PROJECT REPORT ON

Simulation and Comparative Analysis of

Link-State Routing and Distance Vector Routing

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

Submitted by

123003187- PEDDI SATHWIK - CSE

Under the Guidance of

Dr.Dindayal Mahto

School of Computing
SASTRA DEEMED TO BE UNIVERSITY
(A University established under section 3 of the UGC Act, 1956)
Tirumalaisamudram
Thanjavur – 613401
January (2022)

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(SASTRA DEEMED TO BE UNIVERSITY)
(A University Established under section 3 of the UGC Act, 1956)

TIRUMALAISAMUDRAM, THANJAVUR – 613401

BONAFIDE CERTIFICATE

This is to certify that the project titled **"Simulation and Comparative Analysis of Link-State routing and Distance Vector Routing"** submitted as a requirement for the course, CSE302 : **MINI PROJECT** for B.Tech is a bonafide record of work done by **MR.PEDDI SATHWIK(123003187, CSE)** during the academic year 2021-2022, School of Computing, under my supervision.

Signature of project Supervisor :
Name with Affiliation :
Date :

Mini Project Viva Voce held on 11th February,2022.

Examiner – I **Examiner – II**

TABLE OF CONTENTS

S. NO	TOPIC	Page No.
1	ACKNOWLEDGEMENTS	4
2	ABSTRACT	5
3	LIST OF TABLES	6
4	LIST OF FIGURES	6
5	NOTATIONS	9
6	ABBREVIATIONS	9
7	CHAPTER 1 INTRODUCTION	11
8	CHAPTER 2 SOURCE CODE	19
9	CHAPTER 3 RESULTS	65
10	CHAPTER 4 PERFORMANCE EVALUATION	76
11	CHAPTER 5 CONCLUSION AND FUTURE WORKS	82
12	CHAPTER 6 REFERENCES	83

ACKNOWLEDGEMENTS

First of all, I would like to thank God Almighty for his endless blessings.

I would like to express my sincere gratitude to **Dr S. Vaidyasubramaniam, Vice-Chancellor** for his encouragement during the span of my academic life at SASTRA Deemed University.

I would forever remain grateful and I would like to thank **Dr A.Umamakeswri, Dean, School of Computing and R. Chandramouli, Registrar** for their overwhelming support provided during my course span in SASTRA Deemed University.

I am extremely grateful to **Dr. Shankar Sriram, Associate Dean, School of Computing** for his constant support, motivation and academic help extended for the past three years of my life in School of Computing.

I would specially thank and express my gratitude to **Dr. Dindayal Mahto , Associate Professor, School of Computing** for providing me an opportunity to do this project and for his guidance and support to successfully complete the project.

I also thank all the Teaching and Non-teaching faculty, and all other people who have directly or indirectly help me through their support, encouragement and all other assistance extended for completion of my project and for successful completion of all courses during my academic life at SASTRA Deemed University.

Finally, I thank my parents and all others who help me acquire this interest in project and aided me in completing it within the deadline without much struggle.

ABSTRACT

Routing is one of the most fundamental process in network communication. It is a process of routing the packets between multiple networks. Routing protocols are the set of defined rules used by the routers to communicate between the source and destination. These routing protocols learns the existing routes available in the network, builds the routing table and finds the efficient route for communication. It specifies how routers in the network communicate with each other to distribute the information.

There are two types of routing protocols, link state and distance vector routing protocols. The link state routing protocol uses the technique that each router shares the knowledge of its neighbor with every other router in the network. Link state routing is based on the assumption that each node has some knowledge about its neighbor nodes in the network. On the other hand, Distance vector routing protocol uses a routing algorithm in which routers send routing updates periodically to all neighbors by broadcasting the entire routing tables. Here link state routing protocol and distance vector routing protocol are implemented.

KEY WORDS: Computer Networks, Routing, Link State Routing, Distance Vector Routing.

List of Tables

Table No.	Table Name	Page No.
Table 1	Output of results of protocols simulated for 20 nodes on the execution of awk file	76
Table 2	Output of results of protocols simulated for 32 nodes on the execution of awk file.	79

List of Figures

Figure No.	Figure Name	Page No.
Figure 1	Work Flow Diagram and System Design of Link State Routing.	13
Figure 2	Work Flow Diagram and System Design of Distance Vector Routing	14
Figure 3	Simulation screen of 20 nodes DVTCP at the beginning .	65
Figure 4	Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 4,5 is down.	65
Figure 5	Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 9,14 is down.	65
Figure 6	Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 14,17 is down.	66
Figure 7	Simulation screen of 20 nodes DVUDP at beginning.	66
Figure 8	Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 4,5 is down.	66
Figure 9	Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 9,14 is down.	67
Figure 10	Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 14,17 is down.	67
Figure 11	Simulation screen of 20 nodes LSRTCP at the beginning.	67

Figure 12	Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 4,5 is down .	68
Figure 13	Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 9,14 is down .	68
Figure 14	Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 14,17 is down .	68
Figure 15	Simulation screen of 20 nodes LSRUDP at the beginning.	69
Figure 16	Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 4,5 is down.	69
Figure 17	Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 9,14is down.	69
Figure 18	Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 4,5 is down .	70
Figure 19	Simulation screen of 32 nodes DVTCP at the beginning.	70
Figure 20	Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 9,10 is down.	70
Figure 21	Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 7,18 is down .	71
Figure 22	Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 20,21 is down .	71
Figure 23	Simulation screen of 32 nodes DVUDP at the beginning .	71
Figure 24	Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 9,10 is down .	72
Figure 25	Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 7,18 is down .	72
Figure 26	Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 20,21 is down .	72
Figure 27	Simulation screen of 32 nodes LSRTCP at the beginning .	73
Figure 28	Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 9,10 is down.	73

Figure 29	Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 7,18 is down.	73
Figure 30	Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 20,21 is down.	74
Figure 31	Simulation screen of 32 nodes LSRUDP at the beginning.	74
Figure 32	Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 9,10 is down.	74
Figure 33	Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 7,18 is down.	75
Figure 34	Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 20,21 is down.	75
Figure 35	Throughput Analysis graph for the protocols simulated for 20 nodes.	77
Figure 36	Data received Analysis graph for the protocols simulated for 20 nodes.	77
Figure 37	Packet Delivery Ratio Analysis graph for the protocols simulated for 20 nodes.	78
Figure 38	Packet Delivery Ratio Analysis graph for the protocols simulated for 20 nodes.	78
Figure 39	Throughput Analysis graph for the protocols simulated for 32 nodes.	79
Figure 40	Data Received Analysis graph for the protocols simulated for 32 nodes.	80
Figure 41	Packet Delivery Ratio Analysis graph for the protocols simulated for 32 nodes.	80
Figure 42	Packet Loss Analysis graph for the protocols simulated for 32 nodes.	81

NOTATIONS

NOTATION	DESCRIPTION
D(v)	Current Value of Cost of path from source to destination v.
C(i,j)	Link cost from node i to j .
p(v)	Predecessor node along path from source to v i.e., next to v

ABBREVIATIONS

TCP	Transmission Control Protocol
UDP	User Datagram Protocol
DV	Distance vector Routing
LSR	Link-State Routing
LAN	Local Area Network
LSP	Link State Packet
WAN	Wide Area Network
DVTCP	20 nodes Distance Vector Routing using TCP
DVUDP	20 nodes Distance Vector Routing using UDP
LSRTCP	20 nodes Link-State Routing using TCP

LSRUDP	20 nodes Link-State Routing using UDP
DVTCP30	32 nodes Distance Vector Routing using TCP
DVUDP30	32 nodes Distance Vector Routing using UDP
LSRTCP30	32 nodes Link-State Routing using TCP
LSRUDP30	32 nodes Link-State Routing using UDP
PDR	Packet Delivery Ratio
NS-2	Network Simulator 2

CHAPTER 1

INTRODUCTION

Routing is one of the fundamental areas of network communication. Routing protocols are responsible for finding efficient path in network from source to destination. The purpose of routing protocol is to learn the available routes that exist on the network, build routing tables and make routing decisions. A routing protocol specifies how routers communicate with each other.

1.1. LINK STATE ROUTING PROTOCOL:

The basic concept of link state routing is that every node creates a map of the connectivity to the network in the form of graph, showing which nodes are connected to which other nodes. Each node calculates the best path from the other node independently. The collection of best paths will create a shortest path tree, then forms the node's routing table. In link state routing, four different tasks are required to ensure that each node has the routing table showing the least cost node to every node.

1) Creation of the states of the links by each node called Link state packet.

2) Dissemination of LSP data to every other router, called flooding.

3) Formation of a shortest path tree for each node.

4) Calculation of a routing table based on the shortest path tree.

1.1.1. Creation Of Link State Packet:

The node identity and the cost of the links are used to make the topology. Link state packets are generated either when there is a change in the topology or on periodic basis.

1.1.2. Flooding of LSPs:

After a node has created its LSP, it is sent to all the other nodes in the topology. This process is called flooding. It is done in the way that the creating nodes sends a copy of the LSP to its neighbor. The received LSP is compared with already created LSP. If the received LSP is not same as the created LSP, it sends a copy of it to each neighbor.

1.1.3. Formation Of Shortest Path Tree: After receiving the LSPs, each node will have a copy of the whole topology. Then, Dijkstra's algorithm is used to find the shortest path between every node in the topology.

1.1.4. Calculation Of Routing Table:

The routing table is constructed by using the shortest path tree obtained using Dijkstra's algorithm. The created routing table contains the cost of the nodes in the topology.

1.2. DISTANCE VECTOR ROUTING PROTOCOL:

Distance vector routing protocols use distance to find the best routing path for packets in network. It measures the distance based on how many hops data has to pass to get to its destination. The number of hops is essentially the number of routers it takes to reach the destination. In distance vector routing protocol, each router maintains a distance vector table containing the distance between itself and all the other possible destination nodes. In distance vector routing protocol, three different tasks are required to ensure that each node has the routing table showing the least cost node to every node.

1) Knowledge about the whole network,

2) Routing to the neighbors,

3) Information sharing at regular intervals.

1.2.1. Knowledge About The Whole Network:

The router collects the knowledge about the network and sends its collected knowledge to its neighbors.

1.2.2. Routing Only To Neighbors:

The router sends the collected knowledge to only routers on which the router has direct links. In this way, the information is received by the routers in the network and use this information to update its own routing table.

1.2.3. Information Sharing At Regular Intervals:

The router in the network sends the information to the neighboring routers within seconds.

2.WORK FLOW DIAGRAM AND SYSTEM DESIGN

2.1. LINK STATE ROUTING ALGORITHM:

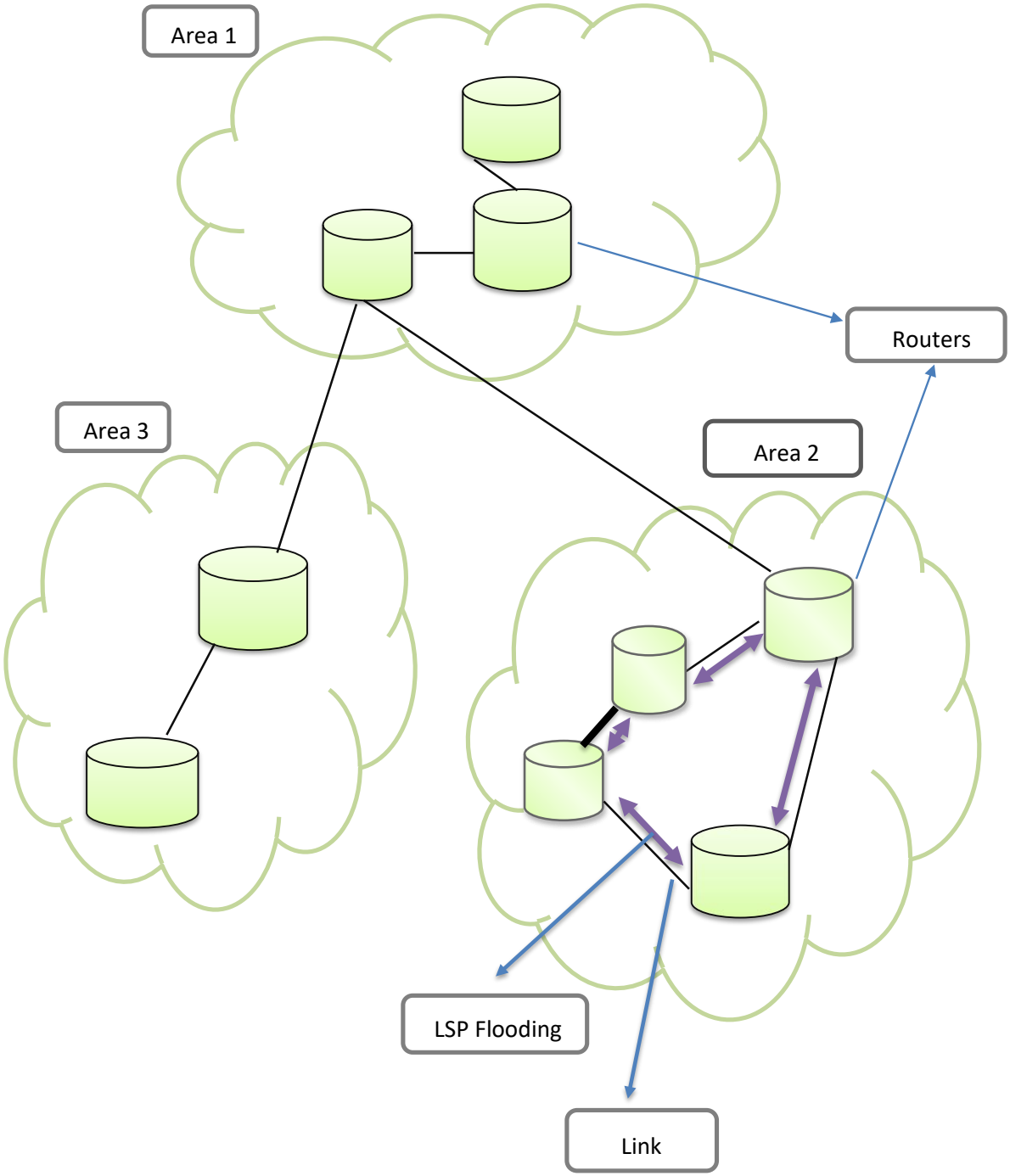

Figure-01: Work Flow Diagram and System Design of Link State Routing

2.2. DISTANCE VECTOR ROUTING ALGORITHM:

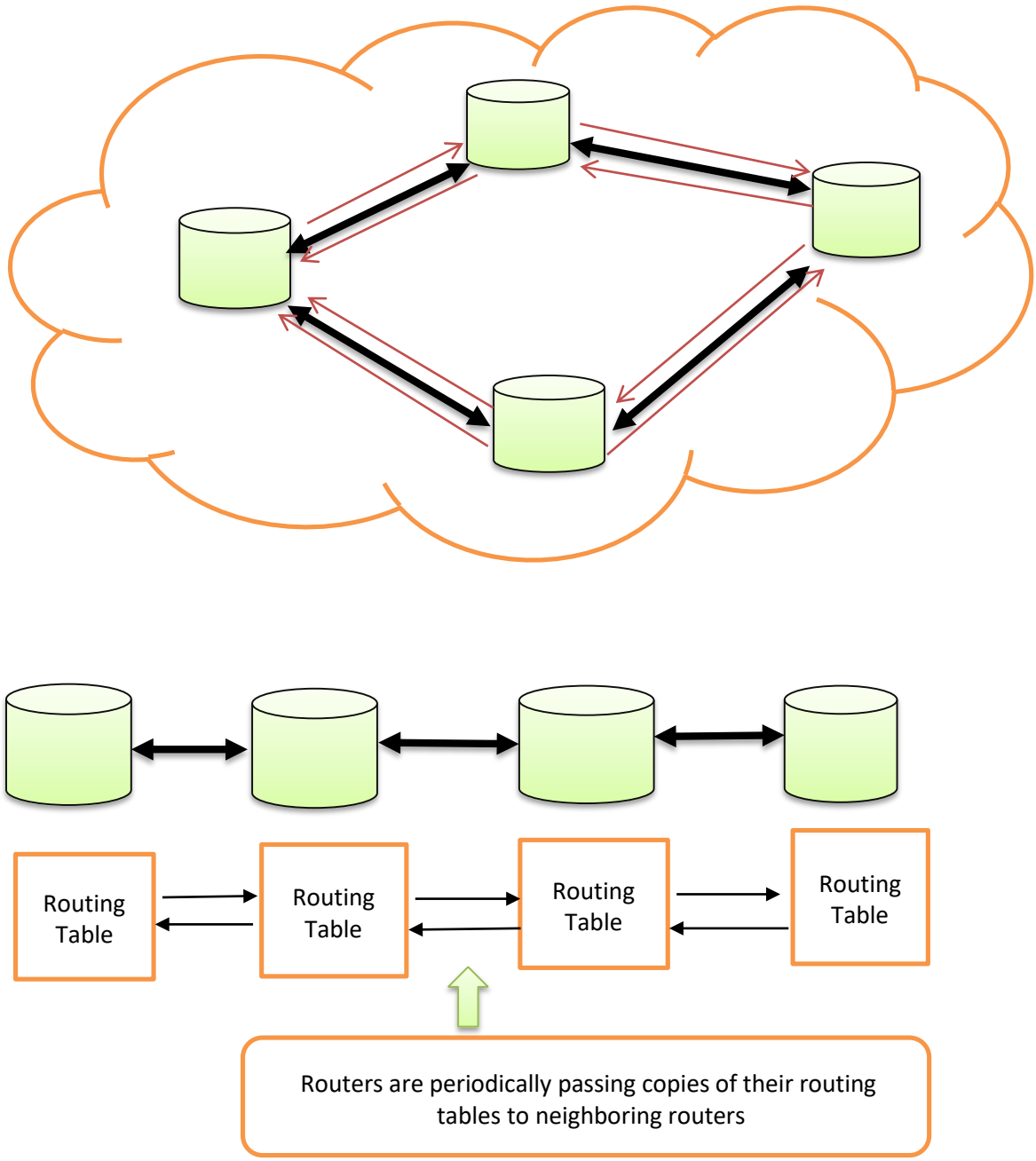

Figure-02: Work Flow Diagram and System Design of Distance Vector Routing

3.METHODOLOGIES USED

3.1. LINK STATE ROUTING PROTOCOL:

In link state routing algorithm, Djikstra algorithm is used to find the shortest path tree in the network topology.

3.1.1.Djikstra's Algorithm:

A shortest path tree is a tree in which the path between the root and every other node is shortest. The algorithm does two following steps:

1) Initialization

2) Iteration

Initialization: It selects a node as the root of the tree and add it to the path. It sets the shortest distances for all nodes in the topology. It sets the shortest distance of the root to zero.

Iteration: It repeats the same process until all nodes are added to the path. It adds a new node by checking whether the node is already added in the path or not. It adds that node if that node is not already added in the path. It stops the iteration when the source node reaches the destination node. At the end, we will get the shortest distance between each node in the topology.

```
Initialization
N = {A}     // A is a root node.
for all nodes v
if v adjacent to A
then D(v) = c(A,v)
else D(v) = infinity
loop
find w not in N such that D(w) is a minimum.
Add w to N
Update D(v) for all v adjacent to w and not in N:
D(v) = min(D(v) , D(w) + c(w,v))
Until all nodes in N
```

3.2.DISTANCE VECTOR ROUTING PROTOCOL:

In Distance-vector routing protocol, the router informs the knowledge of topology to its neighbors periodically. It uses Bellman-Ford algorithm. In Distance-vector routing protocol, each router maintains a distance vector table containing the distance between itself and all possible destination nodes.

3.2.1. DISTANCE VECTOR ALGORITHM:

The main difference between link state and distance vector protocol is that in distance vector protocol, the routers does not know the entire path to reach its destination. Instead, distance vector uses two methods such as direction in which the packets should be forwarded and the distance from its destination.

In distance vector algorithm, a router sends its distance vector to each of its neighbors.

1) Each router in the topology receives the distance vector and saves the most recently received distance vector from each of its neighbors.

2) A router recalculates its distance vector when it receives a distance vector from a neighbor containing different information than previously received distance vector or when it discovers that a link to a neighbor has gone down.

```
At each node x,

Initialization


for all destinations y in N:
Dx(y) = c(x,y)     // If y is not a neighbor then c(x,y) = ∞
for each neighbor w
Dw(y) = ?     for all destination y in N.
for each neighbor w
send distance vector Dx = [ Dx(y)  : y in N ] to w
loop
  wait(until I receive any distance vector from some neighbor w)
  for each y in N:
  Dx(y) = minv{c(x,v)+Dv(y)}
If Dx(y) is changed for any destination y
Send distance vector Dx = [ Dx(y) : y in N ] to all neighbors
forever
```

4.MERITS AND DEMERITS

4.1. LINK STATE ROUTING PROTOCOL:

4.1.1. Merits:

1) Link-state routing protocols use cost metrics to choose the paths in the topology. The cost metric reflects the capacity of the links on those paths.

2) Link-state protocols use triggered updates and LSA floods to immediately report changes in the network topology to all routers in the network and this results in fast convergence times.

3) Each router has a complete and synchronized picture of the network. Therefore, it is very difficult for routing loops to occur.

4) Routers in the link state routing protocols use the latest information to make the best routing decisions.

5) The link-state database sizes can be minimized with careful network design. This leads to faster convergence and smaller Dijkstra calculations.

4.1.2. Demerits:

1) It requires more processor power and memory than distance vector protocols. Hence it is expensive to use.

2) It requires strict hierarchical network design, so that a network can be broken into smaller areas to reduce the size of the topology tables.

3) They flood the network with LSAs during the initial discovery process. This process can reduce the capability of network to transport data. It degrades the network performance.

4.2. DISTANCE VECTOR ROUTING PROTOCOL:

4.2.1.Merits:

1) Distance vector routing protocols are easy to implement.

2) Its configuration is very simple and the maintenance is very simpler than link state routing protocol.

3) It requires less memory and processing power than link state routing protocol.

4) Bandwidth requirements are lesser for LAN networks.

4.2.2. Demerits:

1) Distance vector routing results in larger routing tables than link state for larger networks since each router must know about all other routers. This can also lead to congestion on WAN links.

2) It converges slowly than link state routing protocol.

3) Bandwidth requirements can be large for WAN or complex LAN environments.

CHAPTER 2

SOURCE CODE

DVTCP.tcl:

```
set ns [new Simulator]
set nr [open DVTCP.tr w]
$ns trace-all $nr
set nf [open DVTCP.nam w]
$ns namtrace-all $nf
# ----FINISH PROCEDURE----
proc finish {} {
  global ns nr nf
  $ns flush-trace
  close $nf
  close $nr
  exec nam DVTCP.nam &
  exit 0
}
 # ----CREATION OF NODES----
for {set i 0} {$i < 20} {incr i} {
Set n$i [$ns node]
}
# ----CREATION OF LINKS----
```

```
$ns duplex-link $n0 $n1 6Mb 12ms DropTail

$ns duplex-link $n1 $n2 15Mb 20ms DropTail

$ns duplex-link $n2 $n4 11Mb 15ms DropTail

$ns duplex-link $n4 $n3 15Mb 13ms DropTail

$ns duplex-link $n3 $n6 8Mb 19ms DropTail

$ns duplex-link $n4 $n7 13Mb 12ms DropTail

$ns duplex-link $n0 $n4 15Mb 19ms DropTail

$ns duplex-link $n4 $n6 12Mb 10ms DropTail

$ns duplex-link $n4 $n5 15Mb 18ms DropTail

$ns duplex-link $n7 $n5 20Mb 19ms DropTail

$ns duplex-link $n6 $n8 9Mb 16ms DropTail

$ns duplex-link $n7 $n8 19Mb 17ms DropTail

$ns duplex-link $n5 $n9 17Mb 20ms DropTail

$ns duplex-link $n9 $n11 5Mb 13ms DropTail

$ns duplex-link $n8 $n11 18Mb 11ms DropTail

$ns duplex-link $n5 $n10 13Mb 18ms DropTail

$ns duplex-link $n10 $n12 15Mb 13ms DropTail

$ns duplex-link $n9 $n12 8Mb 15ms DropTail

$ns duplex-link $n9 $n14 16Mb 12ms DropTail

$ns duplex-link $n11 $n13 8Mb 18ms DropTail

$ns duplex-link $n12 $n15 11Mb 19ms DropTail

$ns duplex-link $n14 $n16 6Mb 14ms DropTail

$ns duplex-link $n14 $n17 20Mb 13ms DropTail
```

```
$ns duplex-link $n14 $n18 18Mb 16ms DropTail

$ns duplex-link $n14 $n19 9Mb 5ms DropTail

$ns duplex-link $n15 $n19 20Mb 18ms DropTail

$ns duplex-link $n16 $n17 8Mb 11ms DropTail

$ns duplex-link $n17 $n18 11Mb 18ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n0 $n1 orient down

$ns duplex-link-op $n1 $n2 orient down

$ns duplex-link-op $n2 $n4 orient right-up

$ns duplex-link-op $n3 $n4 orient down

$ns duplex-link-op $n3 $n6 orient right

$ns duplex-link-op $n4 $n7 orient right

$ns duplex-link-op $n0 $n4 orient right-down

$ns duplex-link-op $n4 $n6 orient right-up

$ns duplex-link-op $n4 $n5 orient right-down

$ns duplex-link-op $n7 $n5 orient down

$ns duplex-link-op $n6 $n8 orient right

$ns duplex-link-op $n7 $n8 orient right-up

$ns duplex-link-op $n5 $n9 orient right-up

$ns duplex-link-op $n9 $n11 orient right-up

$ns duplex-link-op $n8 $n11 orient right

$ns duplex-link-op $n5 $n10 orient right

$ns duplex-link-op $n10 $n12 orient right
```

```
$ns duplex-link-op $n9 $n12 orient right-
down

$ns duplex-link-op $n9 $n14 orient right

$ns duplex-link-op $n11 $n13 orient right

$ns duplex-link-op $n12 $n15 orient right

$ns duplex-link-op $n14 $n16 orient right-up

$ns duplex-link-op $n14 $n17 orient right-up

$ns duplex-link-op $n14 $n18 orient right

$ns duplex-link-op $n14 $n19 orient right-
down

$ns duplex-link-op $n15 $n19 orient right

$ns duplex-link-op $n16 $n17 orient right

$ns duplex-link-op $n17 $n18 orient down

$ns rtproto DV

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n4 $n5

$ns rtmodel-at 4.5 up $n4 $n5

$ns rtmodel-at 8.0 down $n9 $n14

$ns rtmodel-at 8.3 up $n9 $n14

$ns rtmodel-at 15.0 down $n14 $n17

$ns rtmodel-at 15.3 up $n14 $n17

# ----CREATING AND ATTACHING AGENTS----

set tcp [new Agent/TCP]

$tcp set class_ 1
```

```
$ns attach-agent $n0 $tcp

set ftp [new Application/FTP]

$ftp set type_ FTP

$ftp set packet_size_ 3000

$ftp attach-agent $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n17 $sink

$ns connect $tcp $sink

$tcp set fid_ 1

$ns color 1 Blue

$ns at 0.0 "$n0 color chocolate"

$ns at 0.0 "$n17 color chocolate"

$ns at 0.3 "$ftp start"

$ns at 20 "finish"

$ns run
```

DVUDP.tcl:

```
set ns [new Simulator]

set nr [open DVUDP.tr w]

$ns trace-all $nr

set nf [open DVUDP.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----
```

```
proc finish {} {
  global ns nr nf
  $ns flush-trace
  close $nf
  close $nr
  exec nam DVUDP.nam &
  exit 0
}
 # ----CREATION OF NODES----
for {set i 0} {$i < 20} {incr i} {
Set n$i [$ns node]
}
# ----CREATION OF LINKS----
$ns duplex-link $n0 $n1 6Mb 12ms DropTail
$ns duplex-link $n1 $n2 15Mb 20ms DropTail
$ns duplex-link $n2 $n4 11Mb 15ms DropTail
$ns duplex-link $n4 $n3 15Mb 13ms DropTail
$ns duplex-link $n3 $n6 8Mb 19ms DropTail
$ns duplex-link $n4 $n7 13Mb 12ms DropTail
$ns duplex-link $n0 $n4 15Mb 19ms DropTail
$ns duplex-link $n4 $n6 12Mb 10ms DropTail
$ns duplex-link $n4 $n5 15Mb 18ms DropTail
$ns duplex-link $n7 $n5 20Mb 19ms DropTail
```

```
$ns duplex-link $n6 $n8 9Mb 16ms DropTail

$ns duplex-link $n7 $n8 19Mb 17ms DropTail

$ns duplex-link $n5 $n9 17Mb 20ms DropTail

$ns duplex-link $n9 $n11 5Mb 13ms DropTail

$ns duplex-link $n8 $n11 18Mb 11ms DropTail

$ns duplex-link $n5 $n10 13Mb 18ms DropTail

$ns duplex-link $n10 $n12 15Mb 13ms DropTail

$ns duplex-link $n9 $n12 8Mb 15ms DropTail

$ns duplex-link $n9 $n14 16Mb 12ms DropTail

$ns duplex-link $n11 $n13 8Mb 18ms DropTail

$ns duplex-link $n12 $n15 11Mb 19ms DropTail

$ns duplex-link $n14 $n16 6Mb 14ms DropTail

$ns duplex-link $n14 $n17 20Mb 13ms DropTail

$ns duplex-link $n14 $n18 18Mb 16ms DropTail

$ns duplex-link $n14 $n19 9Mb 5ms DropTail

$ns duplex-link $n15 $n19 20Mb 18ms DropTail

$ns duplex-link $n16 $n17 8Mb 11ms DropTail

$ns duplex-link $n17 $n18 11Mb 18ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n0 $n1 orient down

$ns duplex-link-op $n1 $n2 orient down

$ns duplex-link-op $n2 $n4 orient right-up

$ns duplex-link-op $n3 $n4 orient down
```

```
$ns duplex-link-op $n3 $n6 orient right

$ns duplex-link-op $n4 $n7 orient right

$ns duplex-link-op $n0 $n4 orient right-down

$ns duplex-link-op $n4 $n6 orient right-up

$ns duplex-link-op $n4 $n5 orient right-down

$ns duplex-link-op $n7 $n5 orient down

$ns duplex-link-op $n6 $n8 orient right

$ns duplex-link-op $n7 $n8 orient right-up

$ns duplex-link-op $n5 $n9 orient right-up

$ns duplex-link-op $n9 $n11 orient right-up

$ns duplex-link-op $n8 $n11 orient right

$ns duplex-link-op $n5 $n10 orient right

$ns duplex-link-op $n10 $n12 orient right

$ns duplex-link-op $n9 $n12 orient right-
down

$ns duplex-link-op $n9 $n14 orient right

$ns duplex-link-op $n11 $n13 orient right

$ns duplex-link-op $n12 $n15 orient right

$ns duplex-link-op $n14 $n16 orient right-up

$ns duplex-link-op $n14 $n17 orient right-up

$ns duplex-link-op $n14 $n18 orient right

$ns duplex-link-op $n14 $n19 orient right-
down

$ns duplex-link-op $n15 $n19 orient right
```

```
$ns duplex-link-op $n16 $n17 orient right

$ns duplex-link-op $n17 $n18 orient down

$ns rtproto DV

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n4 $n5

$ns rtmodel-at 4.5 up $n4 $n5

$ns rtmodel-at 8.0 down $n9 $n14

$ns rtmodel-at 8.3 up $n9 $n14

$ns rtmodel-at 15.0 down $n14 $n17

$ns rtmodel-at 15.3 up $n14 $n17

# ----CREATING AND ATTACHING AGENTS----

set udp [new Agent/UDP]

$udp set class_ 1

$ns attach-agent $n0 $udp

set cbr [new Application/Traffic/CBR]

$cbr set type_ CBR

$cbr set packet_size_ 3000

$cbr attach-agent $udp

set null [new Agent/Null]

$ns attach-agent $n17 $null

$ns connect $udp $null

$udp set fid_ 1

$ns color 1 Red
```

```
$ns at 0.0 "$n0 color chocolate"

$ns at 0.0 "$n17 color chocolate"

$ns at 0.3 "$cbr start"

$ns at 20 "finish"

$ns run
```

LSRTCP.tcl:

```
set ns [new Simulator]

set nr [open LSRTCP.tr w]

$ns trace-all $nr

set nf [open LSRTCP.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----

proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam LSRTCP.nam &

  exit 0

}

 # ----CREATION OF NODES----

for {set i 0} {$i < 20} {incr i} {

Set n$i [$ns node]
```

```
}
# ----CREATION OF LINKS----
$ns duplex-link $n0 $n1 6Mb 12ms DropTail
$ns duplex-link $n1 $n2 15Mb 20ms DropTail
$ns duplex-link $n2 $n4 11Mb 15ms DropTail
$ns duplex-link $n4 $n3 15Mb 13ms DropTail
$ns duplex-link $n3 $n6 8Mb 19ms DropTail
$ns duplex-link $n4 $n7 13Mb 12ms DropTail
$ns duplex-link $n0 $n4 15Mb 19ms DropTail
$ns duplex-link $n4 $n6 12Mb 10ms DropTail
$ns duplex-link $n4 $n5 15Mb 18ms DropTail
$ns duplex-link $n7 $n5 20Mb 19ms DropTail
$ns duplex-link $n6 $n8 9Mb 16ms DropTail
$ns duplex-link $n7 $n8 19Mb 17ms DropTail
$ns duplex-link $n5 $n9 17Mb 20ms DropTail
$ns duplex-link $n9 $n11 5Mb 13ms DropTail
$ns duplex-link $n8 $n11 18Mb 11ms DropTail
$ns duplex-link $n5 $n10 13Mb 18ms DropTail
$ns duplex-link $n10 $n12 15Mb 13ms DropTail
$ns duplex-link $n9 $n12 8Mb 15ms DropTail
$ns duplex-link $n9 $n14 16Mb 12ms DropTail
$ns duplex-link $n11 $n13 8Mb 18ms DropTail
$ns duplex-link $n12 $n15 11Mb 19ms DropTail
```

```
$ns duplex-link $n14 $n16 6Mb 14ms DropTail

$ns duplex-link $n14 $n17 20Mb 13ms DropTail

$ns duplex-link $n14 $n18 18Mb 16ms DropTail

$ns duplex-link $n14 $n19 9Mb 5ms DropTail

$ns duplex-link $n15 $n19 20Mb 18ms DropTail

$ns duplex-link $n16 $n17 8Mb 11ms DropTail

$ns duplex-link $n17 $n18 11Mb 18ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n0 $n1 orient down

$ns duplex-link-op $n1 $n2 orient down

$ns duplex-link-op $n2 $n4 orient right-up

$ns duplex-link-op $n3 $n4 orient down

$ns duplex-link-op $n3 $n6 orient right

$ns duplex-link-op $n4 $n7 orient right

$ns duplex-link-op $n0 $n4 orient right-down

$ns duplex-link-op $n4 $n6 orient right-up

$ns duplex-link-op $n4 $n5 orient right-down

$ns duplex-link-op $n7 $n5 orient down

$ns duplex-link-op $n6 $n8 orient right

$ns duplex-link-op $n7 $n8 orient right-up

$ns duplex-link-op $n5 $n9 orient right-up

$ns duplex-link-op $n9 $n11 orient right-up

$ns duplex-link-op $n8 $n11 orient right
```

```
$ns duplex-link-op $n5 $n10 orient right

$ns duplex-link-op $n10 $n12 orient right

$ns duplex-link-op $n9 $n12 orient right-
down

$ns duplex-link-op $n9 $n14 orient right

$ns duplex-link-op $n11 $n13 orient right

$ns duplex-link-op $n12 $n15 orient right

$ns duplex-link-op $n14 $n16 orient right-up

$ns duplex-link-op $n14 $n17 orient right-up

$ns duplex-link-op $n14 $n18 orient right

$ns duplex-link-op $n14 $n19 orient right-
down

$ns duplex-link-op $n15 $n19 orient right

$ns duplex-link-op $n16 $n17 orient right

$ns duplex-link-op $n17 $n18 orient down

$ns rtproto LS

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n4 $n5

$ns rtmodel-at 4.5 up $n4 $n5

$ns rtmodel-at 8.0 down $n9 $n14

$ns rtmodel-at 8.3 up $n9 $n14

$ns rtmodel-at 15.0 down $n14 $n17

$ns rtmodel-at 15.3 up $n14 $n17

# ----CREATING AND ATTACHING AGENTS----
```

```
set tcp [new Agent/TCP]

$tcp set class_ 1

$ns attach-agent $n0 $tcp

set ftp [new Application/FTP]

$ftp set type_ FTP

$ftp set packet_size_ 3000

$ftp attach-agent $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n17 $sink

$ns connect $tcp $sink

$tcp set fid_ 1

$ns color 1 Blue

$ns at 0.0 "$n0 color chocolate"

$ns at 0.0 "$n17 color chocolate"

$ns at 0.3 "$ftp start"

$ns at 20 "finish"

$ns run
```

LSRUDP.tcl:

```
set ns [new Simulator]

set nr [open LSRUDP.tr w]

$ns trace-all $nr

set nf [open LSRUDP.nam w]
```

```
$ns namtrace-all $nf

# ----FINISH PROCEDURE----

proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam LSRUDP.nam &

  exit 0

}

 # ----CREATION OF NODES----

for {set i 0} {$i < 20} {incr i} {

Set n$i [$ns node]

}

# ----CREATION OF LINKS----

$ns duplex-link $n0 $n1 6Mb 12ms DropTail

$ns duplex-link $n1 $n2 15Mb 20ms DropTail

$ns duplex-link $n2 $n4 11Mb 15ms DropTail

$ns duplex-link $n4 $n3 15Mb 13ms DropTail

$ns duplex-link $n3 $n6 8Mb 19ms DropTail

$ns duplex-link $n4 $n7 13Mb 12ms DropTail

$ns duplex-link $n0 $n4 15Mb 19ms DropTail

$ns duplex-link $n4 $n6 12Mb 10ms DropTail
```

```
$ns duplex-link $n4 $n5 15Mb 18ms DropTail

$ns duplex-link $n7 $n5 20Mb 19ms DropTail

$ns duplex-link $n6 $n8 9Mb 16ms DropTail

$ns duplex-link $n7 $n8 19Mb 17ms DropTail

$ns duplex-link $n5 $n9 17Mb 20ms DropTail

$ns duplex-link $n9 $n11 5Mb 13ms DropTail

$ns duplex-link $n8 $n11 18Mb 11ms DropTail

$ns duplex-link $n5 $n10 13Mb 18ms DropTail

$ns duplex-link $n10 $n12 15Mb 13ms DropTail

$ns duplex-link $n9 $n12 8Mb 15ms DropTail

$ns duplex-link $n9 $n14 16Mb 12ms DropTail

$ns duplex-link $n11 $n13 8Mb 18ms DropTail

$ns duplex-link $n12 $n15 11Mb 19ms DropTail

$ns duplex-link $n14 $n16 6Mb 14ms DropTail

$ns duplex-link $n14 $n17 20Mb 13ms DropTail

$ns duplex-link $n14 $n18 18Mb 16ms DropTail

$ns duplex-link $n14 $n19 9Mb 5ms DropTail

$ns duplex-link $n15 $n19 20Mb 18ms DropTail

$ns duplex-link $n16 $n17 8Mb 11ms DropTail

$ns duplex-link $n17 $n18 11Mb 18ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n0 $n1 orient down

$ns duplex-link-op $n1 $n2 orient down
```

```
$ns duplex-link-op $n2 $n4 orient right-up

$ns duplex-link-op $n3 $n4 orient down

$ns duplex-link-op $n3 $n6 orient right

$ns duplex-link-op $n4 $n7 orient right

$ns duplex-link-op $n0 $n4 orient right-down

$ns duplex-link-op $n4 $n6 orient right-up

$ns duplex-link-op $n4 $n5 orient right-down

$ns duplex-link-op $n7 $n5 orient down

$ns duplex-link-op $n6 $n8 orient right

$ns duplex-link-op $n7 $n8 orient right-up

$ns duplex-link-op $n5 $n9 orient right-up

$ns duplex-link-op $n9 $n11 orient right-up

$ns duplex-link-op $n8 $n11 orient right

$ns duplex-link-op $n5 $n10 orient right

$ns duplex-link-op $n10 $n12 orient right

$ns duplex-link-op $n9 $n12 orient right-
down

$ns duplex-link-op $n9 $n14 orient right

$ns duplex-link-op $n11 $n13 orient right

$ns duplex-link-op $n12 $n15 orient right

$ns duplex-link-op $n14 $n16 orient right-up

$ns duplex-link-op $n14 $n17 orient right-up

$ns duplex-link-op $n14 $n18 orient right
```

```
$ns duplex-link-op $n14 $n19 orient right-
down

$ns duplex-link-op $n15 $n19 orient right

$ns duplex-link-op $n16 $n17 orient right

$ns duplex-link-op $n17 $n18 orient down

$ns rtproto LS

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n4 $n5

$ns rtmodel-at 4.5 up $n4 $n5

$ns rtmodel-at 8.0 down $n9 $n14

$ns rtmodel-at 8.3 up $n9 $n14

$ns rtmodel-at 15.0 down $n14 $n17

$ns rtmodel-at 15.3 up $n14 $n17

# ----CREATING AND ATTACHING AGENTS----

set udp [new Agent/UDP]

$udp set class_ 1

$ns attach-agent $n0 $udp

set cbr [new Application/Traffic/CBR]

$cbr set type_ CBR

$cbr set packet_size_ 3000

$cbr attach-agent $udp

set null [new Agent/Null]

$ns attach-agent $n17 $null
```

```
$ns connect $udp $null

$udp set fid_ 1

$ns color 1 Red

$ns at 0.0 "$n0 color chocolate"

$ns at 0.0 "$n17 color chocolate"

$ns at 0.3 "$cbr start"

$ns at 20 "finish"

$ns run
```

DVTCP30.tcl:

```
set ns [new Simulator]

set nr [open DVTCP30.tr w]

$ns trace-all $nr

set nf [open DVTCP30.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----


proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam DVTCP30.nam &

  exit 0
```

```
}
 # ----CREATION OF NODES----
for {set i 0} {$i < 32} {incr i} {
Set n$i [$ns node]
}
# ----CREATION OF LINKS----
$ns duplex-link $n6 $n8 14Mb 12ms DropTail
$ns duplex-link $n8 $n0 15Mb 16ms DropTail
$ns duplex-link $n0 $n13 16Mb 15ms DropTail
$ns duplex-link $n6 $n13 20Mb 17ms DropTail
$ns duplex-link $n8 $n9 12Mb 19ms DropTail
$ns duplex-link $n0 $n1 10Mb 20ms DropTail
$ns duplex-link $n9 $n10 5Mb 10ms DropTail
$ns duplex-link $n1 $n14 13Mb 12ms DropTail
$ns duplex-link $n14 $n15 15Mb 17ms DropTail
$ns duplex-link $n15 $n30 14Mb 14ms DropTail
$ns duplex-link $n14 $n30 10Mb 18ms DropTail
$ns duplex-link $n10 $n31 19Mb 13ms DropTail
$ns duplex-link $n10 $n3 6Mb 10ms DropTail
$ns duplex-link $n10 $n11 17Mb 15ms DropTail
$ns duplex-link $n31 $n11 10Mb 15ms DropTail
$ns duplex-link $n3 $n4 7Mb 16ms DropTail
$ns duplex-link $n3 $n15 6Mb 17ms DropTail
```

```
$ns duplex-link $n15 $n16 7Mb 10ms DropTail

$ns duplex-link $n11 $n4 18Mb 14ms DropTail

$ns duplex-link $n4 $n16 8Mb 17ms DropTail

$ns duplex-link $n11 $n12 15Mb 17ms DropTail

$ns duplex-link $n4 $n5 12Mb 18ms DropTail

$ns duplex-link $n12 $n2 7Mb 11ms DropTail

$ns duplex-link $n5 $n17 11Mb 15ms DropTail

$ns duplex-link $n12 $n7 12Mb 15ms DropTail

$ns duplex-link $n17 $n7 8Mb 16ms DropTail

$ns duplex-link $n17 $n29 16Mb 20ms DropTail

$ns duplex-link $n7 $n18 9Mb 15ms DropTail

$ns duplex-link $n29 $n23 18Mb 11ms DropTail

$ns duplex-link $n18 $n19 17Mb 16ms DropTail

$ns duplex-link $n18 $n23 14Mb 15ms DropTail

$ns duplex-link $n23 $n24 14Mb 14ms DropTail

$ns duplex-link $n24 $n28 15Mb 14ms DropTail

$ns duplex-link $n23 $n28 5Mb 13ms DropTail

$ns duplex-link $n19 $n20 19Mb 20ms DropTail

$ns duplex-link $n19 $n26 11Mb 15ms DropTail

$ns duplex-link $n24 $n25 8Mb 18ms DropTail

$ns duplex-link $n26 $n25 13Mb 16ms DropTail

$ns duplex-link $n20 $n21 12Mb 16ms DropTail

$ns duplex-link $n21 $n22 9Mb 20ms DropTail
```

```
$ns duplex-link $n22 $n25 13Mb 10ms DropTail

$ns duplex-link $n21 $n27 7Mb 19ms DropTail

$ns duplex-link $n22 $n27 6Mb 19ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n6 $n8 orient right-up

$ns duplex-link-op $n8 $n0 orient down

$ns duplex-link-op $n0 $n13 orient down

$ns duplex-link-op $n6 $n13 orient right-
down

$ns duplex-link-op $n8 $n9 orient right

$ns duplex-link-op $n0 $n1 orient right

$ns duplex-link-op $n9 $n10 orient right

$ns duplex-link-op $n1 $n14 orient down

$ns duplex-link-op $n14 $n15 orient right

$ns duplex-link-op $n15 $n30 orient down

$ns duplex-link-op $n14 $n30 orient right-
down

$ns duplex-link-op $n10 $n31 orient up

$ns duplex-link-op $n10 $n3 orient down

$ns duplex-link-op $n10 $n11 orient right

$ns duplex-link-op $n31 $n11 orient right-
down

$ns duplex-link-op $n3 $n4 orient right

$ns duplex-link-op $n3 $n15 orient down
```

```
$ns duplex-link-op $n15 $n16 orient right

$ns duplex-link-op $n11 $n4 orient down

$ns duplex-link-op $n4 $n16 orient down

$ns duplex-link-op $n11 $n12 orient right

$ns duplex-link-op $n4 $n5 orient right

$ns duplex-link-op $n12 $n2 orient up

$ns duplex-link-op $n5 $n17 orient down

$ns duplex-link-op $n12 $n7 orient right-
down

$ns duplex-link-op $n17 $n7 orient right-up

$ns duplex-link-op $n17 $n29 orient right

$ns duplex-link-op $n7 $n18 orient right

$ns duplex-link-op $n29 $n23 orient right

$ns duplex-link-op $n18 $n19 orient up

$ns duplex-link-op $n18 $n23 orient down

$ns duplex-link-op $n23 $n24 orient right

$ns duplex-link-op $n24 $n28 orient down

$ns duplex-link-op $n23 $n28 orient right-
down

$ns duplex-link-op $n19 $n20 orient right

$ns duplex-link-op $n19 $n26 orient right-
down

$ns duplex-link-op $n24 $n25 orient right

$ns duplex-link-op $n26 $n25 orient right-
down
```

```
$ns duplex-link-op $n20 $n21 orient right

$ns duplex-link-op $n21 $n22 orient down

$ns duplex-link-op $n22 $n25 orient down

$ns duplex-link-op $n21 $n27 orient right

$ns duplex-link-op $n22 $n27 orient right-up

$ns rtproto DV

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n9 $n10

$ns rtmodel-at 4.5 up $n9 $n10

$ns rtmodel-at 8.0 down $n7 $n18

$ns rtmodel-at 8.3 up $n7 $n18

$ns rtmodel-at 15.0 down $n20 $n21

$ns rtmodel-at 15.3 up $n20 $n21

# ----CREATING AND ATTACHING AGENTS----

set tcp [new Agent/TCP]

$tcp set class_ 1

$ns attach-agent $n6 $tcp

set ftp [new Application/FTP]

$ftp set type_ FTP

$ftp set packet_size_ 3000

$ftp attach-agent $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n27 $sink
```

```
$ns connect $tcp $sink

$tcp set fid_ 1

$ns color 1 Blue

$ns at 0.0 "$n6 color chocolate"

$ns at 0.0 "$n27 color chocolate"

$ns at 0.3 "$ftp start"

$ns at 20 "finish"

$ns run
```

DVUDP30.tcl:

```
set ns [new Simulator]

set nr [open DVUDP30.tr w]

$ns trace-all $nr

set nf [open DVUDP30.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----

proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam DVUDP30.nam &

  exit 0
```

```
}
 # ----CREATION OF NODES----
for {set i 0} {$i < 32} {incr i} {
Set n$i [$ns node]
}
# ----CREATION OF LINKS----
$ns duplex-link $n6 $n8 14Mb 12ms DropTail
$ns duplex-link $n8 $n0 15Mb 16ms DropTail
$ns duplex-link $n0 $n13 16Mb 15ms DropTail
$ns duplex-link $n6 $n13 20Mb 17ms DropTail
$ns duplex-link $n8 $n9 12Mb 19ms DropTail
$ns duplex-link $n0 $n1 10Mb 20ms DropTail
$ns duplex-link $n9 $n10 5Mb 10ms DropTail
$ns duplex-link $n1 $n14 13Mb 12ms DropTail
$ns duplex-link $n14 $n15 15Mb 17ms DropTail
$ns duplex-link $n15 $n30 14Mb 14ms DropTail
$ns duplex-link $n14 $n30 10Mb 18ms DropTail
$ns duplex-link $n10 $n31 19Mb 13ms DropTail
$ns duplex-link $n10 $n3 6Mb 10ms DropTail
$ns duplex-link $n10 $n11 17Mb 15ms DropTail
$ns duplex-link $n31 $n11 10Mb 15ms DropTail
$ns duplex-link $n3 $n4 7Mb 16ms DropTail
$ns duplex-link $n3 $n15 6Mb 17ms DropTail
```

```
$ns duplex-link $n15 $n16 7Mb 10ms DropTail

$ns duplex-link $n11 $n4 18Mb 14ms DropTail

$ns duplex-link $n4 $n16 8Mb 17ms DropTail

$ns duplex-link $n11 $n12 15Mb 17ms DropTail

$ns duplex-link $n4 $n5 12Mb 18ms DropTail

$ns duplex-link $n12 $n2 7Mb 11ms DropTail

$ns duplex-link $n5 $n17 11Mb 15ms DropTail

$ns duplex-link $n12 $n7 12Mb 15ms DropTail

$ns duplex-link $n17 $n7 8Mb 16ms DropTail

$ns duplex-link $n17 $n29 16Mb 20ms DropTail

$ns duplex-link $n7 $n18 9Mb 15ms DropTail

$ns duplex-link $n29 $n23 18Mb 11ms DropTail

$ns duplex-link $n18 $n19 17Mb 16ms DropTail

$ns duplex-link $n18 $n23 14Mb 15ms DropTail

$ns duplex-link $n23 $n24 14Mb 14ms DropTail

$ns duplex-link $n24 $n28 15Mb 14ms DropTail

$ns duplex-link $n23 $n28 5Mb 13ms DropTail

$ns duplex-link $n19 $n20 19Mb 20ms DropTail

$ns duplex-link $n19 $n26 11Mb 15ms DropTail

$ns duplex-link $n24 $n25 8Mb 18ms DropTail

$ns duplex-link $n26 $n25 13Mb 16ms DropTail

$ns duplex-link $n20 $n21 12Mb 16ms DropTail

$ns duplex-link $n21 $n22 9Mb 20ms DropTail
```

```
$ns duplex-link $n22 $n25 13Mb 10ms DropTail

$ns duplex-link $n21 $n27 7Mb 19ms DropTail

$ns duplex-link $n22 $n27 6Mb 19ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n6 $n8 orient right-up

$ns duplex-link-op $n8 $n0 orient down

$ns duplex-link-op $n0 $n13 orient down

$ns duplex-link-op $n6 $n13 orient right-down

$ns duplex-link-op $n8 $n9 orient right

$ns duplex-link-op $n0 $n1 orient right

$ns duplex-link-op $n9 $n10 orient right

$ns duplex-link-op $n1 $n14 orient down

$ns duplex-link-op $n14 $n15 orient right

$ns duplex-link-op $n15 $n30 orient down

$ns duplex-link-op $n14 $n30 orient right-down

$ns duplex-link-op $n10 $n31 orient up

$ns duplex-link-op $n10 $n3 orient down

$ns duplex-link-op $n10 $n11 orient right

$ns duplex-link-op $n31 $n11 orient right-down

$ns duplex-link-op $n3 $n4 orient right

$ns duplex-link-op $n3 $n15 orient down

$ns duplex-link-op $n15 $n16 orient right

$ns duplex-link-op $n11 $n4 orient down
```

```
$ns duplex-link-op $n4 $n16 orient down

$ns duplex-link-op $n11 $n12 orient right

$ns duplex-link-op $n4 $n5 orient right

$ns duplex-link-op $n12 $n2 orient up

$ns duplex-link-op $n5 $n17 orient down

$ns duplex-link-op $n12 $n7 orient right-down

$ns duplex-link-op $n17 $n7 orient right-up

$ns duplex-link-op $n17 $n29 orient right

$ns duplex-link-op $n7 $n18 orient right

$ns duplex-link-op $n29 $n23 orient right

$ns duplex-link-op $n18 $n19 orient up

$ns duplex-link-op $n18 $n23 orient down

$ns duplex-link-op $n23 $n24 orient right

$ns duplex-link-op $n24 $n28 orient down

$ns duplex-link-op $n23 $n28 orient right-down

$ns duplex-link-op $n19 $n20 orient right

$ns duplex-link-op $n19 $n26 orient right-down

$ns duplex-link-op $n24 $n25 orient right

$ns duplex-link-op $n26 $n25 orient right-down

$ns duplex-link-op $n20 $n21 orient right

$ns duplex-link-op $n21 $n22 orient down

$ns duplex-link-op $n22 $n25 orient down

$ns duplex-link-op $n21 $n27 orient right
```

```
$ns duplex-link-op $n22 $n27 orient right-up

$ns rtproto DV

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n9 $n10

$ns rtmodel-at 4.5 up $n9 $n10

$ns rtmodel-at 8.0 down $n7 $n18

$ns rtmodel-at 8.3 up $n7 $n18

$ns rtmodel-at 15.0 down $n20 $n21

$ns rtmodel-at 15.3 up $n20 $n21

# ----CREATING AND ATTACHING AGENTS----

set udp [new Agent/UDP]

$udp set class_ 1

$ns attach-agent $n6 $udp

set cbr [new Application/Traffic/CBR]

$cbr set type_ CBR

$cbr set packet_size_ 3000

$cbr attach-agent $udp

set null [new Agent/Null]

$ns attach-agent $n27 $null

$ns connect $udp $null

$udp set fid_ 1

$ns color 1 Red

$ns at 0.0 "$n6 color chocolate"
```

```
$ns at 0.0 "$n27 color chocolate"

$ns at 0.3 "$cbr start"

$ns at 20 "finish"

$ns run
```

LSRTCP30.tcl:

```
set ns [new Simulator]

set nr [open LSRTCP30.tr w]

$ns trace-all $nr

set nf [open LSRTCP30.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----

proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam LSRTCP30.nam &

  exit 0

}

 # ----CREATION OF NODES----

for {set i 0} {$i < 32} {incr i} {

Set n$i [$ns node]

}
```

```
# ----CREATION OF LINKS----
$ns duplex-link $n6 $n8 14Mb 12ms DropTail
$ns duplex-link $n8 $n0 15Mb 16ms DropTail
$ns duplex-link $n0 $n13 16Mb 15ms DropTail
$ns duplex-link $n6 $n13 20Mb 17ms DropTail
$ns duplex-link $n8 $n9 12Mb 19ms DropTail
$ns duplex-link $n0 $n1 10Mb 20ms DropTail
$ns duplex-link $n9 $n10 5Mb 10ms DropTail
$ns duplex-link $n1 $n14 13Mb 12ms DropTail
$ns duplex-link $n14 $n15 15Mb 17ms DropTail
$ns duplex-link $n15 $n30 14Mb 14ms DropTail
$ns duplex-link $n14 $n30 10Mb 18ms DropTail
$ns duplex-link $n10 $n31 19Mb 13ms DropTail
$ns duplex-link $n10 $n3 6Mb 10ms DropTail
$ns duplex-link $n10 $n11 17Mb 15ms DropTail
$ns duplex-link $n31 $n11 10Mb 15ms DropTail
$ns duplex-link $n3 $n4 7Mb 16ms DropTail
$ns duplex-link $n3 $n15 6Mb 17ms DropTail
$ns duplex-link $n15 $n16 7Mb 10ms DropTail
$ns duplex-link $n11 $n4 18Mb 14ms DropTail
$ns duplex-link $n4 $n16 8Mb 17ms DropTail
$ns duplex-link $n11 $n12 15Mb 17ms DropTail
$ns duplex-link $n4 $n5 12Mb 18ms DropTail
```

```
$ns duplex-link $n12 $n2 7Mb 11ms DropTail

$ns duplex-link $n5 $n17 11Mb 15ms DropTail

$ns duplex-link $n12 $n7 12Mb 15ms DropTail

$ns duplex-link $n17 $n7 8Mb 16ms DropTail

$ns duplex-link $n17 $n29 16Mb 20ms DropTail

$ns duplex-link $n7 $n18 9Mb 15ms DropTail

$ns duplex-link $n29 $n23 18Mb 11ms DropTail

$ns duplex-link $n18 $n19 17Mb 16ms DropTail

$ns duplex-link $n18 $n23 14Mb 15ms DropTail

$ns duplex-link $n23 $n24 14Mb 14ms DropTail

$ns duplex-link $n24 $n28 15Mb 14ms DropTail

$ns duplex-link $n23 $n28 5Mb 13ms DropTail

$ns duplex-link $n19 $n20 19Mb 20ms DropTail

$ns duplex-link $n19 $n26 11Mb 15ms DropTail

$ns duplex-link $n24 $n25 8Mb 18ms DropTail

$ns duplex-link $n26 $n25 13Mb 16ms DropTail

$ns duplex-link $n20 $n21 12Mb 16ms DropTail

$ns duplex-link $n21 $n22 9Mb 20ms DropTail

$ns duplex-link $n22 $n25 13Mb 10ms DropTail

$ns duplex-link $n21 $n27 7Mb 19ms DropTail

$ns duplex-link $n22 $n27 6Mb 19ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n6 $n8 orient right-up
```

```
$ns duplex-link-op $n8 $n0 orient down

$ns duplex-link-op $n0 $n13 orient down

$ns duplex-link-op $n6 $n13 orient right-down

$ns duplex-link-op $n8 $n9 orient right

$ns duplex-link-op $n0 $n1 orient right

$ns duplex-link-op $n9 $n10 orient right

$ns duplex-link-op $n1 $n14 orient down

$ns duplex-link-op $n14 $n15 orient right

$ns duplex-link-op $n15 $n30 orient down

$ns duplex-link-op $n14 $n30 orient right-down

$ns duplex-link-op $n10 $n31 orient up

$ns duplex-link-op $n10 $n3 orient down

$ns duplex-link-op $n10 $n11 orient right

$ns duplex-link-op $n31 $n11 orient right-down

$ns duplex-link-op $n3 $n4 orient right

$ns duplex-link-op $n3 $n15 orient down

$ns duplex-link-op $n15 $n16 orient right

$ns duplex-link-op $n11 $n4 orient down

$ns duplex-link-op $n4 $n16 orient down

$ns duplex-link-op $n11 $n12 orient right

$ns duplex-link-op $n4 $n5 orient right

$ns duplex-link-op $n12 $n2 orient up

$ns duplex-link-op $n5 $n17 orient down
```

```
$ns duplex-link-op $n12 $n7 orient right-down

$ns duplex-link-op $n17 $n7 orient right-up

$ns duplex-link-op $n17 $n29 orient right

$ns duplex-link-op $n7 $n18 orient right

$ns duplex-link-op $n29 $n23 orient right

$ns duplex-link-op $n18 $n19 orient up

$ns duplex-link-op $n18 $n23 orient down

$ns duplex-link-op $n23 $n24 orient right

$ns duplex-link-op $n24 $n28 orient down

$ns duplex-link-op $n23 $n28 orient right-down

$ns duplex-link-op $n19 $n20 orient right

$ns duplex-link-op $n19 $n26 orient right-down

$ns duplex-link-op $n24 $n25 orient right

$ns duplex-link-op $n26 $n25 orient right-down

$ns duplex-link-op $n20 $n21 orient right

$ns duplex-link-op $n21 $n22 orient down

$ns duplex-link-op $n22 $n25 orient down

$ns duplex-link-op $n21 $n27 orient right

$ns duplex-link-op $n22 $n27 orient right-up

$ns rtproto LS

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n9 $n10

$ns rtmodel-at 4.5 up $n9 $n10
```

```
$ns rtmodel-at 8.0 down $n7 $n18

$ns rtmodel-at 8.3 up $n7 $n18

$ns rtmodel-at 15.0 down $n20 $n21

$ns rtmodel-at 15.3 up $n20 $n21

# ----CREATING AND ATTACHING AGENTS----

set tcp [new Agent/TCP]

$tcp set class_ 1

$ns attach-agent $n6 $tcp

set ftp [new Application/FTP]

$ftp set type_ FTP

$ftp set packet_size_ 3000

$ftp attach-agent $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n27 $sink

$ns connect $tcp $sink

$tcp set fid_ 1

$ns color 1 Blue


$ns at 0.0 "$n6 color chocolate"

$ns at 0.0 "$n27 color chocolate"

$ns at 0.3 "$ftp start"

$ns at 20 "finish"

$ns run
```

LSRUDP30.tcl:

```
set ns [new Simulator]

set nr [open LSRUDP30.tr w]

$ns trace-all $nr

set nf [open LSRUDP30.nam w]

$ns namtrace-all $nf

# ----FINISH PROCEDURE----

proc finish {} {

  global ns nr nf

  $ns flush-trace

  close $nf

  close $nr

  exec nam LSRUDP30.nam &

  exit 0

}

 # ----CREATION OF NODES----

for {set i 0} {$i < 32} {incr i} {

Set n$i [$ns node]

}

# ----CREATION OF LINKS----

$ns duplex-link $n6 $n8 14Mb 12ms DropTail

$ns duplex-link $n8 $n0 15Mb 16ms DropTail

$ns duplex-link $n0 $n13 16Mb 15ms DropTail
```

```
$ns duplex-link $n6 $n13 20Mb 17ms DropTail

$ns duplex-link $n8 $n9 12Mb 19ms DropTail

$ns duplex-link $n0 $n1 10Mb 20ms DropTail

$ns duplex-link $n9 $n10 5Mb 10ms DropTail

$ns duplex-link $n1 $n14 13Mb 12ms DropTail

$ns duplex-link $n14 $n15 15Mb 17ms DropTail

$ns duplex-link $n15 $n30 14Mb 14ms DropTail

$ns duplex-link $n14 $n30 10Mb 18ms DropTail

$ns duplex-link $n10 $n31 19Mb 13ms DropTail

$ns duplex-link $n10 $n3 6Mb 10ms DropTail

$ns duplex-link $n10 $n11 17Mb 15ms DropTail

$ns duplex-link $n31 $n11 10Mb 15ms DropTail

$ns duplex-link $n3 $n4 7Mb 16ms DropTail

$ns duplex-link $n3 $n15 6Mb 17ms DropTail

$ns duplex-link $n15 $n16 7Mb 10ms DropTail

$ns duplex-link $n11 $n4 18Mb 14ms DropTail

$ns duplex-link $n4 $n16 8Mb 17ms DropTail

$ns duplex-link $n11 $n12 15Mb 17ms DropTail

$ns duplex-link $n4 $n5 12Mb 18ms DropTail

$ns duplex-link $n12 $n2 7Mb 11ms DropTail

$ns duplex-link $n5 $n17 11Mb 15ms DropTail

$ns duplex-link $n12 $n7 12Mb 15ms DropTail

$ns duplex-link $n17 $n7 8Mb 16ms DropTail
```

```
$ns duplex-link $n17 $n29 16Mb 20ms DropTail

$ns duplex-link $n7 $n18 9Mb 15ms DropTail

$ns duplex-link $n29 $n23 18Mb 11ms DropTail

$ns duplex-link $n18 $n19 17Mb 16ms DropTail

$ns duplex-link $n18 $n23 14Mb 15ms DropTail

$ns duplex-link $n23 $n24 14Mb 14ms DropTail

$ns duplex-link $n24 $n28 15Mb 14ms DropTail

$ns duplex-link $n23 $n28 5Mb 13ms DropTail

$ns duplex-link $n19 $n20 19Mb 20ms DropTail

$ns duplex-link $n19 $n26 11Mb 15ms DropTail

$ns duplex-link $n24 $n25 8Mb 18ms DropTail

$ns duplex-link $n26 $n25 13Mb 16ms DropTail

$ns duplex-link $n20 $n21 12Mb 16ms DropTail

$ns duplex-link $n21 $n22 9Mb 20ms DropTail

$ns duplex-link $n22 $n25 13Mb 10ms DropTail

$ns duplex-link $n21 $n27 7Mb 19ms DropTail

$ns duplex-link $n22 $n27 6Mb 19ms DropTail

# ----ORIENTATION OF LINKS----

$ns duplex-link-op $n6 $n8 orient right-up

$ns duplex-link-op $n8 $n0 orient down

$ns duplex-link-op $n0 $n13 orient down

$ns duplex-link-op $n6 $n13 orient right-down

$ns duplex-link-op $n8 $n9 orient right
```

```
$ns duplex-link-op $n0 $n1 orient right

$ns duplex-link-op $n9 $n10 orient right

$ns duplex-link-op $n1 $n14 orient down

$ns duplex-link-op $n14 $n15 orient right

$ns duplex-link-op $n15 $n30 orient down

$ns duplex-link-op $n14 $n30 orient right-down

$ns duplex-link-op $n10 $n31 orient up

$ns duplex-link-op $n10 $n3 orient down

$ns duplex-link-op $n10 $n11 orient right

$ns duplex-link-op $n31 $n11 orient right-down

$ns duplex-link-op $n3 $n4 orient right

$ns duplex-link-op $n3 $n15 orient down

$ns duplex-link-op $n15 $n16 orient right

$ns duplex-link-op $n11 $n4 orient down

$ns duplex-link-op $n4 $n16 orient down

$ns duplex-link-op $n11 $n12 orient right

$ns duplex-link-op $n4 $n5 orient right

$ns duplex-link-op $n12 $n2 orient up

$ns duplex-link-op $n5 $n17 orient down

$ns duplex-link-op $n12 $n7 orient right-down

$ns duplex-link-op $n17 $n7 orient right-up

$ns duplex-link-op $n17 $n29 orient right

$ns duplex-link-op $n7 $n18 orient right
```

```
$ns duplex-link-op $n29 $n23 orient right

$ns duplex-link-op $n18 $n19 orient up

$ns duplex-link-op $n18 $n23 orient down

$ns duplex-link-op $n23 $n24 orient right

$ns duplex-link-op $n24 $n28 orient down

$ns duplex-link-op $n23 $n28 orient right-down

$ns duplex-link-op $n19 $n20 orient right

$ns duplex-link-op $n19 $n26 orient right-down

$ns duplex-link-op $n24 $n25 orient right

$ns duplex-link-op $n26 $n25 orient right-down

$ns duplex-link-op $n20 $n21 orient right

$ns duplex-link-op $n21 $n22 orient down

$ns duplex-link-op $n22 $n25 orient down

$ns duplex-link-op $n21 $n27 orient right

$ns duplex-link-op $n22 $n27 orient right-up

$ns rtproto LS

# ----ALTERING LINKS----

$ns rtmodel-at 4.0 down $n9 $n10

$ns rtmodel-at 4.5 up $n9 $n10

$ns rtmodel-at 8.0 down $n7 $n18

$ns rtmodel-at 8.3 up $n7 $n18

$ns rtmodel-at 15.0 down $n20 $n21

$ns rtmodel-at 15.3 up $n20 $n21
```

```
# ----CREATING AND ATTACHING AGENTS----

set udp [new Agent/UDP]

$udp set class_ 1

$ns attach-agent $n6 $udp

set cbr [new Application/Traffic/CBR]

$cbr set type_ CBR

$cbr set packet_size_ 3000

$cbr attach-agent $udp

set null [new Agent/Null]

$ns attach-agent $n27 $null

$ns connect $udp $null

$udp set fid_ 1

$ns color 1 Red

$ns at 0.0 "$n6 color chocolate"

$ns at 0.0 "$n27 color chocolate"

$ns at 0.3 "$cbr start"

$ns at 20 "finish"

$ns run
```

results.awk:

```
BEGIN {

  start_Time=0

  send_count=0

  received_count=0
```

```
  end_Time=0
  received_data=0
  throughput=0
  pdr=0
 }
{
 if($1=="+" && $3=="0" && $5=="tcp")
  {
   send_count++
   if(flag==0){
    start_Time=$2;
    flag=1
   }
  }
 if($1=="r" && $4=="17" && $5=="tcp")
  {
   received_count++
   end_Time=$2
   received_data+=$6
  }
  if($1=="+" && $3=="0" && $5=="cbr")
  {
   send_count++
```

```
  if(flag==0){
   start_Time=$2;
   flag=1
  }
 }
if($1=="r" && $4=="17" && $5=="cbr")
 {
  received_count++
  end_Time=$2
  received_data+=$6
 }

 if($1=="+" && $3=="6" && $5=="tcp")
 {
  send_count++
  if(flag==0){
   start_Time=$2;
   flag=1
  }
 }
if($1=="r" && $4=="27" && $5=="tcp")
 {
  received_count++
```

```
   end_Time=$2
   received_data+=$6
  }
  if($1=="+" && $3=="6" && $5=="cbr")
  {
   send_count++
   if(flag==0){
    start_Time=$2;
    flag=1
   }
  }
 if($1=="r" && $4=="27" && $5=="cbr")
  {
   received_count++
   end_Time=$2
   received_data+=$6
  }
}

END {
  printf("Send Count : %d \n",send_count)
  printf("Received Count : %d \n",received_count)
  printf("Start time is %f sec \n ",start_Time)
```

```
 printf("End time is %f sec \n ",end_Time)
 printf("Received data is %f bytes \n " ,received_data)
 throughput=(received_data*8)/(end_Time-start_Time)
 printf("Throughput is %f bps \n " ,throughput)
 pdr=received_count/send_count
 printf("Packet Delivery Ratio is %f \n ",pdr)
 packetloss=send_count-received_count
 printf("Packet Loss is %d \n",packetloss)
}
```

CHAPTER 3

RESULTS

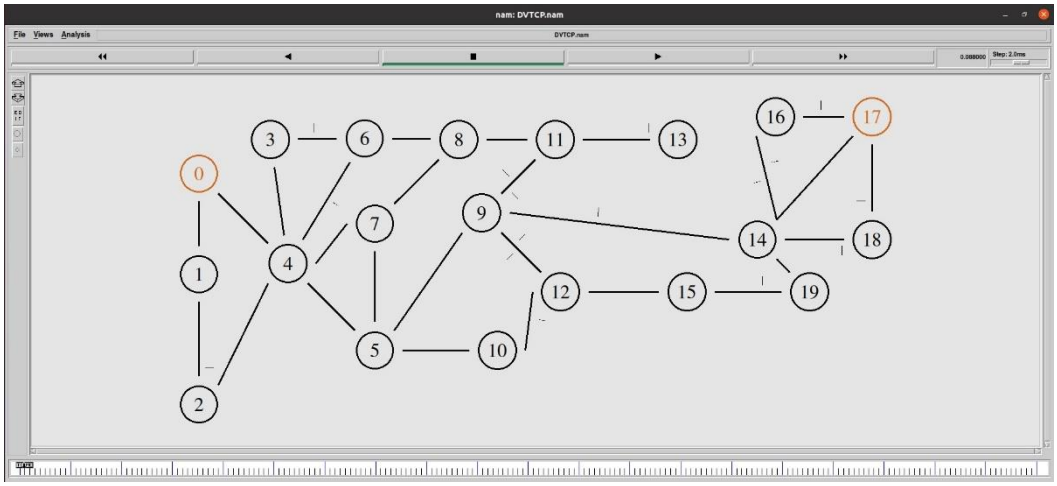

Figure-03: Simulation screen of 20 nodes DVTCP at the beginning.

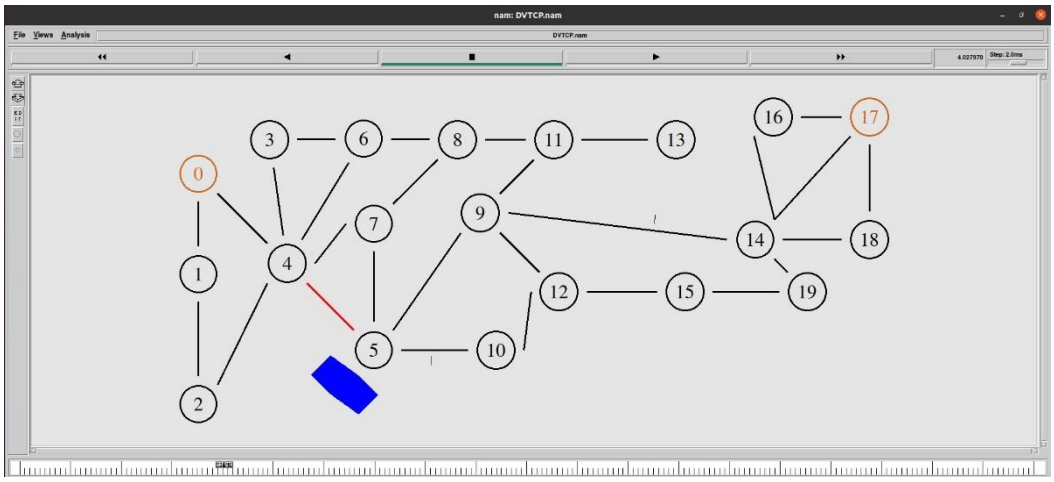

Figure-04: Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 4,5 is down.

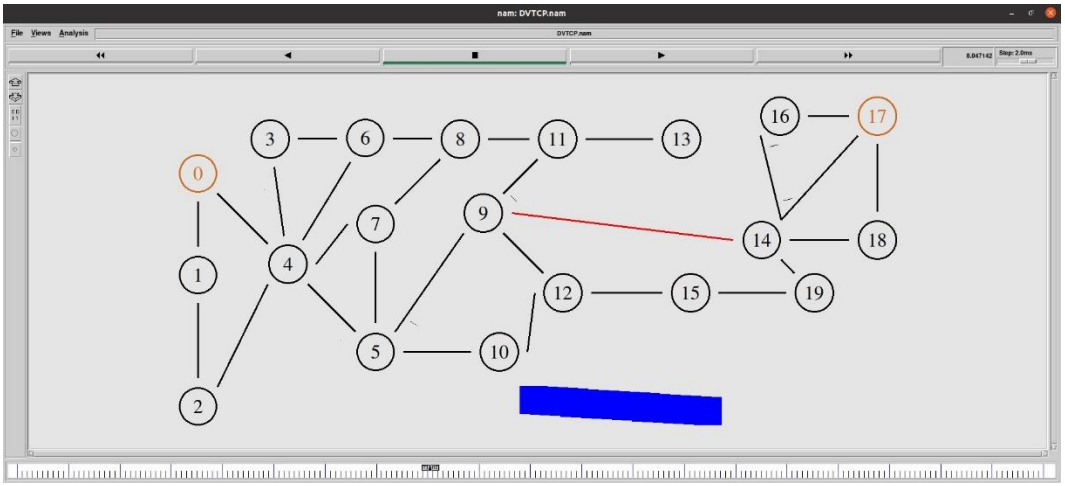

Figure-05: Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 9,14 is down.

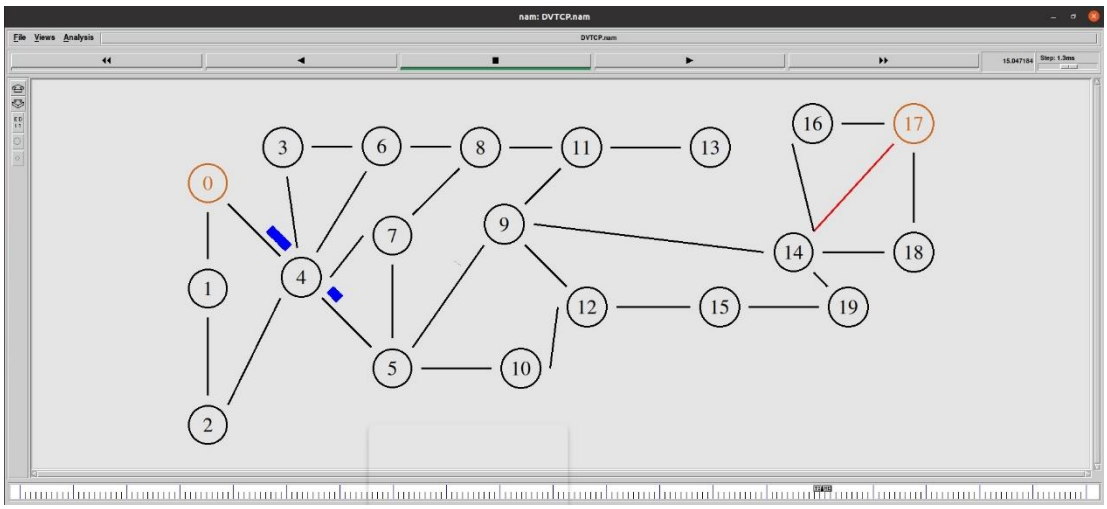

Figure-06: Simulation screen of 20 nodes DVTCP at intermediate time when link between nodes 14,17 is down.

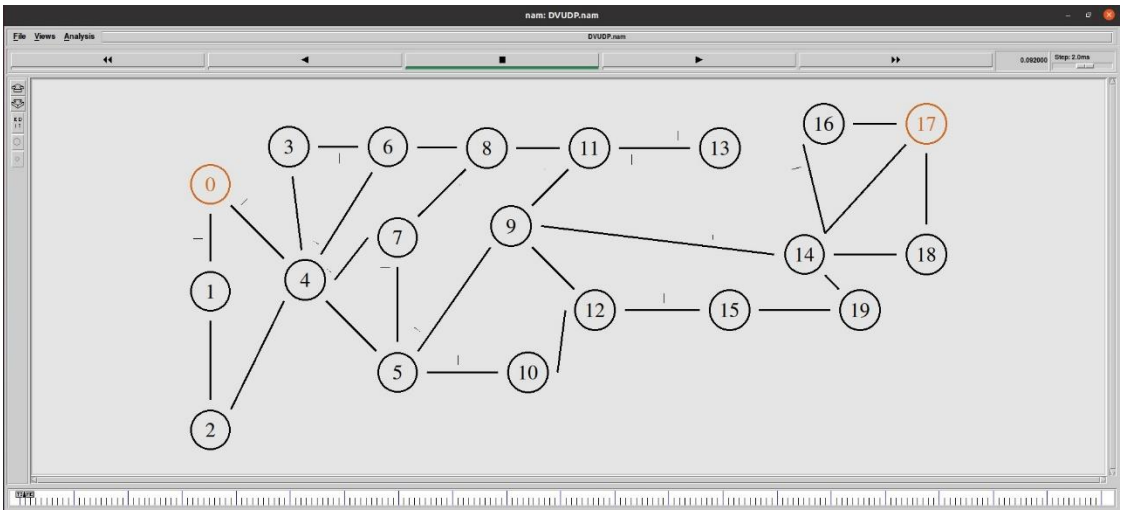

Figure-07: Simulation screen of 20 nodes DVUDP at beginning.

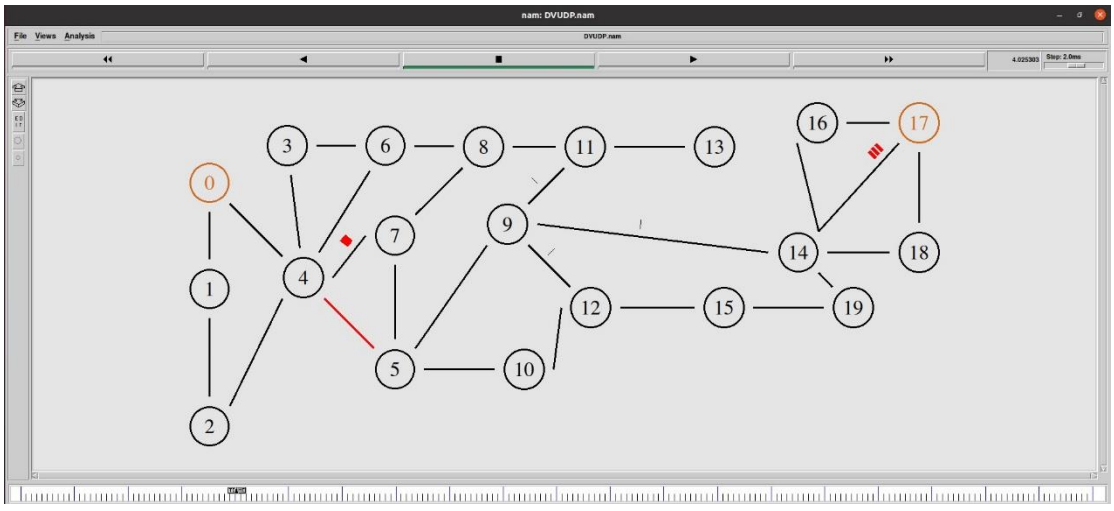

Figure-08: Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 4,5 is down.

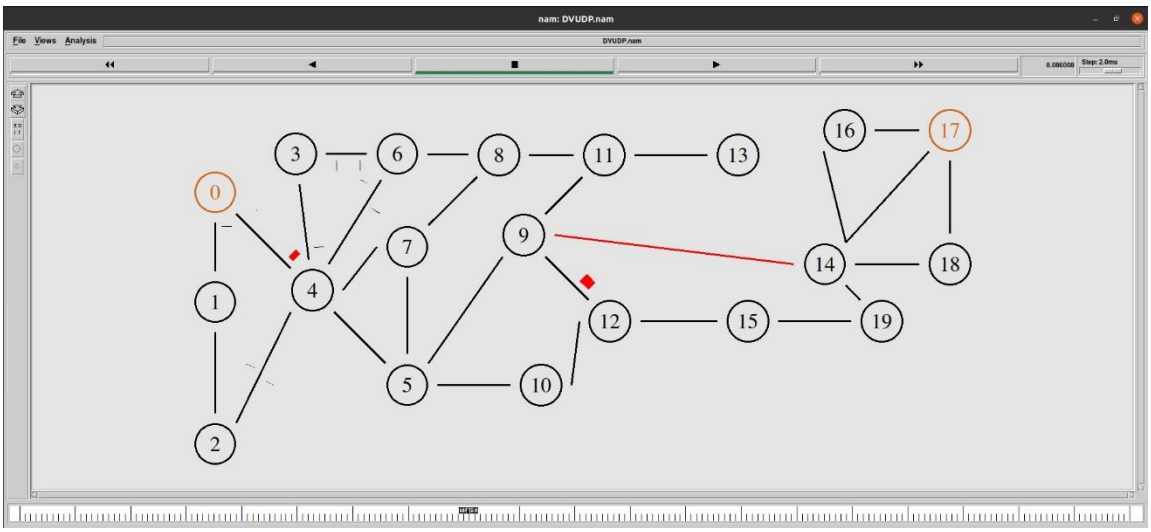

Figure-09: Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 9,14 is down.

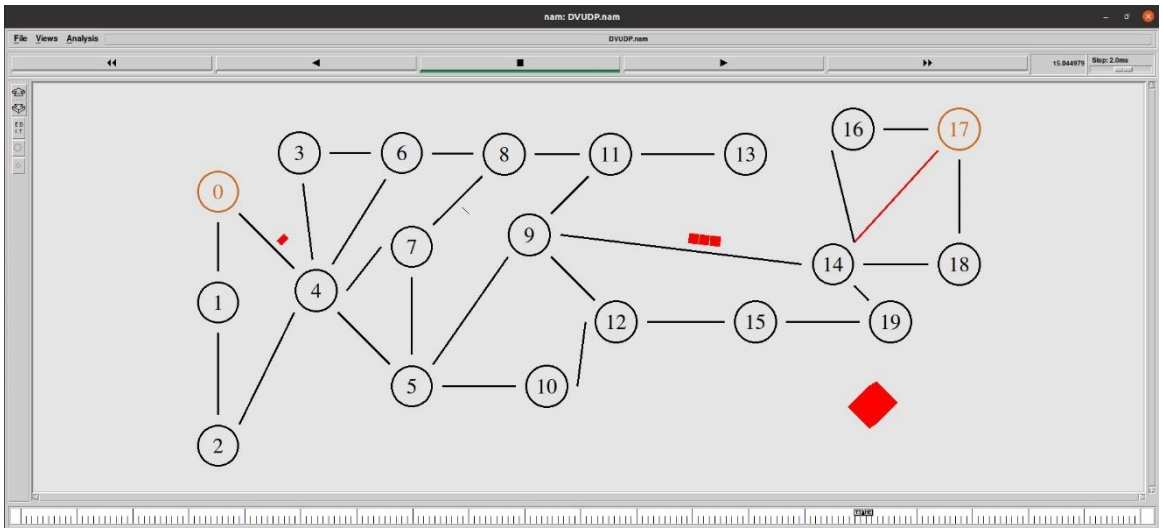

Figure-10: Simulation screen of 20 nodes DVUDP at intermediate time when link between nodes 14,17 is down.

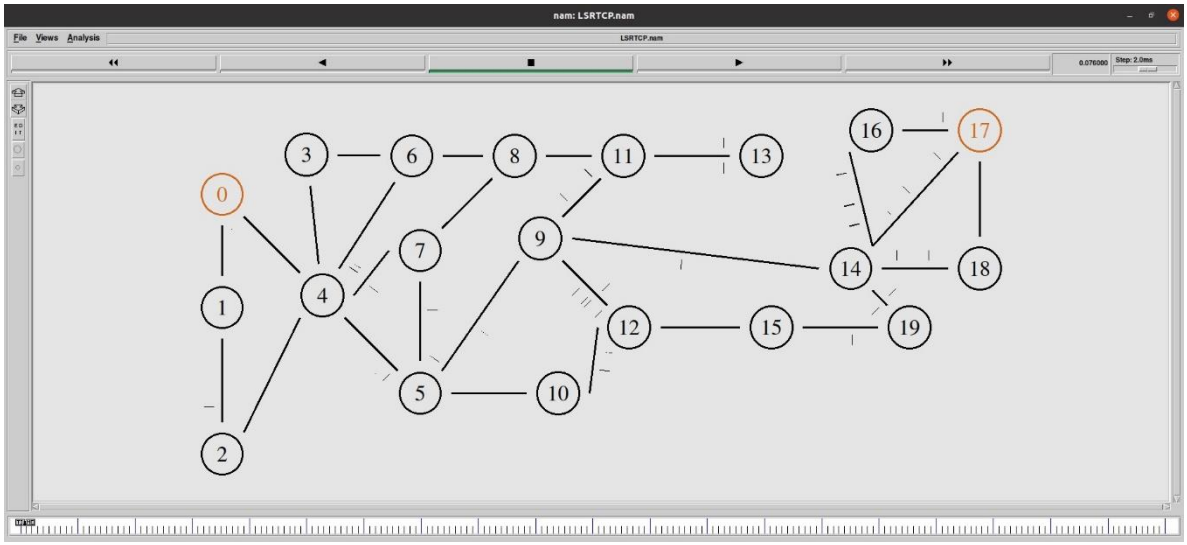

Figure-11: Simulation screen of 20 nodes LSRTCP at the beginning.

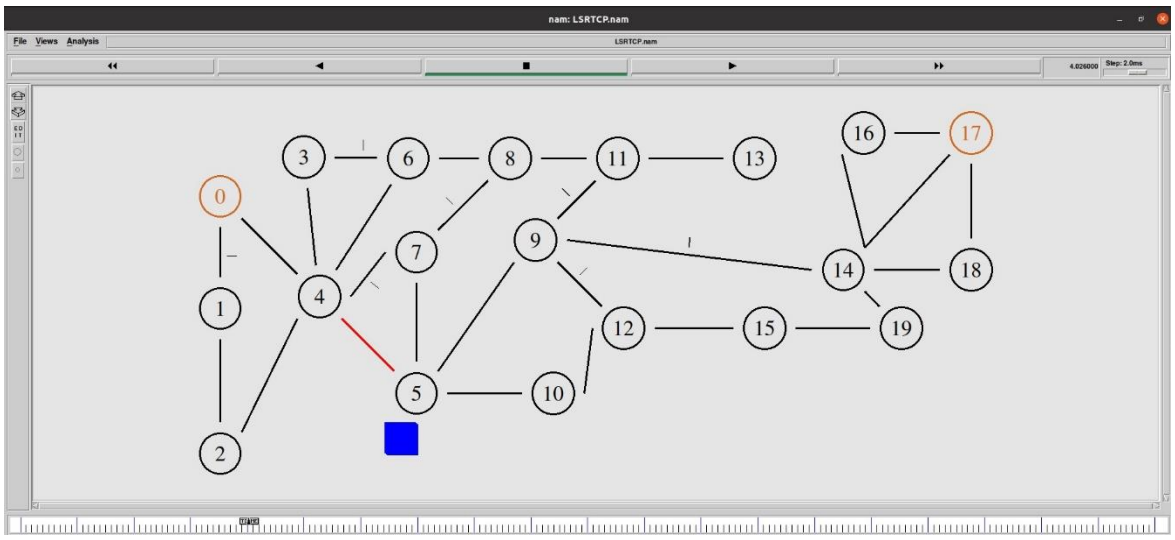

Figure-12: Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 4,5 is down.

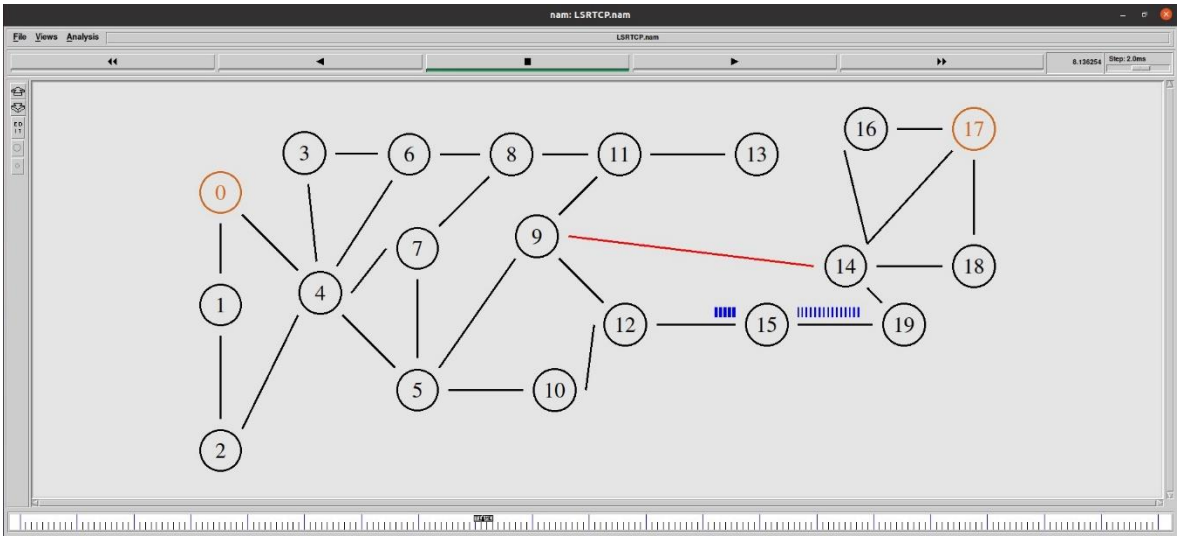

Figure-13: Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 9,14 is down.

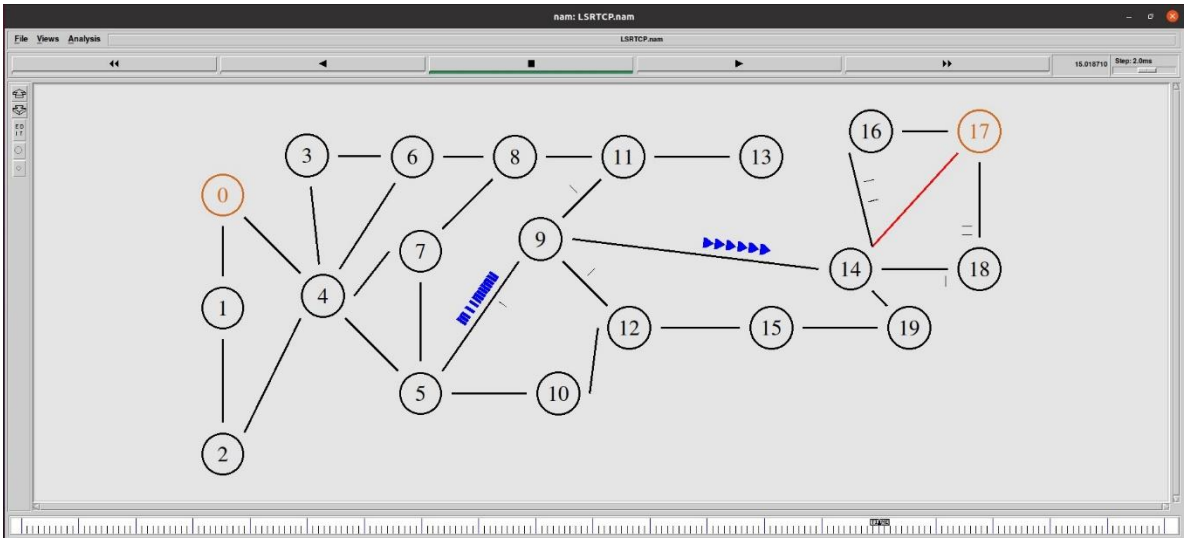

Figure-14: Simulation screen of 20 nodes LSRTCP at intermediate time when link between nodes 14,17 is down.

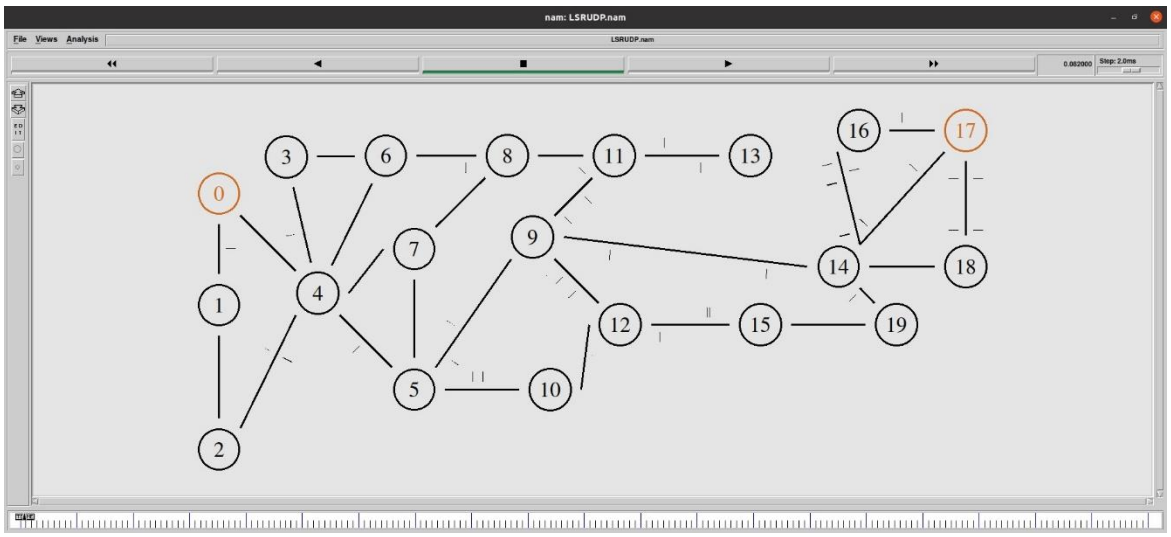

Figure-15: Simulation screen of 20 nodes LSRUDP at the beginning.

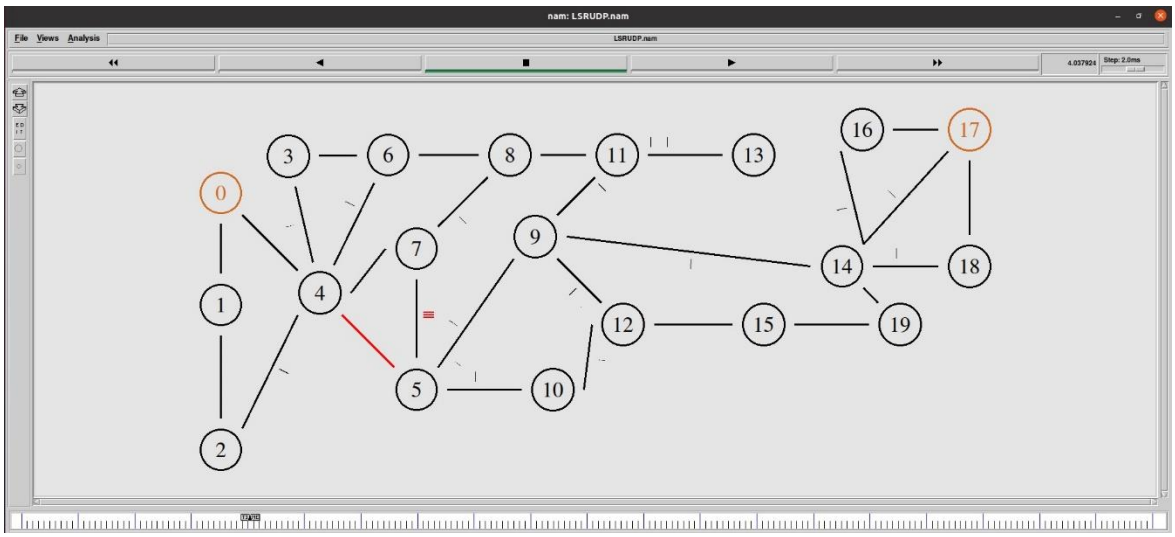

Figure-16: Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 4,5 is down.

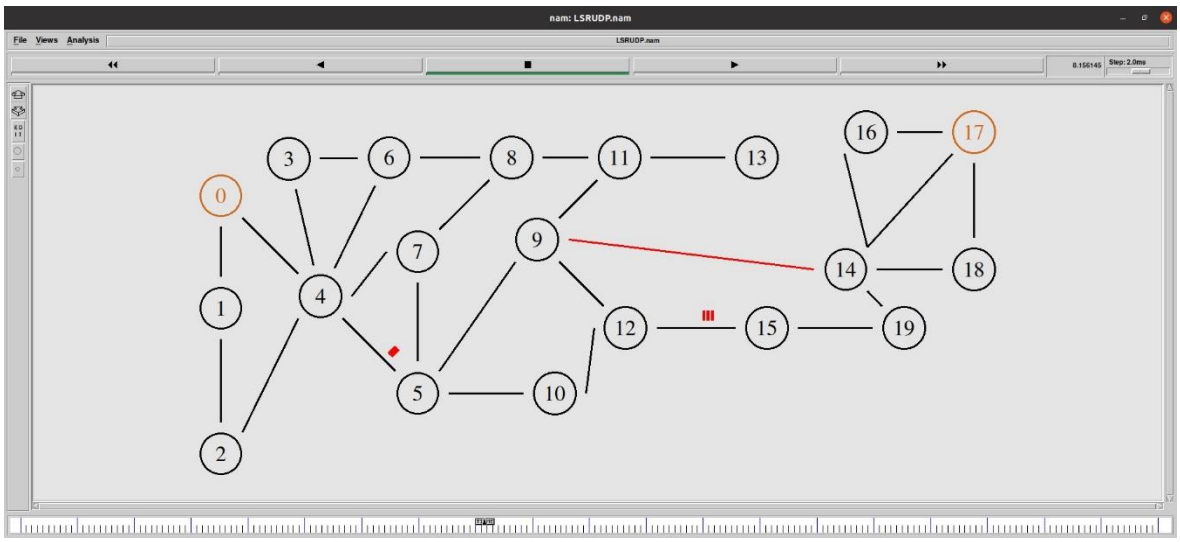

Figure-17: Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 9,14is down.

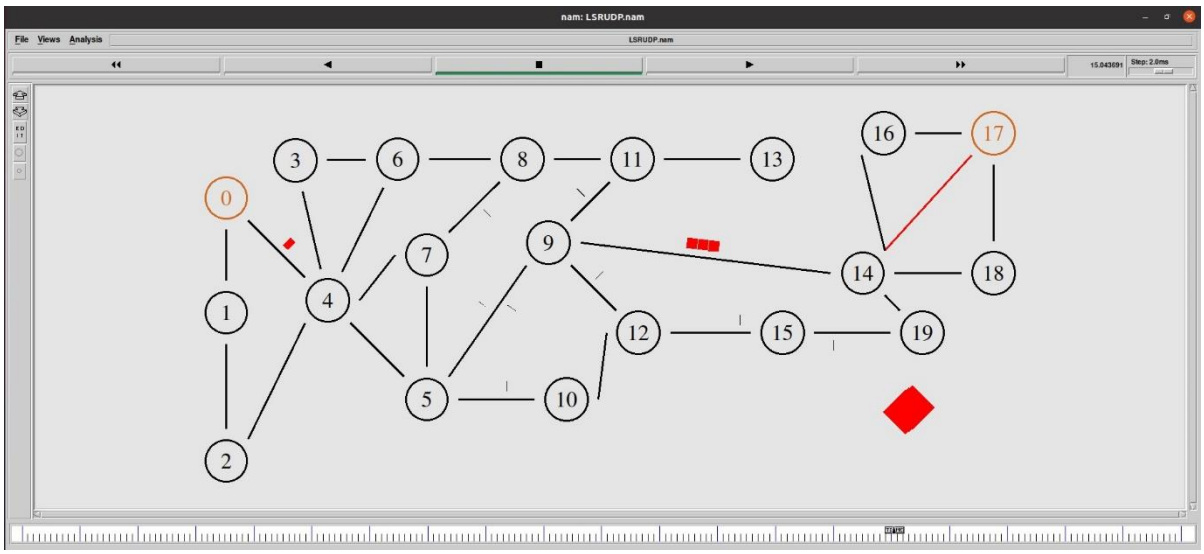

Figure-18: Simulation screen of 20 nodes LSRUDP at intermediate time when link between nodes 4,5 is down.

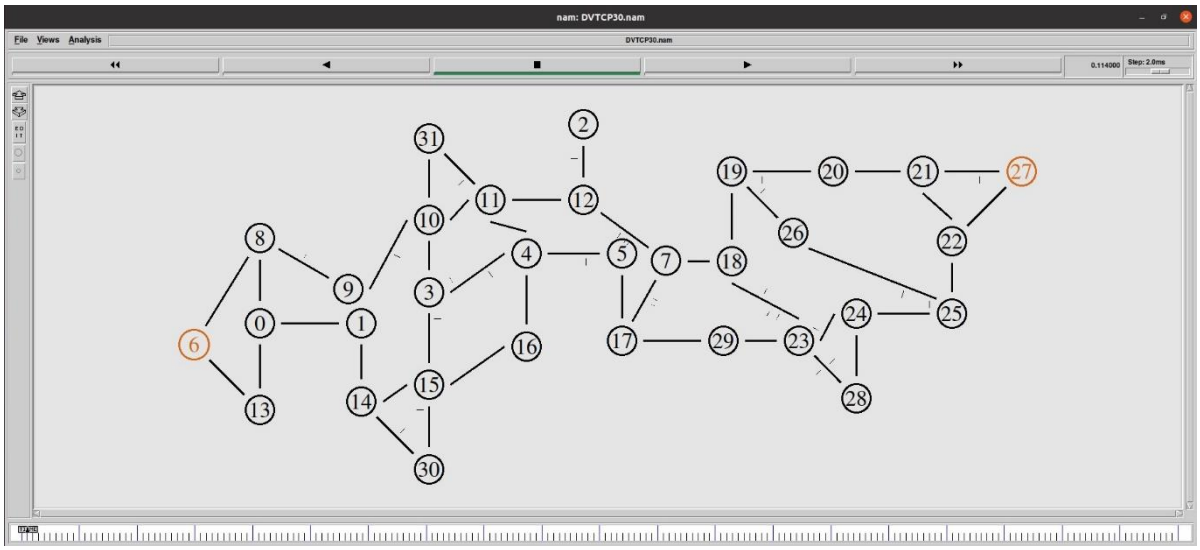

Figure-19: Simulation screen of 32 nodes DVTCP at the beginning

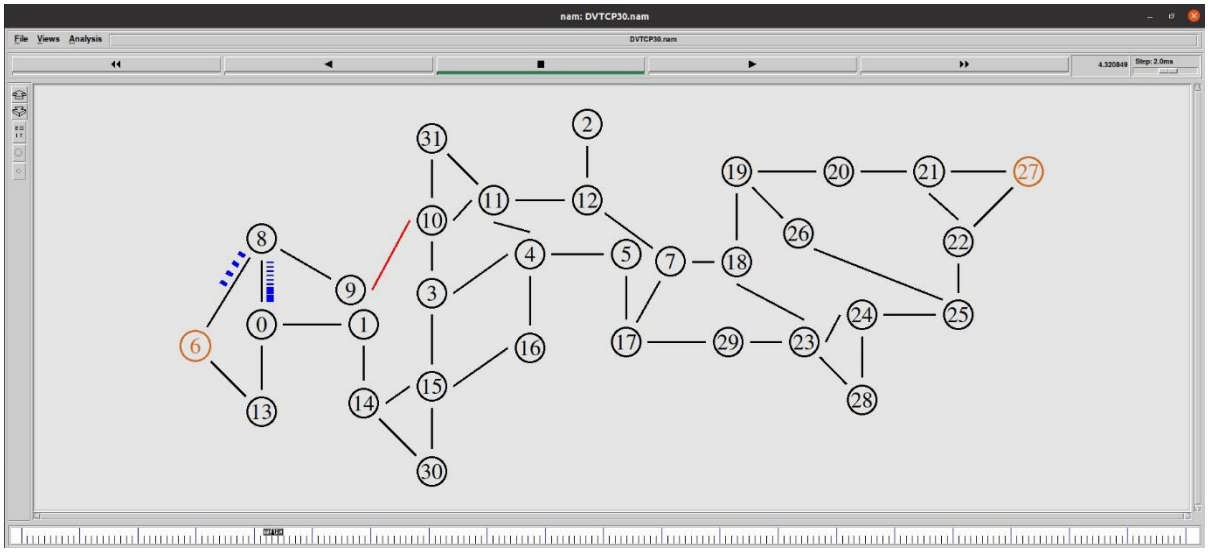

Figure-20: Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 9,10 is down.

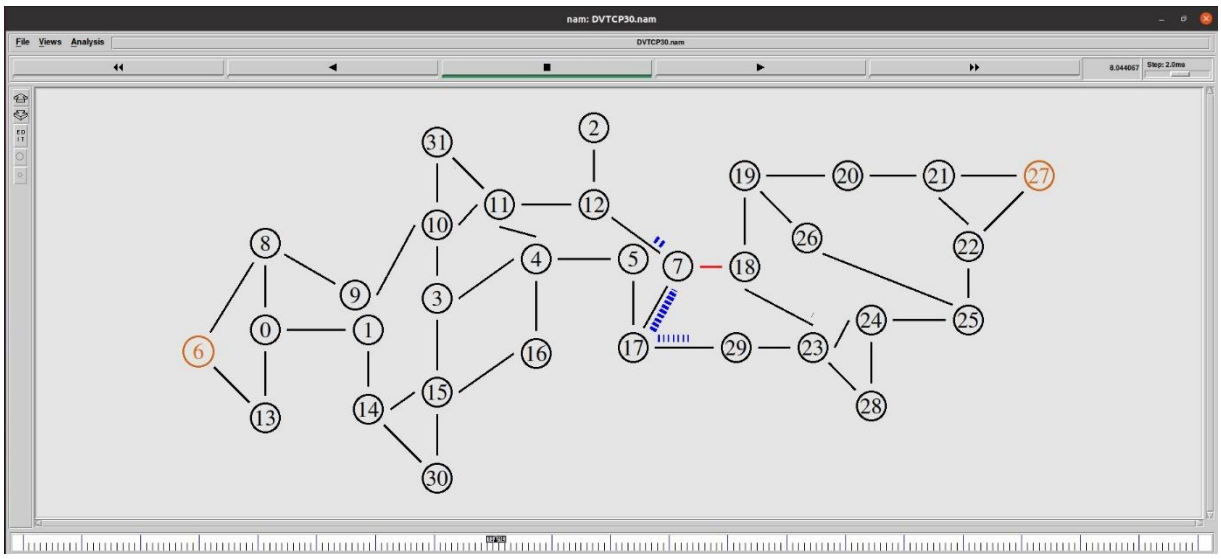

Figure-21: Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 7,18 is down.

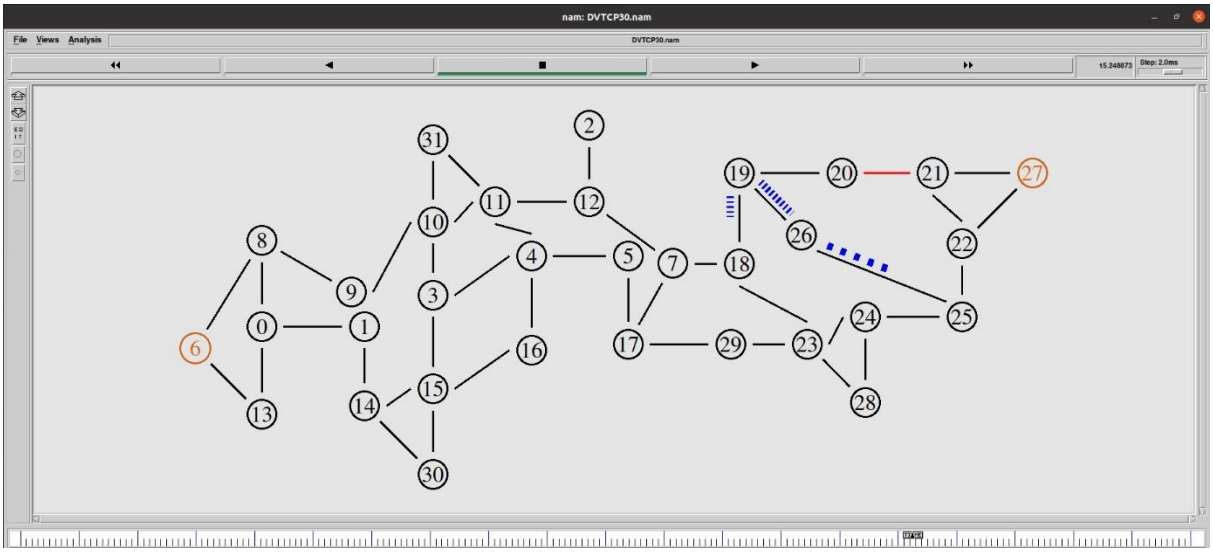

Figure-22: Simulation screen of 32 nodes DVTCP at intermediate time when link between nodes 20,21 is down.

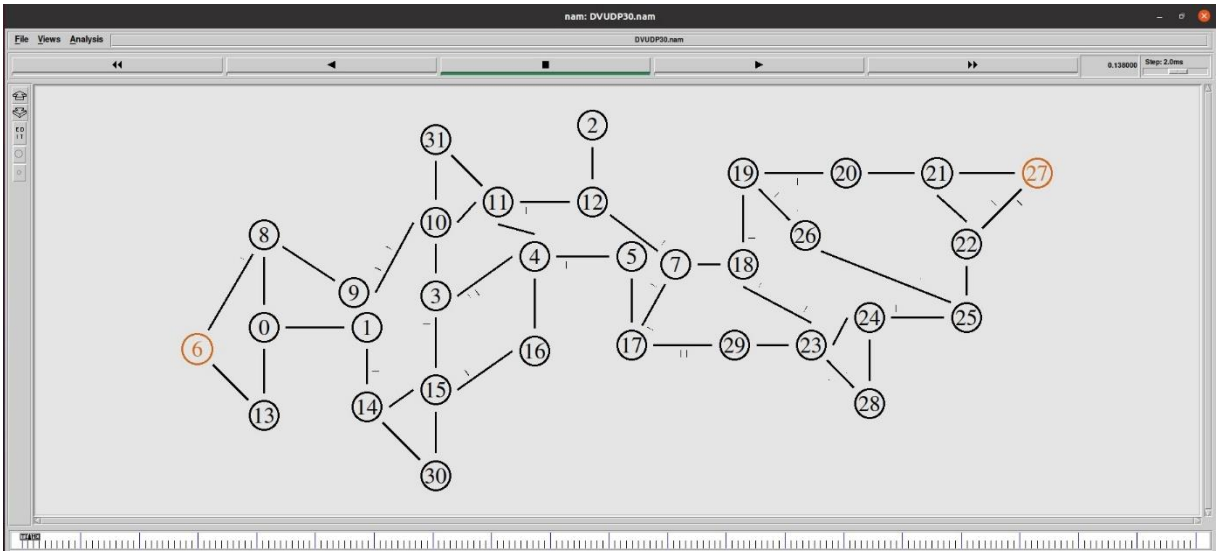

Figure-23: Simulation screen of 32 nodes DVUDP at the beginning.

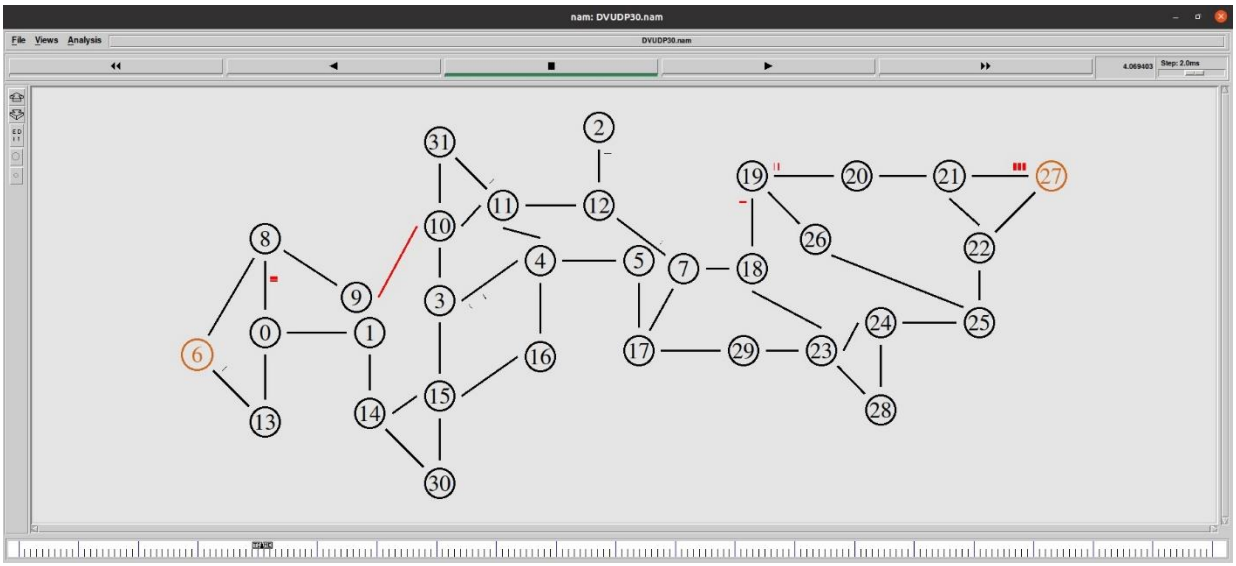

Figure-24: Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 9,10 is down.

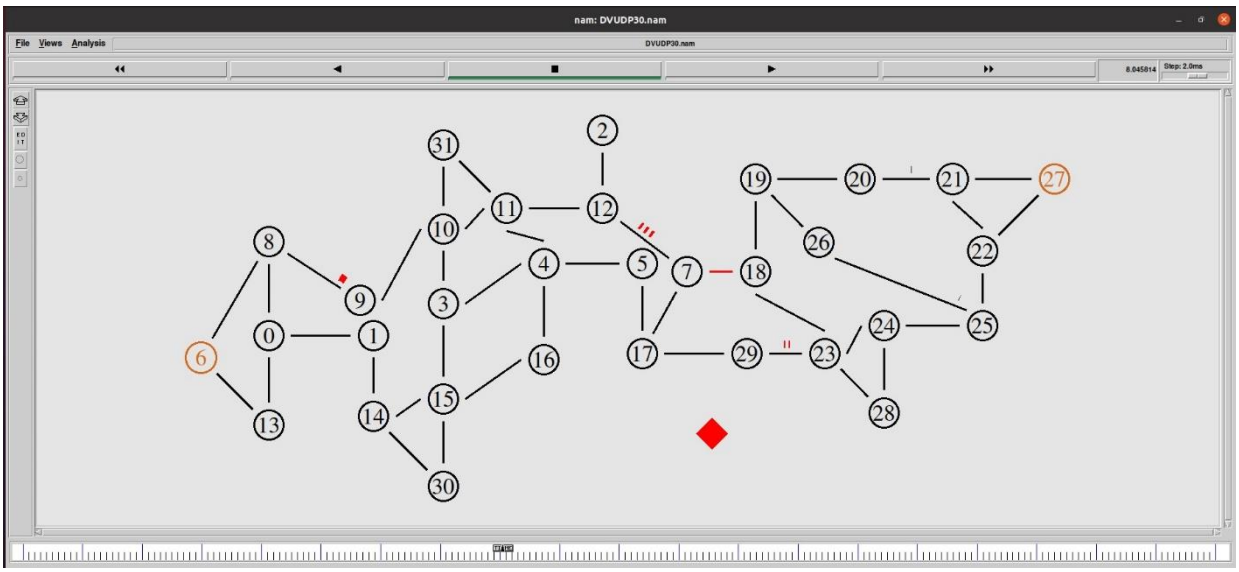

Figure-25: Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 7,18 is down.

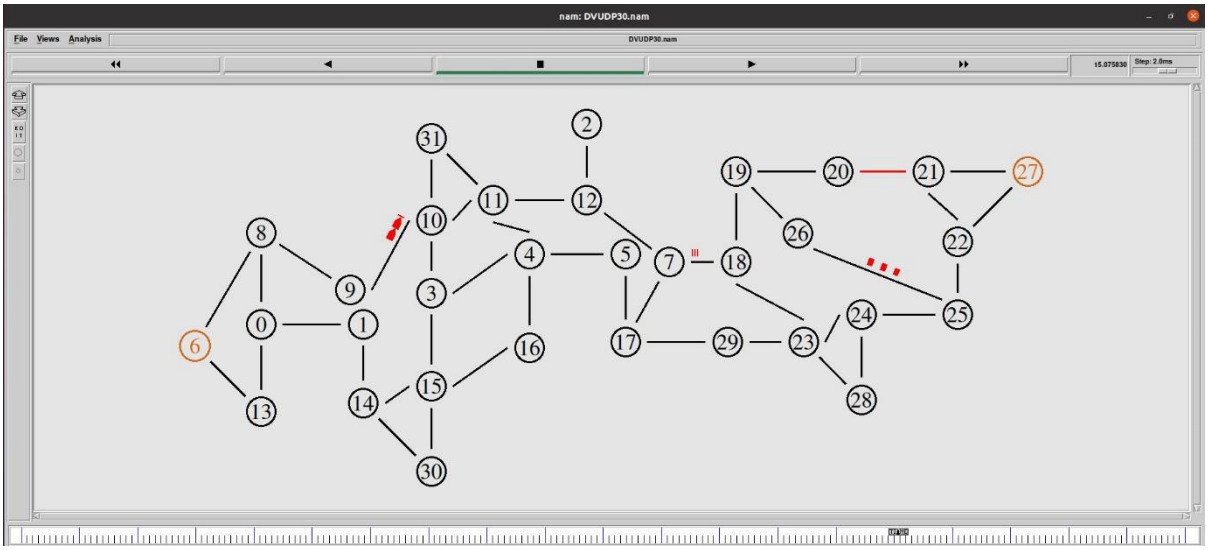

Figure-26: Simulation screen of 32 nodes DVUDP at intermediate time when link between nodes 20,21 is down.

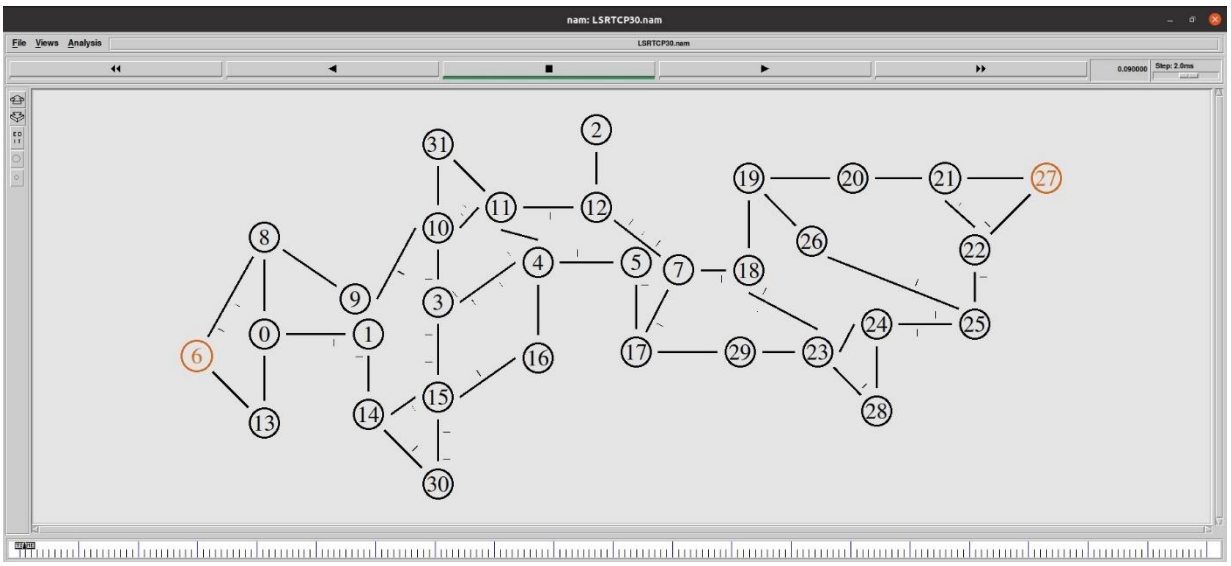

Figure-27: Simulation screen of 32 nodes LSRTCP at the beginning.

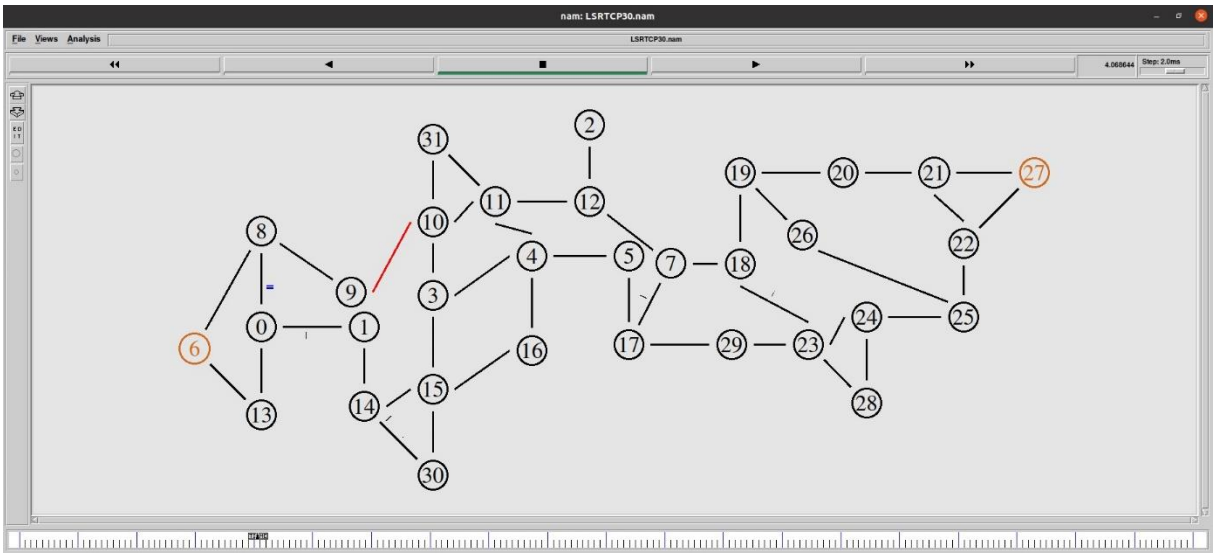

Figure-28: Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 9,10 is down.

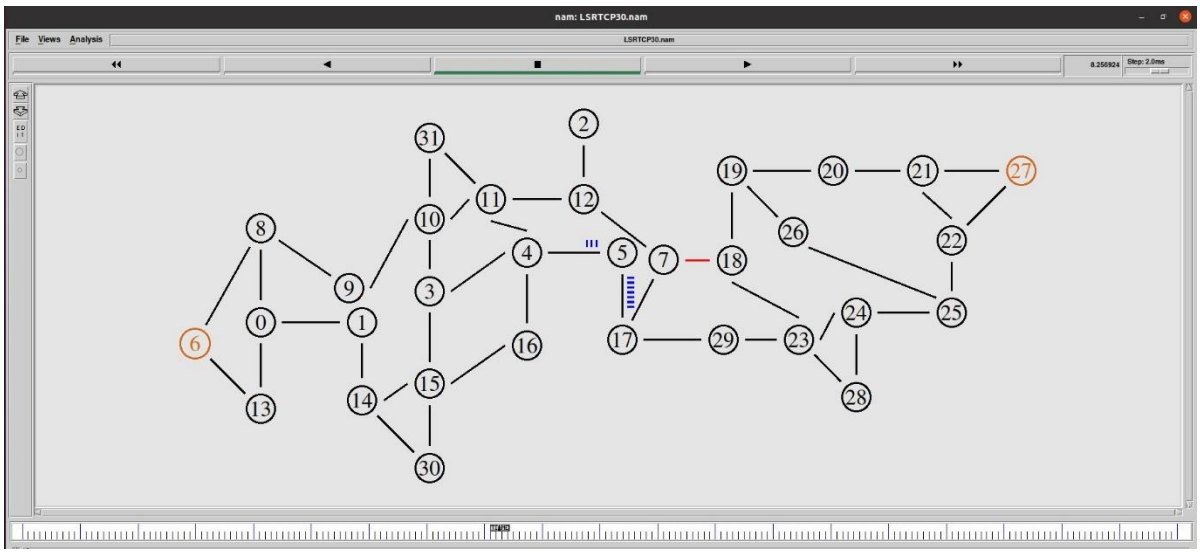

Figure-29: Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 7,18 is down.

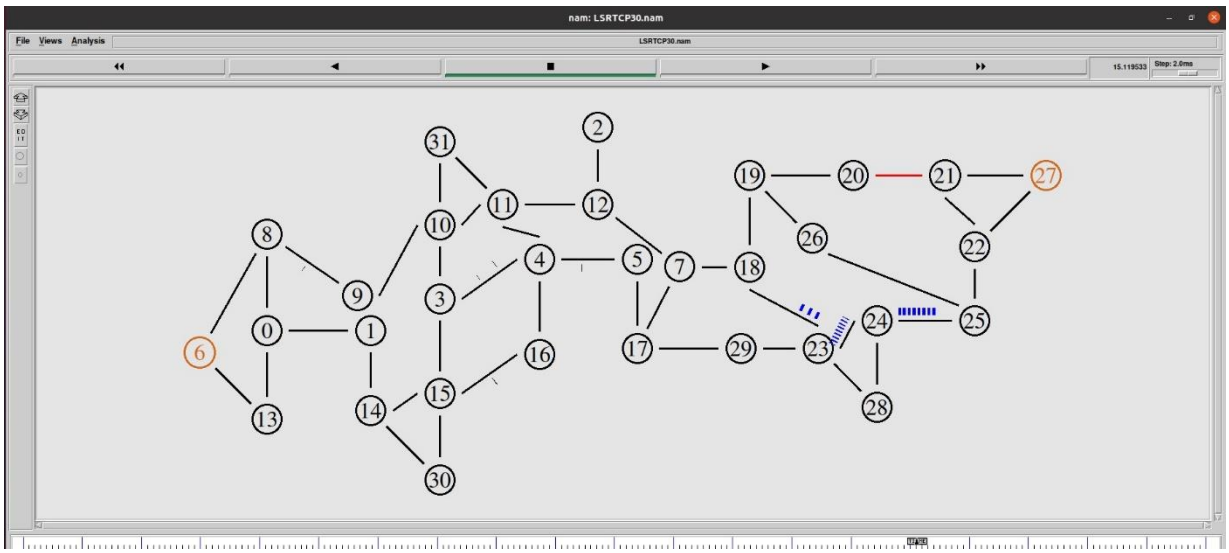

Figure-30: Simulation screen of 32 nodes LSRTCP at intermediate time when link between nodes 20,21 is down.

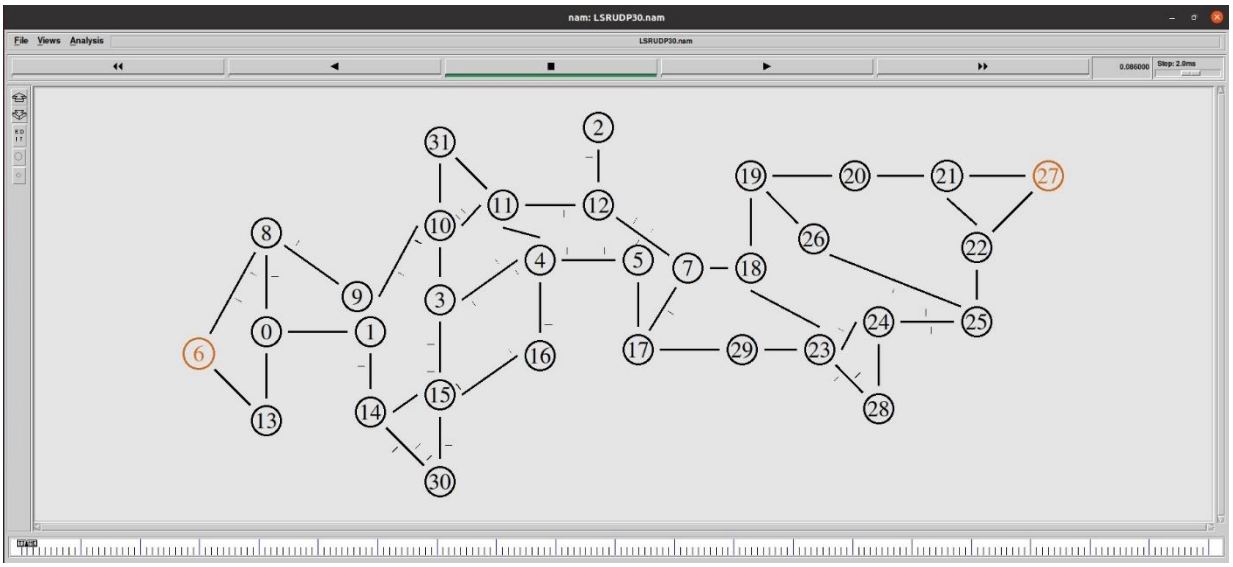

Figure-31: Simulation screen of 32 nodes LSRUDP at the beginning.

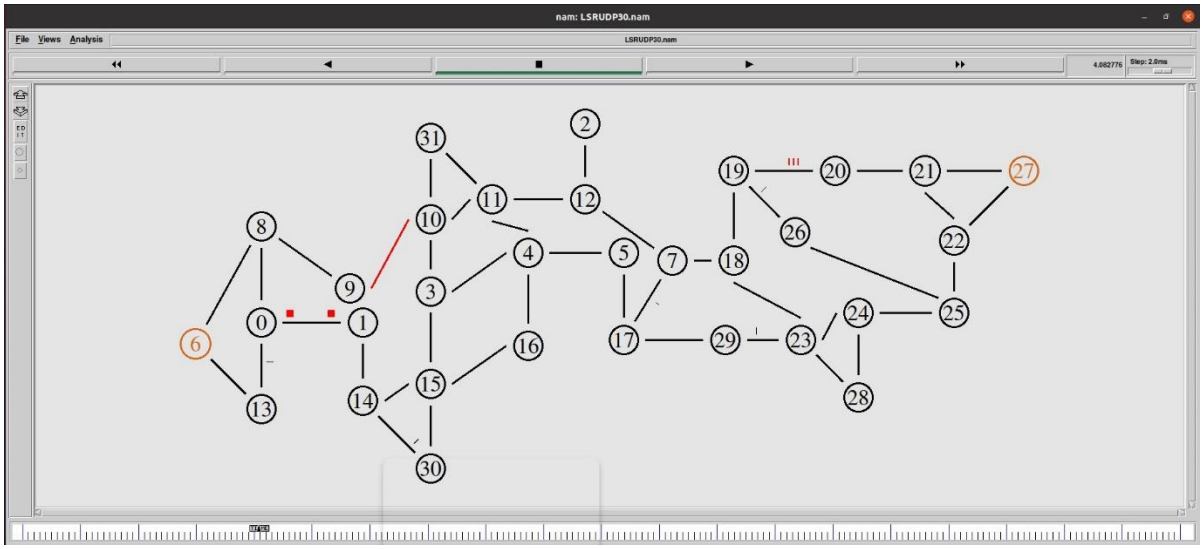

Figure-32: Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 9,10 is down.

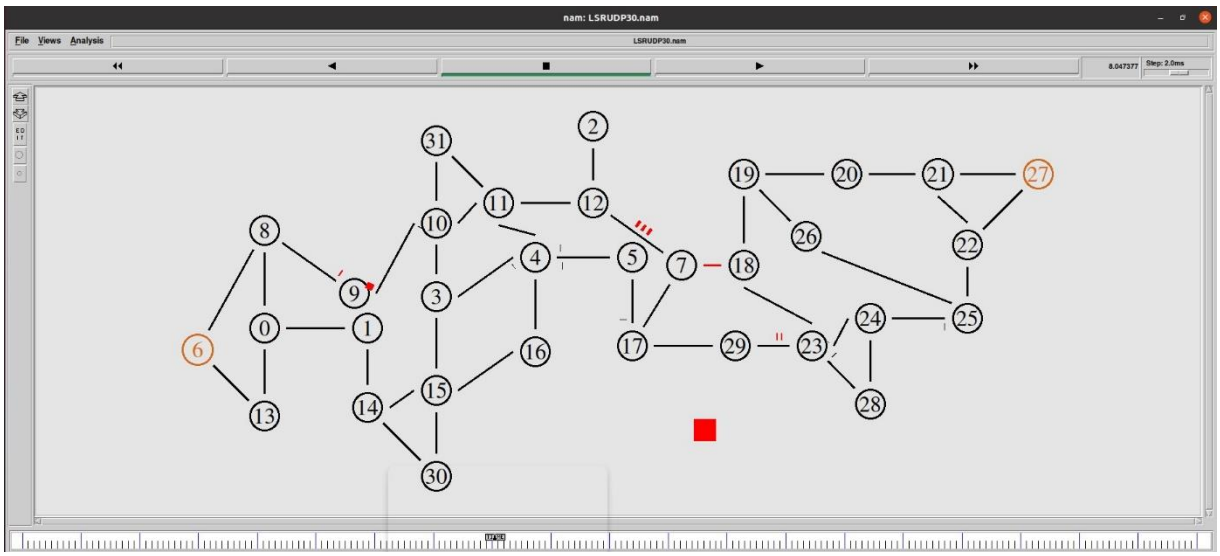

Figure-33: Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 7,18 is down.

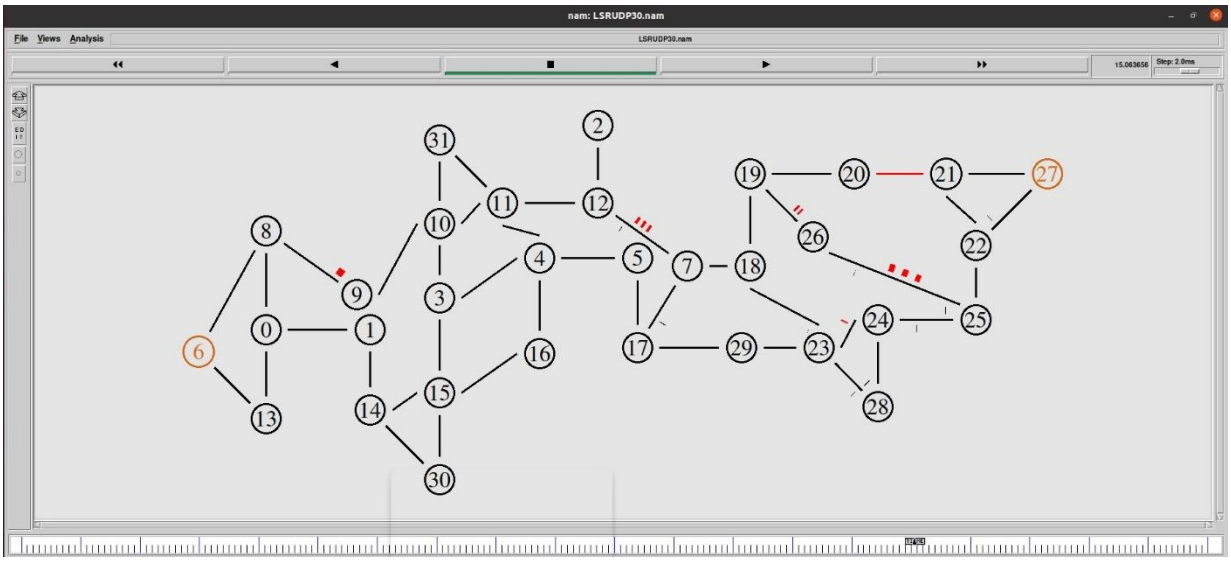

Figure-34: Simulation screen of 32 nodes LSRUDP at intermediate time when link between nodes 20,21 is down.

CHAPTER 4

PERFORMANCE EVALUATION

The Link-State and Distance vector routing protocols are simulated is NS2 simulator. In this we have analyzed two scenarios, in one scenario we have analyzed the parameters Throughput, Packet loss and Packet Delivery Ratio(PDR) by taking 20 nodes, and in the second scenario we have analyzed the same parameters by taking 32 nodes. In both scenarios we have simulated using both TCP and UDP i.e., FTP and CBR traffic.

FIRST SCENARIO:

Simulation done for 20 nodes.

	Start Time (in secs)	End Time (in secs)	Send Count	Received count	Data Received (in bytes)	Throughput (in bps)	Packet Delivery Ratio	Packet loss
DVTCP	0.300000	19.998289	2031	2011	2090440.000000	848983.381247	0.990153	20
DVUDP	0.300000	19.992647	1104	1095	1095000.000000	444836.085266	0.991848	09
LSRTCP	3.300000	19.951267	1615	1612	1675480.000000	804974.180043	0.998142	03
LSRUDP	0.460714	19.992647	1095	1089	1089000.000000	446038.802202	0.994521	06

Table-01 : Output of results of protocols simulated for 20 nodes on the execution of awk file.

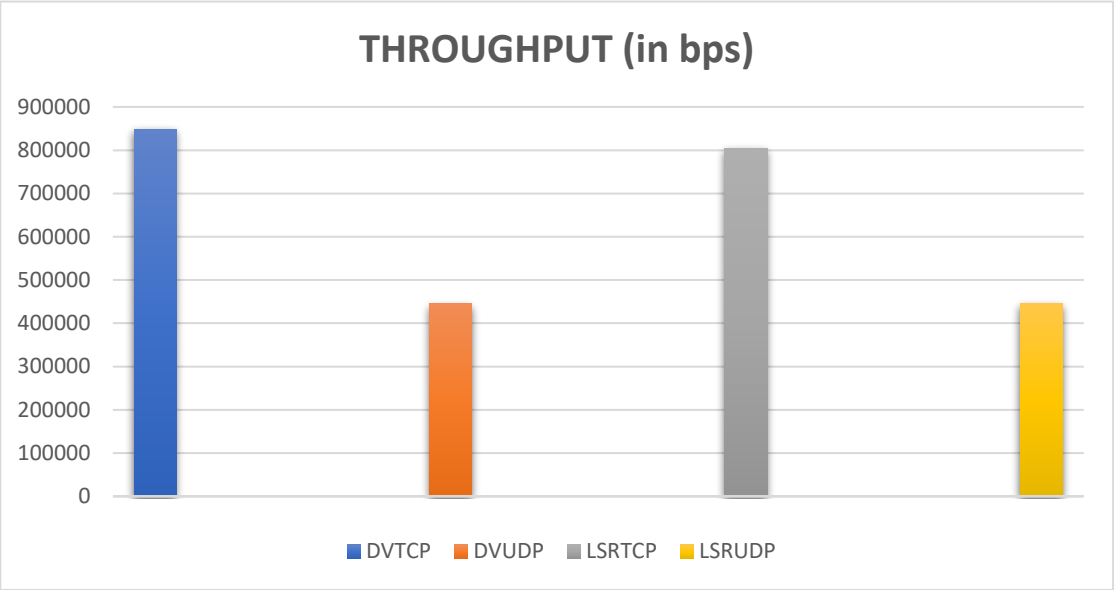

Figure-35: Throughput Analysis graph for the protocols simulated for 20 nodes.

In Figure-35, x-axis indicates the type of protocol used and y-axis indicates throughput in bps.

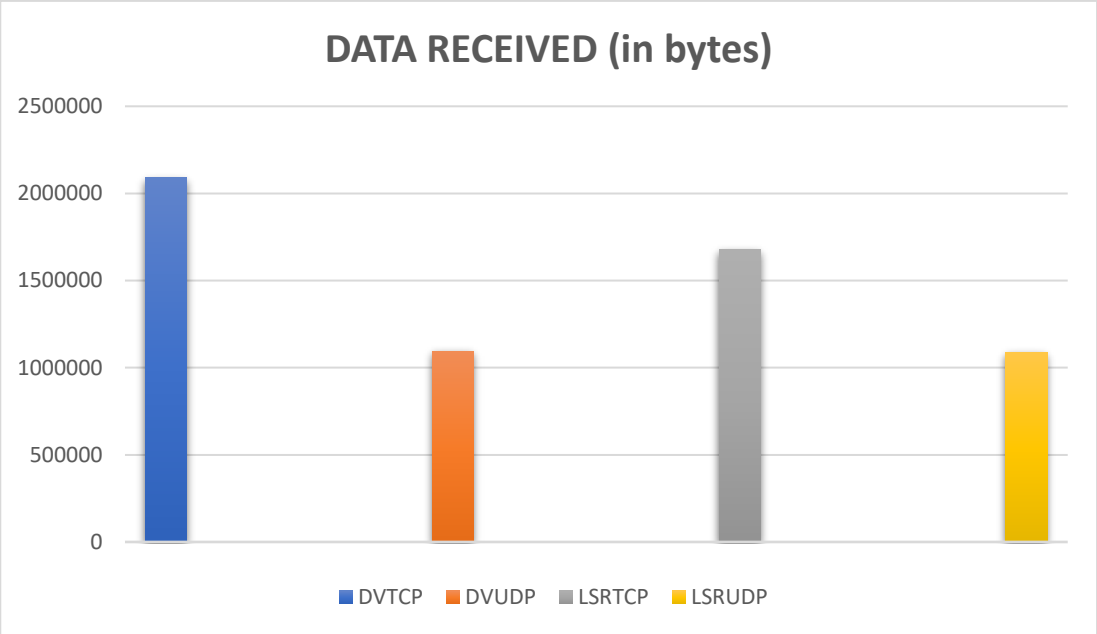

Figure-36: Data received Analysis graph for the protocols simulated for 20 nodes.

In Figure-36, x-axis indicates the type of protocol used and y-axis indicates Data Received in bytes.

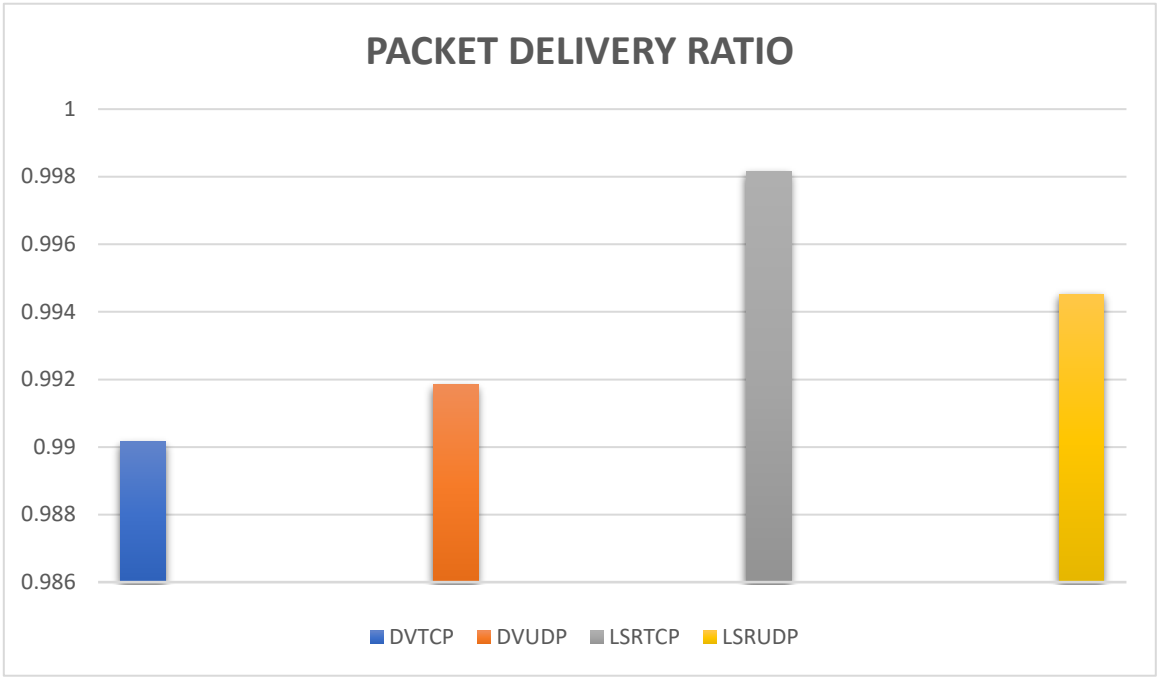

Figure-37: Packet Delivery Ratio Analysis graph for the protocols simulated for 20 nodes.

In Figure-37, x-axis indicates the type of protocol used and y-axis indicates Packet Delivery Ratio.

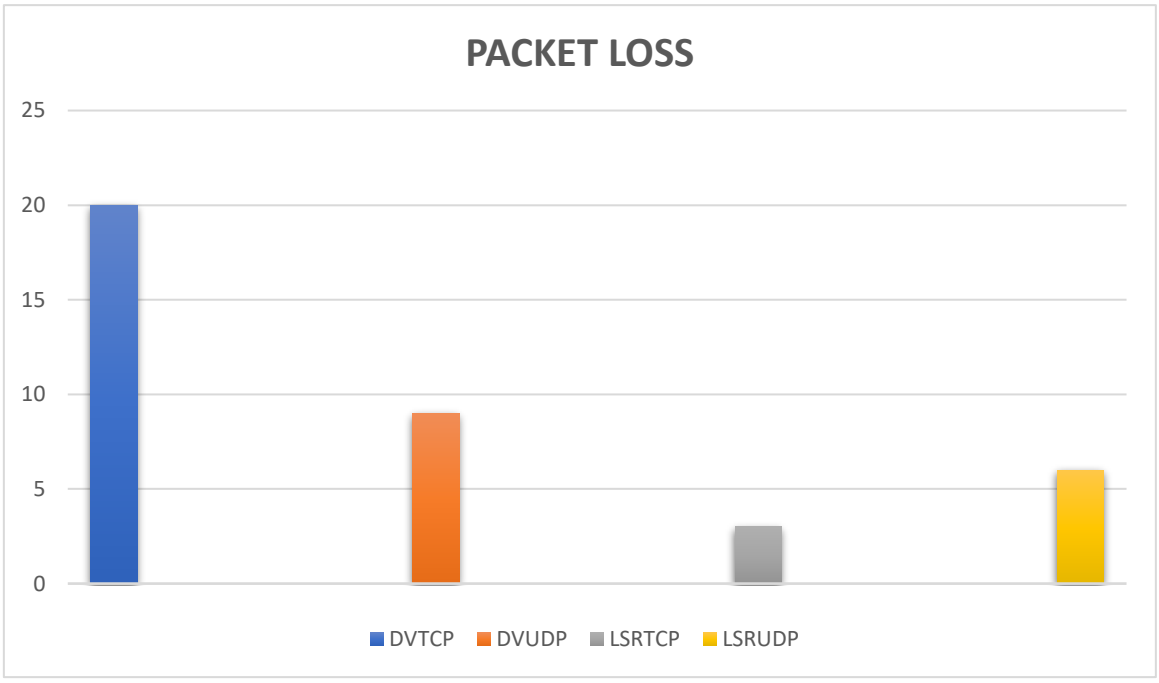

Figure-38: Packet Loss Analysis graph for the protocols simulated for 20 nodes.

In Figure-38, x-axis indicates the type of protocol used and y-axis Packet loss.

SECOND SCENARIO:

Simulation done for 32 nodes.

Program name	Start Time (in secs)	End Time (in secs)	Send Count	Received count	Data Received (in bytes)	Throughput (in bps)	Packet Delivery Ratio	Packet loss
DVTCP30	0.300000	19.962778	1051	1071	1112840.000000	452770.203681	1.019029	-20
DVUDP30	0.300000	19.985299	1131	1132	1132000.000000	460038.732457	1.000884	-1
LSRTCP30	3.300000	19.831608	689	682	708280.000000	342751.896851	0.989840	07
LSRUDP30	0.514286	19.985299	1122	1102	1102000.000000	452775.620868	0.982175	20

Table-02 : Output of Results of protocols simulated for 32 nodes on the execution of awk file

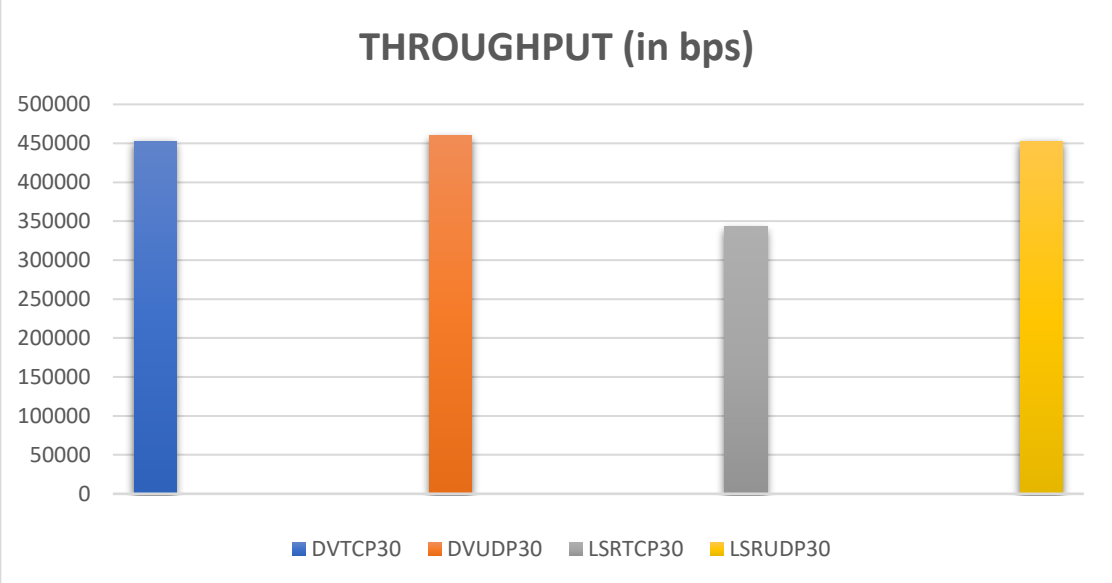

Figure-39: Throughput Analysis graph for the protocols simulated for 32 nodes.

In Figure-39, x-axis indicates the type of protocol used and y-axis indicates throughput in bps.

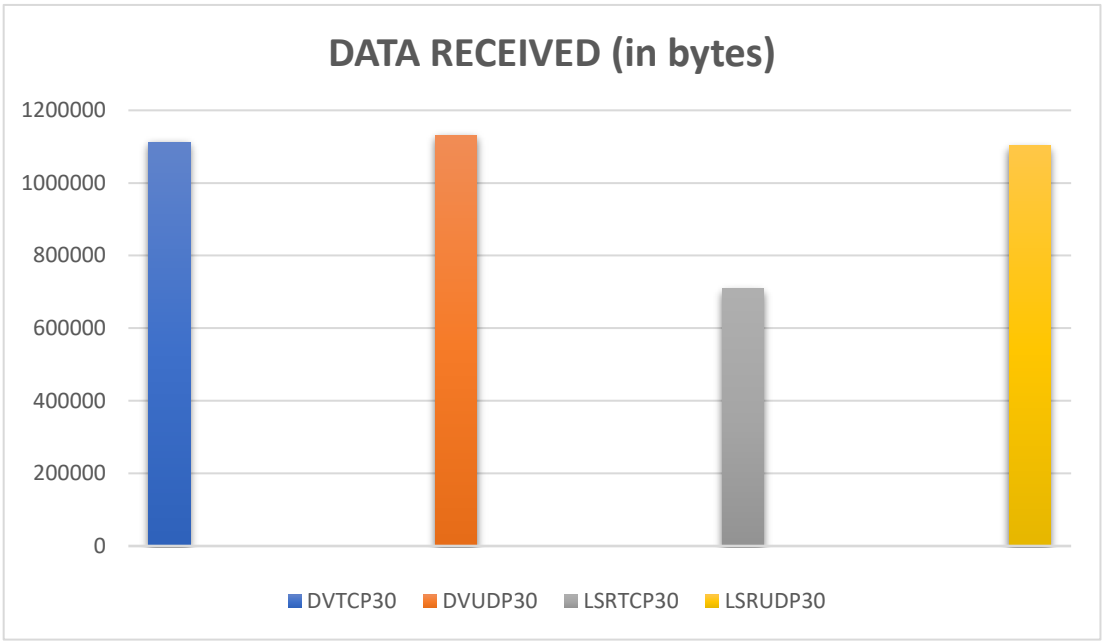

Figure-40: Data Received Analysis graph for the protocols simulated for 32 nodes.

In Figure-40, x-axis indicates the type of protocol used and y-axis indicates Data Received in bytes.

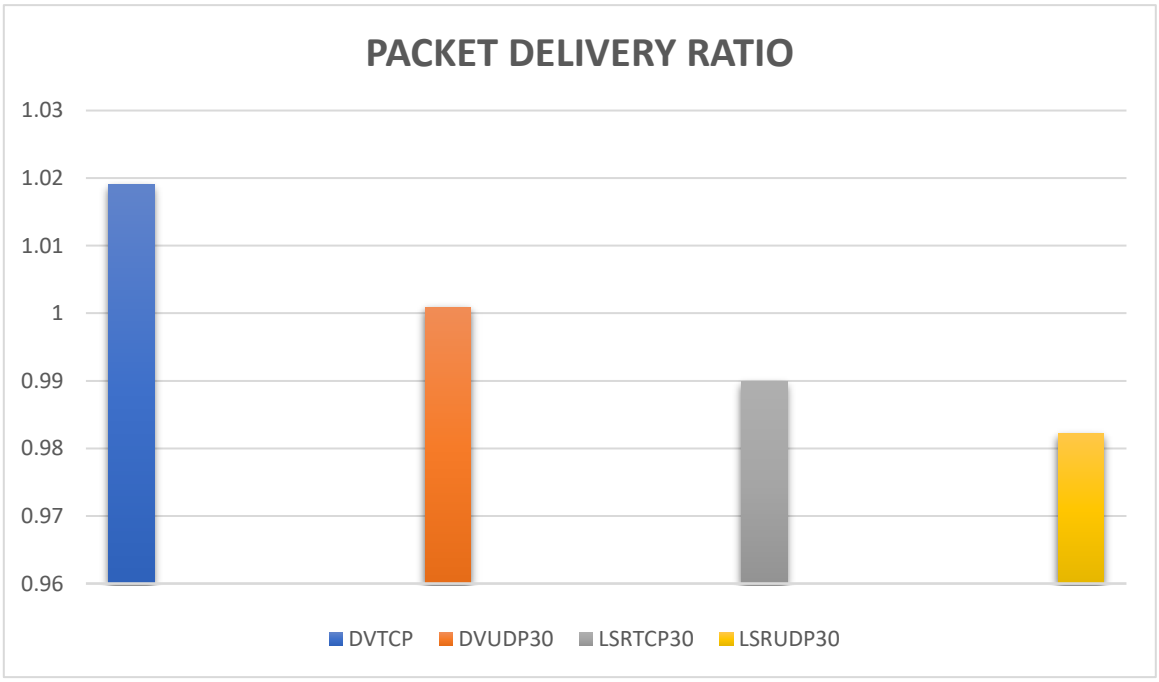

Figure-41: Packet Delivery Ratio Analysis graph for the protocols simulated for 32 nodes.

In Figure-41, x-axis indicates the type of protocol used and y-axis indicates Packet Delivery Ratio.

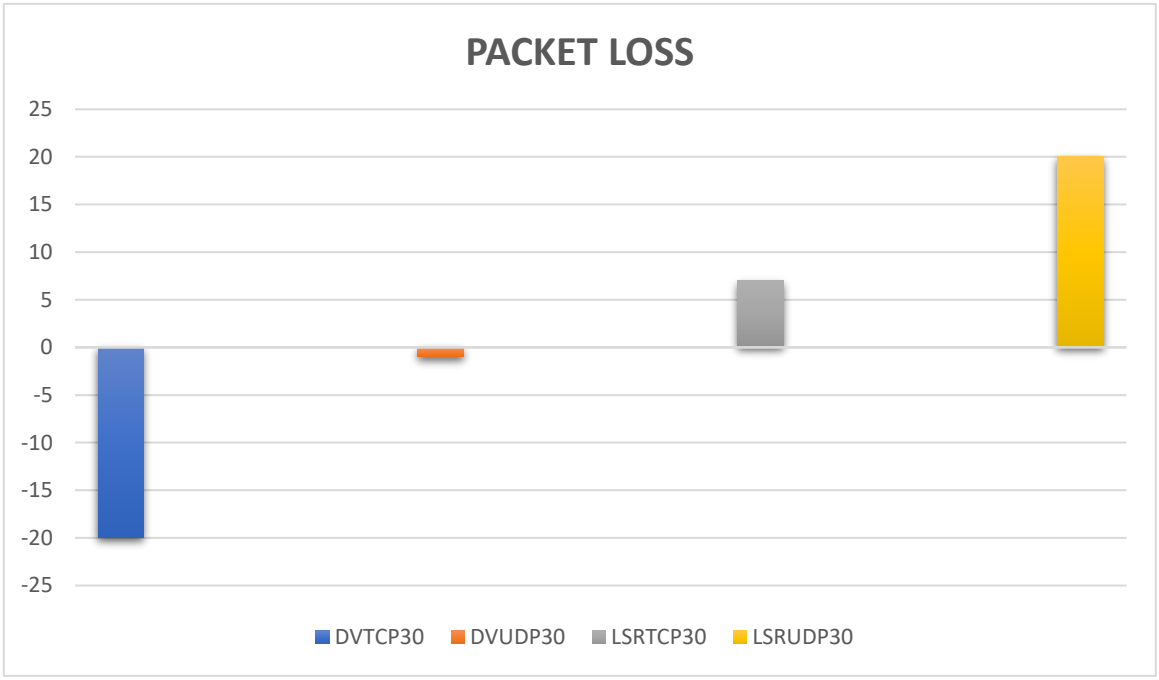

Figure-42: Packet Loss Analysis graph for the protocols simulated for 32 nodes.

In Figure-42, x-axis indicates the type of protocol used and y-axis indicates Packet loss.

NOTE : The Packet Loss is negative if there are duplicates.

CHAPTER 5

CONCLUSION

We have successfully Simulated the routing protocols in ns-2 simulator. In first scenario Link-State routing gave optimal results for throughput, Packet Delivery Ratio, Data received and less packet loss. Whereas in second scenario Distance vector routing gave slightly more optimal results.

But one of the important issue in Distance Vector routing is Count to Infinity Problem.
A routing loop is another name for count to infinity.
Routing loops usually occur when an interface goes down or when two routers send updatesto each other at the same time in Distance vecotor routing.

FUTURE ENHANCEMENT

The Link-State routing protocol floods the network by loading the link state packets. It usually results in reduction in network performance. So, the future work focuses on designing an effective Link-State routing protocol which overcomes the flooding problem.
The Link-State protocol consumes more processor power. Hence the future work focuses on overcoming these problems.

CHAPTER 6

REFERENCES

A. M. Hoque, S. O. Amin, A. Alyyan, B. Zhang, L. Zhang, and L. Wang, ‘‘NLSR: Named-data link state routing protocol,’’ in Proc. ACM SIGCOMM Workshop Inf.-Centric Netw., 2013, pp. 15– 2

B. N. P. Team. Named Data Link State Routing. [Online]. Available: https://github.com/nameddata/NLSR

C. H. Geng, X. Shi, Z. Wang, and X. Yin, “A hop-by-hop dynamic distributed multipath routing mechanism for link state network,” Computer. Communication, vol. 116, pp. 225–239, 2018.

D.] K. Lakshminarayanan, et al., “Achieving convergence-free routing using failure-carrying packets,” in Proc. ACM SIGCOMM, Aug. 2007,pp. 241–252.

E. W. Sa-Ngiamsk, S. Thipchaksurat, and R. Varakutsiripunth, ‘‘A bandwidth based constraint routing algorithm for multi-protocol label switching networks,’’ in Proc. 6th Int. Conf. Adv. Commun. Technol., vol. 2, Feb. 2004, pp. 933–937.

F. S. Chaudhuri, S. R. Das, H. S. Paul, and S. Tirthapura, ‘‘Distributed computing and networking,’’ in Proc. Int. Conf. Distrib. Comput. Netw.,2013.