IMPLEMENTATION OF RTSP,FTP,TCP/IP PROTOCOL

Report submitted to the SASTRA Deemed to be

University as the requirement for the course

CSE302: COMPUTER NETWORKS

Submitted by

PRAVEENKUMAR B

(Reg. No.: 224003081, 3rd - Year(CSE))

DECEMBER 2022

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 OF THE UGC ACT, 1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA – 613 401

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA – 613 401

Bonafide Certificate

This is to certify that the report titled **"IMPLEMENTATION OF RTSP,FTP,TCP/IP PROTOCOL "** submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech. is a bonafide record of the work done by

Shri. PRAVEENKUMAR B, Reg. No.224003081, COMPUTER SCIENCE ENGINEERING during the academic year 2020-21, in the School of Computing.

Project Based Work Viva voce held on _____

Examiner 1 **Examiner 2**

TABLE OF CONTENTS

BONAFIDE CERTFICATE ii

LIST OF FIGURES iv

LIST OF TABLES vi

NOTATIONS vii

ABBREVIATIONS viii

ABSTRACT ix

PROBLEM STATEMENT x

CHAPTER 1 1

INTRODUCTION 1

1.1 RELATED WORKS 4

1.2 PROPOSED WORK 7

1.3 MERITS & DEMERITS 15

CHAPTER 2 16

SOURCE CODE 16

CHAPTER 3 26

SNAPSHOTS 26

CHAPTER 4 34

PERFORMANCE EVALUATION 34

CHAPTER 5 40

CONCLUSION AND FUTURE ENHANCEMENTS 40

CHAPTER 6 41

REFERENCES 41

List of Figures

Figure No.	Figure Name	Page No.
1.1.	Network Architecture	2
1.2.	Applications of UWSNs	3
1.2.1.	Network Architecture (UWSNs)	8
1.2.2.	Network with two embedded sinks.	9
3.1.	100 Nodes and 6 sink nodes plotted	26
3.2.	Delivery status and route of packet transmission of 1st node to 5th node	27
3.3.	Delivery status and route of packet transmission of 6th node to 10th node	27
3.4.	Delivery status and route of packet transmission of 11th node to 15th node	28
3.5.	Delivery status and route of packet transmission of 16th node to 20th node	28
3.6.	Delivery status and route of packet transmission of 21st node to 25th node	29
3.7.	Delivery status and route of packet transmission of 26th node to 30th node	29
3.8.	Delivery status and route of packet transmission of 50th node to 54th node	30

Figure No.	Figure Name	Page No.
3.9.	Delivery status and route of packet transmission of 61st node to 65h node	30
3.10.	Delivery status and route of packet transmission of 71st node to 75th node	31
3.11.	Delivery status and route of packet transmission of 81st node to 85th node	31
3.12.	Delivery status and route of packet transmission of 86th node to 90th node	32
3.13.	Delivery status and route of packet transmission of 91st node to 95th node	32
3.14.	Delivery status and route of packet transmission of 97th node to 100th node and average E2ED, throughput and PDR is also displayed	33
4.1.	Graph between PDR and transmission	37
4.2.	Graph between throughput range and transmission range	37
4.3.	Graph between E2ED and transmission range	37
4.4.	Graph between PDR and number of nodes	39
4.5.	Graph between throughput and number of nodes	39
4.6.	Graph between E2ED and number of nodes	39

List of Tables

Table No.	Table Name	Page No.
4.1.	Parameters and their values used in the simulation	34
4.2.	PDR, Throughput and E2ED for different node transmission range	35
4.3.	PDR, Throughput and E2ED for different node transmission range	36
4.4.	PDR, Throughput and E2ED for different number of nodes	38

NOTATIONS

Notation	Description
E_d	Euclidean distance
D_{lk}	Distance between node l and node
$P_{forwarders}$	PFNs which are considered ad forwarders

ABBREVIATIONS

RTSP	Real Time Streaming Protocol
TCP	Transmission Control Protocol
IP	Internet Protocol
FTP	File Transmission Protocol

ABSTRACT

The primary goal of this project is to integrate RTSP, FTP, and TCP/IP in the Group Conversation Application. In this project, the video and audio in the communication are streamed using the RTSP protocol, and the chat is encrypted using end to end encryption (AES Algorithm). Any sort of file up to 2GB can be sent in a chat using TCP/IP (i.e similar to what's app). It is comparable to a group in which there is just one server and numerous clients. The chat's password is set up for login.These are features of this project.

KEYWORDS: RTSP,FTP,TCP/IP,Encryption,Decryption

CHAPTER – 1

INTRODUCTION

A Group Chat Application which has features like Encrypted Messaging, Video Streaming and File Transfer.

Features:

AES 256 based message Encryption.

Live (webcam) Video/Audio Transmistion .

Supports File Transfer Upto 2 Gb.

All file format supported for File Transfer.

Uses TCP/IP Protocol for Message Transfer

Real Time Streaming Protocol(RTSP):

Real Time Streaming Protocol (RTSP) is an application-level network communication system that transfers real-time data from multimedia to an endpoint device by communicating directly with the server streaming the data.

In the transport layer, RTP (Real-Time Protocol) is used to transmit the stream in real-time. The RTSP function is equivalent to the remote control of a streaming media server. IP cameras can use both TCP and UDP to transmit streaming content. However, it should be noted that UDP does not make any practical sense for this task.

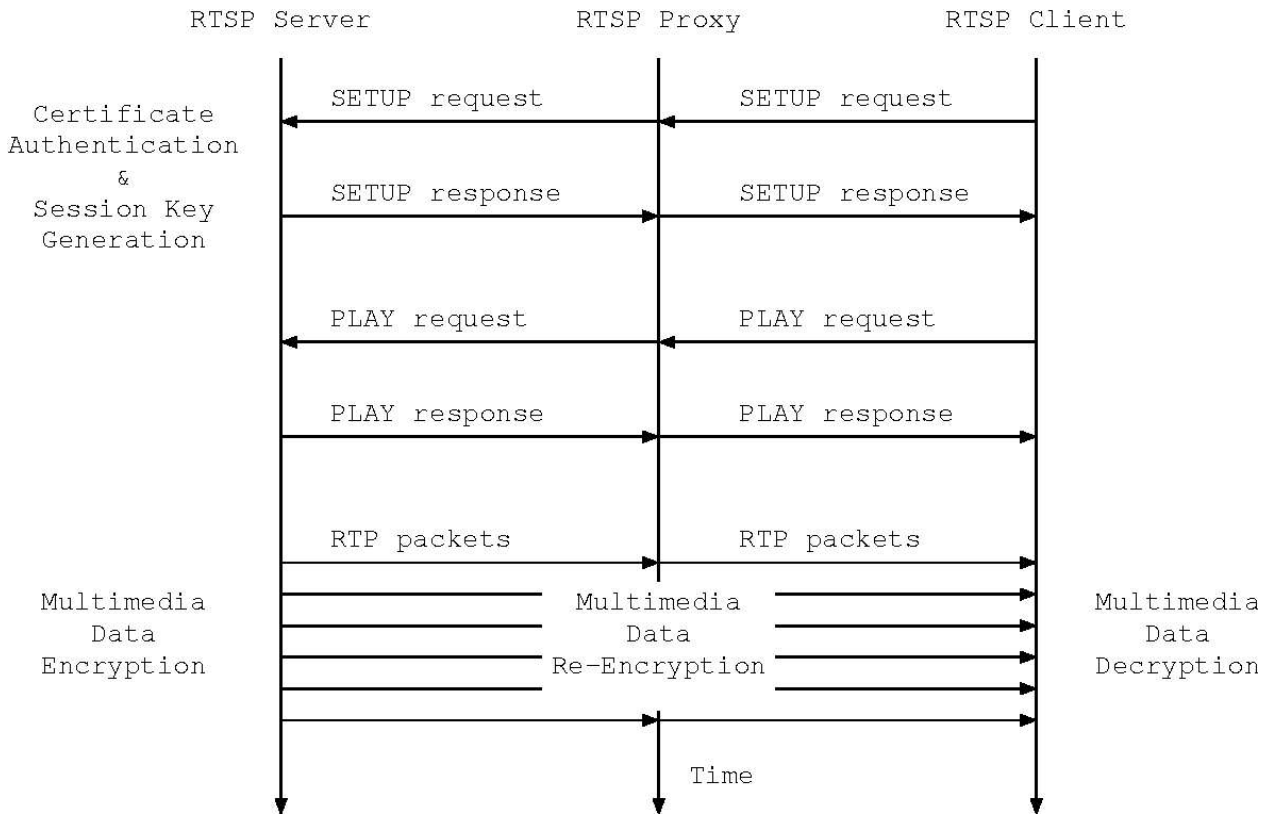


Figure 1.1

RTSP implementation

These routing protocols are however highly susceptible to a problem with void hole by excluding the second hop PFNs of the forwarder, which allow the routing mechanism to choose nodes with no additional PFNs, leading to a void hole and ultimately a decreased packet delivery rate.

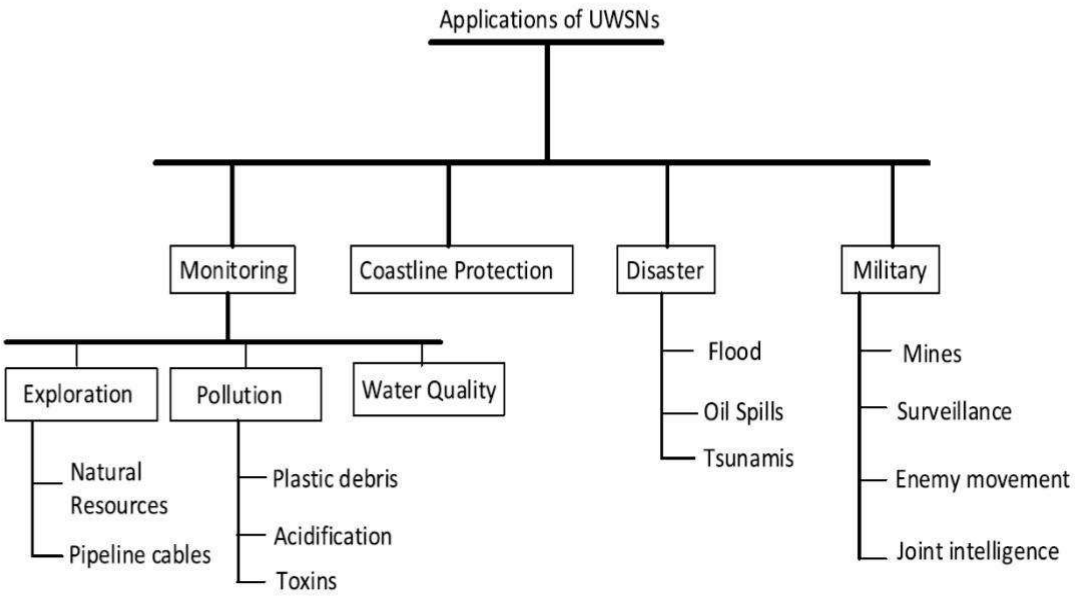


Fig: 1.2 :- Applications of UWSNs

CHAPTER – 1.1

RELATED WORK

UWSN routing protocols are split into geographical details, and geographical routing is free of charge. Many routing protocols that use sender and receiver node geographical details for the forwarding of data packets were introduced in current literature.

An underwater wireless sensor network (UWSN) is created by a community of sensor - connected acoustic nodes. In addition, sensor controls are a main component of the UWSN and is randomly used across the specified volume of the network for tracking, sensing, collection and transmission, because of the collaborative actions of sensing devices in the network: distant locations, physical climate, temperature, humidity, fighting areas, the oceans, volcanos and many more.

UWSN has small battery sensor nodes, which are critical in the creation of a routing strategy. In order to cut implementation and maintenance costs and prolong the network running period, the sustainable deployment of the sensor system also is needed.

UWSN routing protocols are divided into a geographically aware routing scheme. Many routing protocols that use sender and receiver node geographical details for the forwarding of data packets were introduced in current literature.

Saeed et. al (Saeed, 2020) proposed a Stable Energy Efficient and Cooperative Routing Protocol for UWSNs, which included a solid, effective, defensive mechanism to combat submarine attacks. It uses cooperative routing to improve network performance. The minimum formula for enforcing protection is used to keep it acceptable for submarine climate, considering the resource-restricted environment of UWSN.

The findings show that the number of living nodes, the transmission losses decreased by over 50%, the production rose by up to 9 per 100, the energy tax reduced by up to 23 % and the delay decreased by 25 percent.

There is a term called void hole generated because of a massive amount of energy consumption, which does not cause a forwarder node or node to die. This has

become a major networking problem, because most energy is used to build such void holes and because of the imbalance of the critical network implementation.

Hop vector routing approach is used under certain situations. The foundation of existing IP networks is Hop by Hop routing. This means that routing decisions are taken independently and locally on each router.

Therefore, even though a router is compromised it would not lead to a network outage. In fact, hop refers to the instrument that normally transfers the packet to a router.

Diverse protocols for avoiding void hole were suggested, of which a single hop selection based forwarding - weighting depth and forwarding area division-depth routing, which selects at least two optimal communication nodes to reduce the likelihood of the occurrence of vacuum hole, as well as preventing backward transmissions, has been developed, one of which was provided by **Ahmad et al, (Ahmad, 2017).**

Mobile sinks are placed on ships, vehicles etc. to reduce the data charges at intermediate nodes and retrieve data from the void areas.

To obtain interest information from the area of interest. The availability of mobile sinks allows the identification of minerals from waters that are not humanly monitorable, including but not limited to seabed surveys. A grid-based protocol, called powerful and efficient grid-based transformation by leveraging minimum hop count, is developed and evaluated by **Al-Salti et. al (Al-Salti, 2019)** and others (ERGR-EMHC).

This protocol partitions the network into a three-dimensional cell grid that is identified by the surrounding cells on the basis of their minimum hop count. Packets are distributed cell by cell through certain selected nodes known as cell heads.

Coutinho et. al (Coutinho, 2018) proposed to define the topology control algorithms using the main approach of modifying the topology of the network

Faheem et. al (Faheem 2017) suggested for UWSN the latest Quality of service-conscious evolutionary cluster based routing protocol (QERP).

The protocol suggested boosts PDR and decreases average electricity usage and total network energy.

A proposed energy-efficient adaptive cluster routing algorithm (ACUN), which utilizes a hierarchical multilevel network structure based on the distance between cluster heads, sinking kernel, and the cluster head's residual energy to assess competition radius scale, is proposed by **Wan et al. (Wan, 2019).**

Bu et. al (Bu, 2018) suggested a fluctuating logic vector-based routing protocol in which location information and sensor node energy information were all included in the desirability calculations. Moreover, parameters such as valid distance, projection, and battery node level are taken into consideration in the fluctuating logic system.

Energy Scaled and expanded vector-specific forwarding scheme **Wadud et. al**

(Wadud, 2017) proposes to increase holding hours by utilizing the residual energy of the node to the scale and the vector pipeline distance ratio.

There is a major change in the resulting reduction and extension of held time in all transmitting nodes in order to eliminate multiple transmission, which decreases electricity usage and energy balance in the network. It reduces holding time and rapidly transfers data packets upstream if a node is held at least between its neighbors.

Yu et. al (Yu, 2016) suggested an even more effective Depth Based Routing Protocol, called Weighting Depth and Forwarding Area Division-Depth Based Routing, weighting depth and forwarding area division.

The next forwarding nodes are chosen in this protocol according to the weighting total of the difference in depth of two hops, which takes into account not only the current de pth, but also the predicted next hop depth.

CHAPTER – 1.2

PROPOSED WORK

A packet transmission forms a source node, simply transmits the packet and then the packet is received by all sensor nodes in the transmission range. If the transmitting region involves a sink node, the packet is supplied effectively, and is passed directly to the base station.

In the absence of a sink node, the packet is passed to the next forwarder of the next node, and is continued until the packet reaches a sink node. A packet is missed if the transmitting range of the source node does not cover adjacent nodes.

PFNs are considered to pick the next forwarder node. If the sensor node is sending a packet, PFNs must forward it, help to improve reliability and reduce the possibility of a void hole incident, when the sensor node has no PFN, the packet is lost .

Nodes near the sink are like bridges between the sinks and the rest of the network; they are suffering from the high volume of traffic and relay the entire network packets. The nodes first hit the closest sink by the

Euclidean (Ed) formulation.

$$E_d = \sqrt{(x_i\text{-}x_j)^2 + (y_i\text{-}y_j)^2 + (z_i\text{-}z_j)^2}$$

The packet will be successfully transmitted if the gap is beyond the transmitting range. If no sink node is located in the set, distances are measured and the shortest of the nodes is the forwarding system chosen. This process is continued before distribution or shipment failure is complete. If no PFNs are in the set, a packet would be lost.

NETWORK ARCHITECTURE:

The network structure (from the fig) involves sink nodes, relay nodes and fixed nodes.

As the names indicate, sink nodes on the surface of the water are often located on the water bottoms, which gather information from the water beneath the water where they are mobile and travel alongside water currents as the relay nodes randomly placed at various water points and are used for transmission and sensing.

Sink nodes are connected with radio connections where acoustic connections are used to communicate with each other and with the sink nodes as relay nodes and anchored nodes. If a packet enters a sink node further sent to the outside base station, it is successfully transmitted.

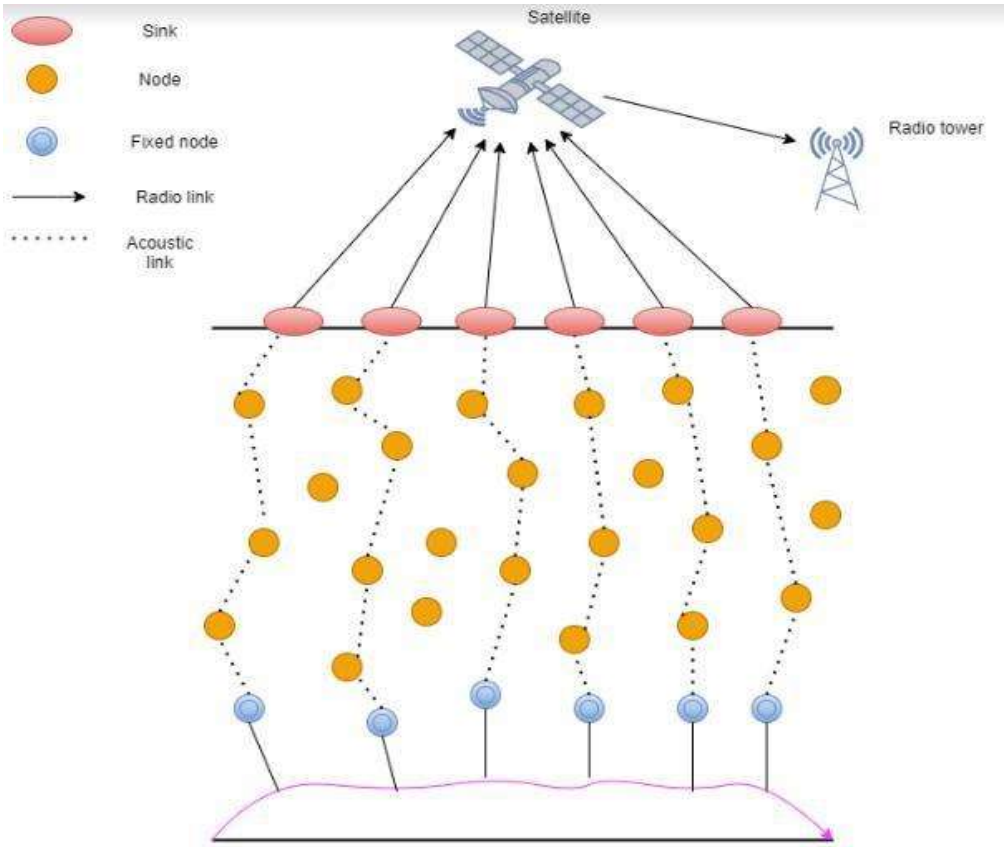


Fig: 1.2.1 :- Network Architecture

SINK DEPLOYMENT:

In network sensor networks, network traffic is a significant limitation, since there are two sinks at these high-density locations in the network region. Packets approaching a sink are called good packets because the packets can safely be sent to the base station.

If the nodes in high traffic areas transfer the received packet to the next sink rather than forward the packet further, high delivery ratio at very low cost can be achieved. Figure 2 displays a 2 embedded sink network.

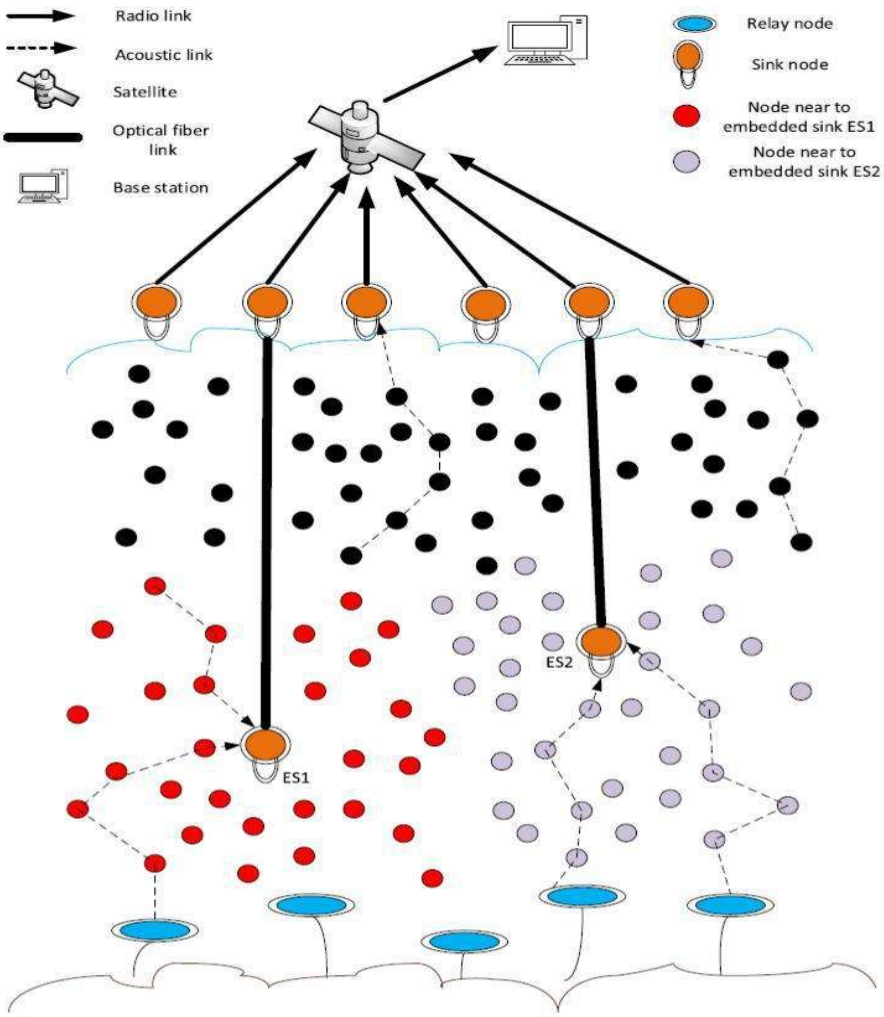


Fig: 1.2.2 :- Network with two embedded sinks.

ALGORITHM FOR THE SELECTION OF APTHOP TO BE TAKEN:

```
for k ← 1 to numNodes by 1 do
    forwarder = N(k).ID
    success = 0
    for l ← 1 to numSinks by 1 do
        calculate distance D_l^k with sink(l)
        if D_l^k <= transmission_range then
            Packet successfully delivered
            success = 1
            break
    shortest_dist = ∞
    for m ← 1 to numNodes by 1 do
        if (m = = k)
            skip the iteration
        calculate distance D_m^k
        if D_m^k <= transmission_range then
            add m to PFNs list

            if node m is not in visited_nodes and D_m^k < shortest_dist
```

```
                shortest_dist = D_m^k

                forwarder = m

                add node m to visited_nodes

P_forwarders = N(forwarder).PFN

while not success do

    if P_forwarders = = 0 then

        packet_drop

        break

    if P_forwarders = = 1 then

        forwarder = PFN_ID

    if P_forwarders > 1 then

        for l ← 1 to numSinks by 1 do

            calculate distance D_l^k with sink(l)

            if D_l^k < transmission_range then

                Packet successfully delivered

                success = 1


                break

        for m ← 1 to numNodes by 1 do
```

calculate distance D_m^k

if D_m^k < transmission_range then

add m to PFNs list

if node m is not in visited_nodes and D_m^k < shortest_dist

shortest_dist = D_m^k

forwarder = m

add node m to visited_nodes

Let numNodes be the overall number and each node tries to send a packet into its nearest sink node, except sinks in UWSN.

In a loop from first node (k = 1) to last node (k = numNodes), let us consider each node transmitting a packet according to the distance between the nodes

The kth node will be chosen step by step before the packet is successfully supplied to the sink node or the packet is lost:

- Initialize the forwarder to k^{th} node, forwarder = N(k).ID
- Initialize the success variable that reveals whether or not the packet was transmitted at the start with 0, that is, the packet has already not been delivered (success 12 P = 0).
- If the success of the vector is equal to 1, the packet was sent successfully to the sink node.

We have to verify if there is a sink node within the transmitting range for the kth node such that the node can transfer the packet directly to the sink node in one hop.

For the l = 1 loop, the distance between the kth node and the lth sink node is determined with Euclidean distance,

$\boldsymbol{D_l^k}$ refers to the distance from *k* node to *l* sink node.

$$E_d = \sqrt{(x_i\text{-}x_j)^2 + (y_i\text{-}y_j)^2 + (z_i\text{-}z_j)^2}$$

If the packet is supplied successfully with a $\boldsymbol{D_l^k}$ < = transmitting range, the success vector is modified to 1 and the loop splits and falls out from the iteration of the kth loop.

If the kth node does not have any sink node in its transmission range it hops to the nearest node in its transmission range, initially give shortest_dist as infinity, shortest_dist =

In a loop for m = 1 to numNodes,

✧ If m = = k, skip the iteration

✧ Calculate the distance between k^{th} node and m^{th} node, $\boldsymbol{D_m^k}$

✧ if D km <= transmission_range,

add node m to PFNs list, they are the neighboring nodes

visited_nodes list includes all the nodes which are visited in the node k transmission route

✓ if node m is not in the list and if $\boldsymbol{D_m^k}$ is less than the

shortest_dist, shortest_dist = $\boldsymbol{D_m^k}$

forwarder = m

node m is added to visited_nodes list

✧ Pforwarders include all the PFNs which are considered as forwarders for the next hop

✧ In a loop while success is not equal to 1,

if there are no $P_{forwarders}$, that is there are no sink nodes or any relay nodes in the transmission range of the kth node, packet is simply dropped, break the loop and come out of the iteration of kth node in the for loop.

if there is only one $P_{forwarders}$, forwarder = PFN_ID

else if Pforwarders > 1,

✧ check if there is any sink node within the transmission range. In a loop, for l = 1 to number of sinks : calculate the distance D_l^k

If D_l^k < = transmission range then, the packet is successfully delivered and success variable is changed to 1, break the loop and come out of the iteration of kth node in the for loop

➢ If the kth node does not have any sink node in its transmission range. In a loop for m =

1 to numNodes,

✧ if m = = k, skip the iteration

✧ calculate the distance between k^{th} node and m^{th} node

✧ if D_m^k <= transmission_range

✧ add node m to PFNs list

if m is not in visited_nodes and if D_m^k < shortest_dist,

shortest_dist = $\boldsymbol{D_m^k}$

forwarder = m , node m is added to visited_nodes list

CHAPTER – 1.3

MERITS & DEMERITS

MERITS:

- Void holes are avoided, PDR and throughput are improved by considering PFNs,
- For Residual Energy Consideration Feature it increase network life time.
- Prioritizing PFNs By holding time Feature, it Reduces Duplicate Packets
- High delivery ratio and low E2ED is achieved by the embedded sinks feature,.
- Control Transmission Power Near to Sink Feature , it decreased E_{tax}.
- Considering greater transmission range and more number of nodes increase the efficiency

DEMERITS:

- For Residual Energy Consideration & Prioritizing PFNs By holding time Feature it increase E2ED.
- High cost on physical connection to surface Sinks achieved by embedded Sinks feature
- Duplicate packets to surface links achieved by Control Transmission power near to sink.

CHAPTER – 2

SOURCE CODE

This code finds the path from the specific node to the sink node for the packet transmission. Each packet transmits all the nodes, it is tested first if there is a sink node inside the transmission range.

When there is one, the distance between the nodes is passed directly or not, search which nodes lie within the range for transmission, locate the node neighbors, find the node closest to them and pick the forwarder accordingly.

This procedure is carried out until the packet is transmitted successfully or the packet is destroyed. Usually, when nodes are in its transfer radius, a packet is lost

CODE:

```
%       close and clear everything running in the command

window clc;


clear  all;

close all;


% Initialize transmission range

transRange = 250;

%       Initialize number of

nodes numNodes = 100;
```

```
%       Initialize minimum range of x,y,z co-ordinates of the network
plot min.x = 0;

min.y = 0;
min.z = 0;

%       Initialize maximum range of x,y,z co-ordinates of the network
plot max.x = 1000;

max.y = 1000;
max.z = -1000;

% Initialize x,y,z co-ordinates for six sinks including two embedded sinks
% embedded sink 1
sink(1,1)=250;
sink(1,2)=250;
sink(1,3)=0;
%       embedded
sink 2
sink(2,1)=250;
sink(2,2)=0;
sink(2,3)=250;
```

```
%       sink 3

sink(3,1)=100;

sink(3,2)=1000;
```

```
sink(3,3)=100;
%       sink 4
sink(4,1)=250;
sink(4,2)=1000;
sink(4,3)=250;

%       sink 5
sink(5,1)=750;
sink(5,2)=100;
sink(5,3)=500;

%       sink 6
sink(6,1)=500;
sink(6,2)=500;
sink(6,3)=500;

%       Plot nodes randomly using createNodes function nodePositions
= createNodes(min, max, numNodes); plot3(nodePositions(:, 1),
nodePositions(:, 2),nodePositions(:, 3), '+'); hold on


% Plot sink nodes
plot3(sink(1, 1), sink(1, 2), sink(1, 3), 'S', 'MarkerFaceColor', 'y');
plot3(sink(2, 1), sink(2, 2), sink(2, 3), 'S', 'MarkerFaceColor', 'y');
```

```
plot3(sink(3, 1), sink(3, 2), sink(3, 3), 'S', 'MarkerFaceColor', 'r');

plot3(sink(4, 1), sink(4, 2), sink(4, 3), 'S', 'MarkerFaceColor', 'r');

plot3(sink(5, 1), sink(5, 2), sink(5, 3), 'S', 'MarkerFaceColor', 'r');

plot3(sink(6, 1), sink(6, 2), sink(6, 3), 'S', 'MarkerFaceColor', 'r');

%       Initialize lost packets and average time taken for one packet delivery as

zero lostPackets = 0;


avgTime = 0;

% Initialize t1 to current starting time

t1 = clock;

% loop for transmitting one packet from each node to a sink node

% for i = 1 to numNodes

for i=1:numNodes

%       Initialize an empty list for visited

nodes visitedNodes = [];


%       Initialize source as ith node and forwarder node as the

source source = i;


forwarder = source;

fprintf('Node %d \n', i);


%       find the route of the packet from the ith node to any of the sink node using the
function
```

```
%       find_route function returns neighbours of the given node, delivery status
(success/failure) and nearest node of the neighbors, this function takes forwarder
node, sink nodes, transmission range, number of nodes, nodes positions and visited
nodes list as parameters

[neighbours, success, nearestNode] = find_route (forwarder, sink, transRange,
numNodes, nodePositions, visitedNodes);

%       Add source to visited nodes

list visitedNodes(end+1) = source;


% if the ith node could not find a sink node in its transmission range

%       while packet status is undelivered or neighbors list is

empty while (success == 0 || isempty(neighbours) == 0)


%       change forwarder to the nearest node obtained

previously forwarder = nearestNode;


%       if forwarder is unreachable, packet is said to be lost and break the

loop if (forwarder == Inf)


success = 0; disp('Packet

Lost') lostPackets =

lostPackets + 1; break;



end

% Add forwarder node to visited nodes list

visitedNodes(end+1) = forwarder;
```

```
% Find the route for the ith packet from the new forwarder to any of the sinks

[neighbours, success, nearestNode] = find_route (forwarder, sink, transRange,
numNodes, nodePositions, visitedNodes);

end

%       display the whole route of the packet transmitted from the ith

node disp('Route in which the packet travelled:')


disp(visitedNodes)

% note the time t2 for ith packet transmission time

t2 = clock;

e = etime(t2,t1);

%       calculate average delay of one packet

transmission avgTime = (avgTime + e)/ i;


end

%       display average end to end delay for a packet transmission in

milliseconds fprintf("Average end to end delay = %f ms \n", avgTime*1000)


% note the ending time after all the packets transmission

t3 = clock;

totalTime = etime(t3,t1);

%       calculate number of packets delivered

packetsDelivered = numNodes - lostPackets;


%                                                   initialize average
                                                    packet size and
```

convert from bytes
to bits 21

```
averagePacketsize = 80*8;

% calculate throughput and display it in kbps

throughput = (packetsDelivered * averagePacketsize)/totalTime;

fprintf("Throughput = %f kbps \n", throughput/1000)

%       calculate packet delivery ratio and

display it pdr = (numNodes -

lostPackets)/numNodes; fprintf('Packet

Delivery Ratio = %f \n', pdr);


% function for randomly generating nodes

function [nodePositions]= createNodes(min, max,numNodes)

for i=1:numNodes

nodePositions(i,1) = (rand) * (max.x);

nodePositions(i,2) = (rand) * (max.y);

nodePositions(i,3) = (rand) * (max.z);

end

end

%       function for finding route of the packet to any of the sinks, if no sink in the
transmission range it returns the nearest node in its range

%       find_route function returns neighbours of the given node, delivery status(
success/failure) and nearest node of the neighbors, this function takes forwardernode,
sink nodes, transmission range, number of nodes, nodes positions and visited nodes
list as parameters
```

```
function[neighbours, success, nearestNode] = find_route(forwarder, sink, transRange,
numNodes, nodePositions, visitedNodes)

%       initialize packet status as undelivered (success = 0) and allocate an empty
neighbors list

success = 0;

neighbours = [];

index = 1;

%       Initialize nearest node and shortestDist to it as

infinity nearestNode = 1/0;


shortestDist = 1/0;

% copy x,y,z co-ordinates of forwarding node

fx = nodePositions(forwarder,1);

fy = nodePositions(forwarder,2);

fz = nodePositions(forwarder,3);

%       Check whether any of the sink is in the transmission

range for i=1: 6


sink_x = sink(i,1);

sink_y = sink(i,2);

sink_z = sink(i,3);


% find the distance between forwarder and nearest sink

dst_sink = sqrt((fx- sink_x)^2 + (fy- sink_y)^2 + (fz- sink_z)^2);
```

```
if (dst_sink < shortestDist)
shortestDist = dst_sink;
end
%       If sink is a neighbour, packet is sent to sink
successfully if( shortestDist <= transRange)

success = 1;
disp('Packet reached at sink node successfully')
return;

en
d
en
d

%       if there is no sink in the range, find the nearest node in the transmission
range for i=1: numNodes

% if forwarder is ith node, skip the iteration and continue
if (forwarder == i)
continue;
end
%       Copy x,y,z co-ordinates of
neighbor x = nodePositions(i,1);
```

```
y = nodePositions(i,2);

z = nodePositions(i,3);
```

```
%       find distance between the forwarder and the ith

node distance = sqrt((fx-x)^2 + (fy-y)^2 + (fz-z)^2);


%       if the distance is less than or equal to the transmission range, add the node to
the neighbors list

if (distance <= transRange)

neighbours(index)=i;

index = index +1;

%       among the neighbors, find the nearest and assign it to

nearestNode neighbours(ismember(neighbours,visitedNodes)) = [];


if (distance < nearestNode & neighbours(ismember(neighbours,i)))

nearestNode = i;


end

end

end

return

end
```

CHAPTER – 3

SNAPSHOTS

PROJECT RESULTS:

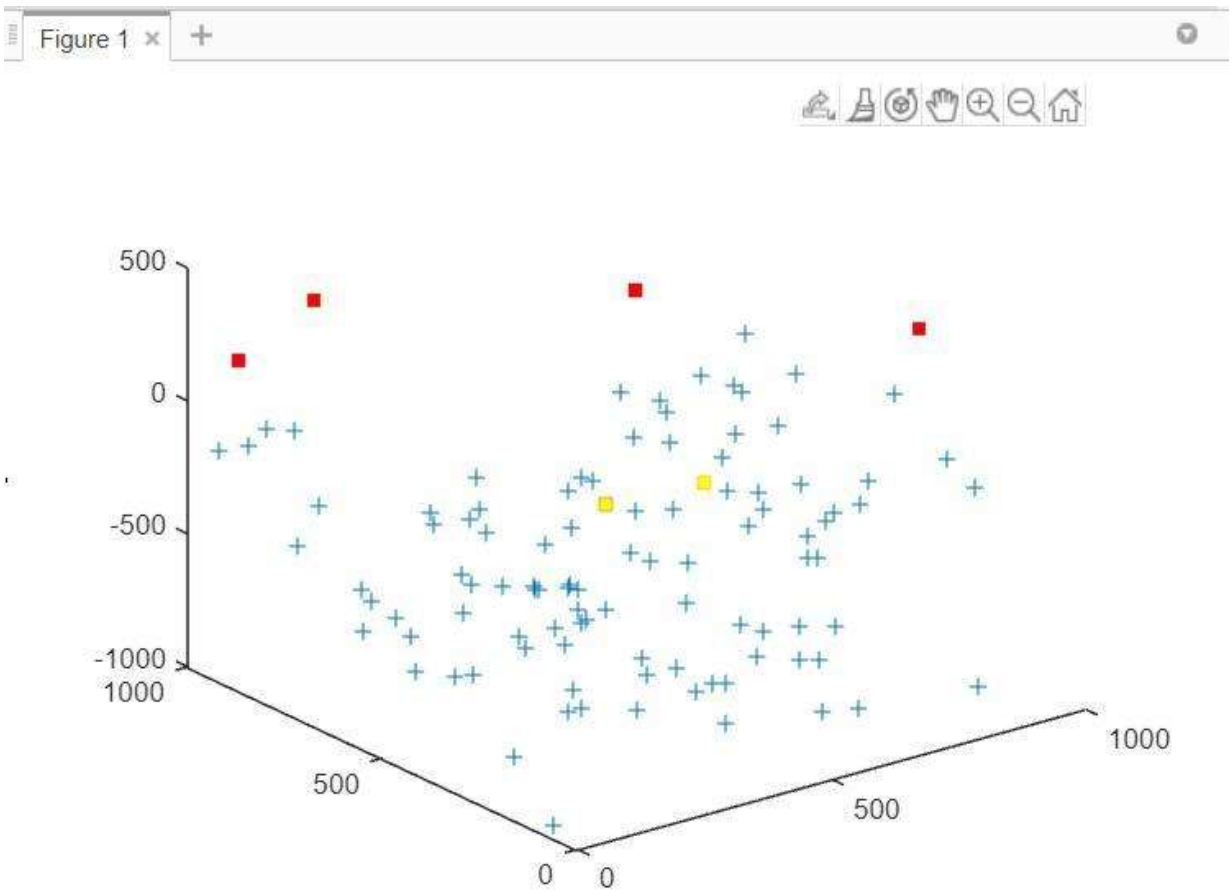


Fig: 3.1 :- 100 Nodes and 6 sink nodes plotted

✧ This figure is a 1000 cubic meter underwater network

✧ If 100 nodes have to be marked as "+," embedded sink nodes are marked as **yellow** square boxes and sink nodes are marked as **red** square boxes on the surface of the water.

```
COMMAND WINDOW
Node 1
Packet Lost
Route in which the packet travelled:
     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 2
Packet Lost
Route in which the packet travelled:
     2     1     5    12    13    27    32    18    65    76    26    15    45    25

Node 3
Packet reached at sink node successfully
Route in which the packet travelled:
     3    10

Node 4
Packet Lost
Route in which the packet travelled:
     4    23    35    17    31    37    39    60

Node 5
Packet Lost
Route in which the packet travelled:
     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60
```

Fig: 3.2 :- Delivery status and route of packet transmission of 1st node to 5th node

```
main9.m   Figure 1
COMMAND WINDOW
Node 6
Packet Lost
Route in which the packet travelled:
     6    79    34    29

Node 7
Packet Lost
Route in which the packet travelled:
     7    50    19    47     4    23    35    17    31    37    39    60

Node 8
Packet reached at sink node successfully
Route in which the packet travelled:
     8    14

Node 9
Packet reached at sink node successfully
Route in which the packet travelled:
     9    52     8    14

Node 10
Packet reached at sink node successfully
Route in which the packet travelled:
    10
```

Fig: 3.3 :- Delivery status and route of packet transmission of 6th node to 10th node

```
main9.m   Figure 1
COMMAND WINDOW
Node 11
Packet Lost
Route in which the packet travelled:
    11    16    43    49    77    17    31    35    23     4    28    53    21    64    68    94    18    32    13     1     2     5    12    15    26    25    45    75    76    65    27

Node 12
Packet Lost
Route in which the packet travelled:
    12     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60

Node 13
Packet Lost
Route in which the packet travelled:
    13     1     2     5    12    15    26    25    45    75    28     4    23    35    17    31    37    39    60

Node 14
Packet reached at sink node successfully
Route in which the packet travelled:
    14

Node 15
Packet Lost
Route in which the packet travelled:
    15     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60
```

Fig: 3.4 :- Delivery status and route of packet transmission of 11th node to 15th node

```
main9.m   Figure 1
COMMAND WINDOW
Node 16
Packet Lost
Route in which the packet travelled:
    16    11

Node 17
Packet Lost
Route in which the packet travelled:
    17    31    35    23     4    28    53    21    64    68    94    18    32    13     1     2     5    12    15    26    25    45    75    76    65    27

Node 18
Packet Lost
Route in which the packet travelled:
    18    32    13     1     2     5    12    15    26    25    45    75    28     4    23    35    17    31    37    39    60

Node 19
Packet Lost
Route in which the packet travelled:
    19    47     4    23    35    17    31    37    39    60

Node 20
Packet Lost
Route in which the packet travelled:
    20    30     1     2     5    12    13    27    32    18    65    76    26    15    45    25
```

Fig: 3.5 :- Delivery status and route of packet transmission of 16th node to 20th node

```
main9.m   Figure 1
COMMAND WINDOW

Node 21
Packet Lost
Route in which the packet travelled:
    21    53    28     4    23    35    17    31    37    39    60

Node 22
Packet reached at sink node successfully
Route in which the packet travelled:
    22     8    14

Node 23
Packet Lost
Route in which the packet travelled:
    23     4    28    53    21    64    68    94    18    32    13     1     2     5    12    15    26    25    45    75    76    65    27

Node 24
Packet reached at sink node successfully
Route in which the packet travelled:
    24    58     9    52     8    14

Node 25
Packet Lost
Route in which the packet travelled:
    25    26     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60
```

Fig: 3.6:- Delivery status and route of packet transmission of 21st node to 25th node

```
main9.m   Figure 1
COMMAND WINDOW

Node 26
Packet Lost
Route in which the packet travelled:
    26     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60

Node 27
Packet Lost
Route in which the packet travelled:
    27    13     1     2     5    12    15    26    25    45    75    28     4    23    35    17    31    37    39    60

Node 28
Packet Lost
Route in which the packet travelled:
    28     4    23    35    17    31    37    39    60

Node 29
Packet Lost
Route in which the packet travelled:
    29    34    79     6

Node 30
Packet Lost
Route in which the packet travelled:
    30     1     2     5    12    13    27    32    18    65    76    26    15    45    25
```

Fig: 3.7 :- Delivery status and route of packet transmission of 26th node to 30th node

```
main9.m   Figure 1
COMMAND WINDOW
Node 50
Packet Lost
Route in which the packet travelled:
    50     7    71    19    47     4    23    35    17    31    37    39    60

Node 51
Packet reached at sink node successfully
Route in which the packet travelled:
    51     3    10

Node 52
Packet reached at sink node successfully
Route in which the packet travelled:
    52     8    14

Node 53
Packet Lost
Route in which the packet travelled:
    53    21    64    28     4    23    35    17    31    37    39    60

Node 54
Packet Lost
Route in which the packet travelled:
    54     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 55
```

Fig: 3.8 :- Delivery status and route of packet transmission of 50th node to 54th node

```
main9.m   Figure 1
COMMAND WINDOW
Node 61
Packet Lost
Route in which the packet travelled:
    61    19    47     4    23    35    17    31    37    39    60

Node 62
Packet reached at sink node successfully
Route in which the packet travelled:
    62    22     8    14

Node 63
Packet reached at sink node successfully
Route in which the packet travelled:
    63

Node 64
Packet Lost
Route in which the packet travelled:
    64    21    53    28     4    23    35    17    31    37    39    60

Node 65
Packet Lost
Route in which the packet travelled:
    65    13     1     2     5    12    15    26    25    45    75    28     4    23    35    17    31    37    39    60
```

Fig: 3.9 :- Delivery status and route of packet transmission of 61st node to 65^{h} node

```
main9.m   Figure 1
COMMAND WINDOW

Node 71
Packet Lost
Route in which the packet travelled:
    71     7    50    19    47     4    23    35    17    31    37    39    60

Node 72
Packet Lost
Route in which the packet travelled:
    72     7    50    19    47     4    23    35    17    31    37    39    60

Node 73
Packet Lost
Route in which the packet travelled:
    73     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 74
Packet Lost
Route in which the packet travelled:
    74     7    50    19    47     4    23    35    17    31    37    39    60

Node 75
Packet Lost
Route in which the packet travelled:
    75     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60
```

Fig: 3.10 :- Delivery status and route of packet transmission of 71st node to 75th node

```
main9.m   Figure 1
COMMAND WINDOW

Node 81
Packet Lost
Route in which the packet travelled:
    81    21    53    28     4    23    35    17    31    37    39    60

Node 82
Packet Lost
Route in which the packet travelled:
    82    20    30     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 83
Packet Lost
Route in which the packet travelled:
    83    56    66

Node 84
Packet reached at sink node successfully
Route in which the packet travelled:
    84     9    52     8    14

Node 85
Packet Lost
Route in which the packet travelled:
    85     4    23    35    17    31    37    39    60
```

Fig: 3.11 :- Delivery status and route of packet transmission of 81st node to 85th node

```
main9.m   Figure 1
COMMAND WINDOW

Node 86
Packet Lost
Route in which the packet travelled:
    86

Node 87
Packet reached at sink node successfully
Route in which the packet travelled:
    87    14

Node 88
Packet Lost
Route in which the packet travelled:
    88    40    20    30     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 89
Packet Lost
Route in which the packet travelled:
    89

Node 90
Packet Lost
Route in which the packet travelled:
    90    83    56    66
```

Fig: 3.12:- Delivery status and route of packet transmission of 86th node to 90th node

```
main9.m   Figure 1
COMMAND WINDOW

Node 91
Packet Lost
Route in which the packet travelled:
    91     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 92
Packet Lost
Route in which the packet travelled:
    92    15     5     1     2    30    20    33    71     7    50    19    47     4    23    35    17    31    37    39    60

Node 93
Packet Lost
Route in which the packet travelled:
    93

Node 94
Packet Lost
Route in which the packet travelled:
    94    18    32    13     1     2     5    12    15    26    25    45    75    28     4    23    35    17    31    37    39    60

Node 95
Packet Lost
Route in which the packet travelled:
    95    28     4    23    35    17    31    37    39    60
```

Fig: 3.13 :- Delivery status and route of packet transmission of 91st node to 95th node

```
main9.m   Figure 1
COMMAND WINDOW
    96    21    53    28     4    23    35    17    31    37    39    60

Node 97
Packet Lost
Route in which the packet travelled:
    97     2     1     5    12    13    27    32    18    65    76    26    15    45    25

Node 98
Packet reached at sink node successfully
Route in which the packet travelled:
    98     9    52     8    14

Node 99
Packet Lost
Route in which the packet travelled:
    99     1     2     5    12    13    27    32    18    65    76    26    15    45    25

Node 100
Packet reached at sink node successfully
Route in which the packet travelled:
   100     9    52     8    14

Average end to end delay = 2.040479 ms
Throughput = 63.079046 kbps
Packet Delivery Ratio = 0.200000
>>
```

Fig: 3.14 :- Delivery status and route of packet transmission of 97th node to 100th node and average E2ED, throughput and PDR is also displayed.

CHAPTER – 4

PERFORMANCE EVALUATION

We provide you a thorough evaluation of our protocol in this section. Initially 100 sensor nodes were implemented with a volume of 1000 m3 (i.e. Length=1000 m, width=1000 m, height=1000 m) in a 3-dimensional space. Initially a node's transmission range is 250 m. The average packet size to be transmitted is 80 bytes (640 bits).

Six sinks are now being installed, two of which are deployed under water in the places that are most likely to flow across the networks, which are built -in sink nodes, and the remaining four sinks are deployed on the water. Different node densities are simulated such that the average volume covered by a node will differ. The following Table shows the parameters used in the simulation.

variable	**value**
Sinks deployed on the water surface	4
Sinks deployed under water	2
Nodes deployed	100
Total Network Area	1000 m x 1000 m x 1000 m
Node transmission range	250 m
Signal propagation speed	1500 m/s
Average Packet Size	80 bytes

Table 4.1 : Parameters and their values used in the simulation

We take into account PDR, output and E2ED for testing protocol results. PDR is the percentage of the cumulative number of packets collected successfully from sink nodes to the network's total packets.

Mathematically,

$$PDR = \frac{Packets\ successfully\ received}{Packets\ sent}$$

The throughput tests the amount of bits obtained at destination per second.

$$\text{Throughput} = \frac{\text{Successful packets} * (\text{average packet_size})}{\text{Total Time sent in delivering that amount of data}}$$

The average time spent between a node beginning transmission and the packet received on the sink is E2ED. Several sinks will accept the packet in a multiple sink situation, the shortest one to pick. E2ED is a mixture of time spent on propagation, time spent on transmission, time cycle and processing.

We analyze the results in different transfer range and for different nodes in order to check the efficiency of the protocol.

Table 4.2 : PDR, Throughput and E2ED for different node transmission range

Node transmission range	PDR	Throughput (in kbps)	E2ED (in ms)
100	0.015000	2.909091	1.10553
150	0.035000	11.606218	1.939672
200	0.240000	15.891207	2.738814

Node transmission range	PDR	Throughput (in kbps)	E2ED (in ms)
250	0.310000	19.919679	3.005178
300	0.955000	25.353312	3.959848
350	1.000000	48.743336	4.191289
400	1.000000	52.315072	5.899672

Table 4.3 : PDR, Throughput and E2ED for different node transmission range

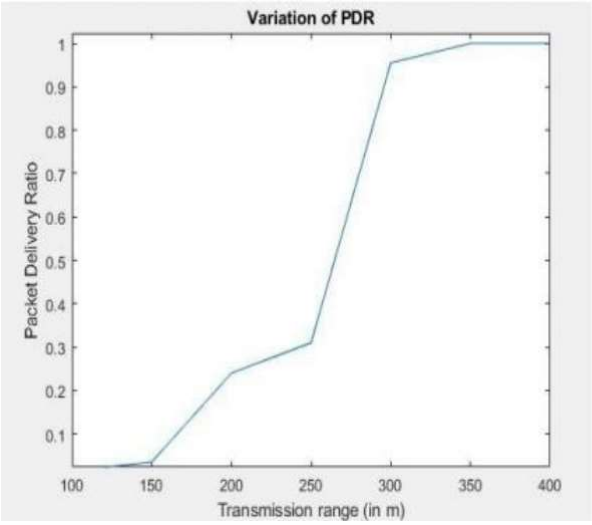
Table 2 & 3 displays PDR, transmission and E2ED values for various node transmission ranges. For the above simulations, the number of nodes is set at 200.

PDR improves as the transmission range increases and reaches to maximum value of 1 at range equal to 350m, with a larger transmission range there is a greater likelihood of having a sink node in its range resulting in a single and direct hop from source node to sink node.

The number of packets transmitted successfully increases, which means that the PDR is increasing. Figure 19 indicates that the PDR increases as the propagation range of the nodes increases.

If the number of active packets increases with time, throughput is also increased with the expansion of the transmission spectrum of nodes. The map demonstrates the improved capability by increasing the transmitting range in

Figure 20. E2ED also expands the propagation range, which is a disadvantage for this protocol. As the time to look up a sink node marginally increases, E2ED increases as well. Figure 21 indicates that the transmission range of E2ED increases.



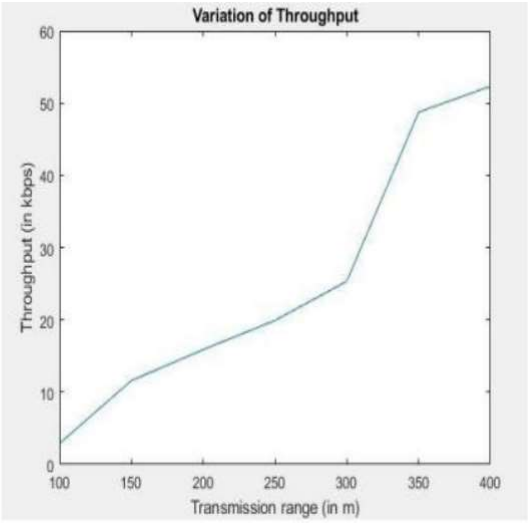


Fig: 4.1 :- Graph between PDR and transmission

Fig: 4.2 :- Graph between throughput range and transmission range

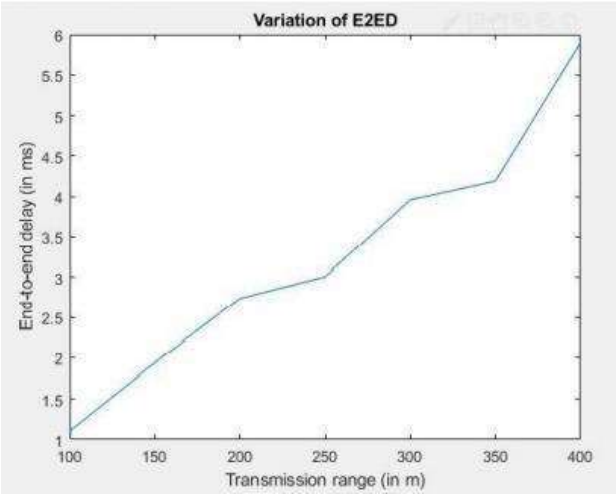


Fig: 4.3 :- Graph between E2ED and transmission range

Node transmission range	E2ED (in ms)	Throughput (in kbps)	PDR
100	1.248675	3.478261	0.010000
150	1.684640	7.119048	0.020000
200	2.035151	7.901235	0.075000
250	2.530129	12.151899	0.148000
300	3.601991	24.318814	0.236667
350	4.790854	32.952779	0.371429
400	4.995006	36.268806	0.482500
450	6.367500	53.258741	0.628889
500	7.569157	57.427210	0.878000

Table 4.4 : PDR, Throughput and E2ED for different node transmission range

For different nodes, Table 4 displays the values PDR, throughput and E2ED. For the above simulations, the transmission range of the nodes shall be set to 150m. PDR grows with increasing numbers of nodes, lower packet loss is creating a higher number of nodes. The graph shows that the PDR has risen with the number of nodes.

Figure 22. If the number of effective deliveries increases in the given time, the output also increases as the number of nodes increases. Figure 23 shows the rising throughput of the number of nodes.

E2ED also improves, which is again a downside of the protocol, by increasing the number of nodes. There is more time to measure and pick distances for more nodes, and therefore there are more hops as the number of nodes is higher, thereby increasing E2ED. The

increases of E2ED in number of nodes as seen in Figure 24

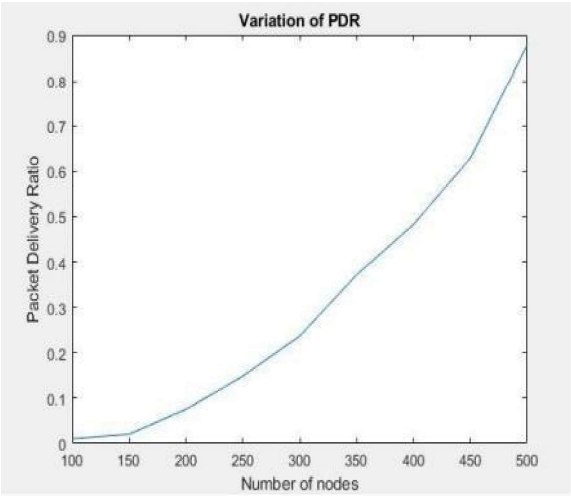


Fig: 4.4 :- Graph between PDR and number of nodes

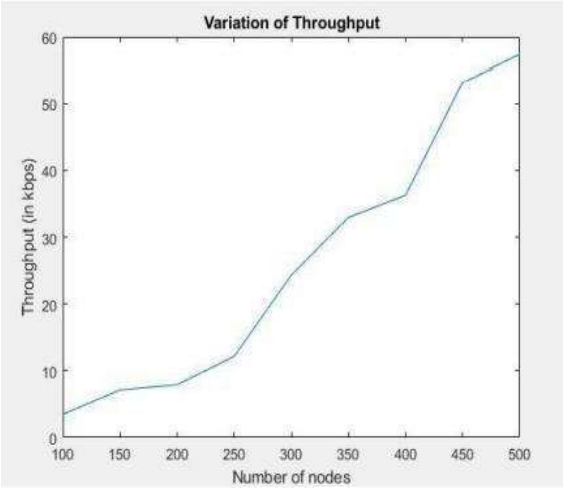


Fig: 4.5 :- Graph between throughput and number of nodes

Fig: 4.6 :- Graph between E2ED and number of nodes

CHAPTER – 5

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION:

There are numerous benefits and flaws of the efficient and expeditious routing protocol suggested. Vacuum holes are prevented, PDR and throughput are enhanced by taking into account PFNs, while E2ED is concurrently increased. With more transmission and more nodes, productivity is increasing, but this is at the cost of an increased E2ED.

The embedded sinks are achieving a high delivery and low E2ED ratio, but high communication is recorded with the surface sinks. Overall, this protocol has increased performance, improved PDR and reduced packages decline at E2ED costs.

The Packet drop is reduced and therefore, PDR is improved with the simultaneous selection of a high node density path. As a result, the performance is also increased.

FUTURE ENHANCEMENTS:

It is possible to broaden the proposed scheme to reduce E2ED. In our protocol to inspect your neighbours, we first compute the distance between a specific node and all the other nodes in the network, and then check which node is in the range of transmission and select an of them; instead we first fix the transmission range for the source node. This can dramatically reduce E2ED and also contribute to improved performance.

CHAPTER – 6

REFERENCES

✧ [1] I. F. Akyildiz, D. Pompili "Under water acoustic sensor networks: Research challenges," Ad Hoc Netw., vol. 3, no. 3, pp. 257–279, Mar. 2005.

✧ [2] W. K. G. Seah and H.-X. Tan, "Multipath virtual sink architecture for underwater sensor networks," in Proc. OCEANS Asia–Pacific, May 2006, pp. 1–6.

✧ [3] H. Yan, Z. J. Shi, and J.-HCui "DBR: Depth- based routing for underwater sensor networks," in Proc. Int. Conf. Res. Netw. Berlin, Germany: Springer, 2008, pp. 72–86.

✧ [4] M. R. Jafri, M. M. Sandhu, K. Latif, Z. A. Khan, A. U. Yasar, and N. Javaid, "Towards delay-sensitive routing in underwater wireless sensor networks," Procedia Comput. Sci., vol.

37, pp. 228–235, Jan. 2014.

✧ [5]D. Braginsky "Rumor routing algorthim for sensor networks," in Proc. 1st Int.

Workshop Wireless Sensor Netw. Appl., 2002, pp. 22–31.

✧ [6] F. Ye, G. Zhong, S. Lu "GRAdient broadcast: A robust data delivery protocol for large scale sensor networks," Wireless Netw., vol. 11, no. 3, pp. 285–298, 2005.

✧ [7] Y. Yao and "The cougar approach to in-network query processing in sensor networks," ACM SIGMOD Rec., vol. 31, no. 3, pp. 9–18, 2002.

✧ [8] P. Xie, Z. Zhou, N. Nicolaou, A. See, J.-H. Cui, and "Efficient vector-based forwarding for underwater sensor networks," EURASIP J. Wireless Commun. Netw., vol. 2010, no. 1,
2010, Art. no. 195910.

✧ [9] F. Ye, H. Luo, J. Cheng, S. Lu, and LZhang "A two- tier data dissemination model for large-scale wireless sensor networks," in Proc. 8th Annu. Int. Conf. Mobile Comput. Netw., 2002, pp. 148–159.

✧ [10] N. Javaid, M. R. Jafri, S. Ahmed, M. Jamil, Z. A. Khan, U. Qasim, and S. S. Al-Saleh, "Delay-sensitive routing schemes for underwater acoustic sensor networks," Int. J. Distrib. Sensor Netw., vol. 11, Mar. 2015, Art. no. 532676.

✧ [11] T. Ali, L. T. Jung, and I. Faye, "Diagonal and vertical routing protocol for underwater wireless sensor network," Proc.—Social Behav. Sci. vol. 129, pp. 372–379, May 2014.

✧ [12] M. Zuba, M. Fagan, Z. Shi, and J.-H. Cui, "A resilient pressure routing scheme for underwater acoustic networks," in Proc. IEEE Global Commun. Conf., Dec. 2014, pp. 637– 642.

✧ [13] Y. Noh, U. Lee, P. Wang, B. S. C. Choi "VAPR: Void-aware pressure routing for underwater sensor networks," IEEE Trans. Mobile Comput., vol. 12, no. 5, pp. 895–908, May 2013.

✧ [14] Z. Wadud, K. Ullah, S. Hussain, X. Yang, and A. B. Qazi, "Dow-pr DOlphin and whale pods routing protocol for underwater wireless sensor networks (UWSNs)," Sensors, vol. 18, no. 5, p. 1529, 2018.

✧ [15] H. Yu, N. Yao, and J. Liu, "An adaptive routing protocol in underwater sparse acoustic sensor networks," Ad Hoc Netw., vol. 34, pp. 121–143, Nov. 2015.

✧ [16] T. Hafeez, N. Javaid, A. R. Hameed, A. Sher, Z. A. Khan, and U. Qasim, "AVN-AHH-VBF: Avoiding void node with adaptive hop-by-hop vector based forwarding for underwater wireless sensor networks," in Proc. 10th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput. (IMIS), Jul. 2016, pp. 49–56.

✧ [17] Z. Wadud, S. Hussain, N. Javaid, S. H. Bouk, N. Alrajeh, M. S. Alabed "A n energy scaled and expanded vector-based forwarding scheme for industrial underwater acoustic sensor networks with sink mobility," Sensors, vol. 17, no. 10, p. 2251, 2017.

✧ [18] E. Magistretti, J. Kong, U. Lee, M. Gerla "A mobile delay-tolerant approach to long-term energy- efficient underwater sensor networking," in Proc. IEEE Wireless Commun. Netw. Conf., Hong Kong, Mar. 2007, pp. 2866–2871.

✧ [19] A. Wahid, S. Lee, H.-J. Jeong, and D. Kim, ‘‘EEDBR: Energy-efficient depth-based routing protocol for underwater wireless sensor networks,’’ in Proc. Int. Conf. Adv. Comput. Sci. Inf. Technol. Berlin, Germany: Springer, 2011, pp. 223–234.

✧ [20] C.-H. Yang and K.-F. Ssu. ‘‘A n energy-efficient routing protocol in underwater sensor networks,’’ in Proc. 3rd Int. Conf. Sens. Technol., Tainan, Taiwan, Nov. /Dec. 2008, pp. 114–118.

✧ [21] O. Cayirpunar, E. Kadioglu-Urtis, and B. Tavli, ‘‘Optimal base station mobility patterns for wireless sensor network lifetime maximization,’’ IEEE Sensors J., vol. 15, no. 11, pp. 6592–6603, Nov. 2015.

✧ [22] S. K. Dhurandher, M. S. Obaidat, and M. Gupta, ‘‘Providing reliable and link stability-based geocasting model in underwater environment,’’ Int. J. Commun. Syst., vol. 25, no. 3, pp.
356–375, 2012.