Distance Vector Routing Algorithm

iterative:

- continues until no nodes exchange info.
- *self-terminating*: no "signal" to stop

asynchronous:

- nodes need *not* exchange info/iterate in lock step!

distributed:

- each node communicates *only* with directly-attached neighbors

Distance Table data structure

- each node has its own
- row for each possible destination
- column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

$$D^X(Y,Z) = \text{distance } \textit{from} \text{ X } \textit{to} \text{ Y, } \textit{via} \text{ Z as next hop}$$

$$= c(X,Z) + \min_w \{D^Z(Y,w)\}$$

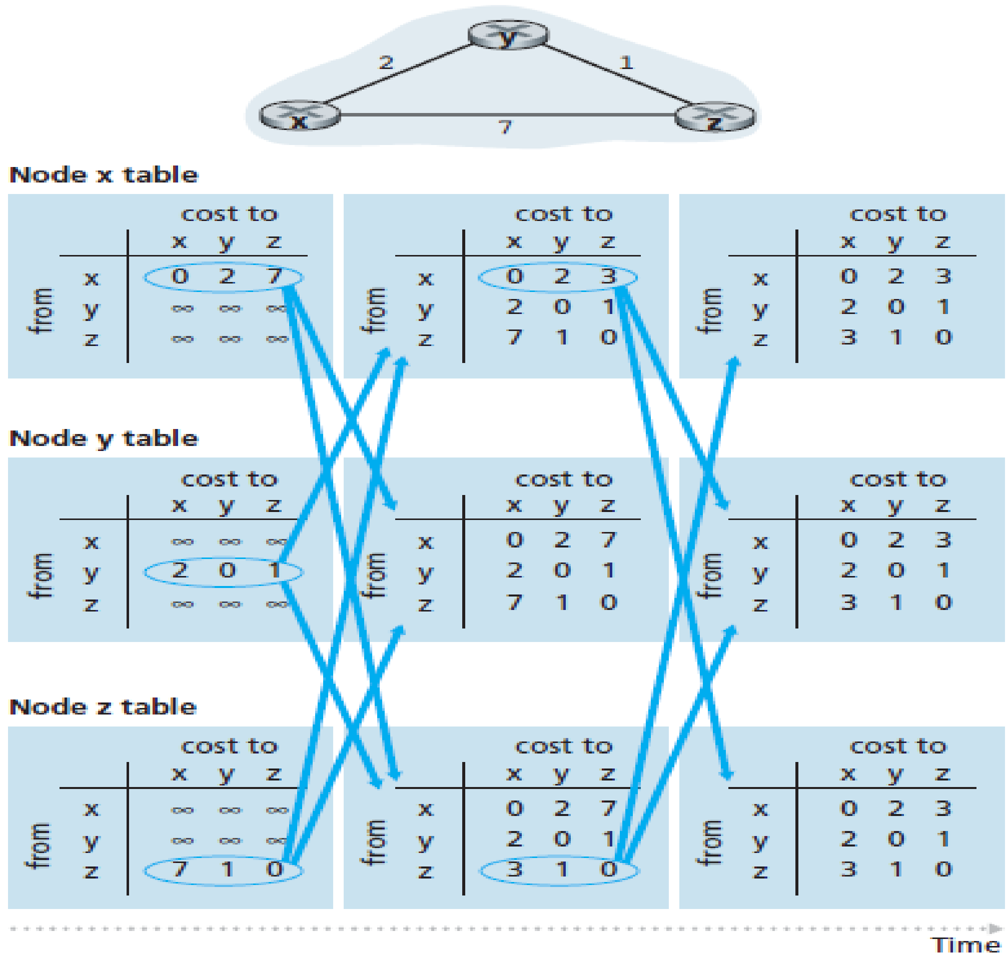
Distance Vector Algorithm

Distance-Vector (DV) Algorithm

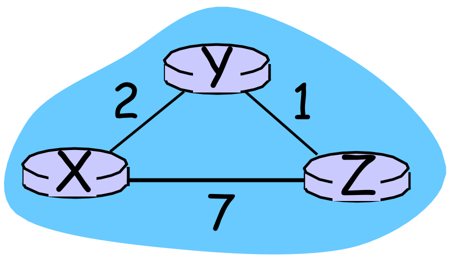
At each node, *x*:

```
Initialization:
   for all destinations y in N:
      D_x(y) = c(x,y)    /* if y is not a neighbor then c(x,y) = ∞ */
   for each neighbor w
      D_w(y) = ? for all destinations y in N
   for each neighbor w
      send distance vector D_x = [D_x(y): y in N] to w

loop
   wait (until I see a link cost change to some neighbor w or
         until I receive a distance vector from some neighbor w)

   for each y in N:
      D_x(y) = min_v{c(x,v) + D_v(y)}

   if D_x(y) changed for any destination y
      send distance vector D_x = [D_x(y): y in N] to all neighbors

forever
```

Distance Vector Algorithm: example



Distance Vector Algorithm: example



D^X	cost via Y	cost via Z
dest Y	2	∞
dest Z	∞	7

D^Y	cost via X	cost via Z
dest X	2	∞
dest Z	∞	1

D^Z	cost via X	cost via Y
dest X	7	∞
dest Y	∞	1

D^X	cost via Y	cost via Z
dest Y	2	8
dest Z	3	7

$$D^X(Y,Z) = c(X,Z) + \min_w\{D^Z(Y,w)\}$$
$$= 7+1 = 8$$

$$D^X(Z,Y) = c(X,Y) + \min_w\{D^Y(Z,w)\}$$
$$= 2+1 = 3$$

Count To Infinity Problem→ Route Poisoning

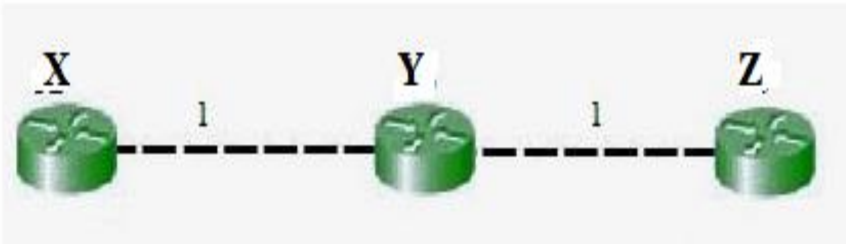


Figure (a)

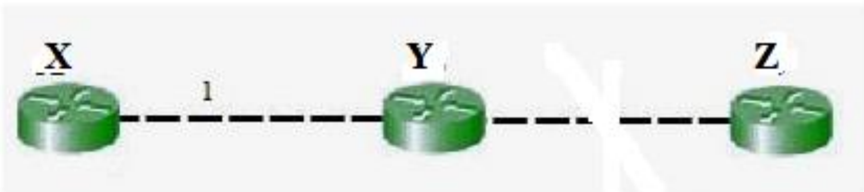


Figure (b)