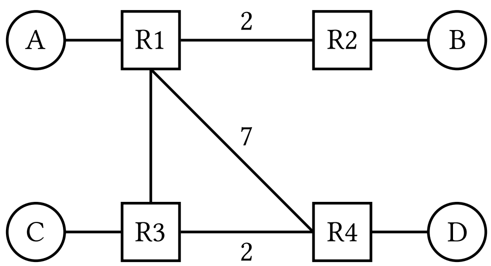
CS 168
Spring 2026

Routing I
Discussion 3

Solutions last updated: June 29, 2025

1 Distance Vector

Consider running the distance-vector protocol on the topology below. Unlabeled links have cost 1.



The routing tables start out initially with direct routes only:

R1's table

Dest.	Hop, Dist.
A	Direct, 1

R2's table

Dest.	Hop, Dist.
B	Direct, 1

R3's table

Dest.	Hop, Dist.
C	Direct, 1

R4's table

Dest.	Hop, Dist.
D	Direct, 1

Assumptions for this question:

- Each subpart continues on from the previous subparts. After finishing each subpart, we suggest first copying your answer to the next subpart before solving the next subpart.
- No other events occur other than the ones specified.
- We use triggered updates: a router sends out advertisements immediately after its table updates.
- We do not use incremental updates: when a router sends out advertisements, it advertises all entries in its table.
- You may not need to fill in all the rows.

1.1 EVENT: **R3 advertises its routes to R1 and R4.**

What do the routing tables look like after receiving R3's routes?

R1's table

Dest.	Hop, Dist.
A	Direct, 1
C	R3, 2

R2's table

Dest.	Hop, Dist.
B	Direct, 1

R3's table

Dest.	Hop, Dist.
C	Direct, 1

R4's table

Dest.	Hop, Dist.
D	Direct, 1
C	R3, 3

1.2 Which routers will advertise their routes after receiving R3's routes?

Since both R1 and R4 updated their tables, they will both advertise their routing tables.

1.3 EVENT: **R1 advertises its routes to R2, R3, and R4.**

What do the routing tables look like after receiving R1's routes?

Entries in **bolded blue** dictate changes caused by new advertisements made in this subpart.

R1's table

Dest.	**Hop, Dist.**
A	Direct, 1
C	R3, 2

R2's table

Dest.	**Hop, Dist.**
B	Direct, 1
A	**R1, 3**
C	**R1, 4**

R3's table

Dest.	**Hop, Dist.**
C	Direct, 1
A	**R1, 2**

R4's table

Dest.	**Hop, Dist.**
D	Direct, 1
C	R3, 3
A	**R1, 8**

1.4 EVENT: **R4 advertises its routes to R1 and R3.**

What do the routing tables look like after receiving R4's routes?

Entries in **bolded blue** dictate changes caused by new advertisements made in this subpart.

R1's table

Dest.	**Hop, Dist.**
A	Direct, 1
C	R3, 2
D	**R4, 8**

R2's table

Dest.	**Hop, Dist.**
B	Direct, 1
A	R1, 3
C	R1, 4

R3's table

Dest.	**Hop, Dist.**
C	Direct, 1
A	R1, 2
D	**R4, 3**

R4's table

Dest.	**Hop, Dist.**
D	Direct, 1
C	R3, 3
A	R1, 8

1.5 EVENT: **R1 advertises its routes to R2, R3, and R4.**

What do the routing tables look like after receiving R1's routes?

Entries in **bolded blue** dictate changes caused by new advertisements made in this subpart.

R1's table

Dest.	**Hop, Dist.**
A	Direct, 1
C	R3, 2
D	R4, 8

R2's table

Dest.	**Hop, Dist.**
B	Direct, 1
A	R1, 3
C	R1, 4
D	**R1, 10**

R3's table

Dest.	**Hop, Dist.**
C	Direct, 1
A	R1, 2
D	R4, 3

R4's table

Dest.	**Hop, Dist.**
D	Direct, 1
C	R3, 3
A	R1, 8

1.6 At this point, what path does R2 use to reach D, and what is the cost?

R2 ⟶ R1 ⟶ R4 with a cost of 10.

R1 has only heard about a route to D from R4. R1 in turn advertises this route to R2.

1.7 EVENT: **R3 advertises its routes to R1 and R4.**

What do the routing tables look like now?

Entries in **bolded blue** dictate changes caused by new advertisements made in this subpart.

R1's table

Dest.	Hop, Dist.
A	Direct, 1
C	R3, 2
D	**R3, 4**

R2's table

Dest.	Hop, Dist.
B	Direct, 1
A	R1, 3
C	R1, 4
D	R1, 10

R3's table

Dest.	Hop, Dist.
C	Direct, 1
A	R1, 2
D	R4, 3

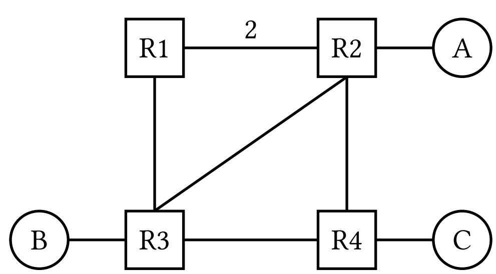
R4's table

Dest.	Hop, Dist.
D	Direct, 1
C	R3, 3
A	**R3, 4**

1.8 Let us now reflect on the routing state after all the advertisements in the previous subparts. In theory, under the most optimal routing state that's attainable, what is the least-cost path that R2 could use to reach D? Do the current routing tables reflect this? If not, what additional advertisement(s) could be done to allow R2 to reach D optimally? If such additional advertisement(s) must be done, express them in this form: `Router X advertises its routes to Router Y`,

The optimal (least-cost) path theoretically attainable from R2 to D is R2 ⟶ R1 ⟶ R3 ⟶ R4 ⟶ D, with optimal cost 6. The current routing state does not reflect this, because in subpart 1.7, R2 still has not heard about R3's routing update to R1 about a lower cost route to D yet. If R1 simply advertises to R2 about its new routes attained after subpart 1.7, then the new routing state will finally allow R2 to reach D optimally!

2 Split Horizon and Poison



All **unlabeled** links have a cost of 1. The parts of the question do **not** build on each other.

2.1 Assume that the routers use **split horizon**. Say that R4 advertises (A: 2, C : 1) to R3. Assuming that R3 has received no other advertisements, what does R3 now tell R4 about R3's path to A?

Nothing. Split Horizon means that we never tell a neighbor about paths that go through that neighbor. So in this case, R3 doesn't tell R4 about its path to A.

2.2 Assume that the routers use **poisoned reverse**. Routing tables have not converged and R3 believes its shortest path to A is through R1 (this path is R3-R1-R2 of length 4). R3 advertises its routes to R4. Now, R4 advertises to R3. R4 bases this advertisement off of its routing table which has: (B: 2, A: 2, C : 1). After recomputing its routes, R3 advertises its routes to R4. What is the advertised distance to A?

R3 will tell R4 that its distance to A is infinitely long, because R3's new shortest route goes through R4.

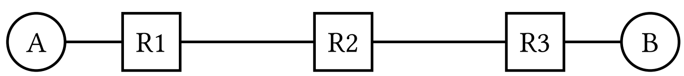
2.3 Consider the simple topology (A-R1-R2-R3). Assume that routing tables have converged, with R1 believing its shortest path to A is through R2 (this path is R1-R2-A of length 3). Then, suppose that link R1-R2 goes down. When R1 advertises to R3 (A: ∞), is this an act of **poisoning a route** or **poisoned reverse**?

R1 is **poisoning a route**. Namely, it tells R3 that its distance is float.inf, not because R1's new path goes through R3, but because R1 actually has no route now.

2.4 **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?

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, we're telling the truth when we advertise a distance of float.inf to this destination (given that an infinitely long path is equivalent to no path).

3 Count to Infinity



For part 1 of this question there is no split-horizon or poisoned reverse, and advertisements are only sent periodically (aka when it is explicitly stated).

3.1 What do the routing tables look like once R1, R2 and R3 converge?

R1's table

Dest.	**Hop, Dist.**
A	Direct, 1
B	R2, 3

R2's table

Dest.	**Hop, Dist.**
A	R1, 2
B	R3, 2

R3's table

Dest.	**Hop, Dist.**
B	Direct, 1
A	R2, 3

3.2 What periodic advertisement will R1 and R2 send to each other? (One such message is given as an example)

From	To	(Destination, Distance)
R1	R2	(A, 1)
R1	R2	(B, 3)
R2	R1	(A, 2)
R2	R1	(B, 2)

3.3 **EVENT**: **The link between R2 and R3 goes down.**

What will R1 and R2 send to each other?

From	To	(Destination, Distance)
R1	R2	(A, 1)
R1	R2	(B, 3)
R2	R1	(A, 2)
R2	R1	(B, 2)

The tables did not change because no routes have expired yet.

3.4 **EVENT**: R2's route to B **finally** expires.

After R1 and R2 exchange advertisements again, what will their routing tables look like?

R1's table

Dest.	Hop, Dist.
A	Direct, 1
B	R2, 3

R2's table

Dest.	Hop, Dist.
A	R1, 2
B	R1, 4

R3's table

Dest.	Hop, Dist.
B	Direct, 1
A	R2, 3

3.5 **EVENT**: R1's route to B expires.

After R1 and R2 exchange advertisements again, what will their routing tables look like?

R1's table

Dest.	Hop, Dist.
A	Direct, 1
B	R2, 5

R2's table

Dest.	Hop, Dist.
A	R1, 2
B	R1, 4

R3's table

Dest.	Hop, Dist.
B	Direct, 1
A	R2, 3

3.6 Is this good?

No! This is called count to infinity. Both switches think they have a path to B for a long time after the path ceases to exist.

For the remainder of this question, there is **split-horizon**, but **no** poisoned reverse, and advertisements are only sent periodically (i.e., when it is explicitly stated). Also, all dropped links are back up, and the routing state starts out converged!

3.7 What will R1 and R2 send to each other after everything has converged?

From	To	(Destination, Distance)
R1	R2	(A, 1)
R2	R1	(B, 2)

3.8 **EVENT**: The link between R2 and R3 goes down.

What will R1 and R2 send to each other?

From	To	(Destination, Distance)
R1	R2	(A, 1)
R2	R1	(B, 2)

3.9 **EVENT**: R2's route to B **finally** expires.

After R1 and R2 exchange advertisements again, what will their routing tables look like?

R1's table

Dest.	Hop, Dist.
A	Direct, 1
B	R2, 3

R2's table

Dest.	Hop, Dist.
A	R1, 2

R3's table

Dest.	Hop, Dist.
B	Direct, 1
A	R2, 3

3.10 Will this end well?

Yes! R1's route to B will expire because it has not been updated in a while.