University of Dhaka

Department of Computer Science and Engineering

CSE-3111 : Computer Networking Lab

Lab Report 7: Implementation of Link State Routing Algorithm

Submitted By:

Name : Abdullah Al Mahmud

Roll No : 15

Name : Zisan Mahmud

Roll No : 23

Submitted On:

March 28, 2024

Submitted To:

Dr. Md. Abdur Razzaque

Dr. Md Mamunur Rashid

Dr. Muhammad Ibrahim

Mr. Md. Redwan Ahmed Rizvee

1 Introduction

The Link State Algorithm is a widely used routing algorithm in computer networks. It enables routers to create a map of the entire network topology and compute the shortest path to each destination. In this lab, our main goal is to design and implement a program that simulates the Link State Routing Algorithm, demonstrating its functionality and gaining practical experience in using the algorithm for routing purposes.

2 Objectives

The preliminary objective of this lab is,

- to develop an understanding of the Link State Algorithm and its applications in computer networks.
- to learn the exchange of information among routers in a topology and how the path is calculated for this purpose.

3 Theory

The link-state routing method allows each node to create a map to find a way to their network connectivity . Then, every node calculates the best logical way to every potential destination in the network by itself. This concept guarantees that every node is aware of the real-time network topology and, therefore, can swiftly recognize the shortest route for data delivery.

In link state algorithm, the network topology and all link costs are known, that is, available as input to the LS algorithm. This is accomplished by having each node broadcast link-state packets to all other nodes in the network, with each link-state packet containing the identities and costs of its attached links. The result of the nodes' broadcast is that all nodes have an identical and complete view of the network. Each node can then run the LS algorithm and compute the same set of least-cost paths as every other node.

In link state algorithm, the algorithm used is know as Dijkstra's algorithm. It computes the least-cost path from one node (the source, which we will refer to as u) to all other nodes in the network. Dijkstra's algorithm is iterative and has the property that after the k-th iteration of the algorithm, the least-cost paths are known to k destination nodes, and among the least-cost paths to all destination nodes, these k paths will have the k smallest costs. The centralized routing algorithm consists of an initialization step followed by a loop. The number of times the loop is executed is equal to the number of nodes in the network. Upon termination, the algorithm will have calculated the shortest paths from the source node u to every other node in the network.

Compared to a Distance Vector Algorithm, the Link State Algorithm has several advantages. Firstly, it can work well on a large-scale network with thousands or even millions of nodes. Secondly, it is more efficient because it does not force the router to announce the entire routing table to each another router was in the network. This way, only information about topology change distributed and overall network traffic can be reduced, thus increasing efficiency.

4 Methodology

1. First, we design a network topology.
2. Then we implement this network that includes several routers, links, packets, and routing algorithms.
3. Read the list of edges for each router.
4. Create Link State Packet (LSP) containing id, TTL, cost of immediate edges and send to all neighbors.
5. Neighbors will broadcast it to their neighbors and so on – this is called flooding.
6. After a node has received a new message, it will run Dijkstra and calculate the current shortest path to all other nodes.
7. Then we update the network topology based on LSPs
8. Finally, we test the performance of the algorithm.

5 Experimental Result

5.1 Network topology

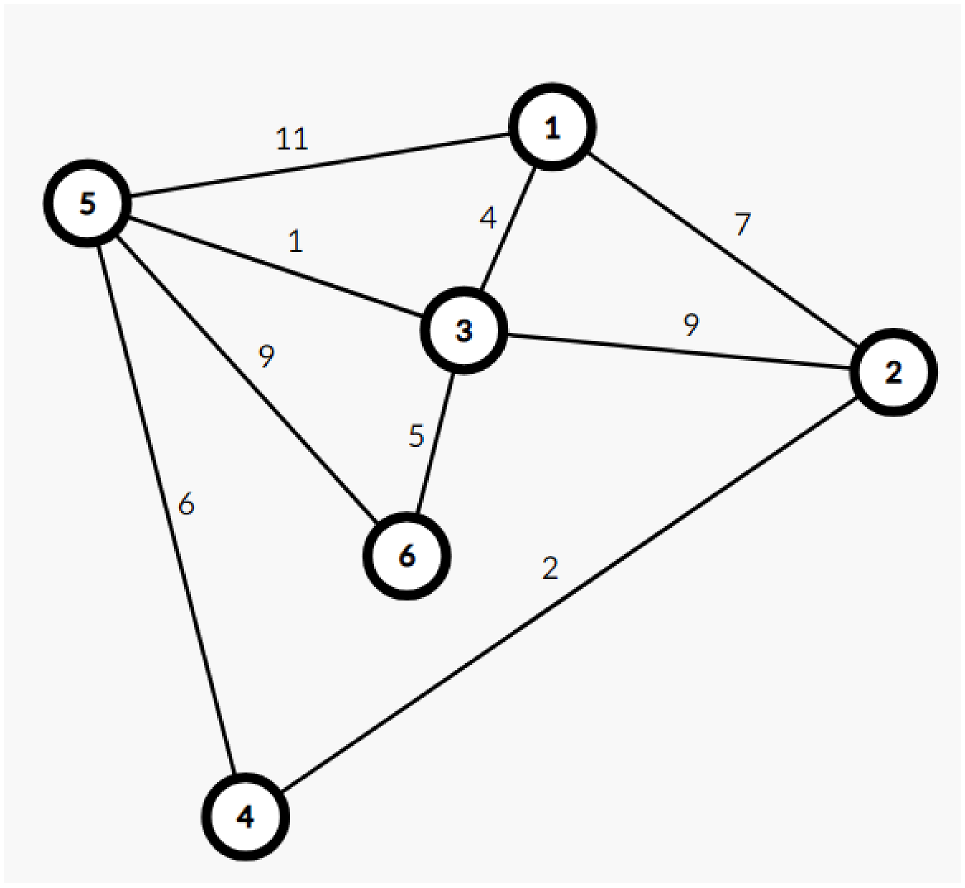
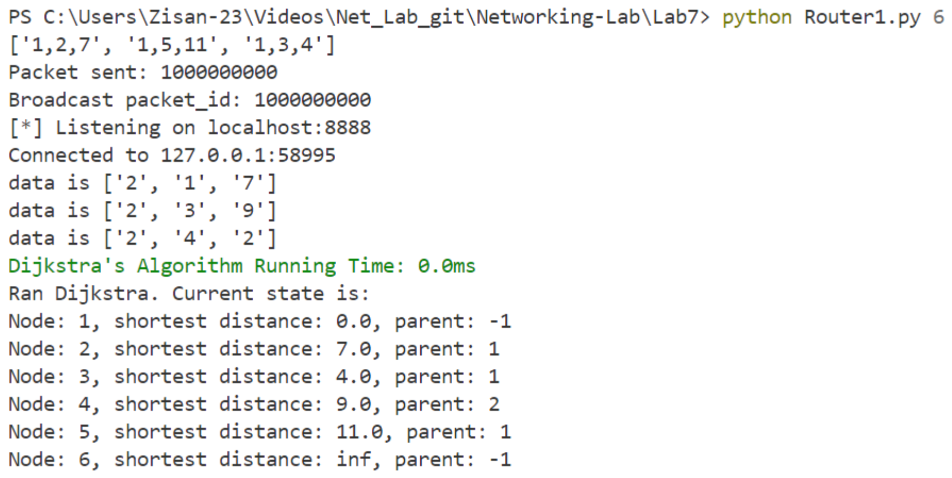


Figure 1: Initial Graph

5.2 Output before link update

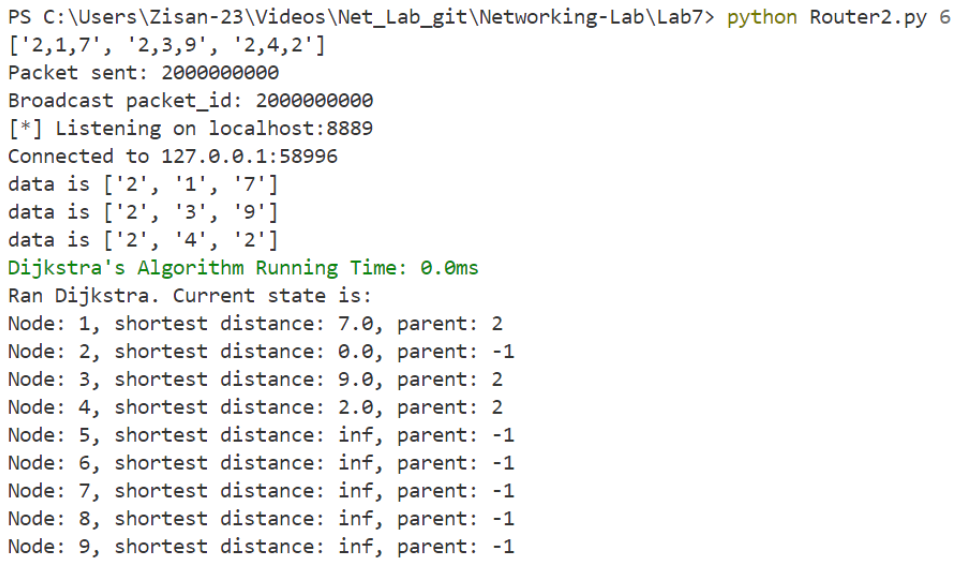
- **Router 1:**



```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router1.py 6
['1,2,7', '1,5,11', '1,3,4']
Packet sent: 1000000000
Broadcast packet_id: 1000000000
[*] Listening on localhost:8888
Connected to 127.0.0.1:58995
data is ['2', '1', '7']
data is ['2', '3', '9']
data is ['2', '4', '2']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 0.0, parent: -1
Node: 2, shortest distance: 7.0, parent: 1
Node: 3, shortest distance: 4.0, parent: 1
Node: 4, shortest distance: 9.0, parent: 2
Node: 5, shortest distance: 11.0, parent: 1
Node: 6, shortest distance: inf, parent: -1
```

Figure 2: Router 1 initial path cost

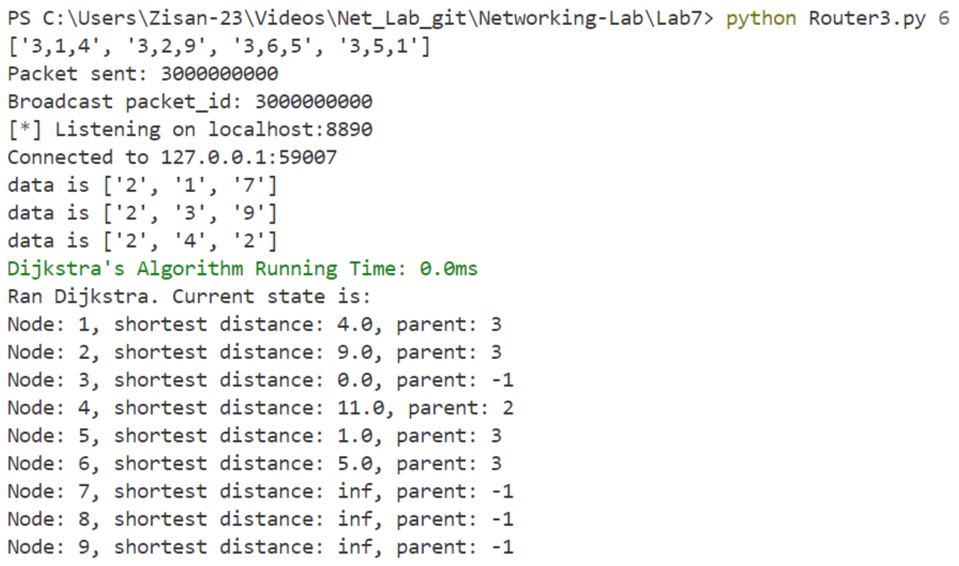
- **Router 2:**



```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router2.py 6
['2,1,7', '2,3,9', '2,4,2']
Packet sent: 2000000000
Broadcast packet_id: 2000000000
[*] Listening on localhost:8889
Connected to 127.0.0.1:58996
data is ['2', '1', '7']
data is ['2', '3', '9']
data is ['2', '4', '2']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 7.0, parent: 2
Node: 2, shortest distance: 0.0, parent: -1
Node: 3, shortest distance: 9.0, parent: 2
Node: 4, shortest distance: 2.0, parent: 2
Node: 5, shortest distance: inf, parent: -1
Node: 6, shortest distance: inf, parent: -1
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
```

Figure 3: Router 2 initial path cost

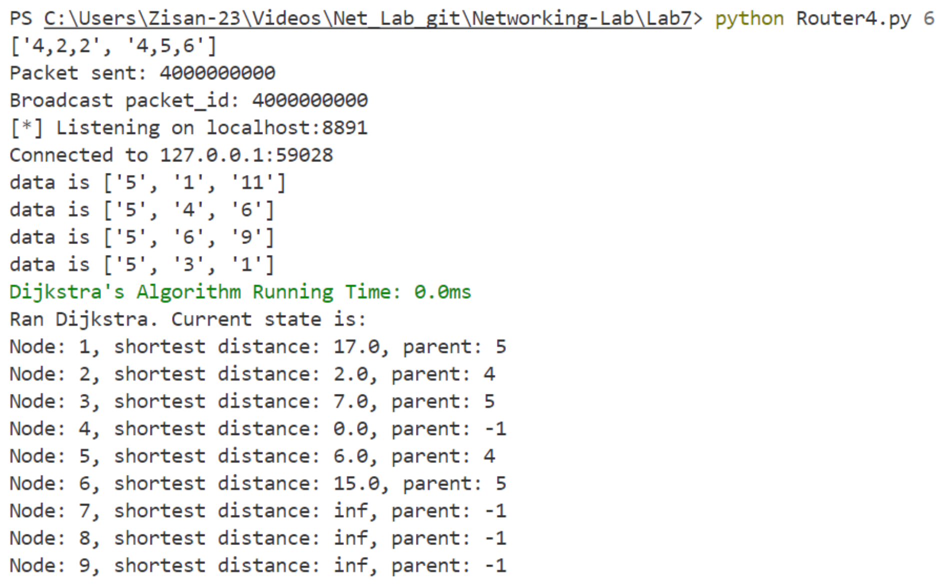
- **Router 3:**



```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router3.py 6
['3,1,4', '3,2,9', '3,6,5', '3,5,1']
Packet sent: 3000000000
Broadcast packet_id: 3000000000
[*] Listening on localhost:8890
Connected to 127.0.0.1:59007
data is ['2', '1', '7']
data is ['2', '3', '9']
data is ['2', '4', '2']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 4.0, parent: 3
Node: 2, shortest distance: 9.0, parent: 3
Node: 3, shortest distance: 0.0, parent: -1
Node: 4, shortest distance: 11.0, parent: 2
Node: 5, shortest distance: 1.0, parent: 3
Node: 6, shortest distance: 5.0, parent: 3
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
```

Figure 4: Router 3 initial path cost

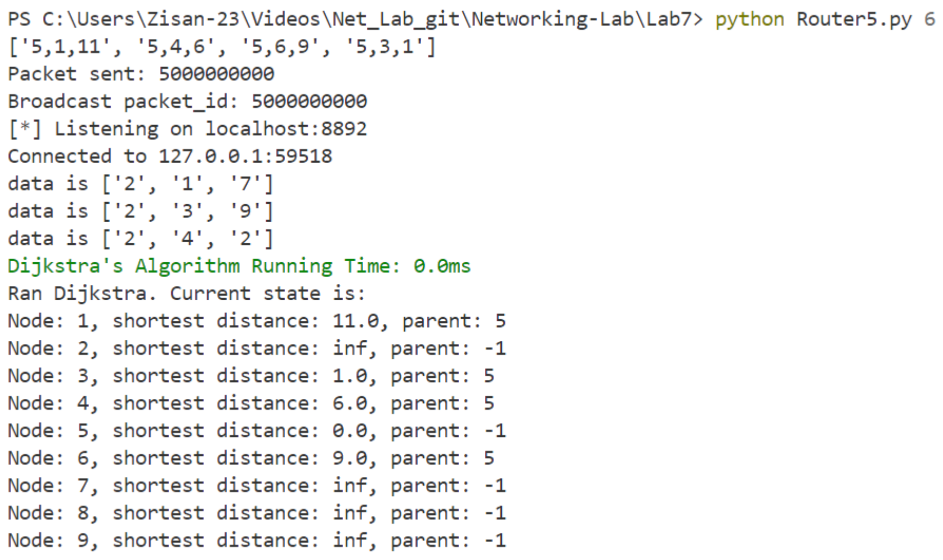
- **Router 4:**



```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router4.py 6
['4,2,2', '4,5,6']
Packet sent: 4000000000
Broadcast packet_id: 4000000000
[*] Listening on localhost:8891
Connected to 127.0.0.1:59028
data is ['5', '1', '11']
data is ['5', '4', '6']
data is ['5', '6', '9']
data is ['5', '3', '1']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 17.0, parent: 5
Node: 2, shortest distance: 2.0, parent: 4
Node: 3, shortest distance: 7.0, parent: 5
Node: 4, shortest distance: 0.0, parent: -1
Node: 5, shortest distance: 6.0, parent: 4
Node: 6, shortest distance: 15.0, parent: 5
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
```

Figure 5: Router 4 initial path cost

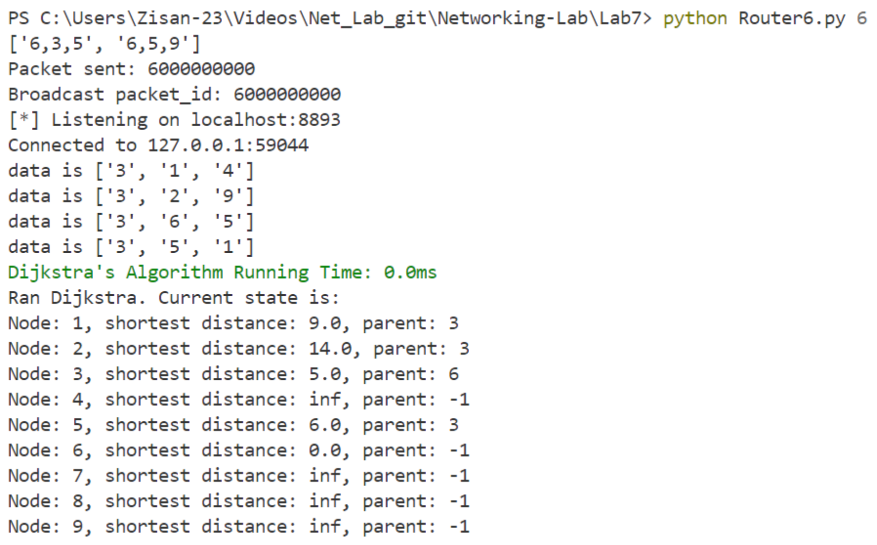
- **Router 5:**



```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router5.py 6
['5,1,11', '5,4,6', '5,6,9', '5,3,1']
Packet sent: 5000000000
Broadcast packet_id: 5000000000
[*] Listening on localhost:8892
Connected to 127.0.0.1:59518
data is ['2', '1', '7']
data is ['2', '3', '9']
data is ['2', '4', '2']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 11.0, parent: 5
Node: 2, shortest distance: inf, parent: -1
Node: 3, shortest distance: 1.0, parent: 5
Node: 4, shortest distance: 6.0, parent: 5
Node: 5, shortest distance: 0.0, parent: -1
Node: 6, shortest distance: 9.0, parent: 5
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
```

Figure 6: Router 5 initial path cost

- **Router 6:**

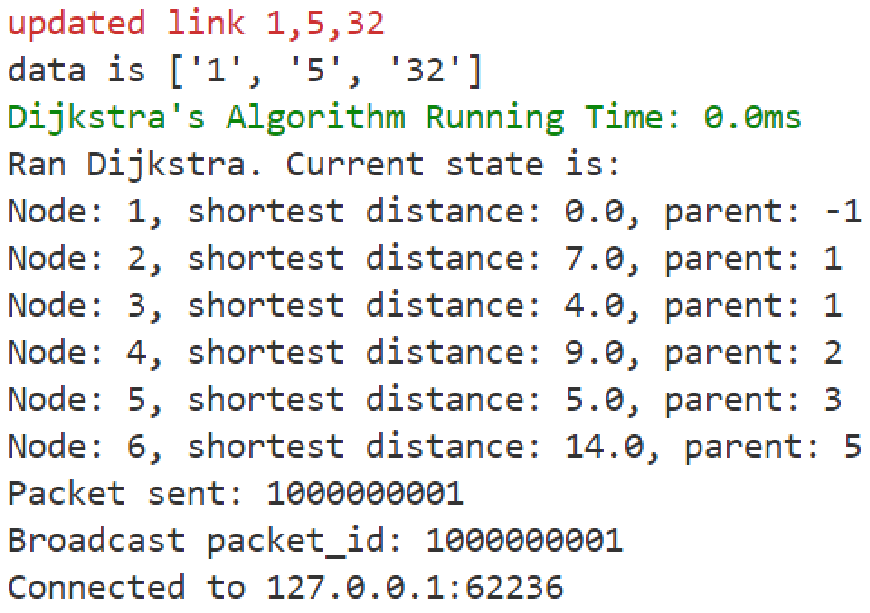


```
PS C:\Users\Zisan-23\Videos\Net_Lab_git\Networking-Lab\Lab7> python Router6.py 6
['6,3,5', '6,5,9']
Packet sent: 6000000000
Broadcast packet_id: 6000000000
[*] Listening on localhost:8893
Connected to 127.0.0.1:59044
data is ['3', '1', '4']
data is ['3', '2', '9']
data is ['3', '6', '5']
data is ['3', '5', '1']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 9.0, parent: 3
Node: 2, shortest distance: 14.0, parent: 3
Node: 3, shortest distance: 5.0, parent: 6
Node: 4, shortest distance: inf, parent: -1
Node: 5, shortest distance: 6.0, parent: 3
Node: 6, shortest distance: 0.0, parent: -1
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
```

Figure 7: Router 6 initial path cost

5.3 Output after link update

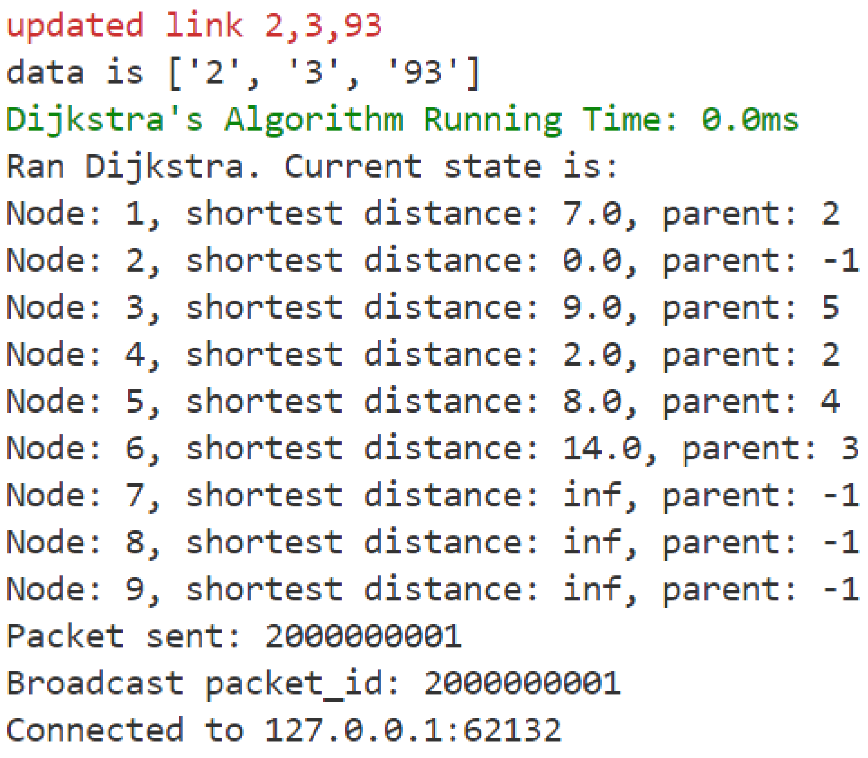
- **Router 1:**



```
updated link 1,5,32
data is ['1', '5', '32']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 0.0, parent: -1
Node: 2, shortest distance: 7.0, parent: 1
Node: 3, shortest distance: 4.0, parent: 1
Node: 4, shortest distance: 9.0, parent: 2
Node: 5, shortest distance: 5.0, parent: 3
Node: 6, shortest distance: 14.0, parent: 5
Packet sent: 1000000001
Broadcast packet_id: 1000000001
Connected to 127.0.0.1:62236
```

Figure 8: Router 1 updated path cost

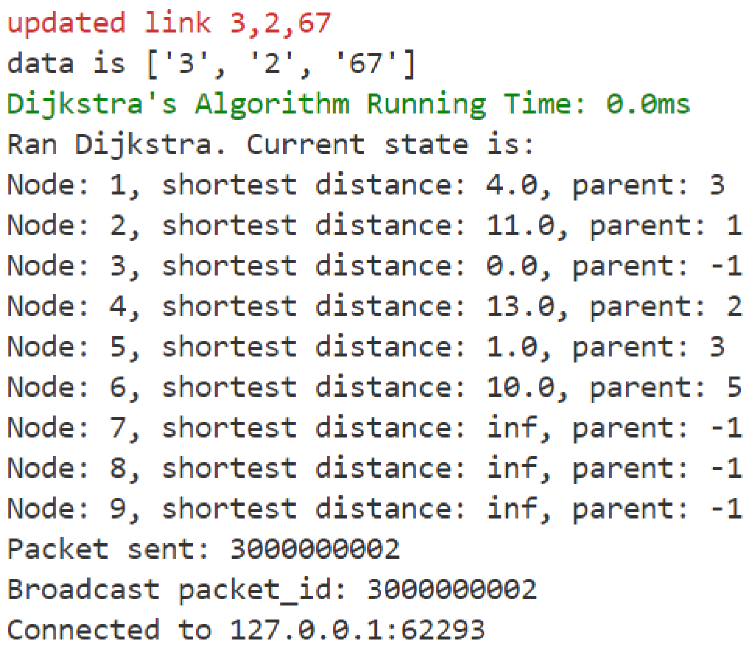
- **Router 2:**



```
updated link 2,3,93
data is ['2', '3', '93']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 7.0, parent: 2
Node: 2, shortest distance: 0.0, parent: -1
Node: 3, shortest distance: 9.0, parent: 5
Node: 4, shortest distance: 2.0, parent: 2
Node: 5, shortest distance: 8.0, parent: 4
Node: 6, shortest distance: 14.0, parent: 3
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
Packet sent: 2000000001
Broadcast packet_id: 2000000001
Connected to 127.0.0.1:62132
```

Figure 9: Router 2 updated path cost

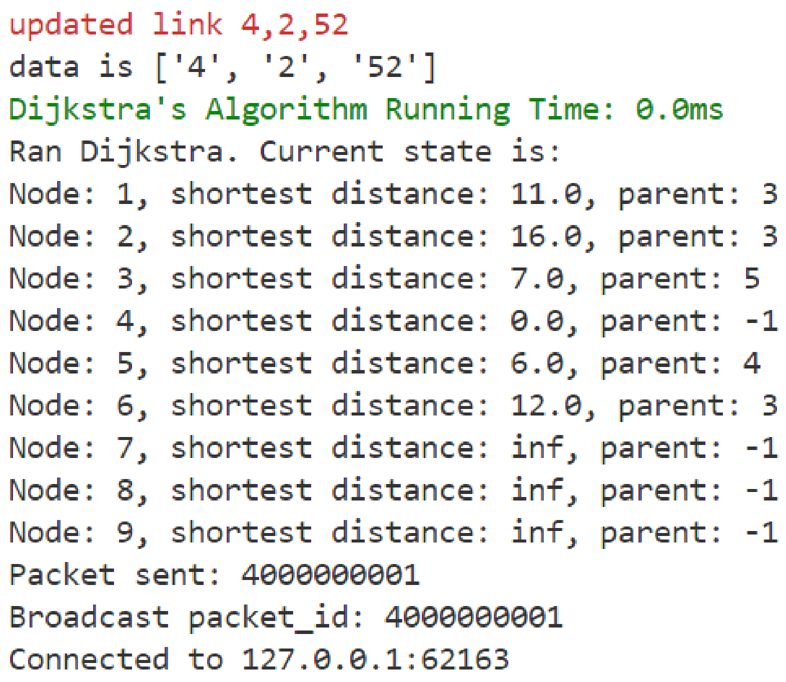
- **Router 3:**



```
updated link 3,2,67
data is ['3', '2', '67']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 4.0, parent: 3
Node: 2, shortest distance: 11.0, parent: 1
Node: 3, shortest distance: 0.0, parent: -1
Node: 4, shortest distance: 13.0, parent: 2
Node: 5, shortest distance: 1.0, parent: 3
Node: 6, shortest distance: 10.0, parent: 5
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
Packet sent: 3000000002
Broadcast packet_id: 3000000002
Connected to 127.0.0.1:62293
```

Figure 10: Router 3 updated path cost

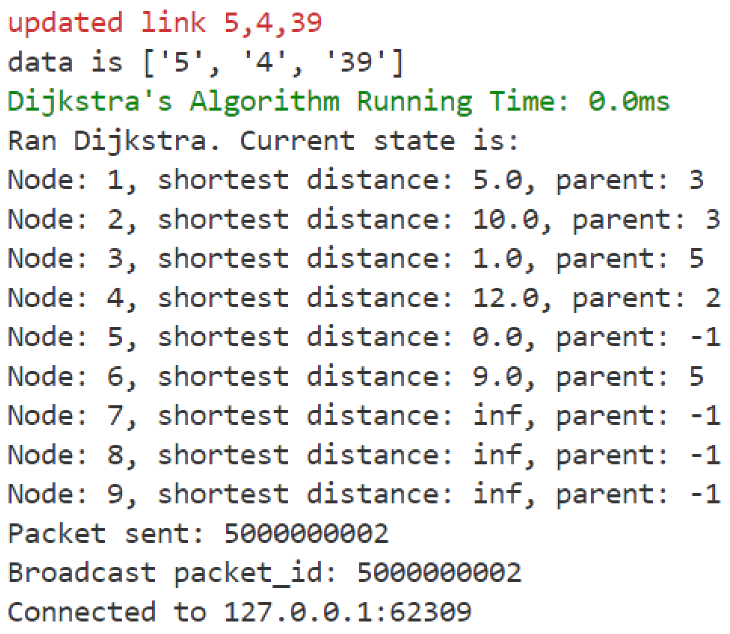
- **Router 4:**



```
updated link 4,2,52
data is ['4', '2', '52']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 11.0, parent: 3
Node: 2, shortest distance: 16.0, parent: 3
Node: 3, shortest distance: 7.0, parent: 5
Node: 4, shortest distance: 0.0, parent: -1
Node: 5, shortest distance: 6.0, parent: 4
Node: 6, shortest distance: 12.0, parent: 3
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
Packet sent: 4000000001
Broadcast packet_id: 4000000001
Connected to 127.0.0.1:62163
```

Figure 11: Router 4 updated path cost

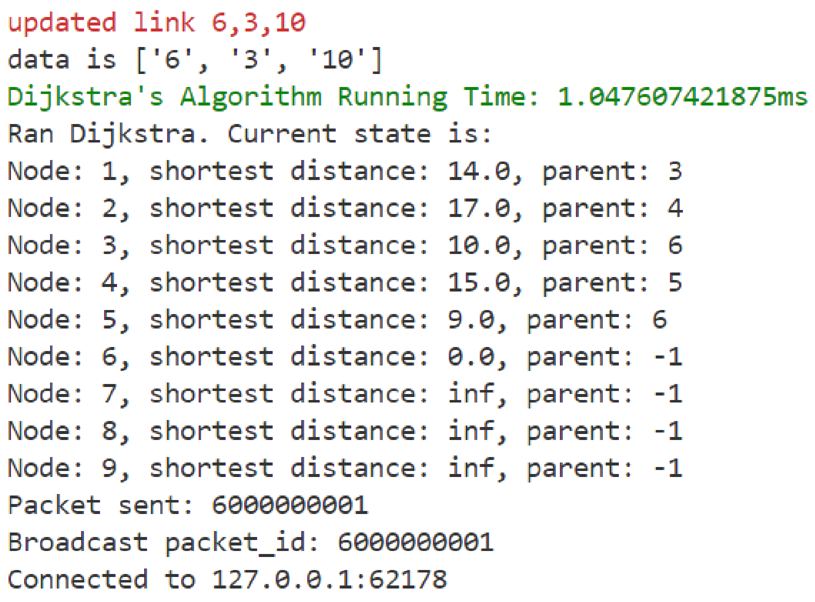
- **Router 5:**



```
updated link 5,4,39
data is ['5', '4', '39']
Dijkstra's Algorithm Running Time: 0.0ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 5.0, parent: 3
Node: 2, shortest distance: 10.0, parent: 3
Node: 3, shortest distance: 1.0, parent: 5
Node: 4, shortest distance: 12.0, parent: 2
Node: 5, shortest distance: 0.0, parent: -1
Node: 6, shortest distance: 9.0, parent: 5
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
Packet sent: 5000000002
Broadcast packet_id: 5000000002
Connected to 127.0.0.1:62309
```

Figure 12: Router 5 updated path cost

- **Router 6:**



```
updated link 6,3,10
data is ['6', '3', '10']
Dijkstra's Algorithm Running Time: 1.047607421875ms
Ran Dijkstra. Current state is:
Node: 1, shortest distance: 14.0, parent: 3
Node: 2, shortest distance: 17.0, parent: 4
Node: 3, shortest distance: 10.0, parent: 6
Node: 4, shortest distance: 15.0, parent: 5
Node: 5, shortest distance: 9.0, parent: 6
Node: 6, shortest distance: 0.0, parent: -1
Node: 7, shortest distance: inf, parent: -1
Node: 8, shortest distance: inf, parent: -1
Node: 9, shortest distance: inf, parent: -1
Packet sent: 6000000001
Broadcast packet_id: 6000000001
Connected to 127.0.0.1:62178
```

Figure 13: Router 6 updated path cost

5.4 Graphical Analysis

The following graph depicts the relationship between the number of nodes and the time required to execute Dijkstra's Algorithm, measured in microseconds.

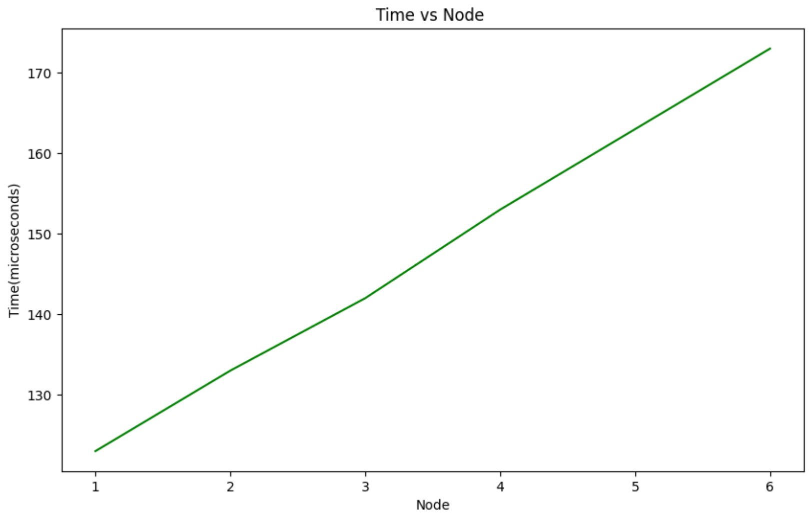


Figure 14: Neighbor-to-neighbor communication

The following graph depicts the relationship between the number of nodes and the auxiliary memory space required to execute Dijkstra's Algorithm, measured in bytes.

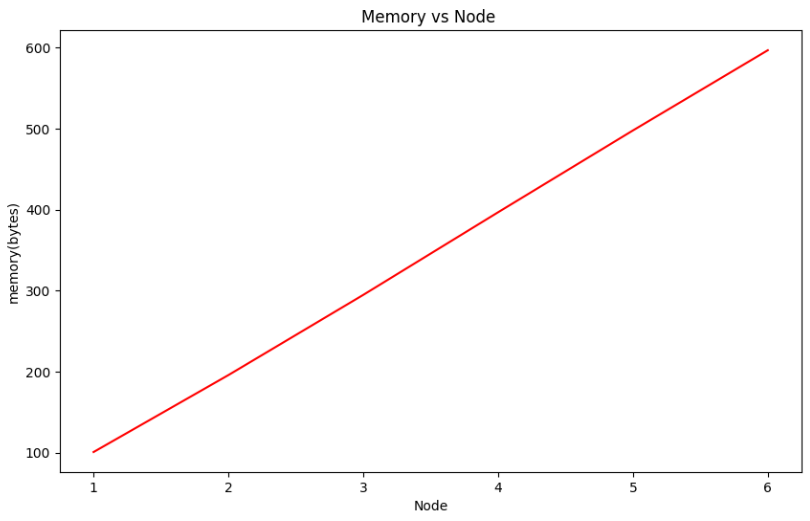


Figure 15: Neighbor-to-neighbor communication

References

[1] James F. Kurose and Keith W. Ross, *Computer Networking: A Top-Down Approach*, 8th Edition, Pearson,, 2022.

[2] GeeksforGeeks, *Unicast Routing – Link State Routing*, `https://www.geeksforgeeks.org/unicast-routing-link-state-routing/`

[3] TJava Point, *Link State Routing*, `https://www.javatpoint.com/link-state-routing-algorithm`

[4] CodingNinjas, *Link state routing*, `https://www.codingninjas.com/studio/library/link-state-routing`