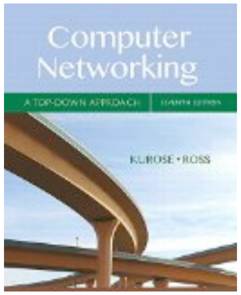
COMP 375: Lecture 33



- **News & Notes:**
  - Project #5:
    - **Protocol Spec Due**: Wednesday
    - **Code Due**: Wed, May 2
- **Reading (Wed, Apr. 25)**
  - Sections 6.1, 6.3

Section 5.2

DISTANCE VECTOR ROUTING

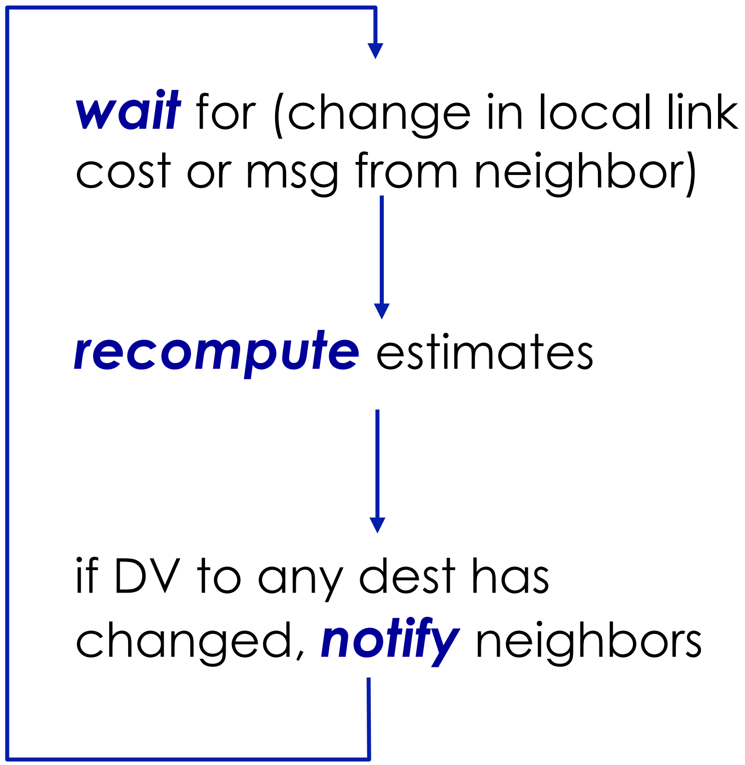
A node's **distance vector** is its estimated costs to every other node in the graph.

$$D_x = [D_x(y) : y \in N]$$

- Node x knows the following about neighbor v:
  - Its cost to v: $c(x,v)$
  - v's Distance Vector: $\mathbf{D}_v = [D_v(y) : y \in N]$

DV is iterative and asynchronous, with link cost changes triggering updates.

Each node:

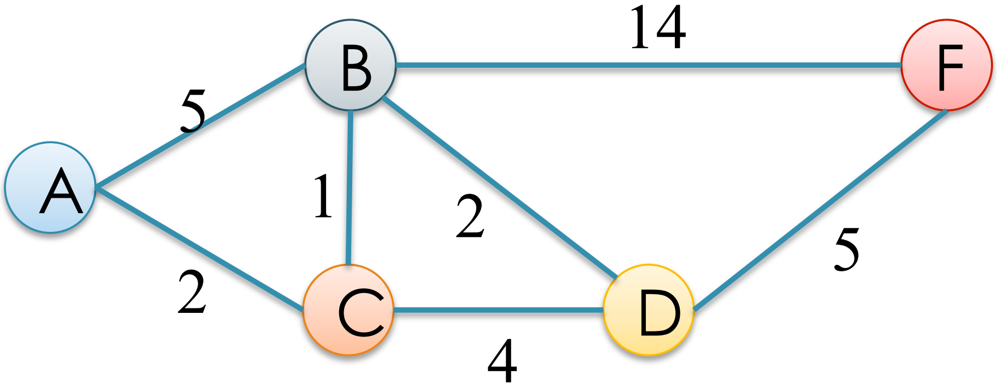


wait for (change in local link cost or msg from neighbor)

recompute estimates

if DV to any dest has changed, ***notify*** neighbors

At the beginning, tables contain info only about direct neighbors.



Router F

Via→ ↓ To	B	D
A		
B	14	
C		
D		5

Router A

Via→ ↓ To	B	C
B	5	
C		2
D		
F		

Router B

Via→ ↓ To	A	C	D	F
A	5			
C		1		
D			2	
F				14

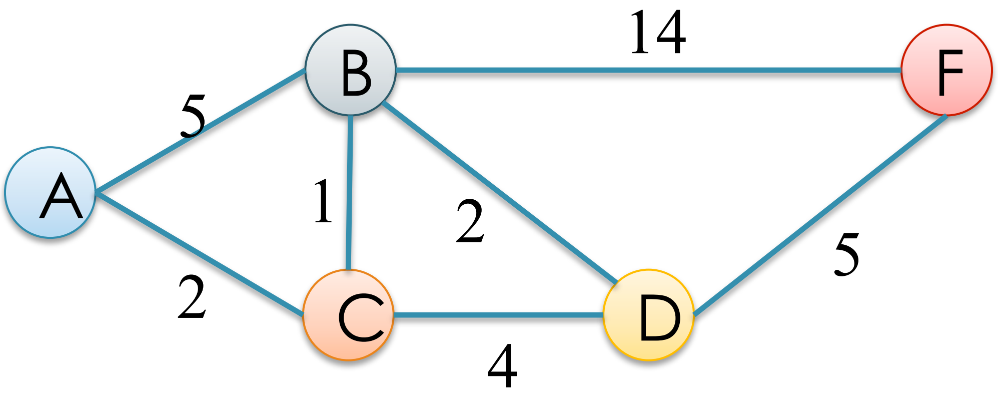
Router C

Via→ ↓ To	A	B	D
A	2		
B		1	
D			4
F			

Router D

Via→ ↓ To	B	C	F
A			
B	2		
C		4	
F			5

After sending distance vectors to direct neighbors, we updated our tables.



Router F

Via→ ↓ To	B	D
A	19	
B	14	7
C	15	9
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	19	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

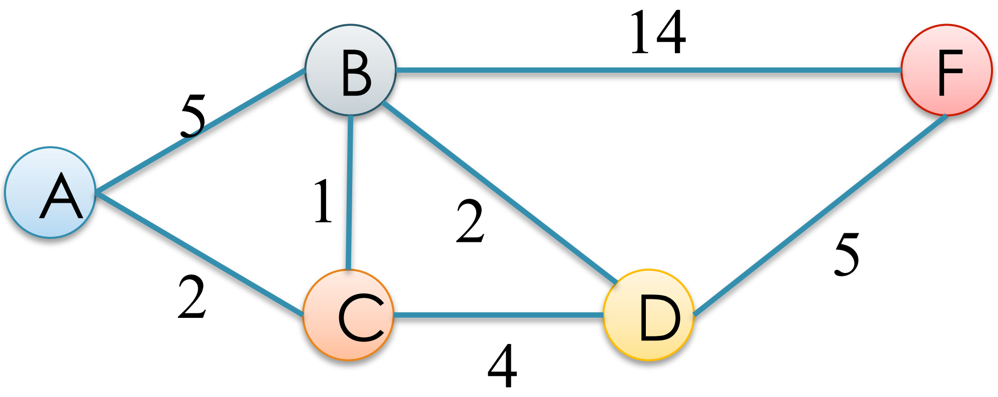
Router C

Via→ ↓ To	A	B	D
A	2	6	
B	7	1	6
D		3	4
F		15	9

Router D

Via→ ↓ To	B	C	F
A	7	6	
B	2	5	19
C	3	4	
F	16		5

At the end of round 1, **how many** routers need to update their forwarding tables?



Router F

Via→ ↓ To	B	D
A	19	
B	14	7
C	15	9
D	16	5

A. 1, **B.** 2, **C.** 3, **D.** 4, [✓ **E.**] 5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	19	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

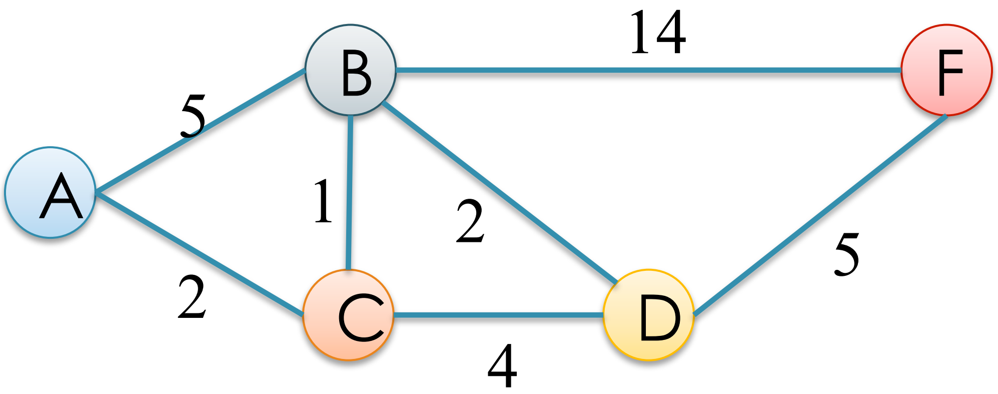
Router C

Via→ ↓ To	A	B	D
A	2	6	
B	7	1	6
D		3	4
F		15	9

Router D

Via→ ↓ To	B	C	F
A	7	6	
B	2	5	19
C	3	4	
F	16		5

For Round 2, everyone advertises their updated distance vector.



Router F

Via→ ↓ To	B	D
A	19	
B	14	7
C	15	9
D	16	5

Nothing new to learn from A or F, so we'll their ignore them.

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	19	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

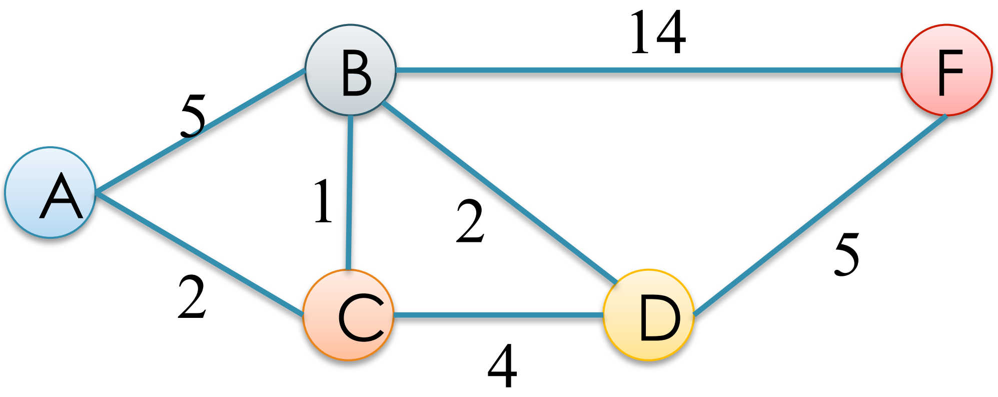
Router C

Via→ ↓ To	A	B	D
A	2	6	
B	7	1	6
D		3	4
F		15	9

Router D

Via→ ↓ To	B	C	F
A	7	6	
B	2	5	19
C	3	4	
F	16		5

B will send to its neighbors (A, C, D, F)



I can get to A in 3, C in 1, D in 2, and F in 7.

Router F

Via→ ↓ To	B	D
A	17	
B	14	7
C	15	9
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	12	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

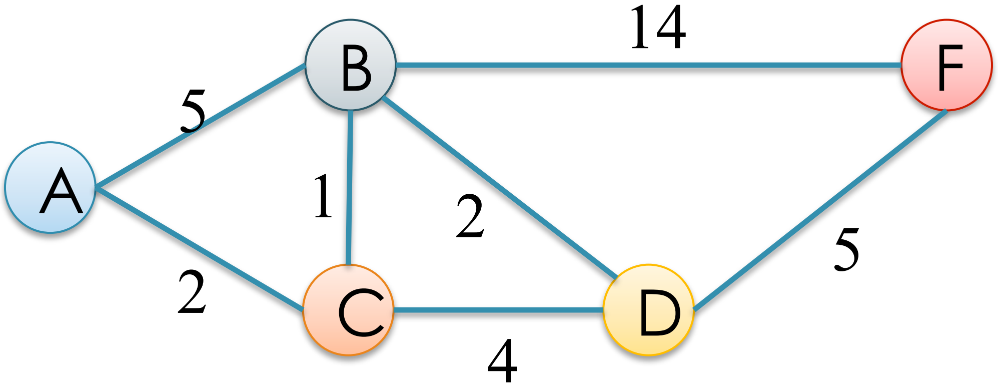
Router C

Via→ ↓ To	A	B	D
A	2	4?	
B	7	1	6
D		3	4
F		8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	
B	2	5	19
C	3	4	
F	9?		5

C will send to its neighbors (A, B, D).



I can get to A in 2, B in 1, D in 3, and F in 9.

Router F

Via→ ↓ To	B	D
A	17	
B	14	7
C	15	9
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	5
F	12	11

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		4?	2	19
F		10	7	14

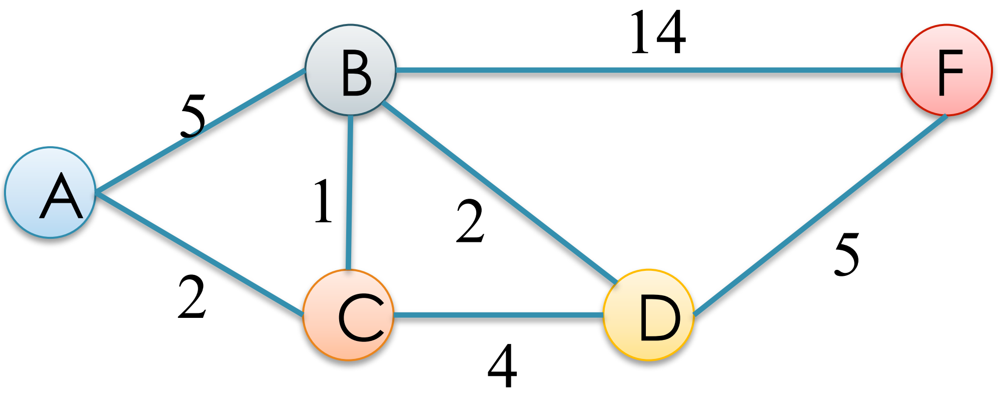
Router C

Via→ ↓ To	A	B	D
A	2	4?	
B	7	1	6
D		3	4
F		8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	
B	2	5	19
C	3	4	
F	9?	13?	5

This process will repeat for multiple rounds...



Router F

Via→ ↓ To	B	D
A	17	
B	14	7
C	15	9
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	5
F	12	11

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		4?	2	19
F		10	7	14

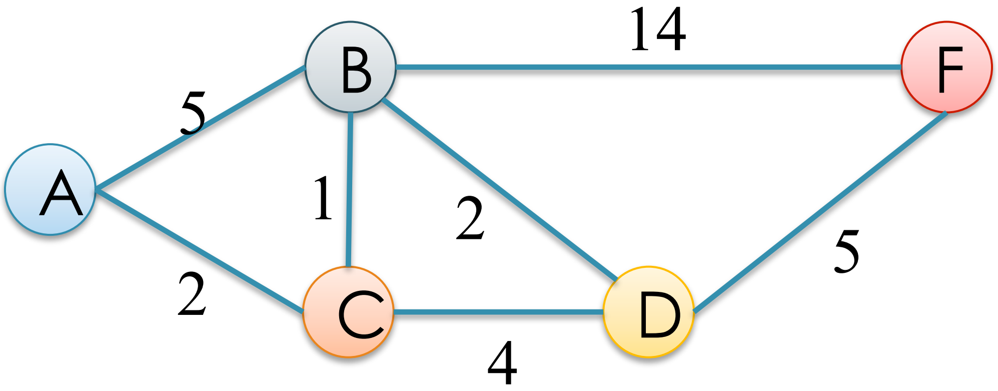
Router C

Via→ ↓ To	A	B	D
A	2	4?	
B	7	1	6
D		3	4
F		8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	
B	2	5	19
C	3	4	
F	9?	13?	5

Eventually we will reach a **converged state**, with no updates required.



Router F

Via→ ↓ To	B	D
A	17	10
B	14	7
C	15	8
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	5
F	12	10

Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	7	14

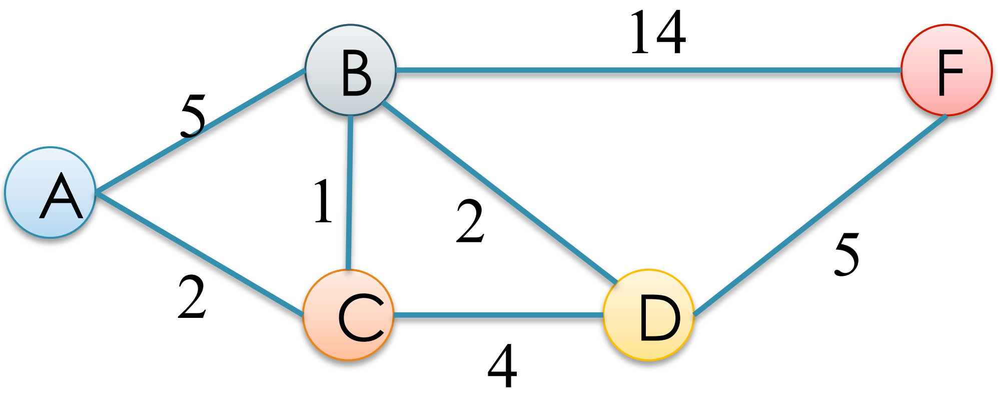
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	5

After convergence, we have the following forwarding tables.



Router F

Via→ ↓ To	B	D
A	17	10
B	14	7
C	15	8
D	16	5

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	5
F	12	10

Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	7	14

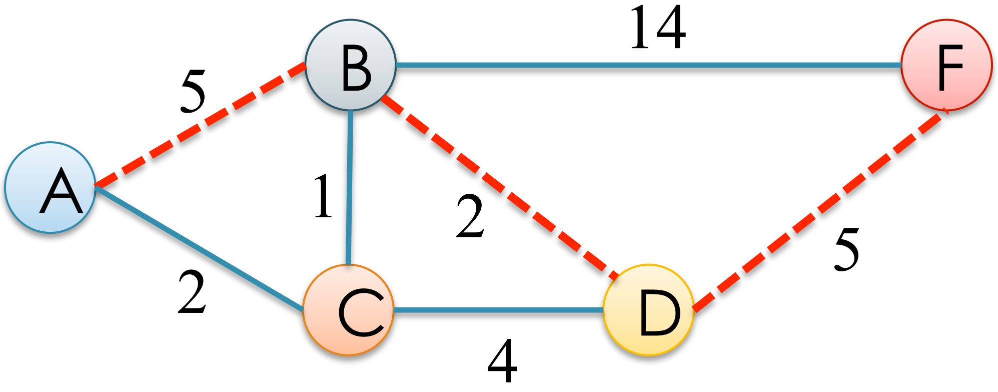
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	5

Of the red, dashed links below, for **how many** would a failure cause a loop?



Router F

Via→ ↓ To	B	D
A	17	10
B	14	7
C	15	8
D	16	5

A. 0, **B.** 1, **C.** 2, [✓ **D.**] 3

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	5
F	12	10

Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	7	14

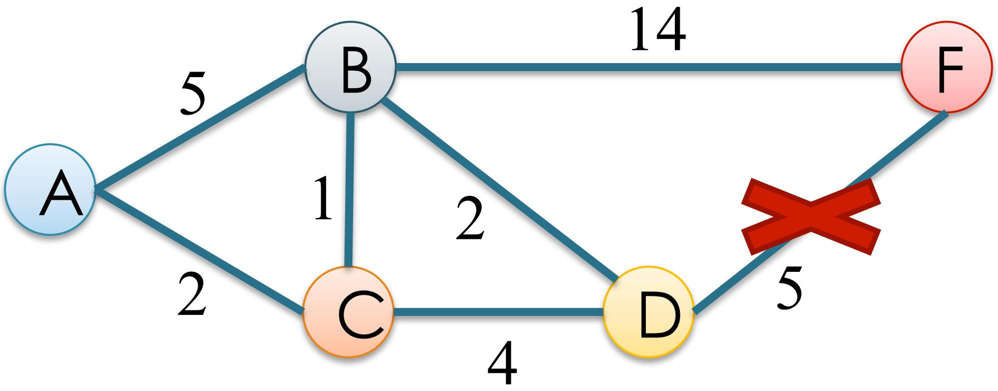
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	5

If the D-F link goes down, D will update its distance vector and advertise it.



Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	7	14

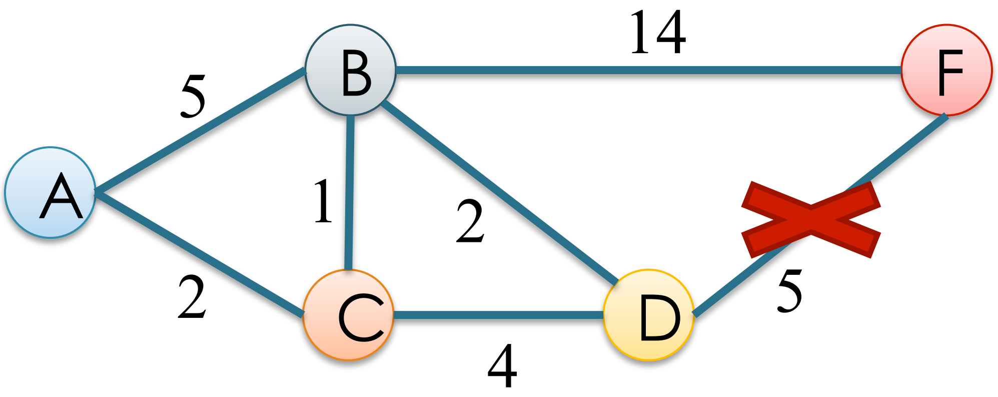
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	∞

B’s distance vector changes, so we send F traffic through C, who sends it back to B!



Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	**11**	14

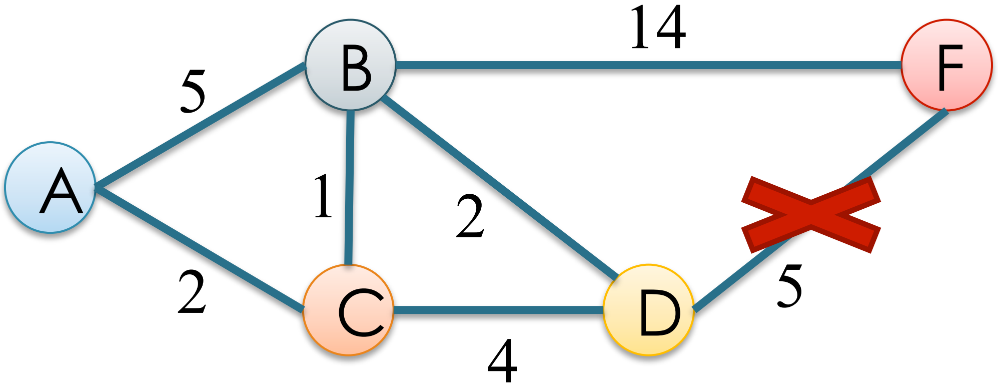
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	**13**

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	∞

Will this routing loop persist forever?



A. **Yes.**

B. **Yes**, but only while D-F link is down.

[✓ **C.**] **No.**

Router B

Via→ ↓ To	A	C	D	F
A	5	3	7	24
C	7	1	4	22
D	10	4	2	19
F	15	9	**11**	14

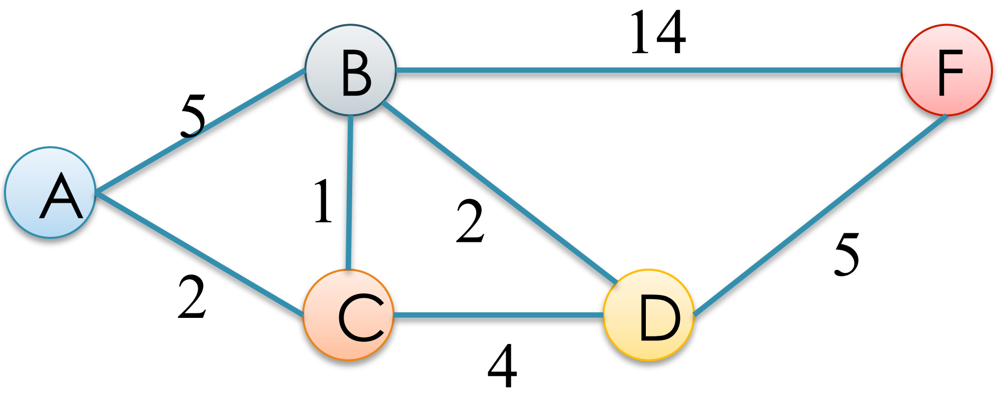
Router C

Via→ ↓ To	A	B	D
A	2	4	9
B	7	1	6
D	7	3	4
F	12	8	**13**

Router D

Via→ ↓ To	B	C	F
A	5	6	15
B	2	5	12
C	3	4	13
F	9	12	∞

Rewind: Distance Vector – Round 2



Router F

Via→ ↓ To	B	D
A	17	
B	14	7
C	15	9
D	16	5

B will send to neighbors (A, C, D, F):
I can get to A in 3, C in 1, D in 2, and F in 7.

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	12	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

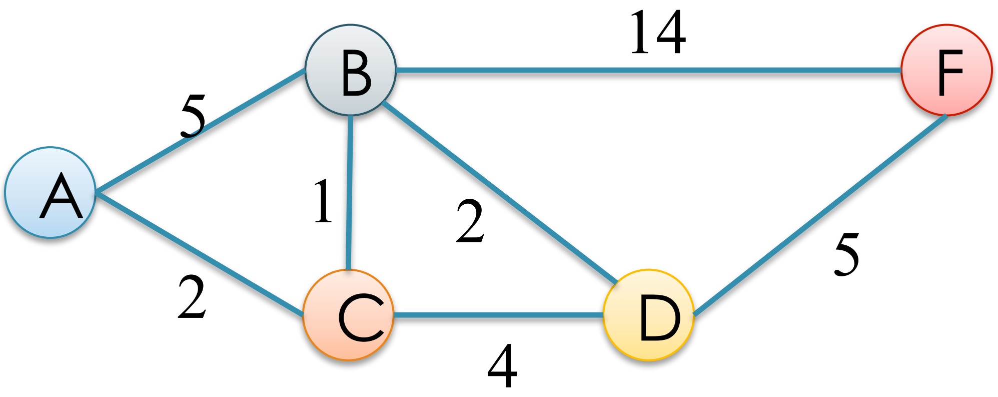
Router C

Via→ ↓ To	A	B	D
A	2	4?	
B	7	1	6
D		3	4
F		8	9

Router D

Via→ ↓ To	B	C	F
A	5	6	
B	2	5	19
C	3	4	
F	9?		5

Rewind: Distance Vector – Round 2



Router F

Via→ ↓ To	B	D
A	17	
B	14	7
C	15	9
D	16	5

Poisoned reverse: Don't advertise a lower value to a neighbor if you go through that neighbor to get there!

Router A

Via→ ↓ To	B	C
B	5	3
C	6	2
D	7	6
F	12	

Router B

Via→ ↓ To	A	C	D	F
A	5	3		
C	7	1	6	
D		5	2	19
F			7	14

Router C

Via→ ↓ To	A	B	D
A	2	~~4?~~	
B	7	1	6
D		3	4
F		8	9

No!

∞

Rout

Via→ ↓ To			
A	[illegible]	[illegible]	[illegible]
B	2	5	19
C	3	4	
F	~~9?~~		5

No!

∞

Loop-Prevention Mechanisms

- Route poisoning helps prevent loops, but doesn't guarantee loop free.

- Other mechanisms help too:
  - Split horizon
  - Hold-down timers

- There will always be a window of vulnerability

Summary: Link-State vs Distance-Vector

Link-State

- Fast convergence (reacts to events quickly)
- Small window of inconsistency

- Large number of messages sent on events
- Large routing tables as network size grows

Distance-Vector

- Distributed (small tables)
- No flooding (fewer messages)

- Slower convergence
- Larger window of inconsistency