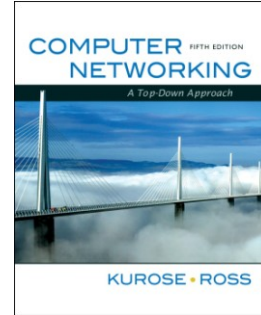


RSC

Part II: Network Layer

5. Network routing (2nd Part)



Redes y Servicios de Comunicaciones
Universidad Carlos III de Madrid

These slides are, mainly, part of the companion slides to the book "Computer Networking: A Top Down Approach" generously made available by their authors (see copyright below). The slides have been adapted, where required, to the teaching needs of the subject above.

All material copyright 1996-2009
J.F Kurose and K.W. Ross, All Rights Reserved

*Computer Networking:
A Top Down Approach
5th edition.*

Jim Kurose, Keith Ross
Addison-Wesley, April
2009.

Network Layer II-1

RSC Part II: Network Layer

- ❑ II. 1 Basic Network layer concepts
- ❑ II.2 Introduction to IP
 - Datagram format
 - ICMP
- ❑ II.3 IP addressing
 - Obtaining addresses, DHCP, NAT
- ❑ II.4 IP in operation
 - ARP
- ❑ II.5 Network routing
 - Link state
 - Distance Vector
- ❑ II.6 Routing in the Internet
 - Hierarchical routing
 - RIP
 - OSPF
 - BGP
 - Broadcast and multicast

Network Layer II-2

Distance Vector Algorithm

Bellman-Ford Equation (dynamic programming)

Define

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

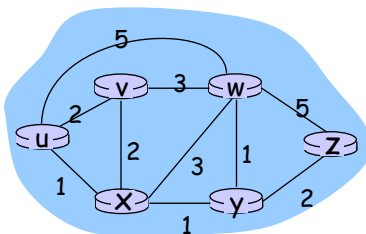
Then

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

where min is taken over all neighbors v of x

Network Layer 4-3

Bellman-Ford example



Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\quad c(u,x) + d_x(z), \\ &\quad c(u,w) + d_w(z) \} \\ &= \min \{ 2 + 5, \\ &\quad 1 + 3, \\ &\quad 5 + 3 \} = 4 \end{aligned}$$

Node that achieves minimum is next
hop in shortest path → forwarding table

Network Layer 4-4

Distance Vector Algorithm

- $D_x(y)$ = estimate of least cost from x to y
- Node x knows cost to each neighbor v :
 $c(x,v)$
- Node x maintains distance vector $D_x = [D_x(y): y \in N]$
- Node x also maintains its neighbors' distance vectors
 - For each neighbor v , x maintains $D_v = [D_v(y): y \in N]$

Network Layer 4-5

Distance vector algorithm (4)

Basic idea:

- From time-to-time, each node sends its own distance vector estimate to neighbors
- Asynchronous
- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:
$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$
- Under minor, natural conditions, the estimate $D_x(y)$ converge to the actual least cost $d_x(y)$

Network Layer 4-6

Distance Vector Algorithm (5)

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

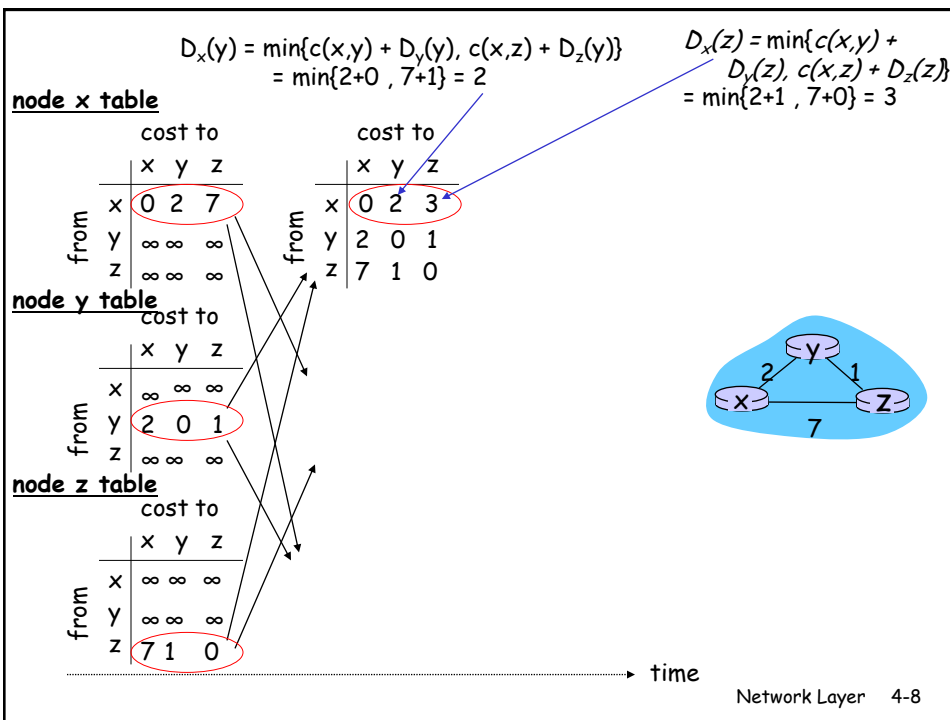
Each node:

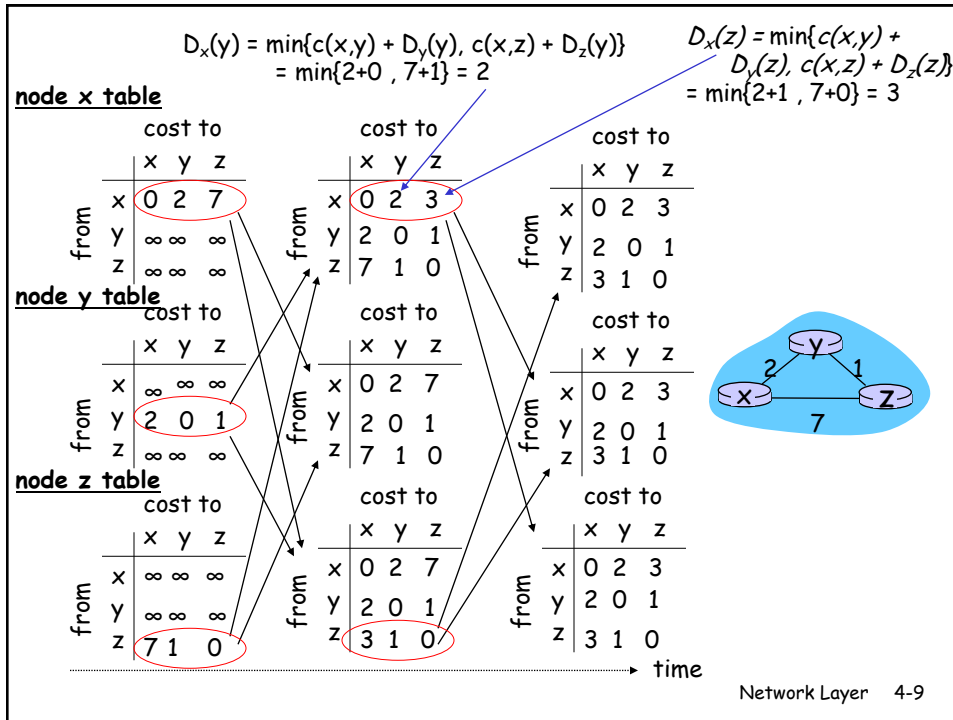
wait for (change in local link cost or msg from neighbor)

recompute estimates

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

Network Layer 4-7

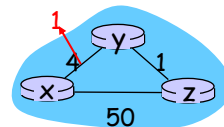




Distance Vector: link cost changes

Link cost changes:

- ❑ node detects local link cost change
- ❑ updates routing info, recalculates distance vector
- ❑ if DV changes, notify neighbors



"good
news
travels
fast"

At time t_0 , y detects the link-cost change, updates its DV, and informs its neighbors.

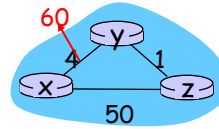
At time t_1 , z receives the update from y and updates its table. It computes a new least cost to x and sends its neighbors its DV

At time t_2 , y receives z's update and updates its distance table. y's least costs do not change and hence y does *not* send any message to z.

Distance Vector: link cost changes

Link cost changes:

- ❑ good news travels fast
- ❑ bad news travels slow - "count to infinity" problem!
- ❑ 44 iterations before algorithm stabilizes: see text



Poisoned reverse:

- ❑ If Z routes through Y to get to X:
 - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- ❑ will this completely solve count to infinity problem?

Network Layer 4-11

Comparison of LS and DV algorithms

Message complexity

- ❑ LS: with n nodes, E links, $O(nE)$ msgs sent
- ❑ DV: exchange between neighbors only
 - convergence time varies

Speed of Convergence

- ❑ LS: $O(n^2)$ algorithm requires $O(nE)$ msgs
 - may have oscillations
- ❑ DV: convergence time varies
 - may be routing loops
 - count-to-infinity problem

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
 - error propagate thru network

Network Layer 4-12