

Project Operations Manual

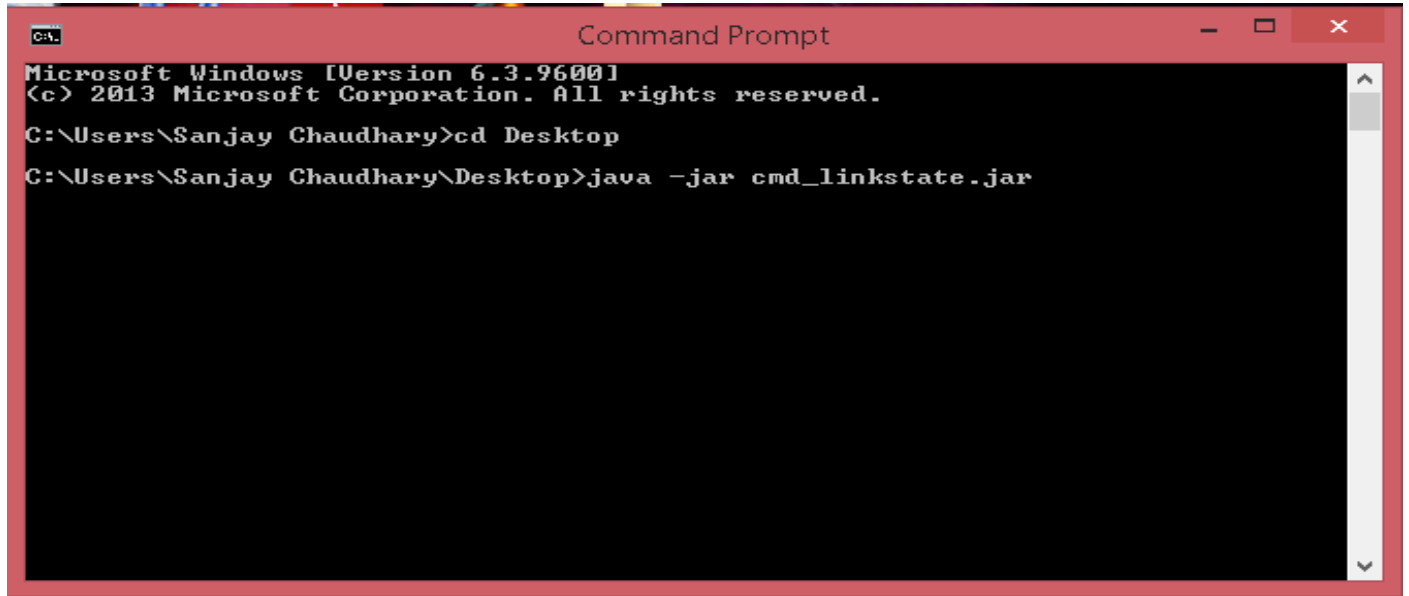
- The link state routing protocol performs following operations:
 1. To build Network Topology.
 2. Build Connection Tables.
 3. Find shortest path between source and destination nodes.
 4. Modify a topology.
 5. Exit.
- User is given option to enter any one of the above operations.
- For the first time, all the operations are to be done in order. This is because
 - Option 1 takes the filename from user and keeps it same throughout until changed by user. Filename has to be with .txt extension.
 - Option 2 takes source node from user and keeps it same throughout until changed by user.
 - Option 3 takes destination node from user and keeps it same throughout until changed by user.
 - Option 4 takes the router node to be taken down from user and keeps it same until changed by user.
- This project is implemented on command prompt as well as using GUI (Java Frames).
- Source code contains two folders:
 - 1) cmd_linkstate.
 - 2) Gui_linkstate.
- To run project on cmd following all files should be kept under folder cmd_files:
 - Input.txt: Text file containing the matrix topology of 8 nodes of the network. First row and First column displays the router nodes (1,2,3....no_of_nodes).
 - Dijkstras.java: Contains code for building connection table of source node.
 - Linkstate.java: Contains code for calling respective methods required for respective option from main menu.
 - ShortestPath.java: Contains code for finding shortest path from source to destination.
 - SP.java: calls ShortestPath class methods for finding the shortest path and prints it.
 - Routerdown.java: Takes router down node and updates connection table accordingly.
 - Routerdownpath.java: Takes router down node and updates the shortest path accordingly.
- To run project using GUI following all files should be kept under folder named: GUI_files:
 - Gui.java, Gui_Dijkstras.java, Gui_SP.java, Routerddown.java, Routerdownpath.java, ShortestPath.java, SP.java and Input.txt

Description of Link State Routing Protocol applied in this project

- For implementing the link state routing protocol, two lists are created:
 - 1) Nodes: contains the no of nodes in the network.
 - 2) M: contains the list of Min_node that is the nodes with minimum costs from source.
- As soon as the Min_node is obtained that node is removed from Nodes list and added to M. This program runs until number of nodes in Nodes list becomes zero.
- To build connection table of each node (source), algo maintains an array Nexthop[]. For all the nodes m in the network, Nexthop[m] is the node which leads to min cost path from source to node m.
- To find shortest path between source and destination node, connection table of source is called. The Nexthop[source] is checked from connection table and taken as new source s. This is done recursively until Nexthop[s] equals the destination node. Nexthop[s] gives the intermediate nodes of shortest path between requested nodes.
- To find minimum cost from source to destination node, algo maintains an array D[m] for each node m in the network. D[m] for each node m stores the minimum distance to that node m from source s.
- To find shortest path after any router being down, the distance to and from the node to other nodes in network are updated to -1 and that router is omitted from the list of Nodes of network.
- After removing that router down node, connection tables are updated for remaining nodes in the network and hence shortest path and min cost is calculated from that.
- If the router down node entered is either source or destination node then user is requested to enter other option as if either source or destination node is brought down it will lead to infinite path.

How to compile and run the project:

- This project is implemented on cmd as well as GUI.
- Executable Jar files for both cmd_linkstate and Gui_linkstate projects is created.
- To run jar file of cmd_linkstate:
 - Open command prompt.
 - Type command “java -jar cmd_linkstate.jar”.
 - This will run the cmd_linkstate program.
 - Then enter the options as per your choice.
- To run using GUI:
 - Click on the Gui_linkstate.jar file.
 - This will run the Gui_linkstate program.
 - Then enter the options as per your choice.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Sanjay Chaudhary>cd Desktop
C:\Users\Sanjay Chaudhary\Desktop>java -jar cmd_linkstate.jar
```

Fig 1: Running cmd_linkstate.jar

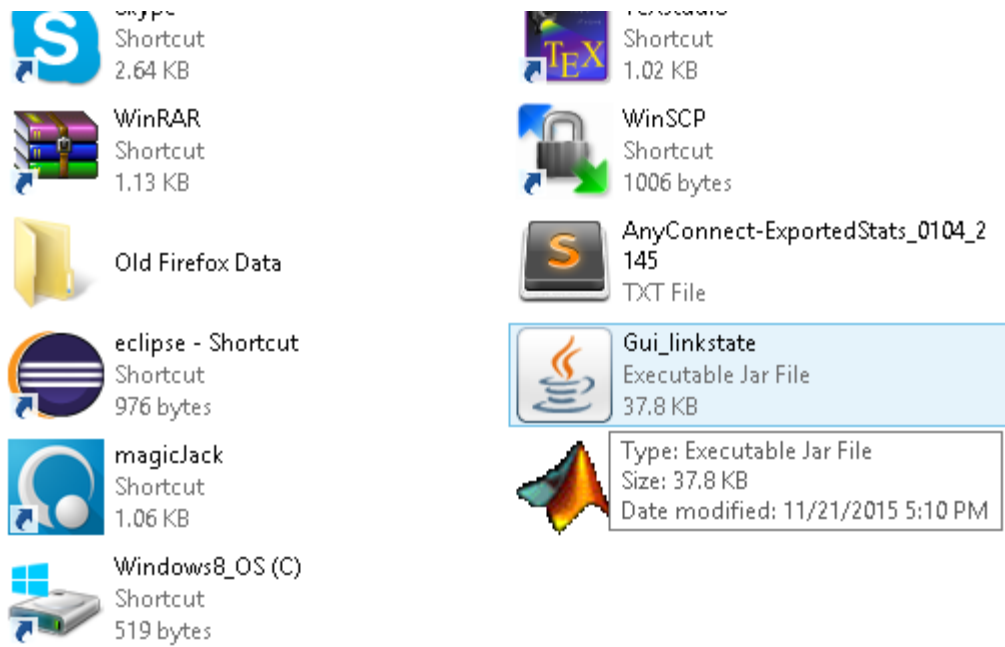
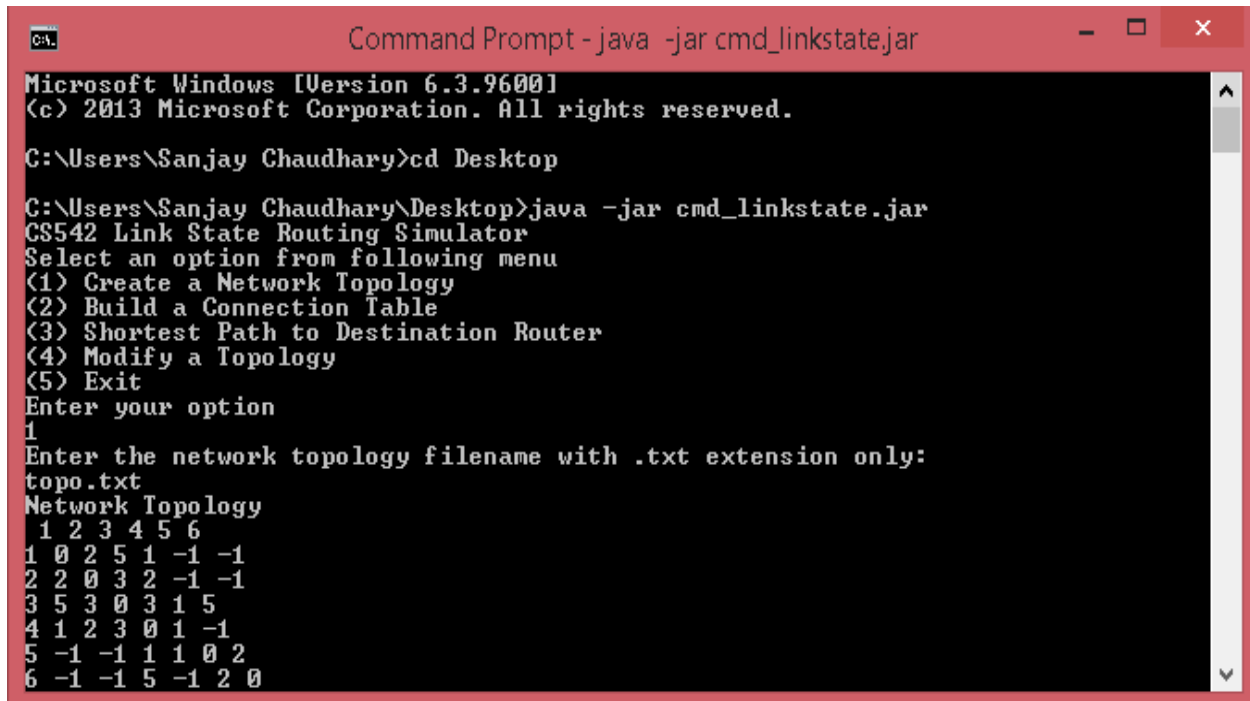


Figure 2: Click on Gui_linkstate to execute GUI of Link state protocol.

TEST REPORT:

- In order to test the program, different files with different topologies were given as input.
- To verify the results, manual calculations were performed for the same input file.

➤ Output by CS542 Link State Routing Protocol for a 6 node topology:



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Sanjay Chaudhary>cd Desktop

C:\Users\Sanjay Chaudhary\Desktop>java -jar cmd_linkstate.jar
CS542 Link State Routing Simulator
Select an option from following menu
(1) Create a Network Topology
(2) Build a Connection Table
(3) Shortest Path to Destination Router
(4) Modify a Topology
(5) Exit
Enter your option
1
Enter the network topology filename with .txt extension only:
topo.txt
Network Topology
 1 2 3 4 5 6
1 0 2 5 1 -1 -1
2 2 0 3 2 -1 -1
3 5 3 0 3 1 5
4 1 2 3 0 1 -1
5 -1 -1 1 1 0 2
6 -1 -1 5 -1 2 0
```

Figure 3: Network Topology of file topo.txt with 6 nodes

```
Command Prompt - java -jar cmd_linkstate.jar

7CS542 Link State Routing Simulator
Select an option from following menu
<1> Create a Network Topology
<2> Build a Connection Table
<3> Shortest Path to Destination Router
<4> Modify a Topology
<5> Exit
Enter your option
2
Building a connection table
Enter source router
1
Router 1      Connection Table
Destination Router      Nexthop
=====
1                        1
2                        2
3                        4
4                        4
5                        4
6                        4
CS542 Link State Routing Simulator
Select an option from following menu
<1> Create a Network Topology
<2> Build a Connection Table
```

Figure 4: Connection table of source node 1

```
Command Prompt - java -jar cmd_linkstate.jar

CS542 Link State Routing Simulator
Select an option from following menu
<1> Create a Network Topology
<2> Build a Connection Table
<3> Shortest Path to Destination Router
<4> Modify a Topology
<5> Exit
Enter your option
3
Shortest Path to Destination Router
Select a destination router
6
56Shortest Path from      1      to      6      is
1      4      5      6
Minimum cost is 4
CS542 Link State Routing Simulator
Select an option from following menu
<1> Create a Network Topology
<2> Build a Connection Table
<3> Shortest Path to Destination Router
<4> Modify a Topology
<5> Exit
Enter your option
4
Select a router to take it down
```

Figure 5: Shortest path with minimum cost from source node 1 to destination node 6.

```
Command Prompt - java -jar cmd_linkstate.jar

Enter your option
4
Select a router to take it down
4
Destination Router      Nexthop
=====
1                        1
2                        2
3                        2
5                        3
6                        3
Shortest Path after removing down node from 1 to 6 is
1      3      5      6
Minimum Cost 2
CS542 Link State Routing Simulator
Select an option from following menu
<1> Create a Network Topology
<2> Build a Connection Table
<3> Shortest Path to Destination Router
<4> Modify a Topology
<5> Exit
Enter your option
```

Figure 6: After router 4 being taken down, updated shortest path from source 1 to destination 6

Manual Calculations:

Results were successfully verified using manual calculations.

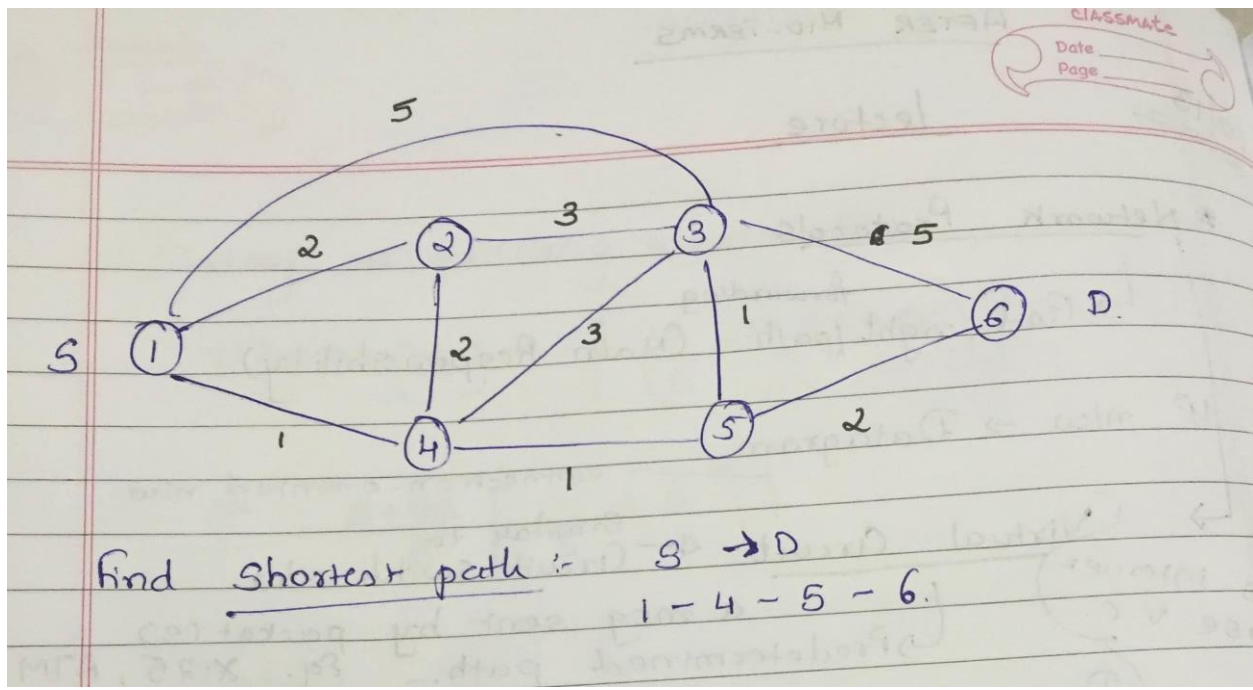


Figure 7: Topology of Input file.

⇒ Dijkstra's Algo (^{Routing} link state algo)

d. D_n : Cost of the least-cost path from node S^n to node n .

N : Set of nodes in the n/w.

S : Source node.

M : Set of nodes, so far incorporated in the algo.

d_{ij} : link cost from node i to node j .

$$d_{ii} = \phi$$

$$d_{ij} = \infty \quad \text{if } i, j \text{ are not adj.}$$

Algo.

Step 1: Initialization

$$M = \{S\}$$

Figure 8: Calculating using Dijkstra's Algo.

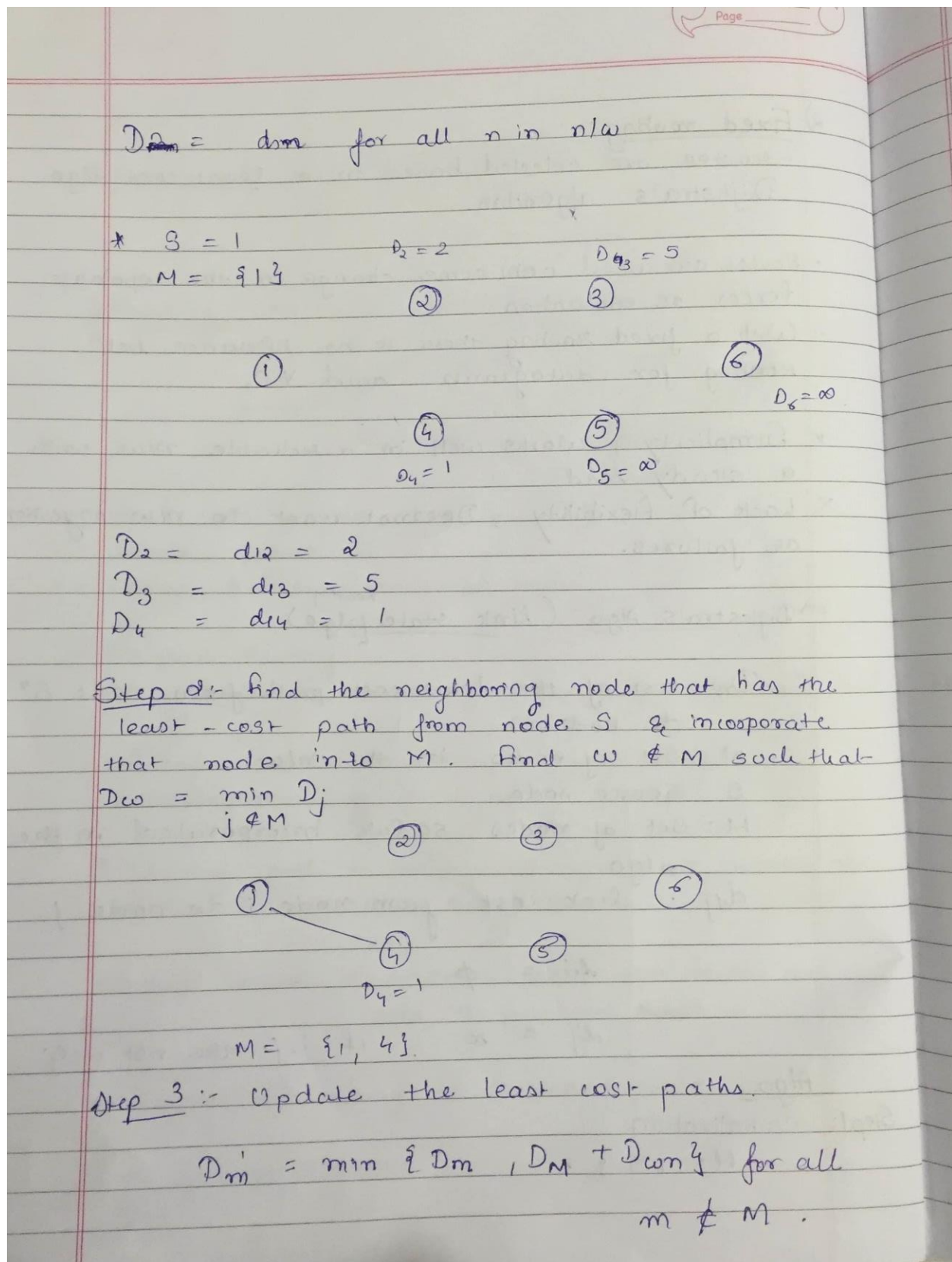


Figure 9: Calculating using Dijkstra's Algo

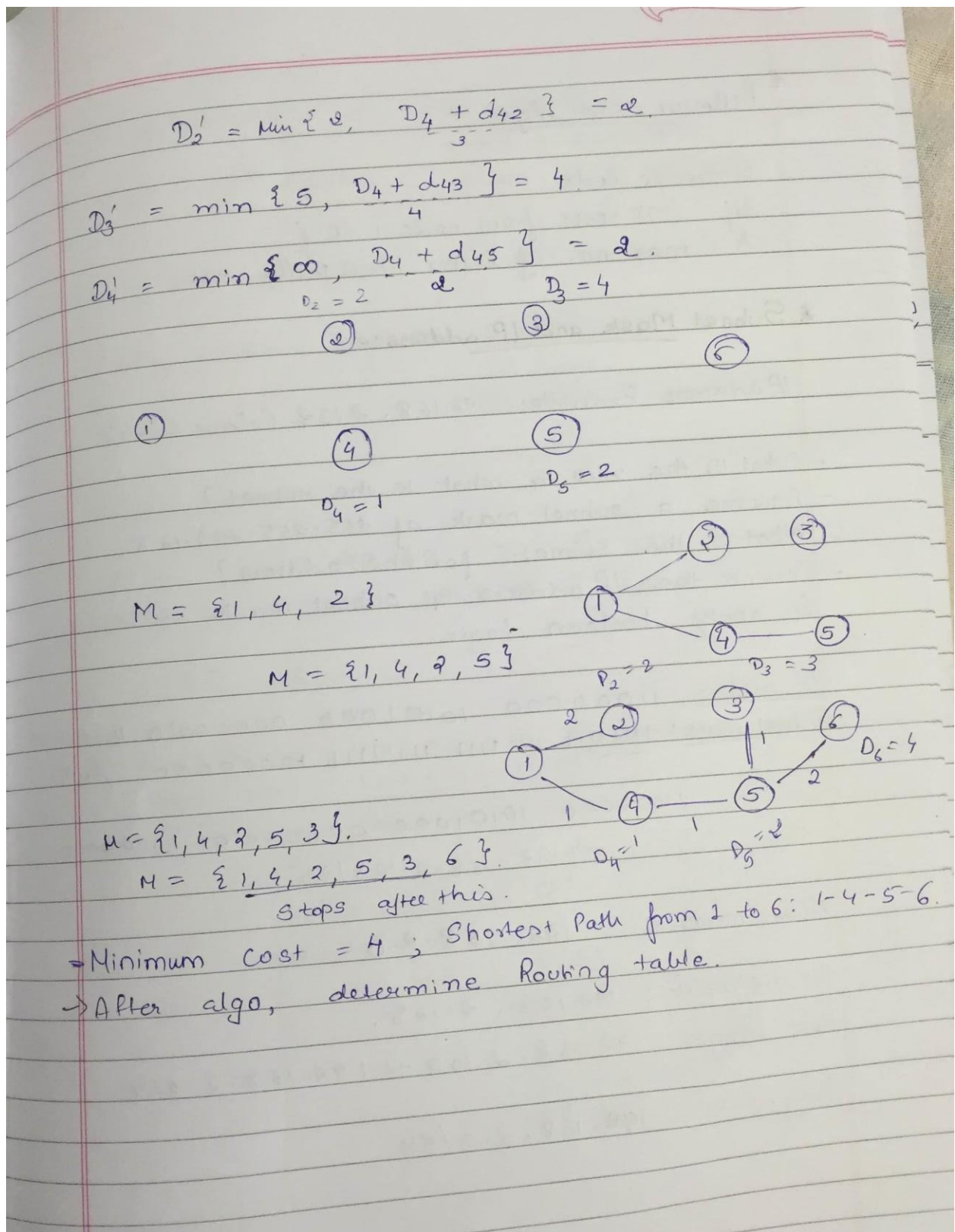


Figure 10: Final Spanning tree obtained using Dijkstra's Algo and shortest path between node 1 and 6.