

Routing, Multicast, and Broadcast

1. LS and BGP

- (a) Referring to Figure 1, compute the least-cost paths from router a to all the other routers in AS 23 using Dijkstra's algorithm. Use a table similar to Table 4.3 on p.356 of the textbook to record your work. Here N' represents the set of routers for which the least-cost path is known, $D(v)$ represents the current best guess for distance to node v and $p(v)$ is the predecessor of v on the least-cost path towards it.

step	N'	$D(b),p(b)$	$D,p(c)$	$D,p(d)$	$D,p(e)$	$D,p(f)$	$D,p(g)$	$D,p(h)$	$D,p(i)$	$D,p(j)$
0	a	2,a	3,a	∞	∞	1,a	∞	∞	2,a	∞
1	af	2,a	3,a	∞	7,f		5,f	∞	2,a	∞
2	afb		3,a	9,b	6,b		5,f	∞	2,a	∞
3	afbi		3,a	9,b	6,b		5,f	11,i		6,i
4	afbic			8,c	6,b		5,f	11,i		6,i
5	afbicg			8,c	6,b			7,g		6,i
6	afbicge			8,c				7,g		6,i
7	afbicgej			8,c				7,g		
8	afbicgejh			8,c						
9	afbicgejhd									

- (b) Suppose that the gateway routers of AS 23 receive the following BGP advertisements from their BGP peers:

Network	AS Path
AS 56	
1.2.3.0/24	56 83 99
1.3.8.0/23	56 75
1.4.8.5/24	56 97
AS 83	
1.7.128.0/17	83 117
1.4.8.4/24	83 62 103
AS 88	
2.3.0.0/16	88 107 56 23
2.7.9.0/24	88 107
AS 45	
7.5.8.0/22	45
7.12.0.0/16	45 75 23
1.2.3.0/24	45 99

Show the routing table in router a that will be formed as a result of these advertisements. Assume that no routes are rejected due to local policy rules. Represent the routing table in this form:

Network	Interface
5.5.5.0/24	if1

Network	Interface	
1.3.8.0/23	if1	
1.4.8.4/23	if1	(aggregated prefix from AS56 and AS83)
1.7.128.0/17	if1	
2.7.9.0/24	if2	
7.5.8.0/22	if4	(AS45 reached through router j due to hot potato routing)
7.12.0.0/16	if4	
1.2.3.0/24	if4	(AS45 has shorter AS PATH for this network than AS56)

2. Distance Vector Routing

- (a) Suppose AS 23 from the last question runs a synchronous version of a distance vector algorithm. Every minute, at exactly the same time, each router sends its current distance vector to its neighbors. Then it updates its own vector based on what it receives, and sends out the updated vector the next minute.

Starting all routers in an initial state where they only know the cost of reaching their neighbors (i.e. the distance to all other routers is infinity), show the updated distance vector table at each router after the first exchange of distance vectors.

Each row in this table represents the distance vector table at a given router.

		cost to									
		a	b	c	d	e	f	g	h	i	j
from	a	0	2	3	8	6	1	5	11	2	6
	b	2	0	5	7	4	3	∞	∞	4	∞
	c	3	5	0	5	∞	4	∞	∞	5	∞
	d	8	7	5	0	11	∞	∞	∞	∞	∞
	e	6	4	∞	11	0	6	10	∞	10	∞
	f	1	3	4	∞	6	0	4	6	3	8
	g	5	∞	∞	∞	10	4	0	2	8	5
	h	11	∞	∞	∞	∞	6	2	0	7	3
	i	2	4	5	∞	10	3	8	7	0	4
	j	6	∞	∞	∞	∞	8	5	3	4	0

- (b) How many minutes will it take for the routes to stabilize? Can you characterize how many rounds the DV algorithm will take to converge on an arbitrary graph?

It will take 4 minutes for the routes to stabilize. Observe that after k exchanges, the DV algorithm has considered all paths of length $k + 1$. The least-cost path between d and h is 5 hops long, and it will be discovered after 4 minutes. All other least-cost paths between two nodes are shorter than 5 hops, so after 4 minutes the routes will stabilize.

For a general graph, the number of rounds is one less than the length of the longest least-cost path between two vertices (i.e. out of all least-cost paths between two vertices, the one that is the longest). Note that this is not necessarily the diameter of the graph; for example, the diameter of the graph in Figure 1 is 4.

- (c) Referring to Figure 2, suppose the cost of the link between a and b changes to 100. Assuming synchronous updates once again, write down the distance vector table changes in routers b , c , and d for the next 5 minutes after the change. Assume that poison reverse is used in this case. How many rounds will it take for the routes to stabilize?

It is easy to see that the only distance vector entries that will change will be those representing the distance to a . Here is how the first five minutes will look. The table records the current distance and the next hop recorded in each router for a .

minute	b	c	d
0	8,d	2,b	4,c
1	8,d	9,b	4,c
2	8,d	9,b	11,c
3	15,d	9,b	11,c
4	15,d	16,b	11,c
5	15,d	16,b	18,c

The distance from b to d is increased by 7 every 3 minutes, so after 39 minutes, we will have:

minute	b	c	d
39	99,d	93,b	95,c
40	99,d	100,b	95,c
41	99,d	100,b	102,c
42	100,a	100,b	102,c
43	100,a	101,b	102,c
44	100,a	101,b	103,c

After this the routes will be stable.

NOTE: I am assuming here that when b upgrades the cost of the link to a to 100, it remembers the last distance vector it received from d advertising a (supposedly) shorter path to a . If b does not cache the vectors from d , the DV progression may instead happen like this:

minute	b	c	d
0	100,a	2,b	4,c
1	8,d	101,b	4,c
2	8,d	9,b	103,c
3	100,a	9,b	11,c
4	15,d	101,b	11,c
5	15,d	16,b	103,c

3. Broadcast

Assume that router a in Figure 1 sends a broadcast packet to all other routers in AS 23.

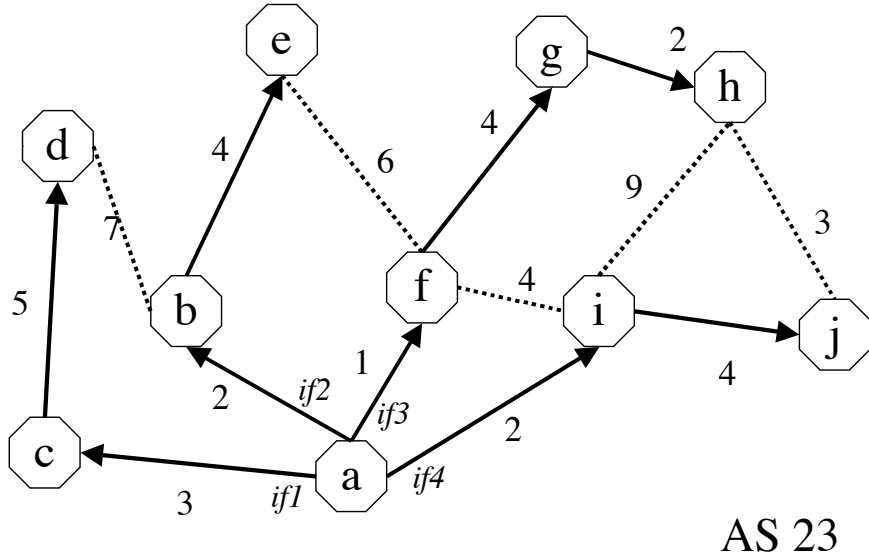
- (a) If application-level broadcast is used, what is the total cost of sending this message? (Add up all the link costs traversed by each packet; i.e. if 3 packets go over a link, count three times the cost.)

The total cost will be the sum of the costs of the least-cost paths to each router, which were computed in 1(a).

$$2 + 3 + 8 + 6 + 1 + 5 + 7 + 2 + 6 = 40$$

- (b) If Reverse-Path Forwarding is used, list how many copies of the packet each router receives. (Hint: there is a fast way to count this.)

Consider the following diagram, depicting the least-cost paths towards every router from a :



AS 23

Each router will receive one packet from its upstream neighbor on the least-cost path tree, and another packet on every link that is not part of the tree. E.g. b will receive one packet from a and another from d. It will not receive a packet from e because e will discard the packet it receives from f, and it will not forward the packet from b back to b.

So the answer is: $a : 0, b : 2, c : 1, d : 2, e : 2, f : 3, g : 1, h : 3, i : 3, j : 2$.

- (c) What is the total cost of sending the broadcast using RPF?

Using the logic of the last problem, simply count the costs of the links that are used to send the packets.

$$(0) + (2 + 7) + (3) + (5 + 7) + (4 + 6) + (1 + 4 + 6) + (4) + (2 + 9 + 3) + (2 + 4 + 9) + (4 + 3) = 85$$

Another way to compute it is to notice that every link in the least-cost path tree is traversed once, and every link not in the tree is traversed twice. So we have:

$$(3 + 5 + 2 + 4 + 1 + 4 + 2 + 2 + 4) + 2 \cdot (7 + 6 + 4 + 9 + 3) = 85$$

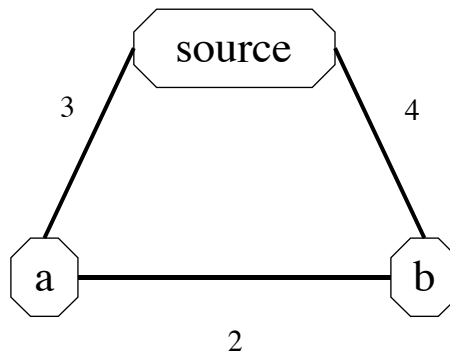
- (d) What is the total cost of sending the broadcast using a least-cost path tree rooted at router a?

$$3 + 5 + 2 + 4 + 1 + 4 + 2 + 2 + 4 = 27$$

4. Multicast

In this question, we are going to consider a modification to the MOSPF protocol that would calculate the minimum weight tree that connects all the routers in a multicast group and use that instead of a source-based least-cost path tree. (Such trees are called *Steiner trees*. They are similar to minimum spanning trees of a graph, but differ in that they do not (necessarily) cover all routers in a network graph, only the selected routers. This problem is, incidentally, *NP*-complete, but we will ignore this fact for the purposes of this problem.)

- (a) Show an example network graph where the least-cost-path source-based tree has larger total weight than the Steiner tree.



The source-based least-cost path tree will exclude the link a-b and have the total weight of 7, whereas the Steiner tree will exclude the link source-b and will have a total weight of 5.

- (b) Assuming that link costs in MOSPF represent the total delay on a link, would you use the Steiner tree or the source-based least-cost path tree for multicast?

The source-based least-cost path tree will provide better performance, because when it comes to delay, what matters is the delay between the source and each node, not the sum of all delays. Using the above example, Steiner tree would create a delay of 5 for b, where the least-cost path tree would leave it with only 4.

- (c) Now suppose the costs represent the level of congestion; which tree would you use?

In this case, the Steiner tree would be better, since it allows one to avoid highly congested links.

- (d) Suppose now that all routers are part of the multicast group, in which case the Steiner tree will be a minimum spanning tree for the entire network. Consider the link cost being 1 for all links; i.e. the total distance between two nodes is just the number of hops. Which tree would you use for this multicast (really, broadcast) routing?

This is a trick question, since in this example, every spanning tree is minimal, so the least-cost path tree will also be a Steiner tree.