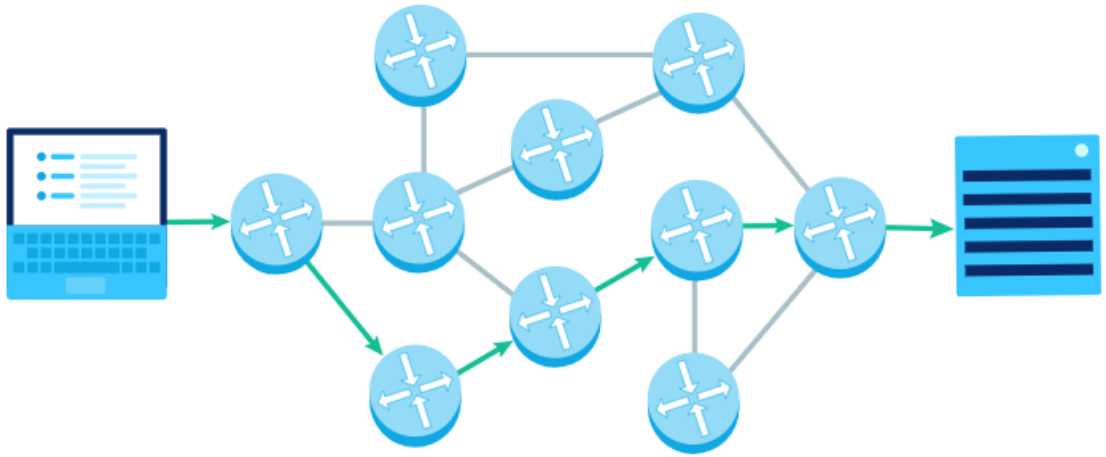
PSIT
Kanpur

Computer Networks

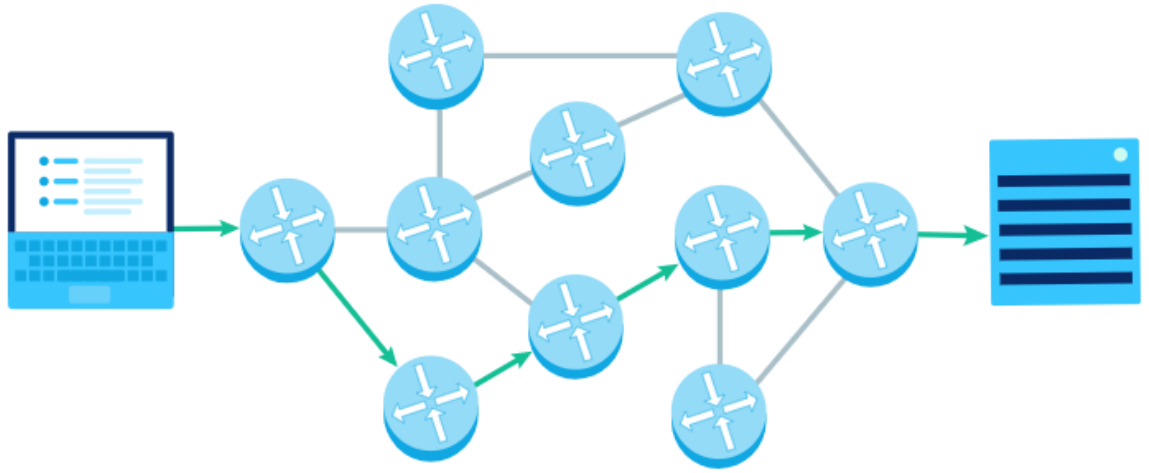
(RCS-601)

Routing - II



PSIT
Kanpur

Dynamic / Adaptive
Routing Algorithms



Distance vector	Link state	Path vector
RIP	OSPF	BGP

PSIT
Kanpur

Distance Vector Routing

- Distance Vector Algorithm (based on **Bellman-Ford algorithm**) is a Adaptive / Dynamic algorithm.

- Key features:
  - In Distance Vector Routing Algorithm, each node shares its routes in the network only to the neighbors and does not broadcast it. (not to all)
  - Whenever any node receives the Routing Information it updates its own routing table and inform to its neighbors.

PSIT Kanpur

Distance Vector Routing (contd.)

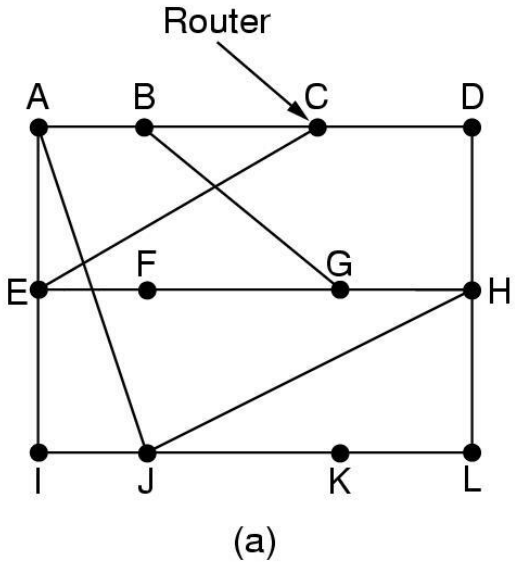
Algorithm

- A router transmits its distance vector to each of its neighbors in a routing packet.
- Each router receives and saves the most recently received distance vector from each of its neighbors.
- A router recalculates its distance vector when:
  - It receives a distance vector from a neighbor containing different information than before.
  - It discovers that a link to a neighbor has gone down.
- The DV (Distance Vector) calculation is based on minimizing the cost to each destination

Note –

- From time-to-time, each node sends its own distance vector estimate to neighbors.
- When a node x receives new DV estimate from any neighbor v, it saves v's distance vector and it updates its own DV.

Distance Vector Routing (contd.)



(a)

To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

(b)

Routing Table for A

To	cost	via
A	0	-
B	12	B
C	25	B
D	40	B
E	14	E
F	23	E
G	18	B
H	17	J
I	21	E
J	9	J
K	24	J
L	29	J

PSIT Kanpur

Distance Vector Routing (contd.)

Drawback of Distance Vector Algorithm

Count to Infinity Problem

The main issue with **D**istance Vector **R**outing (DVR) protocols is Routing Loops, since Bellman-Ford Algorithm cannot prevent loops. This routing loop in Distance Vector routing network causes **Count to Infinity Problem**. Routing loops usually occur when any interface goes down or two-routers send updates at the same time.

How fast are changes propagated?

- Good news?
- Bad news?

Distance Vector Routing (contd.)

Count to Infinity Problem

How fast are changes propagated?

Good news?

A	B	C	D	E	
	∞	∞	∞	∞	Initially
	1	∞	∞	∞	After 1 exchange
	1	2	∞	∞	After 2 exchanges
	1	2	3	∞	After 3 exchanges
	1	2	3	4	After 4 exchanges

(a)

Bad news?

A	B	C	D	E	
	1	2	3	4	Initially
	3	2	3	4	After 1 exchange
	3	4	3	4	After 2 exchanges
	5	4	5	4	After 3 exchanges
	5	6	5	6	After 4 exchanges
	7	6	7	6	After 5 exchanges
	7	8	7	8	After 6 exchanges
		⋮			
	∞	∞	∞	∞	

(b)

From the previous figure, it should be clear why bad news travels slowly: no router ever has a value more than one higher than the minimum of all its neighbors. Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity. For this reason, it is wise to set infinity to the longest path plus 1.

If the metric is time delay, there is no well-defined upper bound, so a high value is needed to prevent a path with a long delay from being treated as down. Not entirely surprisingly, this problem is known as the **count-to-infinity** problem.

PSIT
Kanpur

Thank you