Network Routing

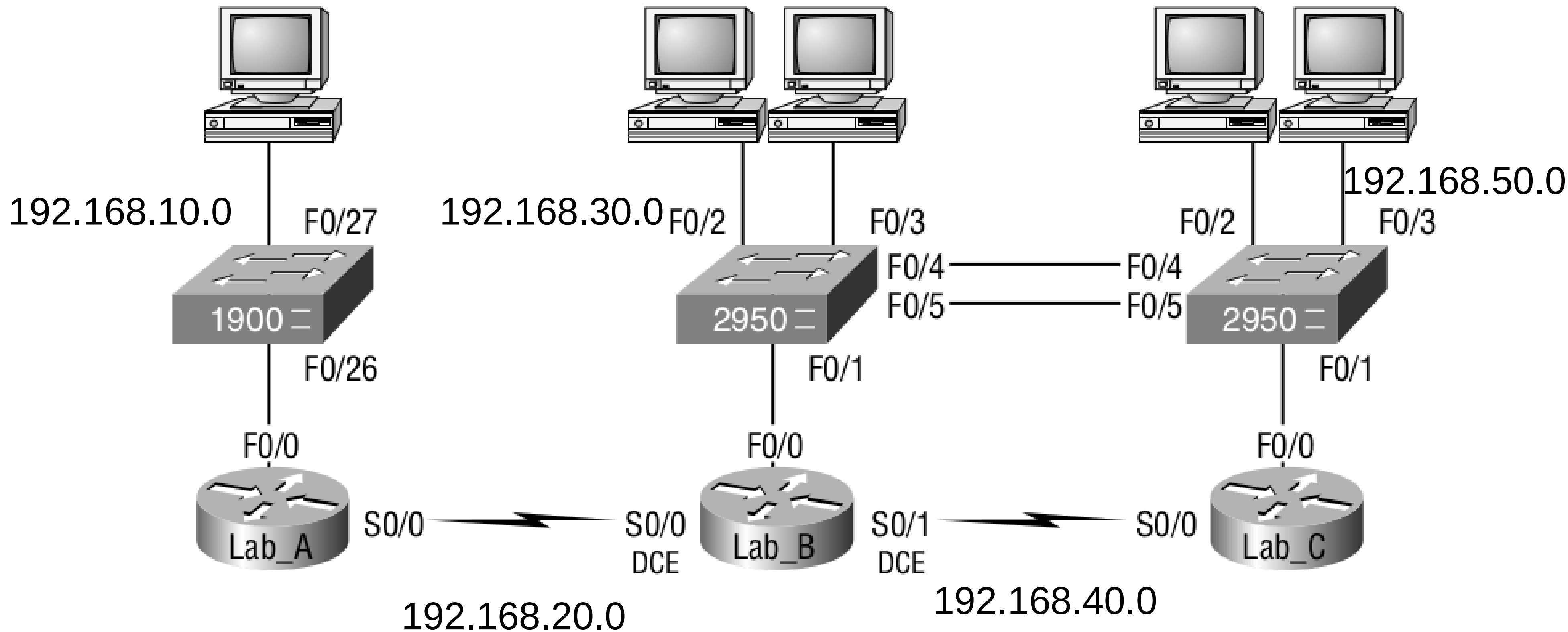
Computer Science Faculty
Kandahar University

Routing

Default Routing

- In computer networking, the default route is a setting on a computer that defines the packet forwarding rule to use when no specific route can be determined for a given Internet Protocol (IP) destination address.
- All packets for destinations not established in the routing table are sent via the default route.

Default Routing- Stub-Network



IP Addresses

Router	Network Address	Interface	Address
Lab_A	192.168.10.0	fa0/0	192.168.10.1
Lab_A	192.168.20.0	s0/0	192.168.20.1
Lab_B	192.168.20.0	s0/0	192.168.20.2
Lab_B	192.168.40.0	s0/1	192.168.40.1
Lab_B	192.168.30.0	fa0/0	192.168.30.1
Lab_C	192.168.40.0	s0/0	192.168.40.2
Lab_C	192.168.50.0	fa0/0	192.168.50.1

Default Route Application

- Default routes are useful when dealing with a network with a single exit point.
- It is also useful when a bulk of destination networks have to be routed to a single next-hop device.
- To configure a default route, you use wildcards in the network address and subnetmask

LAB C configuration DEMO

Dynamic Routing

- Dynamic routing is when protocols, called routing protocols, are used to build the routing tables across the network.
- Using a routing protocol is easier than static routing and default routing.
- more expensive in terms of CPU and bandwidth usage.
- Diffirrent rules for different routing protocols.

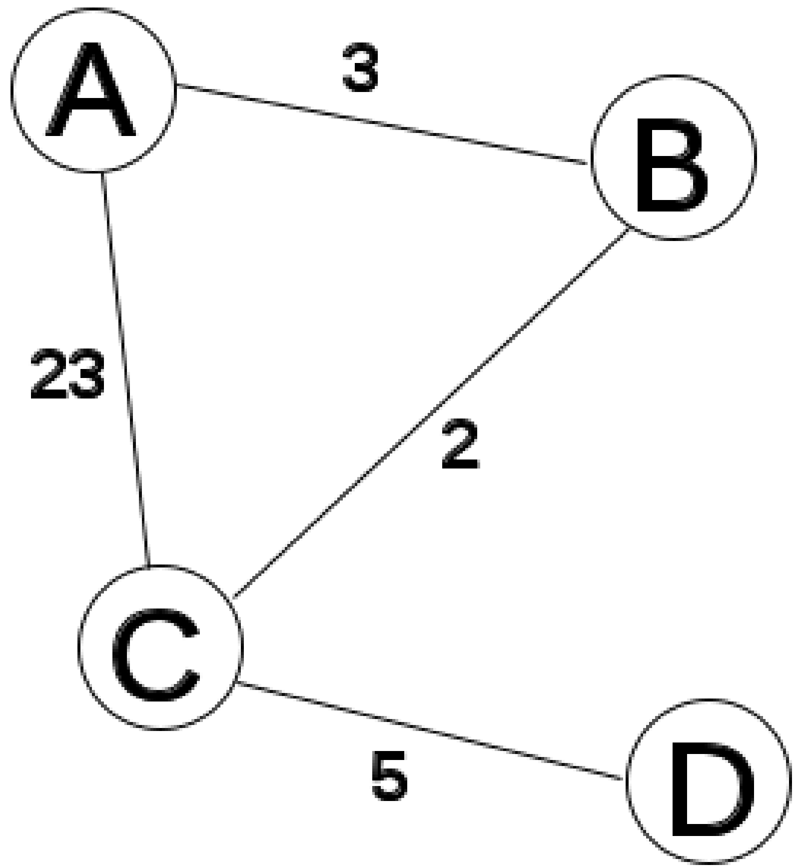
Classes of routing algorithms

- Distance vector algorithms
- Link-state algorithms
- Hybrid protocols

Distance vector algorithms

- Uses Bellman-Ford algorithm to calculate paths.
- A distance-vector routing protocol requires that a router inform its neighbors of topology changes periodically.
- The term distance vector refers to the fact that the protocol manipulates vectors (arrays) of distances to other nodes in the network.
- The distance vector algorithm was the original ARPANET routing algorithm
- RIPv1 and RIPv2, IGRP, EIGRP, and Babel.

Bellman-Ford algorithm-Path calculation



Link-state algorithms

- When applying link-state algorithms, each node uses as its fundamental data a map of the network in the form of a graph.
- The basic concept of link-state routing is that every node constructs a map of the connectivity to the network, in the form of a graph.
- showing which nodes are connected to which other nodes.
- Each node then independently calculates the next best logical path from it to every possible destination in the network
- Open Shortest Path First (OSPF) and intermediate system to intermediate system (IS-IS).

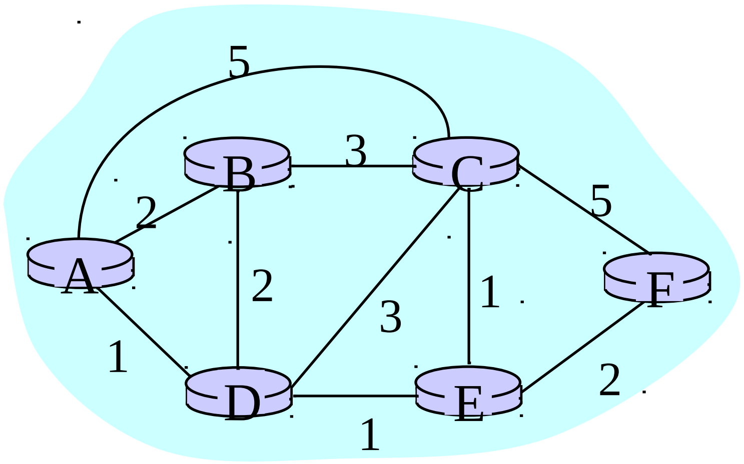
Network Routing: algorithms & protocols

Goal: find "good" path to each destination

- Graph abstraction of a network
  - Nodes: routers
  - Edges: physical links (with assigned cost)

route computation algorithms

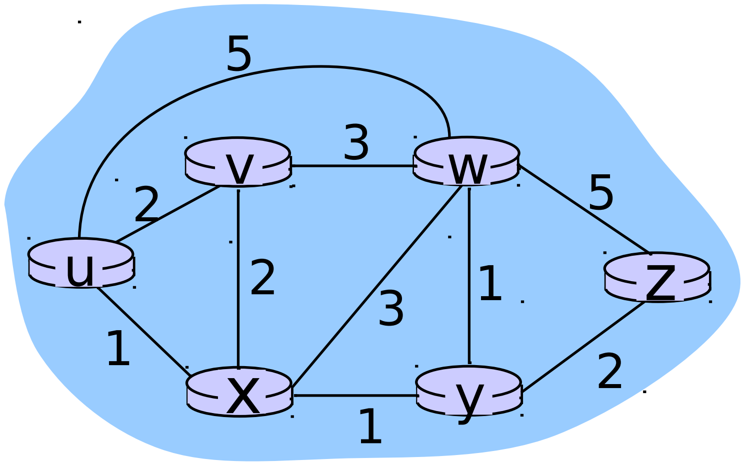
- link-state (Dijkstra)
  - each router knows complete topology & link cost information
  - Run routing algorithm to calculate shortest path to each destination
- distance-vector (Bellman-Ford)
  - Each router knows direct neighbors & link costs to neighbors
  - Calculate the shortest path to each destination through an *iterative* process *based on the neighbors distances* to each destination



Routing protocols

- define the format of routing information exchanges
- define the computation upon receiving routing updates
- network topology changes over time, routing protocol must continuously update the routers with latest changes

Graph abstraction: costs



- c(x,x') = cost of link (x,x')

  - e.g., c(w,z) = 5

- cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

Cost of path $(x_1, x_2, x_3,\ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Dijkstra's algorithm

- Assume net topology, link costs is known
- computes least cost paths from one node to all other nodes
- Create forwarding table for that node

Notation:

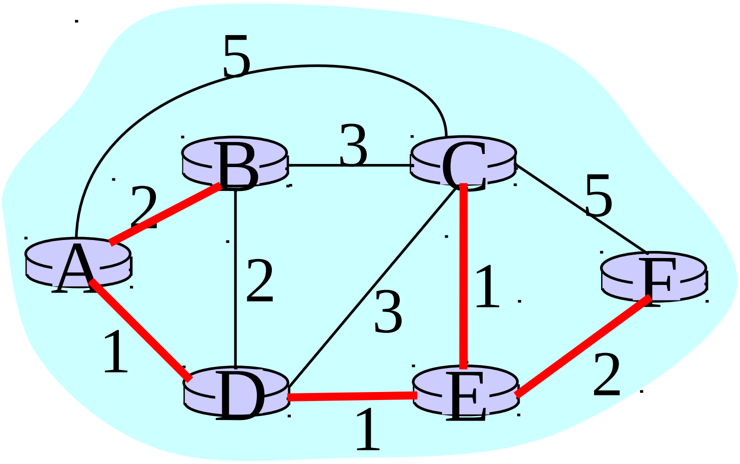
- c(i,j): link cost from node i to j (∞ if not known)
- D(v): current value of cost of path from source to dest. V
- p(v): predecessor node along path from source to v, (neighbor of v)
- N': set of nodes whose least cost path already known

```
Initialization:
 N' = {A}
 for all nodes v
   if v adjacent to A
     then D(v) = c(A,v)
     else D(v) = ∞

 Loop
   find w not in N' such that D(w) is
   minimum
   add w to N'
   update D(v) for all v adjacent to w
   and not in N':
      D(v) = min( D(v), D(w) + c(w,v) )
   /* new cost to v is either the old
   cost, or known shortest path cost to
   w plus cost from w to v */
 until all nodes in N'
```

Dijkstra's algorithm: example

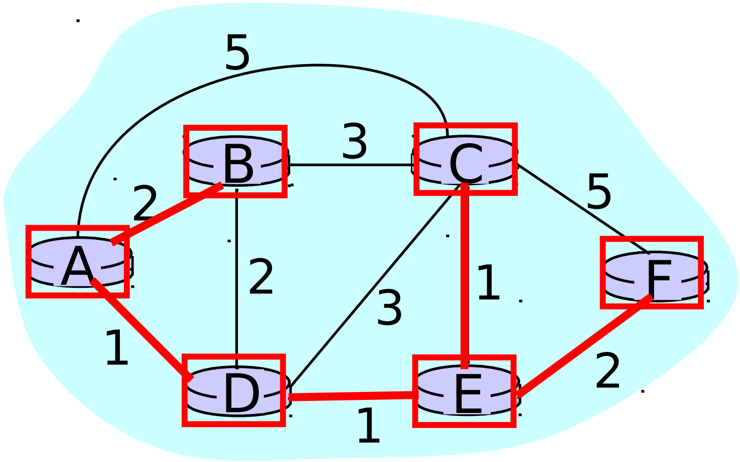
Step	start N'	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D		2,D	infinity
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					



Dijkstra's algorithm: example

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D		2,D	infinity
2	ADB		4,D		2,D	infinity
3	ADBE		3,E			4,E
4	ADBEC					4,E
5	ADEBCF					

Resulting shortest-path tree for A:



Resulting forwarding table at A:

destination	link
B	(A, B)
D	(A, D)
E	(A, D)
C	(A, D)
F	(A, D)

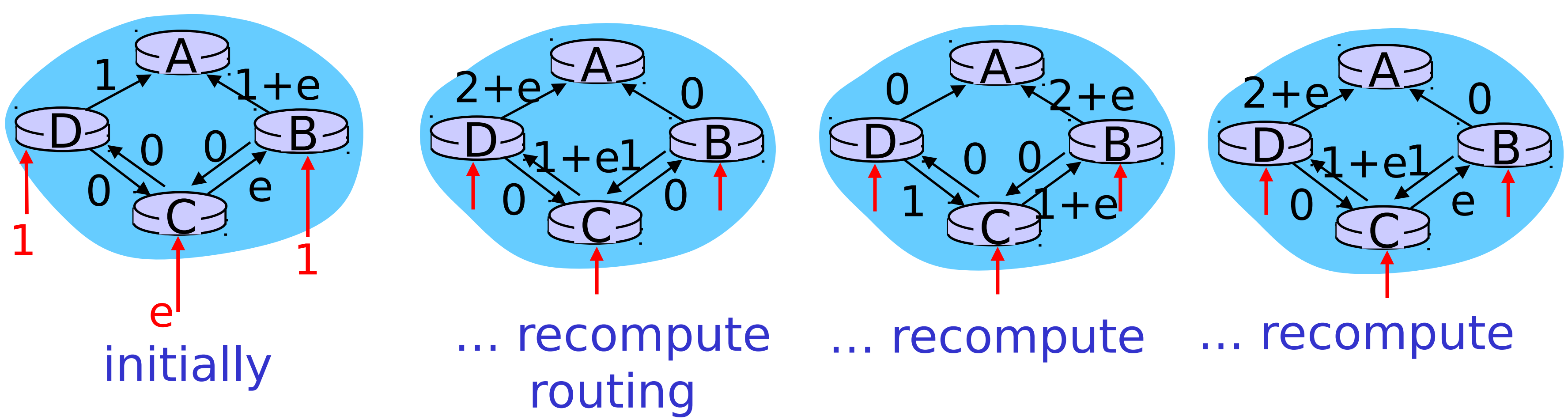
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w, not in N
- n(n+1)/2 comparisons: $O(n^2)$
- more efficient implementations possible: O(nlogn)

Oscillations possible:

- e.g., link cost = amount of carried traffic

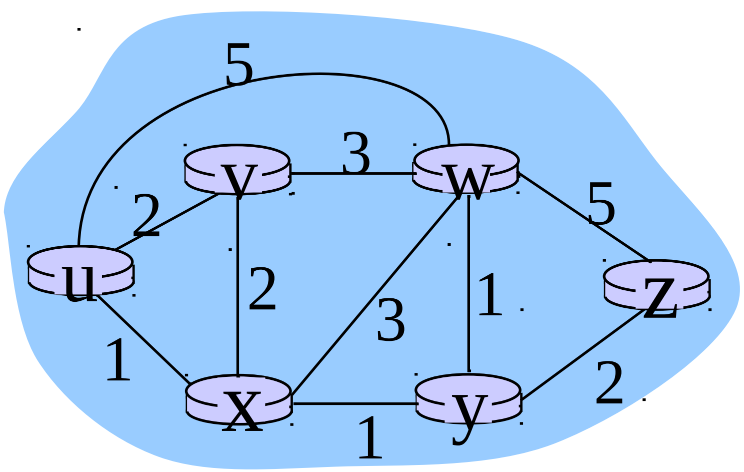


Bellman-Ford Equation

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

Then $D_x(y) = \min \{c(x,v) + D_v(y)\}$

- where min is taken over *all* neighbors v of x



$$\begin{aligned} D_u(z) = \min \{ & c(u,v) + D_v(z), \\ & c(u,x) + D_x(z), \\ & c(u,w) + D_w(z) \} \end{aligned}$$

$$\begin{aligned} = \min \{ & 2 + 5, \\ & 1 + 3, \\ & 5 + 3\} = 4 \end{aligned}$$

Node leading to shortest path is next hop ➜ forwarding table

Distance vector protocl (1)

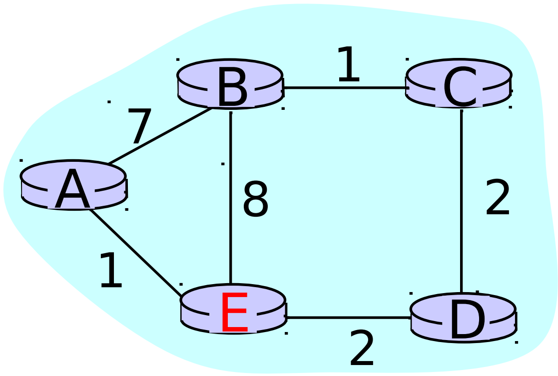
Basic idea:

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

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

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

Distance Table: example



cost to destination via

$D^E()$	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

destination

D^E	Outgoing link
A	A,1
B	D,5
C	D,4
D	D,2

destination

forwarding table

Distance Vector Protocol (2)

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 of msg from neighbor)

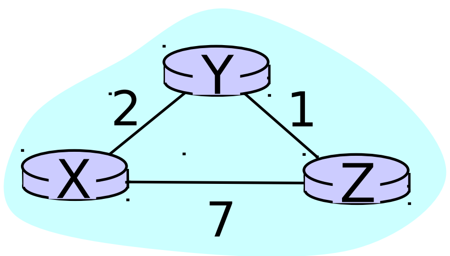
↓

recompute estimates

↓

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

Distance Vector: an example



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

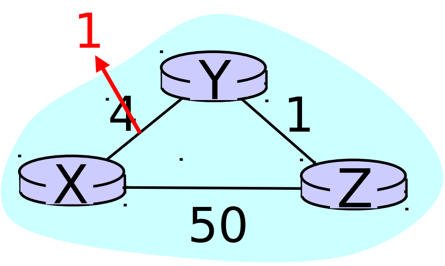
D^Y	cost via X	cost via Z
dest X	2	∞
dest Z	∞	1

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

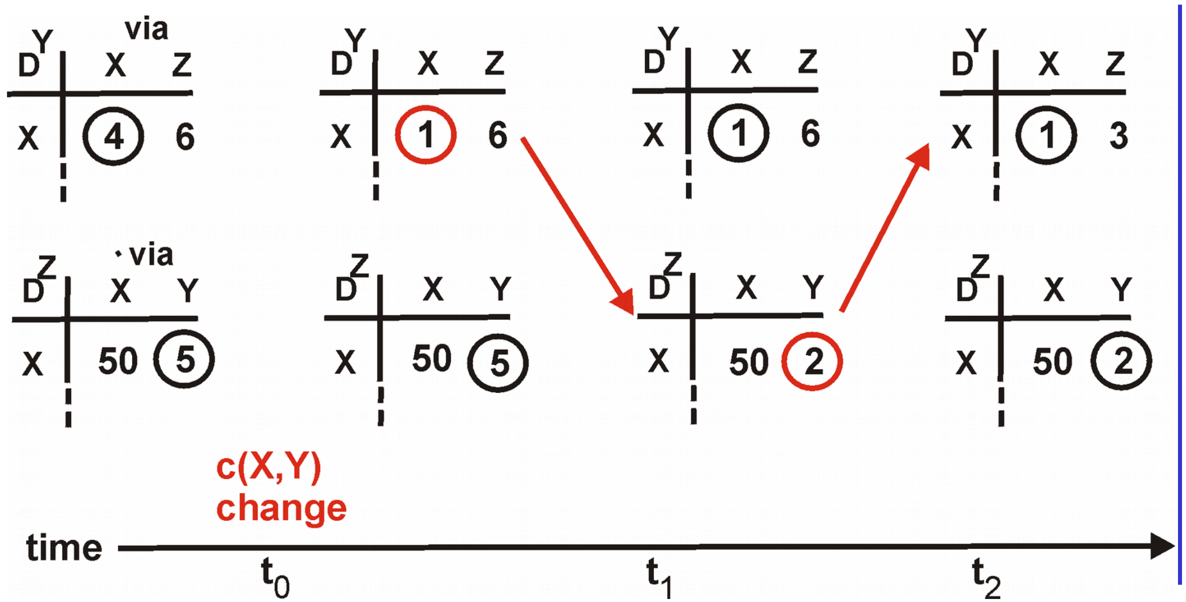
Distance Vector: link cost changes

Link cost changes:

node detects local link cost change
updates distance table (line 15)
if cost change in least cost path, notify neighbors (lines 23,24)



"good news travels fast"

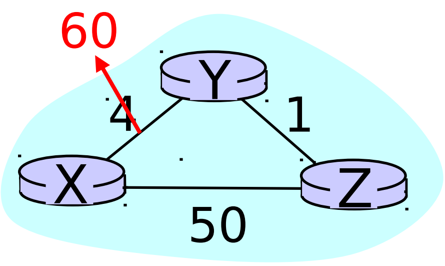


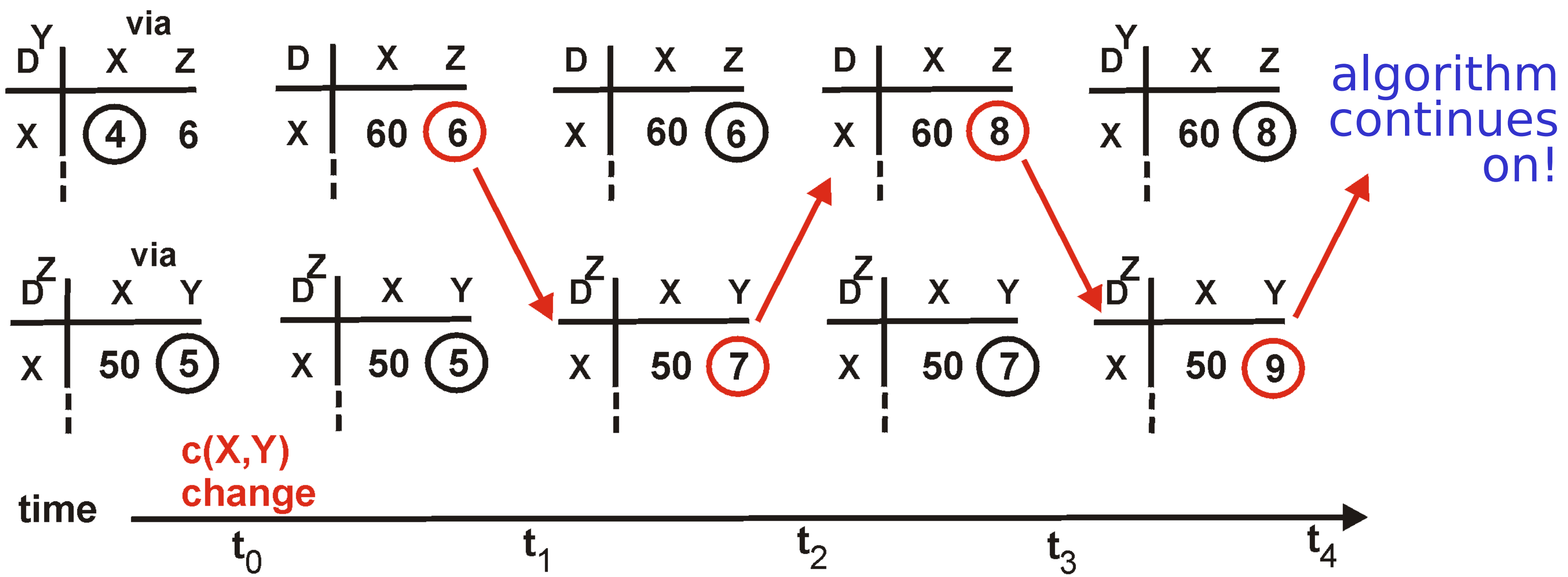
algorithm terminates

Distance Vector: link cost changes (2)

Link cost changes:

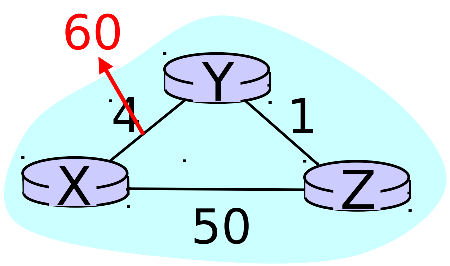
bad news travels slow - “count to infinity” problem!

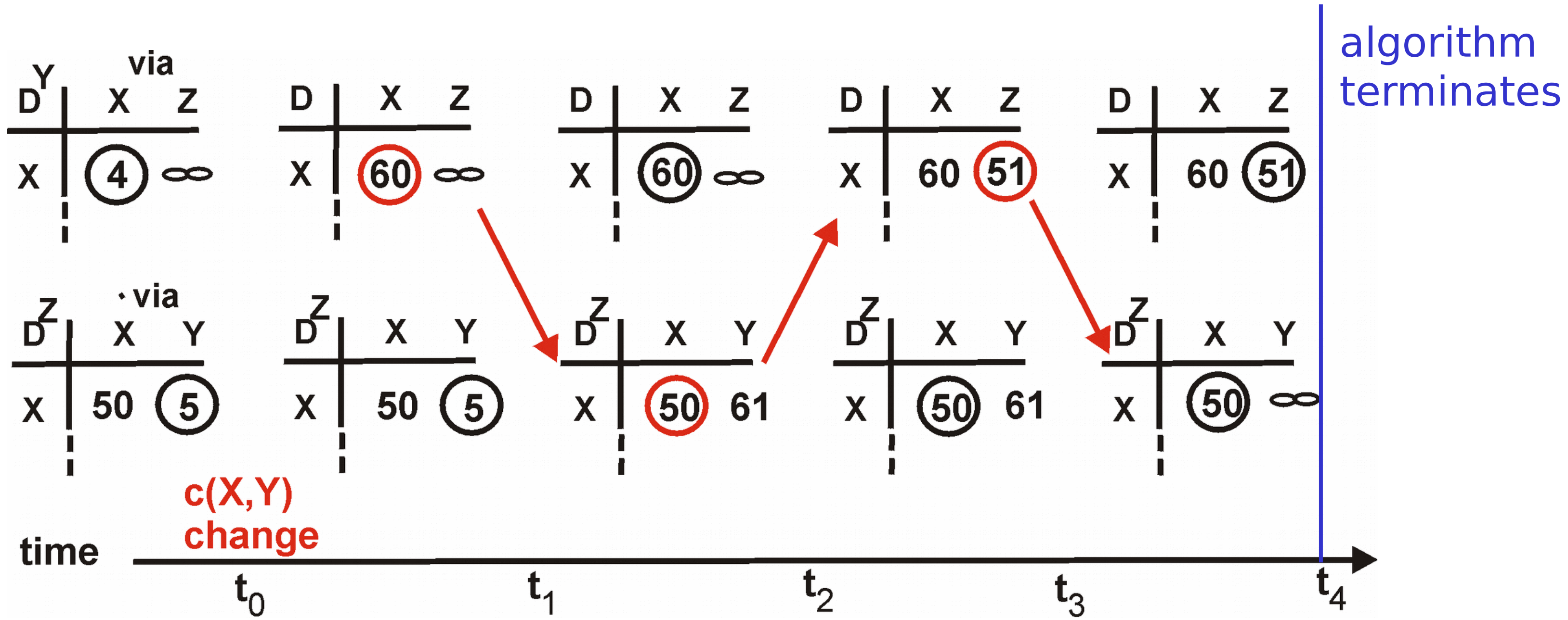




Distance Vector: poisoned reverse

- 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?

Administrative Distance

- Administrative Distances AD:is used to rate the trustworthiness of routing information received on a router from a neighbor router.
- An administrative distance is an integer from 0 to 255.
- Where 0 is the most trusted and 255 means no traffic will be passed via this route.

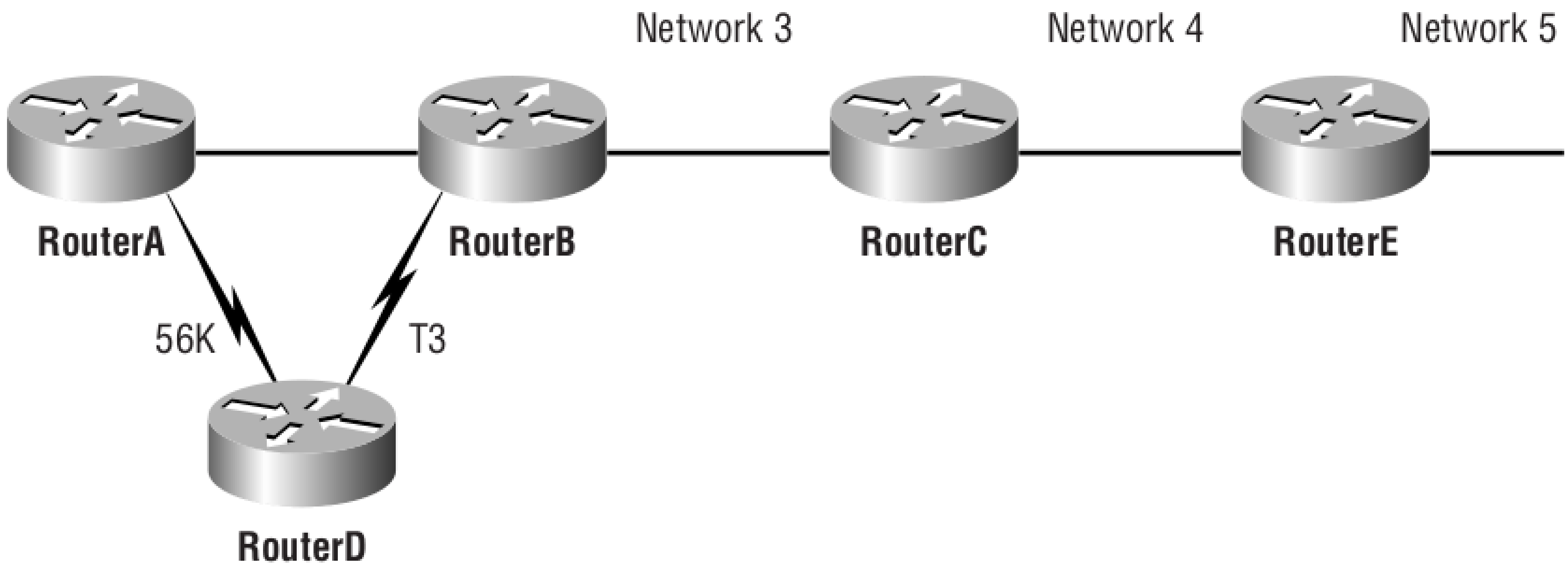
Administrative Distance

Route Source	Default AD
Connected interface	0
Static route	1
EIGRP	90
IGRP	100
OSPF	110
RIP	120
External EIGRP	170
Unknown	255 (this route will never be used)

Routing Loops

- Distance-vector routing protocols keep track of any changes to the internetwork by broadcasting periodic routing updates out all active interfaces.
- This broadcast includes the complete routing table.

Routing Loops



Maximum Hop Count

- The routing loop problem just described is called counting to infinity, and it’s caused by gossip (broadcasts).
- Wrong information being communicated and propagated throughout the internetwork.
- One way of solving this problem is to define a maximum hop count.
- Distance vector protocols have maximum hope counts.

Split Horizon

- Another solution to the routing loop problem is called split horizon.
- Distance-vector protocols solve this issue by enforcing the rule that routing information cannot be sent back in the direction from which it was received.
- This would have prevented Router A from sending the updated information it received from Router B back to Router B.

Route Poisoning

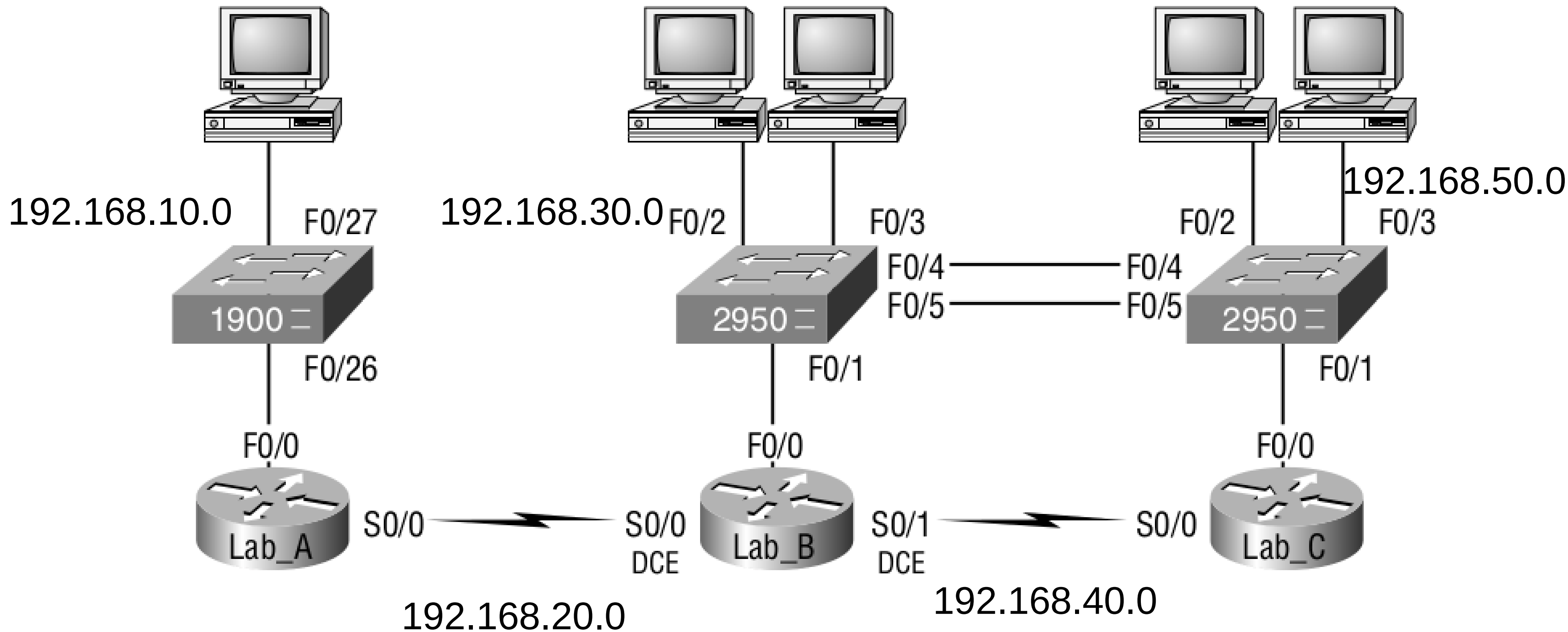
- when Network 5 goes down, Router E initiates route poisoning by advertising Network 5 as 16, or unreachable

Routing Information Protocol (RIP)

- RIP is a true distance-vector routing protocol.
- It sends the complete routing table out to all active interfaces every 30 seconds.
- RIP only uses hop count to determine the best way to a remote network.
- Maximum hops up to 15.
- 16 is infinite or unreachable.

Configuring RIP Routing

Topology



192.168.10.0
F0/27
1900
F0/26
F0/0
Lab_A
S0/0
192.168.30.0
F0/2
F0/3
F0/4
F0/5
2950
F0/1
F0/0
S0/0
DCE
Lab_B
S0/1
DCE
192.168.20.0
192.168.40.0
192.168.50.0
F0/2
F0/3
F0/4
F0/5
2950
F0/1
F0/0
S0/0
Lab_C

Holding Down RIP Propagations

- You probably don’t want your RIP network advertised everywhere on your LAN and WAN.
- There’s not a whole lot to be gained by advertising your RIP network to the Internet
- *passive-interface serial 0/0*