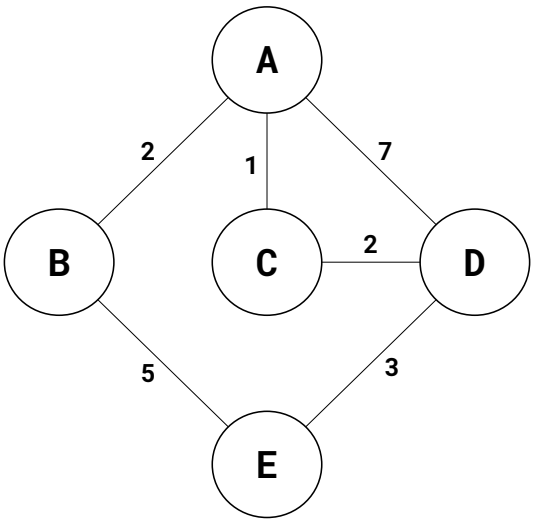
CS 168
Fall 2017

Introduction to the Internet
Scott Shenker

Discussion 4: Routing II

1 Distance-Vector Routing



The nodes in the above network communicate with each other using distance-vector routing. Below are the initial routing tables for each node, and a table showing the costs for each of their neighboring links.

In the routing tables, each row represents a neighbor and a column represents a destination. Each cell entry is of the format `(shortest known distance to dest, next hop)`.

Node A

Nbr	*Cost*
A	0
B	2
C	1
D	7

To / From	A	B	C	D
A	0, A	2, B	1, C	7, D
B	-	0	-	-
C	-	-	0	-
D	-	-	-	0

Node B

Nbr	*Cost*
A	2
B	0
E	5

To / From	A	B	E
A	0	-	-
B	2, A	0, B	5,E
E	-	-	0

Node C

Nbr	*Cost*
A	1
C	0
D	2

To / From	A	C	D
A	0	-	-
C	1, A	0, C	2, D
D	-	-	0

Node D

Nbr	*Cost*
A	7
C	2
D	0
E	3

To / From	A	C	D	E
A	0	-	-	-
C	-	0	-	-
D	7, A	2, C	0, D	3, E
E	-	-	-	0

Node E

Nbr	*Cost*
B	5
D	3
E	0

To / From	B	D	E
B	0	-	-
D	-	0	-
E	5, B	3, D	0, E

The following questions indicate events that happen consecutively. You can assume that there are no other packet exchanges than the ones specified.

EVENT: *C sends its update to A and D.*

(1) What do the routing tables for *A* and *D* look like after receiving *C*'s update? (You may not need to fill in all columns)

Node A

Neighbor	Cost
A	0
B	2
C	1
D	7

To / From	*A*	*B*	*C*	*D*	
A	0,*A*	2,*B*	1,*C*	3,*C*	
B	–	0	–	–	
C	1	–	0	2	
D	–	–	–	0	

Node D

Neighbor	Cost
A	7
C	2
D	0
E	3

To / From	*A*		*C*	*D*	*E*
A	0		–	–	–
C	1		0	2	–
D	3,*C*		2,*C*	0,*D*	3,*E*
E	–		–	–	0

(2) Which nodes among *A* and *D* are expected to send routing updates after receiving *C*'s update?

Solution: Since both *A* and *D* updated their shortest paths, they will both send routing updates.

EVENT: *A sends its update to B, C, and D.*

(3) What do the routing tables for *B*, *C*, and *D* look like after receiving *A*'s update? (You may not need to fill in all columns)

Node B

Neighbor	Cost
A	2
B	0
E	5

To / From	*A*	*B*	*C*	*D*	*E*
A	0	2	1	3	–
B	2,*A*	0	3,*A*	5,*A*	5,*E*
E	–	–	–	–	0

Node C

Neighbor	Cost
A	1
C	0
D	2

To / From	*A*	*B*	*C*	*D*	
A	0	2	1	3	
C	1,*A*	3,*A*	0,*C*	2,*D*	
D	–	–	–	0	

Node D

Neighbor	Cost
A	7
C	2
D	0
E	3

From \ To	A	B	C	D	E
A	0	2	1	3	–
C	1	–	0	2	–
D	$3,C$	$9,A$	$2,C$	$0,D$	$3,E$
E	–	–	–	–	0

(4) At this point, what route does D use to reach B? It knows that it can route to A via C with total distance 3 and that A can reach B with distance 2. Should it use this information to optimize the route to B or should it wait for an update from C?

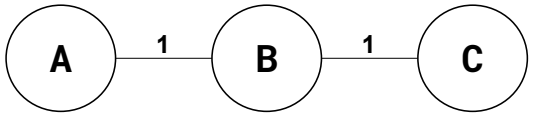
Solution: $D \rightarrow A \rightarrow B$
It should *not* use this information now, as it has no way of knowing whether C is aware of the route to B through A and if C will be able to forward a packet with destination B to A. D needs to wait until it explicitly receives a distance vector from C confirming this, before it can optimize its route to B.

(5) Which nodes among B, C, and D are expected to send routing updates after receiving A's update?

Solution: B, C, and D will all send routing updates since they all updated their distance vectors.

2 Poison Reverse

Consider the following simple network topology:



(1) Assuming that poison reverse was used when exchanging route information, what does B's routing table look like before the link from A to B goes down?

Node B

Neighbor	Cost
A	1
B	0
C	1

	A	B	C
A	0	1	∞
B	$1,A$	$0,B$	$1,C$
C	∞	1	0

EVENT: *B detects a link outage between A and B and sends an update to C.*

(2) What information is contained in B's update?

Solution: B sends an update containing (A: ∞).

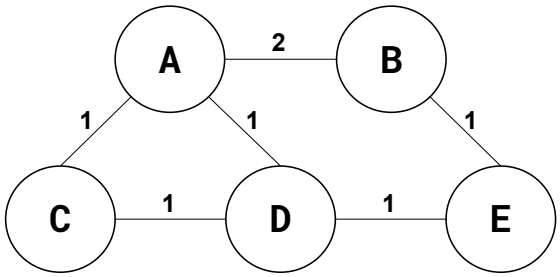
(3) What does C's routing table look like after receiving the update?

Node C

Neighbor	Cost
B	1
C	0

	A	B	C
B	∞	0	1
C	∞	$1,B$	$0,C$

3 Split Horizon and Poisoned Reverse



(1) Assume that the routers use **split horizon**. Say that E sends its initial update (B: 1, D: 1) to D. Assuming that D has received no other updates, what does D now tell E about D's path to B?

Solution: Nothing. Split Horizon means that we never tell a neighbor about paths that go through that neighbor. So in this case, D doesnt tell E about its path to B.

(2) Assume that the routers use **poisoned reverse**. Furthermore, assume that the routing tables haven't converged, and D believes its shortest path to B is D-A-B (length 3). D sends this update to E. Now, E sends its first update (D: 1, B: 1) to D. After recomputing its routes, D sends an update to E. In this update, what is the advertised distance to B?

Solution: D will tell E that its distance to B is infinitely long, because D's new shortest route goes through E.

(3) Now assume that the routers use **split horizon *and* poisoned reverse**. After the same scenario as in (2), what distance to B does D advertise to E?

Solution: D still tells E that D is infinitely far away from B. Split horizon and poisoned reverse are not mutually exclusive but actually typically used at the same time. In this case, even though split horizon says to not tell your neighbor about your paths through it, D must give E some update about its new path to B even though this route goes through E. Namely, if D doesnt update E, E will still believe D uses a route to B of length 3. We can avoid this by using poisoned reverse, which specifies that D should actively lie to E and say that its distance to B is ∞.

(4) Consider the simple topology (A-B-C) from (2). After the routing tables have converged, link A-B goes down. When B sends C an update containing (A: ∞), is this an act of **poisoning a route** or **poisoned reverse**?

Solution: B is **poisoning a route**. Namely, it tells C that its distance is ∞, not because B's new path goes through C, but because B actually has no route now.

(5) **Poisoning a route** and **poisoned reverse** might sound similar, but actually we can think of one of them as being honest while the other one is lying. Which one tells the truth, and which one tells a white lie to keep the network functioning?

Solution: **Poisoned reverse** encourages routers to tell a white lie. With poisoned reverse, we tell a neighbor that we have no path to a certain destination if our path goes through that neighbor. Since we actually do have a path, our message is not strictly true. On the other hand, **poisoning a route** happens when a link goes down, and we actually lose our path to some destination. Thus, were telling the truth when we advertise a distance of ∞ to this destination (given that a infinitely long path is equivalent to no path).