Dynamic Routing Simulation

CPE 400

Mason Harlan

Engin Arslan

Features & Implementation

The dynamic routing simulation was implemented in JavaScript and HTML designed specifically for use in Google Chrome. The basis for the simulation uses the SPF algorithm to calculate forwarding tables for each node. There are two modes included in the project: broadcast and instant. Broadcast can be run and accessed from broadcast/index.html and instant can be run and accessed from original/index.html. On loading either simulation, the user will be met with a blank simulation area and an add button in the top left. To get started, press the add button. This will add a node to the simulation area. To view details about the node, click the node once, this will open that node's menu. The user can add as many nodes as they want. The user can also move these nodes by clicking and dragging the nodes in the simulation area. In the node menu, the user can delete the node, add connections to other nodes, or disable the node. To add connections the user can click on connect and then click on the node that they wish to link. Once a connection has been added, the user can change the weight of the link by entering it into the corresponding box or by click the arrows in the weight box for that link. This updated weight will change on both nodes in the link. To view the forwarding table for a node, click the right arrow next to **Links:**. To return to the links table, click the left arrow next to **Paths:**. These two tables will be updated automatically as the user adds nodes, deletes nodes, adds links, deletes links, disables links, and disables nodes. In the broadcast version of the simulation, the user can press F12 and navigate to the console to see a detailed read out of when a disabled node was detected in the network.

Broadcast & Instant

In the instant version of the simulation, it is assumed that the network detects that a node/link is down and broadcasts this information to the entire network automatically. Whenever a node/link is disabled or enabled the forwarding tables and path information on all nodes in the simulation are updated almost instantly. In the broadcast version of the simulation, the assumption is different. It is assumed that there is a constant flow of packets between all connected nodes in the simulation. When a node is disabled, it takes a neighboring node about 1 second to realize that its packets are being dropped and it is not receiving any information from the disabled nodes. At this point it broadcasts that the node is down. It will continue sending test packets to that node until it receives a response. When it does receive a valid response, it will broadcast to all other nodes that the node is enabled and valid again. When a node receives one of these broadcast messages it will recalculate its forwarding table. In this implementation, the link detection was not implemented. Due to the initial implementation and time constraints, only the node failure detection was implemented as a broadcast message. Link failure detection will be described below as part of the dynamic routing protocol.

Dynamic Protocol

When a node has not received any acknowledgements or traffic from a neighboring node after a set amount of time, the node will broadcast a message across the network. Before doing this, it will calculate its forwarding table again to see if the unresponsive node is reachable via another route. If the node is reachable via another route, it will update its forwarding table and send out a broadcast message informing the network that all nodes need to update their forwarding table to not include the downed link. If it is not reachable this means the node might be disabled and a similar broadcast message is issued. The node that detected the possible failure will continue to poll the invalidated node. If the invalidated node sends traffic back to the network, a new broadcast message informing the network that the invalidated node is reachable again will be sent. When a node receives this message, it will recalculate its forwarding table with the assumption that the invalidated node/link is usable again.

This protocol could be expensive in regard to time and does not guarantee finding all faulty or restored nodes. For example, if a node is restored and no traffic enters that area of the network, that area of the network would never know that service had been restored. One way to get around this is to define in the protocol a specific response that is needed from the polling packets sent by the node that originally detected the unresponsive node. This could increase congestion in the network and require extra work for the node that detected the failure. Network congestion is probably the biggest downside to this protocol. Depending on the timeout time and the number of failed networks, several broadcast messages and unreceived packets will be flowing through the network. Reducing the timeout time might seem like it would reduce this congestion since less unreceived packets would be sent in the network destined for the unresponsive node. However, there is a trade off between the broadcast messages and the lost packets. If the timeout time is reduced, then more nodes might be detected as unresponsive by mistake since they have a shorter window to keep alive in. This would lead to more broadcast messages and congestion in the network. Ideally, there would need to be a balanced timeout time. Another point to consider in this protocol is if a broadcast message does not make it to one of the nodes in the network. In that case, the node would continue sending traffic to the unresponsive node and would increase congestion in the network. One possible strategy to mitigate this would be to store a table of unresponsive nodes in all nodes in the network. When one of these nodes sees traffic going to an unresponsive node it will drop that packet and send a message back to the sender similar to the broadcast message. This message would inform that node to stop sending traffic to the unresponsive node. Similarly, this could happen if an unresponsive node is restored and some nodes in the network never find out about it. The greedy solution would be to have them poll that node themselves to discover if it is responsive again or not. This is not a good solution and can be expensive. A better solution would be similar to the solution above. Neighboring nodes would be able to determine if the unresponsive node is up again if it has received an up broadcast or one of their neighbors have received one. They would send this information out to its neighbors. This strategy is very optimistic and would require that the network would be okay with working together to meet a common goal. In reality, people might take advantage of this protocol and dramatically increase the traffic in the network.

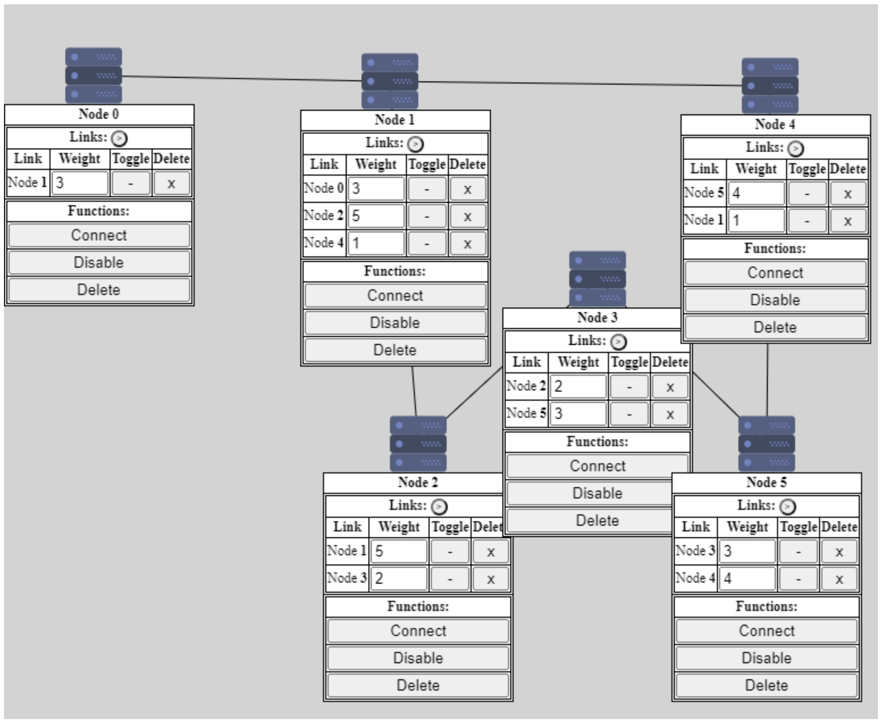
Another difficulty of this protocol is that it will not be able to determine just by the packet timeout whether there was a link failure or a node failure. This can lead to a large increase in detection time and possible errors. For example, if a node becomes seemingly unresponsive, there may be another route to this node and that needs to be calculated first before determining whether the node is unresponsive or only the link. Based on timing of this event it could lead to errors. A neighboring node to the disabled node might continue to advertise that the link is valid which would cause the original node to believe that there was a link failure instead of a node failure. This would eventually resolve over time since the other neighboring node would detect that its unreachable and inform the rest of the network. This sacrifices time but adds more confidence in disabled node/link detection.

Test Cases

Below are some test cases that show the functionality of the system in action and how the protocol acts differently in an ideal situation (instant) vs a more realistic scenario (broadcast).

Case 1

Figure 1A shows a simple network that contains a cycle. Its link weights can be seen in Figure 1A and the forwarding tables for select node can be seen in Figure 1B. This network was modeled in both the instant version and the broadcast version of the simulation.



Node 0

Links:

Link	Weight	Toggle	Delete
Node 1	3	-	x

Node 1

Links:

Link	Weight	Toggle	Delete
Node 0	3	-	x
Node 2	5	-	x
Node 4	1	-	x

Node 4

Links:

Link	Weight	Toggle	Delete
Node 5	4	-	x
Node 1	1	-	x

Node 3

Links:

Link	Weight	Toggle	Delete
Node 2	2	-	x
Node 5	3	-	x

Node 2

Links:

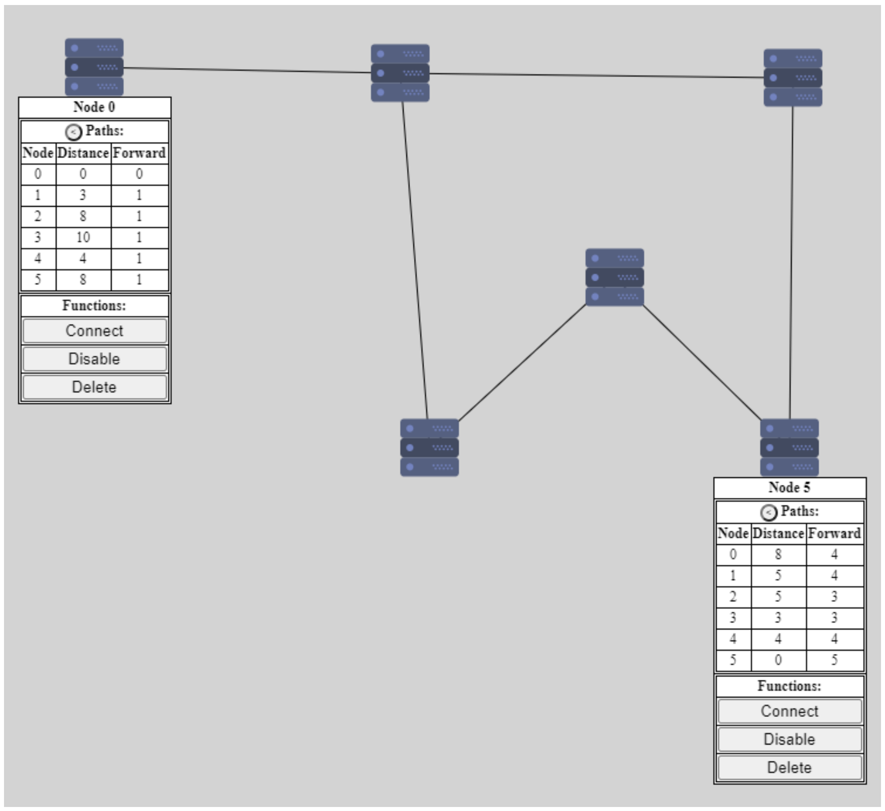
Link	Weight	Toggle	Delet
Node 1	5	-	x
Node 3	2	-	x

Node 5

Links:

Link	Weight	Toggle	Delete
Node 3	3	-	x
Node 4	4	-	x

Figure 1A: A simple network that demonstrates a cycle.



Node 0

Node	Distance	Forward
0	0	0
1	3	1
2	8	1
3	10	1
4	4	1
5	8	1

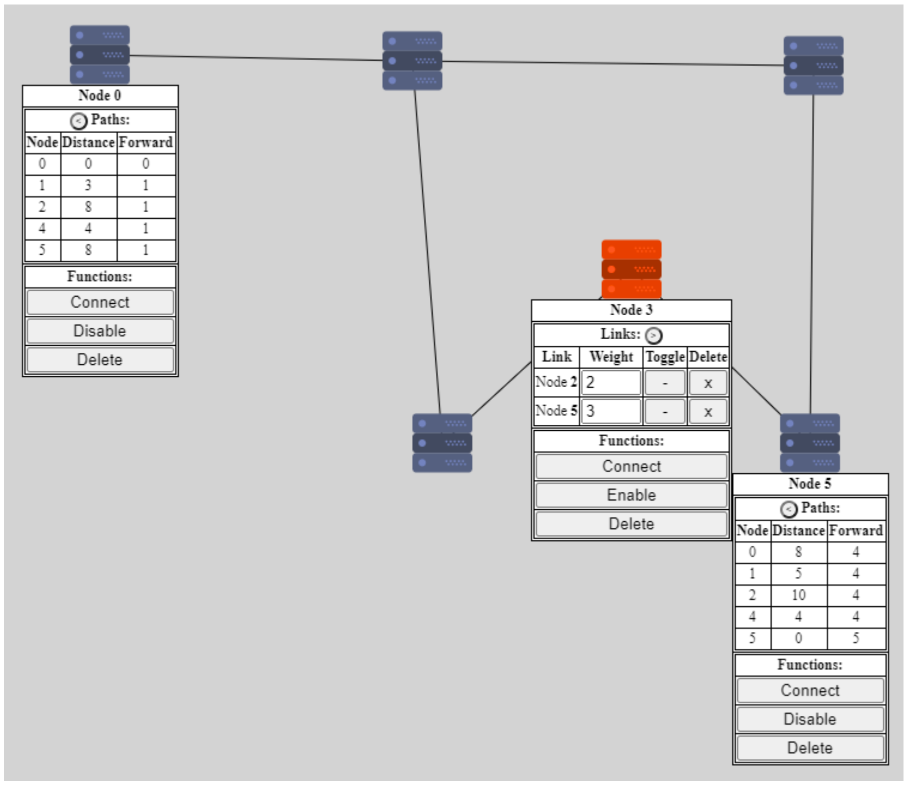
Node 5

Node	Distance	Forward
0	8	4
1	5	4
2	5	3
3	3	3
4	4	4
5	0	5

Figure 1B: A simple network that demonstrates a cycle.

The forwarding tables for Node 0 and Node 5 can be seen.

Figure 1C shows what the forwarding tables of some of the nodes might look like if Node 3 were disabled in the instant simulation. Similarly, Figure 1E shows what the forwarding tables would look like if Node 3 were disabled in the broadcast simulation. Figure 1F shows the output log in the console. There are some subtle differences between these two figures. As we can see from Figure 1F, it took time for the nodes in the network to become aware of the outage in the network. The forwarding table for all of these nodes was not updated until the broadcast message was received, and the nodes were able to recalculate the shortest available path. This increases significant delay in comparison to the instant version where the tables are updated instantly. The broadcast simulation provides a more realistic view of how long it will take for the dynamic protocol to inform neighboring nodes that there is a faulty node in the network. For example, Figure 1F shows that it took about 4 clock ticks (154 – 150) for the broadcasted message to reach Node 0. During this time, Node 0 could have been routing packets to the disabled nodes and those packets are pointless. They congest the network and will be lost eventually. Similarly, it takes about 4 clock ticks for Node 0 to receive the broadcast message that the node has been reenabled. This is wasted time that Node 0 could have spent routing traffic to the reenabled node. This broadcast protocol is clearly slower than the ideal instant version of the simulation which has no delay. In the instant version, no packets would be lost, and no time would be wasted.



Node 0

Paths:

Node	Distance	Forward
0	0	0
1	3	1
2	8	1
4	4	1
5	8	1

Functions:

Node 3

Links:

Link	Weight	Toggle	Delete
Node 2	2	-	x
Node 5	3	-	x

Functions:

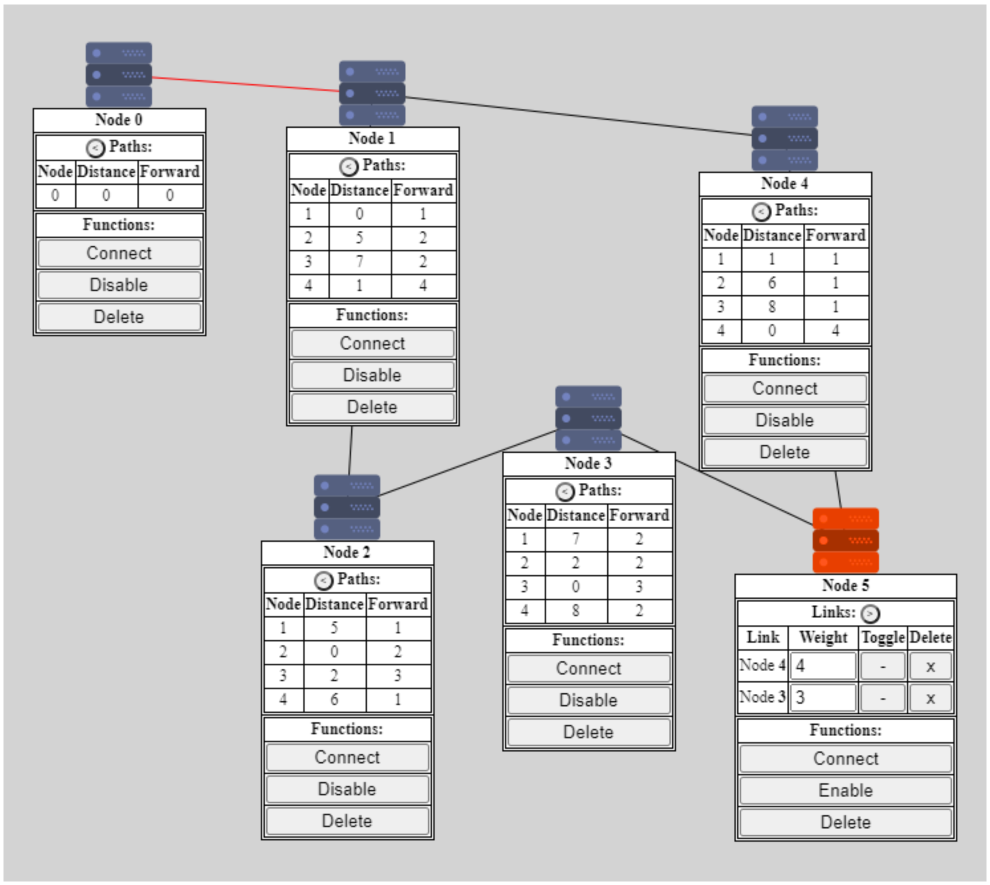
Node 5

Paths:

Node	Distance	Forward
0	8	4
1	5	4
2	10	4
4	4	4
5	0	5

Functions:

Figure 1C: The network from Figure 1A but with Node 3 disabled in the instant simulation.



Node 0

Paths:

Node	Distance	Forward
0	0	0

Node 1

Paths:

Node	Distance	Forward
1	0	1
2	5	2
3	7	2
4	1	4

Node 4

Paths:

Node	Distance	Forward
1	1	1
2	6	1
3	8	1
4	0	4

Node 3

Paths:

Node	Distance	Forward
1	7	2
2	2	2
3	0	3
4	8	2

Node 2

Paths:

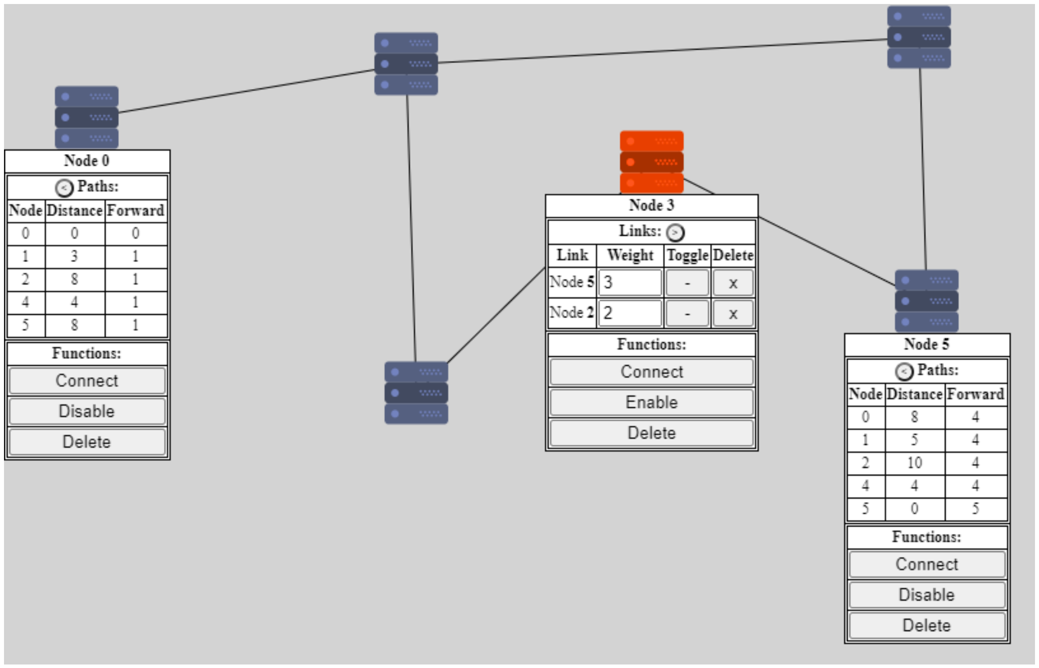
Node	Distance	Forward
1	5	1
2	0	2
3	2	3
4	6	1

Node 5

Links:

Link	Weight	Toggle	Delete
Node 4	4	-	x
Node 3	3	-	x

Figure 1D: The network from Figure 1A but with a link and Node 5 disabled in the instant simulation.



Node 0

Paths:

Node	Distance	Forward
0	0	0
1	3	1
2	8	1
4	4	1
5	8	1

Node 3

Links:

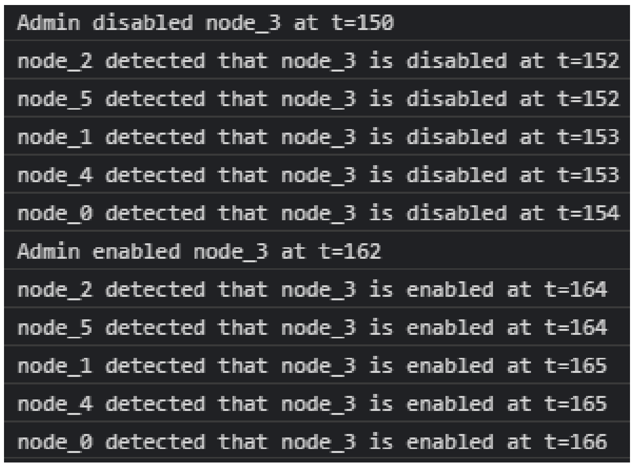
Link	Weight	Toggle	Delete
Node 5	3	-	x
Node 2	2	-	x

Node 5

Paths:

Node	Distance	Forward
0	8	4
1	5	4
2	10	4
4	4	4
5	0	5

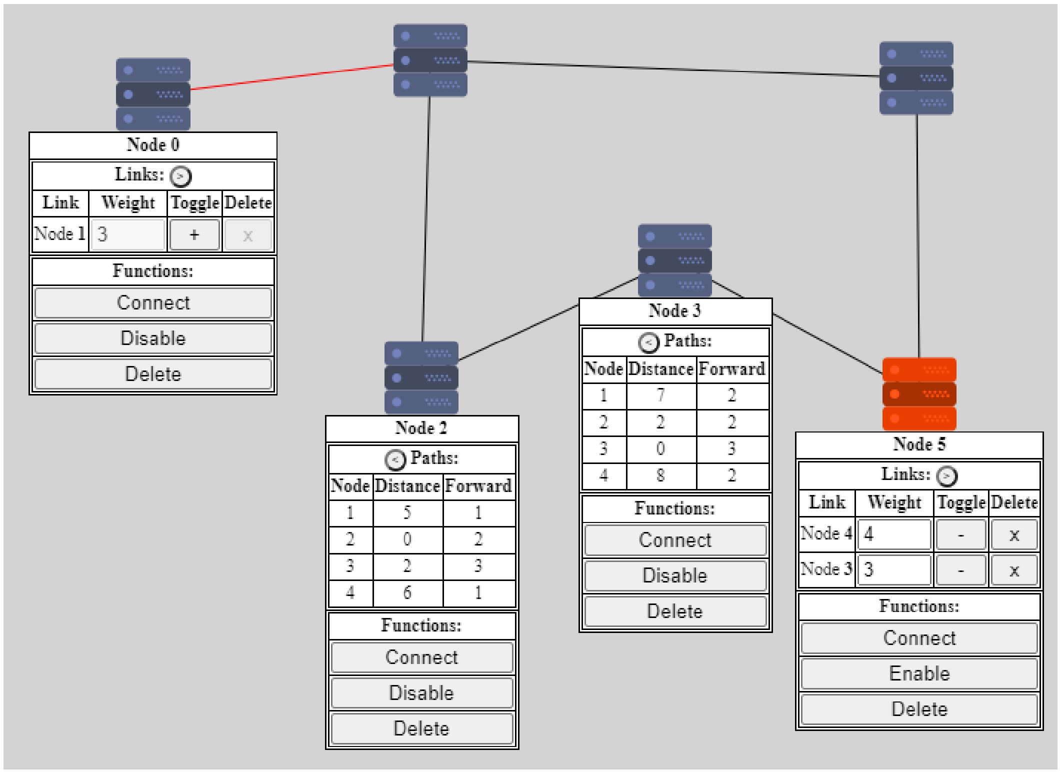
Figure 1E: The network from Figure A with Node 3 disabled in the broadcast simulation.



```
Admin disabled node_3 at t=150
node_2 detected that node_3 is disabled at t=152
node_5 detected that node_3 is disabled at t=152
node_1 detected that node_3 is disabled at t=153
node_4 detected that node_3 is disabled at t=153
node_0 detected that node_3 is disabled at t=154
Admin enabled node_3 at t=162
node_2 detected that node_3 is enabled at t=164
node_5 detected that node_3 is enabled at t=164
node_1 detected that node_3 is enabled at t=165
node_4 detected that node_3 is enabled at t=165
node_0 detected that node_3 is enabled at t=166
```

Figure 1F: The logged information from disabling and enabling Node 3 as shown in Figure 1E.

Figure 1D and Figure 1G show another faulty network in the instant and broadcast simulations, respectively. In this version of the network, a link and Node 5 are disabled. The forwarding tables can be seen in the Figures for each simulation. Similar to above, it takes some time before the forwarding tables can be updated in Figure 1G. This is because it takes the neighboring nodes time to realize that Node 5 is down (in this case 1 clock tick) and to broadcast the message to other nodes instructing them to recalculate their forwarding tables. Node 0 is cut off by a faulty link from the rest of the network, so it will not receive the broadcasted message. The logged message activity can be seen in Figure 1H. Figure 1H shows that it took about 3 clock ticks for the message to reach some of the nodes in the network. Again, this time delay is much larger than the optimistic scenario as implemented in the instant simulation.



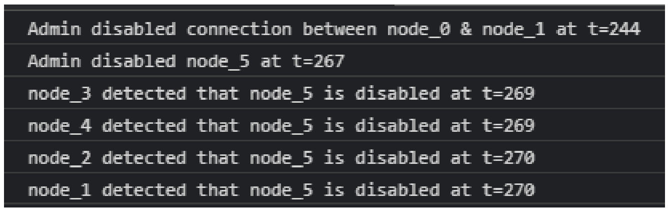
Link	Weight	Toggle	Delete
Node 1	3	+	x

Node	Distance	Forward
1	5	1
2	0	2
3	2	3
4	6	1

Node	Distance	Forward
1	7	2
2	2	2
3	0	3
4	8	2

Link	Weight	Toggle	Delete
Node 4	4	-	x
Node 3	3	-	x

Figure 1G: The network from Figure A with a link and Node 5 disabled in the broadcast simulation.

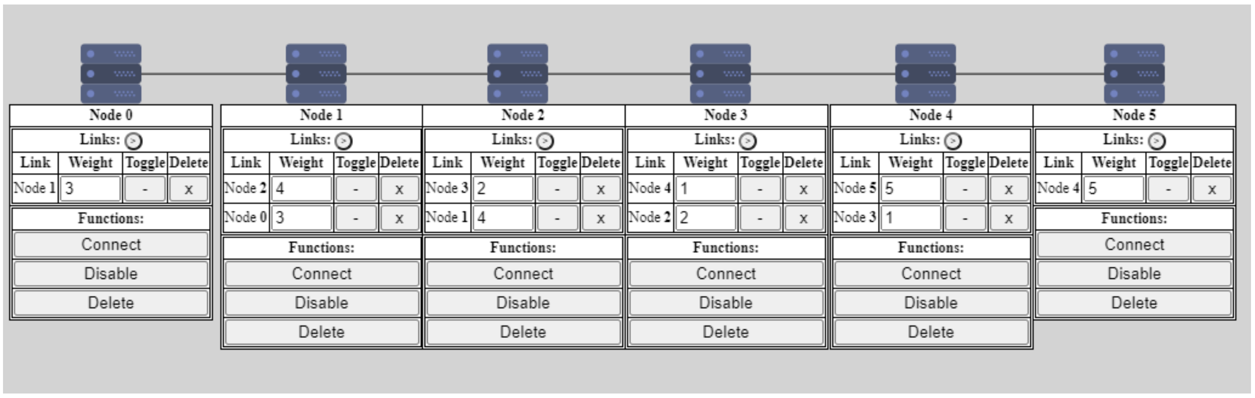


```
Admin disabled connection between node_0 & node_1 at t=244
Admin disabled node_5 at t=267
node_3 detected that node_5 is disabled at t=269
node_4 detected that node_5 is disabled at t=269
node_2 detected that node_5 is disabled at t=270
node_1 detected that node_5 is disabled at t=270
```

Figure 1H: The logged information from disabling Node 5 as shown in Figure 1G.

Case 2

Figure 2A shows a simple network that contains a long chain of nodes. Its link weights can be seen in Figure 2A and the forwarding tables for each node can be seen in Figure 2B. This network was modeled in both the instant version and the broadcast version of the simulation.



Node 0

Links:

Link	Weight	Toggle	Delete
Node 1	3	-	x

Node 1

Links:

Link	Weight	Toggle	Delete
Node 2	4	-	x
Node 0	3	-	x

Node 2

Links:

Link	Weight	Toggle	Delete
Node 3	2	-	x
Node 1	4	-	x

Node 3

Links:

Link	Weight	Toggle	Delete
Node 4	1	-	x
Node 2	2	-	x

Node 4

Links:

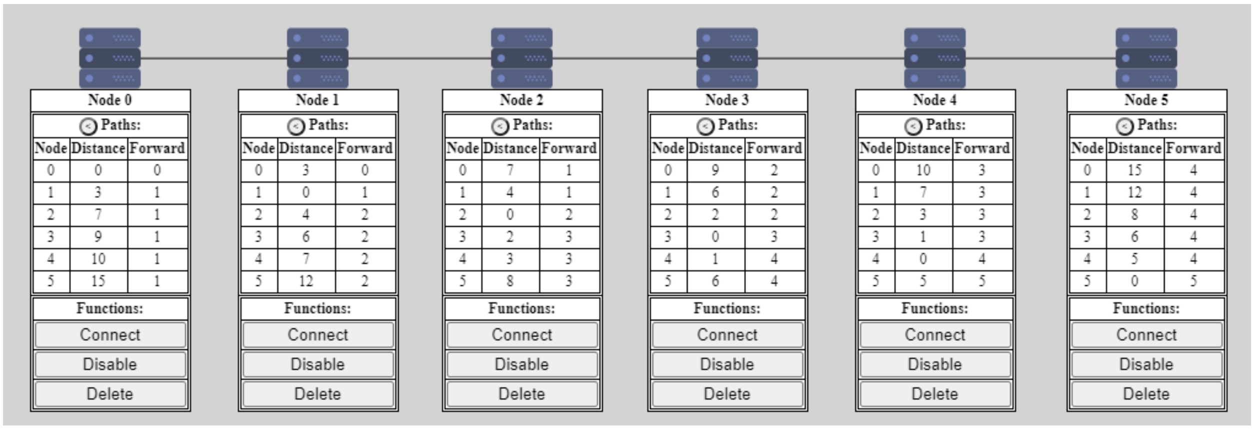
Link	Weight	Toggle	Delete
Node 5	5	-	x
Node 3	1	-	x

Node 5

Links:

Link	Weight	Toggle	Delete
Node 4	5	-	x

Figure 2A: A simple network that demonstrates a chain of nodes.



Node 0

Paths:

Node	Distance	Forward
0	0	0
1	3	1
2	7	1
3	9	1
4	10	1
5	15	1

Node 1

Paths:

Node	Distance	Forward
0	3	0
1	0	1
2	4	2
3	6	2
4	7	2
5	12	2

Node 2

Paths:

Node	Distance	Forward
0	7	1
1	4	1
2	0	2
3	2	3
4	3	3
5	8	3

Node 3

Paths:

Node	Distance	Forward
0	9	2
1	6	2
2	2	2
3	0	3
4	1	4
5	6	4

Node 4

Paths:

Node	Distance	Forward
0	10	3
1	7	3
2	3	3
3	1	3
4	0	4
5	5	5

Node 5

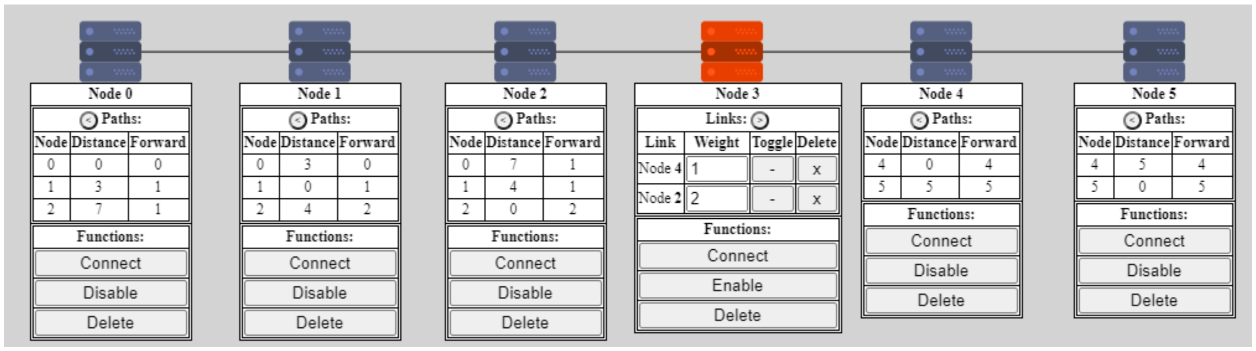
Paths:

Node	Distance	Forward
0	15	4
1	12	4
2	8	4
3	6	4
4	5	4
5	0	5

Figure 2B: A simple network that demonstrates a chain of nodes.

The forwarding tables for all nodes can be seen.

Figure 2C shows what the forwarding tables will look like if Node 3 were disabled in the instant simulation. Similarly, Figure 2E shows what the forwarding tables would look like if Node 3 were disabled in the broadcast simulation. Figure 2F shows the output log in the console. This figure also illustrates how long the delay of the dynamic routing protocol can be in the case of a network with a long chain. In this network, it takes about 1 clock tick for the neighboring nodes to discover that Node 3 is down and about 2 more clock ticks for the broadcast to reach all nodes in the chain. If this chain were longer, the time could be even more significant. The forwarding table for all nodes in the network were not updated until the broadcast message was received, and the nodes were able to recalculate the shortest available path to the disabled nodes. In this case there was no alternative shortest path, so the nodes removed the disabled node from their forwarding tables. This increases significant delay in comparison to the instant version where the tables are updated instantly. It is clear that a broadcast solution is slower than the ideal instant version of the simulation which has no delay.


Node 0

Paths:

Node	Distance	Forward
0	0	0
1	3	1
2	7	1

Node 1

Paths:

Node	Distance	Forward
0	3	0
1	0	1
2	4	2

Node 2

Paths:

Node	Distance	Forward
0	7	1
1	4	1
2	0	2

Node 3

Links:

Link	Weight	Toggle	Delete
Node 4	1	-	x
Node 2	2	-	x

Node 4

Paths:

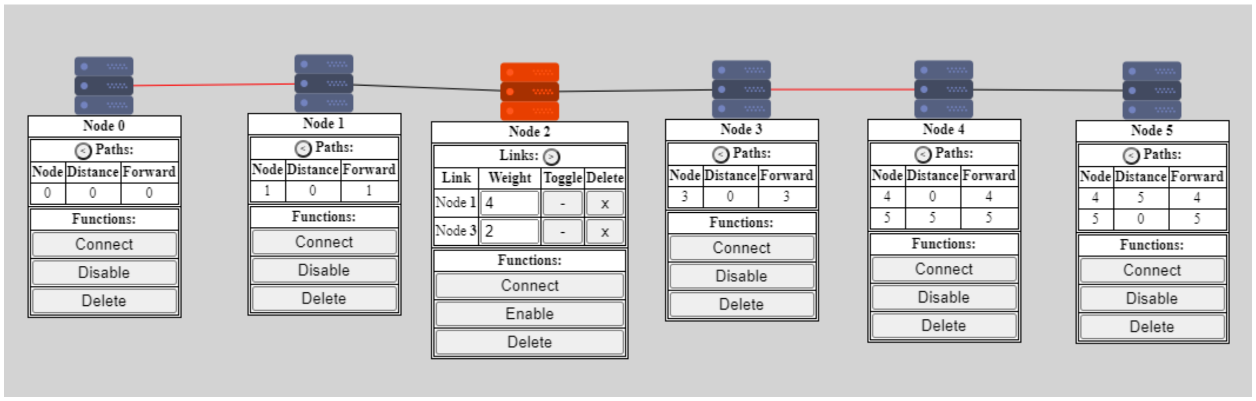
Node	Distance	Forward
4	0	4
5	5	5

Node 5

Paths:

Node	Distance	Forward
4	5	4
5	0	5

Figure 2C: The network from Figure 2A but with Node 3 disabled in the instant simulation.


Node 0

Paths:

Node	Distance	Forward
0	0	0

Node 1

Paths:

Node	Distance	Forward
1	0	1

Node 2

Links:

Link	Weight	Toggle	Delete
Node 1	4	-	x
Node 3	2	-	x

Node 3

Paths:

Node	Distance	Forward
3	0	3

Node 4

Paths:

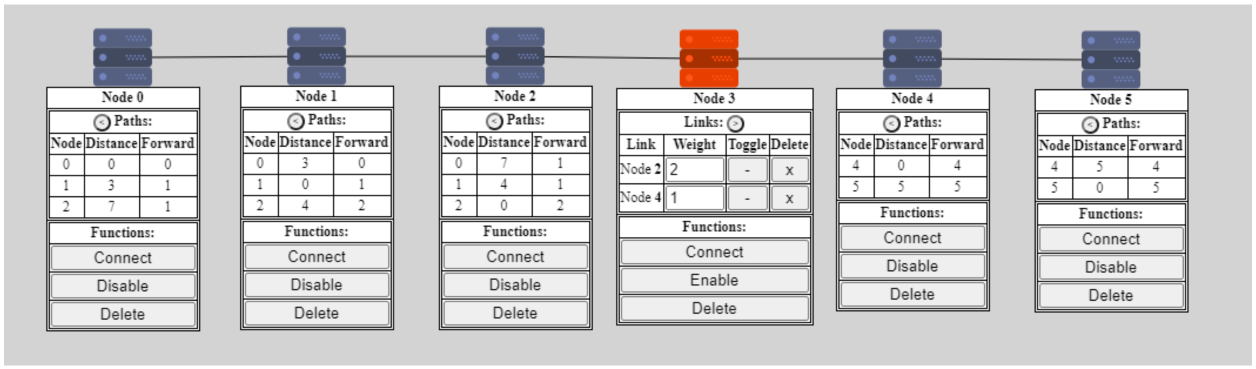
Node	Distance	Forward
4	0	4
5	5	5

Node 5

Paths:

Node	Distance	Forward
4	5	4
5	0	5

Figure 2D: The network from Figure 2A but with 2 links and Node 2 disabled in the instant simulation.


Node 0

Paths:

Node	Distance	Forward
0	0	0
1	3	1
2	7	1

Node 1

Paths:

Node	Distance	Forward
0	3	0
1	0	1
2	4	2

Node 2

Paths:

Node	Distance	Forward
0	7	1
1	4	1
2	0	2

Node 3

Links:

Link	Weight	Toggle	Delete
Node 2	2	-	x
Node 4	1	-	x

Node 4

Paths:

Node	Distance	Forward
4	0	4
5	5	5

Node 5

Paths:

Node	Distance	Forward
4	5	4
5	0	5

Figure 2E: The network from Figure 2A with Node 3 disabled in the broadcast simulation

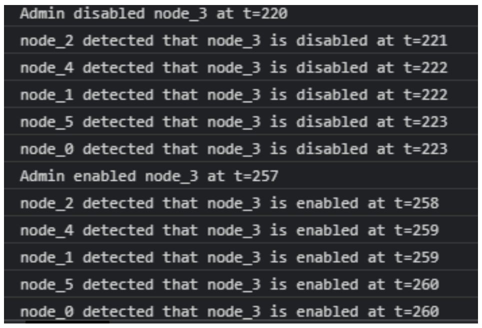
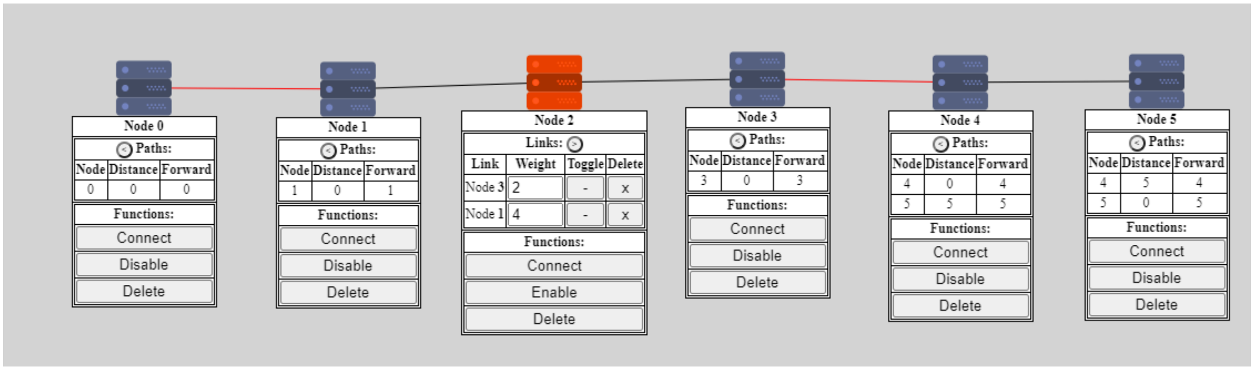

```
Admin disabled node_3 at t=220
node_2 detected that node_3 is disabled at t=221
node_4 detected that node_3 is disabled at t=222
node_1 detected that node_3 is disabled at t=222
node_5 detected that node_3 is disabled at t=223
node_0 detected that node_3 is disabled at t=223
Admin enabled node_3 at t=257
node_2 detected that node_3 is enabled at t=258
node_4 detected that node_3 is enabled at t=259
node_1 detected that node_3 is enabled at t=259
node_5 detected that node_3 is enabled at t=260
node_0 detected that node_3 is enabled at t=260
```

Figure 2F: The logged information from disabling and enabling Node 3 as shown in Figure 2E.

Figure 2D and Figure 2G show another faulty network in the instant and broadcast simulations, respectively. In this version of the network, 2 links and Node 2 are disabled. The forwarding tables can be seen in the Figures for each simulation. Similar to above, it takes some time before the forwarding tables are updated as seen in Figure 2G. This is because it takes the neighboring nodes of Node 2 some time to realize that its down (in this case 1 clock tick) and to broadcast the message to other nodes instructing them to recalculate their forwarding tables. Node 0 is cut off by a faulty link from the rest of the network, so it will not receive the broadcasted message. Node 4 and Node 5 are also cut off by a faulty link and will not receive the broadcast message either. The logged message activity can be seen in Figure 1H. Figure 1H shows that it took about 1 clock tick for Node 1 and Node 3 to realize Node 2 was down. In this case no broadcast messages were sent to the network because Node 1 and Node 3 do not have any valid neighbors to broadcast about the failure. In this case, the time difference between instant and broadcast was not as large; however, it is still significant. In the dynamic routing protocol, it takes time to detect if packets are lost and depending on the timeout time set in the implementation of the protocol (1 clock tick in the broadcast simulation), it could be a long delay before the rest of the network learns about a failure.



Node 0 — Paths:

Node	Distance	Forward
0	0	0

Node 1 — Paths:

Node	Distance	Forward
1	0	1

Node 2 — Links:

Link	Weight	Toggle	Delete
Node 3	2	-	x
Node 1	4	-	x

Node 3 — Paths:

Node	Distance	Forward
3	0	3

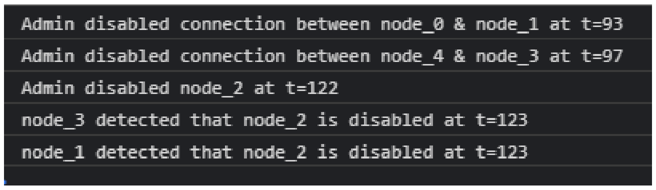
Node 4 — Paths:

Node	Distance	Forward
4	0	4
5	5	5

Node 5 — Paths:

Node	Distance	Forward
4	5	4
5	0	5

Figure 2G: The network from Figure 2A but with 2 links and Node 2 disabled in the instant simulation.



```
Admin disabled connection between node_0 & node_1 at t=93
Admin disabled connection between node_4 & node_3 at t=97
Admin disabled node_2 at t=122
node_3 detected that node_2 is disabled at t=123
node_1 detected that node_2 is disabled at t=123
```

Figure 2H: The logged information from disabling Node 2 as shown in Figure 2G.