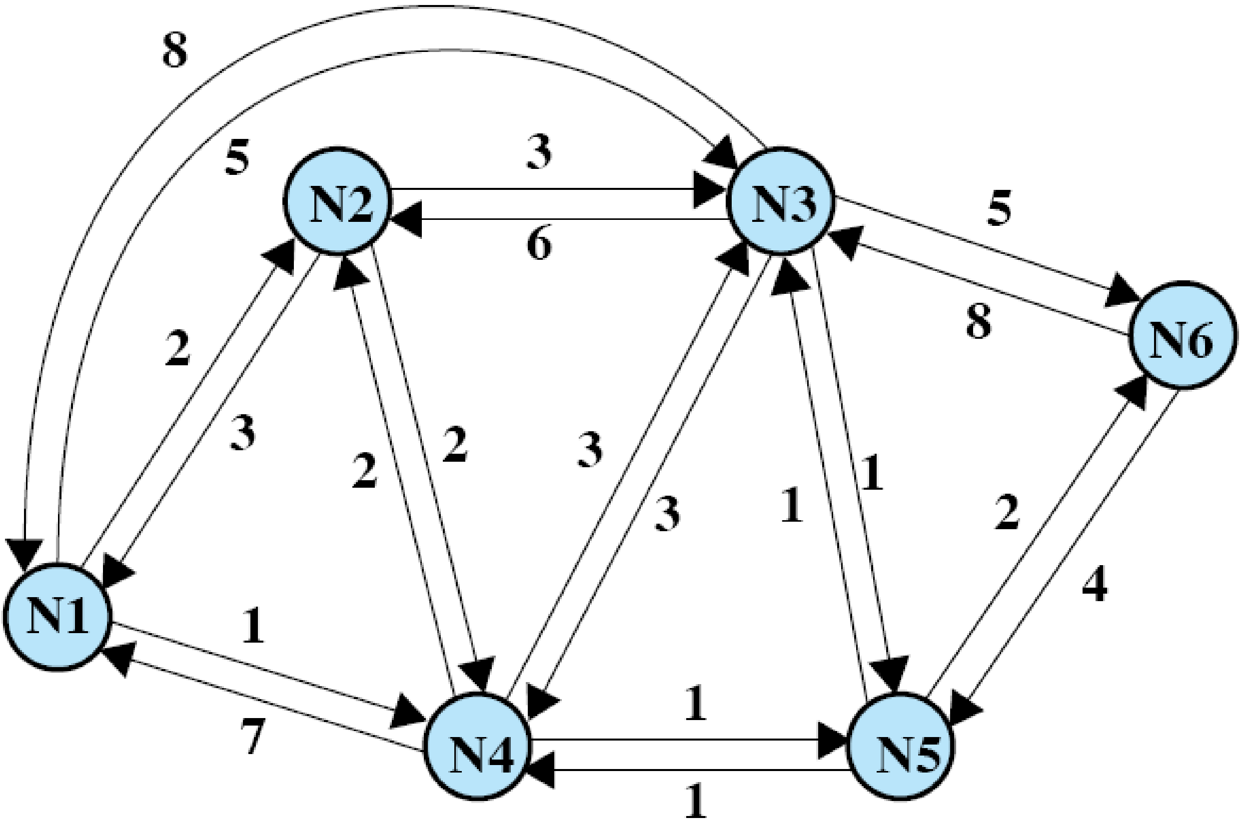
ITS 323 –Routing Practice Answers

1 Scenario

In the “Routing in Switched Networks” lecture notes we covered an example of “Link State Routing” for the network below.



In the lecture, we calculated the shortest path tree from node N1. That is, the least-cost paths from N1 to N2, from N1 to N3, and so on. We used Dijkstra’s algorithm to calculated these paths.

2 Dijkstra’s Algorithm for Finding Least-Cost Paths

In the same network as above, use Dijkstra’s to calculate the least-cost paths from node N6. It is recommended you perform the steps manually, as in slide 38 of the lecture notes.

The answer that you should get (that is, the shortest paths), which you can check from the diagram above, is:

P(N1) = N6 →N5 →N4 →N2 →N1

P(N2) = N6 →N5 →N4 →N2

P(N3) = N6 →N5 →N3

P(N4) = N6 →N5 →N4

P(N5) = N6 →N5

2.1 Answer

Iteration 1

Step 1 [Initialisation]

N = {N1, N2, N3, N4, N5, N6}

T = {N6}

We determine the least-cost paths using only the links from node N6:

L(N1) = ∞ P(N1) = -

L(N2) = ∞ P(N2) = -

L(N3) = 8 P(N3) = N6 →N3

L(N4) = ∞ P(N4) = -

L(N5) = 4 P(N5) = N6 →N5

Summary of results after iteration 1:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5

Iteration 2

Step 2 [Get Next Node]

The node (which is not yet in T) with the lowest cost is N5.

x = N5

T = {N6, N5}

Step 3 [Update Least-Cost Paths]

L(N1) = min [∞, L(N5) + w(N5, N1)]
= min [∞, 4 + ∞]
= ∞ (no change, so path P(N1) remains same)

L(N2) = min [∞, L(N5) + w(N5, N2)]
= min [∞, 4 + ∞]
= ∞ (no change, so path P(N2) remains same)

L(N3) = min [8, L(N5) + w(N5, N3)]
= min [8, 4 + 1]
= 5

The alternative path (via N5) gives a new lower cost path, and hence P(N3) is updated:

P(N3) = P(N5) + link(N5,N3)
= N6 →N5 →N3

L(N4) = min [∞, L(N5) + w(N5, N4)]
= min [∞, 4 + 1]
= 5

The alternative path (via N5) gives a new lower cost path, and hence P(N4) is updated:

P(N4) = P(N5) + link(N5,N4)
= N6 →N5 →N4

Summary of results after iteration 2:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5
2	6,5	∞	-	∞	-	5	6-5-3	5	6-5-4	4	6-5

Iteration 3

Step 2 [Get Next Node]

The node (which is not yet in T) with the lowest cost is N3 (or N4 – I choose N3).

x = N3

T = {N6, N5, N3}

Step 3 [Update Least-Cost Paths]

L(N1) = min [∞, L(N3) + w(N3, N1)]
= min [∞, 5 + 8]
= 13 (changed, so update path)

P(N1) = P(N3) + link(N3,N1)
= N6 →N5 →N3 →N1

L(N2) = min [∞, L(N3) + w(N3, N2)]
= min [∞, 5 + 6]
= 11 (changed, so update path)

P(N2) = P(N3) + link(N3,N2)
= N6 →N5 →N3 →N2

L(N4) = min [5, L(N3) + w(N3, N4)]
= min [5, 5 + 3]
= 5 (no change)

Summary of results after iteration 2:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5
2	6,5	∞	-	∞	-	5	6-5-3	5	6-5-4	4	6-5
3	6,5,3	13	6-5-3-1	11	6-5-3-2	5	6-5-3	5	6-5-4	4	6-5

Iteration 4

Step 2 [Get Next Node]

The node (which is not yet in T) with the lowest cost is N4.

x = N4

T = {N6, N5, N3, N4}

Step 3 [Update Least-Cost Paths]

L(N1) = min [13, L(N4) + w(N4, N1)]

= min [13, 5 + 7]

= 12 (changed, so update path)

P(N1) = P(N4) + link(N4,N1)

= N6 →N5 →N4 →N1

L(N2) = min [11, L(N4) + w(N4, N2)]

= min [11, 5 + 2]

= 7 (changed, so update path)

P(N2) = P(N4) + link(N4,N2)

= N6 →N5 →N4 →N2

Summary of results after iteration 2:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5
2	6,5	∞	-	∞	-	5	6-5-3	5	6-5-4	4	6-5
3	6,5,3	13	6-5-3-1	11	6-5-3-2	5	6-5-3	5	6-5-4	4	6-5
4	6,5,3,4	12	6-5-4-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5

Iteration 5

Step 2 [Get Next Node]

The node (which is not yet in T) with the lowest cost is N2.

x = N2

T = {N6, N5, N3, N4, N2}

Step 3 [Update Least-Cost Paths]

L(N1) = min [12, L(N2) + w(N2, N1)]

= min [12, 7 + 3]

= 10 (changed, so update path)

P(N1) = P(N2) + link(N2,N1)

= N6 →N5 →N4 →N2 →N1

Summary of results after iteration 2:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5
2	6,5	∞	-	∞	-	5	6-5-3	5	6-5-4	4	6-5
3	6,5,3	13	6-5-3-1	11	6-5-3-2	5	6-5-3	5	6-5-4	4	6-5
4	6,5,3,4	12	6-5-4-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5
5	6,5,3,4,2	10	6-5-4-2-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5

Iteration 6

Step 2 [Get Next Node]

The node (which is not yet in T) with the lowest cost is N1.

x = N1

T = {N6, N5, N3, N4, N2, N1}

Step 3 [Update Least-Cost Paths]

Since all nodes are in T, there are no costs to update. We are finished, and now have the least cost paths.

Summary of results after iteration 2:

		N1		N2		N3		N4		N5	
Iter.	T	L	P	L	P	L	P	L	P	L	P
1	6	∞	-	∞	-	8	6-3	∞	-	4	6-5
2	6,5	∞	-	∞	-	5	6-5-3	5	6-5-4	4	6-5
3	6,5,3	13	6-5-3-1	11	6-5-3-2	5	6-5-3	5	6-5-4	4	6-5
4	6,5,3,4	12	6-5-4-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5
5	6,5,3,4,2	10	6-5-4-2-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5
6	6,5,3,4,2, 1	10	6-5-4-2-1	7	6-5-4-2	5	6-5-3	5	6-5-4	4	6-5

3 Routing Table for Node N6

From the least-cost paths you calculated above, give the routing table to be stored at node N6. (Note, if you did not complete the Dijkstra's algorithm, then you can still answer this question, because I gave you the paths).

3.1 Answer

Routing Table for N6:

Destination	Next Hop
N1	N5
N2	N5
N3	N5
N4	N5
N5	N5

4 Flooding in the Example Network

Link state routing requires Link State Packets (LSP) to be sent through the network in order for nodes to discover the network topology. What is the cost of Node N6 sending a single LSP through the network? Assume the hop count (limit) in the LSP is 4.

4.1 Answer

N6 sends to neighbours:

N6 $\rightarrow$N3: cost 8

N6 $\rightarrow$N5: cost 4 Total cost so far: 12

N3 sends to neighbours (except N6):

N3 $\rightarrow$N1: 8

N3 $\rightarrow$N2: 6

N3 $\rightarrow$N4: 3

N3 $\rightarrow$N5: 1 Total cost so far: 30

N5 sends to neighbours (except N6):

N5 $\rightarrow$N3: 1

N5 $\rightarrow$N4: 1 Total cost so far: 32

(We have assumed that N5 sends to N3 before N5 receives from N3, and vice versa – that is the transmissions from N3 to N5 and N5 to N3 occur at the same time).

N1 sends to neighbours (except N3):

N1 $\rightarrow$ N2: 2

N1 $\rightarrow$ N4: 1 Total cost so far: 35

N2 sends to neighbours (except N3):

N2 $\rightarrow$ N1: 3

N2 $\rightarrow$ N4: 2 Total cost so far: 40

N4 sends to neighbours (except N3 and N5):

N4 $\rightarrow$ N1: 7

N4 $\rightarrow$ N2: 2 Total cost so far: 49

(Again we assumed N1 sends to N2 at the same time as N2 sends to N1, and similar for N4). Now each node has received a copy of the LSP, and hence no more copies will be sent.

Total cost of flooding the LSP from N6 is 49.