ECCS-3631
Networks and Data Communications

Module 2-2
Distance Vector Routing Algorithm

Dr. Ajmal Khan

Distance Vector Algorithm

Distance-Vector algorithm uses Bellman-Ford equation to calculate least-cost path from x to y

Bellman-Ford equation

Define

$d_x(y)$:= cost of least-cost path from x to y

then

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\},$$

$d_v(y)$: cost from neighbor v to destination y

$c(x,v)$: cost to neighbor v

$\min_v$: *min* taken over all neighbors v of x

Distance Vector Algorithm

Basic idea:

- from time-to-time, each node sends its own distance vector estimate to neighbors
- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow min_v\{c(x,v) + D_v(y)\} \text{ for each node } y \in N$$

- In normal cases, the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$

Distance Vector Algorithm

iterative, asynchronous:

Each local iteration caused by:

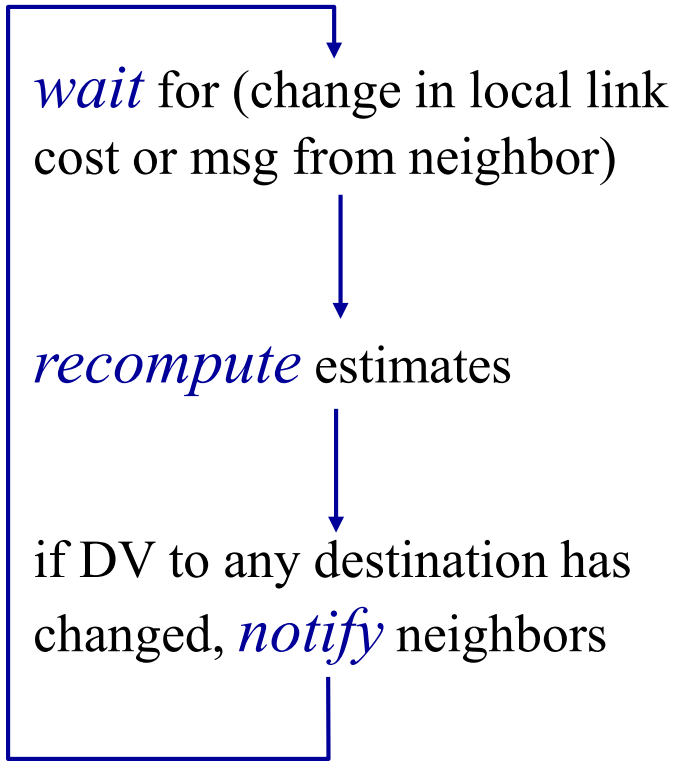
- local link cost change
- DV update message from neighbor

distributed:

Each node notifies neighbors *only* when its DV changes

- neighbors then notify their neighbors if necessary

Each Node:



$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

node x table

from	cost to x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from	cost to x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from	cost to x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from	cost to x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from	cost to x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

from	cost to x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

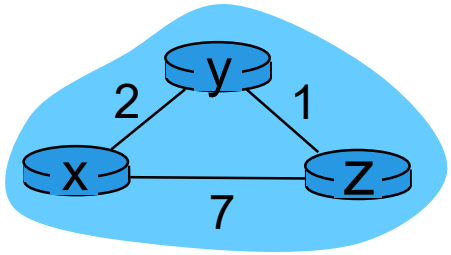
node z table

from	cost to x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from	cost to x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from	cost to x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

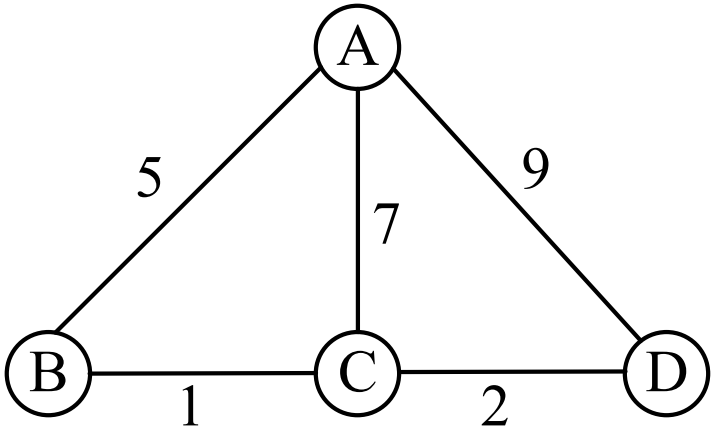
time



Distance-Vector: Example

Computer the short-path from Noda A to all network nodes using Distance-Vector Algorithm

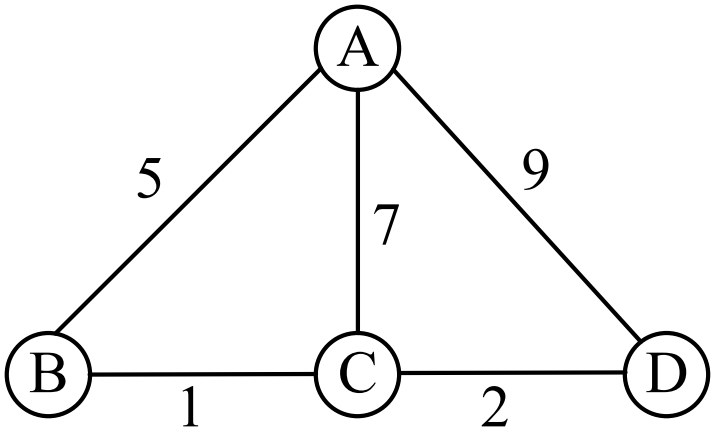
Step 1: Node A finds cost to directly connected neighbors



	A	B	C	D
A	0	5	7	9
B	∞	∞	∞	∞
C	∞	∞	∞	∞
D	∞	∞	∞	∞

Distance-Vector: Example

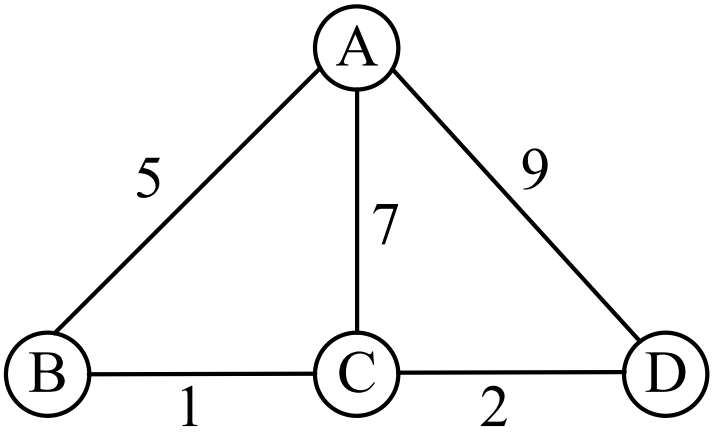
Step 2: Node A receives cost from all neighbors



	A	B	C	D
A	0	5	6	9
B	5	0	1	∞
C	7	1	0	2
D	9	∞	2	0

Distance-Vector: Example

Step 3: Node A calculates minimum cost from Node A to all nodes



	A	B	C	D
A	0	5	6	8
B	5	0	1	3
C	6	1	0	2
D	9	3	2	0

Comparison of LS and DV Algorithms

distance vector:

- distribute one's own routing table to neighbors
  - routing update can be large in size, but travels only one link
- each node only knows distances to other destinations

link state

- broadcast raw topology information to entire net
  - routing update is small in size, but travels over all links in the net
- each node knows entire topology

Performance measure: Message complexity, Time to convergence

Robustness: what happens if router malfunctions?

LS:

- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others

What we have talked about routing

Dijkstra routing algorithm

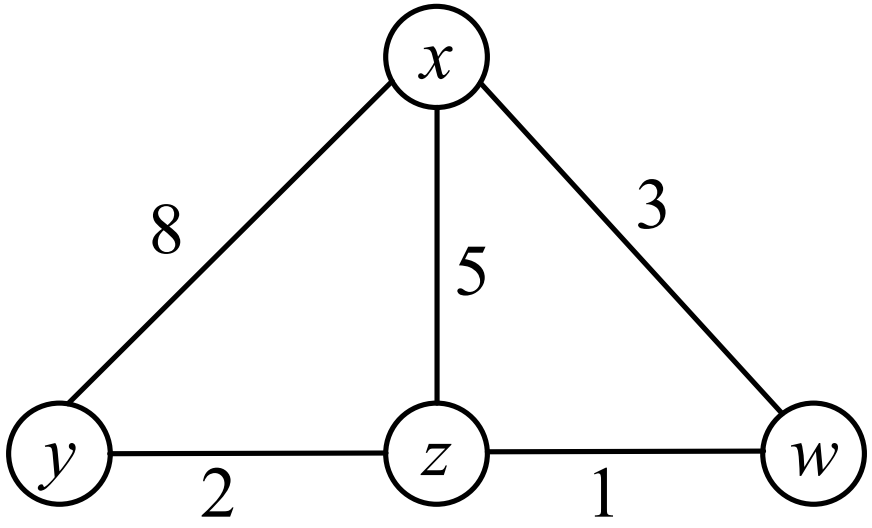
- Given *a topology map*, compute the shortest paths to all the other nodes

Bellman-Ford routing algorithm

- Given *the lists of distance to all destinations* from all the neighbors, compute the shortest path to destination

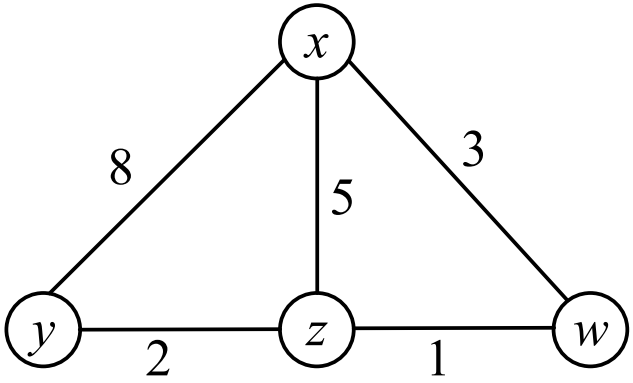
DV Algorithm: Practice Problem #1

Consider the following network. With the indicated link costs, compute the shortest-path from **x** to all network nodes using Distance-Vector Routing Algorithm.



DV Algorithm: Practice Problem #1

STEP 1

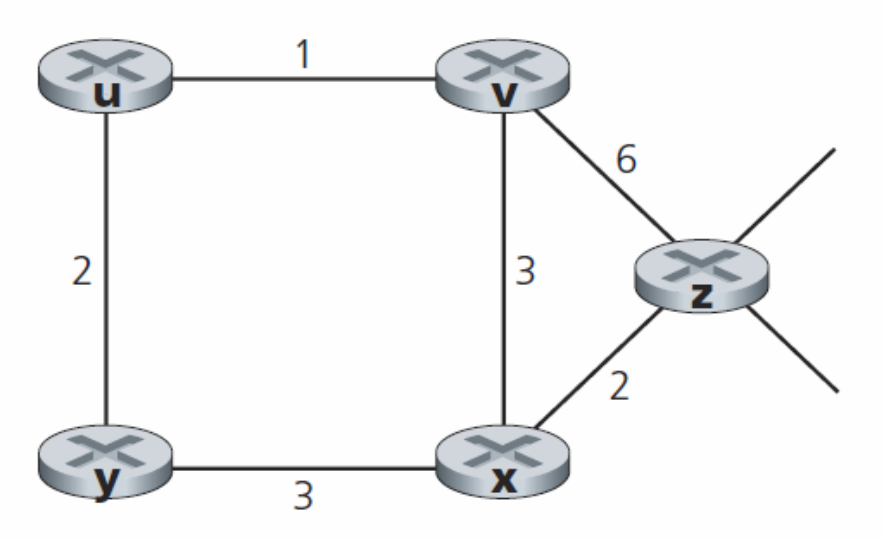


STEP 3

STEP 2

DV Algorithm: Practice Problem #2

Compute the shortest-path from **z** to all network nodes, using Distance-Vector Routing Algorithm.



DV Algorithm: Practice Problem #2