{\sum\limits_{i=1}^{G} H_i}{G} \tag{4.3}$$

Table 4.5: Performance metrics for Backward-forwarding, Intar, and E-VAR- Average (Avg.) hop count, Avg. distance to reach the sink.

Sl. No.	**No. of source nodes (G)**	**Avg. hop count**			**Avg. distance (units)**		
		E-VAR	**Backward-forwarding**	**Intar**	**E-VAR**	**Backward-forwarding**	**Intar**
1	9	1.22	7.11	7.11	157.63	830.04	830.04
2	19	6.10	8.89	8.94	808.56	1036.71	1044.10
3	29	10.51	12.34	12.44	1223.87	1373.35	1387.48
4	39	7.48	11.35	15.56	635.42	1190.24	1775.70
5	49	5.46	7.08	12.10	577.39	762.12	1279.08
6	59	5.28	6.54	11.15	616.58	767.81	1320.87
7	69	4.30	6.05	11.14	514.74	715.58	1318.12
8	79	4.24	6.02	14.17	505.68	673.57	1461.38
9	89	6.26	6.29	13.47	613.19	726.24	1453.67
10	99	5.18	5.20	13.83	509.53	586.78	1555.39
11	109	3.44	3.47	11.55	422.40	424.24	1284.91
12	119	3.64	4.31	13.26	443.35	478.46	1453.47
13	129	3.54	3.55	13.42	426.26	426.62	1814.14
14	139	3.60	3.61	14.04	431.86	432.19	1908.77
15	149	3.59	3.60	14.59	427.71	428.01	1970.87

$$\text{Average distance} = \frac{\sum_{i=1}^{G} D_{i,S}}{G} \tag{4.4}$$

where $H_i \rightarrow$ hop count, $D_{i,S} \rightarrow$ distance to reach the sink from the source i, and $G \rightarrow$ the number of source nodes. In Eq. (4.3), H_i = 30 is used for unreachable nodes to the sink in case of looping. Similarly, $D_{i,S}$ is considered as the actual distance (in units) in 30 hops, for the unreachable nodes to the sink, in case of looping.

Table 4.5 shows that E-VAR outperforms 22.60% and 57.30% in comparison with Backward-forwarding and Intar, respectively, in terms of number average hop count. As the number of deployed nodes increases in a unit area, the difference between nodes unreachable to sink, average hop count, & distance, reduces among Backward-forwarding, Intar, & E-VAR. Although at some stage the increase in the number of deployed nodes results in a similar performance in Backward-forwarding, Intar, & E-VAR.

4.3 SUMMARY

This chapter of the thesis presents the E-VAR protocol, which uses the status of the neighboring nodes as a primary attribute. Further, Euclidean distance between neighbors to the sink is a secondary attribute in selecting the next hop. E-VAR outperforms 2.87% and 34% in comparison with Backward-forwarding and Intar, respectively, in terms of the number of reachable source nodes. Further, E-VAR outperforms 40% and 88.78% in comparison with Backward-forwarding and Intar, respectively, in terms of the number of unreachable source nodes.

The following chapter presents the proposed Link Quality-based Routing Protocol (LQRP) for UASNs.

CHAPTER 5

DESIGN AND EVALUATION OF LINK QUALITY-BASED ROUTING PROTOCOL (LQRP)

This chapter elaborates on the Link Quality-based Routing Protocol (LQRP) for UASNs published at the 11th International Conference on Computing, Communication, and Networking Technologies 2020 (ICCCNT 2020) held at IIT, Kharagpur (Nazareth and Chandavarkar 2020). LQRP uses the two-hop link quality of neighbors as an attribute to select the next hop. The next hop is decided based on the cost of the neighbors, which is computed by applying appropriate weight to the two-hop link quality. The detailed design and simulation, followed by its comparison with state-of-art Link Quality Estimation based Routing (LQER) (Jiang et al. 2006) protocol in terms of Packet Delivery Ratio (PDR) is presented in Section 5.1 and Section 5.2 respectively.

5.1 DESIGN OF THE LQRP PROTOCOL

This section discusses the design of the LQRP protocol in detail. Signal-to-noise ratio (SNR) is used to measure the link quality (Tan et al. 2012). Section 5.1.1 discusses the computation of the SNR from various physical parameters. Section 5.1.2 elaborates on determining the next hop in LQRP using computed SNR.

5.1.1 Computation of SNR

This section elaborates on the computation of the SNR between two nodes underwater. The SNR mainly depends on the source level, transmission loss, and noise. Further, the transmission loss depends on various physical parameters such as temperature, pH, and salinity. Besides, the noise caused due to thermal, turbulence, wind, shipping, the distance between nodes, and the depth at which the nodes are deployed. This section discusses the modeling of the SNR for deep underwater surface duct model (Domingo 2008). The SNR is computed using Eq. (5.1) (Urick 1983)

$$SNR = SL \ - \ TL \ - \ N(f) \ + \ DI \tag{5.1}$$

where SL is the source level, TL is the transmission loss, $N(f)$ is the noise level, and DI is the directivity index. Transmission loss TL is computed using Eq. (5.2)

$$TL = \begin{cases} 20 \ log \ r + (\alpha + \alpha_L) \ r \times 10^{-3} \\ where \ r < 350H^{1/2}(short \ ranges) \\ 10 \log \ r_0 + 10 \log \ r + (\alpha + \alpha_L) \ r \times 10^{-3} \\ where \ r > 350H^{1/2}(long \ ranges) \end{cases} \tag{5.2}$$

where r is the distance between nodes, $10 \ log \ r_0 \ = \ 20.9 + 5 \ log \ H$, where H is the mixed-layer depth (in meters), α_L is calculated by the Eq. (5.3)

$$\alpha_L = \frac{26.6f(1.4)^S}{((1452 + 3.52t)H)^{1/2}} \tag{5.3}$$

where f is the frequency of transmission (in kHz), S is the sea-state and t is the temperature. The total transmission loss depends on the absorption coefficient α in dB/km. The α is calculated by the Eq. (5.4).

$$\alpha = \frac{A_1 P_1 f_1 f^2}{f + {f_1}^2} + \frac{A_2 P_2 f_2 f^2}{f + {f_2}^2} + A_3 P_3 f^2 \tag{5.4}$$

The terms used in the Eq. (5.4) are explained in Eq. (5.5) through Eq. (5.11). A_1 and P_1 are the pure water component and depth pressure of pure water. In Eq. (5.4), f_1 is the relaxation frequency (in kHz) for Boric acid and it is calculated using Eq. (5.5)

$$f_1 = 2.8 \frac{SY}{35}^{0.5} \times 10^{(4-1245/(273+t)})} \tag{5.5}$$

where SY is the salinity of water (in parts/1000), and t is the temperature (in °C). The relaxation frequency f_2 of Magnesium sulphate is calculated using Eq. (5.6).

$$f_2 = \frac{8.17 \times 10^{8-1990/(273+t)}}{1 + 0.0018(s - 35)} \tag{5.6}$$

Boric acid level component A_1 is calculated using Eq. (5.7)

$$A_1 = \frac{8.68}{c} 10^{(0.78pH-5)} \tag{5.7}$$

where pH is the pH of water and c is the speed of sound in water, it is calculated using Eq. (5.8)

$$c = 1412 + 3.21T + 1.19S + 0.0167z \tag{5.8}$$

where z is the depth in meters. A_2 is the Magnesium sulphate component in seawater and it is calculated using Eq. (5.9)

$$A_2 = 21.44 \frac{S}{c} (1 + 0.025T) \tag{5.9}$$

P_1 and P_2 are the depth pressure of the Boric acid and Magnesium sulphate term in seawater. P_1 = 1 and P_2, P_3 are given the Eq. (5.10) and Eq. (5.11) respectively.

$$P_2 = 1 - 1.37 \times 10^{-4} z + 6.2 \times 10^{-9} z^2 \tag{5.10}$$

$$P_3 = 1 - 3.83 \times 10^{-5} z + 4.910^{-10} z^2 \tag{5.11}$$

Apart from transmission loss, SNR is affected by ambient noise. The ambient noise is the sum of the thermal, turbulence, waves, and shipping noise (Domingo 2008). The overall power spectral density on the ambient noise is calculated using Eq. (5.12)

$$N(f) = N_{th}(f) + N_t(f) + N_w(f) + N_s(f) \tag{5.12}$$

where $N_{th}(f)$ is thermal noise, $N_t(f)$ is noise due to turbulence, $N_w(f)$ noise due to wind , $N_s(f)$ is the noise due to shipping activity and are determined using Eq. (5.13) through Eq. (5.16), respectively.

$$10\ log\ N_{th}\ (f) = -15\ + 20\ log\ f \tag{5.13}$$

$$10\ log\ N_t\ (f) = 17\ - 30\ log\ f \tag{5.14}$$

$$10\ log\ N_w\ (f) = 50\ + 7.5 w^{1/2} + 20 \log\ f - 40\ log(f\ +\ 0.4) \tag{5.15}$$

where w is the wind speed in m/s.

$$10\ log\ N_s\ (f) = 40\ +\ 20\ (s - 0.5)\ +\ 26 f -\ 60\ log(f +\ 0.03) \tag{5.16}$$

where s is the shipping activity that ranges from 0 for the lowest shipping activity to 1 for the highest shipping activity.

5.1.2 Identifying a next hop in LQRP

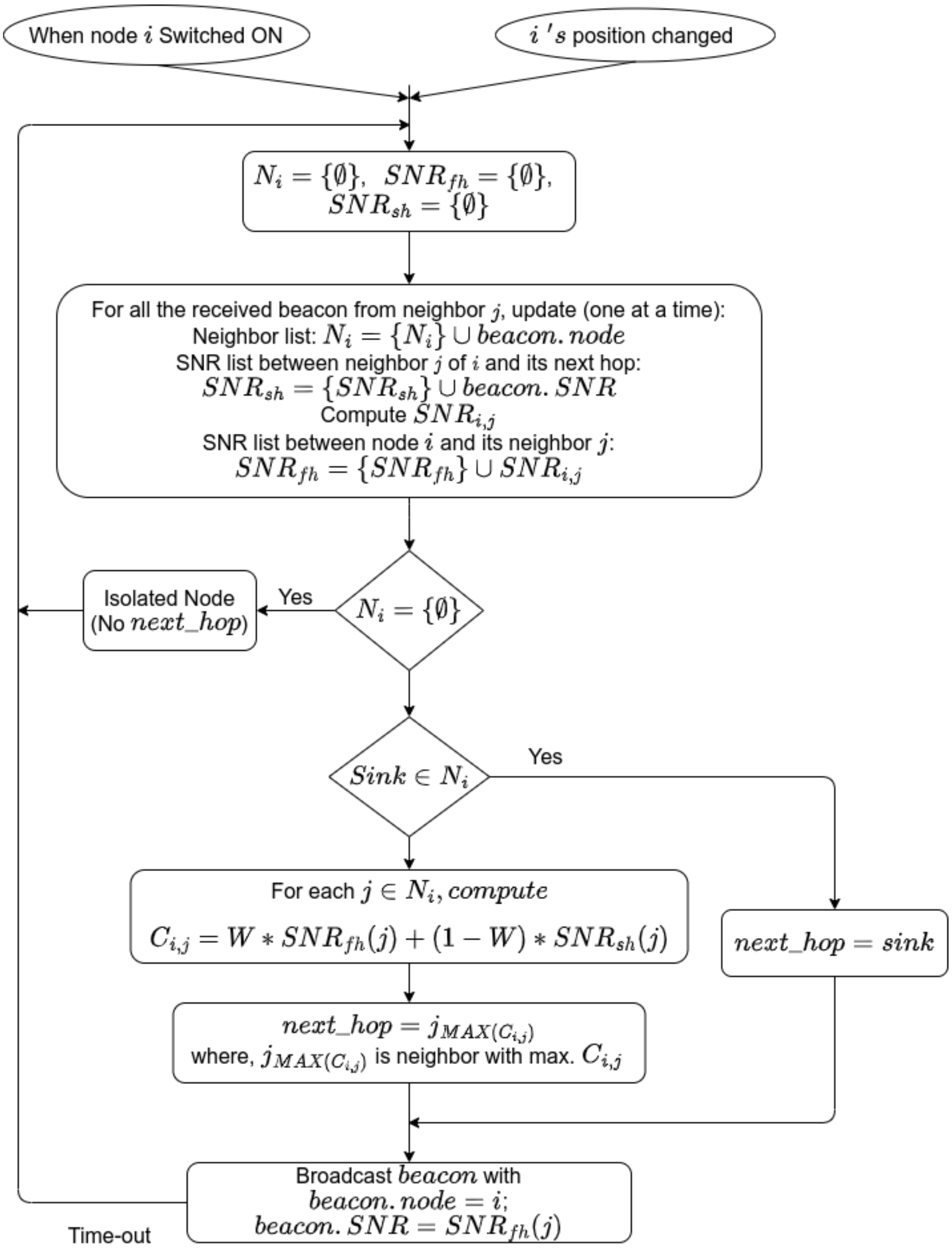
When node i Switched ON
i 's position changed
$N_i = \{\emptyset\}$, $SNR_{fh} = \{\emptyset\}$, $SNR_{sh} = \{\emptyset\}$
For all the received beacon from neighbor j, update (one at a time):
Neighbor list: $N_i = \{N_i\} \cup beacon.node$
SNR list between neighbor j of i and its next hop:
$SNR_{sh} = \{SNR_{sh}\} \cup beacon.SNR$
Compute $SNR_{i,j}$
SNR list between node i and its neighbor j:
$SNR_{fh} = \{SNR_{fh}\} \cup SNR_{i,j}$
Isolated Node (No $next\_hop$)
Yes
$N_i = \{\emptyset\}$
$Sink \in N_i$
Yes
For each $j \in N_i, compute$
$C_{i,j} = W * SNR_{fh}(j) + (1 - W) * SNR_{sh}(j)$
$next\_hop = sink$
$next\_hop = j_{MAX(C_{i,j})}$
where, $j_{MAX(C_{i,j})}$ is neighbor with max. $C_{i,j}$
Broadcast $beacon$ with
$beacon.node = i;$
$beacon.SNR = SNR_{fh}(j)$
Time-out

Figure 5.1: Identifying the next hop in LQRP.

The flowchart for next hop identification in LQRP is shown in Figure 5.1. Every deployed node executes this mechanism under scenarios such as switched on, the position changed, or time-out (regular interval). Any deployed node i initializes $N_i = \{\emptyset\}$, indicating that it does not have any neighbors; SNR_{fh} indicates the first hop, SNR of i with its neighbor j and is initialized to $SNR_{fh} = \{\emptyset\}$. Additionally, SNR_{sh} indicates the second hop SNR between neighbor j of i with its best next hop and is initialized to SNR_{sh} = $\{\emptyset\}$. When a node i receives beacon from the node j, N_i and SNR_{sh} are updated according to $beacon.node$ and $beacon.SNR$ respectively. Moreover, node i computes SNR with its neighbor j ($SNR_{i,j}$, Refer Section 5.1.1) and SNR_{fh} is updated. In the case of N_i = $\{\emptyset\}$, node i concludes itself as an isolated node without next hop ($next\_hop$). However, in the case of sink $\in N_i$, i selects sink as its next hop. Otherwise, cost of each neighbor j with reference to the node i is computed using Eq. (5.17).

$$C_{i,j} = W * SNR_{fh}(j) + (1 - W) * SNR_{sh}(j) \tag{5.17}$$

where W is the weight, ranging from 0 to 1.

Upon computing $C_{i,j}$, the neighbor j, with the highest $C_{i,j}$, is selected as a next hop. Besides, node i broadcasts its id, and SNR between itself, with its best next hop ($SNR_{i,next\_hop}$) information, using beacon $beacon.node$, and $beacon.SNR$, respectively.

5.2 SIMULATION AND EVALUATION OF THE LQRP

This section discusses the simulation and evaluation of the LQRP in detail. Simulation of LQRP is carried out using MATLAB. Further, its performance is compared with the state-of-the-art LQER protocol in terms of PDR (Jiang et al. 2006). The PDR is used as a performance metric for the comparison. Simulation topology and parameters are elaborated in Section 5.2.1. The results and analysis of LQRP are elaborated in Section 5.2.2.

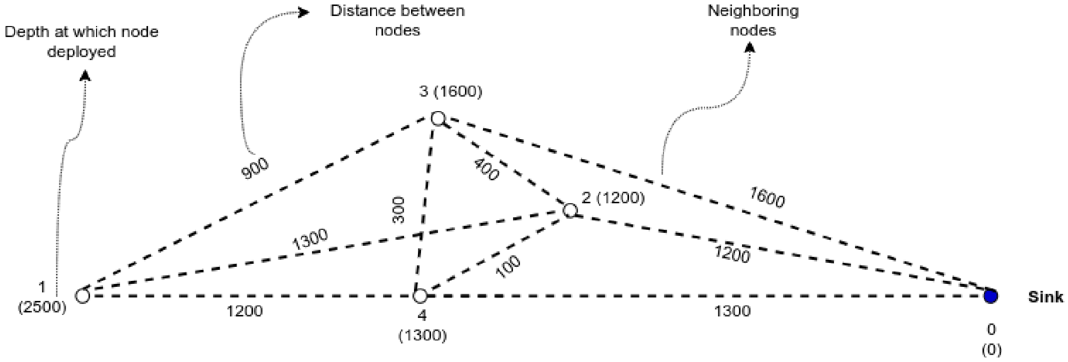

Figure 5.2: Simulation topology to evaluate LQRP with LQER.

5.2.1 Simulation setup

The topology shown in Figure 5.2 is used to evaluate the LQRP with LQER. In the given topology, node 1 is the source, and node 0 is the sink.

Table 5.1: Simulation parameters used in LQRP and LQER.

Parameters	**Values**
Number of nodes	5
Shipping activity ($N_s(f)$)	0 to 1 (increased in step of 0.1)
Temperature (T)	36 to 18 (decrease in step of 2 °C)
Salinity (S)	25 to 34 ppm(increased in step of 1)
Source level (SL)	105 dB/1 m Pa/1 m
Frequency (f)	10 kHz
pH (pH)	7.5

The simulation parameters used in evaluating LQRP with LQER are shown in Table 5.1. Thorough simulations are carried out by generating different input combinations of SNR between nodes for a topology, as shown in Figure 5.2, and the performance is evaluated. During every input combination, shipping activity, temperature, and salinity are varied. Additionally, using Eq. 5.1, first hop SNR (SNR_{fh}) between i and j, and second hop SNR (SNR_{sh}) between j and its best next hop are computed.

Algorithm 1 Generation of input combination of SNR between nodes.

```
Input: $D_{i,j} = Distance\ between\ node\ i\ and\ its\ neighbor\ j; D_{j,sh} = Distance\ between\ neighbor\ j\ of\ i\ and\ its\ best\ hop\ sh; Source\ level = 105; Frequency = 10kHz; pH = 7.5$
for $Shipping = 0.1 : 0.1 : 1$ do
    for $Temp = 36 : -2 : 18$ do
        for $Salinity = 25 : 1 : 34$ do
            $SNR_{i,j} = compute\_snr(Shipping, Temp, Salinity, D_{i,j}, Source\ level, Frequency, pH);$
            $SNR_{j,sh} = compute\_snr(Shipping, Temp, Salinity, D_{j,sh}, Source\ level, Frequency, pH);$
        end for
    end for
end for
Output: $SNR_{i,j}, SNR_{j,sh}$
```

The overall process of generating different input combinations is shown in Algorithm 1. For every input combination of SNR, cost of neighbor j, with reference to source/forwarder i, is computed (Refer Eq. 5.17) and a neighbor that has maximum $C_{i,j}$ is selected as next hop. In this way, the route path from the source to the sink is determined.

5.2.2 Results and analysis

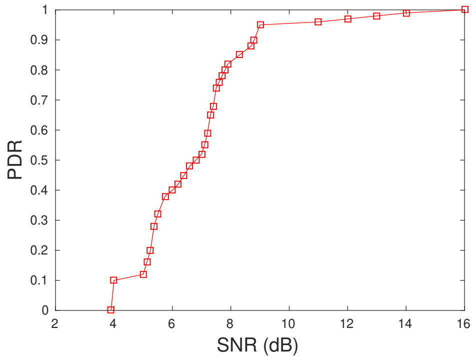

Figure 5.3: SNR-PDR profile (Tan et al. 2012).

This section discusses the results of LQRP, and its PDR is compared with the state-of-the-art LQER. Once the routing path, from the source to the sink, is determined (Refer Section 5.2.1), further, the overall PDR (PDR between source to the sink) is computed. In general, PDR is computed in two steps. In the first step, the corresponding PDR of individual link's from SNR, between the source to the sink is determined. The PDR of individual links can be obtained by using SNR-PDR profile (Tan et al. 2012). The SNR-PDR profile is shown in Figure 5.3.

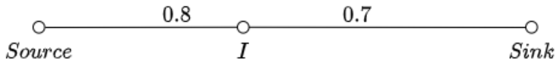

Figure 5.4: Computation of overall PDR.

In the second step, the overall PDR is calculated using the PDR of individual links, which is obtained in the previous step. The calculation of overall PDR is elaborated with the help of Figure 5.4.

The PDR from $Source$ to the intermediate node I is 0.8. It indicates that suppose 1 packet is sent from the $Source$ node, only 0.8 packets is received at the node I. Besides, out of 0.8 packets sent from node I, only 0.7 are received at the sink. That is,

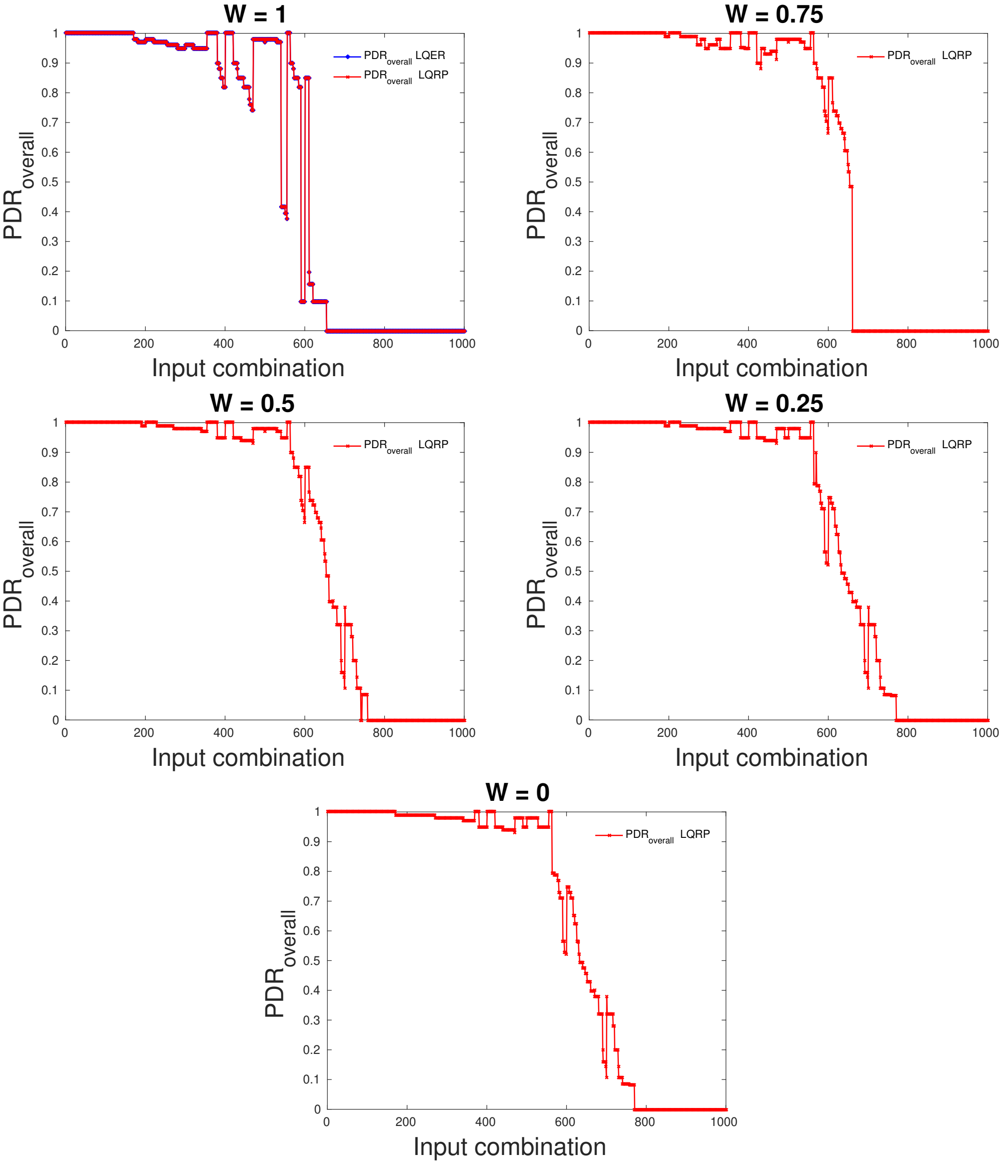

Figure 5.5: Overall PDR between a source to the sink when (a) $W = 1$ (b) $W = 0.75$ (c) $W = 0.5$ (d) $W = 0.25$ (e) $W = 0$.

$0.8 \times 0.7 = 0.56$ packet is received at the sink. Thus, the overall PDR from the source to the sink (Two hop topology as shown in Figure 5.4) is obtained by multiplying PDRs of individual links encountered in the path.

Figures 5.5(a) through 5.5(e) show average PDR in LQRP and LQER. In the case of the weight W = 1, in LQRP the complete weightage is given for $SNR_{fh}(j)$ (Refer Section 5.1.2). Accordingly, the next hop selection remains the same in both LQRP and LQER and achieves PDR of 57.22% as shown in Figure 5.5 (a). In LQRP, when weight

Table 5.2: Average PDR in LQRP for various weight W.

Weight	**Avg. PDR %**	
	LQRP	**LQER**
1	57.22	57.22
0.75	62.33	$*$
0.50	64.87	
0.25	63.87	
0	63.77	

$*$ Weight = 0.75, 0.5, 0.25, and 0 is not applicable in the case of LQER.

W is varied as 0.75, 0.5, 0.25, & 0, these weightages are used for $SNR_{fh}(j)$. Further, the remaining weightage 0.25, 0.5, 0.75 & 1 are used for $SNR_{sh}(j)$, respectively. The average PDR achieved in LQRP when W = 1, 0.75, 0.5, 0.25 & 0 and $W = 1$ for LQER are given in table 5.2.

5.3 SUMMARY

This part of the study presents LQRP for UASNs. LQRP uses the two-hop link quality, with appropriate weight, to the first and its best hop, to decide on the next hop. LQRP is evaluated with weights of 1, 0.75, 0.5, 0.25, and 0, and in all scenarios LQRP outperforms LQER in terms of PDR. Thus, LQRP is suitable for applications that demand high reliability.

The following chapter presents the proposed Location-Free Void Avoidance Routing (LFVAR) protocol for UASNs.

CHAPTER 6

DESIGN AND EVALUATION OF LOCATION-FREE VOID AVOIDANCE ROUTING (LFVAR) PROTOCOL

This chapter elaborates on Location-Free Void Avoidance Routing (LFVAR) protocol for UASNs published in Springer's Wireless Personal Communication (Nazareth and Chandavarkar 2021b). LFVAR uses the neighboring node's status (void or normal) as a primary attribute, followed by hop count to reach the sink, and the depth (z-coordinate), as secondary attributes to select the next hop. The detailed design and implementation using UnetStack (Chitre 2022b) are presented in Section 6.1 and 6.2 respectively. Additionally, Section 6.3 presents the comparison of LFVAR protocol with the state-of-the-art Interference-aware routing (Intar) (Javaid et al. 2018) in terms of average hop count, end-to-end delay, PDR, throughput, and energy consumption.

6.1 DESIGN OF THE LFVAR PROTOCOL

This section elaborates on the design of the LFVAR protocol, which uses a void avoidance mechanism to forward the packets. LFVAR protocol has following two phases:

- **Phase-I: Initialization of hop count, status, and depth followed by the broadcasting of beacon by the sink**
  During this phase, all nodes in the UASNs initialize their hop count, status, and depth. Further, during the regular interval, the sink node broadcasts the beacons in the network (Refer Section 6.1.1).

- **Phase-II: Identifying the next hop**

  During this phase, every node (except the sink) in the network identifies its next hop based on information such as hop count, status, and depth, propagated through the beacons. The sink initiates the propagation of the beacon (Phase-I), which is further updated, and forwarded by the other nodes in the network. The detailed design of Phase-II is presented in Section 6.1.2.

6.1.1 Phase-I: Initialization of hop count, status, and depth followed by the broadcasting of beacon, by the sink

During this phase, every deployed node i (except the sink) initializes,

- Hop count (H_i) $\rightarrow$ 99 (indicates node i is not reachable to the sink).
- Status (V_i) $\rightarrow$ 1 (indicates node i as a void).
- Depth (D_i) $\rightarrow$ z-coordinate of node i.
- Neighbour list (N_i) $\rightarrow \{\emptyset\}$ (indicates no neighbours to node i).
- Neighbour table ($NT^i_{p\times q}$) $\rightarrow$ [ ], holds the attributes (node id, hop count, status, and the depth) value of the neighbor(s) j of node i and $j \in N_i$, $p = |N_i|$, $q = 4$.
- Potential Relay Node(s) (PRNs) of node i (PRN_i) $\rightarrow \{\emptyset\}$ (indicating no PRN to node i). PRN_i can be defined as per Eq. (6.1).

$$PRN_i = \{j : \forall\ j \in N_i \ \&\&\ NT^i[I_j][4] < D_i \ \&\&\ NT^i[I_j][3] == 0\} \quad (6.1)$$

where I_j represents the index of neighboring node j in N_i. $NT^i[I_j][3]$, and $NT^i[I_j][4]$ represent the status and depth of the node j, respectively. However, the sink node i initializes,

- Hop count (H_i) $\rightarrow$ 0 (indicates sink i itself is the destination).
- Status (V_i) $\rightarrow$ 0 (indicates sink node i is a normal node).

- Depth (D_i) → z-coordinate of sink node i.

The sink node i broadcasts the $beacon$, consisting of $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$ at regular interval.

6.1.2 Phase-II: Identifying the next hop

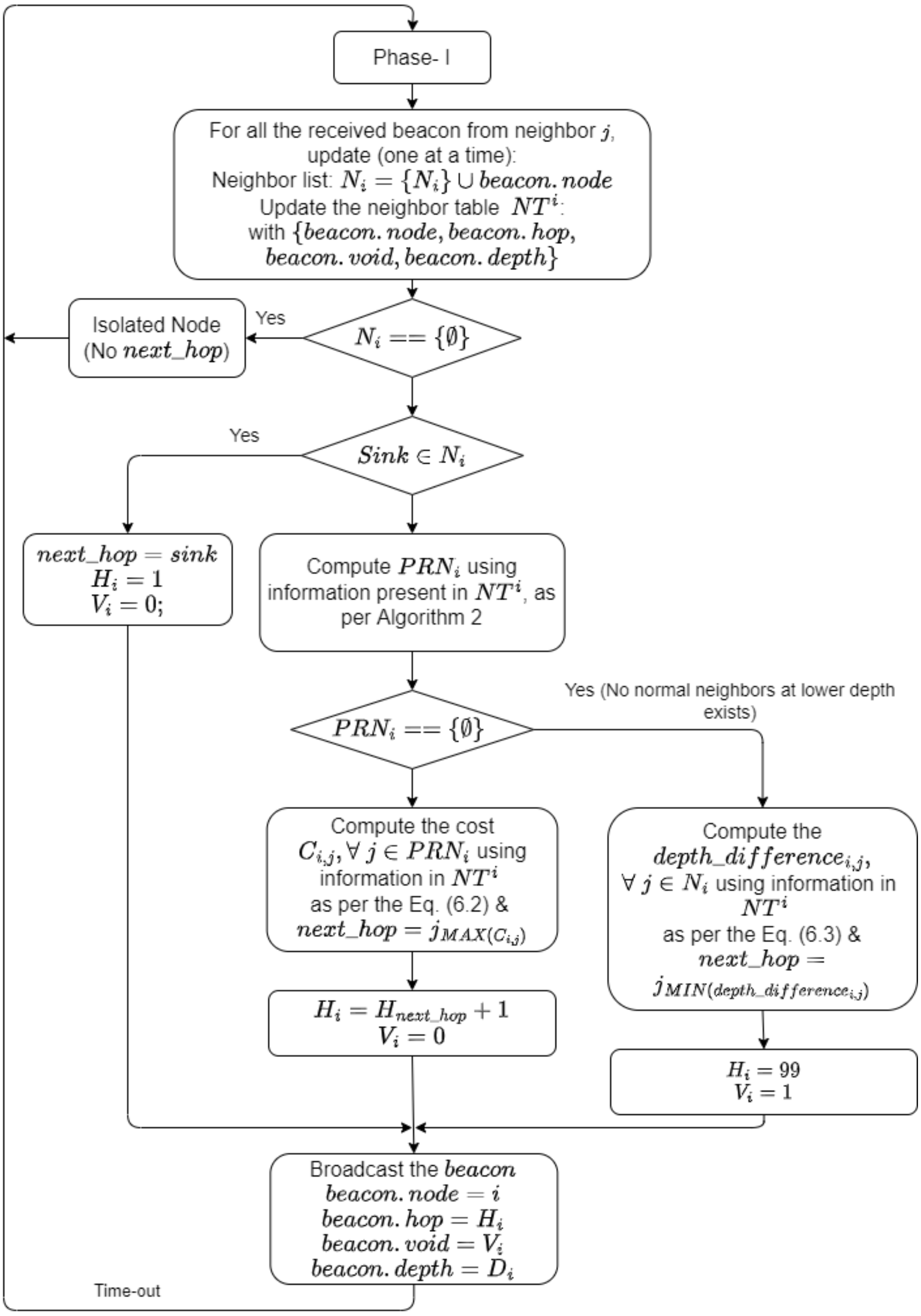
Phase- I
For all the received beacon from neighbor j, update (one at a time):
Neighbor list: $N_i = \{N_i\} \cup beacon.node$
Update the neighbor table NT^i:
with $\{beacon.node, beacon.hop, beacon.void, beacon.depth\}$
$N_i == \{\emptyset\}$
Yes
Isolated Node (No $next\_hop$)
$Sink \in N_i$
Yes
$next\_hop = sink$ $H_i = 1$ $V_i = 0;$
Compute PRN_i using information present in NT^i, as per Algorithm 2
$PRN_i == \{\emptyset\}$
Yes (No normal neighbors at lower depth exists)
Compute the cost $C_{i,j}, \forall j \in PRN_i$ using information in NT^i as per the Eq. (6.2) & $next\_hop = j_{MAX(C_{i,j})}$
Compute the $depth\_difference_{i,j}$, $\forall j \in N_i$ using information in NT^i as per the Eq. (6.3) & $next\_hop = j_{MIN(depth\_difference_{i,j})}$
$H_i = H_{next\_hop} + 1$ $V_i = 0$
$H_i = 99$ $V_i = 1$
Broadcast the $beacon$ $beacon.node = i$ $beacon.hop = H_i$ $beacon.void = V_i$ $beacon.depth = D_i$
Time-out

Figure 6.1: Phase-II: Determining the next hop at node i (except the sink).

During this phase, the next hop in the LFVAR protocol is identified. The detailed design of Phase- II is shown in Figure 6.1. The N_i and NT^i are updated at node i, when it receives beacon. Selection of the next hop at every node i in LFVAR protocol for different scenarios is explained as follows:

- **Node i does not have any neighbor** ($N_i = \{\emptyset\}$)

  The node i is the isolated node; thus, it does not have any next hop.

- **One of the neighbors is the sink node** ($Sink \in N_i$)

  The node i selects the sink as its next hop ($next\_hop = Sink$). Further, node i updates its hop count to 1 ($H_i = 1$), status to normal ($V_i = 0$). Also, it broadcasts the $beacon$ which consists of $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$.

Algorithm 2 Updating PRN_i at the node i.

```
count = 1
while $(count \le |N_i|)$ do
    if $(NT^i[count][4] \le D_i$ && $(NT^i[count][3] == 0)$ then
        $PRN_i \leftarrow PRN_i \cup \{NT^i[count][1]\}$
    end if
    $count = count + 1$
end while
```

The remaining scenarios depend on the computation of PRN_i. Computation of the PRN_i is shown in the Algorithm 2. The PRN_i is computed using the depth and status of nodes in N_i.

- **Node i has PRNs** ($PRN_i \neq \{\emptyset\}$)

  In the case PRN_i is non-empty, node i computes the cost ($C_{i,j}$) of $j \in PRN_i$ with reference to i given as follows

  $$C_{i,j} = \frac{(D_i - NT^i[I_j][4])}{Range} \times \frac{1}{NT^i[I_j][2]} \tag{6.2}$$

  where $Range$ is the coverage range. The value of the $C_{i,j}$ ranges from 0 to 1. Further, node i selects a neighbor j which has the highest $C_{i,j}$ as its next hop

$(next\_hop = j_{MAX(C_{i,j})})$, and sets its status as normal ($V_i = 0$). Moreover, node i updates its H_i, by incrementing hop count of selected $next\_hop$ (H_{next_hop}). Finally, node i broadcasts the $beacon$ which consists of $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$.

- **Node i does not have any PRNs** ($PRN_i = \{\emptyset\}$)
  This scenario indicates that node i does not have any normal neighbors at the lower depth. In such cases, node i selects its next hop, deployed at a higher depth, using Eq. (6.3)

$$depth\_difference_{i,j} = (NT^i[I_j][4] - D_i) \tag{6.3}$$

  where $depth\_difference_{i,j}$ is the depth difference of node j present at higher depth with reference to i and $j \in N_i$. Node i selects a node j with the lowest $depth\_difference_{i,j}$ as its next hop ($next\_hop = j_{MIN(depth\_difference_{i,j})}$). Accordingly, node i updates its status as void ($V_i = 1$), hop count (H_i) as 99. Additionally, node i broadcasts $beacon$ which consists of $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$.

6.2 IMPLEMENTATIONS OF LFVAR

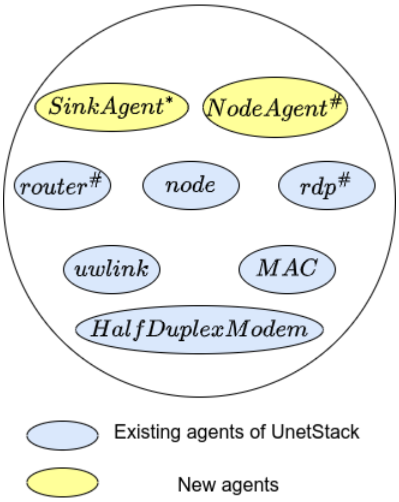

Figure 6.2: Agents used in the implementation of LFVAR protocol in UnetStack.
* Agents only applicable to the sink node.
$^\#$ Agents only applicable to all nodes (except to the sink).

Here, implementation of the LFVAR in UnetStack is described. UnetStack is a agent-based UASNs simulation tool (Chitre 2022b; Chitre et al. 2014b). As discussed in Section 6.1, the LFVAR protocol is executed at all the nodes in the network. The various agents used in implementing the LFVAR are shown in Figure 6.2. The agent, *SinkAgent*, implements the sink node initialization and broadcasting of the beacon. The agent, *NodeAgent*, implements the identification of the next hop at every node (except at the sink). The major features of UnetStack followed by implementation details of *SinkAgent* and *NodeAgent* are elaborated in Section 6.2.1, 6.2.2 and 6.2.3, respectively.

6.2.1 UnetStack

UnetStack [1] is based on the Framework for Java and Groovy Agents (Fjage) (Chitre 2022a). One of the salient features of the UnetStack is that, it organises many software-oriented agents. An agent is similar to a layer in the conventional network stack. However, agents provide more flexibility in terms of resource consumption. Thus cross-layer optimization is fulfilled easily when compared to traditional protocol stack architecture. There are many advantages of using UnetStack in the simulation of underwater networks. UnetStack supports both discrete and real-time simulation. Discrete-event simulation is instrumental in executing extensive simulations in a short time, thereby allowing robust testing of protocols. Real-time simulation is helpful in testing protocols with the intent of a real-time environment. Another significant advantage of UnetStack, compared to most other tools, is that it provides the transition of compiled binary code of simulation to UnetStack-compliant underwater modem without any changes (Luo et al. 2017; Chitre et al. 2014b,a). Thus there is no need to write separate code for simulation and real-time modem.

The *router* and *rdp* agents provide ROUTING and ROUTE_MAINTENANCE services, respectively. The ROUTING service assists in maintaining the routing table and forwarding the data accordingly. ROUTE_MAINTENANCE determines new routes and notifies the changes in the routes, to the ROUTING service (Chitre 2022b). *MAC* agent provides access to the channel with reduced data packet collision and enhanced

---

[1] https://unetstack.net/

throughput. The *node* agent provides NODE_INFO service, which helps obtain node information. The *uwlink* provides reliable transmission link. The *HalfDuplexModem* agent implements a PHYSICAL service that enables direct access to the channel.

6.2.2 The *SinkAgent*

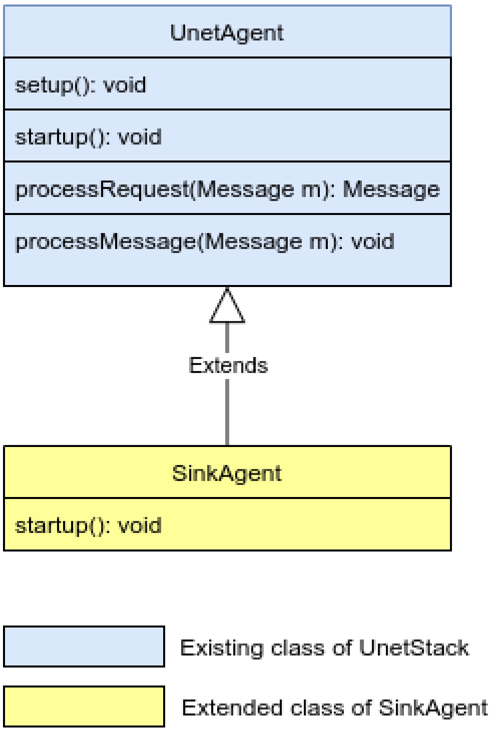
UnetAgent
setup(): void
startup(): void
processRequest(Message m): Message
processMessage(Message m): void
Extends
SinkAgent
startup(): void
Existing class of UnetStack
Extended class of SinkAgent

Figure 6.3: Organization of *SinkAgent*.

The *SinkAgent* is implemented at the sink node, responsible for initializing and broadcasting the beacon at regular intervals. The organization of *SinkAgent* is shown in Figure 6.3. The *startup()* method of the *SinkAgent* overrides *startup()* method, defined in the *UnetAgent* base class. The base class *UnetAgent* defines most of the necessary behaviors. With reference to Section 6.1, the sink node i creates and broadcasts the beacon consisting of $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$. The node id i and depth D_i, at which the sink is deployed, are obtained using services provided by the *node* agent. The beacon is sent by using *TickerBehavior* defined by Fjage (Chitre 2022a). A *TickerBehavior* is executed repeatedly with a specified time between calls.

6.2.3 The *NodeAgent*

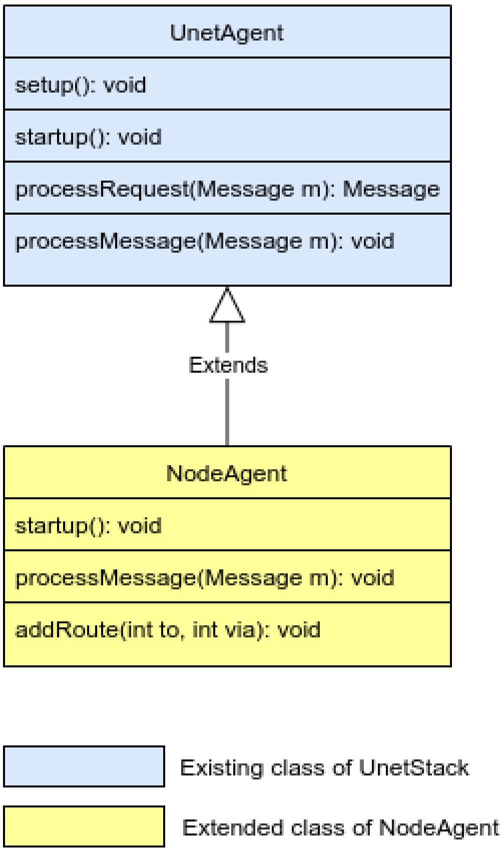
UnetAgent
setup(): void
startup(): void
processRequest(Message m): Message
processMessage(Message m): void
Extends
NodeAgent
startup(): void
processMessage(Message m): void
addRoute(int to, int via): void
Existing class of UnetStack
Extended class of NodeAgent

Figure 6.4: Organization of *NodeAgent*.

The *NodeAgent* is deployed at all nodes (except at the sink) in UASNs for identification of the next hop and is extended from the *UnetAgent* class. The *startup()* and *processMessage()* methods of the *NodeAgent*, override the ones defined in the *UnetAgent* base class of the UnetStack. Additionally, *NodeAgent* class introduces *addRoute()* method as shown in Figure 6.4. The implementation of each method is explained as follows:

- *startup()*: This method obtains the agent id's of the physical, node, rdp, MAC, and uwlink. A physical agent id is required to broadcast the beacon and notify the incoming beacon messages from other nodes and the *node* agent id is required to obtain the node's id. Using the router agent id, the next hop is configured in the routing table of UnetStack.

- *processMessage()*: It is the core method in identifying the next hop in LFVAR protocol. Whenever a node i receives the beacon from a neighboring node, *processMessage()* method is triggered, and it stores the information present in the beacon into the neighbor table NT^i and finds the next hop using Eq. (6.2) or (6.3).

- *addRoute()*: This method is called by *processMessage()*; when there is a change in next node. The *addRoute()* generates *RouteDiscoveryNtf* notification, and is sent to the router agent to update the routing table.

6.3 SIMULATION AND EVALUATION OF THE LFVAR PROTOCOL

This section discusses the simulation and evaluation of the LFVAR protocol in detail. Simulation of LFVAR is carried out using UnetStack. Section 6.3.1 discusses the next hop selection in the Intar, and LFVAR. Moreover, Section 6.3.2 presents the comparison of LFVAR protocol with the state-of-the-art Interference-aware routing (Intar) (Javaid et al. 2018), in terms of average hop count, end-to-end delay, Packet Delivery Ratio (PDR), throughput, and energy consumption.

6.3.1 Next hop selection in Intar and LFVAR

In this subsection discussion is on the next hop selection in the Intar and LFVAR which is done using an 8-node topology. Nodes deployed in a 3-dimensional location (x, y, z) are manually fixed in a 500 $\times$ 500 $\times$ 500 units area with the coverage range of nodes as 100 units.

A. Next hop selection in Intar

The next hop selection in Intar is explained with the help of the topology as shown in Figure 6.5. The next hop selection in Intar is based on the cost, which is calculated based on Eq. (4.2) (Refer Section 4.2.1) (Javaid et al. 2018). A PFN that has the highest cost is selected as the next hop. In the case of non-availability of PFN, Intar selects a neighbor, with the minimum Euclidean distance to the sink, as the next hop

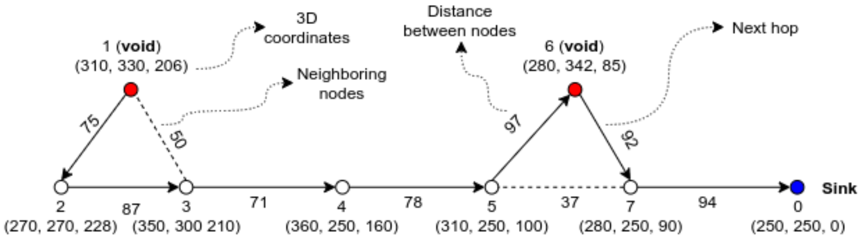

Figure 6.5: Next hop selection in the Intar.

among the non-PFN, without applying Eq. (4.2). As shown in Figure 6.5, node 7 is directly reachable to the sink (node 0). Thus node 7 selects sink as its next hop. Node 6 has 5 and 7 as its neighbors at higher depths (non-PFNs). Node 7 has a lower Euclidean distance to the sink (94.86 units) than 5 (116.61 units). Thus node 6 selects 7 as its next hop.

Table 6.1: Cost of PFN of node 5.

Node (j)	**Number of neighbors** ($\|N_j\|$)	**Distance from node 5** ($D_{5,j}$)	**Hop distance from the sink** (H_j)	**Cost** ($C_{5,j}$)
6	2	97	2	$\frac{97}{2 \times 2} = 24.25$
7	3	37	1	$\frac{37}{3 \times 1} = 12.33$

Nodes 6 and 7 are the candidate PFNs to 5. As shown in Table 6.1, $C_{5,6}$ and $C_{5,7}$ (Eq. (4.2)) are calculated. $C_{5,6}$ is higher than $C_{5,7}$, resulting in node 5 selecting 6 as the next hop. Further, node 4 has only a PFN 5, thus, node 4 selects 5 as its next hop. Node 3 has candidate nodes 4 and 1. Also, by computing $C_{3,4}$ and $C_{3,1}$, node 3 selects 4 as its next hop (Eq. (4.2)). Similarly, node 2 selects 3 as its next hop. However, node 1 is a void node, and it does not have any PFN; thus, it selects non-PFN at the lowest distance to the sink as its next hop. Nodes 2 and 3 are non-PFN to void node 1. Node 2 has a lower Euclidean distance to the sink (229.74 units) than that of 3 (237.9). Thus node 1 selects 2 as its next hop.

B. Next hop selection in LFVAR

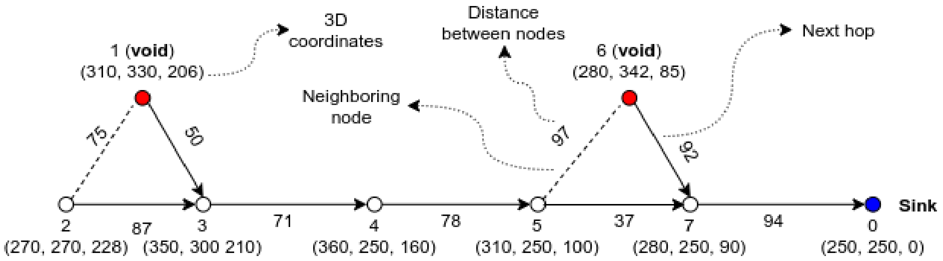

Figure 6.6: Next hop selection in the LFVAR.

The next hop selection in LFVAR is discussed with reference to the topology shown in Figure 6.6 (similar to topology shown in Figure 6.5). Node 7 has only the sink as its PRN. Thus, node 7 selects the sink as its next hop (Refer Section 6.1.2).

Table 6.2: Depth difference with neighbors of nodes 6 and 1.

Node	Depth	depth_difference
Node 6 (Depth = 85)		
5	100	100 - 85 = 15
7	90	90 - 85 = 5
Node 1 (Depth = 206)		
2	228	228 - 206 = 22
3	210	210 - 206 = 4

Node 6 does not have any PRN, hence, $depth\_difference_{6,5}$ & $depth\_difference_{6,7}$ are computed (Eq. (6.3)), as shown in Table 6.2. Further, node 6 selects 7 as its next hop since $depth\_difference_{6,7}$ is lower than $depth\_difference_{6,5}$. Nodes 5 and 4 have lone PRNs 7 and 5, respectively, and they are selected as their next hop. Similarly, nodes 3 and 2 have lone PRNs 4 and 3, respectively, becoming their next hop. For void node 1, there are no PRNs'; thus, depth differences with nodes deployed at higher depths are computed. Accordingly, node 1 computes $depth\_difference_{1,2}$ and $depth\_difference_{1,3}$. Node 1 selects 3 as its next hop since $depth\_difference_{1,3}$ is lower than $depth\_difference_{1,2}$, as shown in Table 6.2.

6.3.2 Performance comparison between LFVAR and Intar

Table 6.3: Simulation parameters used in the evaluation of the LFVAR.

Parameters	**Value**
The number of manually deployed nodes	5, 9, 13, 17, 21 and the number of void nodes are 1, 2, 3, 4, 5, respectively
Transmission energy	2 W
Reception energy	0.1 W
Simulation duration	1800 seconds
Transmission range	100 meters
Load	3 packets/minute
Size of data packet	384 bits

In addition to comparing LFVAR with Intar in Section 6.3.1, further performance is compared in terms of average hop count, end-to-end delay, PDR, throughput, and energy consumption. The simulation parameters are shown in Table 6.3. Results and its analysis of individual performance metrics are elaborated as follows:

A. Average hop count

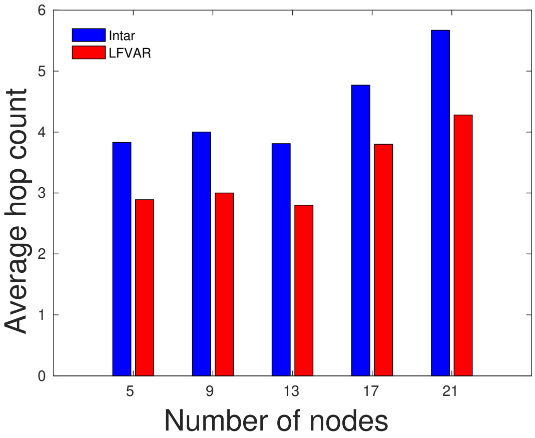

Figure 6.7: Average hop count in LFVAR and Intar.

It is the average number of hops to reach the sink. The lower the average hop count, the better the performance. As shown in Figure 6.7, the next hop selection in LFVAR results in a lower hop count than Intar. LFVAR prevents the selection of void node and trap node as the next hop. In Intar, even a void node can be part of the routing. Intar follows the void recovery mechanism; in the case of a packet stuck at the void node, it results in an increased hop count. In general, simulations show that LFVAR results in 23.87 % less hop count, compared to Intar.

B. Average end-to-end delay

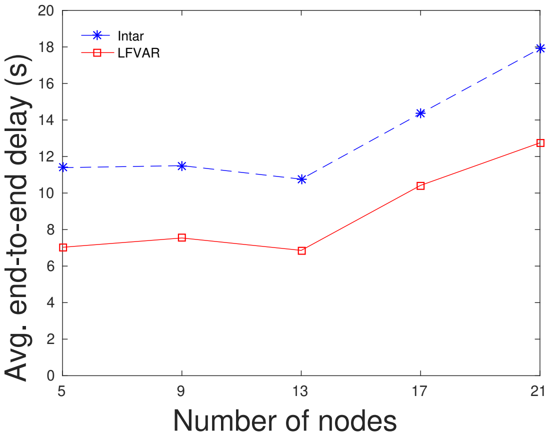

Figure 6.8: Average end-to-end delay in LFVAR and Intar.

The end-to-end delay is the time elapsed from the packet generation at the source to the reception of the packet at the sink. The end-to-end delay includes propagation delay, processing delay, and transmission delay. The node introduces transmission and processing delays as packets forward through each intermediate hop. The transmission delay is the time required to transmit all bits of a packet into media by the node. Processing delay is the time needed to decode the packet header. As elaborated in Section 6.3.1, LFVAR results in a lower hop count than Intar. Thus, overall transmission and processing delay is decreased in LFVAR, than in Intar, resulting in reduced end-to-end delay as shown in Figure 6.8. In general, simulations show that LFVAR results in 32.32 % better performance than Intar in terms of average end-to-end delay.

C. PDR

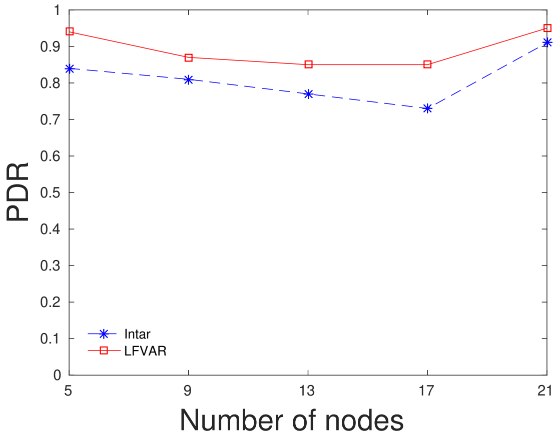

Figure 6.9: PDR of LFVAR and Intar.

PDR is the ratio of the distinct packet received by the sink, to the total number of packets sent by the source nodes in the network. As shown in Figure 6.9, LFVAR achieves a better PDR than Intar. The transmitted packet, by different sources, takes more time to deliver to the sink node in the Intar (Refer Section 6.3.1 and 6.3.2B). Thus packets remain in the network for a longer duration, which causes a collision between packets. In conclusion, simulations show the PDR of LFVAR is 9.87 % more than that of Intar.

D. Throughput

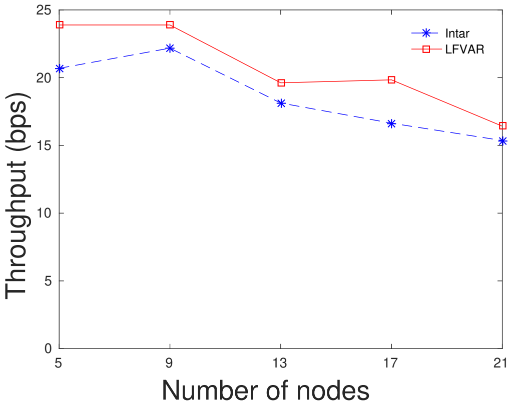

Figure 6.10: Throughput of LFVAR and Intar.

Figure 6.10 shows that LFVAR achieves better throughput than Intar. One of the reasons for achieving better throughput in LFVAR is, more packets are delivered to the sink than Intar. As packets travel through fewer intermediate hops, there is a lower chance of packet loss. With reference to Figure 6.7, LFVAR results in a lower number of intermediate hops to reach the sink, therefore, a lower packet loss than Intar.

E. Energy consumption

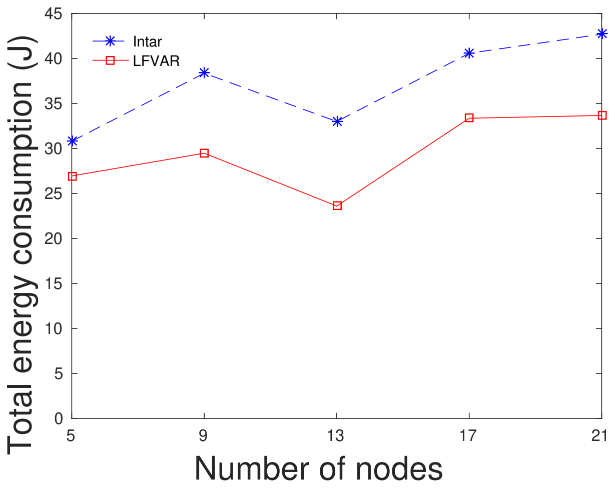

Figure 6.11: Energy consumption in LFVAR and Intar.

Figure 6.11 shows the total energy consumption with an increase in the number of nodes. As per Figure 6.7, the average hop count increases as the number of nodes increases. Thus as the average hop count increases, the number of transmissions and reception of packets increases, resulting in higher energy consumption. In terms of energy consumption, LFVAR always consumes less energy than Intar. This is due to LFVAR resulting in a lower hop count, with fewer number of transmissions and receptions of the packet. When compared, LFVAR consumes 20.54 % less energy than Intar in the simulations.

6.4 SUMMARY

This part of the thesis presents the LFVAR protocol. Results and analysis show that the LFVAR protocol outperforms 23.87% and 32.33% in comparison with Intar in terms of hop count and end-to-end delay, respectively. Additionally, LFVAR protocol outperforms 9.87% and 20.54% in comparison with Intar, in terms of PDR and energy consumption, respectively. Thus, the LFVAR protocol is suitable for delay-sensitive, reliable underwater applications.

The following chapter presents the proposed Link and Void Aware Routing (LVAR) protocol.

CHAPTER 7

DESIGN AND EVALUATION OF LINK AND VOID-AWARE ROUTING (LVAR) PROTOCOL

This chapter elaborates on the Link and Void Aware Routing (LVAR) protocol for UASNs, which is another version of LFVAR (Chapter 6). LVAR protocol has been published in the 12$^{\text{th}}$ International Conference on Computing, Communication, and Networking Technologies 2021 (ICCCNT 2021) held at IIT, Kharagpur (Nazareth and Chandavarkar 2021a). LVAR uses neighboring node's status (void or normal) as a primary attribute. Additionally, LVAR uses the hop count, the depth (z coordinate), and Packet Delivery Probability (PDP) with its neighbor as secondary attributes to select the next hop. The detailed design, and the simulation are followed by the comparison with state-of-the-art Interference-aware routing (Intar) (Javaid et al. 2018) as presented in Section 7.1, and 7.2 respectively. Hop count, end-to-end delay, and throughput are used as performance metrics for the comparison.

7.1 DESIGN OF THE LVAR PROTOCOL

As mentioned above, LVAR is another version of LFVAR. Thus, the design of LVAR is explained briefly. As explained in the LFVAR, LVAR also contains two phases (Refer Section 6.1).

- **Phase-I: Initialization of hop count, status, and depth, followed by the broadcasting of beacon by the sink**

- **Phase-II: Identifying the next hop**

  In this phase, information in the received beacon is stored locally at the node. Further, based on this information, a node identifies its next hop.

The Phase-I of LVAR is the same as that of LFVAR. However, only in Phase-II of the following scenario the next hop selection in LVAR varies in comparison with LFVAR:

Node i has Potential Relay Nodes (PRNs) ($PRN_i \neq \{\emptyset\}$)
In case the PRN_i is non-empty, node i computes the cost $C_{i,j}$ of all neighbors j, where $j \in PRN_i$ as per Eq. (7.1).

$$C_{i,j} = \frac{(D_i - NT^i[I_j][4]) \times P_{i,j}}{Range \times NT^i[I_j][2]} \tag{7.1}$$

where $P_{i,j}$ is the PDP between source/forwarder node i with j (Refer Section 7.1.1), $Range$ is the communication range of the node. The value of the $C_{i,j}$ ranges from 0 to 1. Further, node i selects a neighbor j which has the highest cost as its next hop ($next\_hop = j_{MAX(C_{i,j})}$) and sets its status as normal ($V_j = 0$). Additionally, node i updates its hop count (H_i) by incrementing hop count of selected $next\_hop$. Finally, node i broadcasts $beacon$ with $beacon.node = i$, $beacon.hop = H_i$, $beacon.void = V_i$, and $beacon.depth = D_i$.

7.1.1 Computation of packet delivery probability

The Packet Delivery Probability (PDP) is an important parameter in deciding the transmission reliability between pairs of nodes. It mainly depends on path loss, the distance between nodes, and packet size. The path loss over a distance is given by (Domingo 2008):

$$A(d, f) = d^k \alpha(f)^d \tag{7.2}$$

where d, k, $\alpha(f)$ are distance between nodes, spreading factor $(1 \leq k \leq 2)$ and absorption coefficient respectively. The following equation calculates the absorption coefficient:

$$\alpha(f) = \frac{0.1f^2}{1+f^2} + \frac{40f^2}{4100+f^2} + 2.75 \times 10^{-4} f^2 + 0.003 \tag{7.3}$$

where f is the frequency in kHz. The average Signal-to-Noise Ratio (SNR), over the distance d ($SNR(d)$), is computed by the following equation (Noh et al. 2015):

$$SNR(d) = \frac{E_b}{N_0 d^k \alpha(f)^d} \tag{7.4}$$

where E_b is the average transmission energy per bit, and N_0 is the noise power density in a nonfading Additive White Gaussian Noise (AWGN) channel. Bit error probability $P_e(d)$ over a distance d is computed by (Noh et al. 2015):

$$p_e(d) = 0.5 \left(1 - \sqrt{\frac{SNR(d)}{1+SNR(d)}}\right) \tag{7.5}$$

Finally, if two nodes are separated by a distance d and packet size is m bits, then PDP $P_{i,j}$ can be calculated by the equation:

$$P_{i,j} = (1 - p_e(d))^m \tag{7.6}$$

7.2 SIMULATION AND EVALUATION OF THE LVAR PROTOCOL

This section discusses the simulation and evaluation of the LVAR protocol in detail. Simulation of LVAR is carried out using UnetStack. Further, its performance is compared with the state-of-the-art Intar (Javaid et al. 2018) protocol. Section 7.2.1 discuss the next hop selection in the LVAR compared to the Intar. Section 7.2.2 presents the results of LVAR compared to Intar. Hop count, end-to-end delay, and throughput are used as performance metrics for the comparison.

7.2.1 Next hop selection in Intar and LVAR

Next hop selection has a critical influence on the performance of UASNs. The next hop selection in the Intar and LVAR are discussed below.

A. Next hop selection in Intar

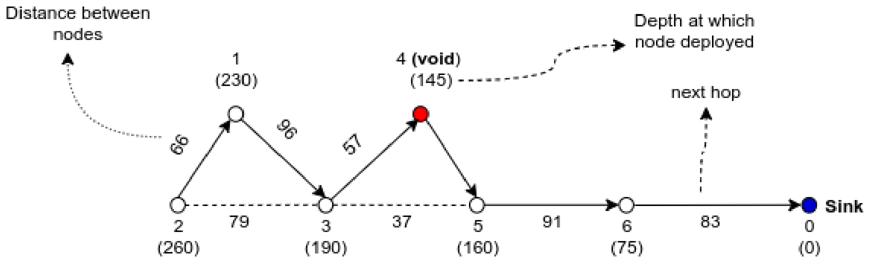

Figure 7.1: Next hop selection in the Intar.

This subsection aims at understanding the next hop selection in Intar with reference to the topology shown in Figure 7.1. Next hop selection in Intar is based on the cost function (CF), which is calculated based on Eq. (4.2) (Javaid et al. 2018).

Table 7.1: Cost of PFNs of node 5 & 3.

Node (j)	**Number of neighbors** ($\lvert N_j \rvert$)	**Distance between** i **&** j ($D_{i,j}$)	**Hop distance from the sink** (H_j)	**Cost** ($CF_{i,j}$)
Cost of PFNs of node 5				
6	2	91	1	$\frac{91}{2 \times 1} = 45.5$
4	2	60	3	$\frac{60}{2 \times 3} = 10$
Cost of PFNs of node 3				
4	2	57	3	$\frac{57}{2 \times 3} = 9.5$
5	3	37	2	$\frac{37}{3 \times 2} = 6.16$

In Intar, a PFN that has the highest cost is selected as the next hop. In the case of non-availability of PFN, Intar selects a neighbor, with a minimum Euclidean distance to the sink, as the next hop among the non-PFN, without applying Eq. (4.2) (Refer Section 4.2.1). Node 6 has only the sink as the PFN & selecting it as the next hop.

As shown in Figure 7.1, nodes 6 and 4 are the candidate PFN at node 5. As shown in Table 7.1, $CF_{5,6}$ and $CF_{5,4}$ (Eq. (4.2)) are calculated, resulting in node 1 as the next hop. However, at node 4 (void node), due to the non-availability of PFN, node 5 (non-PFN) is selected as the next hop. As shown in Figure 7.1, nodes 4 and 5 are the candidate PFN at node 3. As shown in Table 7.1, $CF_{3,4}$ and $CF_{3,5}$ (Eq. (4.2)) are calculated, resulting in node 4 (void node) as the next hop. Similarly, node 1 has only a PFN 3. Thus, node 1 selects 3 as its next hop. Node 2 has PFN 1 and 3. Computing $CF_{2,1}$ and $CF_{2,3}$ (Eq. (4.2)), node 2 selects 1 as its next hop.

B. Next hop selection in LVAR

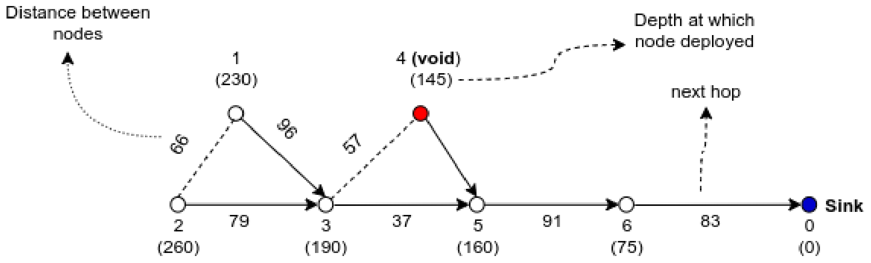

Figure 7.2: Next hop selection in the LVAR.

The next hop selection is shown in the Figure 7.2 (topology is similar to Figure 7.1). Node 6 is directly reachable to node 0 (sink), thus node 6 selects node 0 as its next hop. Node 5 has 3 neighbors, 3, 4, and 6. Node 3 is at a higher depth than 5, and 4 is a void. Node 5 has only a PRN 6. Thus node 5 selects 6 as its next hop. Void node 4 has two neighbors 3 and 5 at a higher depth. In such case node 4 computes $depth\_difference_{4,3}$ and $depth\_difference_{4,5}$ (Eq. 6.3) and they are 45, and 15 respectively. Thus, node 4 selects 5 as its next hop as it is at a lower depth difference. Node 3 has four neighbors 1, 2, 4, and 5. However, nodes 1 and 2 are at a higher depth than 3, and 4 is void. Thus, node 3 has only node 5 PRN, therefore it selects 5 as its next hop. Node 2 has two PRNs, 1 and 3. Using Eq. (7.1), node 2 selects 3 as its next hop, eventually selecting 3 as its next hop, as it is only a PRN.

Table 7.2: Simulation parameters used in LVAR and Intar.

Parameters	**Values**
Number of nodes	7
Transmission range	100 m
Simulation duration	1800 seconds
Packet duration	1 packet/ 14 seconds

7.2.2 Simulation results

The performance of LFVAR is compared with state-of-the-art Intar in terms of hop count, average end-to-end delay, and throughput. The simulations are carried out using the topology shown in Figure 7.1. The simulation parameters are shown in Table 7.2. Results and its analysis of individual performance metrics are elaborated as follows:

A. Hop count

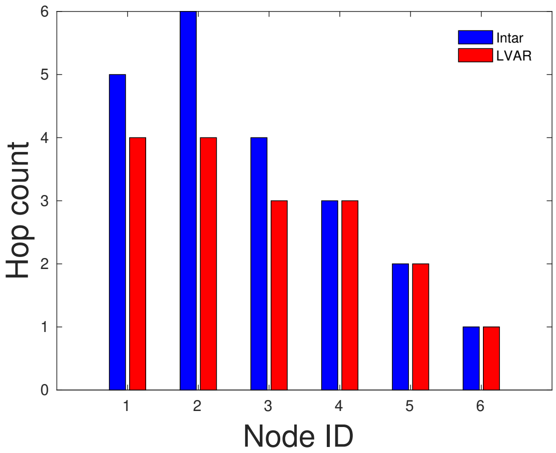

Figure 7.3: Hop count in LVAR and Intar.

As discussed in Section 7.2.1, hop count of individual nodes are shown in Figure 7.3. There are two major reasons for the LVAR resulting in the lower hop count. First, LVAR avoids void nodes through the knowledge of status updates through beacons. Second, the next hop selection strategy is used in the LVAR.

B. Average end-to-end delay

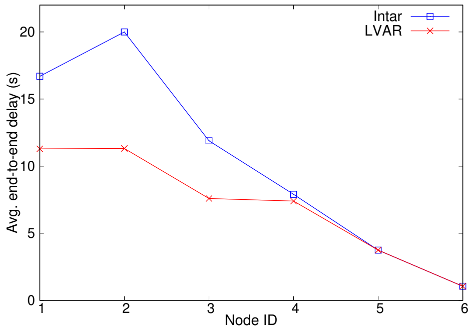

Figure 7.4: Average end-to-end delay in LVAR and Intar.

The average end-to-end delay of individual node id is shown in Figure 7.4. The LVAR results in a lower hop count than Intar. As discussed in Section 6.3.2B., the reduced hop count in the LVAR results in a lower average end-to-end delay.

C. Throughput

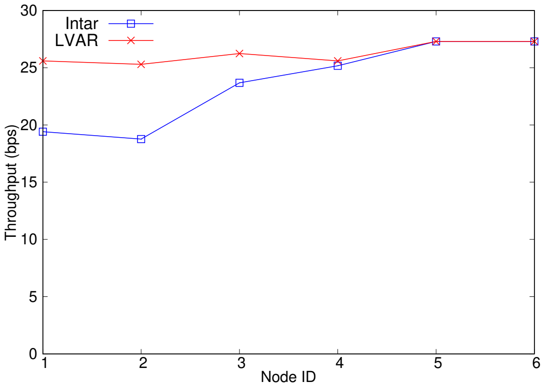

Figure 7.5: Throughput in LVAR and Intar.

One of the reasons for achieving better throughput in LVAR is more packets are delivered to the sink than Intar. As packets travel through fewer intermediate hops, there is a lower chance of packet loss and a higher packet delivery ratio. Thus, LVAR results in higher throughput than Intar, as shown in Figure 7.5.

7.3 SUMMARY

To summarize, this chapter elaborates on the LVAR protocol, which uses the status of neighboring nodes as a primary attribute. Further, hop count, depth, and PDP with neighboring nodes are used as secondary attributes in selecting the next hop. LVAR is implemented in UnetStack. Additionally, hop count, end-to-end delay, and throughput are used as performance metrics to compare LVAR with state-of-the-art Intar. The result shows that LVAR outperforms Intar in these metrics.

The following chapter presents the proposed Cluster-based Multi-Attribute Routing (CMAR) protocol for UASNs.

CHAPTER 8

DESIGN AND EVALUATION OF CLUSTER-BASED MULTI-ATTRIBUTE ROUTING (CMAR) PROTOCOL

This chapter elaborates on the Cluster-based Multi-Attribute Routing (CMAR) protocol for UASNs, submitted to Springer's Wireless Personal Communications. The CMAR is sender-based, an opportunistic underwater routing protocol. In CMAR, every node (except the sink) uses its neighboring node's status (void or normal) as a primary attribute, and advancement and Packet Delivery Probability (PDP) with neighboring nodes deployed at lower depths, as secondary attributes, in selecting the next hop(s). The detailed design of CMAR is elaborated in Section 8.1. Simulation of the CMAR is carried out using MATLAB. Further, its performance is compared with the state-of-the-art HydroCast (Noh et al. 2015), in terms of the number of forwarding nodes, number of clusters formed, expected packet advancement, number of times void nodes are selected as part of the forwarding set, and transmission reliability in Section 8.2.

8.1 DESIGN OF THE CMAR PROTOCOL

This section elaborates on the design of the CMAR protocol. The CMAR consists of the following two phases:

- **Phase-I: Score computation and cluster formation**

  In CMAR, as first step every node i determines their PRN_i (Refer Eq. (6.1), Section 6.1.1). Additionally, node i computes the score of the nodes in PRN_i. The scores indicate the preference of the nodes in becoming part of the candidate forwarding set of source/forwarder i. In CMAR, the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) (Hwang and Yoon 1981) is used to compute the score of nodes in the PRN_i, and scores of nodes in PRN_i are used in the cluster(s) formation. One of the reasons for selecting a candidate forwarding set in the form of cluster(s) is to avoid duplicate data transmission and hidden-node problems. When node i has data, it forwards it to its selected cluster(s). More details of TOPSIS, score computation of nodes in PRN_i, and cluster formation in the CMAR is explained in Section 8.1.1.

- **Phase-II: Data forwarding**

  As nodes in the candidate forwarding set (a cluster) receive the data, further forwarding is done by coordination among them. The nodes receive the data in the respective cluster and achieve coordination by using hold time. More details on the data forwarding approach used in CMAR is explained in Section 8.1.2.

8.1.1 Phase-I: Score computation and clustering

This section elaborates on the details of the TOPSIS. Also, the score computation of nodes in PRN_i using TOPSIS, followed by cluster formation in CMAR, is discussed in detail.

A. TOPSIS

TOPSIS is the well-known and widely used Multi-Attribute Decision Making (MADM) technique in different fields of engineering (Behzadian et al. 2012; Chandavarkar and Guddeti 2016). In TOPSIS, attributes of nodes in the PRN_i are either benefit or cost attributes. The benefit attributes, in which the higher the attribute value, the better the priority (rank) and the more preferred. In the case of cost attribute, the higher the attribute value, poor is the priority (rank) and is least preferred in the selection. Benefit attributes in the UASNs routing are advancement, PDP, Signal-to-Noise Ratio (SNR), the remaining energy of neighboring nodes, etc., and the cost attributes are hop count, distance to the sink, etc.

Score computation using TOPSIS

In the CMAR, PRN_i is determined using the Eq. (6.1) and attributes of nodes in PRN_i act as the input to the TOPSIS. The steps involved in computing scores of nodes in PRN_i by the source/forwarder i are elaborated as follows:

Step 1: Construction of decision matrix of nodes in PRNs

This step involves the creation of a decision matrix representing n attributes (advancement and PDP) of m nodes of PRN_i. Nodes are represented in rows, and attributes of the corresponding node are represented in columns. The intersection of x^{th} node with y^{th} attribute is represented by $DM^i[x][y]$. Thus, decision matrix of nodes in PRN_i are represented as per Eq. (8.1).

$$(DM^i[x][y])_{m\times n} \tag{8.1}$$

where $DM^i[x][y]$ holds the y^{th} attribute of x^{th} node in PRN_i, $m = |PRN_i|$, and n is the number of attributes to be considered, $x \in \{1, 2, 3, ...m\}$, and $y \in \{1, 2, 3, ...n\}$

Step 2: Attribute normalization

This step involves normalization of decision matrix ($(DM^i[x][y])_{m\times n}$). Normalization of attributes involves the transformation of the attributes from dimension to dimensionless and put into a common scale, so that they are comparable (Vafaei et al. 2018). In CMAR, the sum normalization method is used to normalize the attributes. The Eq. (8.2) indicates, normalization of attribute at y^{th} column, corresponding to node at x^{th} row of the matrix given in Eq. (8.1).

$$R^i[x][y] = \frac{DM^i[x][y]}{\sum_{x=1}^{m} DM^i[x][y]} \tag{8.2}$$

The normalized matrix is represented as shown in Eq. (8.3).

$$(R^i[x][y])_{m\times n} \tag{8.3}$$

Step 3: Weight computation

Normalized matrix is obtained from the decision matrix of source/forwarder i. The attributes used in the decision matrix or normalized matrix do not have equal importance. Every attribute has to have a different priority. The weight of the attributes indicates the importance or priority. Thus, the weights of each attribute need to be determined. There are many techniques to compute the weights; some of them are Simplified and Improved Analytical Hierarchy Process (SI-AHP) (Chandavarkar and Guddeti 2015), Entropy, Variance (Wang and Kuo 2012) etc. However, the Entropy weighting method is appropriate for the TOPSIS when the Sum normalization method is used (Chandavarkar and Guddeti 2016). Thus CMAR uses the Entropy weight method to compute the weights of the attribute. The computation of weight of the attribute at y^{th} column, $W^i[y]$ using Entropy weight method is shown in Eq. (8.4).

$$W^i[y] = \frac{1 - E^i[y]}{\sum_{y=1}^{n}(1 - E^i[y])} \tag{8.4}$$

where $E^i[y]$ is the entropy of attribute at y^{th} column, and computed using Eq. (8.5).

$$E^i[y] = -\frac{\sum_{x=1}^{m}(R^i[x][y] * ln(R^i[x][y])}{ln(m)} \tag{8.5}$$

The weight matrix is represented as shown in Eq. (8.6).

$$(W^i[y])_{1\times n} \tag{8.6}$$

Step 4: Computation of Weight-Cost matrix

The Weight-Cost matrix is computed using a normalized matrix and weights of the attribute. The Eq. (8.7) shows the Weight-Cost computation of y^{th} attribute of the node at x^{th} row.

$$WC^i[x][y] = R^i[x][y] * W^i[y] \tag{8.7}$$

The Weight-Cost matrix is represented as shown in Eq. (8.8).

$$(WC^i[x][y])_{m\times n} \tag{8.8}$$

Step 5: Computation of positive and negative ideal solutions for each attribute:

$$A^{i+} = \{A[1]^+, A[2]^+, A[3]^+, ..., A[n]^+\} \text{ where } A[y]^+ = \{MAX(WC^i[x][y]),\ \forall\, y \in \text{ benefit attributes}\} \,\&\&\, MIN(WC^i[x][y]),\ \forall\, y \in \text{ cost attributes}\} \tag{8.9}$$

$$A^{i-} = \{A[1]^-, A[2]^-, A[3]^-, ..., A[n]^-\} \text{ where } A[y]^- = \{MIN(WC^i[x][y]),\ \forall\, y \in \text{ benefit attributes } \&\&\, MAX(WC^i[x][y]),\ \forall\, y \in \text{ cost attributes}\} \tag{8.10}$$

The positive-ideal solution maximizes the benefit and minimizes the cost criteria. The positive ideal solution $A[y]^+$ of attribute y is calculated as per Eq. (8.9). The negative-ideal solution maximizes the cost criteria and minimizes the benefit criteria. The negative ideal solution $A[y]^-$ of attribute y is calculated as per Eq. (8.10). In short, the positive-ideal solution is made up of all the best possible criteria values. In contrast, the negative-ideal solution comprises of all the worst possible criteria values.

Step 6: Finding Euclidean distance from positive and negative ideal solutions for each node in PRN

Euclidean distance of node at x^{th} row, from positive ideal solution $ED^i[x]^+$ is obtained by using Eq. (8.11).

$$ED^i[x]^+ = \sqrt{\sum_{y=1}^{n}(WC^i[x][y] - A[y]^+)^2} \tag{8.11}$$

Matrix consists of Euclidean distance from the positive ideal solutions are represented by Eq. (8.12)

$$(ED^i[x]^+)_{m\times 1} \tag{8.12}$$

Euclidean distance of node at x^{th} row, from negative ideal solution $ED^i[x]^-$ is obtained by using Eq. (8.13).

$$ED^i[x]^- = \sqrt{\sum_{y=1}^{n}(WC^i[x][y] - A[y]^-)^2} \tag{8.13}$$

Matrix consists of Euclidean distance from the negative ideal solutions are represented by Eq. (8.14)

$$(ED^i[x]^-)_{m\times 1} \tag{8.14}$$

Step 7: Score computation of each node in PRNs:

Score of node correspond to x^{th} row is computed by the Eq. (8.15).

$$Score^i[x] = \frac{ED^i[x]^-}{ED^i[x]^- + ED^i[x]^+} \tag{8.15}$$

The matrix consists of scores of all nodes represented, as shown in Eq. (8.16). The node that has a high score indicates higher priority and is more preferred.

$$(Score^i[x])_{m\times 1} \tag{8.16}$$

In the state-of-the-art HydroCast, which is used to compare with CMAR, the suitability of neighboring nodes is measured by computing the Normalized Advancement (NADV) of the nodes present at the lower depth than the source/forwarder i (Noh et al. 2015). The NADV is calculated by Eq. (8.17) (Lee et al. 2005b; Noh et al. 2015).

$$NADV_{i,j} = ADV_{i,j} \times P_{i,j} \tag{8.17}$$

where $NADV_{i,j}$ is NADV of neighbor j deployed at lower depth with reference to source/forwarder i. $ADV_{i,j}$ is advancement of the node j with reference to source/forwarder i (Eq. (1.1)) and it is the depth difference between the source/forwarder i with node j. $P_{i,j}$ is PDP between source/forwarder i and j (Eq. (7.6)) (More details of HydroCast are given in Section 2.3).

B. Cluster formation

Algorithm 3 Cluster formation in CMAR at the node i.

```
Initialization:
Number of clustered nodes of $i$ ($NC_i$) $\rightarrow$ 0
$PRN_i = \{j_1, j_2, j_3, ...j_p\}$, where priority of $j_1 > j_2 > j_3 > ... > j_p$
Threshold number of nodes to be clustered ($Th$)
$k^{th}$ cluster of node $i$ ($Cluster_{i,k}$); $k \rightarrow 1$
First node in the cluster $CH$
Distance between $CH$ and $node$ ($D_{CH,node}$)
$l^{th}$ index of the node in $PRN_i$ is not included in any of the cluster and it is represented by $V[l]$ = 0
End of Initialization
for ($l = 1 \; to \; |PRN_i|$) do
  $V[l]$ = 0
end for
while ($NC_i < Th$ && $|NC_i| \neq |PRN_i|$) do
  for ($l = 1 \; to \; |PRN_i|$) do
    if ($V[l] == 0$) then
      $Cluster_{i,k} \leftarrow PRN_i[l]$
      $CH \leftarrow PRN_i[l]$
      $V[l] = 1$
      $break$
    end if
  end for
  for ($l = (l+1) \; to |PRN_i|$) do
    $node = PRN_i[l]$
    if ($D_{CH,node} \leq \frac{Range}{2}$ && $V[l] == 0$) then
      $Cluster_{i,k} \leftarrow Cluster_{i,k} \cup \{node\}$
      $V[l] = 1$
    end if
  end for
  $k = k + 1$
  $NC_i = NC_i + |Cluster_{i,k}|$
end while
```

Once the scores of PRN_i are computed, the next step in CMAR is determining the candidate forwarding set, by forming cluster(s) of suitable nodes. The clustering approach used in the CMAR requires only the Threshold (Th) number of nodes from PRN_i to be clustered. Once Th number of nodes are clustered, the clustering process is terminated.

The process of clustering in the CMAR is described in Algorithm 3. The clustering process is initiated by the source/forwarder i (line no. 1 - 9). The clustering process begins with a node that has the highest priority and is not a part of any cluster; that node is referred to as CH. Additionally, node CH is marked as included in the cluster (line no. 14 - 21). Further, i obtains the neighbors of node CH, which are at Euclidean distance $\leq (\frac{Range}{2})$, where $Range$ is the coverage area. Finally, node CH, along with common nodes (not part of any cluster) in the PRN_i and neighbors of node CH at the Euclidean distance $\leq (\frac{Range}{2})$ are included in the cluster (line no. 22 - 28). The cluster formation process is repeated until the Th number of nodes are included in cluster (line no. 13). In this way, node i generates its clusters ($Cluster_{i,1}$, $Cluster_{i,2}$, ..., $Cluster_{i,k}$).

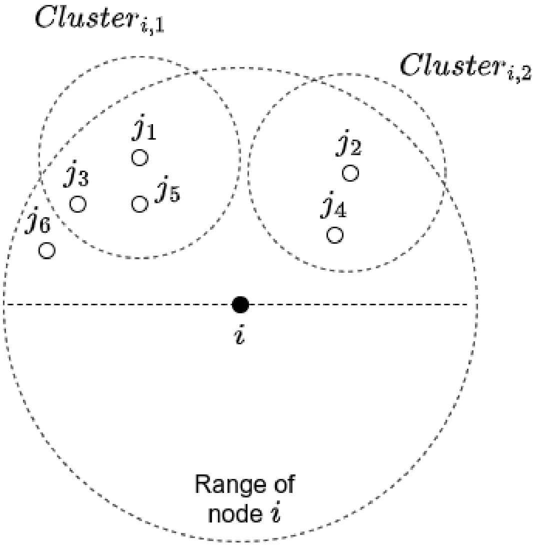

Figure 8.1: Example of the cluster formation of node i in the CMAR with Th = 4.

The cluster formation process with Th = 4, is explained with the example shown in Figure 8.1. The node i is the source/forwarder node. The node i computes its PRN_i by using Eq. 6.1 and subsequently determines their priority (or score), assuming that

$PRN_i = \{j_1, j_2, j_3, j_4, j_5, j_6\}$, are arranged in decreasing order of their priority using TOPSIS. Further, using the Algorithm 3, node i selects the highest priority node j_1 as the CH and includes it in its first cluster ($cluster_{i,1}$). Additionally, node i finds nodes that are common in its PRN_i and nodes that are at a Euclidean distance of ($\frac{Range}{2}$) of CH (using two-hop information). Accordingly, j_3 and j_5 are included in $Cluster_{i,1}$. Moreover, no more nodes exist in the ($\frac{Range}{2}$) of j_1 and $Cluster_{i,1} = \{j_1, j_3, j_5\}$. However, only 3 nodes are included in the cluster, and Th = 4; thus, further clustering continues. Nodes j_2, j_4, j_6 are not part of any cluster, however, among these nodes, j_2 has the highest priority cluster. Further, node i selects j_2 as CH in second cluster ($Cluster_{i,2}$) and subsequently includes j_4 in it. Further, no more nodes exists in the ($\frac{Range}{2}$) of j_2 and $Cluster_{i,2} = \{j_2, j_4\}$. Thus, total number of nodes included in cluster is 5 (in both $Cluster_{i,1}$ and $Cluster_{i,2}$), satisfying Th = 4.

The state-of-the-art HydroCast routing protocol, which is used to compare with CMAR, uses Normalized Advancement (NADV) of neighbors deployed at the lower depth. Also, NADV is used to create a cluster(s). The major difference between the CMAR and HydroCast is the inclusion of neighbor nodes into the cluster(s). In CMAR, only the PRNs of the source/forwarder node are considered; thereby, it avoids the inclusion of void nodes into the cluster. Further, in CMAR, only Th number of nodes are clustered. However, in HydroCast status of the neighbors is not considered. Void nodes can also be part of the clusters, and subsequently, they may become part of the candidate forwarding set. Further, HydroCast requires all neighbor nodes to be clustered, and the Expected Packet Advancement (EPA) of individual clusters is computed by using Eq. (8.18).

$$EPA(S_l) = \sum_{j=1}^{l} ADV_{i,j} P_{i,j} \prod_{k=0}^{j-1} (1 - P_{i,k}) \tag{8.18}$$

where S_l is the set nodes of cluster having priorities $n_1 > n_2 > \cdots > n_l$, $ADV_{i,j}$ and $P_{i,j}$ are the advancement and packet delivery probability of node j, with reference to source/forwarder i. Finally, a cluster that has the highest EPA is selected as its candidate forwarding set. In conclusion, in most cases, HydroCast results in the formation of more clusters than CMAR. Thus, HydroCast increases the complexity of cluster formation.

However, only Th number of nodes in the CMAR are clustered. The Th number of nodes is adjusted based on the underwater condition/application.

8.1.2 Phase-II: Forwarding received data

This section elaborates on the mechanism used in forwarding the received data in the CMAR. Node i forwards the data to its first cluster ($Cluster_{i,1}$). In the case of unsuccessful delivery of data to the first cluster, the data is forwarded to the second cluster ($Cluster_{i,2}$), and so on. Once data is delivered to the cluster, further forwarding of data by nodes of the respective cluster, is coordinated using hold time computation.

Every node j belonging to the respective cluster, which receives the data, computes its hold time T^j_{HOLD}. Hold time defines the duration of time the packet is held by the node j before it is further forwarded to their cluster. If a node j overhears the transmission of the same data packet during the hold time, it discards the packet from its buffer, to prevent duplicate transmission. In case the transmission of a packet is not overheard before the expiry of the hold time T^j_{HOLD}, no other nodes forwards the same data packet. Thus node j forwards the packet to its cluster(s). The hold time T^j_{HOLD} computation of node j is elaborated as follows:

A. Hold time computation

The hold time computation by every node j in the respective cluster, which receives the data, is computed using Eq. (8.19).

$$T^j_{HOLD} = T^j_{PAUSE} + T^j_{OVERHEAR} + T^j_{TRANS} + T^j_{PROC} \tag{8.19}$$

where T^j_{HOLD} is the hold time of the node j in the corresponding cluster, T^j_{PAUSE} is the pause period of the node j, which is used to achieve synchronization among all nodes in the cluster. Further, lower priority nodes, within the cluster, delay their packet forwarding by overhearing time $T^j_{OVERHEAR}$, transmission time T^j_{TRANS}, and processing time T^j_{PROC}. The pause time of node j (T^j_{PAUSE}) is defined as in Eq. (8.20).

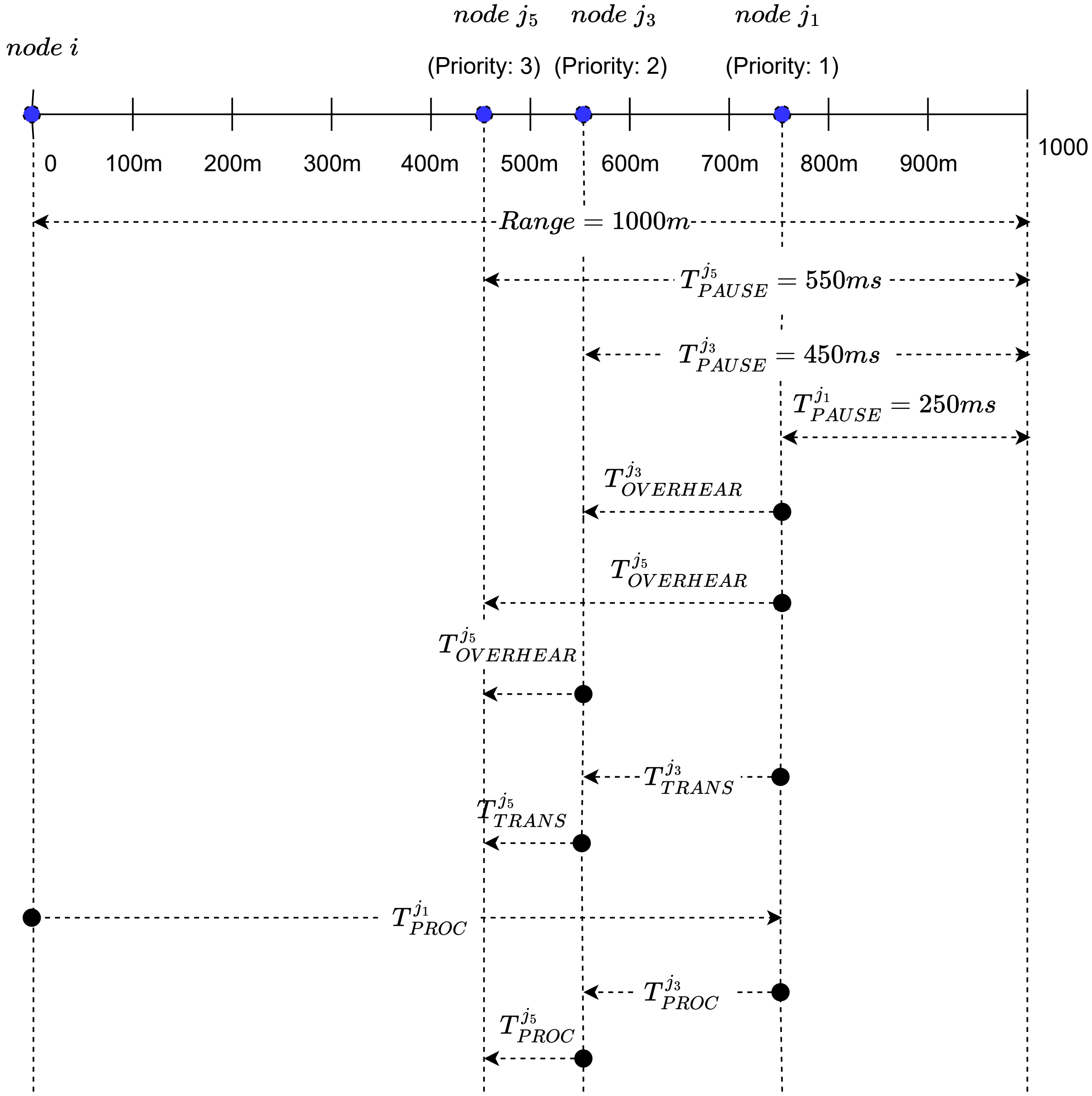

Figure 8.2: Hold time computation for node $j_1(P_{j_1} = 1), j_3(P_{j_3} = 2), j_5(P_{j_5} = 3)$.

$$T^{j}_{PAUSE} = \frac{Range - D_{i,j}}{V} \tag{8.20}$$

where V is the propagation speed of the acoustic signal underwater. The total overhearing time of the node j ($T^{j}_{OVERHEAR}$) is calculated as per the Eq. (8.21).

$$T^{j}_{OVERHEAR} = (P_j - 1) \times \frac{Range}{V} \tag{8.21}$$

where P_j is the priority of the node j in the corresponding cluster. Further, total transmission time of the node j (T^j_{TRANS}) is calculated as per the Eq. (8.22).

$$T^j_{TRANS} = (P_j - 1) \times T_{TRANS} \tag{8.22}$$

where T_{TRANS} is the transmission time of a packet at the node. The total processing time of node j (T^j_{PROC}) is calculated as per the Eq. (8.23).

$$T^j_{PROC} = P_j \times T_{PROC} \tag{8.23}$$

Table 8.1: Hold time calculation of nodes with reference to Figure 8.2 with T_{TRANS} = 0.5 sec, T_{PROC} = 0.25 sec, and V = 100 m/s.

Node id (j)	P_j	T^j_{PAUSE}	$T^j_{OVERHEAR}$	T^j_{TRANS}	T^j_{PROC}	T^j_{HOLD}
j_1	1	2.5	0	0	$1 \times T_{PROC}$	2.75 sec
j_3	2	4.5	10	$1 \times T_{TRANS}$	$2 \times T_{PROC}$	15.5 sec
j_5	3	5.5	20	$2 \times T_{TRANS}$	$3 \times T_{PROC}$	27.25 sec

where T_{PROC} is the packet processing delay at the node. As per the Eq. (8.19) through (8.23), a higher priority node has a lower hold time, and a lower priority node has a higher hold time.

Figure 8.2 presents the scenario of hold time computation of node $j_1(P_{j_1} = 1)$, $j_3(P_{j_3} = 2)$, $j_5(P_{j_5} = 3)$ belongs to the cluster of source/forwarder i (as per Figure 8.1), when the $Range$ is 1000m. Table 8.1 presents the hold time calculation of nodes in the cluster.

B. Forwarding of data

Once node j receives the packet, it has to forward it further, when the hold time T^j_{HOLD} expires. However, there are some situations in which the Th number of nodes of source/forwarder j may not be present in a single cluster and are distributed among

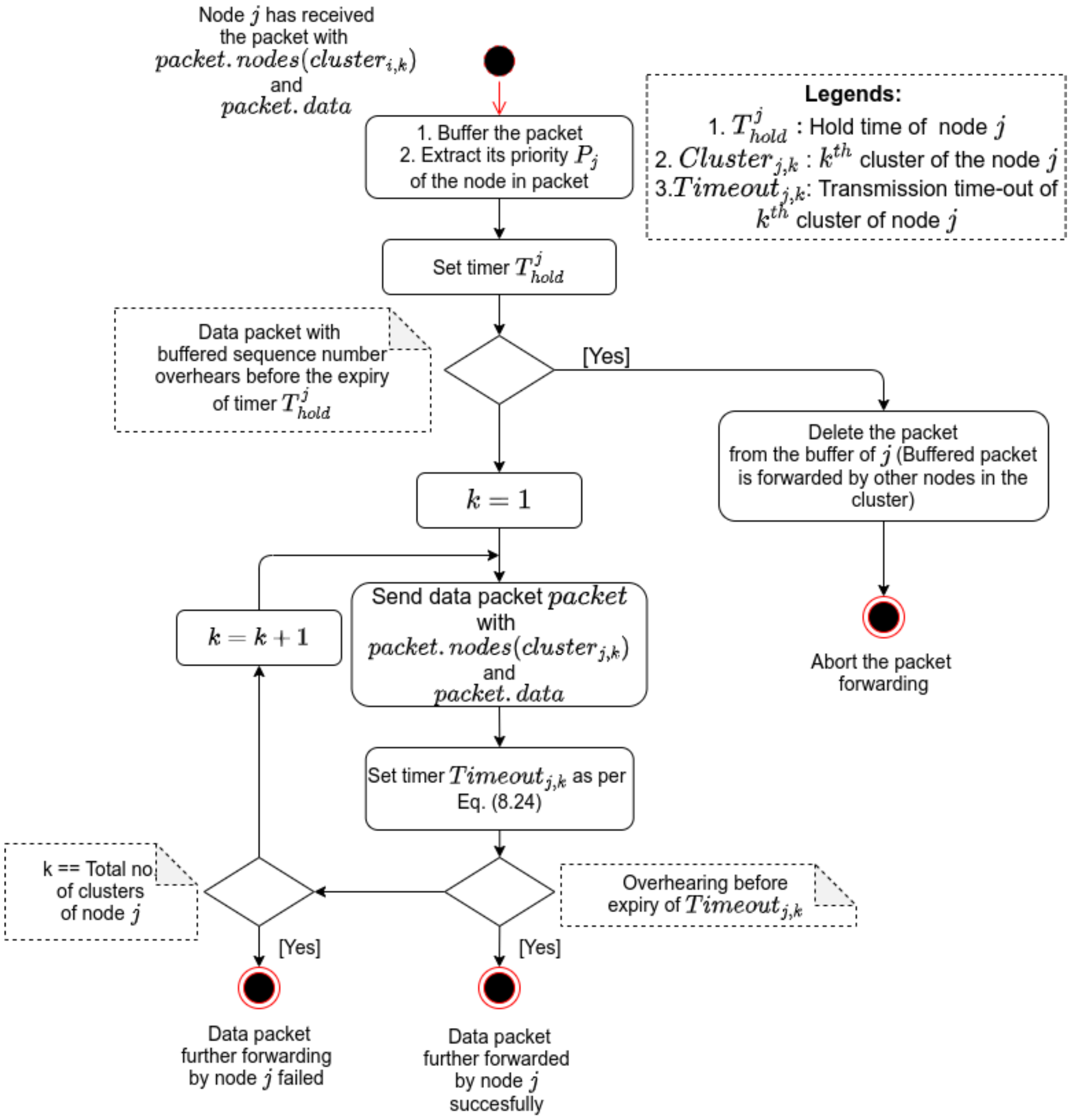
Node j has received the packet with $packet.nodes(cluster_{i,k})$ and $packet.data$
1. Buffer the packet
2. Extract its priority P_j of the node in packet
Legends:
1. T^j_{hold} : Hold time of node j
2. $Cluster_{j,k}$: k^{th} cluster of the node j
3.$Timeout_{j,k}$: Transmission time-out of k^{th} cluster of node j
Set timer T^j_{hold}
Data packet with buffered sequence number overhears before the expiry of timer T^j_{hold}
[Yes]
Delete the packet from the buffer of j (Buffered packet is forwarded by other nodes in the cluster)
$k = 1$
Abort the packet forwarding
$k = k + 1$
Send data packet $packet$ with $packet.nodes(cluster_{j,k})$ and $packet.data$
Set timer $Timeout_{j,k}$ as per Eq. (8.24)
k == Total no of clusters of node j
Overhearing before expiry of $Timeout_{j,k}$
[Yes]
[Yes]
Data packet further forwarding by node j failed
Data packet further forwarded by node j succesfully

Figure 8.3: Packet reception and forwarding by node j in CMAR.

more than one cluster. In such cases, if nodes in the first cluster fail to receive a data packet, the forwarder j forwards it to the next cluster. The overall process of the data packet forwarding is shown in Figure 8.3. The packet forwarding scenarios are explained as follows:

- **Scenario 1: All Th nodes are in single cluster of source/forwarder j**
  If Th nodes are present in the single cluster, the forwarder j forwards the data packet to its cluster and upon receiving, a node in the cluster further forwards the data packet. If no nodes in the cluster receive the data, the transmission fails.

- **Scenario 2: Th nodes are distributed among more than one cluster of source/-forwarder j**

  There are situations where Th nodes of j are distributed among more than one cluster. In such cases, data packet forwarding takes place in the following way. Initially, the data packet is sent to the nodes of the first cluster ($Cluster_{j,1}$). The forwarding node j confirms the data reception by the cluster nodes using overhearing for further forwarding. Therefore, the forwarder j waits for the $Timeout_{j,k}$ period to overhear further forwarding. The $Timeout_{j,k}$ is given in Eq. (8.24).

  $$Timeout_{j,k} = 2 \times \frac{R}{V} + T^{l}_{HOLD} \tag{8.24}$$

  where T^{l}_{HOLD} is the hold time of the least priority node l of k^{th} cluster of j. During $Timeout_{j,k}$ duration, forwarding node j fails to overhear the further forwarding by any of the nodes of $Cluster_{j,1}$, and it is sent to its next cluster ($Cluster_{j,2}$). The same process continues until the forwarding node j overhears the transmission or all clusters of j fail to forward the data further.

8.2 SIMULATION AND EVALUATION OF THE CMAR PROTOCOL

This section discusses the simulation and evaluation of CMAR in detail. Simulation of CMAR is carried out using MATLAB. Section 8.2.1 elaborates on the simulation setup used to simulate CMAR. Further, its performance is compared with the state-of-the-art HydroCast (Noh et al. 2015) in terms of the number of forwarding nodes, number of clusters formed, expected packet advancement, number of times void nodes are selected as part of the forwarding set, and transmission reliability, in Section 8.2.2. The CMAR is flexible in using any number of attributes. However, CMAR uses similar attributes in the state-of-the-art HydroCast, so the comparison is compatible.

8.2.1 Simulation setup

Table 8.2: Simulation parameters used in the evaluation of the CMAR.

Parameters	**Value**
No. of neighbors	10
No. of void nodes	2
Communication range	1000 m
Variation of depth parameter	200 m to 800 m in the step of 200 m
Packet size	1600 bits

The simulation parameters are listed in Table 8.2. There is a single source and 10 neighboring nodes in the simulation. Out of 10 neighboring nodes, any two nodes are selected as void nodes and remain so until the completion of the simulation. CMAR considers two attributes, the advancement and PDP of nodes in PRNs of the source/-forwarder. The advancement and PDP is varied to perform exhaustive simulations, and 2,097,152 combinations of inputs are generated and executed.

8.2.2 Simulation results

Table 8.3: Number of nodes in the selected cluster(s) in the CMAR and HydroCast.

No. of nodes in the selected cluster(s)[*]	**CMAR with Th number of nodes**						**HydroCast**
	$Th = 3$	$Th = 4$	$Th = 5$	$Th = 6$	$Th = 7$	$Th = 8$	
	% of input combinations						
1	0	0	0	0	0	0	0.04
2	0	0	0	0	0	0	9.21
3	43.85	0	0	0	0	0	36.77
4	34.55	48.48	0	0	0	0	33.31
5	16.31	34.78	55.28	0	0	0	15.06
6	4.54	13.77	34.20	65.33	0	0	4.51
7	0.68	2.73	9.54	30.25	79.22	0	0.93
8	0.04	0.21	0.97	4.41	20.77	100	0.12
9	0	0	0	0	0	0	0.01
10	0	0	0	0	0	0	0.00

[*] In the HydroCast, total nodes in the finally selected cluster, whereas nodes in the other clusters are not included. In the case of CMAR, other nodes are not in the cluster.

The CMAR is compared with state-of-the-art HydroCast in terms of the number of forwarding nodes, number of clusters formed, expected packet advancement, number of times void nodes are selected as part of the forwarding set, and transmission reliability. These are elaborated as follows:

A. Number of forwarding nodes selected

Table 8.4: Various scenarios of cluster formation in CMAR with $Th = 3$.

Scenario	No. of nodes in selected cluster	Cluster 1	Cluster 2	Cluster 3
1	3	3	-	-
2	3	1	1	1
3	3	1	2	-
4	3	2	1	-
5	4	4	-	-
6	4	1	1	2
7	5	1	1	3
8	6	1	1	4
9	7	1	1	5
10	8	1	1	6

The number of forwarding nodes selected in the CMAR and HydroCast for a percentage of input combinations is shown in Table 8.3. The number of forwarding nodes selected indicates the potential of the source/forwarder node to forward the data successfully. As more nodes are present in the selected cluster, there are high chances of successful data forwarding. In the case of CMAR with $Th = 3$ through 8, there is a high possibility of four or more nodes being selected compared when compared to HydroCast. It shows that CMAR provides increased chances of successful packet forwarding from the source/forwarder to its neighboring nodes. As shown in Table 8.3, for example, in the CMAR, even though $Th = 3$, the number of nodes in the selected cluster(s) is 3 or even more. When $Th = 3$, a maximum of three clusters can be formed, considering one node each in the three clusters, and a minimum of one cluster is created when all nodes are in the same cluster.

Various scenarios for the selection of nodes in the case of $Th = 3$ are shown in Table 8.4. In case the number of nodes in the cluster(s) is three, there are four possible scenarios. In scenario 1, all three nodes are in the first cluster, and the number of nodes chosen fulfills the Th requirements. In scenario 2, the first cluster has only one node; therefore, it will continue clustering until a minimum of three nodes are included in cluster(s). Even the second cluster has one node; thus, it looks for the third cluster and has one node. In this way, three nodes are distributed in 3 different clusters. Similarly, in scenarios 3 and 4, the first cluster has one node, the second cluster has two nodes, and vice-versa. The important aspect of CMAR is even though $Th = 3$, in some scenarios, more than three nodes are included in the cluster. Consider scenario 5, in which four nodes are selected, and this is because the first cluster itself has four nodes. In scenario 6, the first two clusters have one node each. Additionally, the third cluster has two nodes. Thus four nodes are chosen from three clusters even though $Th = 3$. Likewise, even though $Th = 3$, there are chances of even 5, 6, 7, and 8 nodes being selected, as shown in scenarios 7 through 10. However, there is no possibility of selecting 9 or 10 forwarding nodes since CMAR completely avoids void nodes in the cluster (when two nodes are void nodes).

B. Number of clusters formed

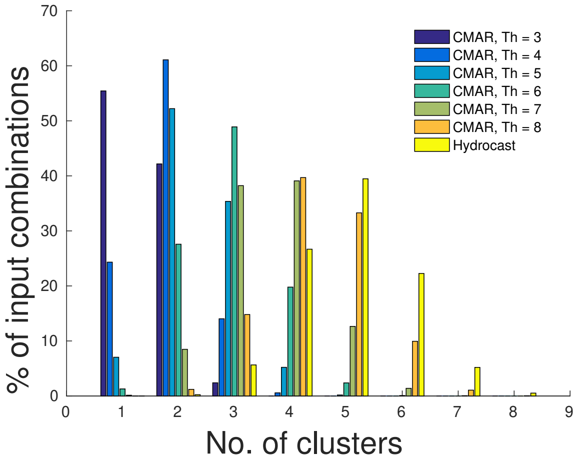

Figure 8.4: Comparison of CMAR and HydroCast w.r.t. chances of forwarding packets.

The number of clusters formed in the CMAR for Th ranges from 3 to 8 nodes and HydroCast is shown in Figure 8.4. In almost 98% of input combinations, forwarding nodes are accommodated within 1 or 2 clusters for CMAR with $Th = 3$. Similarly, in 85% of input combinations, CMAR with Th = 4 is accommodated in just 1 or 2 clusters. However, in just 6% of input combinations, HydroCast is accommodated with 1 to 3 clusters, and in 65% of input combinations, HydroCast forms 5 to 8 clusters.

The CMAR clustering process is initiated from a node with the highest score in the PRNs. Once a cluster is formed, nodes present in it become part of the candidate forwarding set, and the clustering process stopped once Th nodes are reached. Thus in CMAR, all nodes of PRN_i need not be clustered. It results in a lower number of clusters. However, in the case of HydroCast, clustering is done until every node deployed at a lower depth than the source/forwarder is clustered. Further, by computing the EPA of each cluster, the best cluster is shortlisted, and nodes present in it are part of the forwarding nodes.

In summary, HydroCast forms more number of clusters even though data is forwarded to only one cluster. It results in increased overhead in the formation of clusters. However, CMAR results in a lower number of clusters and depends on the configurable value of Th, which is based on underwater conditions.

C. Average Expected Packet Advancement (EPA)

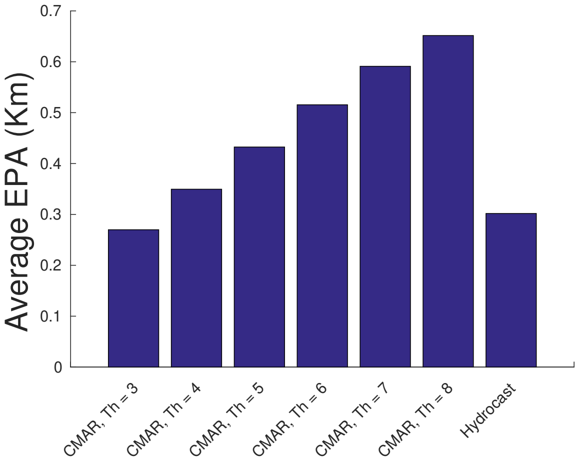

Figure 8.5: Average EPA of CMAR and HydroCast.

EPA is the normalized sum of advancements made by the selected nodes in the cluster. EPA describes how much closer a packet can be forwarded to the sink during transmission. The EPA is computed using Eq. (8.18). EPA of CMAR with $Th = 3$ to 8 and HydroCast is given in Figure 8.5. CMAR results in a higher average EPA, when compared to HydroCast, due to the higher number of forwarding nodes.

D. Nodes at priority 1 and 2

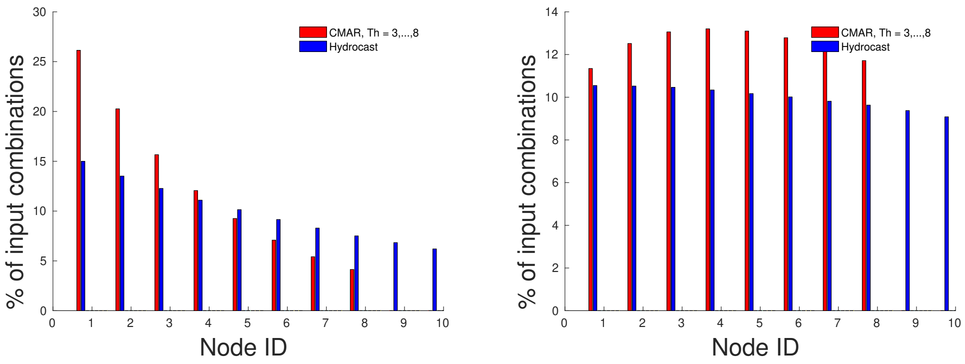

Figure 8.6: Percentage (%) input combinations, Nodes at (a) Priority 1 (left) (b) Priority 2 (right).

Void nodes, as part of the forwarding nodes, result in reduced performance. The Figure 8.6(a) and (b), shows that the percentage of input combinations nodes are at priority 1 and 2. Node id 9 and 10 are not selected as priority 1 or 2 nodes in CMAR. However, in the case of HydroCast, nodes id 9 and 10 are selected as priority 1 & 2 nodes even though they are void nodes. In conclusion, the CMAR prevents void nodes from becoming cluster members. Thereby void nodes are not selected as forwarding nodes. However, HydroCast do not avoid void nodes. The involvement of void nodes during routing has severely impacted the network's performance.

E. Transmission reliability

Figure 8.7 shows the transmission reliability of the CMAR and HydroCast. As CMAR reaches the sink through normal nodes, results in a lower number of hops, having higher chances of delivering the data. Thus, it results in higher transmission reliability. However, in the HydroCast even void nodes are part of the path to reach the sink. It results in a longer path to reach the sink, causing more chances of losing a

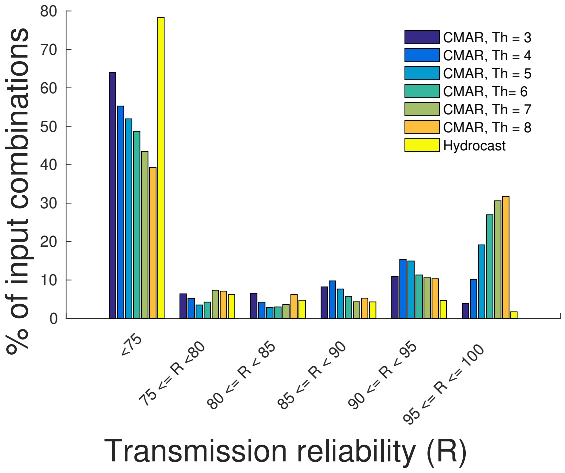

Figure 8.7: Transmission reliability vs percentage of input combinations.

packet than CMAR.

8.3 SUMMARY

This chapter of the thesis presents the CMAR protocol that uses the MADM technique to prioritize the neighbor nodes. Further, a candidate forwarding set is determined using node priority and a novel clustering approach. The simulations shows that in almost 98% of input combinations, forwarding nodes are accommodated within 1 or 2 clusters for CMAR with $Th = 3$. Similarly, in 85% of input combinations, CMAR with Th = 4 is accommodated in just 1 or 2 clusters. However, in just 6% of input combinations, HydroCast is accommodated with 1 to 3 clusters, and in 65% of input combinations, HydroCast forms 5 to 8 clusters. Additionally, the void node is not a part of the candidate forwarding set in CMAR.

.

CHAPTER 9

CONCLUSIONS AND FUTURE SCOPE

Avoiding void node(s) and selecting the best next hop is one of the critical factors that significantly influences the performance of underwater communication. This research aims at designing and developing void-aware routing protocols for underwater. Three objectives (identification of suitable attributes, designing a void-aware routing algorithm using the MADM approach, and deployment, simulation, & evaluation of the proposed algorithm) are perceived as a part of this research work, resulting in unique algorithms, which work on different sets of attributes.

9.1 SUMMARY OF CONTRIBUTIONS

A summary of the contributions of this thesis is listed below:

- Enhanced Void-Aware Routing (E-VAR) protocol has been designed using status (void or normal) and Euclidean distance of neighbor to the sink as attributes. Further, E-VAR has been compared with state-of-the-art Backward-forwarding and Intar to demonstrate void avoidance and looping through the manual deployment in MATLAB. Additionally, in random deployment, 2.87% and 34% improvement has been observed in E-VAR, compared with the other two respectively, regarding the number of reachable source nodes to the sink. Also, 22.60% and 57.30% improvement in terms of average hop count is recorded in E-VAR compared to the other two, respectively.

- The Link Quality-based Routing Protocol (LQRP) uses two-hop link quality to decide the next hop. Appropriate weight has been assigned to link quality between the node to its neighbor and neighbor to its best next hop. Further, by computing the cost, the next hop is determined. The SNR is used to measure the link quality. The SNR is computed by using the physical characteristics of underwater, such as temperature, salinity, pH, shipping activity, and wind. Additionally, up to 7.65% of improvement in terms of PDR has been demonstrated in LQRP in comparison with state-of-the-art Link Quality Estimation based Routing protocol (LQER) through MATLAB simulations.

- Location-Free Void Avoidance Routing (LFVAR) protocol has been designed using status, hop count, and depth as attributes. Further, through UnetStack simulations, 23.87%, 32.32%, 9.87%, 11.22%, and 20.54% improvements in terms of average hop count, end-to-end delay, Packet Delivery Ratio (PDR), throughput, and energy efficiency, respectively are observed in LFVAR in comparison with Intar.

- Link and Void Aware Routing (LVAR) protocol has been designed using status, hop count, and PDP as attributes. Further, through UnetStack simulations, 17.64%, 35.64%, and 9.45% improvements in terms of hop count, end-to-end delay, and throughput have been observed in LVAR when compared with Intar.

- Cluster-based Multi-Attribute Routing (CMAR) general-purpose protocol has been designed using status, progress, and PDP as attributes. Further, by determining PRNs and their score using TOPSIS, priorities have been identified. Additionally, using threshold-based clustering and hold time computations, data is forwarded in CMAR. Moreover, through MATLAB simulations, 75% and more transmission reliability has been demonstrated in CMAR in comparison with state-of-the-art HydroCast.

9.2 FUTURE SCOPE

Progress-based void status detection and beacon-based void status propagation have been proposed in all the contributions of this research work. Moreover, void status has been used as a primary attribute in all the proposed algorithms to eliminate void nodes from the routing. Further, a unique subset of attributes (distance, hop count, depth, progress, PDP, and physical characteristics of underwater) have been considered to decide the suitable next hop. Further, directions to improve the scope of the work in this thesis are listed as follows:

- Based on the subset of attributes, a suitable underwater application can be decided for each of the proposed algorithms.
- Different applications have varied requirements like delay, reliability, and energy consumption. Weights of attributes can be fine-tuned based on the application's requirements to satisfy those criteria.
- The LFVAR and LVAR routing protocols are implemented in UnetStack. Further, as an extension of this work, those protocols can be ported to a test-bed environment, and real-time performance can be measured and analyzed.
- CMAR provides a general framework for a routing protocol, which uses the MADM technique. However, CMAR proposed in the thesis uses advancement and PDP attributes to achieve compatible evaluation with the HydroCast. However, considering in which application CMAR is deployed can vary the attributes.

BIBLIOGRAPHY

Akyildiz, I. F., Pompili, D., and Melodia, T. (2004). Challenges for efficient communication in underwater acoustic sensor networks. *SIGBED Rev.*, 1(2):3–8.

Akyildiz, I. F., Pompili, D., and Melodia, T. (2005). Underwater acoustic sensor networks: research challenges. *Ad hoc networks*, 3(3):257–279.

Akyildiz, I. F., Pompili, D., and Melodia, T. (2006). State-of-the-art in protocol research for underwater acoustic sensor networks. In *Proceedings of the 1st ACM international workshop on Underwater networks*, pages 7–16.

Alasarpanahi, H., Ayatollahitafti, V., and Gandomi, A. (2020). Energy-efficient void avoidance geographic routing protocol for underwater sensor networks. *International Journal of Communication Systems*, 33(6):e4218.

Barbeau, M., Blouin, S., Cervera, G., Garcia-Alfaro, J., and Kranakis, E. (2015). Location-free link state routing for underwater acoustic sensor networks. In *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1544–1549.

Basagni, S., Petrioli, C., Petroccia, R., and Spaccini, D. (2015). CARP: a channel-aware routing protocol for underwater acoustic wireless networks. *Ad Hoc Networks*, 34:92–104.

Behzadian, M., Otaghsara, S. K., Yazdani, M., and Ignatius, J. (2012). A state-of the-art survey of topsis applications. *Expert Systems with applications*, 39(17):13051–13069.

Boukerche, A. and Darehshoorzadeh, A. (2014). Opportunistic routing in wireless networks: Models, algorithms, and classifications. *ACM Computing Surveys (CSUR)*, 47(2):1–36.

Burrowes, G. E. and Khan, J. Y. (2010). Investigation of a short-range underwater acoustic communication channel for mac protocol design. In *2010 4th International Conference on Signal Processing and Communication Systems*, pages 1–8. IEEE.

Cayirci, E., Tezcan, H., Dogan, Y., and Coskun, V. (2006). Wireless sensor networks for underwater survelliance systems. *Ad hoc networks*, 4(4):431–446.

Chakchouk, N. (2015). A survey on opportunistic routing in wireless communication networks. *IEEE Communications Surveys & Tutorials*, 17(4):2214–2241.

Chandavarkar, B. R. and Guddeti, R. M. R. (2015). Simplified and improved analytical hierarchy process aid for selecting candidate network in an overlay heterogeneous networks. *Wireless Personal Communications*, 83(4):2593–2606.

Chandavarkar, B. R. and Guddeti, R. M. R. (2016). Simplified and improved multiple attributes alternate ranking method for vertical handover decision in heterogeneous wireless networks. *Computer Communications*, 83:81–97.

Chitre, M. (2020 (accessed June 24, 2022)a). fjage documentation. `https://buildmedia.readthedocs.org/media/pdf/fjage/dev/fjage.pdf`.

Chitre, M. (2022 (accessed July 20, 2022)b). Underwater networks handbook. `https://unetstack.net/handbook/unet-handbook.pdf`.

Chitre, M., Bhatnagar, R., Ignatius, M., and Suman, S. (2014a). Baseband signal processing with unetstack. In *2014 Underwater Communications and Networking (UComms)*, pages 1–4. IEEE.

Chitre, M., Bhatnagar, R., and Soh, W.-S. (2014b). Unetstack: An agent-based software stack and simulator for underwater networks. In *2014 Oceans-St. John's*, pages 1–10. IEEE.

Coutinho, R. W., Boukerche, A., Vieira, L. F., and Loureiro, A. A. (2015a). Geographic and opportunistic routing for underwater sensor networks. *IEEE Transactions on Computers*, 65(2):548–561.

Coutinho, R. W., Boukerche, A., Vieira, L. F., and Loureiro, A. A. (2015b). Geographic and opportunistic routing for underwater sensor networks. *IEEE Transactions on Computers*, 65(2):548–561.

Coutinho, R. W., Boukerche, A., Vieira, L. F., and Loureiro, A. A. (2018). Underwater wireless sensor networks: A new challenge for topology control–based systems. *ACM Computing Surveys (CSUR)*, 51(1):1–36.

Coutinho, R. W., Vieira, L. F., and Loureiro, A. A. (2013). Movement assisted-topology control and geographic routing protocol for underwater sensor networks. In *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*, pages 189–196.

Domingo, M. C. (2008). Overview of channel models for underwater wireless communication networks. *Physical Communication*, 1(3):163–182.

Erol-Kantarci, M., Mouftah, H. T., and Oktug, S. (2010). Localization techniques for underwater acoustic sensor networks. *IEEE Communications Magazine*, 48(12):152–158.

Erol-Kantarci, M., Mouftah, H. T., and Oktug, S. (2011a). A survey of architectures and localization techniques for underwater acoustic sensor networks. *IEEE Communications Surveys & Tutorials*, 13(3):487–502.

Erol-Kantarci, M., Mouftah, H. T., and Oktug, S. (2011b). A survey of architectures and localization techniques for underwater acoustic sensor networks. *IEEE Communications Surveys Tutorials*, 13(3):487–502.

Felemban, E., Shaikh, F. K., Qureshi, U. M., Sheikh, A. A., and Qaisar, S. B. (2015). Underwater sensor network applications: A comprehensive survey. *International Journal of Distributed Sensor Networks*, 11(11):896832.

Figueira, J. R., Greco, S., Roy, B., and Słowiński, R. (2013). An overview of electre methods and their recent extensions. *Journal of Multi-Criteria Decision Analysis*, 20(1-2):61–85.

Ghoreyshi, S. M., Shahrabi, A., and Boutaleb, T. (2015). An inherently void avoidance routing protocol for underwater sensor networks. In *2015 International Symposium on wireless communication systems (ISWCS)*, pages 361–365. IEEE.

Ghoreyshi, S. M., Shahrabi, A., and Boutaleb, T. (2016). A novel cooperative opportunistic routing scheme for underwater sensor networks. *Sensors*, 16(3):297.

Ghoreyshi, S. M., Shahrabi, A., and Boutaleb, T. (2017). Void-handling techniques for routing protocols in underwater sensor networks: Survey and challenges. *IEEE Communications Surveys & Tutorials*, 19(2):800–827.

Guan, Q., Ji, F., Liu, Y., Yu, H., and Chen, W. (2019). Distance-vector-based opportunistic routing for underwater acoustic sensor networks. *IEEE Internet of Things Journal*, 6(2):3831–3839.

Han, G., Jiang, J., Bao, N., Wan, L., and Guizani, M. (2015). Routing protocols for underwater wireless sensor networks. *IEEE Communications Magazine*, 53(11):72–78.

Hwang, C.-L. and Yoon, K. (1981). Methods for multiple attribute decision making. In *Multiple attribute decision making*, pages 58–191. Springer.

Ismail, A., Wang, X., Hawbani, A., Alsamhi, S., and Abdel Aziz, S. (2022a). Routing protocols classification for underwater wireless sensor networks based on localization and mobility. *Wireless Networks*, pages 1–30.

Ismail, A., Wang, X., Hawbani, A., Alsamhi, S., and Abdel Aziz, S. (2022b). Routing protocols classification for underwater wireless sensor networks based on localization and mobility. *Wireless Networks*, pages 1–30.

Javaid, N., Majid, A., Sher, A., Khan, W. Z., and Aalsalem, M. Y. (2018). Avoiding void holes and collisions with reliable and interference-aware routing in underwater wsns. *Sensors*, 18(9):3038.

Jiang, P., Huang, Q., Wang, J., Dai, X., and Lin, R. (2006). Research on wireless sensor networks routing protocol for wetland water environment monitoring. In *First International Conference on Innovative Computing, Information and Control-Volume I (ICICIC'06)*, volume 3, pages 251–254. IEEE.

Jiang, S. (2018). On reliable data transfer in underwater acoustic networks: A survey from networking perspective. *IEEE Communications Surveys Tutorials*, 20(2):1036–1055.

Karim, S., Shaikh, F. K., Chowdhry, B. S., Mehmood, Z., Tariq, U., Naqvi, R. A., and Ahmed, A. (2021). Gcorp: Geographic and cooperative opportunistic routing protocol for underwater sensor networks. *IEEE Access*, 9:27650–27667.

Kheirabadi, M. T. and Mohamad, M. M. (2013). Greedy routing in underwater acoustic sensor networks: a survey. *International Journal of Distributed Sensor Networks*, 9(7):701834.

Kumar, P., Kumar, P., Priyadarshini, P., et al. (2012). Underwater acoustic sensor network for early warning generation. In *2012 Oceans*, pages 1–6. IEEE.

Lee, S., Bhattacharjee, B., and Banerjee, S. (2005a). Efficient geographic routing in multihop wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 230–241.

Lee, S., Bhattacharjee, B., and Banerjee, S. (2005b). Efficient geographic routing in multihop wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 230–241.

Luo, H., Wu, K., Ruby, R., Hong, F., Guo, Z., and Ni, L. M. (2017). Simulation and experimentation platforms for underwater acoustic sensor networks: Advancements and challenges. *ACM Computing Surveys (CSUR)*, 50(2):28.

Luo, J., Chen, Y., Wu, M., and Yang, Y. (2021). A survey of routing protocols for underwater wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 23(1):137–160.

Mohamed, N., Jawhar, I., Al-Jaroodi, J., and Zhang, L. (2010). Monitoring underwater pipelines using sensor networks. In *2010 IEEE 12th International Conference on High Performance Computing and Communications (HPCC)*, pages 346–353. IEEE.

Nazareth, P. and Chandavarkar, B. (2020). Link quality-based routing protocol for underwater acoustic sensor networks. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6. IEEE.

Nazareth, P. and Chandavarkar, B. (2021a). Link and void aware routing protocol for underwater acoustic sensor networks. In *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–6. IEEE.

Nazareth, P. and Chandavarkar, B. (2021b). Location-free void avoidance routing protocol for underwater acoustic sensor networks. *Wireless Personal Communications*, pages 1–26.

Nazareth, P. and Chandavarkar, B. R. (2019). E-VAR: enhanced void avoidance routing algorithm for underwater acoustic sensor networks. *IET Wireless Sensor Systems*, 9(6):389–398.

Noh, Y., Lee, U., Lee, S., Wang, P., Vieira, L. F., Cui, J.-H., Gerla, M., and Kim, K. (2015). Hydrocast: Pressure routing for underwater sensor networks. *IEEE Transactions on Vehicular Technology*, 65(1):333–347.

Noh, Y., Lee, U., Wang, P., Choi, B. S. C., and Gerla, M. (2012). VAPR: Void-aware pressure routing for underwater sensor networks. *IEEE Transactions on Mobile Computing*, 12(5):895–908.

Panda, M. and Jagadev, A. K. (2018). TOPSIS in multi-criteria decision making: a survey. In *2018 2nd International Conference on Data Science and Business Analytics (ICDSBA)*, pages 51–54. IEEE.

Pompili, D. and Akyildiz, I. F. (2009). Overview of networking protocols for underwater wireless communications. *IEEE Communications Magazine*, 47(1):97–102.

Pompili, D. and Melodia, T. (2005). Three-dimensional routing in underwater acoustic sensor networks. In *Proceedings of the 2nd ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 214–221.

Rao, R. V. (2007). Introduction to multiple attribute decision-making (madm) methods. *Decision Making in the Manufacturing Environment: Using Graph Theory and Fuzzy Multiple Attribute Decision Making Methods*, pages 27–41.

Shin, D., Hwang, D., and Kim, D. (2012). DFR: an efficient directional flooding-based routing protocol in underwater sensor networks. *Wireless Communications and Mobile Computing*, 12(17):1517–1527.

Su, X., Ullah, I., Liu, X., and Choi, D. (2020). A review of underwater localization techniques, algorithms, and challenges. *Journal of Sensors*, 2020.

Tan, W. L., Hu, P., and Portmann, M. (2012). SNR-Based Link Quality Estimation. In *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, pages 1–5.

Tzu-Chiang, C., Jia-Lin, C., Yue-Fu, T., and Sha-Pai, L. (2013). Greedy geographical void routing for wireless sensor networks. *International Journal of Computer and Information Engineering*, 7(6):769–777.

Urick, R. J. (1983). Principles of underwater sound 3rd edition. *Peninsula Publising Los Atlos, California*, 22:23–24.

Vafaei, N., Ribeiro, R. A., and Camarinha-Matos, L. M. (2018). Data normalisation techniques in decision making: case study with topsis method. *International journal of information and decision sciences*, 10(1):19–38.

Vinogradova, I. (2019). Multi-attribute decision-making methods as a part of mathematical optimization. *Mathematics*, 7(10):915.

Virmani, D. and Jain, N. (2016). Intelligent information retrieval for tsunami detection using wireless sensor nodes. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1103–1109. IEEE.

Wang, L. and Kuo, G.-S. G. (2012). Mathematical modeling for network selection in heterogeneous wireless networks—a tutorial. *IEEE Communications Surveys & Tutorials*, 15(1):271–292.

Wati, S., Rakesh, N., and Astya, P. N. (2019). Data communication issues in underwater sensor network. In *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pages 150–155. IEEE.

Xie, P., Cui, J.-H., and Lao, L. (2006). VBF: Vector-based forwarding protocol for underwater sensor networks. In *International conference on research in networking*, pages 1216–1221. Springer.

Xie, P., Zhou, Z., Peng, Z., Cui, J.-H., and Shi, Z. (2009). Void avoidance in three-dimensional mobile underwater sensor networks. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 305–314. Springer.

Yang, X., Ong, K. G., Dreschel, W. R., Zeng, K., Mungle, C. S., and Grimes, C. A. (2002). Design of a wireless sensor network for long-term, in-situ monitoring of an aqueous environment. *Sensors*, 2(11):455–472.

Yu, H., Yao, N., and Liu, J. (2015). An adaptive routing protocol in underwater sparse acoustic sensor networks. *Ad Hoc Networks*, 34:121–143.

Zeng, K., Lou, W., Yang, J., and Brown, D. R. I. (2007). On geographic collaborative forwarding in wireless ad hoc and sensor networks. In *International Conference on Wireless Algorithms, Systems and Applications (WASA 2007)*, pages 11–18. IEEE.

Zhang, B., Sukhatme, G. S., and Requicha, A. A. (2004). Adaptive sampling for marine microorganism monitoring. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 2, pages 1115–1122. IEEE.

Zheng, L., Wu, B.-q., Zhong, J., and Sun, X.-l. (2019). Research on underwater communication based on acoustic wave. In *2019 12th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pages 343–345.

PUBLICATIONS

Journals

1. **Pradeep Nazareth** and B. R. Chandavarkar (2019). E-VAR: enhanced void avoidance routing algorithm for underwater acoustic sensor networks. Wireless Sensor Systems, IET, Volume 9, pp. 389-398. (DOI: `https://doi.org/10.1049/iet-wss.2019.0032`).

2. **Pradeep Nazareth** and B. R. Chandavarkar (2022). Location-Free Void Avoidance Routing Protocol for Underwater Acoustic Sensor Networks. Wireless Personal Communications, Springer, Volume 123, pp. 575-600. (DOI: `https://doi.org/10.1007/s11277-021-09147-y`).

3. **Pradeep Nazareth** and B. R. Chandavarkar. Hop-based Void Avoidance Routing Protocol for Underwater Acoustic Sensor Networks. International Journal of Ad Hoc and Ubiquitous Computing, Inderscience [**Under Review**].

4. **Pradeep Nazareth** and B. R. Chandavarkar. Cluster-based Multi-Attribute Routing Protocol for Underwater Acoustic Sensor Networks. Wireless Personal Communications, Springer, [**Under Review**].

Conferences

1. **Pradeep Nazareth** and B. R. Chandavarkar (2019). Void avoidance node deployment strategy for underwater sensor networks. In 2^{nd} International Conference on Smart IoT Systems - Innovations in Computing-2019 (SSIC-2019) held at Manipal University, Jaipur, pp. 493-502. Springer, Singapore. (DOI: `https://doi.org/10.1007/978-981-13-8406-6_47`).

2. **Pradeep Nazareth** and B. R. Chandavarkar (2020). Void-aware routing protocols for underwater communication networks: A survey. In 1^{st} International Conference on Evolutionary Computing And Mobile Sustainable Networks-2020 (ICECMSN-2020) held at Sir MVIT, Bangalore, pp. 747-760. Springer, Singapore. (DOI: https://doi.org/10.1007/978-981-15-5258-8_69).

3. **Pradeep Nazareth** and B. R. Chandavarkar (2020). Link Quality-based Routing Protocol for Underwater Acoustic Sensor Networks. In 11^{th} International Conference on Computing, Communication and Networking Technologies-2020 (ICCCNT-2020) held at IIT Kharagpur, pp. 1-6. IEEE. (DOI: https://doi.org/10.1109/icccnt49239.2020.9225289).

4. **Pradeep Nazareth** and B. R. Chandavarkar (2021). Link and Void Aware Routing Protocol for Underwater Acoustic Sensor Networks. In 12^{th} International Conference on Computing, Communication and Networking Technologies-2021 (ICCCNT-2021) held at IIT Kharagpur, pp. 1-6. IEEE. (DOI: https://doi.org/10.1109/icccnt51525.2021.9580080).

5. **Pradeep Nazareth**, B. R. Chandavarkar, and Priyansh Dubey (2022). Trilateration based Localization for Underwater Sensor Networks. In 4^{th} International Conference on VLSI, Communication and Signal Processing-2021 (VCAS-2021) held at MNNIT Allahabad, pp. 657-669. Springer, Singapore. (DOI: https://doi.org/10.1007/978-981-19-2631-0_57).

6. Ketan Bhujange , Afrah Nayeem, Anusha P. Das, B. R. Chandavarkar, and **Pradeep Nazareth**. Opportunistic Underwater Routing Protocols: A Survey. In 2^{nd} International Conference on computational Electronics for wireless communications-2022 (ICCWC-2022) held at NITK, Surathkal, pp. 593-604 . Springer, Singapore. (https://link.springer.com/chapter/10.1007/978-981-19-6661-3_54

7. Afrah nayeem, B. R. Chandavarkar, and **Pradeep Nazareth**. Implementation of Next Hop Selection in Underwater Routing using TOPSIS. In International

Conference on Intelligent Computing Systems and Applications-2022 (ICICSA-2022) held at NIT, Silchar **(accepted and presented)**.

8. **Pradeep Nazareth**, B. R. Chandavarkar, and Anusha P. Das. Cluster Formation for Underwater Routing in UnetStack3. In International Conference on Intelligent Computing Systems and Applications-2022 (ICICSA-2022) held at NIT, Silchar **(accepted and presented)**.

9. Sahil Kumar, B. R. Chandavarkar, and **Pradeep Nazareth**. Hop-count Based Routing Protocols for Underwater Acoustic Sensor Networks: A Survey. In 13th International Conference on Computing Communication and Networking Technologies-2022 (ICCCNT- 2022), sponsored by IIT Patna, pp. 1-6.(`https://ieeexplore.ieee.org/abstract/document/9984590`).

BIO-DATA

Permanent address:

Pradeep Nazareth, H. No. BM. 6/7-2,
Madku Road, Nekkare,
Belman Post and Village
Karkala TQ, Udupi District -576 111
Karnataka
Mobile No.: 9164525591/9480353356

Qualifications:

M.Tech. Software Engineering, Sri Jayachamarajendra College of Engineering, Mysore, Visvesvaraya Technologial University, Belagavi, 2009.

B.E. Computer Science and Engineering, KVG College of Engineering, Sullia, Visvesvaraya Technologial University, Belagavi, 2005.