University of BRISTOL

Advanced Networks

(EENGM4211)

Rasheed Hussain

rasheed.hussain@bristol.ac.uk

Best contact : Microsoft Team

bristol.ac.uk

Slides courtesy: Prof. Reza Nejabati and Slides from Ross's Book

University of BRISTOL

Unit Outline

1. Introduction
2. Internet Routing and Switching
3. IP Multicast
4. Networking for Real-time Applications
5. Routing in Wireless Networks
6. Quality of Service

University of BRISTOL

Learning outcomes

- At the end of the lecture, the students will be able to:
  - Demonstrate their knowledge of the routing
  - Discuss the rationale for and benefits of distance vector routing algorithm
  - Understand and demonstrate the knowledge of Bellman-Ford routing algorithm
  - Understand the difference between routing and switching
  - Discuss the procedures by which routing tables are built up by the routers

University of BRISTOL

Internet Routing

Objective

- Describe the major routing/switching mechanisms used in routing in the Internet - unicast, multicast, switching

Outcomes

- Describe and explain routing algorithms and protocols
- Compare routing and switching
- Illustrate the above through examples

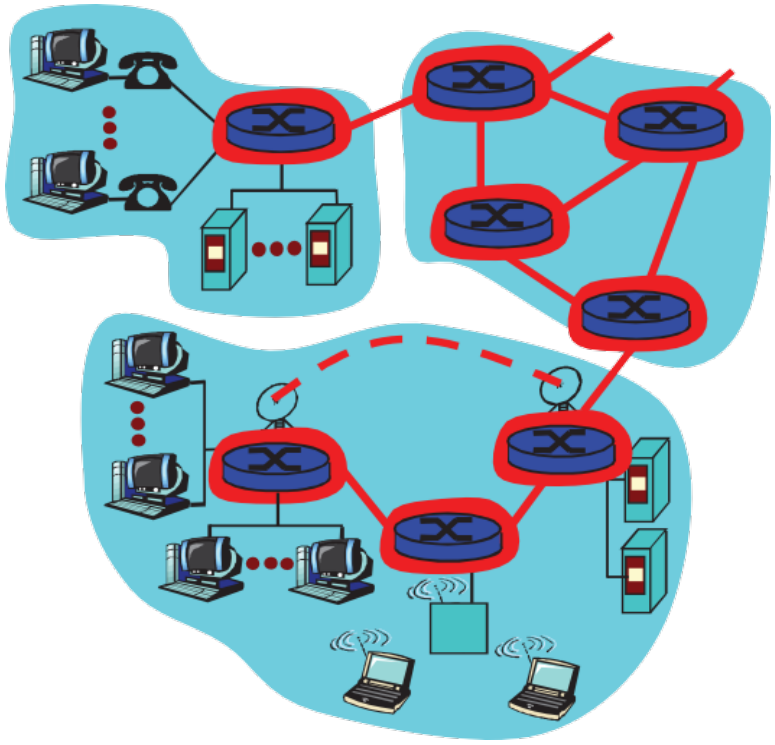
University of BRISTOL

Part 2:
Internet Routing and Switching

University of BRISTOL

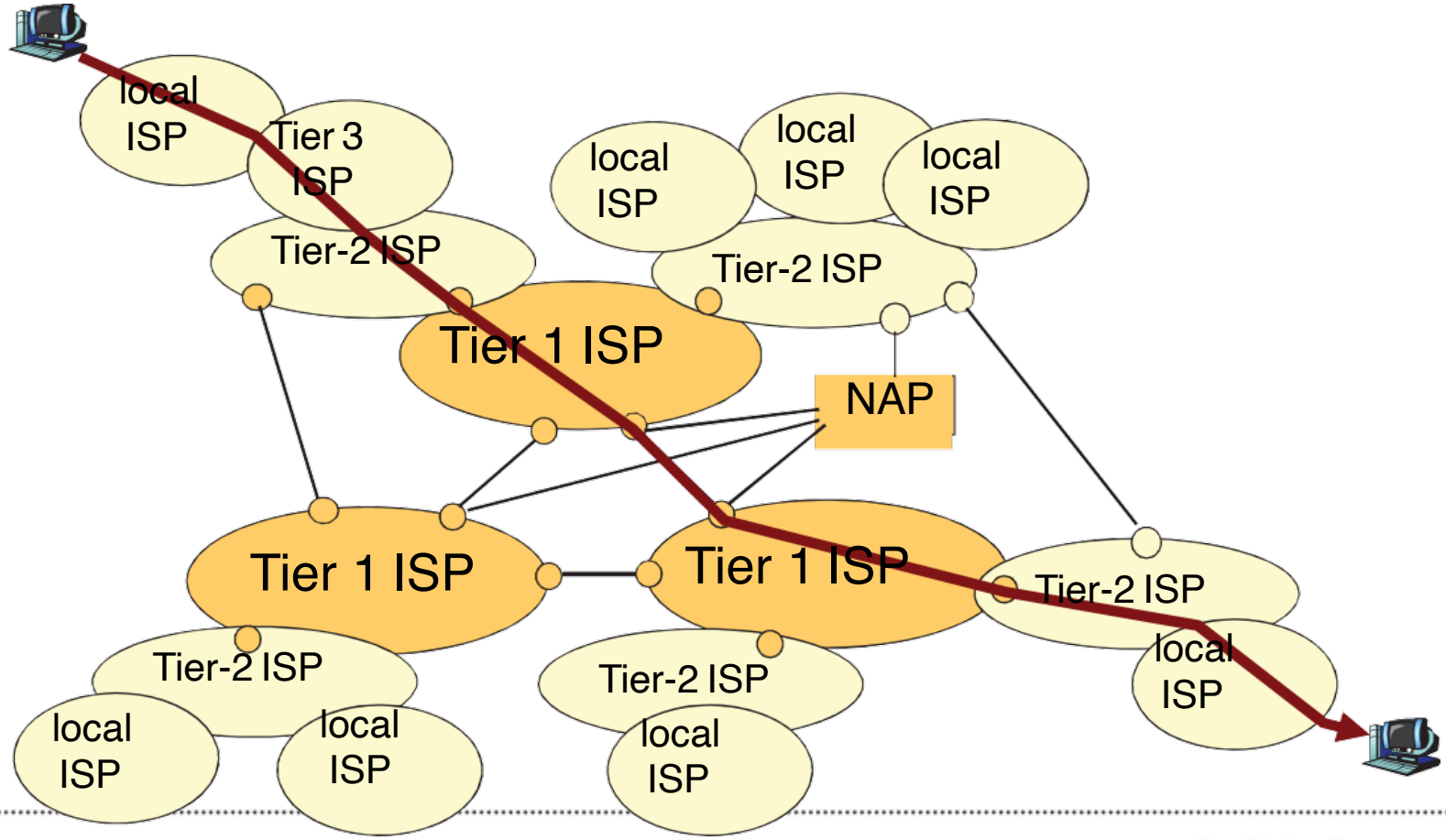
The Network Core

- Mesh of interconnected routers: network of networks
- The fundamental question: how is data transferred through the network system?
  - circuit switching: Dedicated circuit per call: telephone net
  - packet-switching: Data sent through the net in packets (discrete chunks)



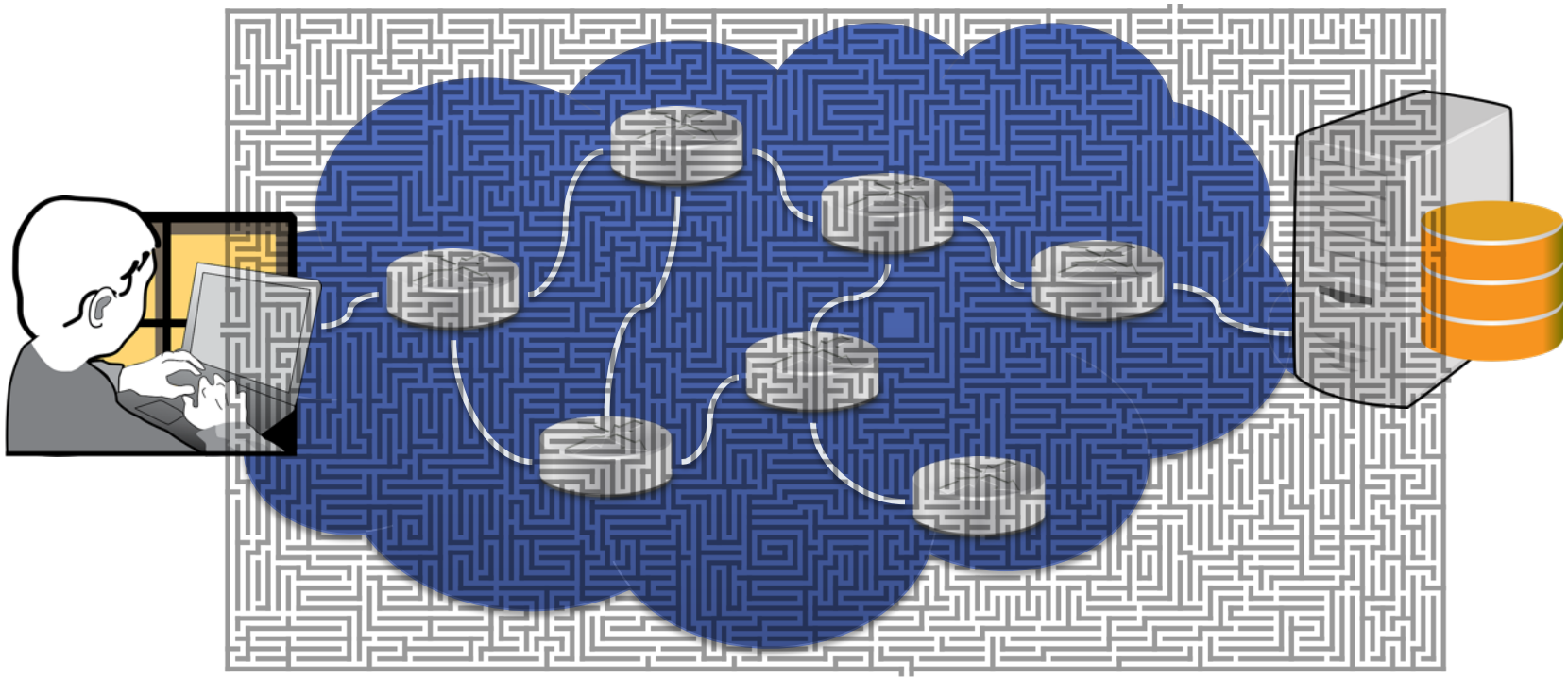
Internet structure: network of networks

- A packet passes through many networks!



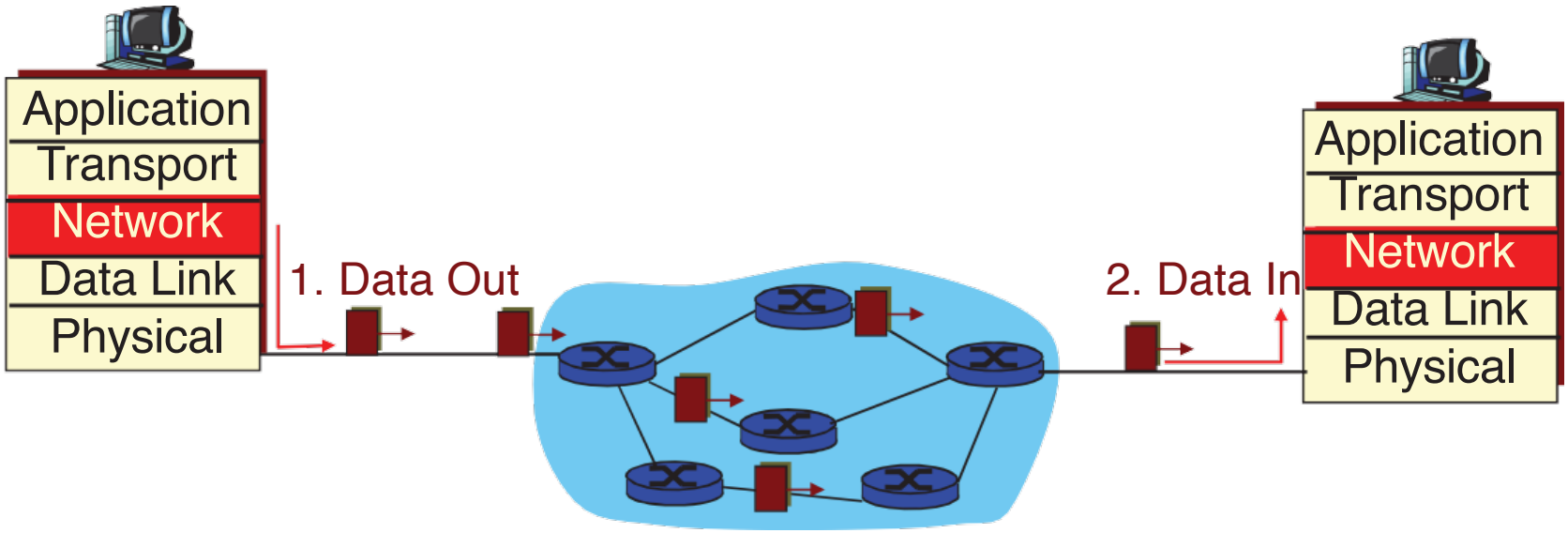
University of BRISTOL

Navigating through the Maze



Datagram networks: The Internet model

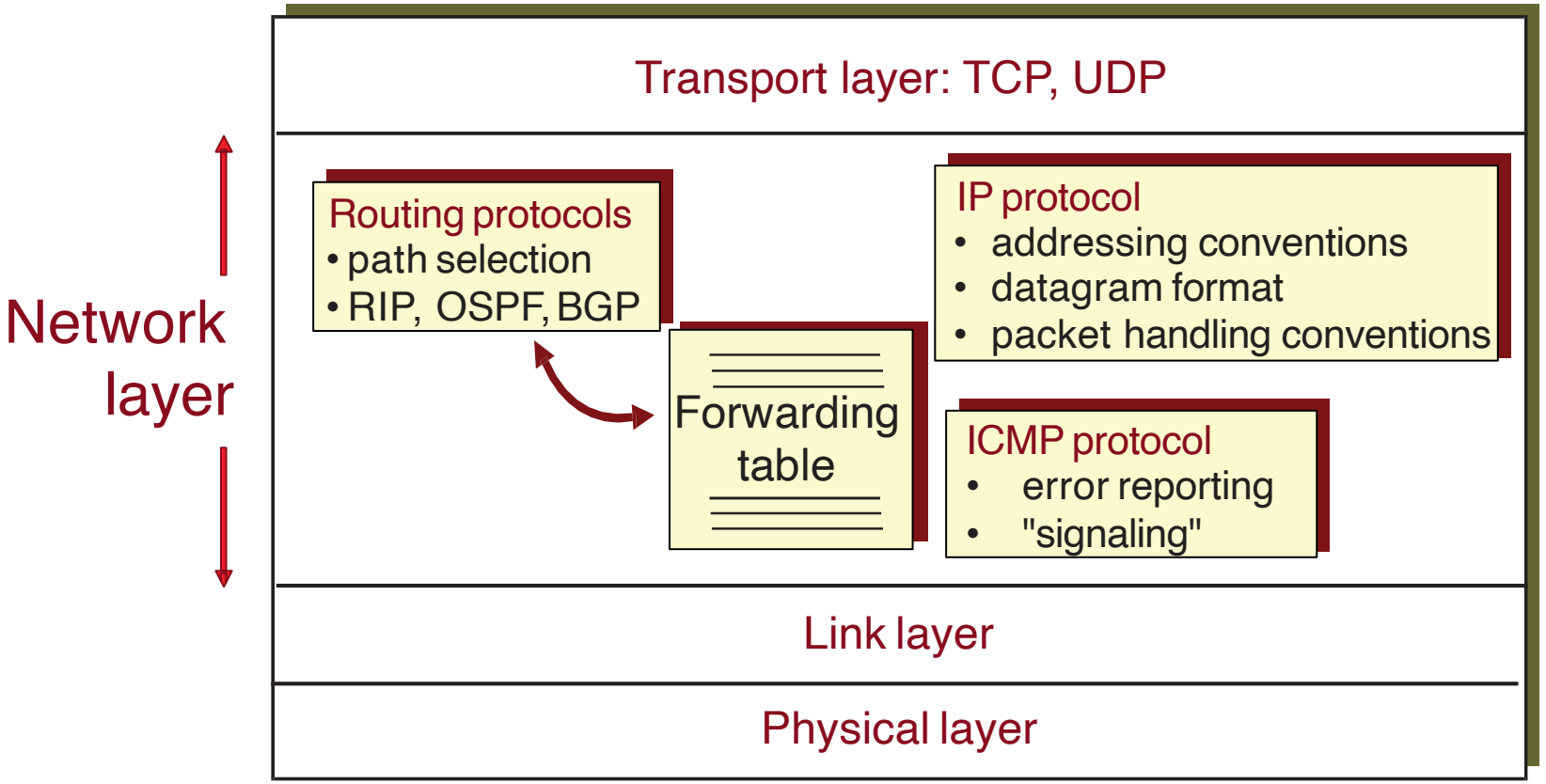
- No call setup at network layer
- Routers: no state about end-to-end connections
  - no network-level concept of "connection"
- Packets forwarded using destination host address
  - Packets between same *source-dest* pair may take different paths



University of BRISTOL

The Internet Network layer

Host and router network layer functions:



University of BRISTOL

Revision: What is Routing?

- **Network operations:** routing + forwarding (+ management)
- Information needed to get packets from source to destination.
- **Performance metrics** include: hop count, physical distance, link speed, link state (delay, load, up/down), link cost.
- **Decision:** per packet, per session.
- **Locality:** distributed, centralised, source, destination.
- Information source: none, local knowledge, adjacent nodes, nodes on a route, all nodes.
- **Discovery strategy:** fixed, adaptive, random, flooding.
- **Update strategy:** continuous, periodic, event-driven on change of topology, load, delay, cost.

University of BRISTOL

What is Internet Routing?

Routing Protocol Purpose:

"To discover routes and
adapt them to changing network conditions"

What does it involve?

- Path Discovery
  - Choose the best option. *Shortest is not necessarily best* (Fastest? More Reliable?)
  - Avoid loops
  - Cost?
- Algorithms
  - Distance Vector, Link State, Dijkstra, Bellman-Ford
- Protocols
  - RIP, OSPF, BGP, EGP
- Network organisation
  - Areas, subnets, regions, Autonomous Systems

University of BRISTOL

Routing Algorithm Classification

Global or decentralized information

Global

- All routers have complete topology and link cost info
- "Link state" algorithms (e.g. Dijkstra)

Decentralized

- Each router knows physically-connected neighbors, link costs to them
- Iterative computation, exchange of info with neigh. "Distance Vector" algorithms (e.g. Belman-Ford)

Static vs Dynamic

Static

- Routes change slowly over time

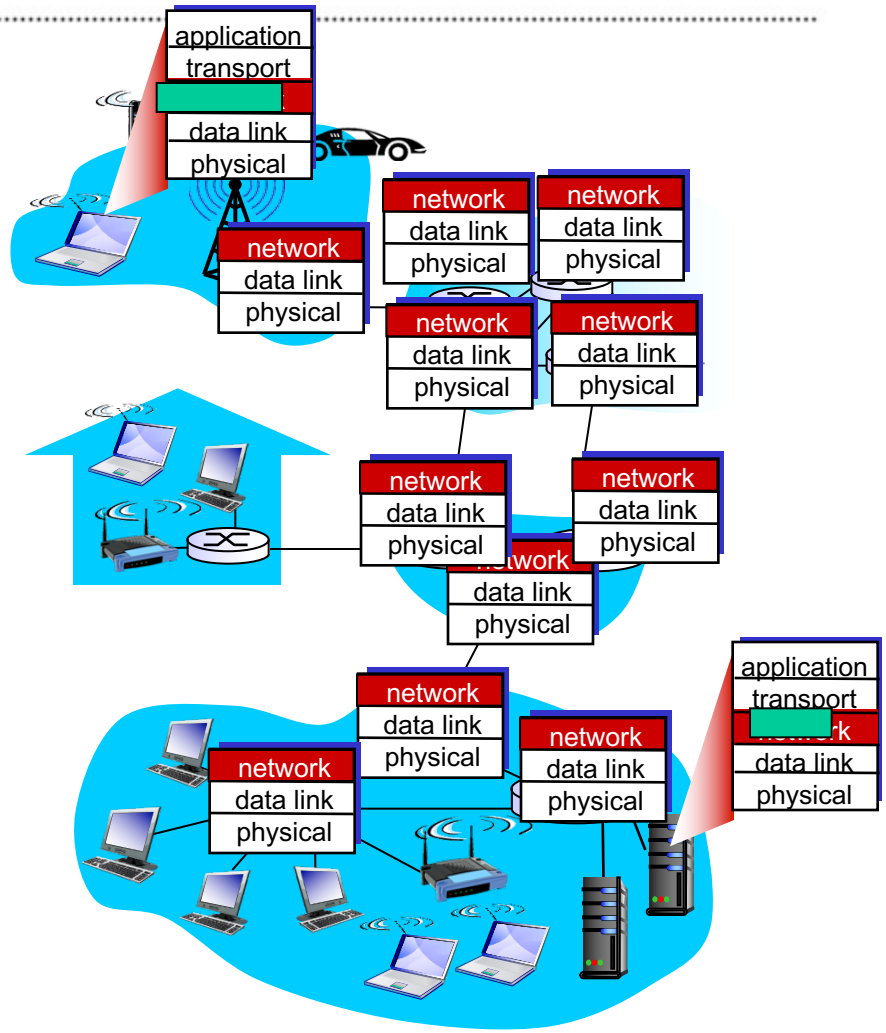
Dynamic

- Routes change more quickly
  - Periodic update
  - In response to link cost changes

University of BRISTOL

Network Layer

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



Slides from Ross Book

Two key network-layer functions

network-layer functions:

- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*

analogy: taking a trip

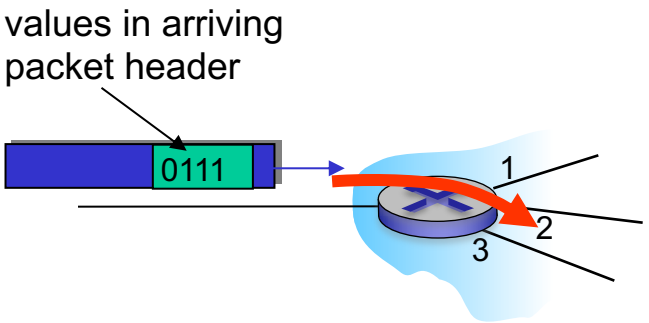
- *forwarding:* process of getting through single interchange
- *routing:* process of planning trip from source to destination

Slides from Ross Book

Network layer: data plane, control plane

Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function



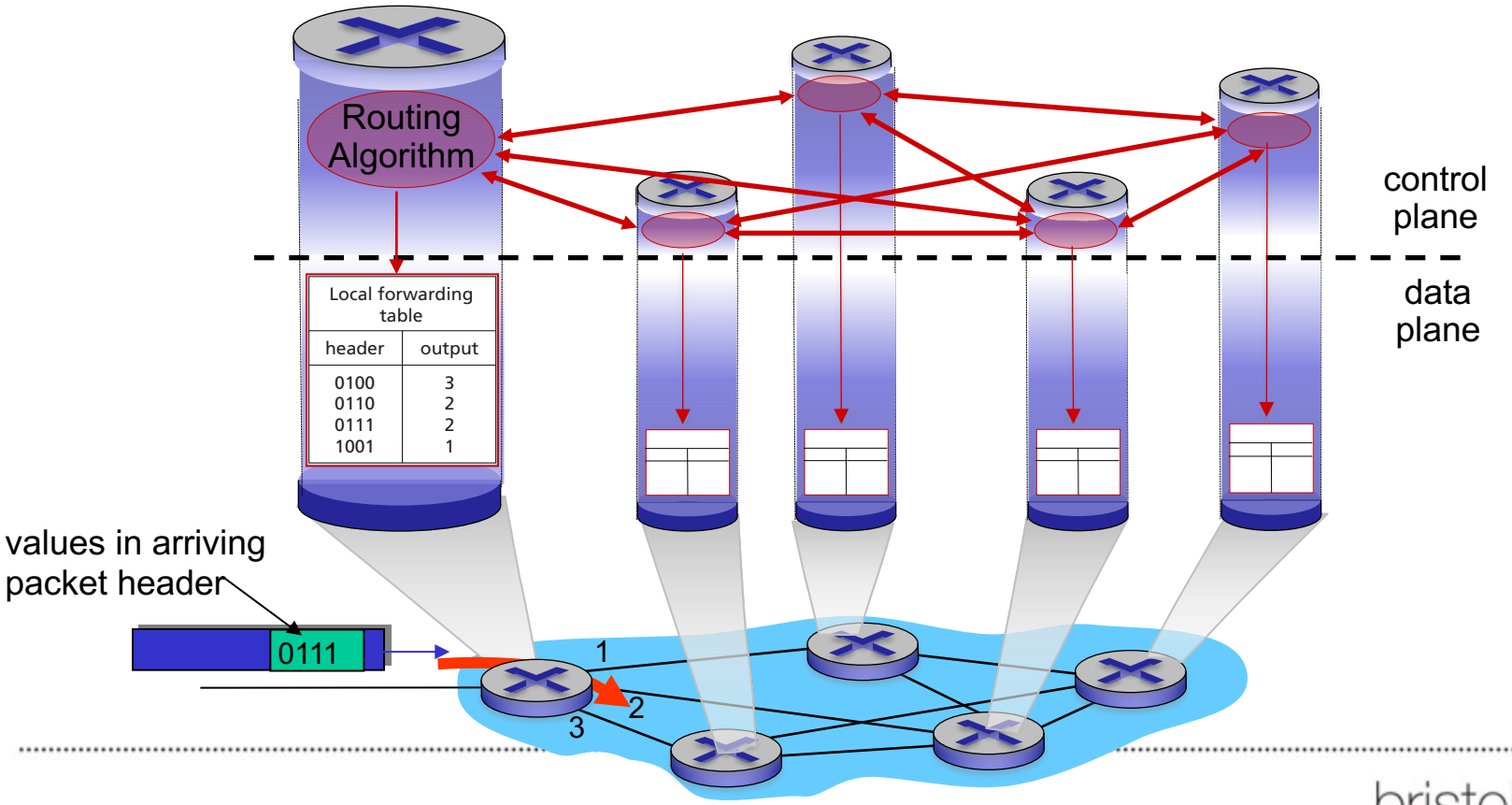
Control plane

- network-wide logic
- determines how datagram is routed among routers along end-end path from source host to destination host
- two control-plane approaches:
  - *traditional routing algorithms:* implemented in routers
  - *software-defined networking (SDN)*: implemented in (remote) servers

Slides from Ross Book

Per-router control plane

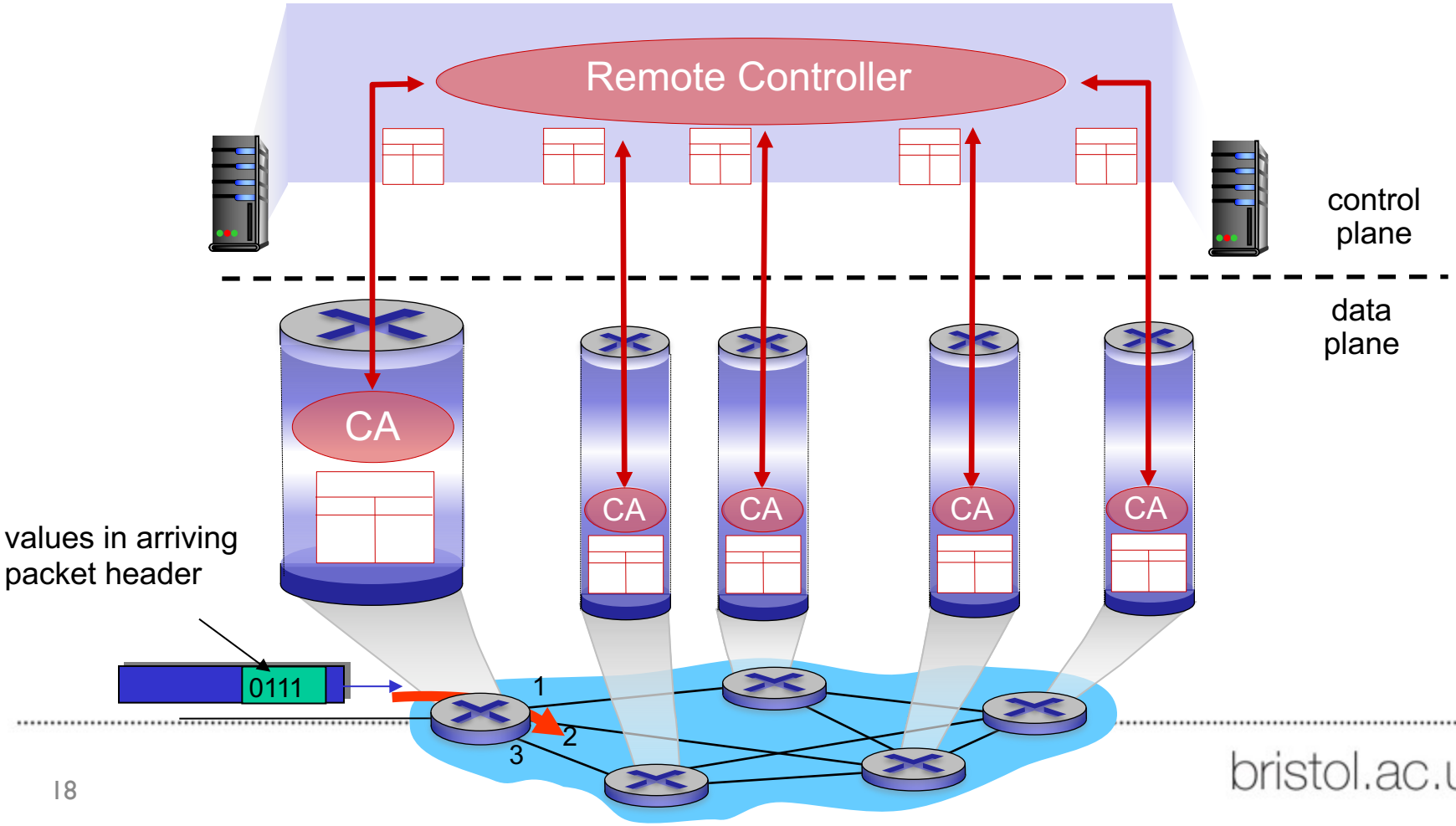
Individual routing algorithm components *in each and every router* interact in the control plane



Slides from Ross Book

Logically centralized control plane

A distinct (typically remote) controller interacts with local control agents (CAs)



Slides from Ross Book

University of BRISTOL

Distance Vector and Link State Algorithms

University of BRISTOL

Distance Vector Algorithms

- Routing table in each router contains information on:
  - best known "distance" to all other routers, and
  - Which line to use for each router.
- Router 'R' sends its table to each neighbour.
- Each neighbour periodically sends 'R' its own routing table
- 'R' can therefore update its own routing table from this info. *(Any problem with this approach???!!)*
- Problem: process may not converge quickly enough:
  - "Bouncing Effect" Good news travel fast, Bad news do not
  - "Count to Infinity" problem

University of BRISTOL

Bellman-Ford Algorithm

Notation

- h = number of hops being considered
- $D^{(h)}_n$ = cost of hop-path from s to n (effectively distance)

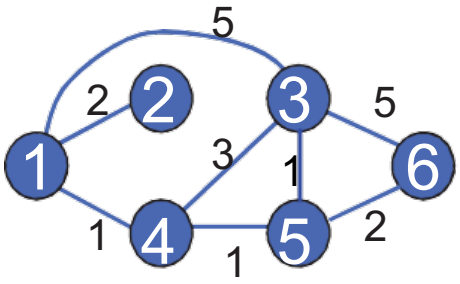
Method

- Find all nodes 1 hop, 2 hops, 3 hops, etc. away.

- Initialize $D^{(h)}_n$ = infinity for all (n != s), $D^{(h)}_s = 0$ for all h
- Find j^{th} node for which h+1 cost is the least of

$$D^{(h+1)}_n = \min_j[D^{(h)}_j + d_{jn}]$$

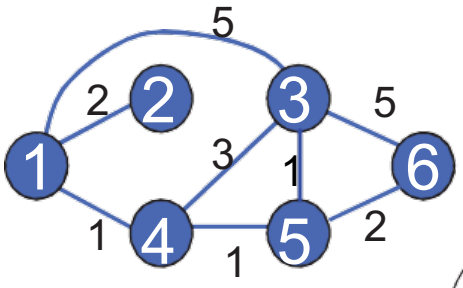
Example: Bellman-Ford



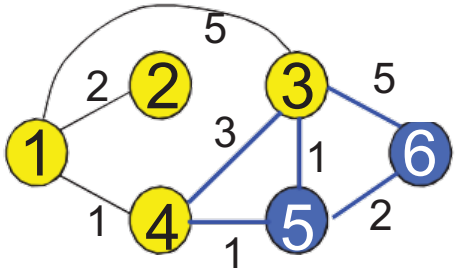
Cold start

h	$D^{(h)}_2$	Path	$D^{(h)}_3$	Path	$D^{(h)}_4$	Path	$D^{(h)}_5$	Path	$D^{(h)}_6$	Path
0	inf	-	inf	-	inf	-	inf	-	inf	-

Example: Bellman-Ford



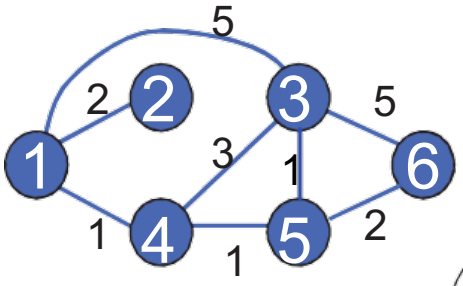
Cold start



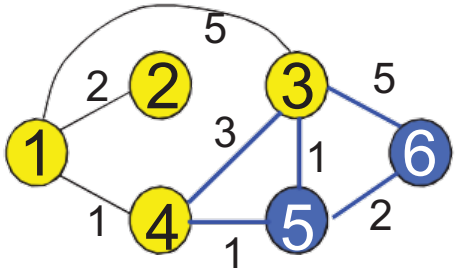
h = 1

h	$D^{(h)}_2$	Path	$D^{(h)}_3$	Path	$D^{(h)}_4$	Path	$D^{(h)}_5$	Path	$D^{(h)}_6$	Path
0	inf	-	inf	-	inf	-	inf	-	inf	-
1	2	1-2	5	1-3	1	1-4	inf	-	inf	-

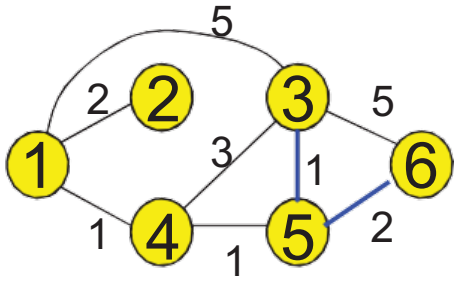
Example: Bellman-Ford



Cold start



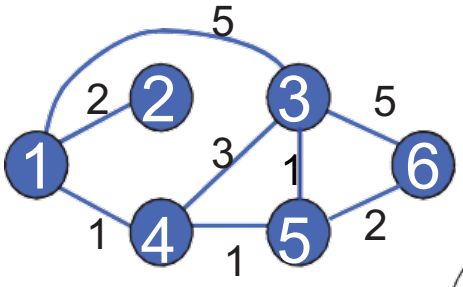
h = 1



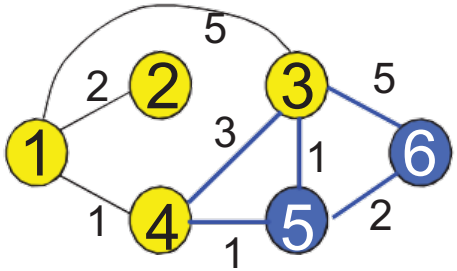
h = 2

h	$D^{(h)}_2$	Path	$D^{(h)}_3$	Path	$D^{(h)}_4$	Path	$D^{(h)}_5$	Path	$D^{(h)}_6$	Path
0	inf	-	inf	-	inf	-	inf	-	inf	-
1	2	1-2	5	1-3	1	1-4	inf	-	inf	-
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6

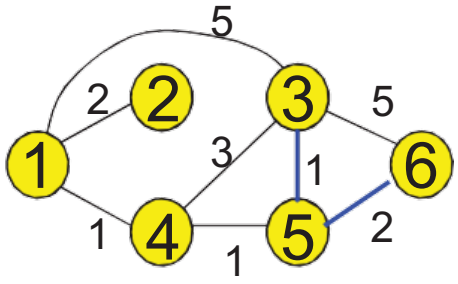
Example: Bellman-Ford



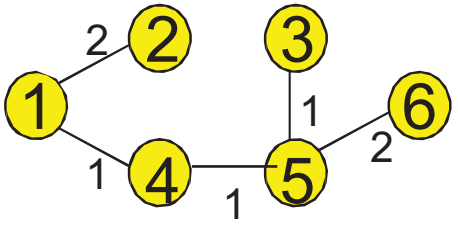
Cold start



h = 1



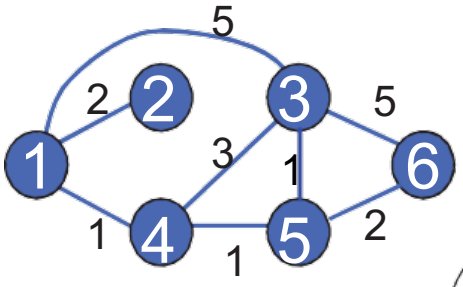
h = 2



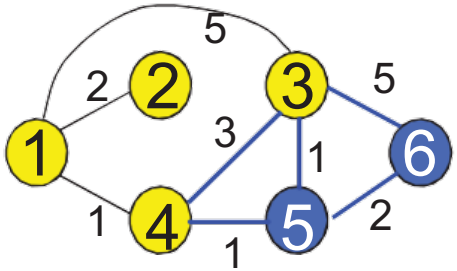
h = 3

h	$D^{(h)}_2$	Path	$D^{(h)}_3$	Path	$D^{(h)}_4$	Path	$D^{(h)}_5$	Path	$D^{(h)}_6$	Path
0	inf	-	inf	-	inf	-	inf	-	inf	-
1	2	1-2	5	1-3	1	1-4	inf	-	inf	-
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6
3	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

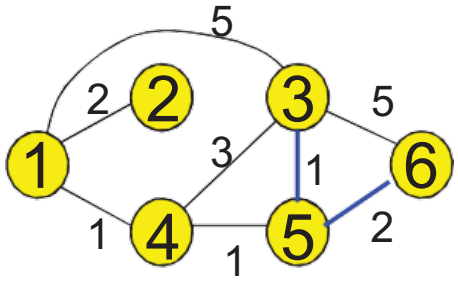
Example: Bellman-Ford



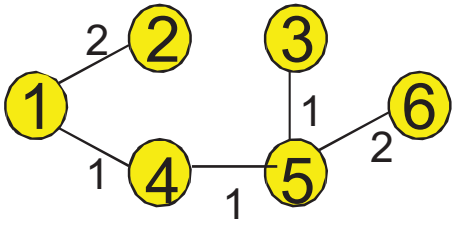
Cold start



h = 1



h = 2

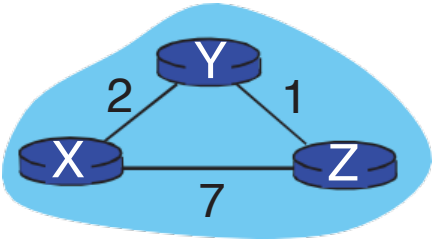


h = 3

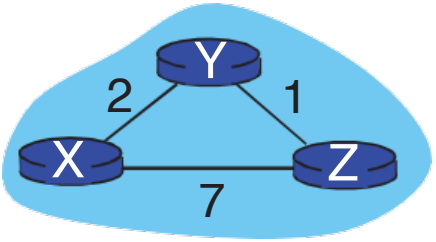
h	$D^{(h)}_2$	Path	$D^{(h)}_3$	Path	$D^{(h)}_4$	Path	$D^{(h)}_5$	Path	$D^{(h)}_6$	Path
0	inf	-	inf	-	inf	-	inf	-	inf	-
1	2	1-2	5	1-3	1	1-4	inf	-	inf	-
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6
3	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
4	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

Distance Vector Algorithm: example

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



Distance Vector Algorithm: example



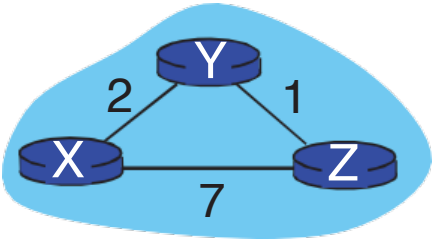
X

dest	cost via Y	cost via Z
Y	2	∞
Z	∞	7

Y

dest	cost via X	cost via Z
X	2	∞
Z	∞	1

Distance Vector Algorithm: example



D^X

dest	cost via Y	cost via Z
Y	2	∞
Z	∞	7

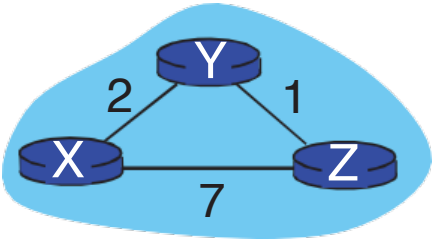
D^Y

dest	cost via X	cost via Z
X	2	∞
Z	∞	1

D^Z

dest	cost via X	cost via Y
X	7	∞
Y	∞	1

Distance Vector Algorithm: example



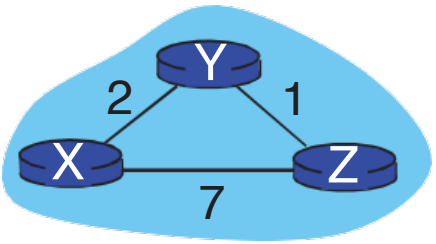
X	cost via Y	cost via Z
dest Y	2	∞
dest Z	∞	7

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

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

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

Distance Vector Algorithm: example



X

dest	cost via Y	cost via Z
Y	2	∞
Z	∞	7

X

dest	cost via Y	cost via Z
Y	2	8
Z	3	7

Y

dest	cost via X	cost via Z
X	2	∞
Z	∞	1

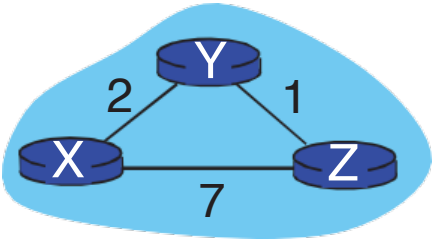
Y

dest	cost via X	cost via Z
X	2	8
Z	9	1

Z

dest	cost via X	cost via Y
X	7	∞
Y	∞	1

Distance Vector Algorithm: example



X

dest	cost via Y	cost via Z
Y	2	∞
Z	∞	7

dest	cost via Y	cost via Z
Y	2	8
Z	3	7

Y

dest	cost via X	cost via Z
X	2	∞
Z	∞	1

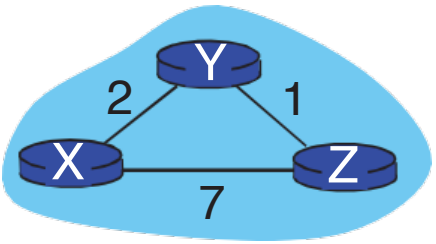
dest	cost via X	cost via Z
X	2	8
Z	9	1

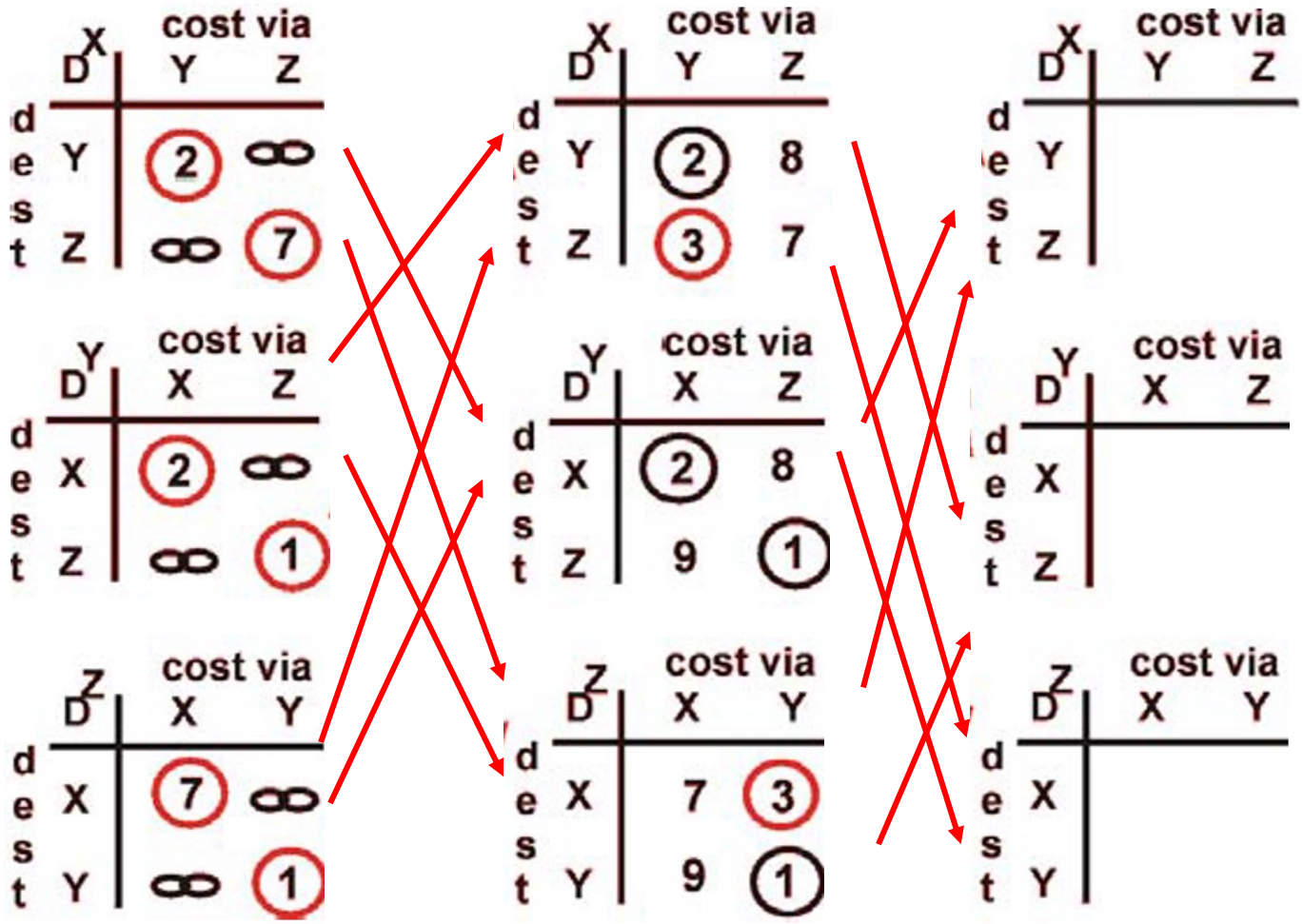
Z

dest	cost via X	cost via Y
X	7	∞
Y	∞	1

dest	cost via X	cost via Y
X	7	3
Y	9	1

Distance Vector Algorithm: example

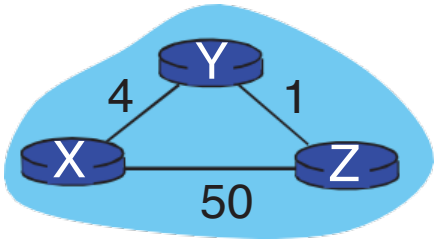




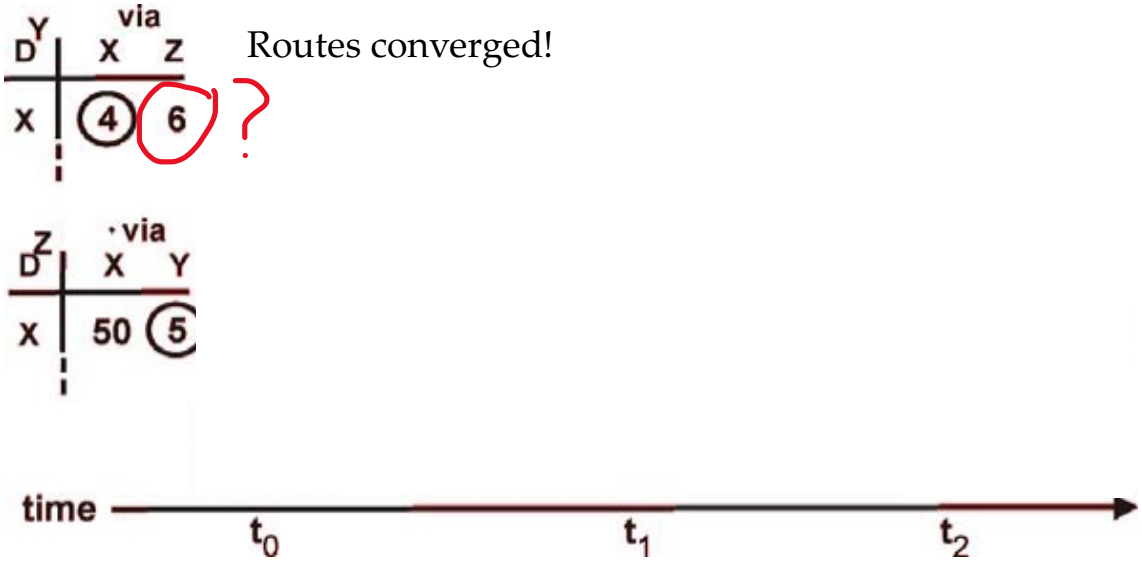
Distance Vector: Link cost changes

Link cost changes:

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



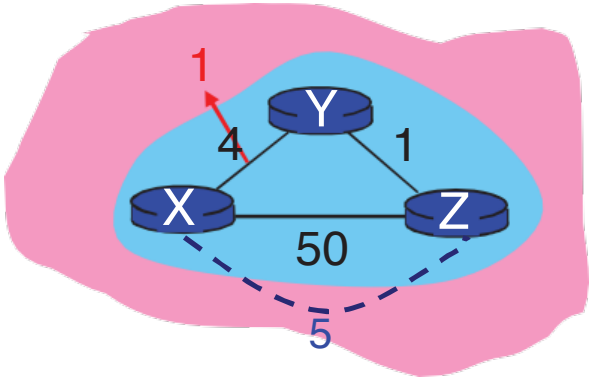
Good news travels fast



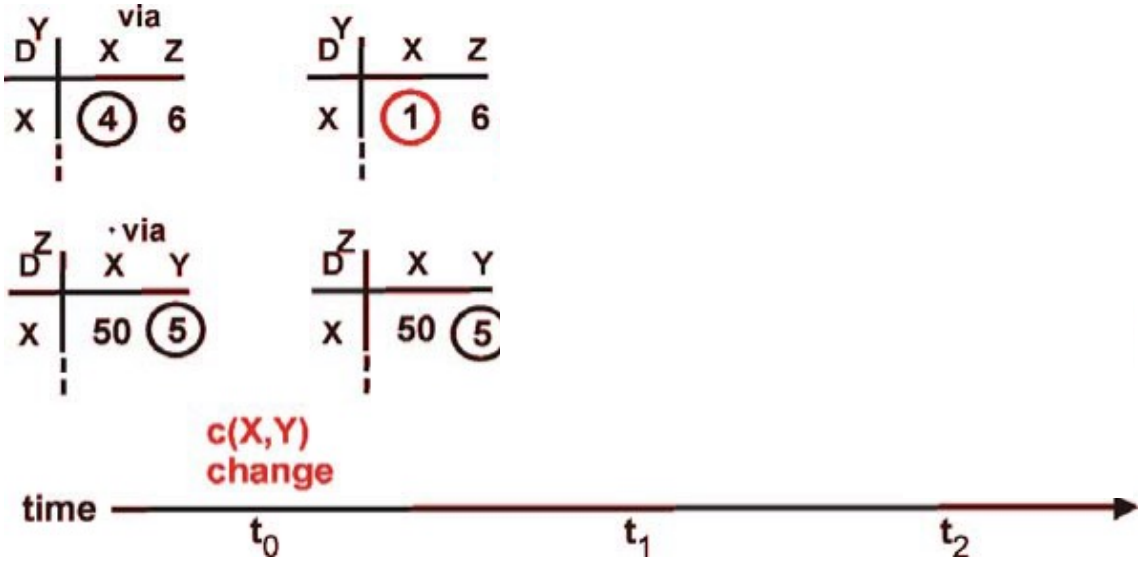
Distance Vector: Link cost changes

Link cost changes:

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



Good news travels fast

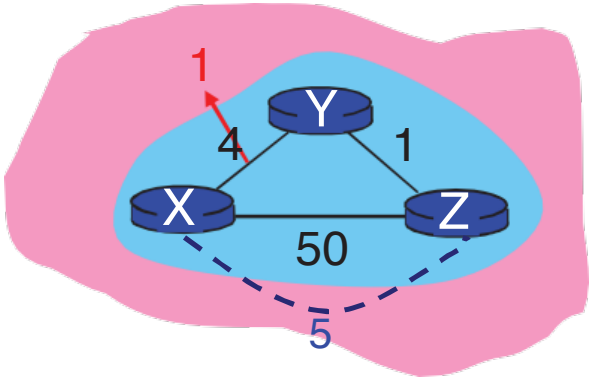


Algorithm terminates

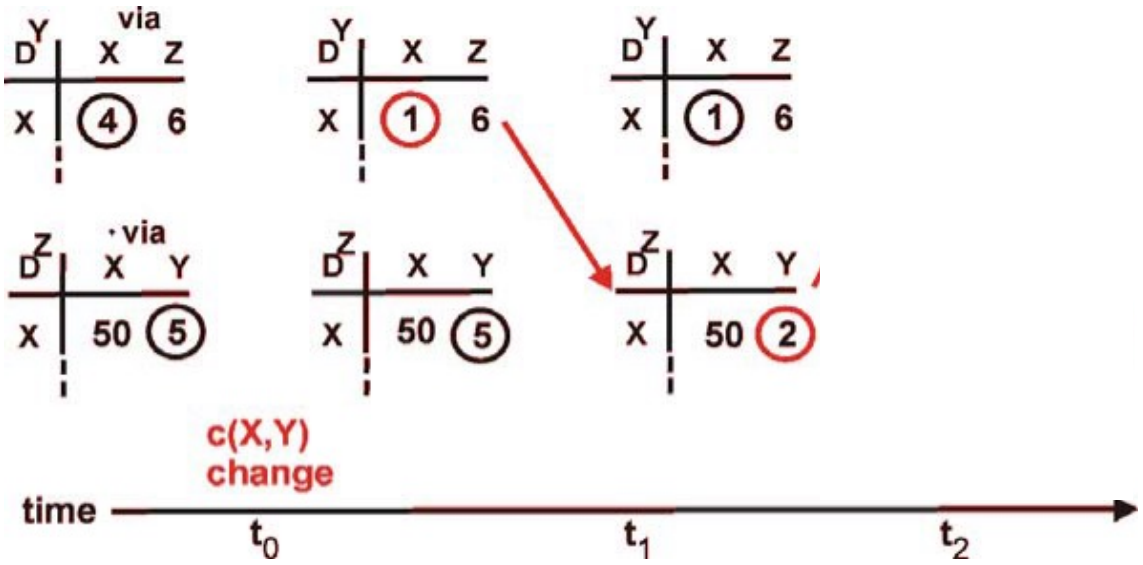
Distance Vector: Link cost changes

Link cost changes:

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



Good news travels fast

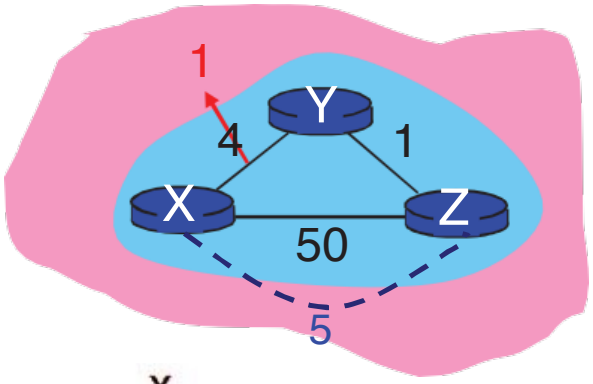


Algorithm terminates

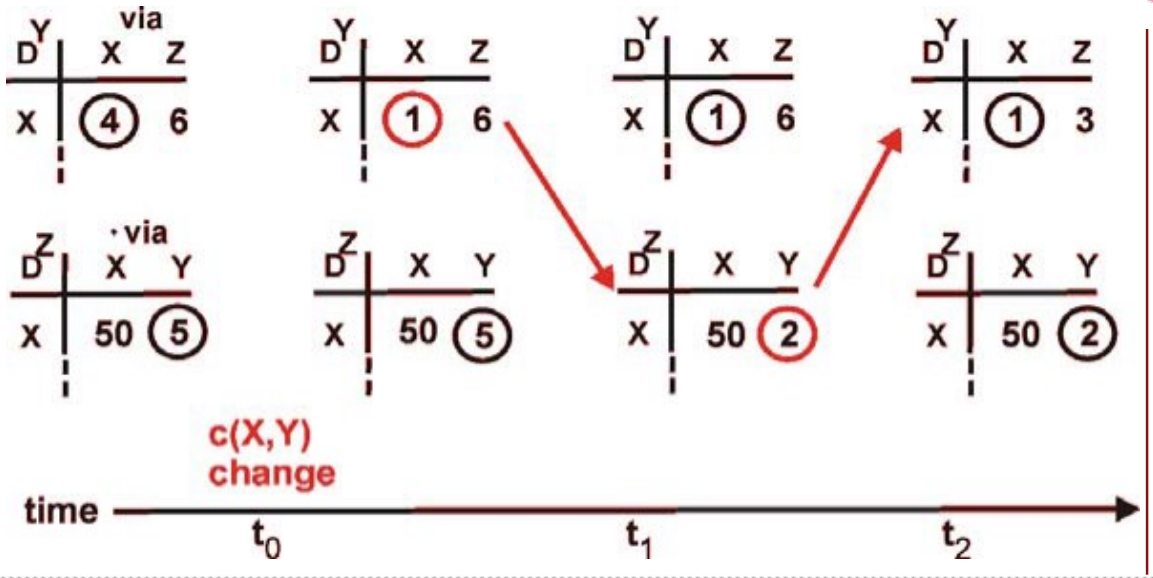
Distance Vector: Link cost changes

Link cost changes:

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



Good news travels fast

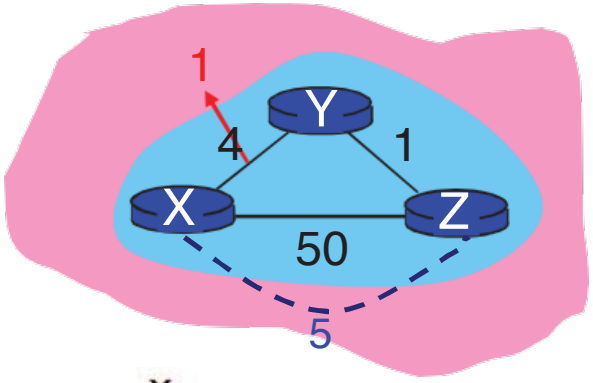


Algorithm terminates

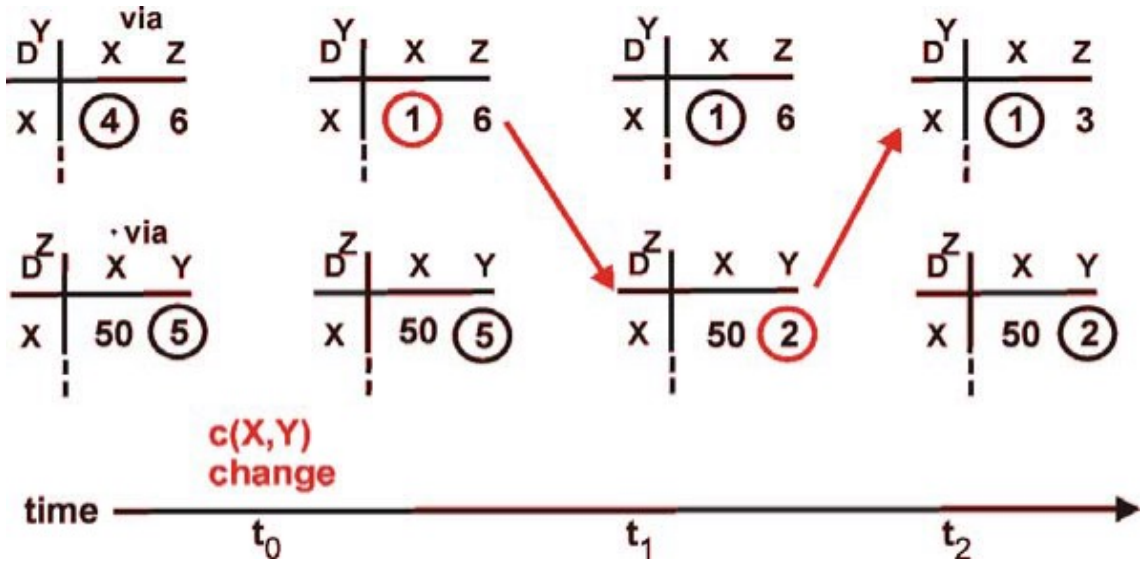
Distance Vector: Link cost changes

Link cost changes:

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



Good news travels fast

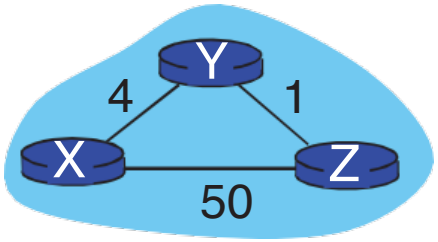


Algorithm terminates

Distance Vector: Link cost changes

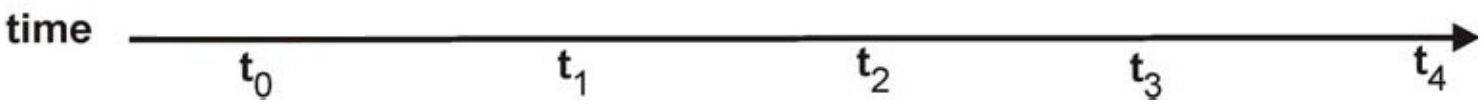
Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!



Y	via X	via Z
X	(4)	6

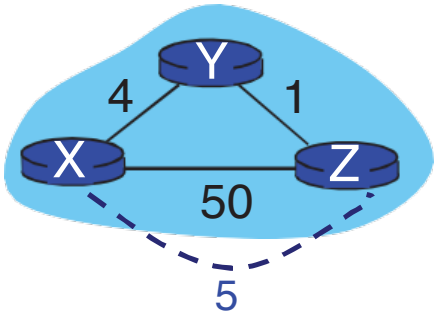
Z	via X	via Y
X	50	(5)

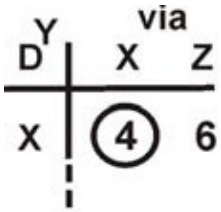


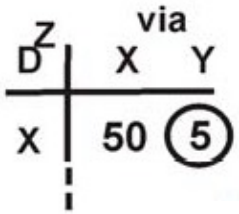
Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!



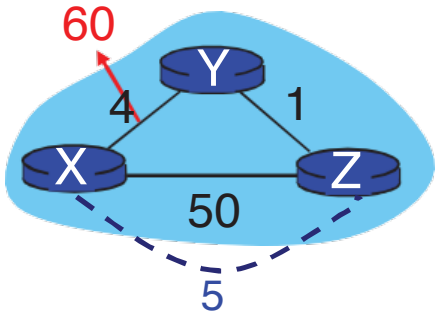


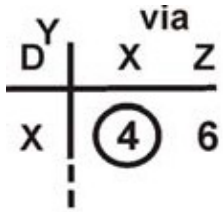


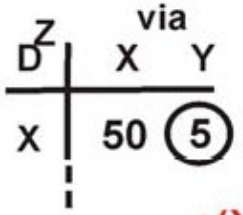
Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!



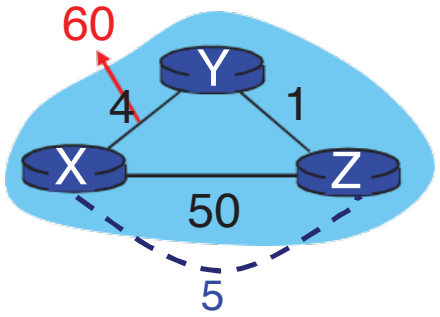


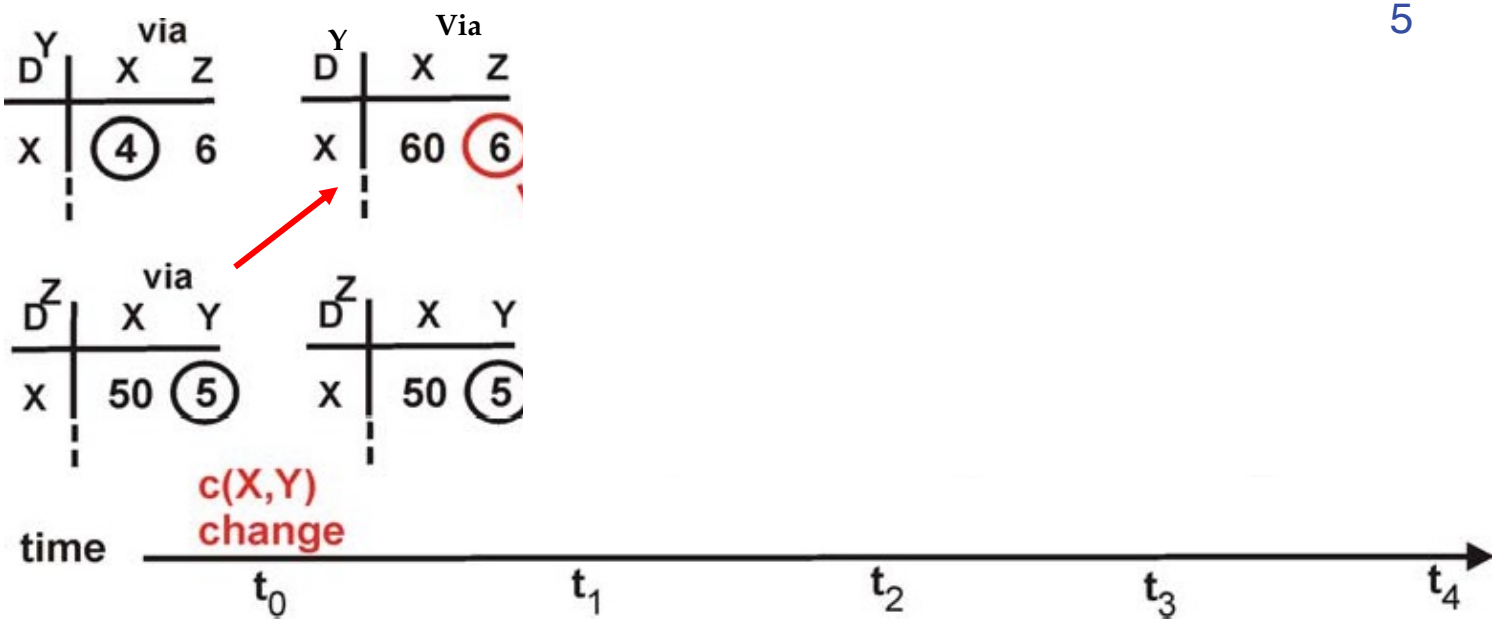


Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!

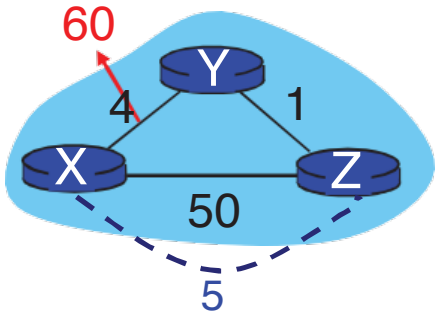


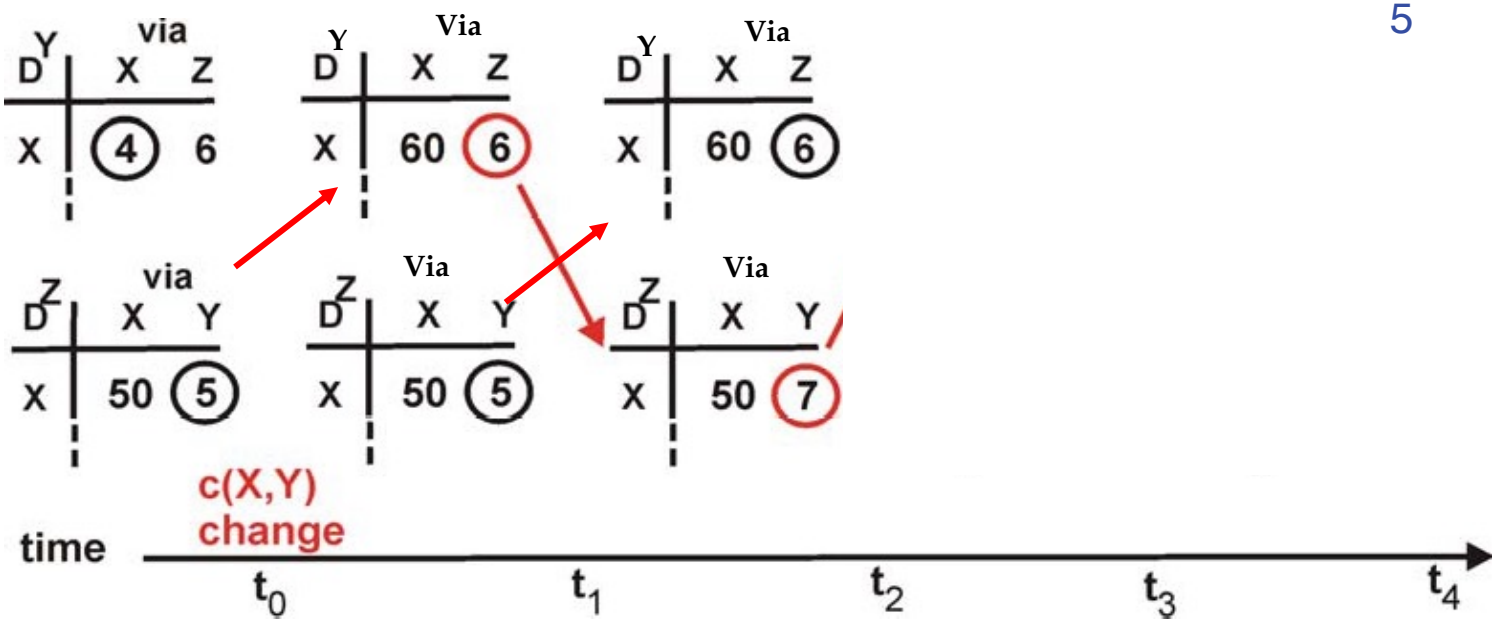


Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!

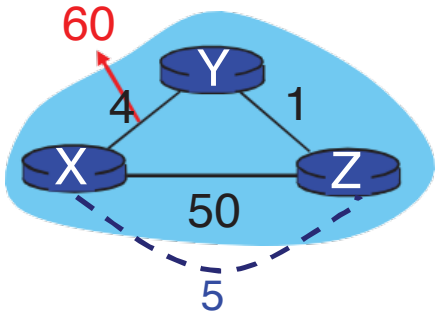


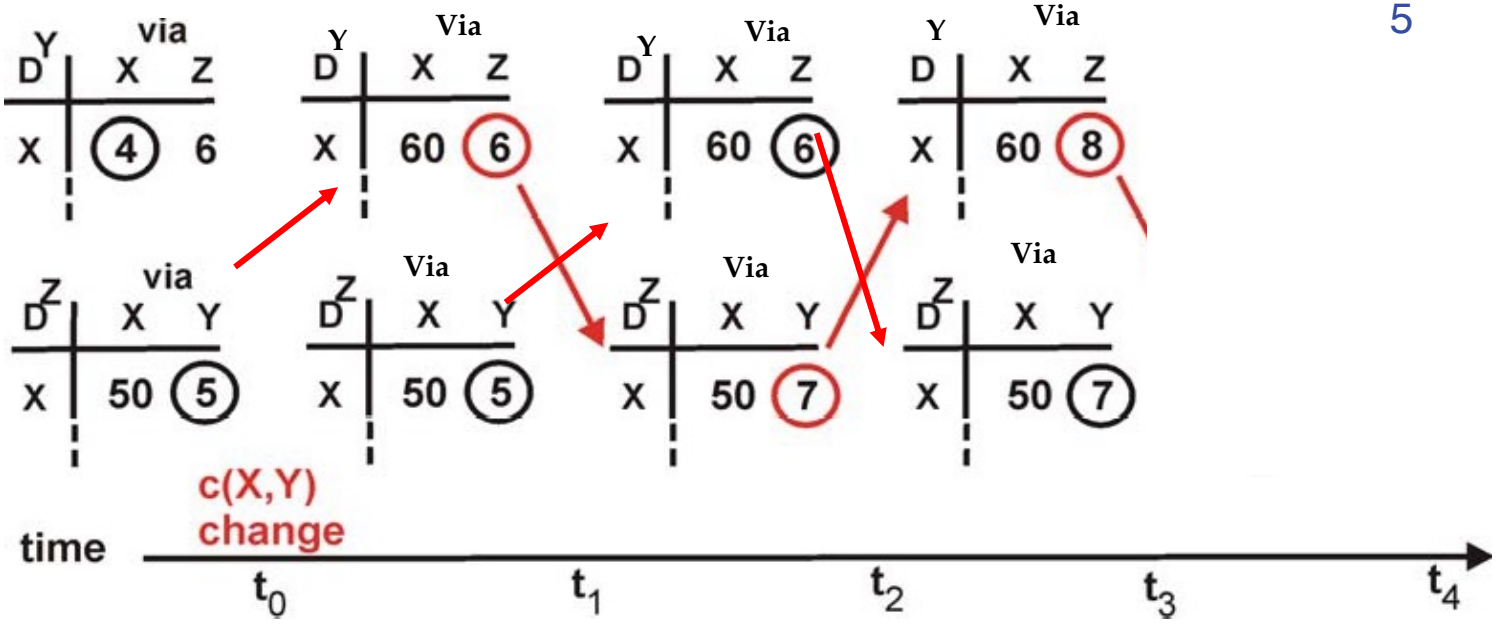


Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!

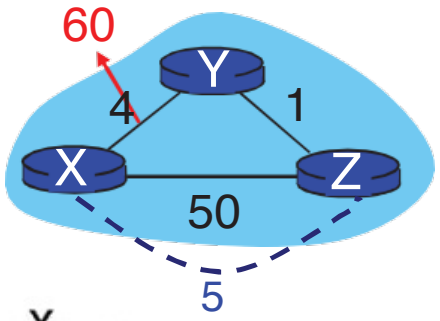


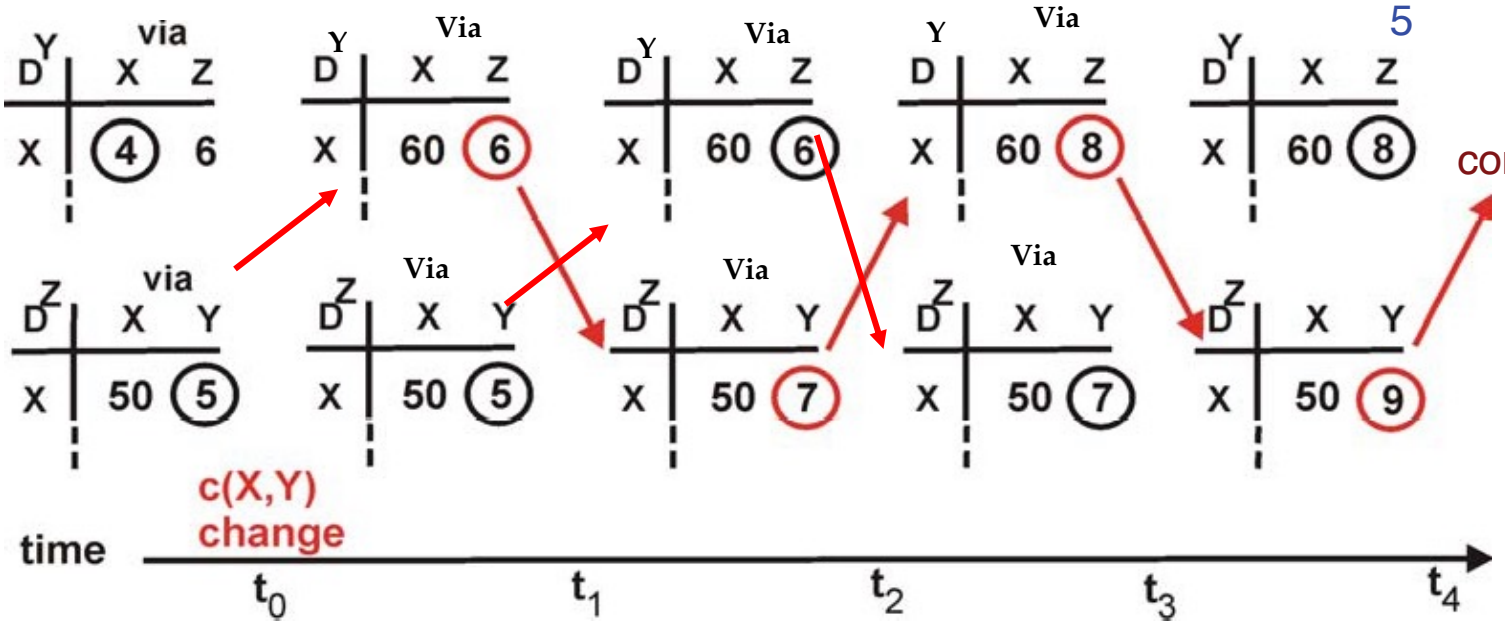


Distance Vector: Link cost changes

Link cost changes:

- Good news travels fast
- But not bad news…
- "Count to Infinity" problem!





University of BRISTOL

Summary

- Routing (packet from source to destination)
- Distributed or centralized
- Local or global
- Distance vector
- Change in circumstances

University of BRISTOL

Reading

- A.S. Tanenbaum, "Computer Networks" 3rd Ed. - Chapter 5.

Or

- L.L. Peterson & B.S. Davies "Computer Networks - A Systems Approach". Chapter 4.

Or

- Kurose, J., "Computer Networks: A Top-Down Approach featuring the Internet", Chapter 4.
- RFC1058 at http://www.ietf.org/rfc.html (original RIP specification - historic status.
- Current RIP standard is described in RFC2453)
- RFC2328 - OSPF version 2 (standard).
- https://www.thecoldwire.com/what-is-the-difference-between-routing-and-forwarding/