Chapter 4: outline

Part 2 (of 3): Routing (2)

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

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

then

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

cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

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:

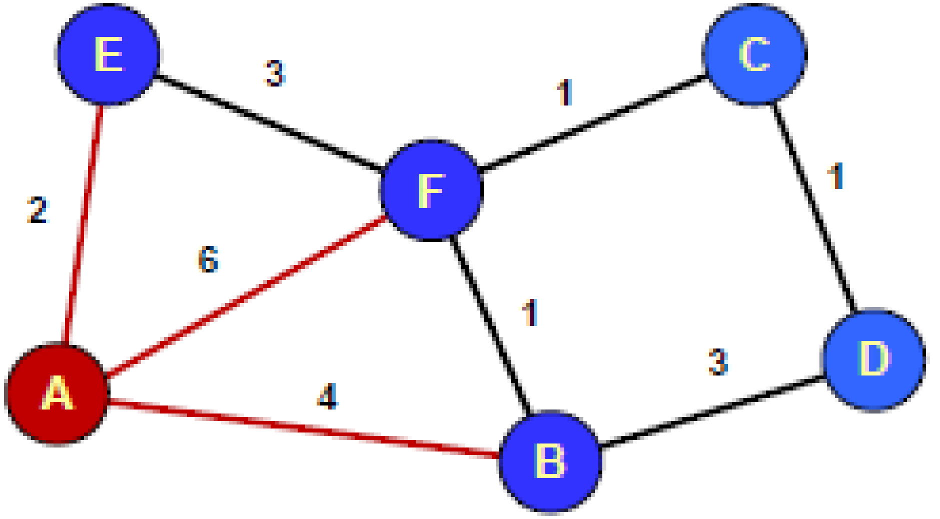
wait for (change in local link cost or msg from neighbor)

recompute estimates

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

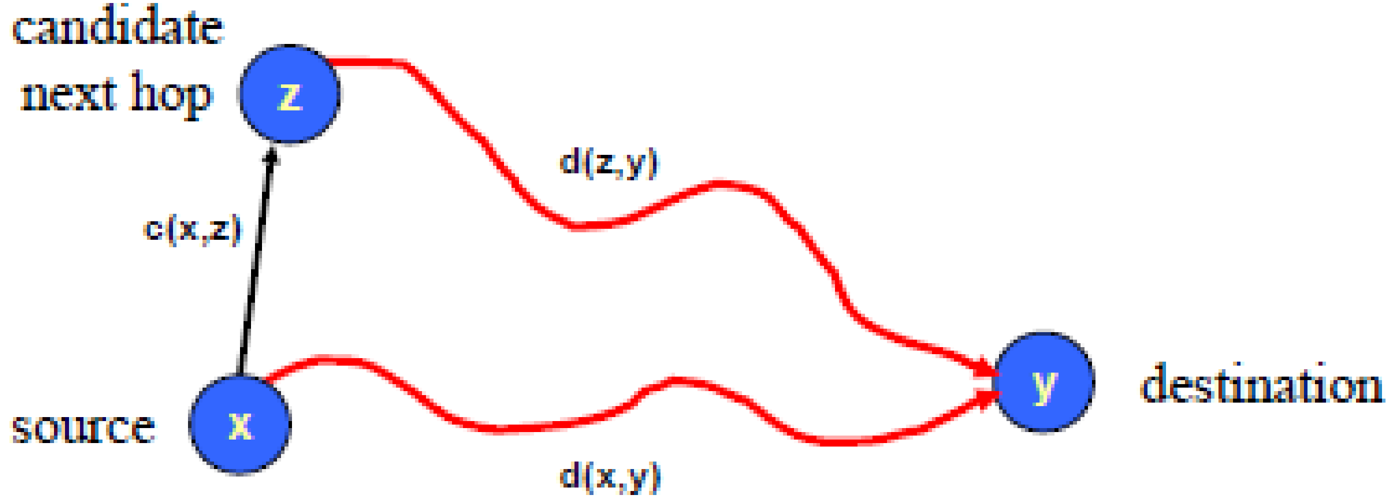
Distance-Vector Method

Initial Table for A		
Dest	**Cost**	**Next Hop**
A	0	A
B	4	B
C	∞	–
D	∞	–
E	2	E
F	6	F



- Idea
  - At any time, have cost/next hop of best known path to destination
  - Use cost ∞ when no path known
- Initially
  - Only have entries for directly connected nodes

Distance-Vector Update



- Update(x,y,z)

  d ← c(x,z) + d(z,y)   # Cost of path from x to y with first hop z

  if d < d(x,y)

  # Found better path

  return d,z   # Updated cost / next hop for destination y

  else

  return d(x,y), nexthop(x,y)   # Existing cost / next hop

Algorithm

- Bellman-Ford algorithm
- Repeat

  For every node x

  For every neighbor **z**

  For every destination y

  d(x,y) ← Update(x,y,**z**)
- Until converge

Start

Optimum 1-hop paths

Table for A			Table for B		
Dst	Cst	Hop	Dst	Cst	Hop
A	0	A	A	4	A
B	4	B	B	0	B
C	∞	–	C	∞	–
D	∞	–	D	3	D
E	2	E	E	∞	–
F	6	F	F	1	F

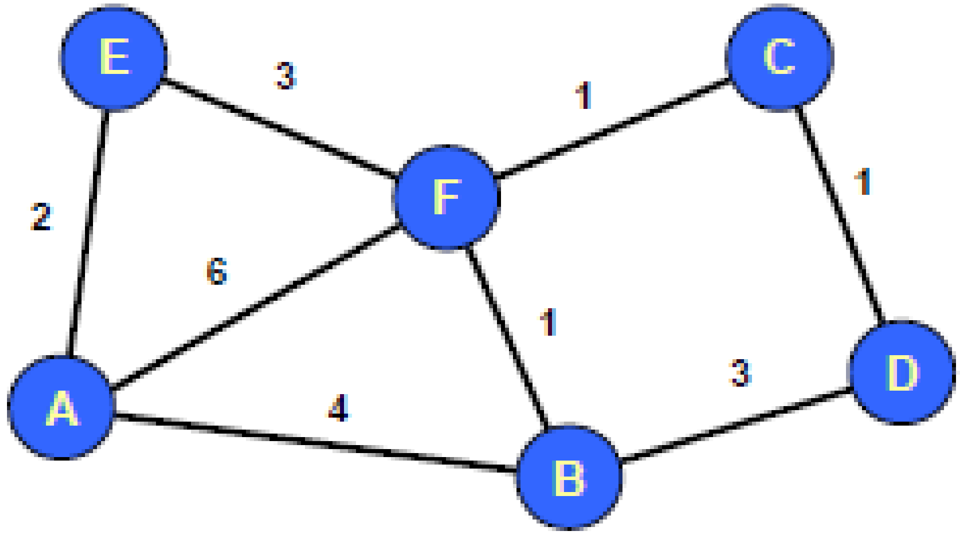


Table for C			Table for D			Table for E			Table for F		
Dst	Cst	Hop	Dst	Cst	Hop	Dst	Cst	Hop	Dst	Cst	Hop
A	∞	–	A	∞	–	A	2	A	A	6	A
B	∞	–	B	3	B	B	∞	–	B	1	B
C	0	C	C	1	C	C	∞	–	C	1	C
D	1	D	D	0	D	D	∞	–	D	∞	–
E	∞	–	E	∞	–	E	0	E	E	3	E
F	1	F	F	∞	–	F	3	F	F	0	F

Iteration #1

Optimum 2-hop paths

Table for A

Dst	Cst	Hop
A	0	A
B	4	B
C	7	F
D	7	B
E	2	E
F	5	E

Table for B

Dst	Cst	Hop
A	4	A
B	0	B
C	2	F
D	3	D
E	4	F
F	1	F

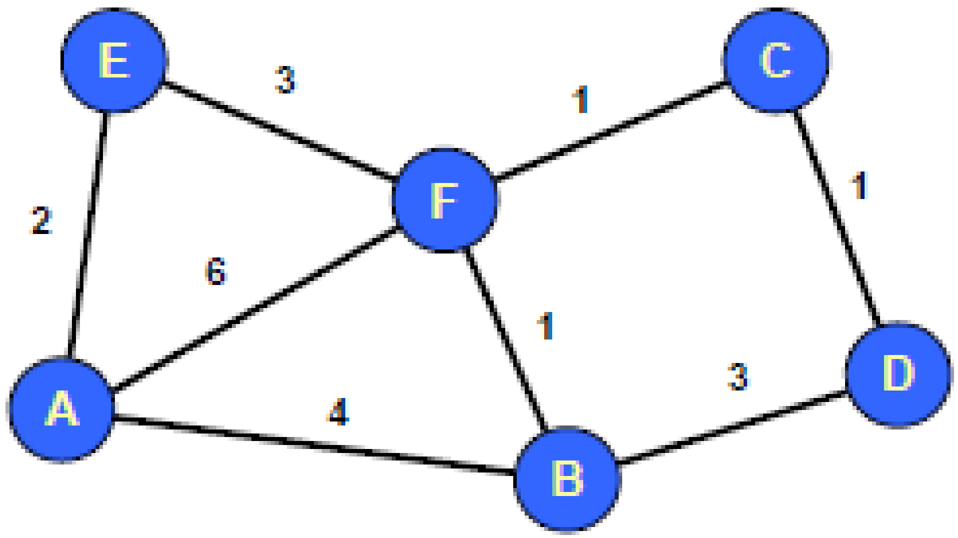


Table for C

Dst	Cst	Hop
A	7	F
B	2	F
C	0	C
D	1	D
E	4	F
F	1	F

Table for D

Dst	Cst	Hop
A	7	B
B	3	B
C	1	C
D	0	D
E	∞	–
F	2	C

Table for E

Dst	Cst	Hop
A	2	A
B	4	F
C	4	F
D	∞	–
E	0	E
F	3	F

Table for F

Dst	Cst	Hop
A	5	B
B	1	B
C	1	C
D	2	C
E	3	E
F	0	F

Iteration #2

Optimum 3-hop paths

Table for A

Dst	Cst	Hop
A	0	A
B	4	B
C	6	E
D	7	B
E	2	E
F	5	E

Table for B

Dst	Cst	Hop
A	4	A
B	0	B
C	2	F
D	3	D
E	4	F
F	1	F

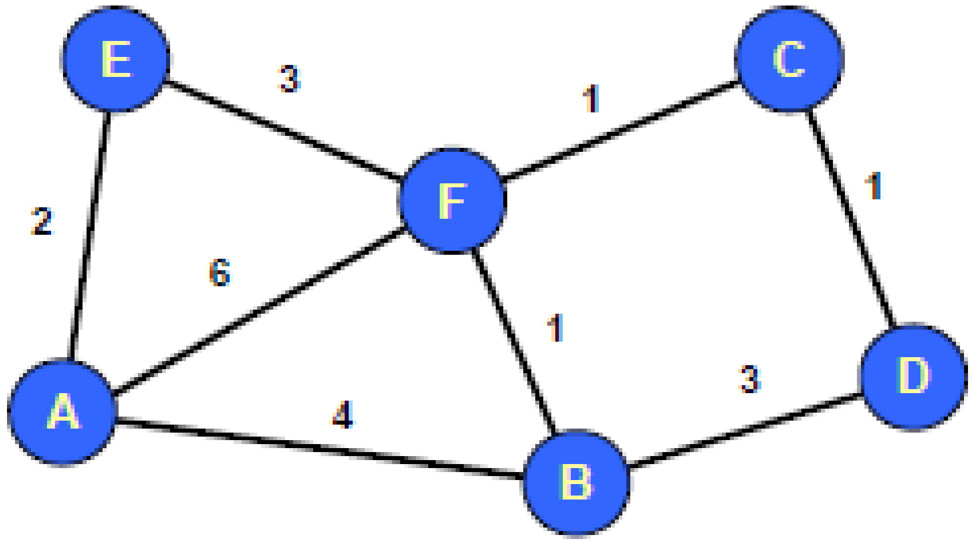


Table for C

Dst	Cst	Hop
A	6	F
B	2	F
C	0	C
D	1	D
E	4	F
F	1	F

Table for D

Dst	Cst	Hop
A	7	B
B	3	B
C	1	C
D	0	D
E	5	C
F	2	C

Table for E

Dst	Cst	Hop
A	2	A
B	4	F
C	4	F
D	5	F
E	0	E
F	3	F

Table for F

Dst	Cst	Hop
A	5	B
B	1	B
C	1	C
D	2	C
E	3	E
F	0	F

Distance Vector: Link Cost Changes

Link cost changes:

- Node detects local link cost change
- Updates distance table
- If cost change in least cost path, notify neighbors

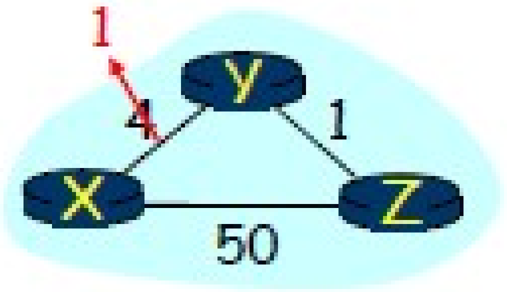


Table at

Node Y

D^Y	via X	via Z
x	4	6

Node Z

D^Z	via X	via Y
x	50	5

c(X,Y) change

time t_0 t_1 t_2

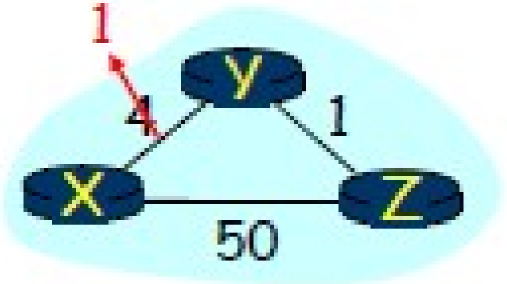
algorithm terminates

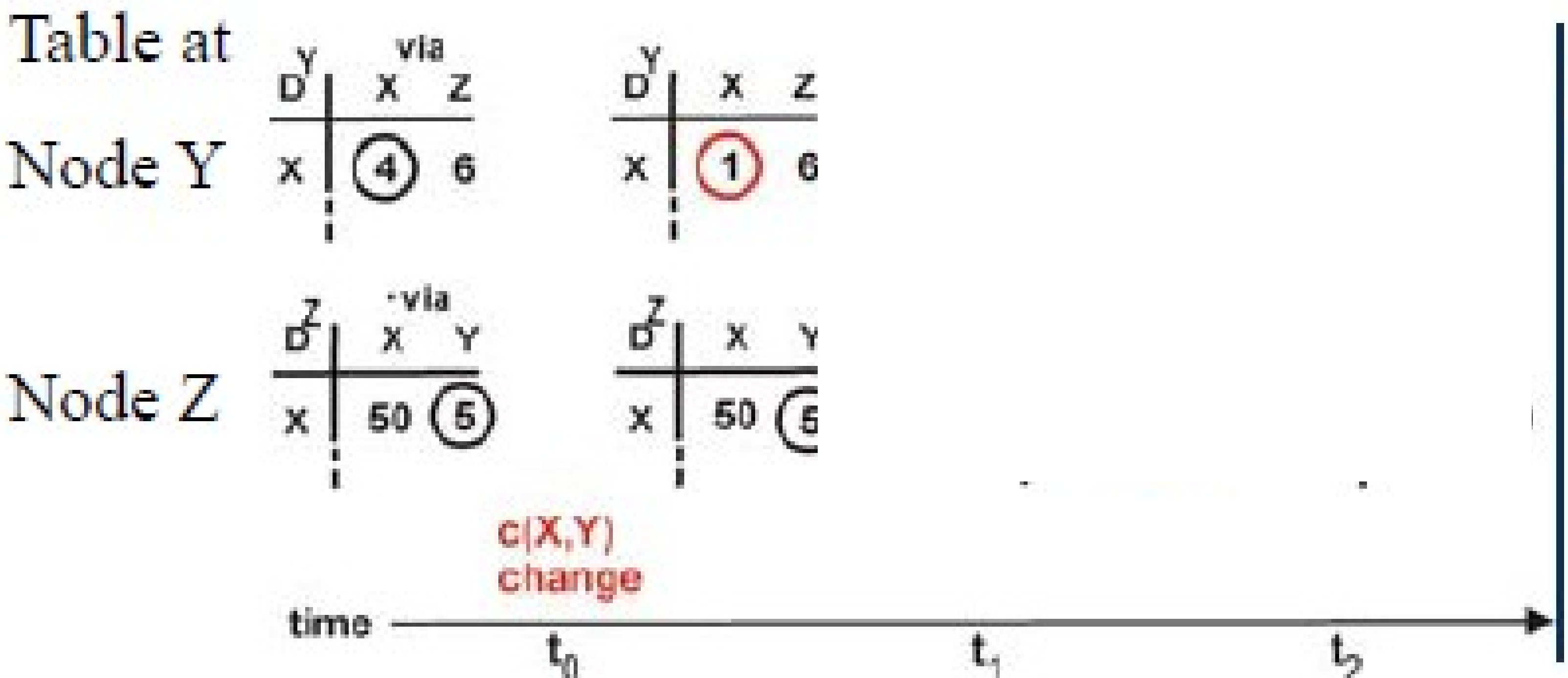
"good news travel fast"

Distance Vector: Link Cost Changes

Link cost changes:

- Node detects local link cost change
- Updates distance table
- If cost change in least cost path, notify neighbors



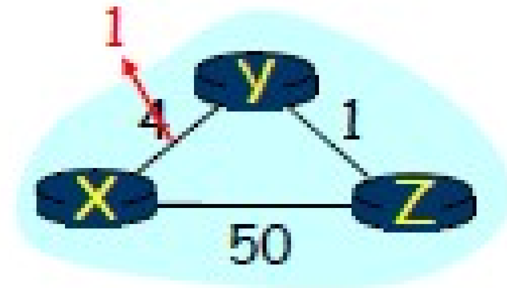


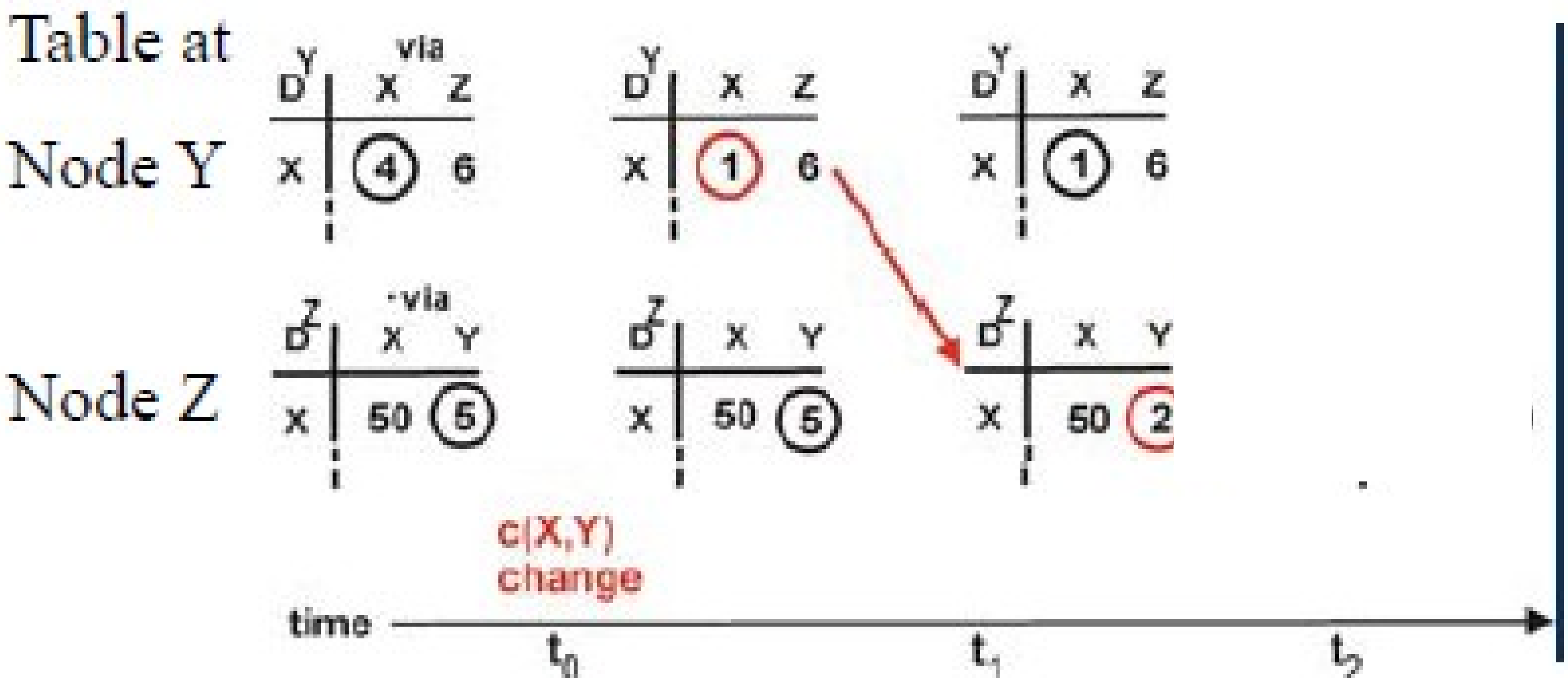
"good news travel fast"

Distance Vector: Link Cost Changes

Link cost changes:

- Node detects local link cost change
- Updates distance table
- If cost change in least cost path, notify neighbors





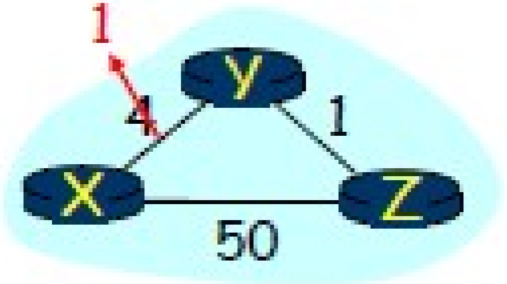
algorithm terminates

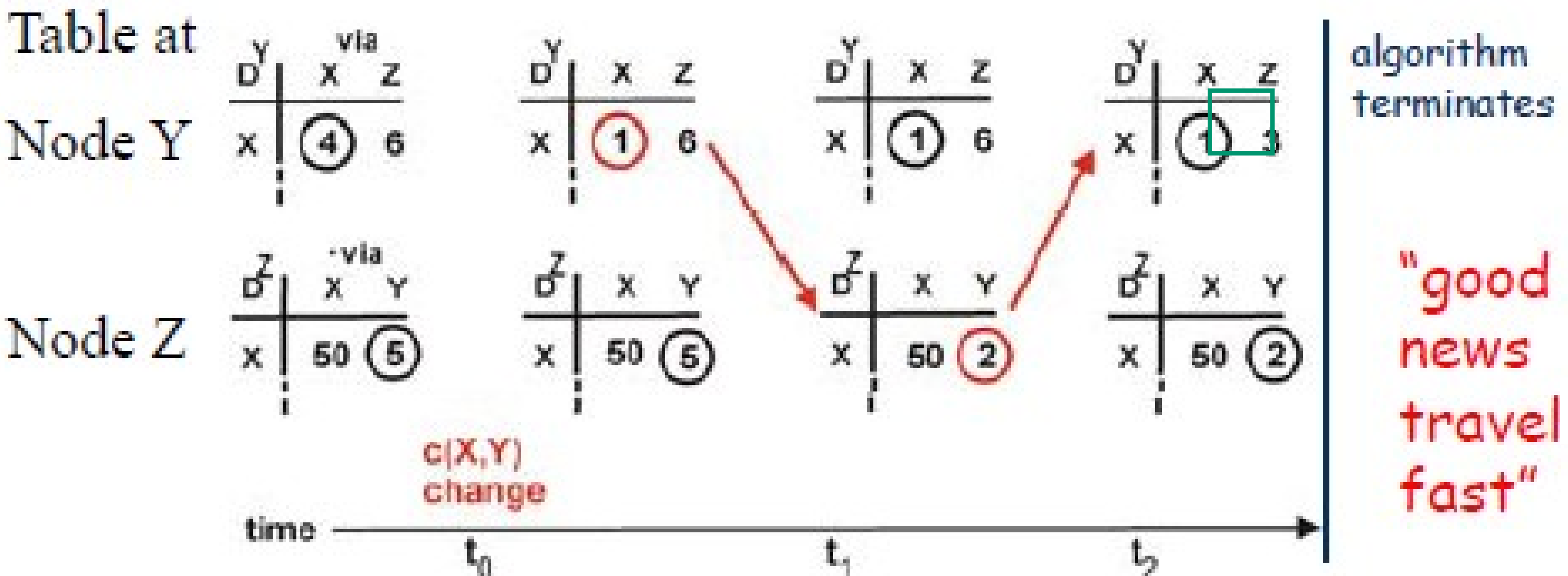
"good news travel fast"

Distance Vector: Link Cost Changes

Link cost changes:

- Node detects local link cost change
- Updates distance table
- If cost change in least cost path, notify neighbors

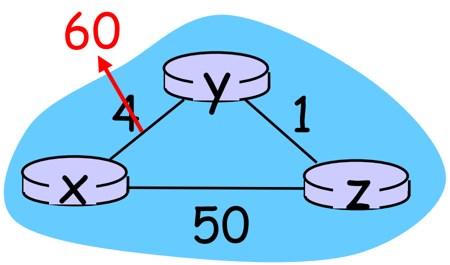




Distance vector: link cost changes

link cost changes:

- node detects local link cost change
- *bad news travels slow* - "count to infinity" problem!
- 44 iterations before algorithm stabilizes



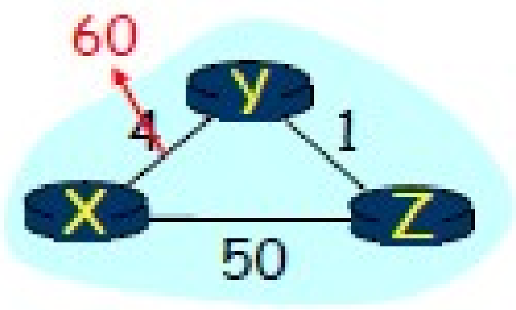
poisoned reverse:

- 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?

Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slowly - "count to infinity" problem!



Y

D^Y	via X	via Z
x	4	6

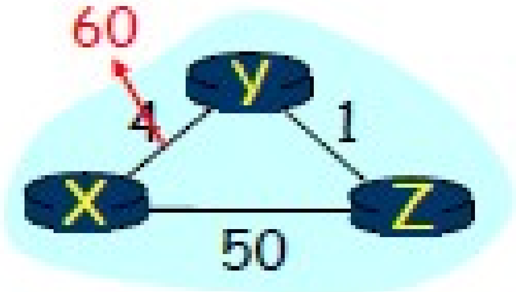
Z

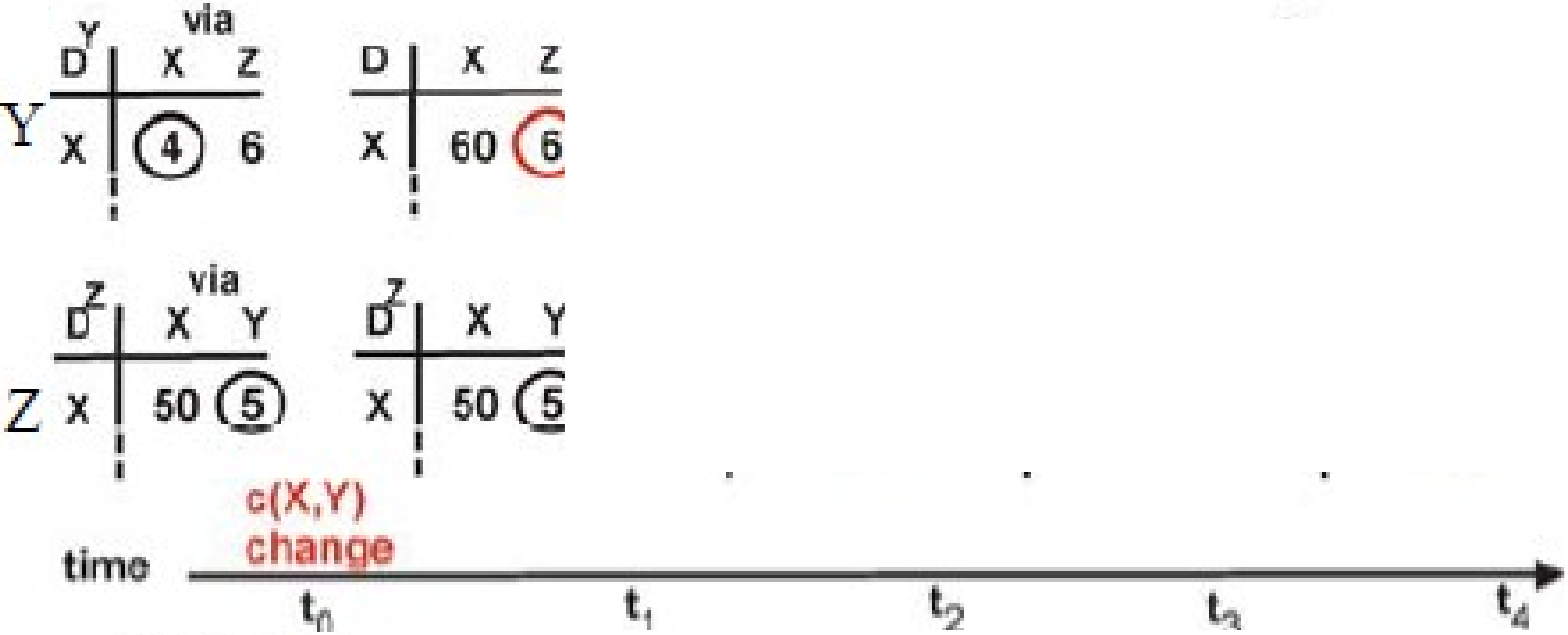
D^Z	via X	via Y
x	50	5

Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slowly - "count to infinity" problem!

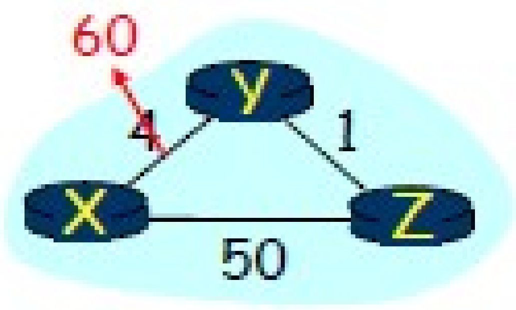


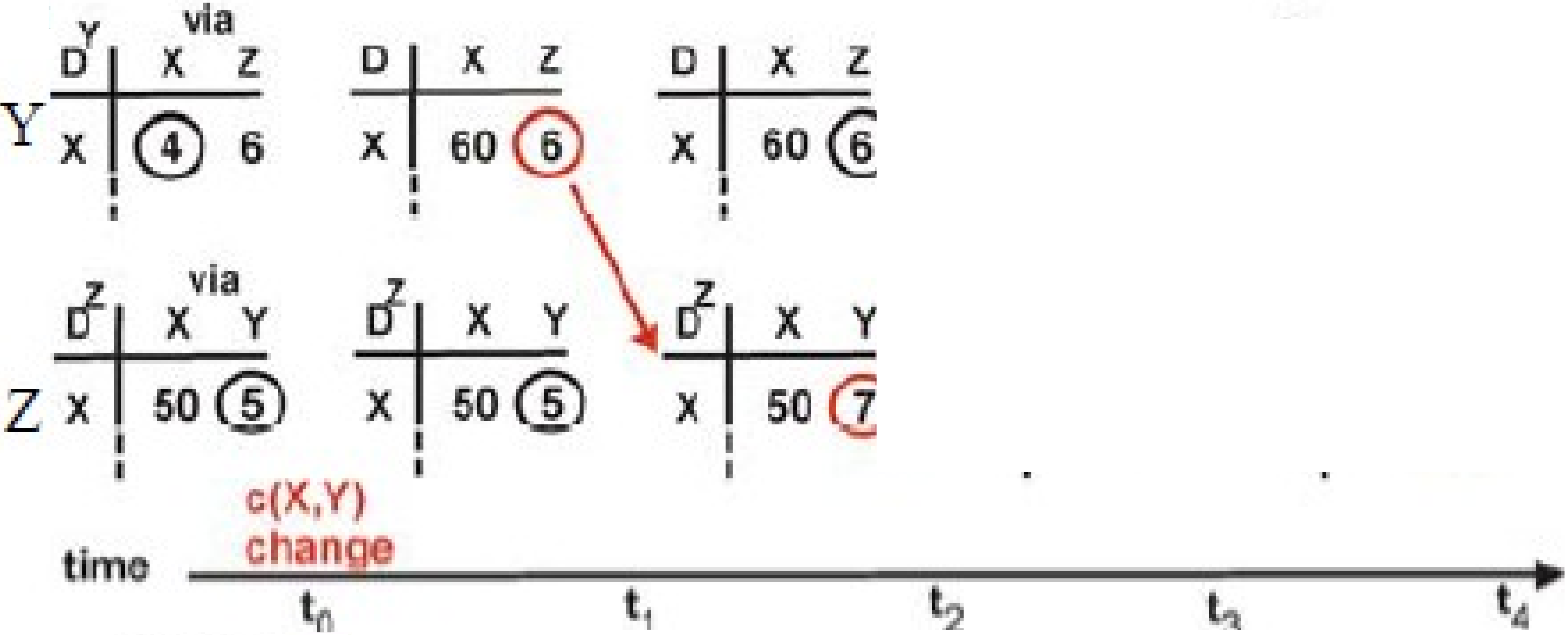


Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slowly - "count to infinity" problem!

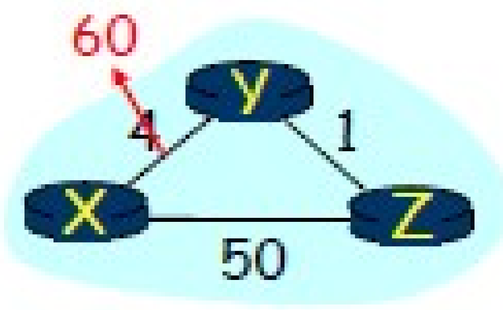


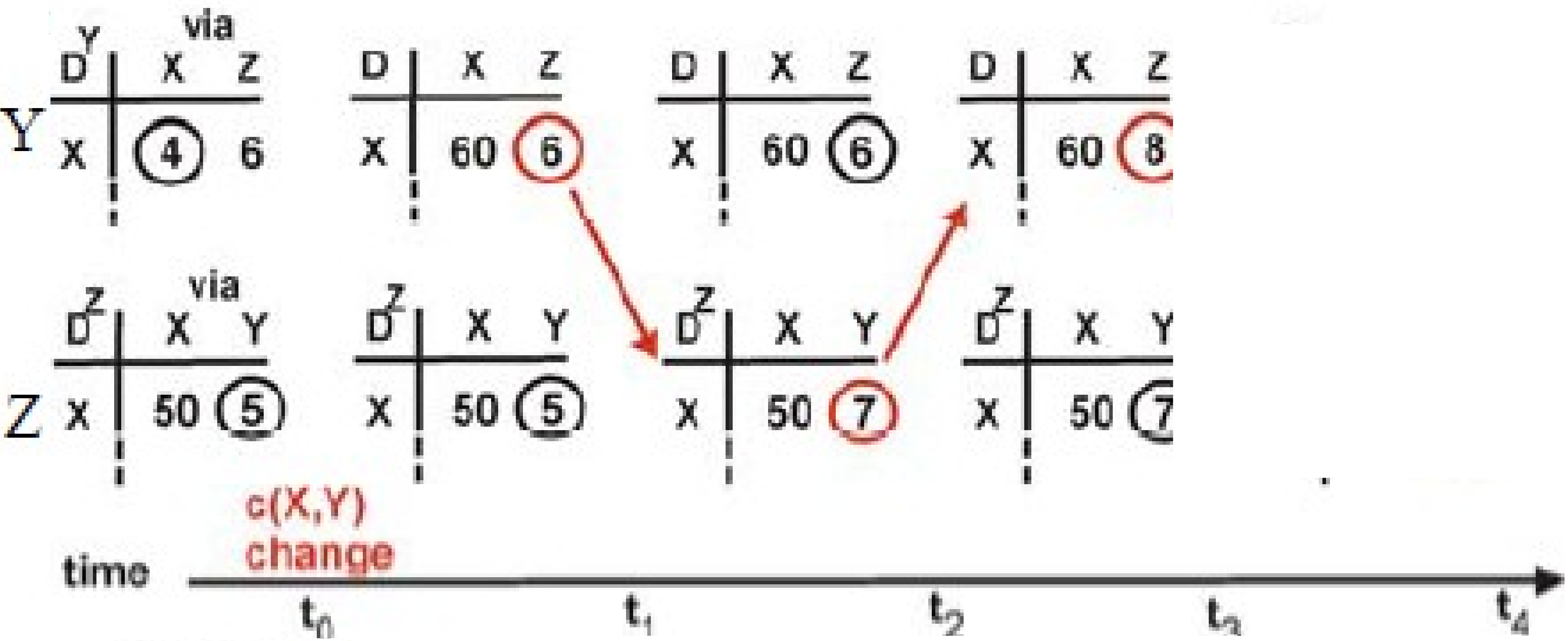


Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slowly - "count to infinity" problem!

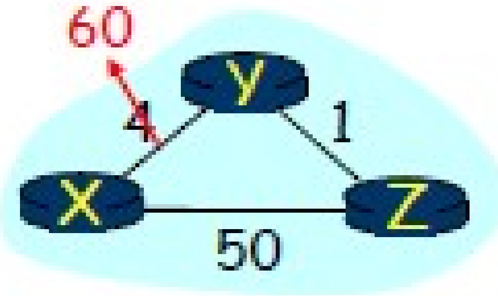


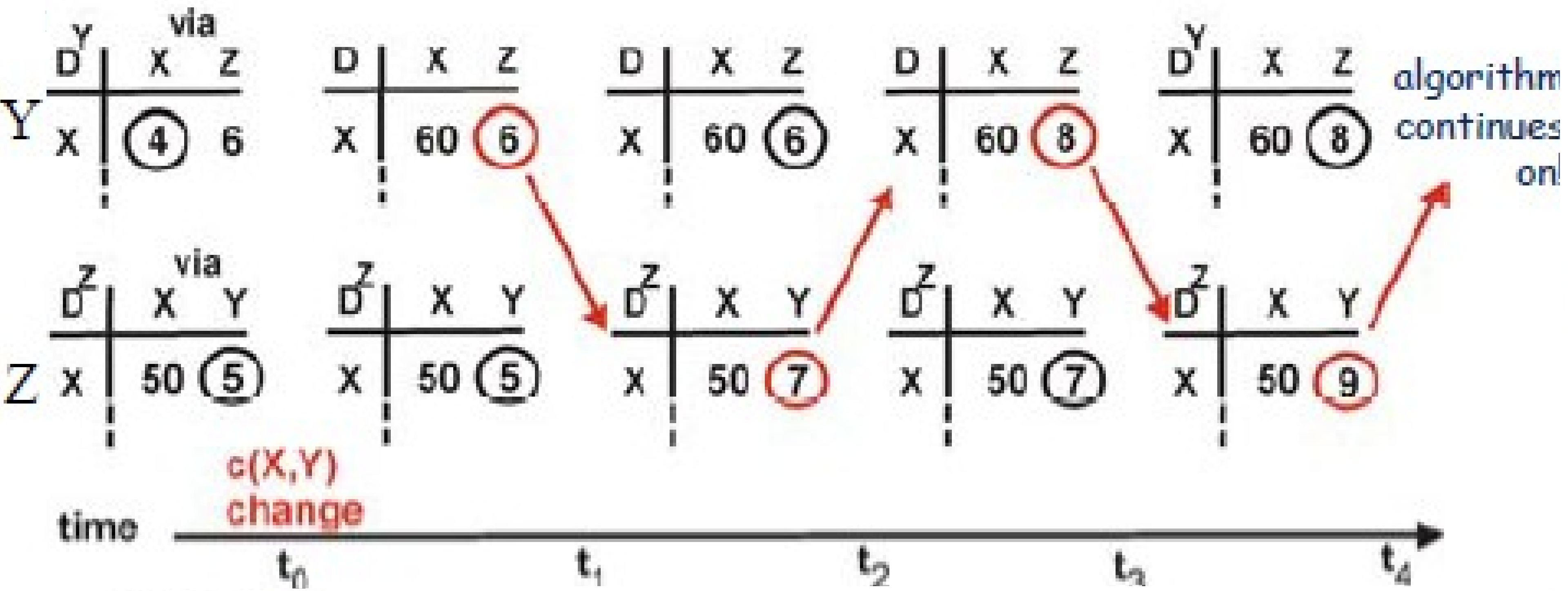


Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slowly - "count to infinity" problem!





Bad News Travels Slowly

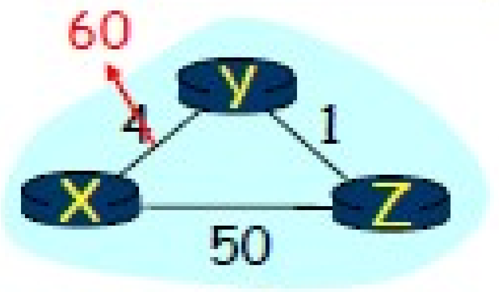
Is this a problem? Yes!

- After a path cost increases, it can take a very long time before paths stabilize, and
- During this process, the network has a routing loop

What is the cause?

- Nodes refuse to accept the up-to-date information, because they prefer the older, better cost
- Outdated information based on the older, lower path cost loops around the network

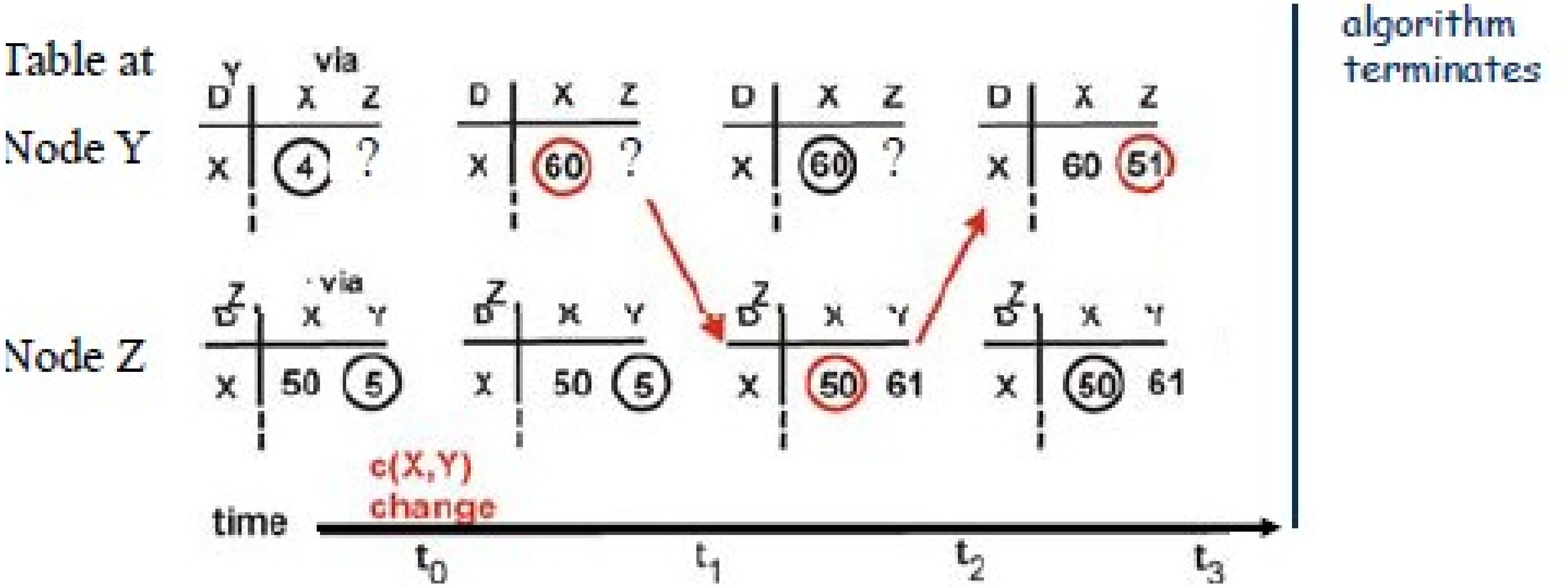
Distance Vector: Split Horizon



Problem: if Z routes through Y to get to X, it still advertises its path back to Y

- This serves no purpose and causes the loops

Solution: Z does not advertise its route back to Y



- What if a link becomes completely unavailable?

Poison Reverse Failures

Table for A			Table for B			Table for D			Table for F		
Dst	Cst	Hop	Dst	Cst	Hop	Dst	Cst	Hop	Dst	Cst	Hop
C	7	F	C	8	A	C	9	B	C	1	C

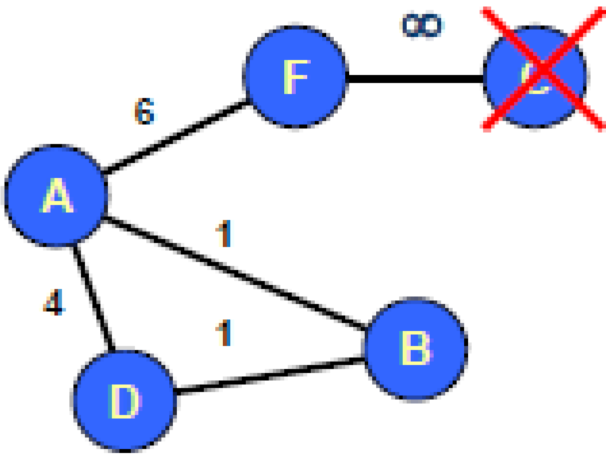


Table for A		
Dst	Cst	Hop
C	∞	–

Forced Update

Forced Update

Table for F		
Dst	Cst	Hop
C	∞	–

Table for A		
Dst	Cst	Hop
C	13	D

Better Route

Forced Update

Table for B		
Dst	Cst	Hop
C	14	A

Forced Update

Table for D		
Dst	Cst	Hop
C	15	B

Table for A		
Dst	Cst	Hop
C	19	D

Forced Update

•
•
•

- Split horizon does not help!
- Especially bad if a link goes down: "Count to infinity"
- Solution:
  - Make "infinity" smaller
  - Force the cost "infinity" to all interfaces and wait
  - Helps network converge faster

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

Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Hierarchical routing

our routing study thus far - idealization

- all routers identical
- network "flat"

… *not* true in practice

scale: with 600 million destinations:

- can't store all dest's in routing tables!
- routing table exchange would swamp links!

administrative autonomy

- internet = network of networks
- each network admin may want to control routing in its own network

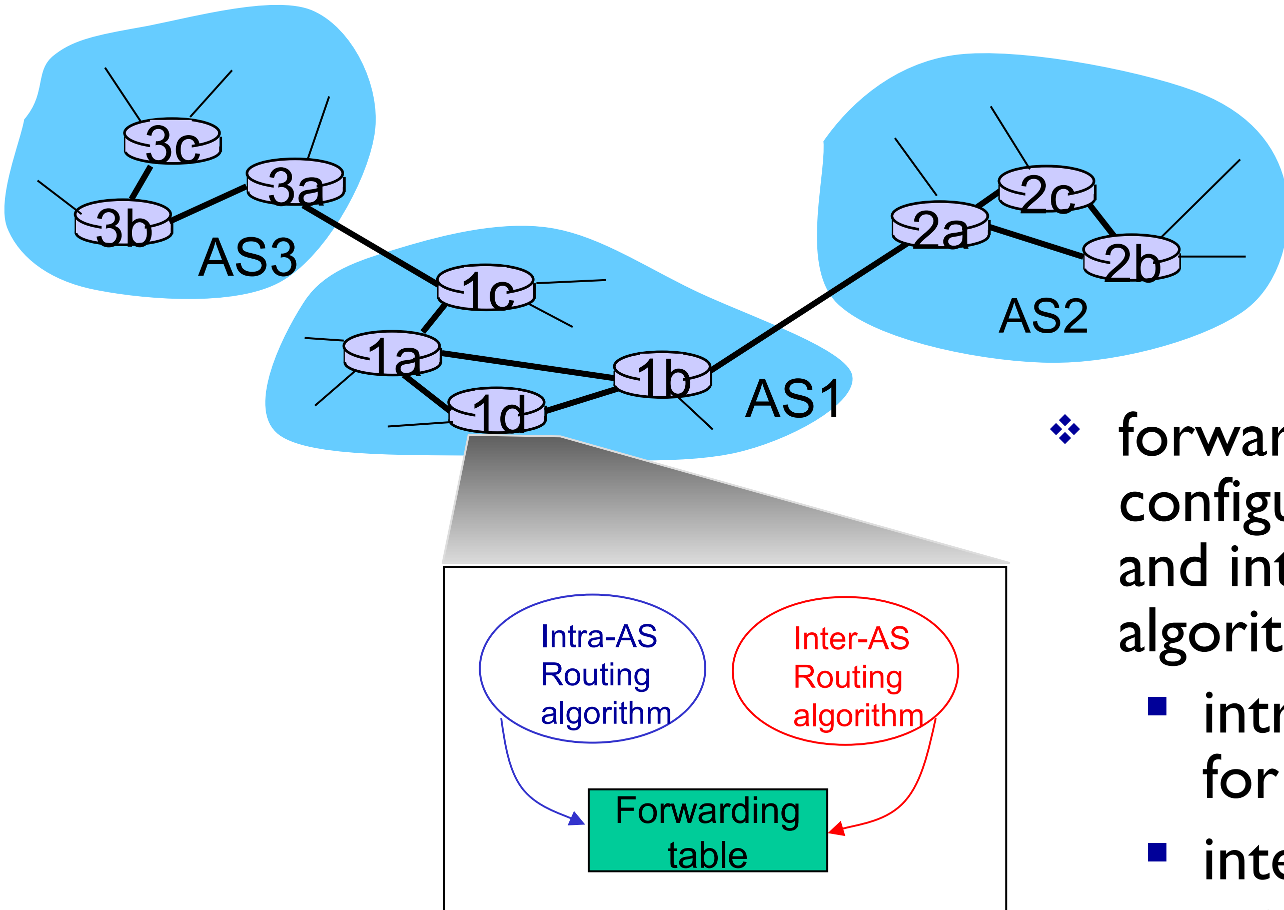
Hierarchical routing

- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

gateway router:

- at "edge" of its own AS
- has link to router in another AS

Interconnected ASes



- forwarding table configured by both intra- and inter-AS routing algorithm
  - intra-AS sets entries for internal dests
  - inter-AS & intra-AS sets entries for external dests

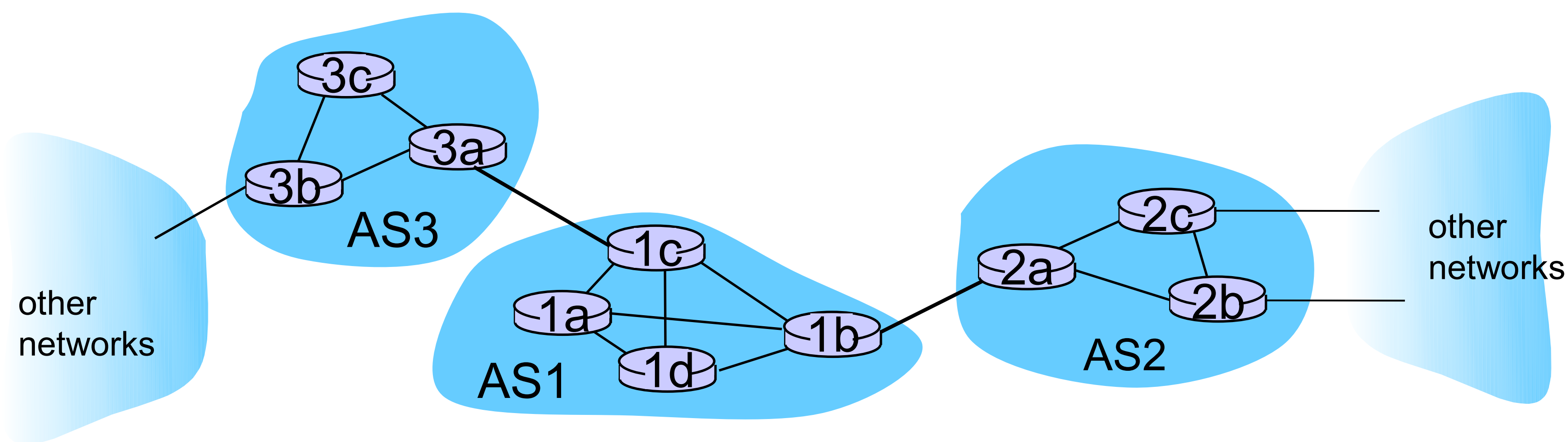
Inter-AS tasks

- suppose router in AS1 receives datagram destined outside of AS1:
  - router should forward packet to gateway router, but which one?

AS1 must:

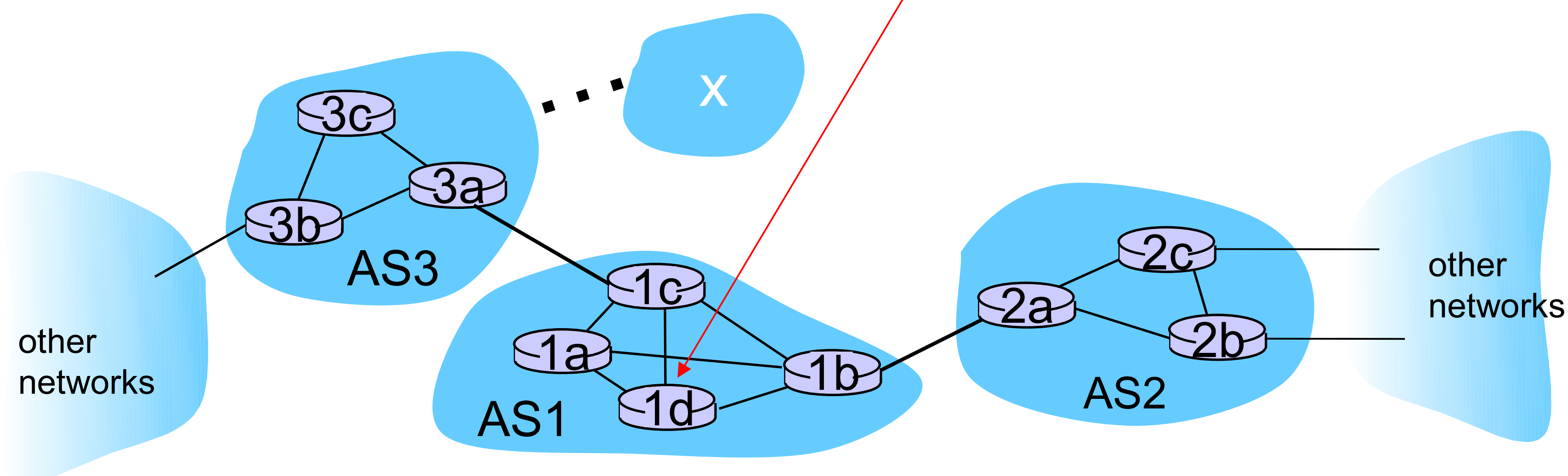
1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

job of inter-AS routing!



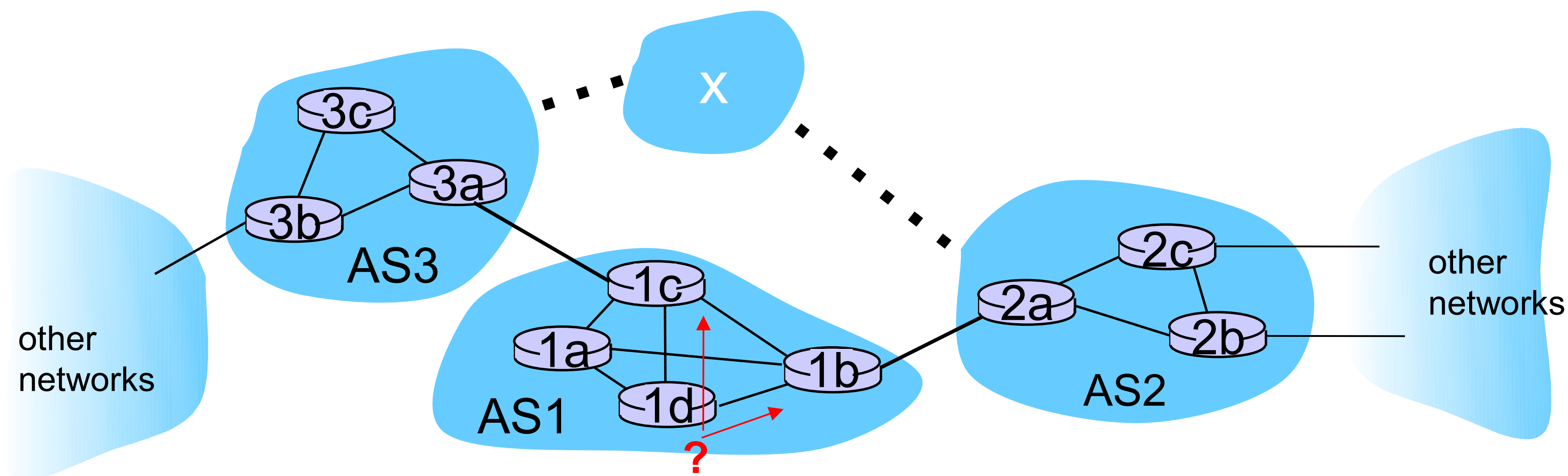
Example: setting forwarding table in router 1d

- suppose AS1 learns (via inter-AS protocol) that subnet *x* reachable via AS3 (gateway 1c), but not via AS2
  - inter-AS protocol propagates reachability info to all internal routers
- router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c
  - installs forwarding table entry *(x,I)*



Example: choosing among multiple ASes

- now suppose AS1 learns from inter-AS protocol that subnet *x* is reachable from AS3 *and* from AS2.
- to configure forwarding table, router 1d must determine which gateway it should forward packets towards for dest x
  - this is also job of inter-AS routing protocol!



Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

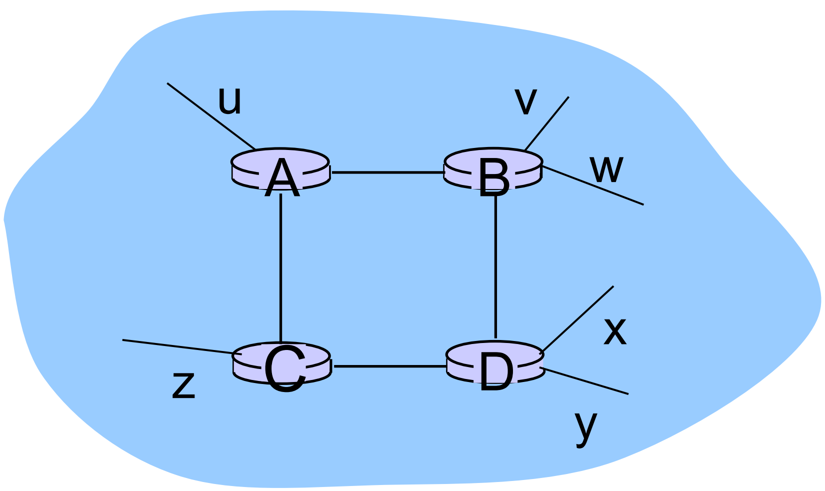
4.7 broadcast and multicast routing

Intra-AS Routing

- also known as *interior gateway protocols (IGP)*
- most common intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

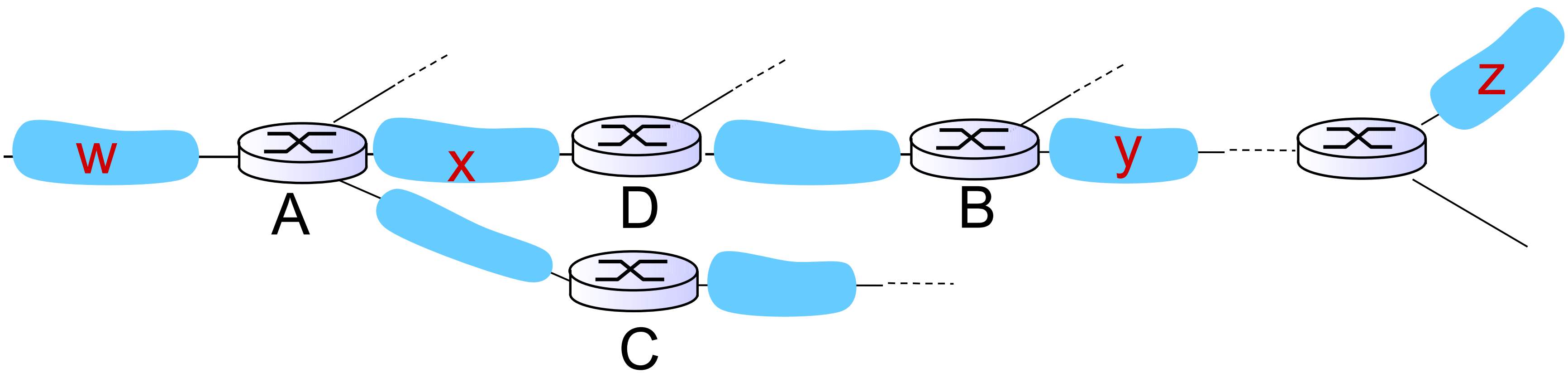
- included in BSD-UNIX distribution in 1982
- distance vector algorithm
  - distance metric: # hops (max = 15 hops), each link has cost 1
  - DVs exchanged with neighbors every 30 sec in response message (aka advertisement)
  - each advertisement: list of up to 25 destination *subnets* *(in IP addressing sense)*



from router A to destination *subnets:*

subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP: example



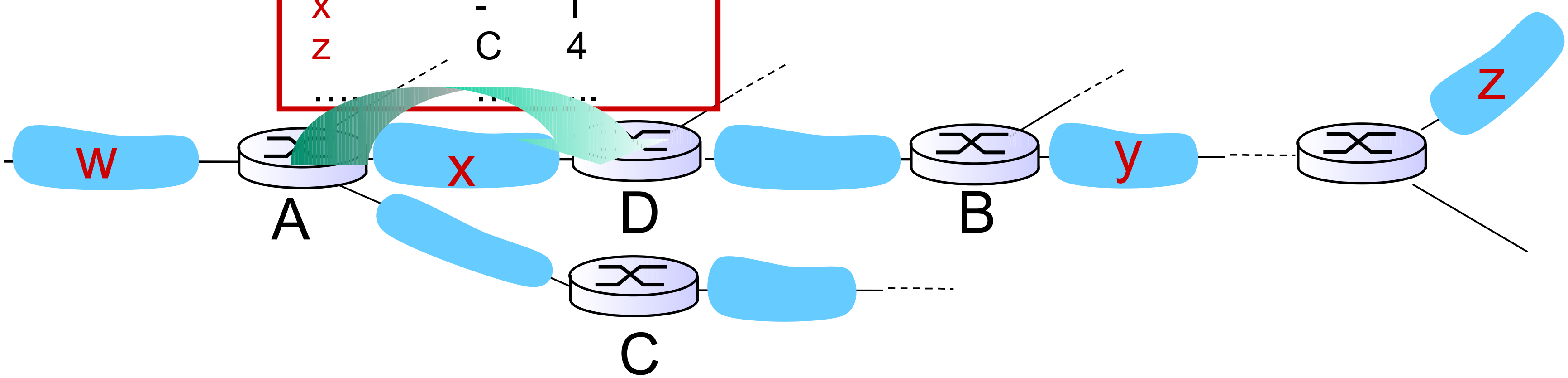
routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
….	….	….

RIP: example

A-to-D advertisement

dest	next	hops
w	-	1
x	-	1
z	C	4
….	…	...



routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
….	….	….

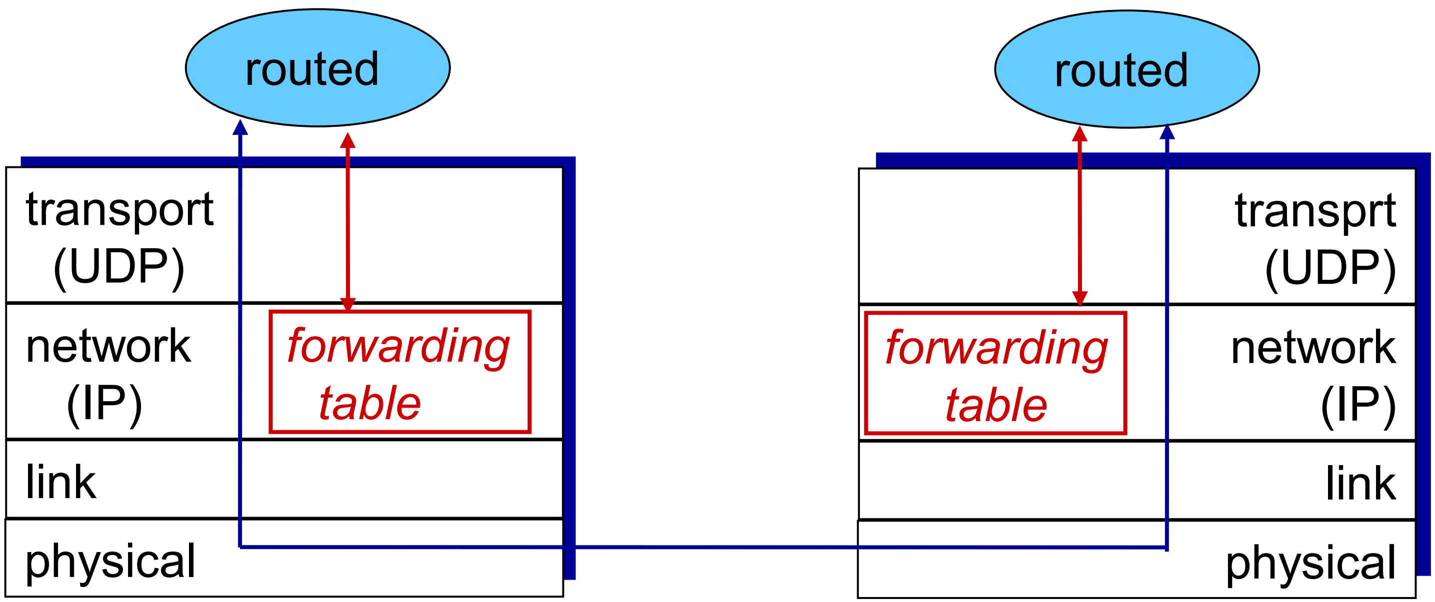
RIP: link failure, recovery

if no advertisement heard after 180 sec --> neighbor/link declared dead

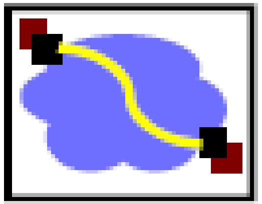
- routes via neighbor invalidated
- new advertisements sent to neighbors
- neighbors in turn send out new advertisements (if tables changed)
- link failure info quickly (?) propagates to entire net
- *poison reverse* used to prevent ping-pong loops (infinite distance = 16 hops)

RIP table processing

- RIP routing tables managed by *application-level* process called route-d (daemon)
- advertisements sent in UDP packets, periodically repeated

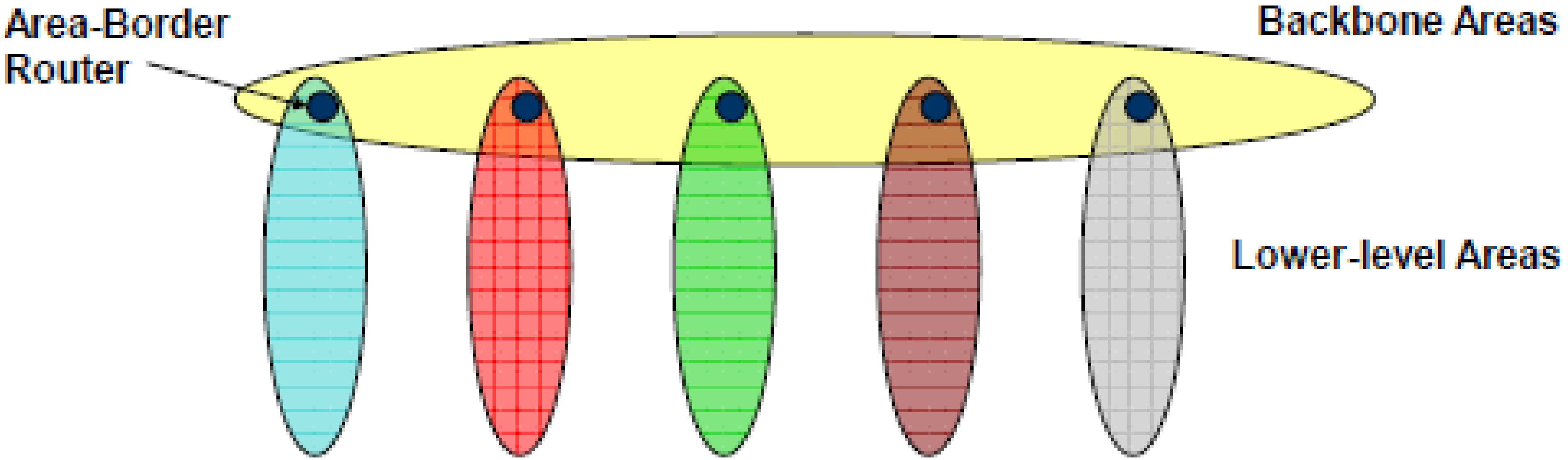


OSPF Routing Protocol



- Open standard created by IETF
- Shortest-path first
  - Another name for Dijkstra's algorithm
- Replaced RIP
  - RIP is dated, given today's requirements
  - OSPF has fast convergence when configuration changes
  - OSPF can scale to very large networks using "areas"

Areas: Scaling to Larger Networks



- Within area: Each node has routes to every other node
- Outside area: Each node has routes for other top-level areas only
  - Inter-area packets are routed to nearest border router
  - Constraint: no path between two sub-areas of an area can exit that area
  - May no longer have shortest path routes

OSPF (Open Shortest Path First)

- "open": publicly available
- uses link state algorithm
  - LS packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm
- OSPF advertisement carries one entry per neighbor
- advertisements flooded to *entire* AS
  - carried in OSPF messages directly over IP (rather than TCP or UDP
- *IS-IS routing* protocol: nearly identical to OSPF

OSPF "advanced" features (not in RIP)

- *security:* all OSPF messages authenticated (to prevent malicious intrusion)
- multiple same-cost paths allowed (only one path in RIP)
- for each link, multiple cost metrics for different TOS (e.g., satellite link cost set "low" for best effort ToS; high for real time ToS)
- integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- hierarchical OSPF in large domains.