Master Course
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle

Christian Grothoff, Ph.D.

Chair for Network Architectures and Services

Institut für Informatik
Technische Universität München
http://www.net.in.tum.de

Technische Universität München

Chapter 4: Network Layer

Chapter goals:

- understand principles behind network layer services:
    - network layer service models
    - forwarding versus routing
    - how a router works
    - routing (path selection)
    - dealing with scale
    - advanced topics: IPv6, mobility
- instantiation, implementation in the Internet

Chapter 4: Network Layer

Part 1

- **Introduction**
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- Virtual circuit and datagram networks
- What's inside a router
- NAT

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- Broadcast and multicast routing

Network layer

- transport segment from sending to receiving host

- on sending side encapsulates segments into datagrams

- on rcving side, delivers segments to transport layer

- network layer protocols in *every* host, router

- router examines header fields in all IP datagrams passing through it
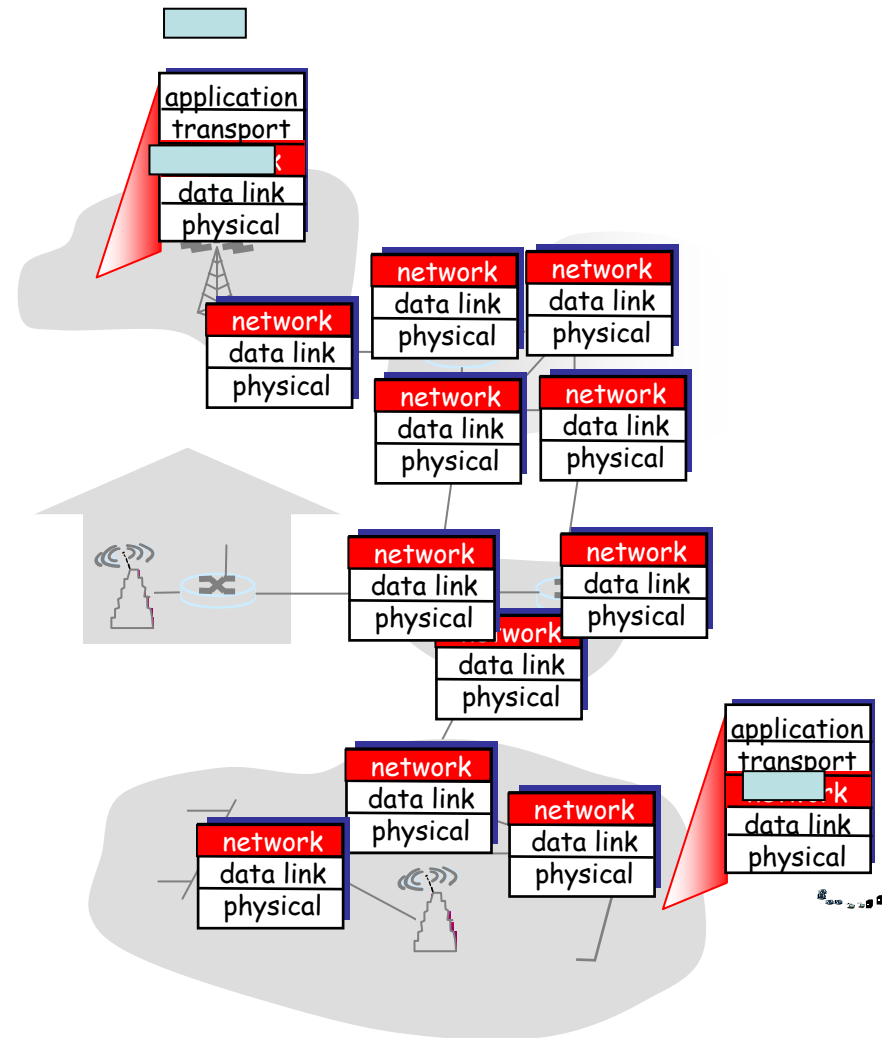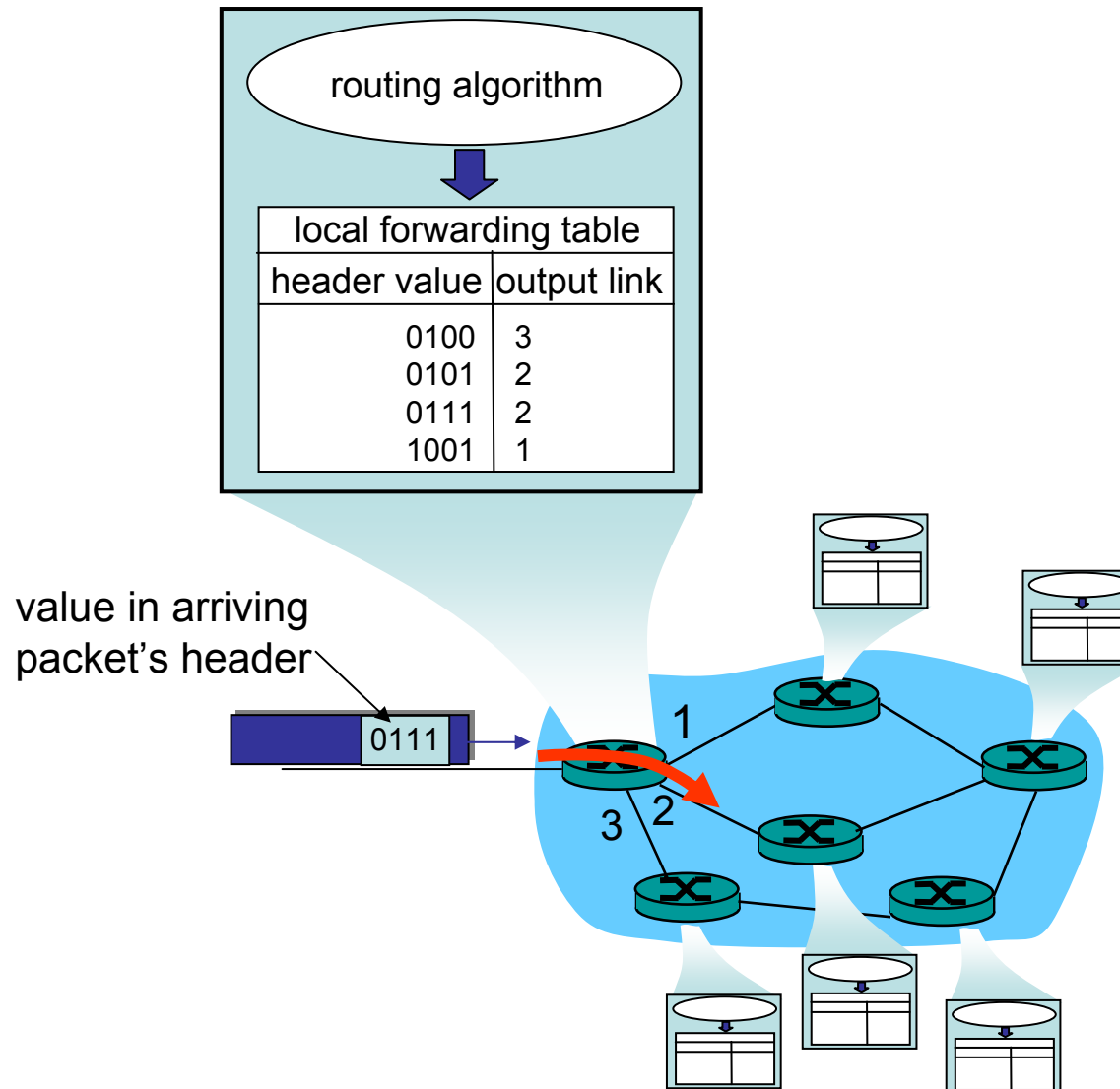
Two Key Network-Layer Functions

❑ *routing:* determine route taken by packets from source to dest.

   ▪ *routing algorithms*

❑ *forwarding:* move packets from router's input to appropriate router output

<u>analogy:</u>

❑ routing: process of planning trip from source to dest

❑ forwarding: process of getting through single interchange
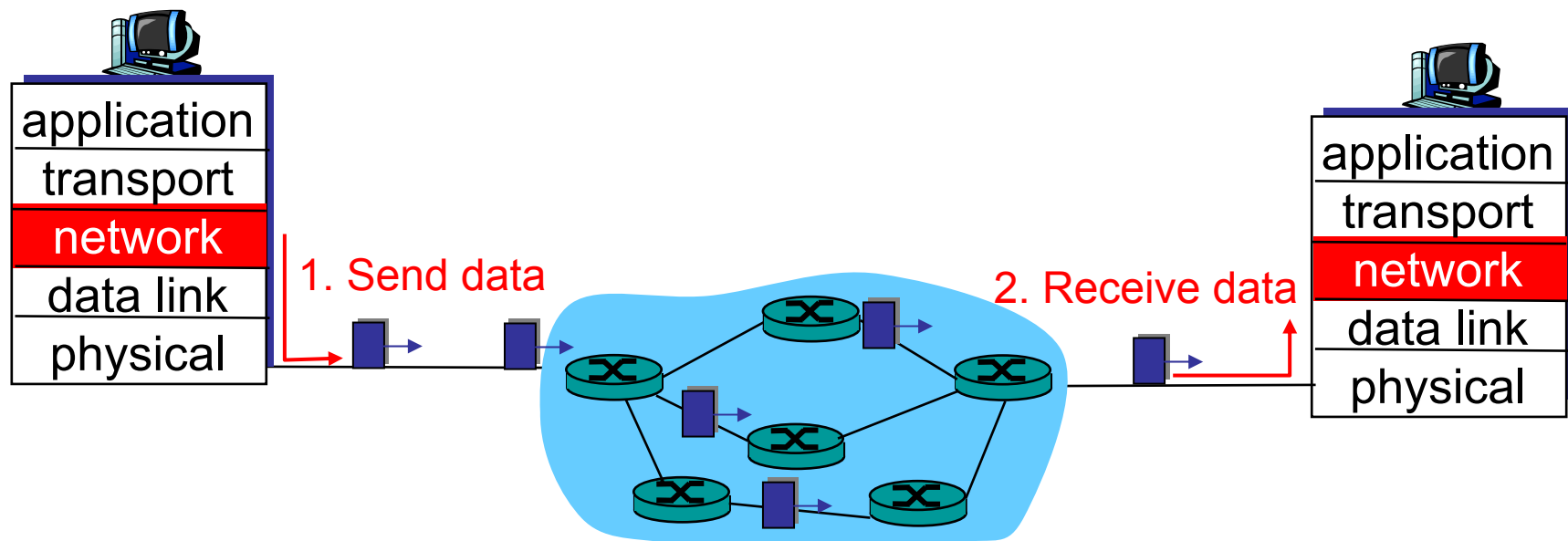
Interplay between routing and forwarding

Datagram networks

❑ 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

application
transport
network
data link
physical

1. Send data

application
transport
network
data link
physical

2. Receive data

Forwarding table

4 billion possible entries

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Longest prefix matching

<u>Prefix Match</u>	<u>Link Interface</u>
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
otherwise	3

Examples

DA: 11001000 00010111 00010110 10100001   Which interface?

DA: 11001000 00010111 00011000 10101010   Which interface?

Chapter 4: Network Layer

Part 1

- Introduction
- **IP: Internet Protocol**
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- Virtual circuit and datagram networks
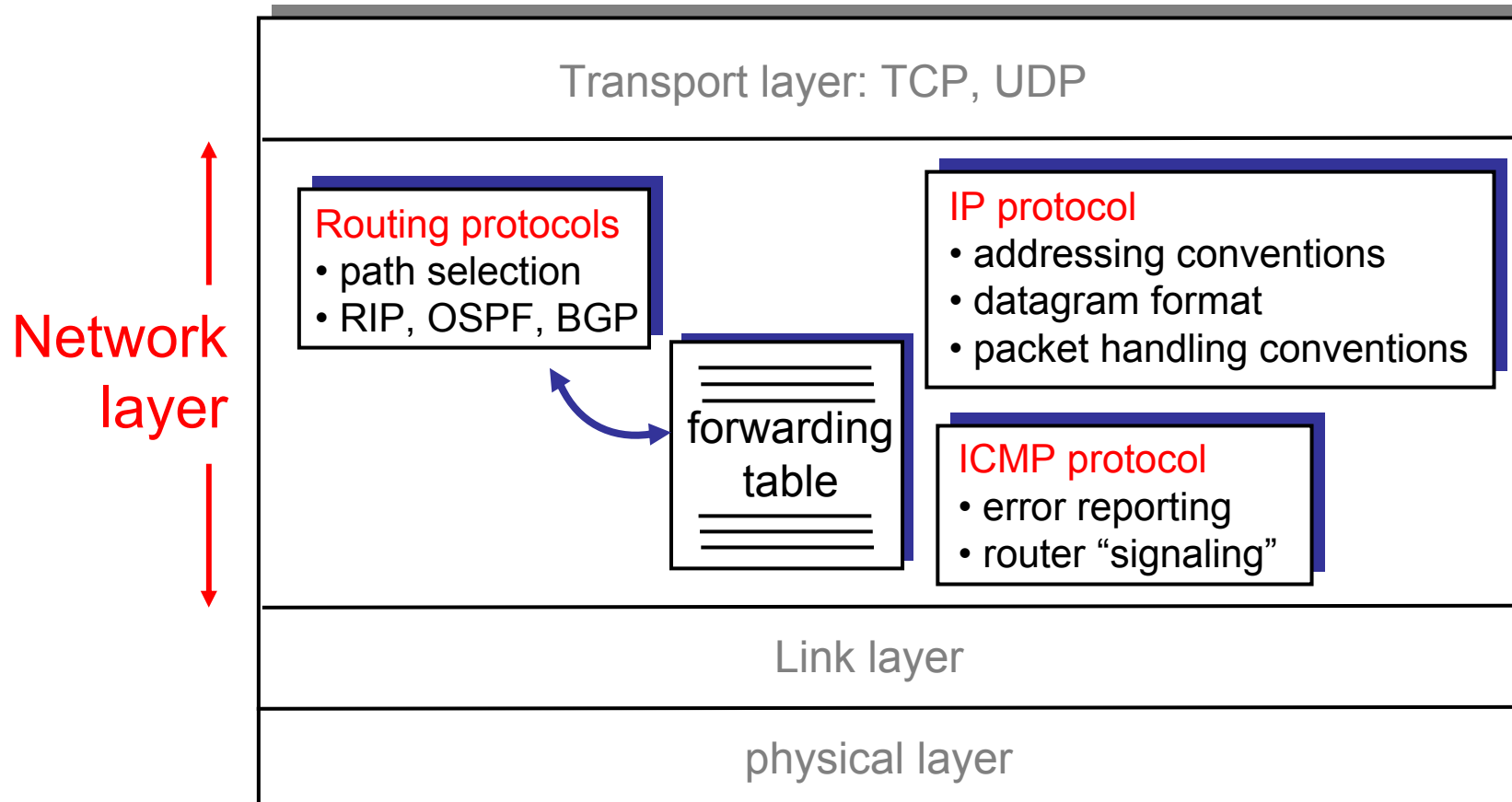- What's inside a router
- NAT

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- Broadcast and multicast routing

The Internet Network layer

Host, router network layer functions:

Network layer

Transport layer: TCP, UDP

Routing protocols
- path selection
- RIP, OSPF, BGP

forwarding table

IP protocol
- addressing conventions
- datagram format
- packet handling conventions

ICMP protocol
- error reporting
- router "signaling"

Link layer

physical layer

IP datagram format

IP protocol version number

header length (bytes)

"type" of data

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to

← 32 bits →

total datagram length (bytes)

for fragmentation/ reassembly

ver	head. len	type of service	length
16-bit identifier		flgs	fragment offset
time to live	upper layer	header checksum	
32 bit source IP address			
32 bit destination IP address			
Options (if any)			
data (variable length, typically a TCP or UDP segment)			

E.g. timestamp, record route taken, specify list of routers to visit.

how much overhead with TCP?

❑ 20 bytes of TCP

❑ 20 bytes of IP

❑ = 40 bytes + app layer overhead

IP Fragmentation & Reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments

fragmentation:
in: one large datagram
out: 3 smaller datagrams
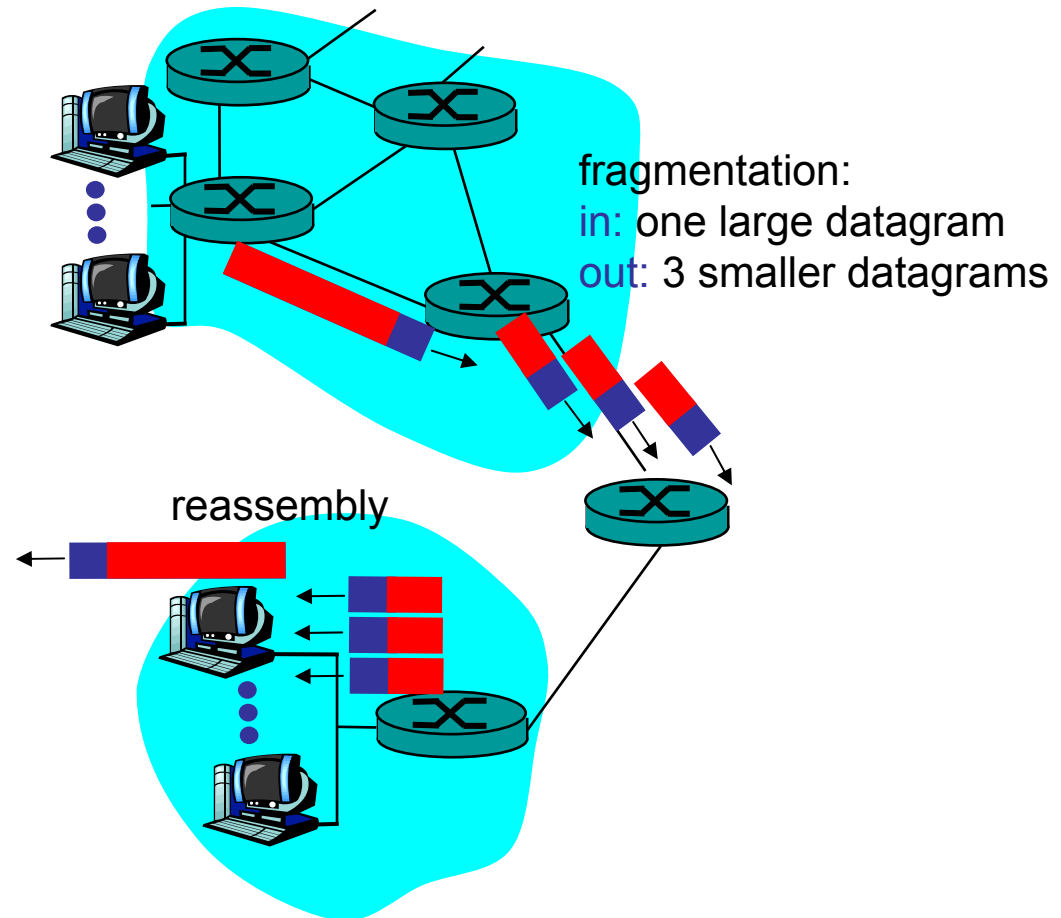
reassembly

IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- MTU = 1500 bytes

| length =4000 | ID =x | fragflag =0 | offset =0 | |

One large datagram becomes several smaller datagrams

1480 bytes in data field

offset = 1480/8

| length =1500 | ID =x | fragflag =1 | offset =0 | |

| length =1500 | ID =x | fragflag =1 | offset =185 | |

| length =1040 | ID =x | fragflag =0 | offset =370 | |

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ **IP: Internet Protocol**

  ▪ Datagram format

  ▪ **IPv4 addressing**

  ▪ ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

❑ NAT

Part 3

❑ Routing algorithms

  ▪ Link state

  ▪ Distance Vector

  ▪ Hierarchical routing

❑ Routing in the Internet

  ▪ RIP

  ▪ OSPF

  ▪ BGP

❑ Broadcast and multicast routing

IP Addressing: introduction

- ❑ **IP address:** 32-bit identifier for host, router *interface*

- ❑ *interface:* connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one interface
  - IP addresses associated with each interface

223.1.1.1

223.1.2.1

223.1.1.2

223.1.1.4   223.1.2.9

223.1.1.3   223.1.3.27

223.1.2.2

223.1.3.1   223.1.3.2
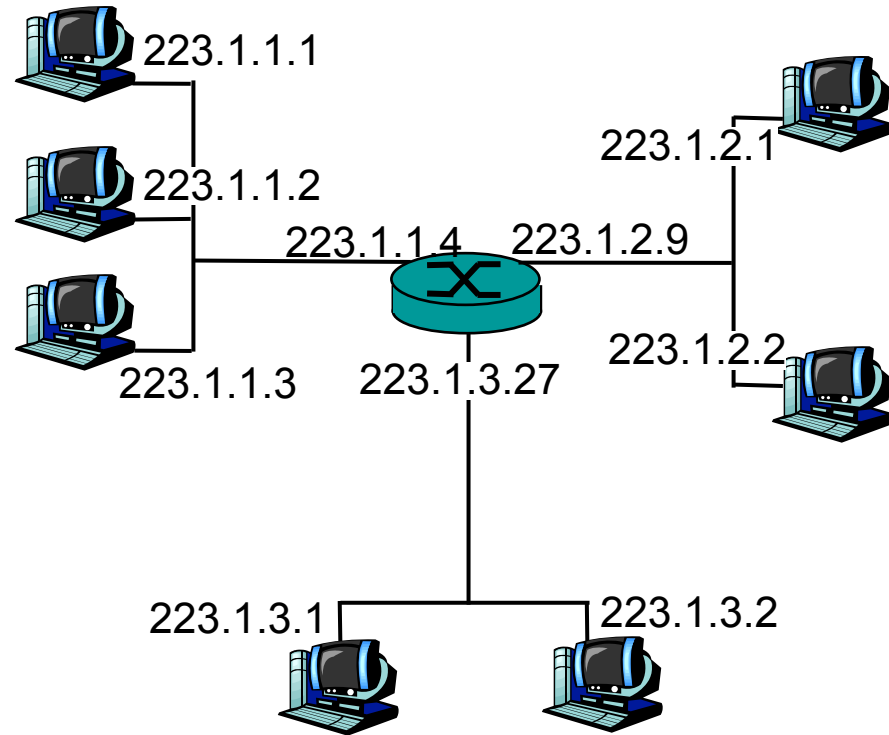
223.1.1.1 = 11011111 00000001 00000001 00000001

                223   1   1   1

Subnets

- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router
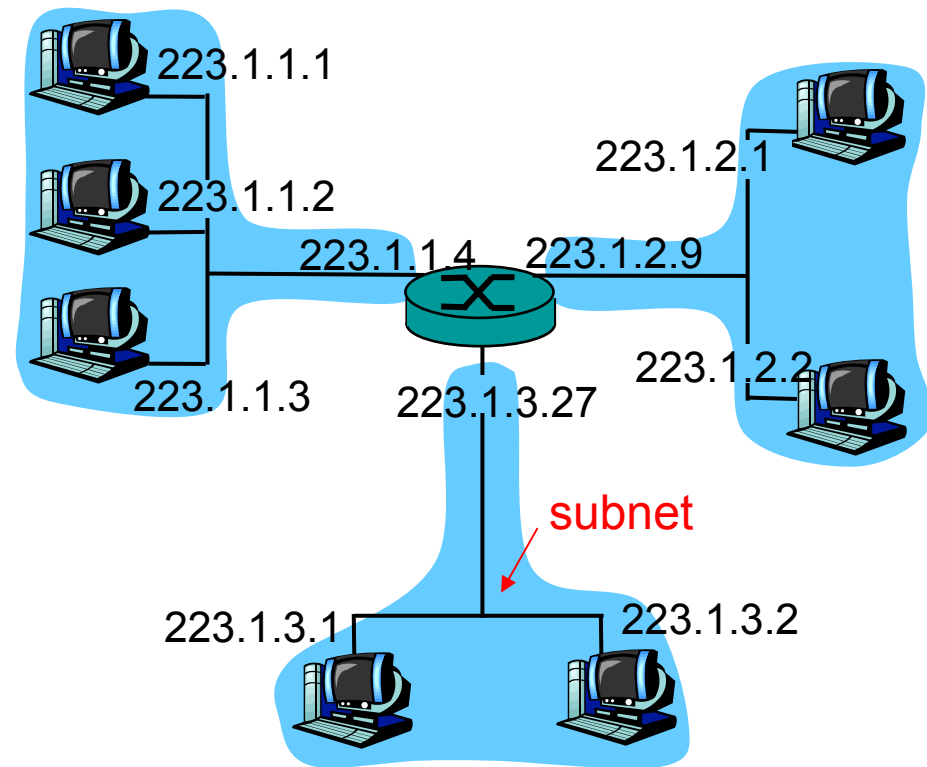
network consisting of 3 subnets

Subnets

Recipe

❑ To determine the subnets, detach each interface from its host or router, creating islands of isolated networks. Each isolated network is called a subnet.
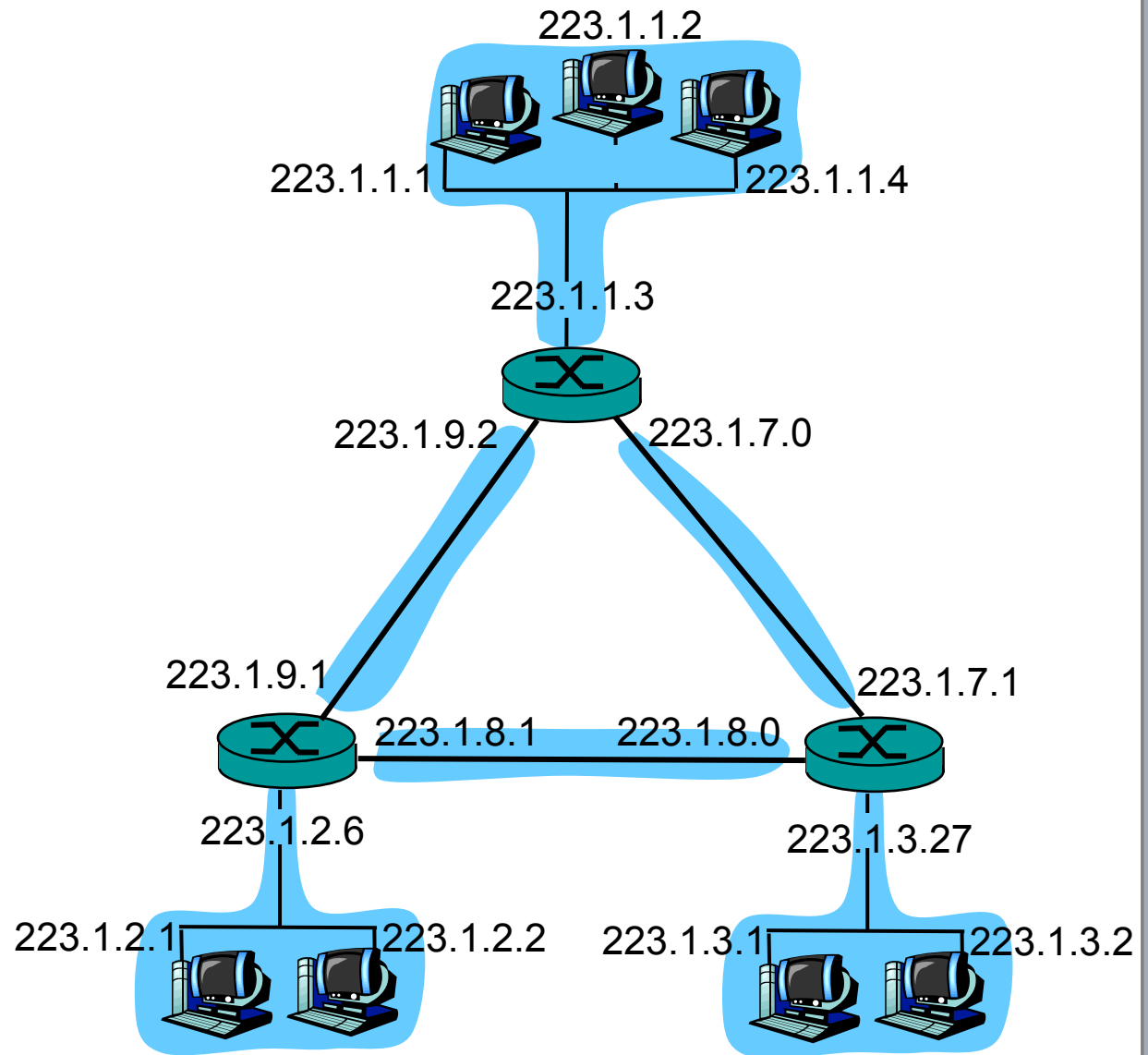
223.1.1.0/24

223.1.2.0/24

223.1.3.0/24

Subnet mask: /24

How many?

223.1.1.2

223.1.1.1   223.1.1.4

223.1.1.3

223.1.9.2   223.1.7.0

223.1.9.1   223.1.7.1

223.1.8.1   223.1.8.0

223.1.2.6   223.1.3.27

223.1.2.1   223.1.2.2   223.1.3.1   223.1.3.2
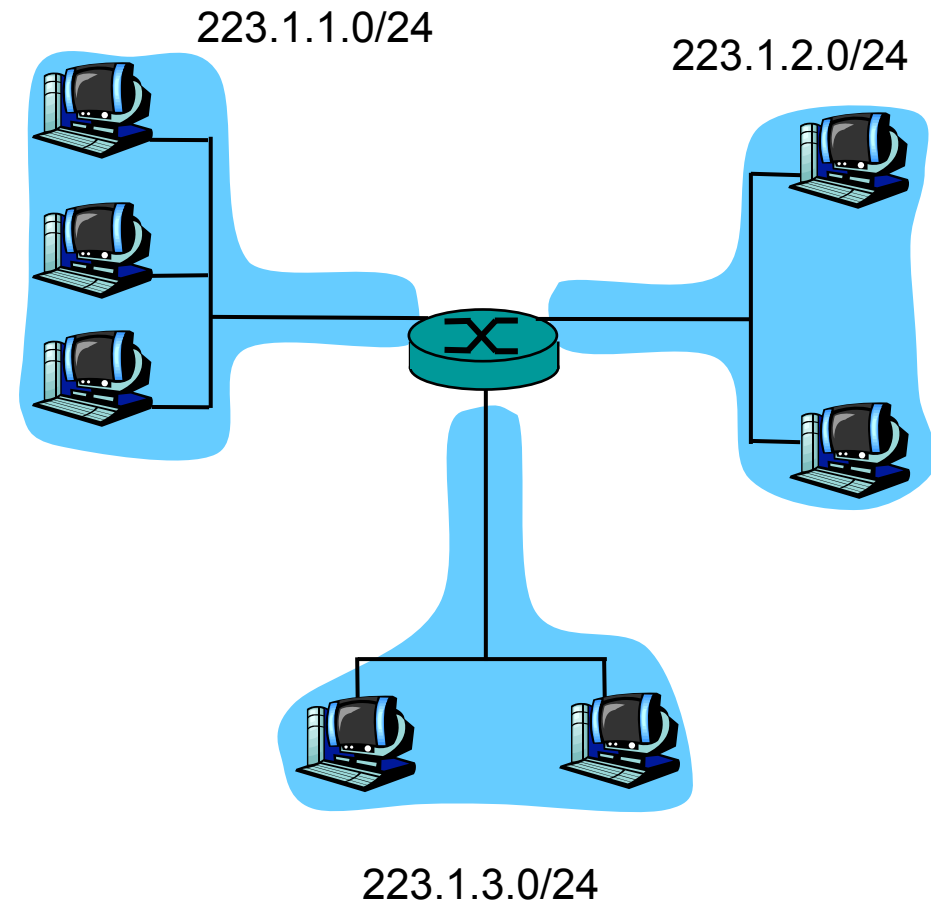
IP addressing: CIDR

CIDR: Classless InterDomain Routing

- subnet portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in subnet portion of address

subnet part ←——————————→ host part ←——→

11001000 00010111 0001000 0 00000000

200.23.16.0/23
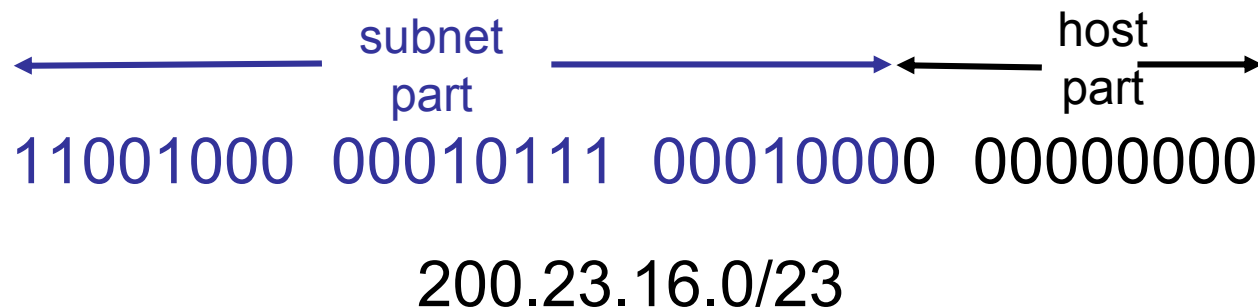
IP addresses: how to get one?

<u>Q:</u> How does a *host* get IP address?

❑ hard-coded by system admin in a file
- Windows: control-panel->network->configuration->tcp/ip->properties
- UNIX: /etc/rc.config

❑ DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
- "plug-and-play"

DHCP: Dynamic Host Configuration Protocol

- Goal: allow host to dynamically obtain its IP address from network server when it joins network
  - Can renew its lease on address in use
  - Allows reuse of addresses (only hold address while connected an "on")
  - Support for mobile users who want to join network (more shortly)
- DHCP overview:
  - host broadcasts "DHCP discover" msg
  - DHCP server responds with "DHCP offer" msg
  - host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

DHCP client-server scenario

A 223.1.1.1

223.1.1.2

223.1.1.4   223.1.2.9

B

223.1.1.3   223.1.3.27

223.1.3.1   223.1.3.2

DHCP server   223.1.2.1

223.1.2.2   E

arriving DHCP client needs address in this network

DHCP client-server scenario

DHCP server: 223.1.2.5

arriving client

DHCP discover

src : 0.0.0.0, 68
dest.: 255.255.255.255,67
yiaddr: 0.0.0.0
transaction ID: 654

DHCP offer

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 654
Lifetime: 3600 secs

DHCP request

src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs

time

DHCP ACK

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs

IP addresses: how to get one?

Q: How does *network* get subnet part of IP addr?

A: gets allocated portion of its provider ISP's address space

ISP's block   11001000 00010111 00010000 00000000   200.23.16.0/20

Organization 0   11001000 00010111 00010000 00000000   200.23.16.0/23
Organization 1   11001000 00010111 00010010 00000000   200.23.18.0/23
Organization 2   11001000 00010111 00010100 00000000   200.23.20.0/23
    ...   .
Organization 7   11001000 00010111 00011110 00000000   200.23.30.0/23

Hierarchical addressing: route aggregation

Hierarchical addressing allows efficient advertisement of routing information:

Organization 0
200.23.16.0/23

Organization 1
200.23.18.0/23

Organization 2
200.23.20.0/23

Organization 7
200.23.30.0/23

Fly-By-Night-ISP

"Send me anything with addresses beginning 200.23.16.0/20"

Internet

ISPs-R-Us

"Send me anything with addresses beginning 199.31.0.0/16"
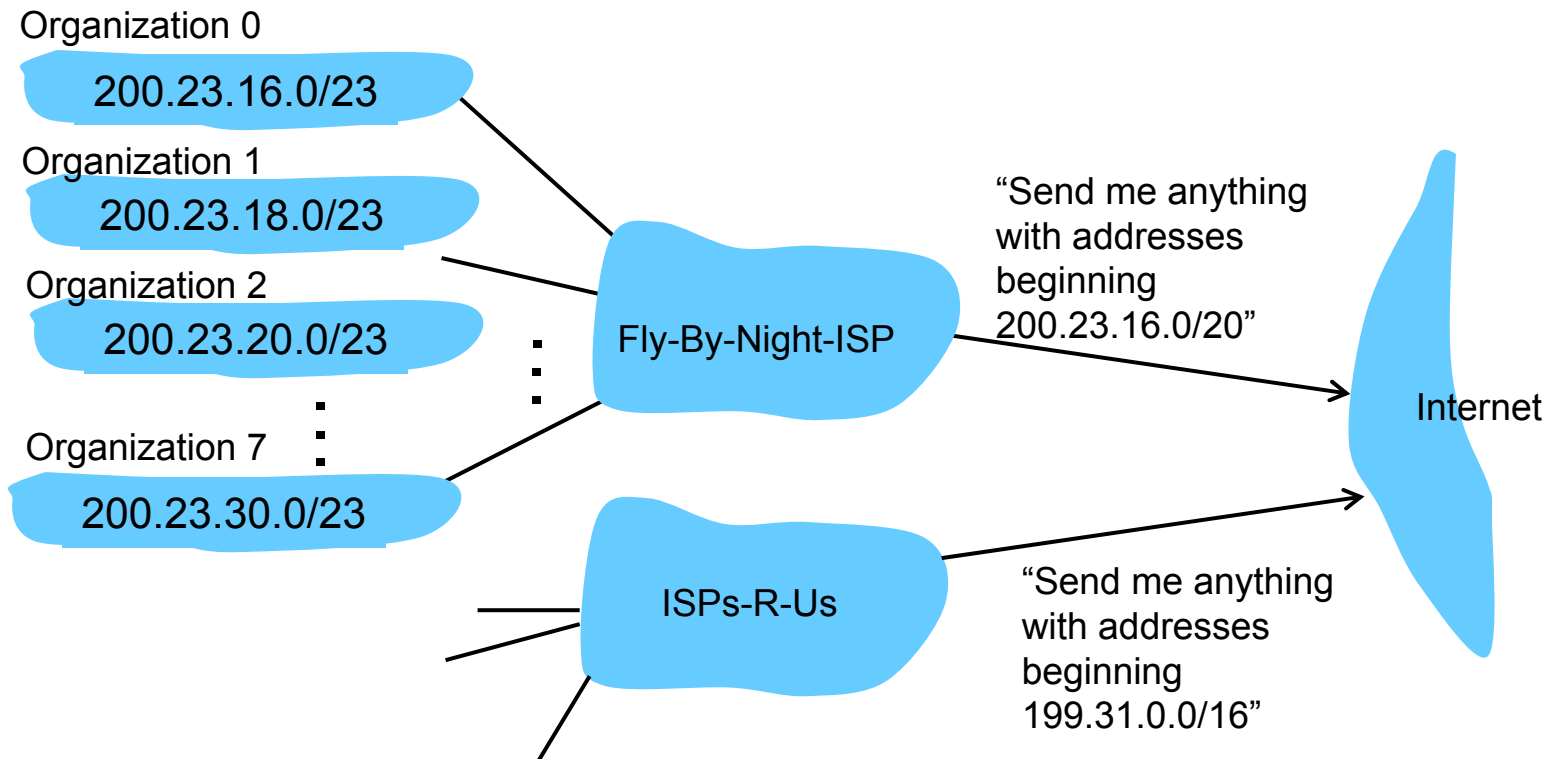
Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1

Organization 0
200.23.16.0/23

Organization 2
200.23.20.0/23

Organization 7
200.23.30.0/23

Organization 1
200.23.18.0/23

Fly-By-Night-ISP

"Send me anything with addresses beginning 200.23.16.0/20"

ISPs-R-Us

"Send me anything with addresses beginning 199.31.0.0/16 or 200.23.18.0/23"
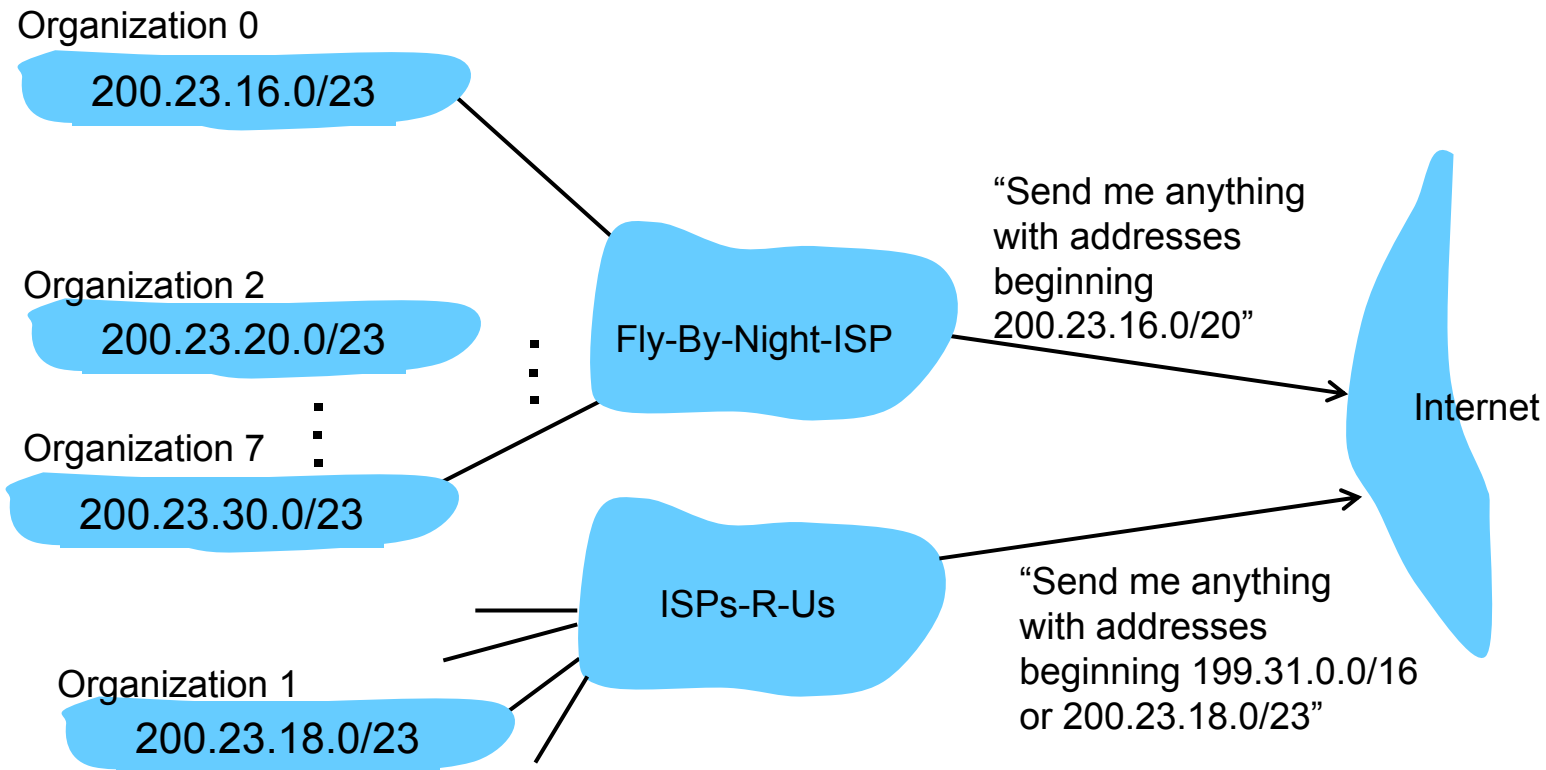
Internet

IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

Chapter 4: Network Layer

Part 1

- ❏ Introduction
- ❏ **IP: Internet Protocol**
  - ▪ Datagram format
  - ▪ IPv4 addressing
  - ▪ **ICMP**

Part 2

- ❏ IPv6
- ❏ Virtual circuit and datagram networks
- ❏ What's inside a router
- ❏ NAT

Part 3

- ❏ Routing algorithms
  - ▪ Link state
  - ▪ Distance Vector
  - ▪ Hierarchical routing
- ❏ Routing in the Internet
  - ▪ RIP
  - ▪ OSPF
  - ▪ BGP
- ❏ Broadcast and multicast routing

ICMP: Internet Control Message Protocol

- used by hosts & routers to communicate network-level information
    - error reporting: unreachable host, network, port, protocol
    - echo request/reply (used by ping)
- network-layer "above" IP:
    - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Traceroute and ICMP

- Source sends series of UDP segments to dest
  - First has TTL =1
  - Second has TTL=2, etc.
  - Unlikely port number
- When nth datagram arrives to nth router:
  - Router discards datagram
  - And sends to source an ICMP message (type 11, code 0)
  - Message includes name of router& IP address

- When ICMP message arrives, source calculates RTT
- Traceroute does this 3 times

Stopping criterion

- UDP segment eventually arrives at destination host
- Destination returns ICMP "dest port unreachable" packet (type 3, code 3)
- When source gets this ICMP, stops.
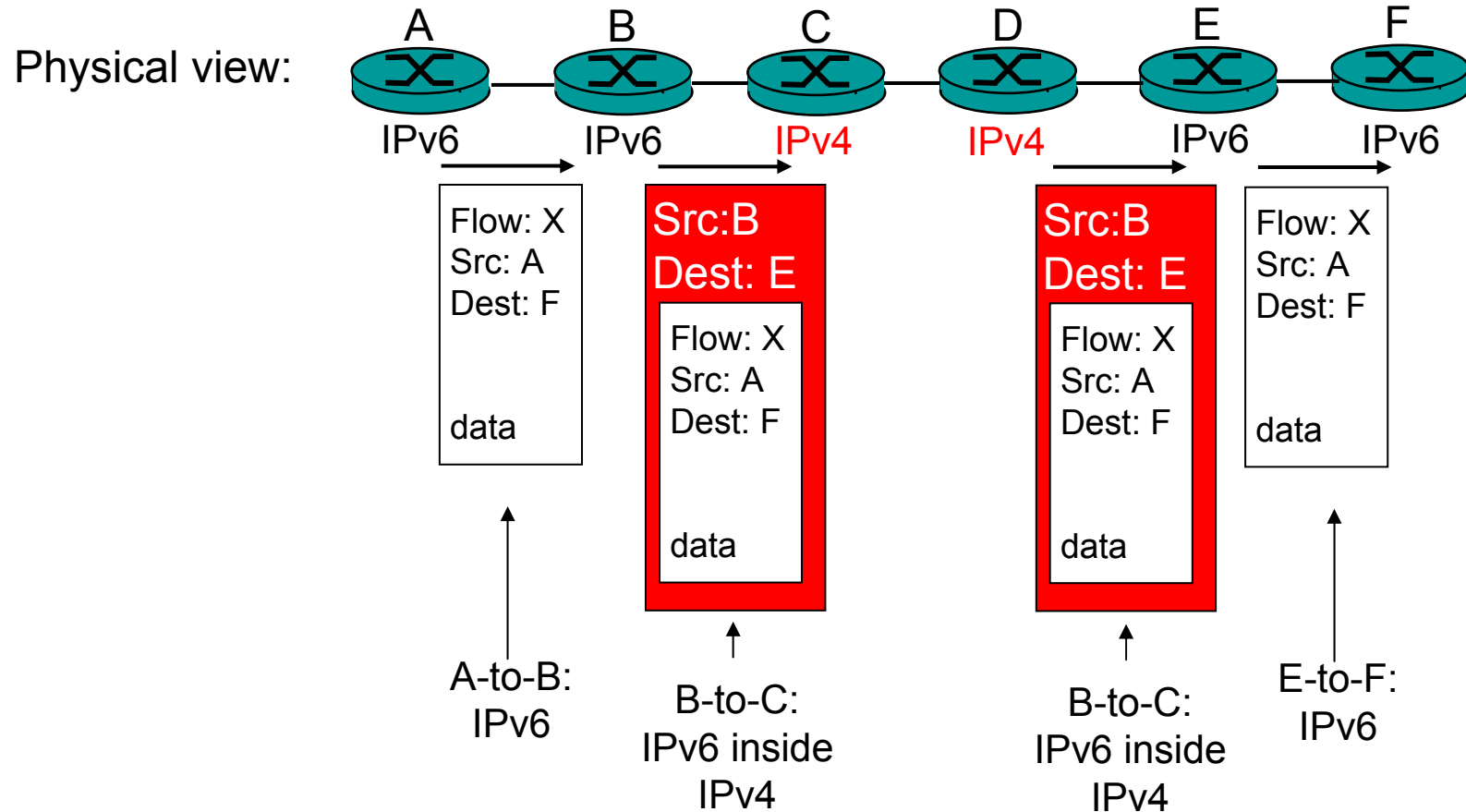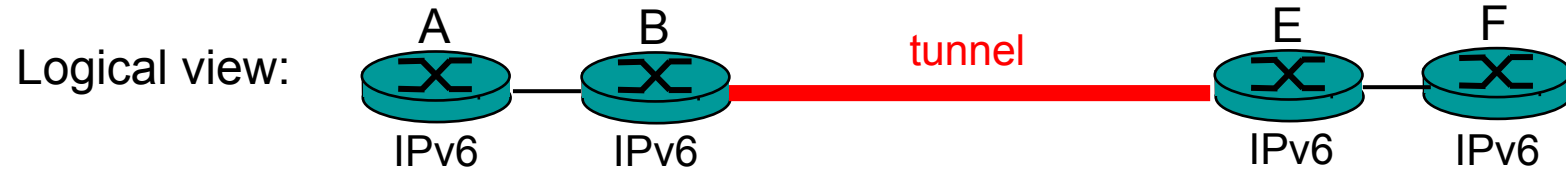
Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- **IPv6**
- Virtual circuit and datagram networks
- What's inside a router
- NAT

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- Broadcast and multicast routing

IPv6

- **Initial motivation:** 32-bit address space soon to be completely allocated.

- Additional motivation:
    - header format helps speed processing/forwarding
    - header changes to facilitate QoS

  IPv6 datagram format:
    - fixed-length 40 byte header
    - no fragmentation allowed

Priority: identify priority among datagrams in flow
Flow Label: identify datagrams in same "flow."
            (concept of"flow" not well defined).
Next header: identify upper layer protocol for data

ver	pri	flow label		
payload len		next hdr		hop limit
source address (128 bits)				
destination address (128 bits)				
data				

⟵ **32 bits** ⟶
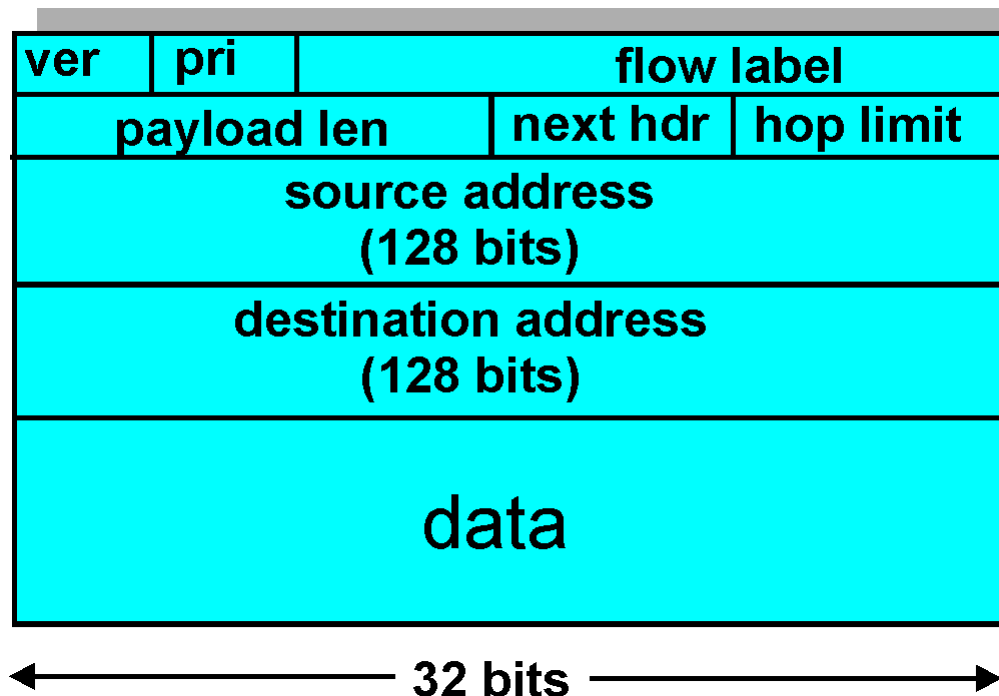
Other Changes from IPv4

- *Checksum*: removed entirely to reduce processing time at each hop

- *Options:* allowed, but outside of header, indicated by "Next Header" field

- *ICMPv6:* new version of ICMP

  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition From IPv4 To IPv6

❑ Not all routers can be upgraded simultaneous

  ▪ no "flag days"

  ▪ How will the network operate with mixed IPv4 and IPv6 routers?

❑ *Tunneling:* IPv6 carried as payload in IPv4 datagram among IPv4 routers

Tunneling
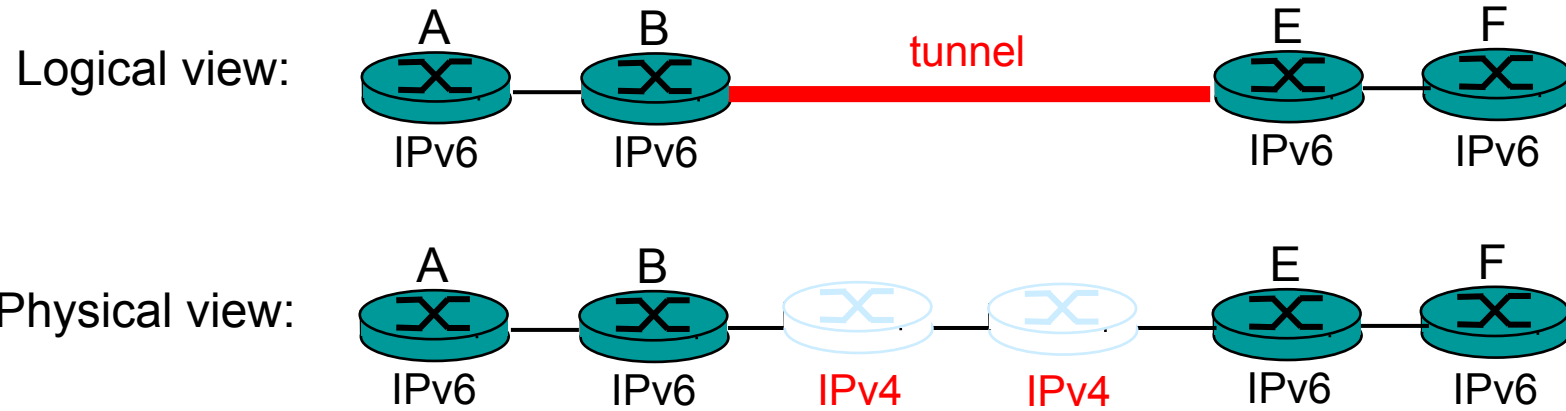
Logical view:

A — B ——— tunnel ——— E — F
IPv6   IPv6   IPv6   IPv6

Physical view:

A — B —— — —— E — F
IPv6   IPv6   IPv4   IPv4   IPv6   IPv6

Tunneling

Logical view:

A — B ======== tunnel ======== E — F

IPv6   IPv6   IPv6   IPv6

Physical view:

A — B — C — D — E — F

IPv6   IPv6   IPv4   IPv4   IPv6   IPv6

Flow: X
Src: A
Dest: F

data

Src:B
Dest: E

Flow: X
Src: A
Dest: F

data

Src:B
Dest: E

Flow: X
Src: A
Dest: F

data

Flow: X
Src: A
Dest: F

data

A-to-B:
IPv6

B-to-C:
IPv6 inside
IPv4

B-to-C:
IPv6 inside
IPv4

E-to-F:
IPv6

Chapter 4: Network Layer

Part 1

❏ Introduction

❏ IP: Internet Protocol

  ▪ Datagram format

  ▪ IPv4 addressing

  ▪ ICMP

Part 2

❏ IPv6

❏ **Virtual circuit and datagram networks**

❏ What's inside a router

❏ NAT

Part 3

❏ Routing algorithms

  ▪ Link state

  ▪ Distance Vector

  ▪ Hierarchical routing

❏ Routing in the Internet

  ▪ RIP

  ▪ OSPF

  ▪ BGP

❏ Broadcast and multicast routing

Connection setup

❑ In addition to routing and forwarding, 3rd important function in some network architectures:

  ▪ ATM, frame relay, X.25

❑ before datagrams flow, two end hosts and intervening switches/routers establish virtual connection

  ▪ switches/routers get involved

❑ network vs transport layer connection service:

  ▪ network: between two hosts (may also involve intervening switches/routers in case of VCs)

  ▪ transport: between two processes

Network service model

Q: What *service model* for "channel" transporting datagrams from sender to receiver?

Example services for individual datagrams:

❑ guaranteed delivery

❑ guaranteed delivery with less than 40 msec delay

Example services for a flow of datagrams:

❑ in-order datagram delivery

❑ guaranteed minimum bandwidth to flow

❑ restrictions on changes in inter-packet spacing

Network layer service models

Network Architecture	Service Model	Guarantees ?				Congestion feedback
		Bandwidth	Loss	Order	Timing	
Internet	best effort	none	no	no	no	no (inferred via loss)
ATM	CBR	constant rate	yes	yes	yes	no congestion
ATM	VBR	guaranteed rate	yes	yes	yes	no congestion
ATM	ABR	guaranteed minimum	no	yes	no	yes
ATM	UBR	none	no	yes	no	no

Network layer connection and connection-less service

❑ datagram network provides network-layer connectionless service

❑ VC network provides network-layer connection service

❑ analogous to the transport-layer services, but:

  ▪ service: host-to-host

  ▪ no choice: network provides one or the other

  ▪ implementation: in network core

Virtual circuits

"source-to-dest path behaves much like telephone circuit"

- performance-wise
- network actions along source-to-dest path

- ❑ call setup, teardown for each call *before* data can flow
- ❑ each packet carries VC identifier (not destination host address)
- ❑ *every* router on source-dest path maintains "state" for each passing connection
- ❑ link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)

VC implementation

a VC consists of:

1. path from source to destination

2. VC numbers, one number for each link along path

3. entries in forwarding tables in routers along path

❑ packet belonging to VC carries VC number (rather than dest address)

❑ VC number can be changed on each link.

- New VC number comes from forwarding table

Forwarding table

VC number

interface number
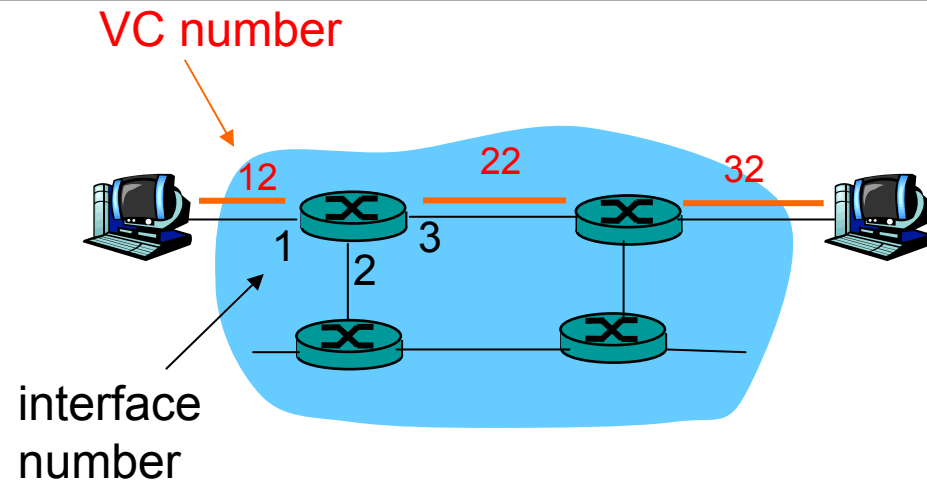
Forwarding table in northwest router:

Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

Routers maintain connection state information!

Virtual circuits: signaling protocols

❑ used to setup, maintain teardown VC

❑ used in ATM, frame-relay, X.25

❑ not used in today's Internet
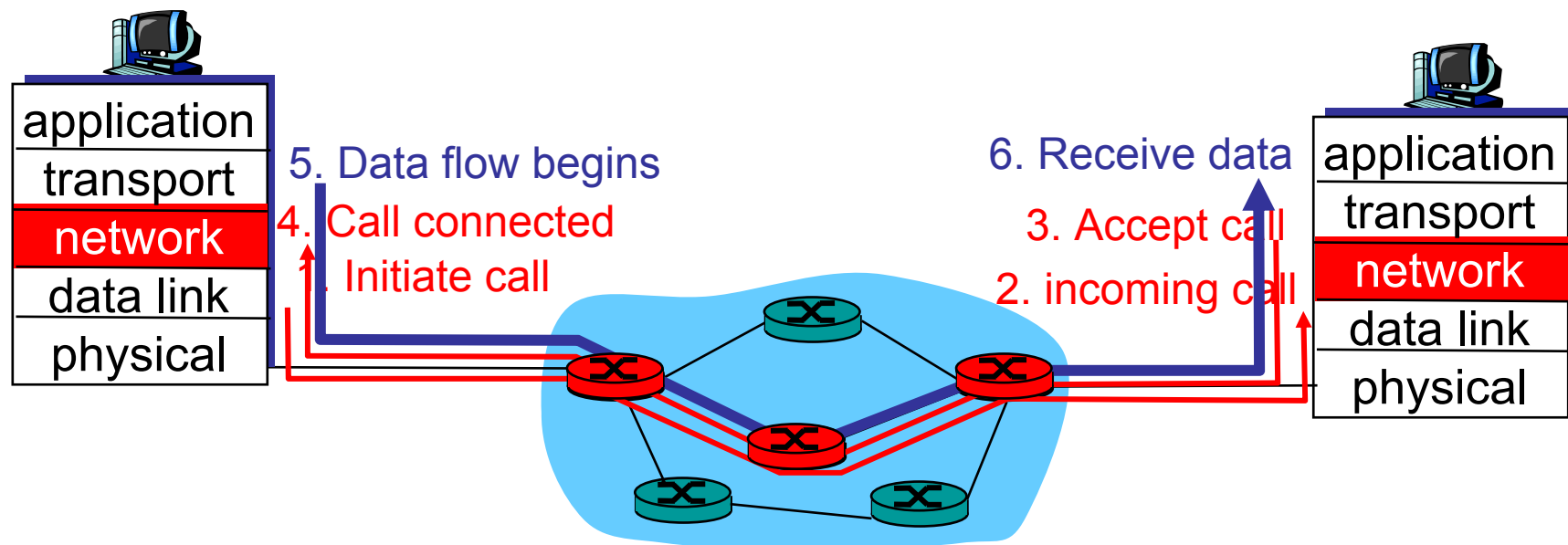
application

transport

network

data link

physical

5. Data flow begins

4. Call connected

1. Initiate call

6. Receive data

3. Accept call

2. incoming call

application

transport

network

data link

physical

Datagram networks

❑ 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

application
transport
network
data link
physical

1. Send data

2. Receive data

application
transport
network
data link
physical
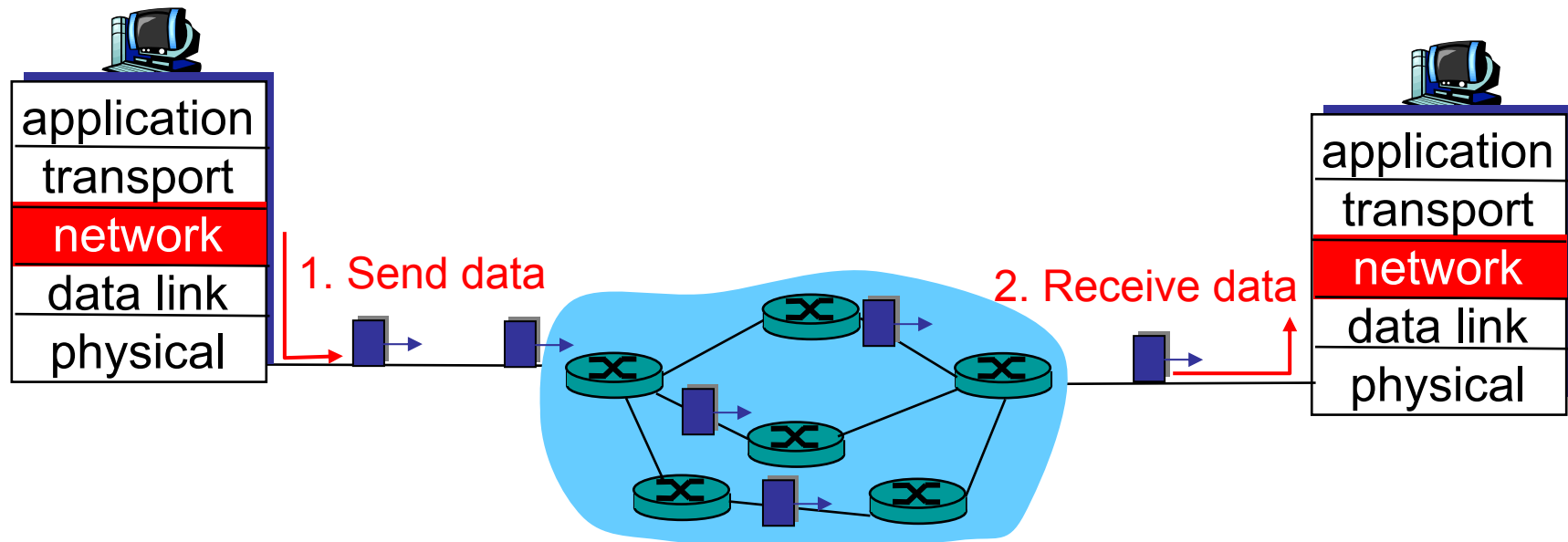
Datagram or VC network: why?

Internet (datagram)

- ❑ data exchange among computers
  - ▪ "elastic" service, no strict timing req.
- ❑ "smart" end systems (computers)
  - ▪ can adapt, perform control, error recovery
  - ▪ simple inside network, complexity at "edge"
- ❑ many link types
  - ▪ different characteristics
  - ▪ uniform service difficult

ATM (VC)

- ❑ evolved from telephony
- ❑ human conversation:
  - ▪ strict timing, reliability requirements
  - ▪ need for guaranteed service
- ❑ "dumb" end systems
  - ▪ telephones
  - ▪ complexity inside network

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol
  ▪ Datagram format
  ▪ IPv4 addressing
  ▪ ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ **What's inside a router**

Part 3

❑ Routing algorithms
  ▪ Link state
  ▪ Distance Vector
  ▪ Hierarchical routing

❑ Routing in the Internet
  ▪ RIP
  ▪ OSPF
  ▪ BGP

❑ Broadcast and multicast routing

First-Generation IP Routers
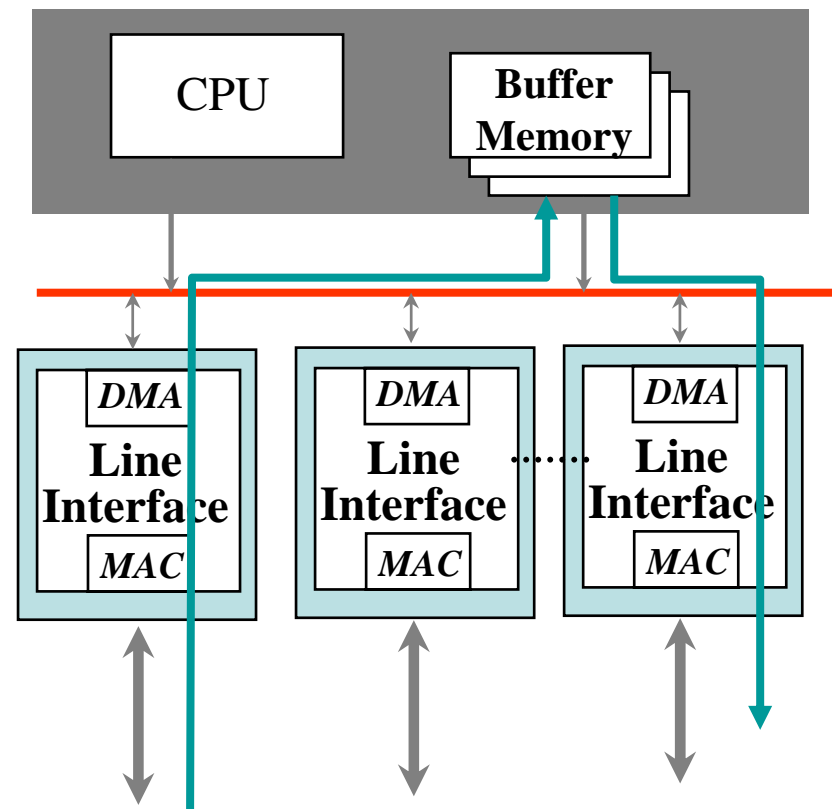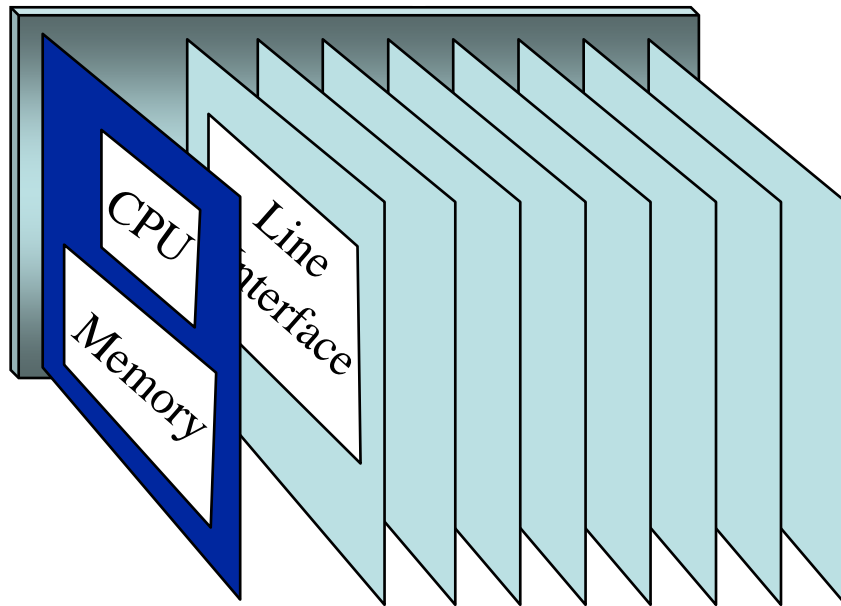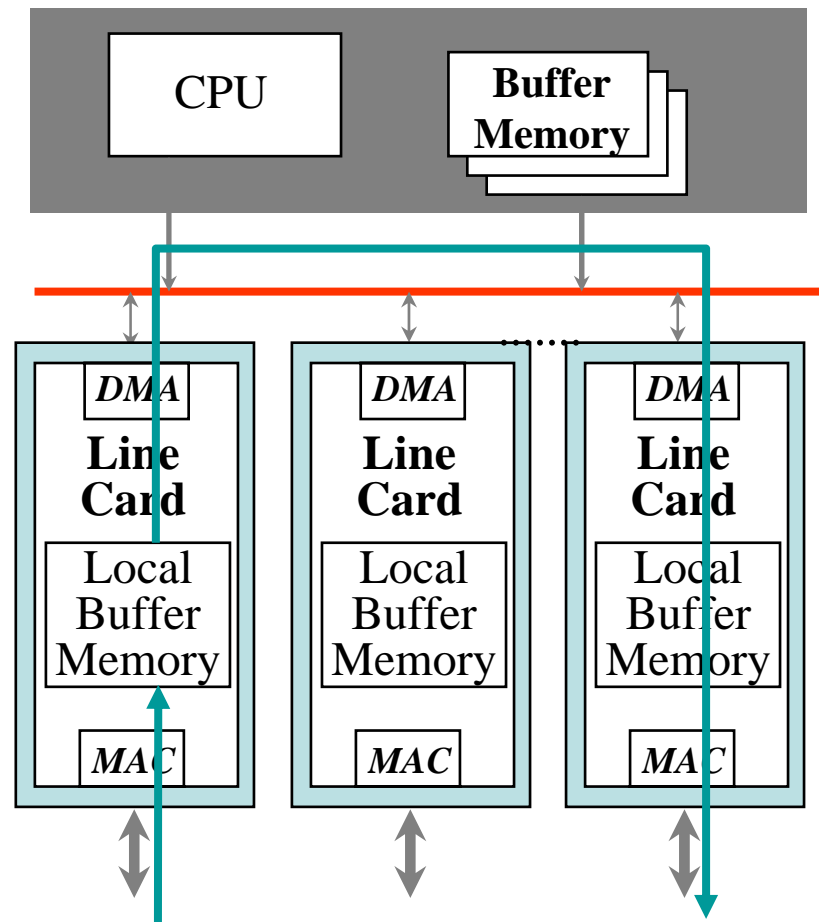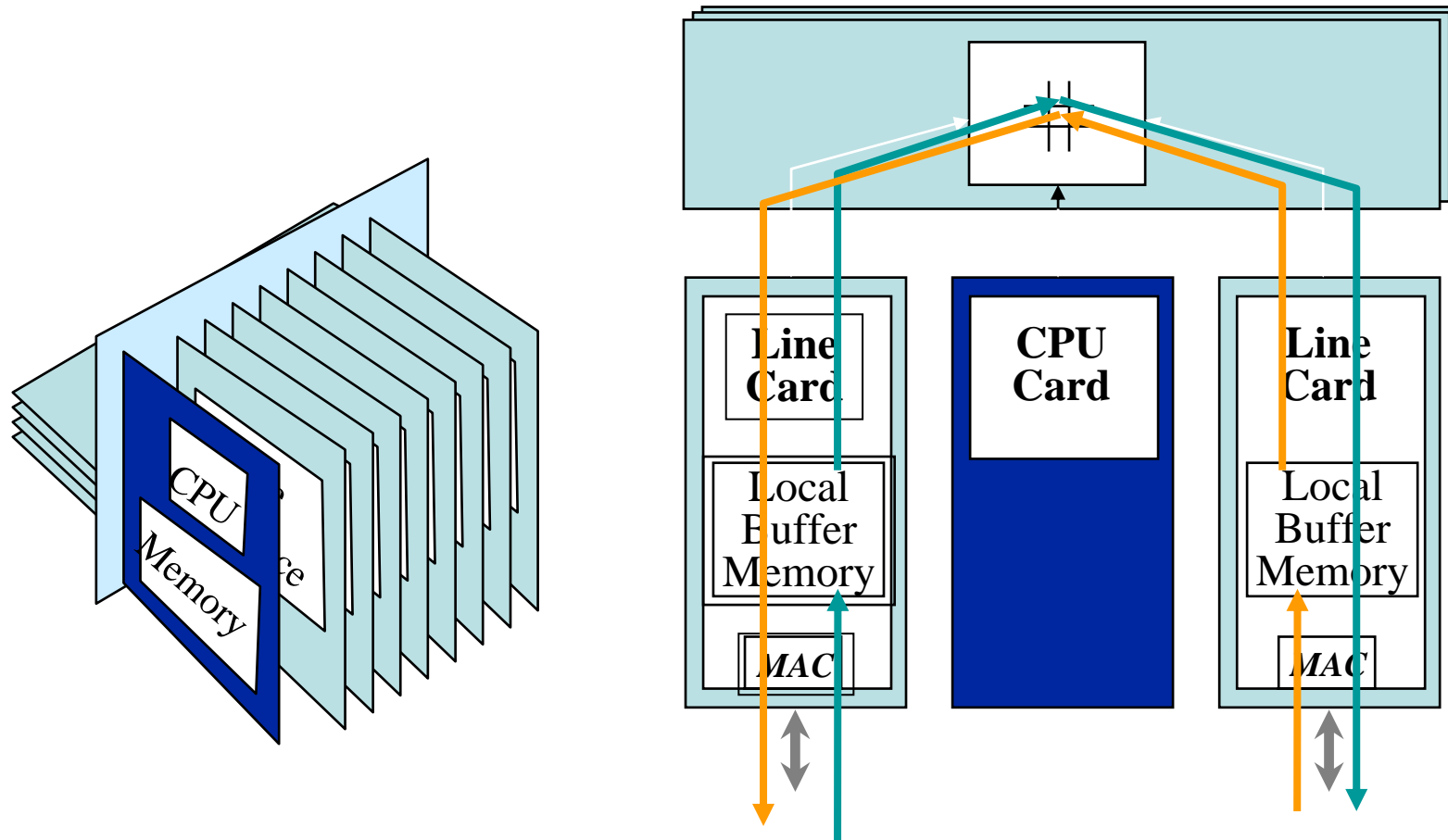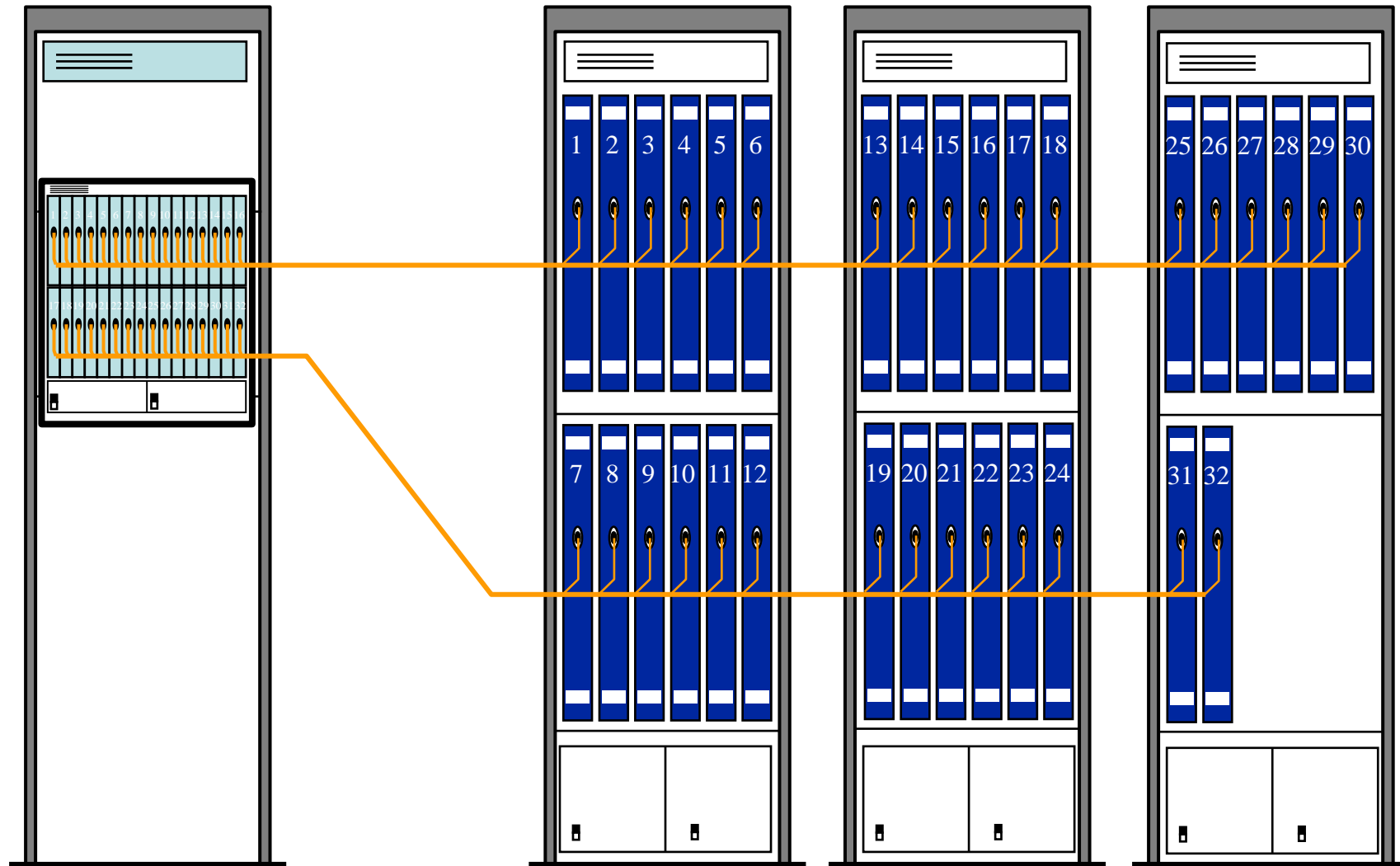
Router Architecture Overview

Two key router functions:

❑ run routing algorithms/protocol (RIP, OSPF, BGP)

❑ *forwarding* datagrams from incoming to outgoing link
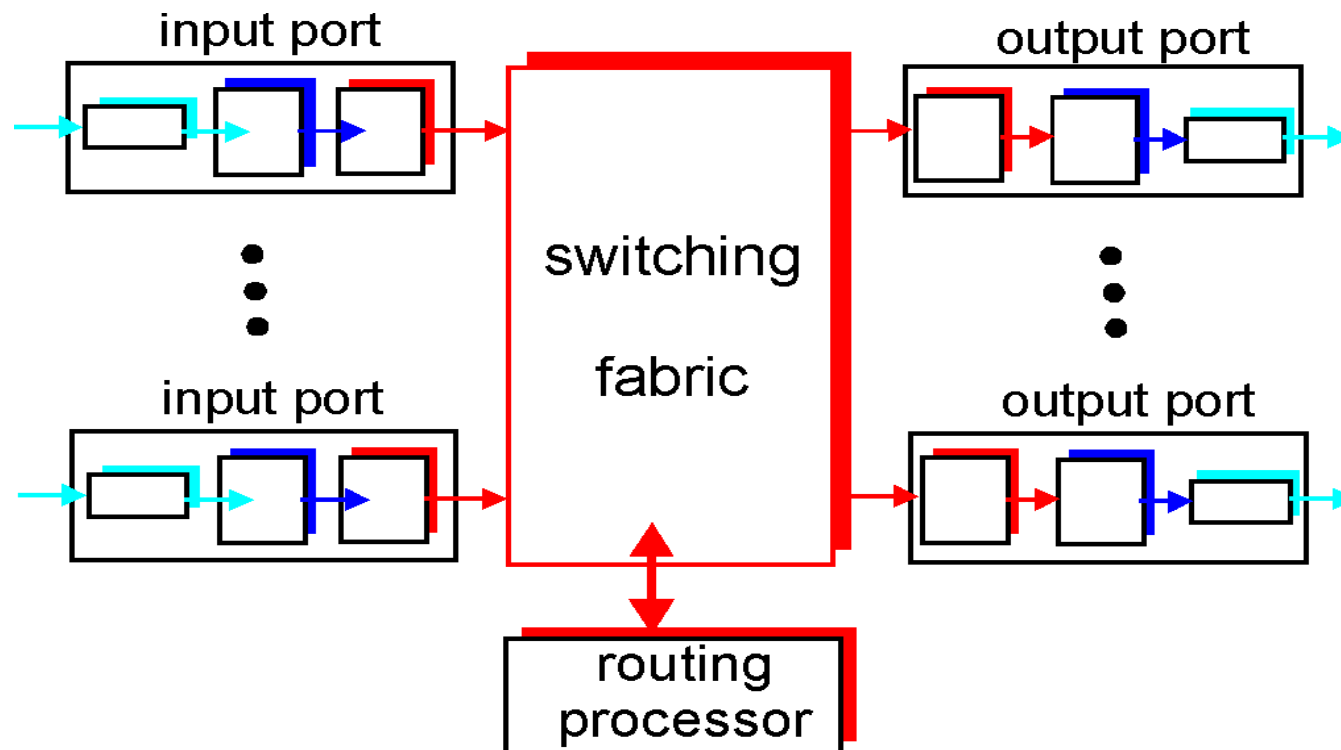
Input Port Functions
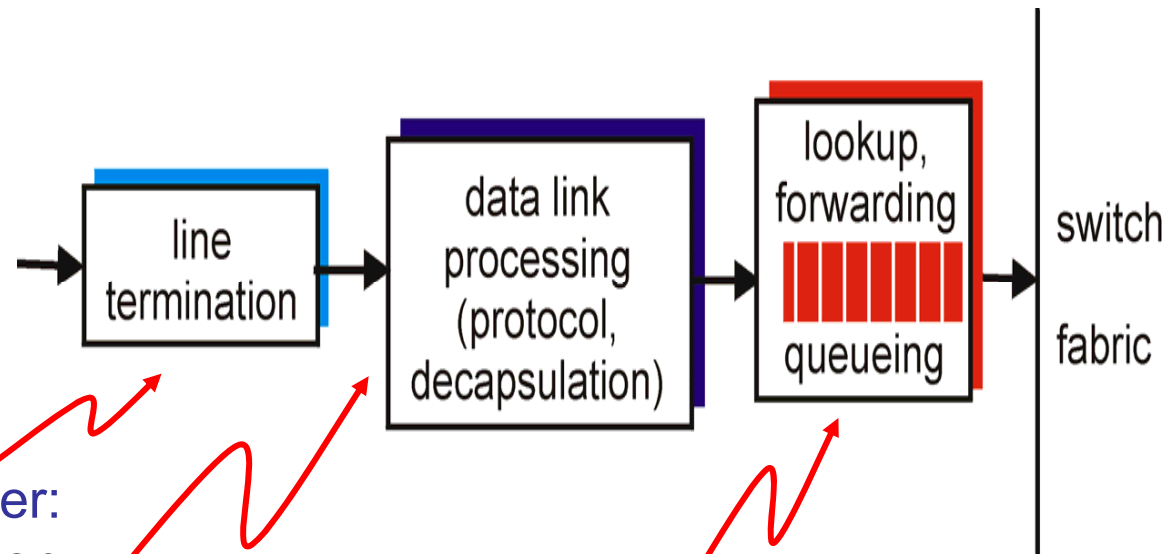
Physical layer:
bit-level reception

Data link layer:
e.g., Ethernet
see chapter 5

Decentralized switching*:*

- given datagram dest., lookup output port using forwarding table in input port memory
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Three types of switching fabrics

memory

bus

crossbar
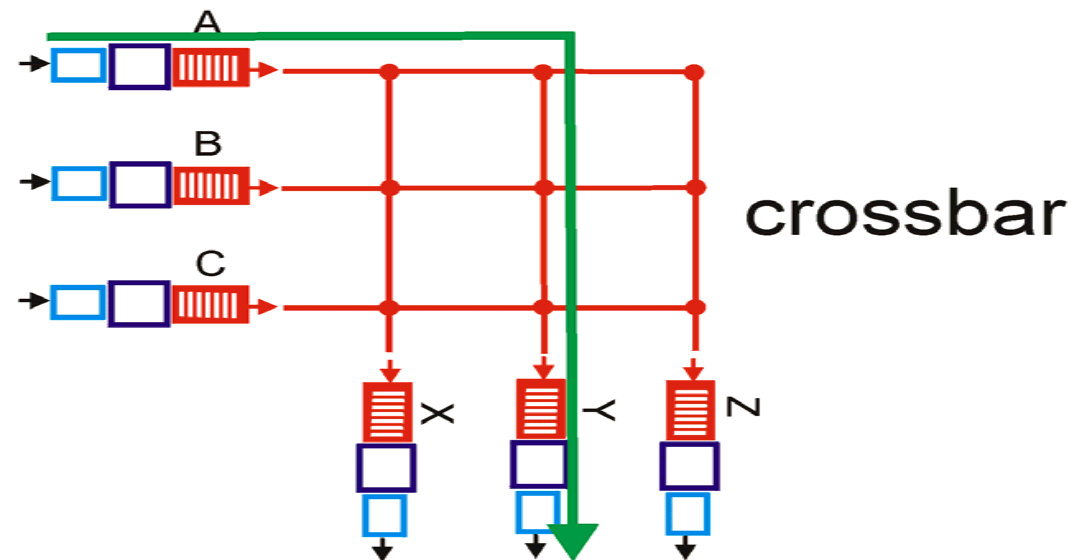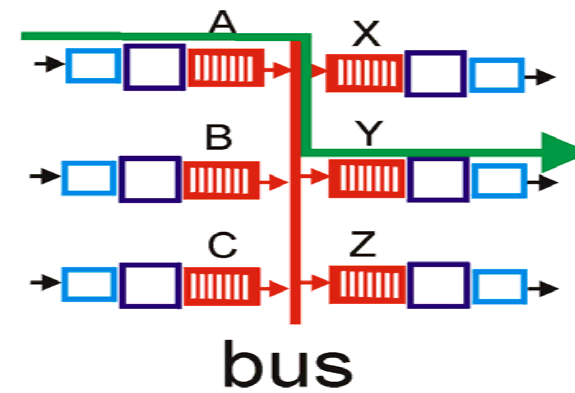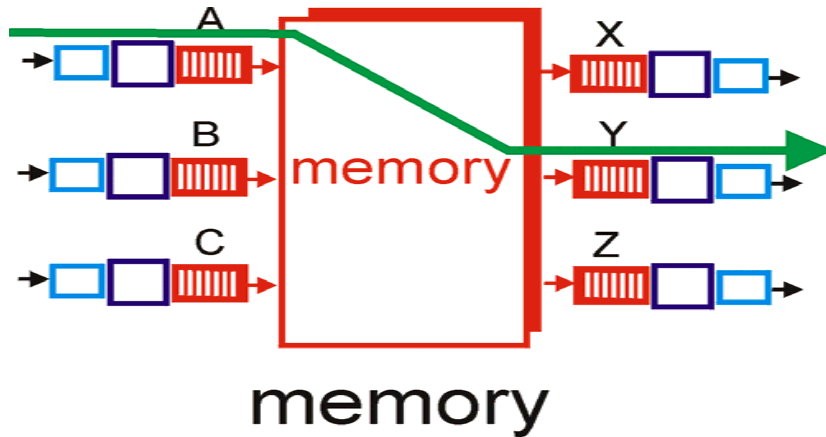
Switching Via Memory

First generation routers:

❑ traditional computers with switching under direct control of CPU

❑ packet copied to system's memory

❑ speed limited by memory bandwidth (2 bus crossings per datagram)

Input
Port

Memory

Output
Port

System Bus
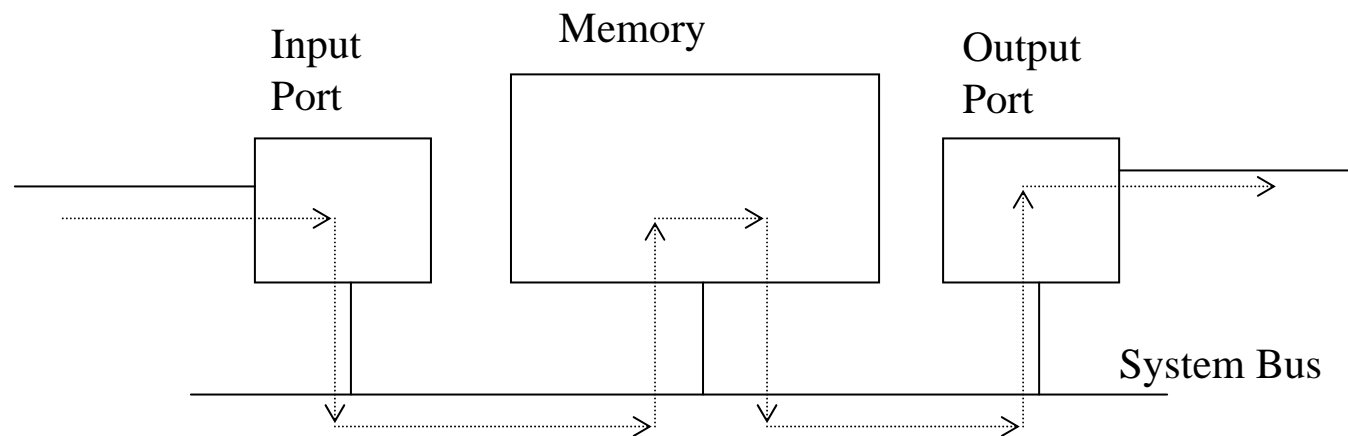
Switching Via a Bus

❑ datagram from input port memory
  to output port memory via a shared bus

❑ bus contention: switching speed limited
  by bus bandwidth

❑ 32 Gbps bus, Cisco 5600: sufficient
  speed for access and enterprise routers

bus
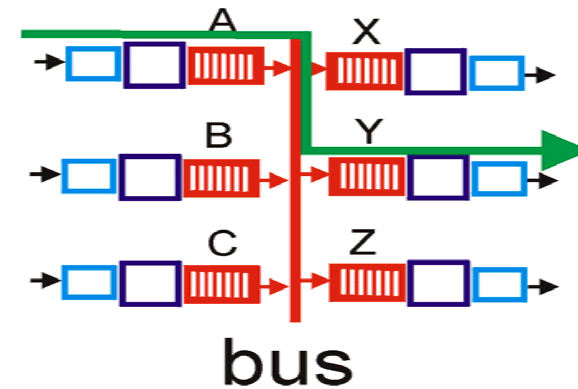
Switching Via An Interconnection Network

- ❑ overcome bus bandwidth limitations
- ❑ Banyan networks, other interconnection nets initially developed to connect processors in multiprocessor
- ❑ advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric.
- ❑ Cisco 12000: switches 60 Gbps through the interconnection network

Banyan network:

For output #7
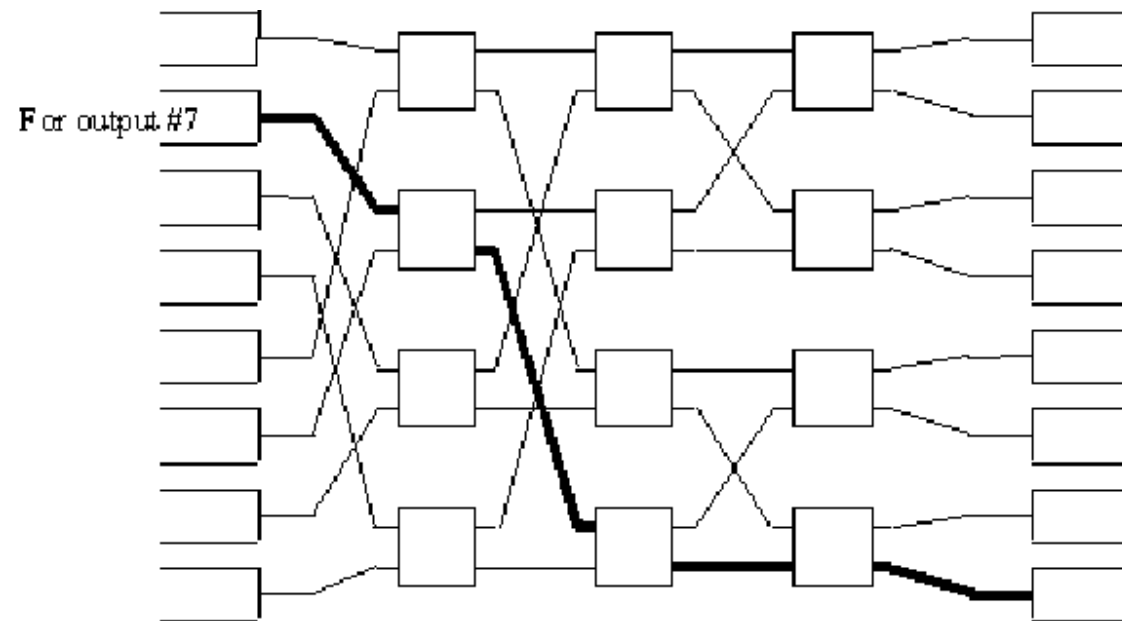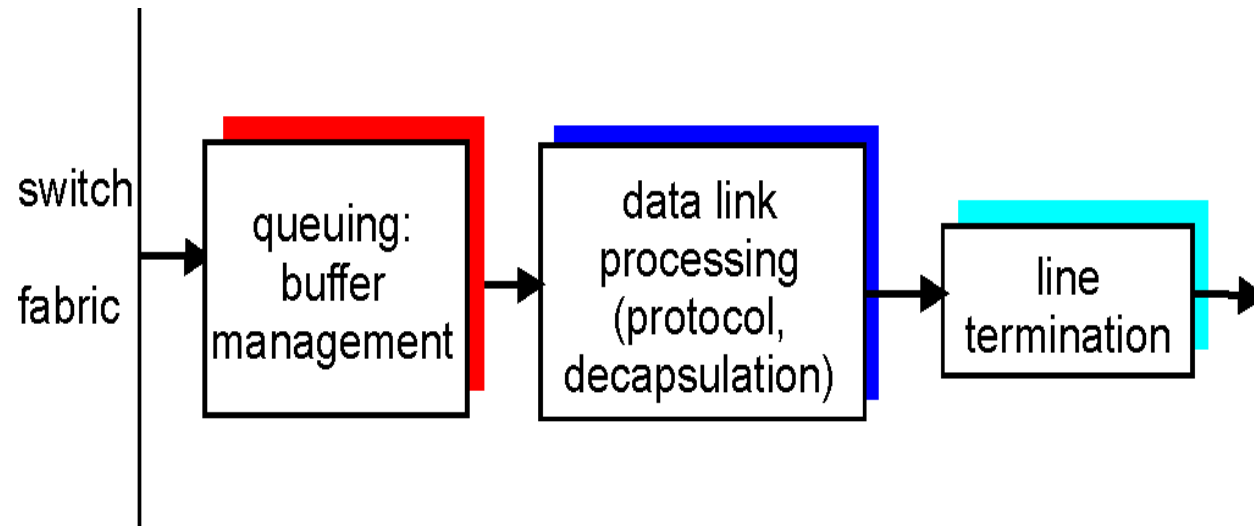
Output Ports

❑ *Buffering* required when datagrams arrive from fabric faster than the transmission rate

❑ *Scheduling discipline* chooses among queued datagrams for transmission

Output port queueing

Output Port Contention at Time *t*

One Packet Time Later
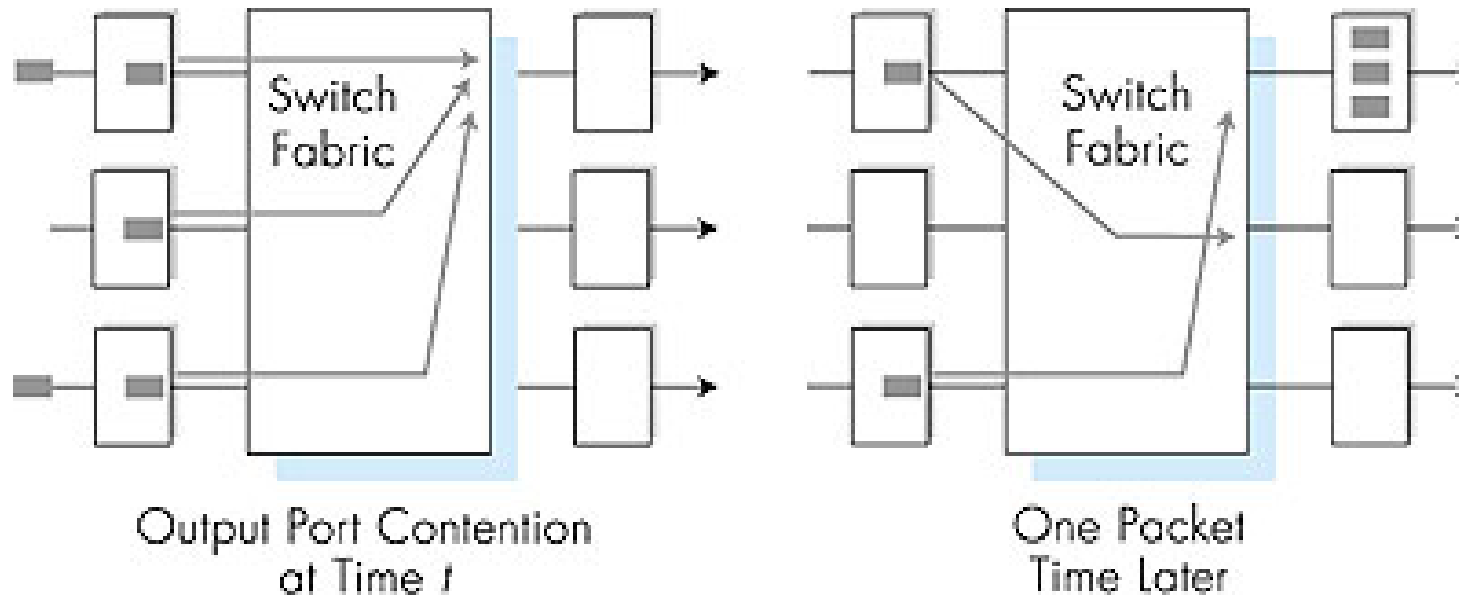
❑ buffering when arrival rate via switch exceeds output line speed

❑ *queueing (delay) and loss due to output port buffer overflow!*

Input Port Queuing

- Fabric slower than input ports combined -> queueing may occur at input queues
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward
- *queueing delay and loss due to input buffer overflow!*
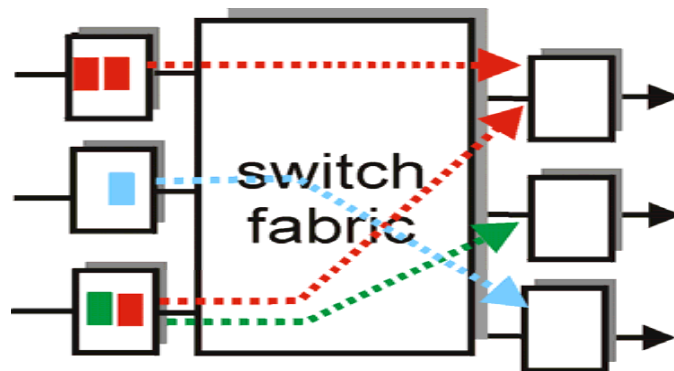
output port contention
at time t - only one red
packet can be transferred
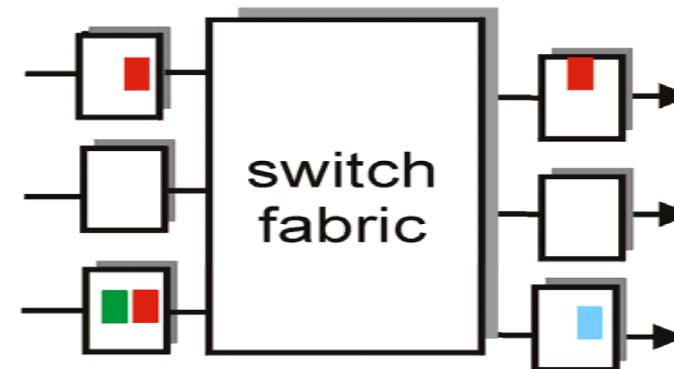
green packet
experiences HOL blocking

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol

  ▪ Datagram format

  ▪ IPv4 addressing

  ▪ ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

Part 3

❑ **Routing algorithms**

  ▪ Link state

  ▪ Distance Vector

  ▪ Hierarchical routing

❑ Routing in the Internet

  ▪ RIP

  ▪ OSPF

  ▪ BGP

❑ Broadcast and multicast routing

Interplay between routing, forwarding

routing algorithm

local forwarding table	
header value	output link
0100	3
0101	2
0111	2
1001	1

value in arriving packet's header

0111

1

3 2
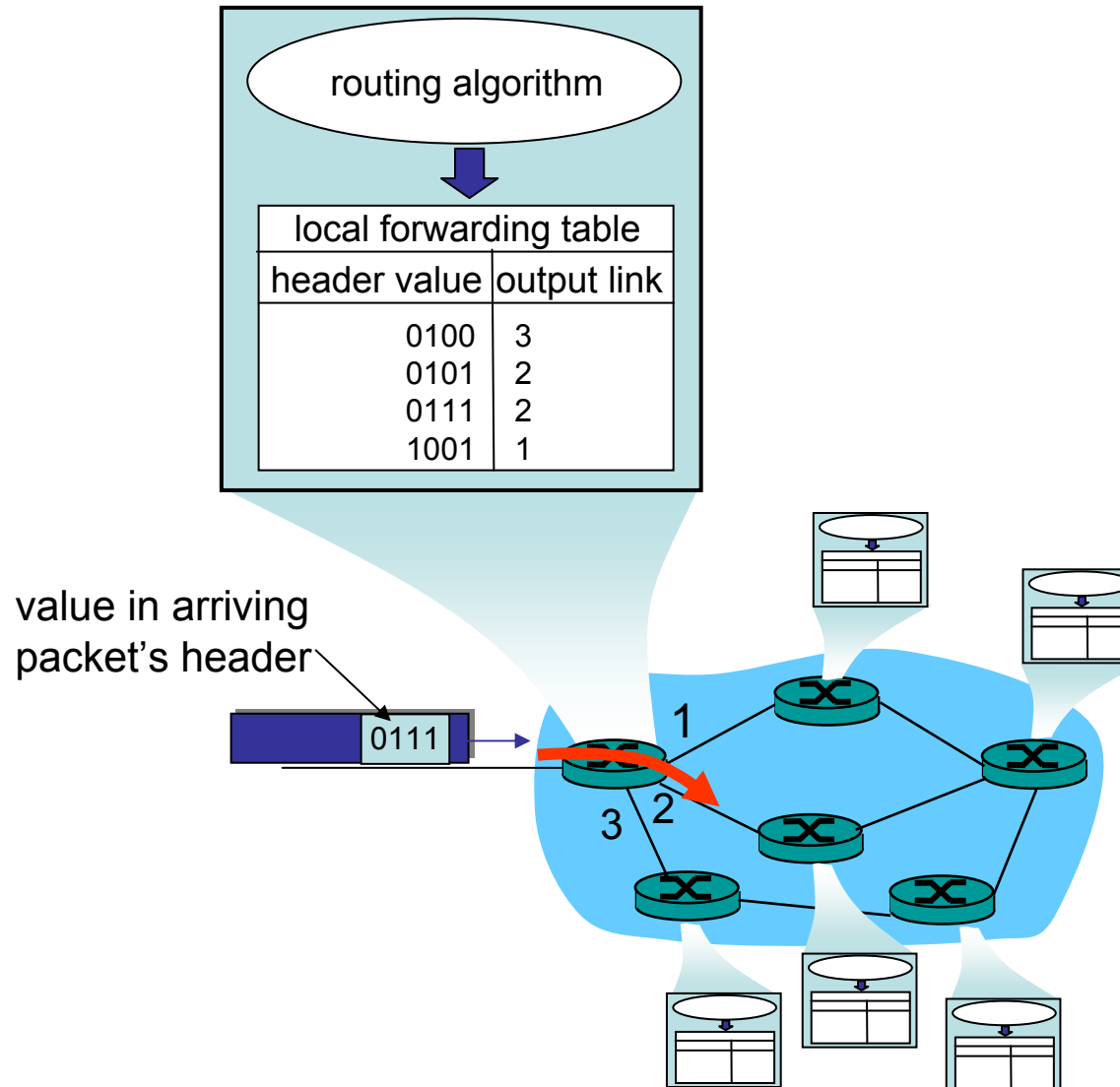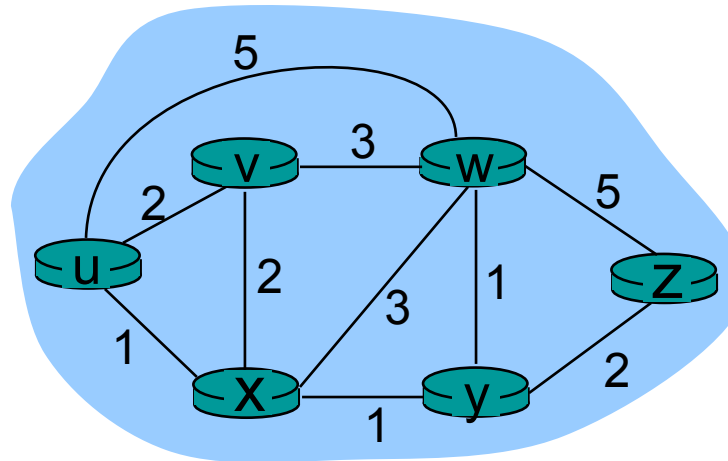
Graph abstraction

Graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Remark: Graph abstraction is useful in other network contexts

Example: P2P, where N is set of peers and E is set of TCP connections

Graph abstraction: costs
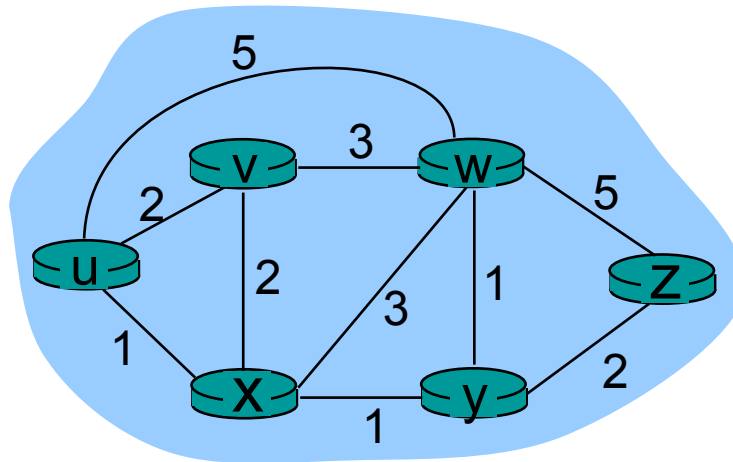
- $c(x,x') = $ cost of link (x,x')

  - 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

Routing Algorithm classification

Global or decentralized information?

Global:

- all routers have complete topology, link cost info
- "link state" algorithms

Decentralized:

- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Static or dynamic?

Static:

- routes change slowly over time

Dynamic:

- routes change more quickly
  - periodic update
  - in response to link cost changes

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- Virtual circuit and datagram networks
- What's inside a router
- NAT

Part 3

- **Routing algorithms**
  - **Link state**
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- Broadcast and multicast routing

A Link-State Routing Algorithm

Dijkstra's algorithm

- net topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same info
- computes least cost paths from one node ('source") to all other nodes
  - gives forwarding table for that node
- iterative: after k iterations, know least cost path to k dest.'s

Notation:

- $c(x,y)$: link cost from node x to y; = ∞ if not direct neighbors
- $D(v)$: current value of cost of path from source to dest. v
- $p(v)$: predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known

1 **Initialization:**
2   N' = {u}
3   for all nodes v
4   if v adjacent to u
5   then D(v) = c(u,v)
6   else D(v) = ∞
7
8 **Loop**
9   find w not in N' such that D(w) is a minimum
10   add w to N'
11   update D(v) for all v adjacent to w and not in N' :
12   D(v) = min(D(v), D(w) + c(w,v))
13   /* new cost to v is either old cost to v or known
14   shortest path cost to w plus cost from w to v */
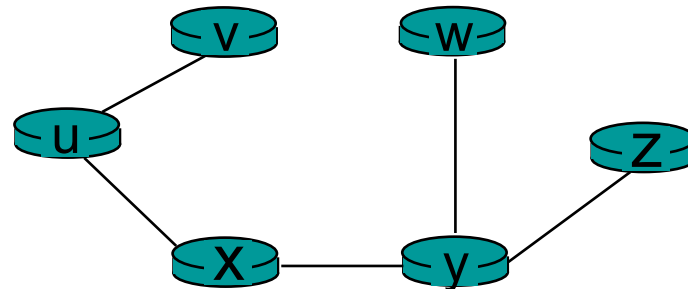15 **until all nodes in N'**

Dijkstra's algorithm: example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

Resulting shortest-path tree from u:

Resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

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: $O(n\log n)$

Oscillations possible:

- ❑ e.g., link cost = amount of carried traffic
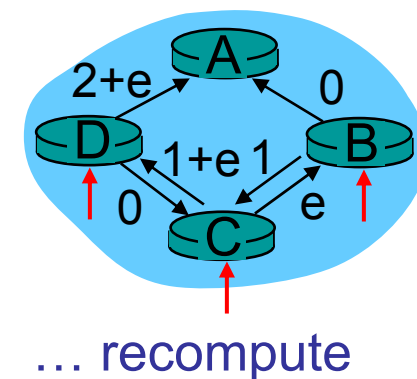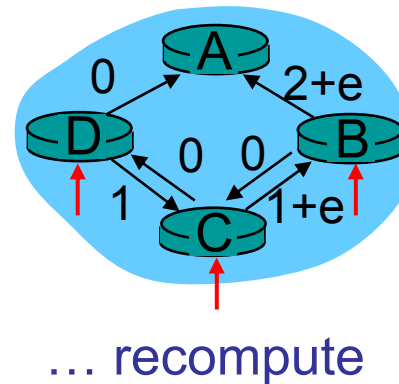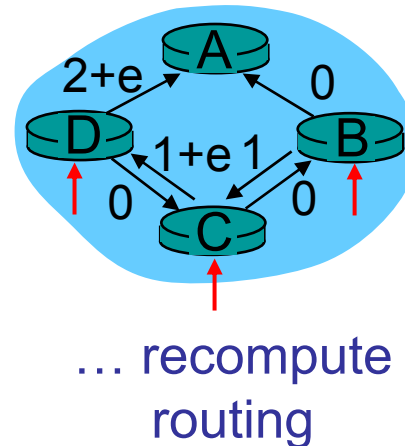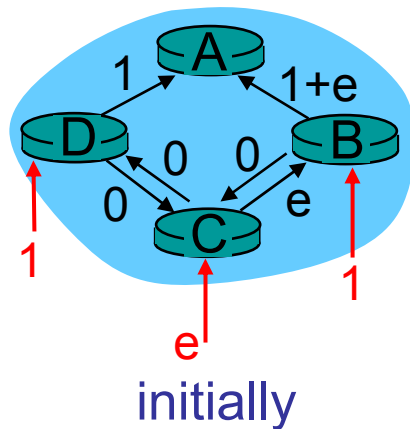
initially … recompute routing … recompute … recompute

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol

- ▪ Datagram format
- ▪ IPv4 addressing
- ▪ ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

❑ NAT

Part 3

❑ **Routing algorithms**

- ▪ Link state
- ▪ **Distance Vector**
- ▪ Hierarchical routing

❑ Routing in the Internet

- ▪ RIP
- ▪ OSPF
- ▪ BGP

❑ Broadcast and multicast routing

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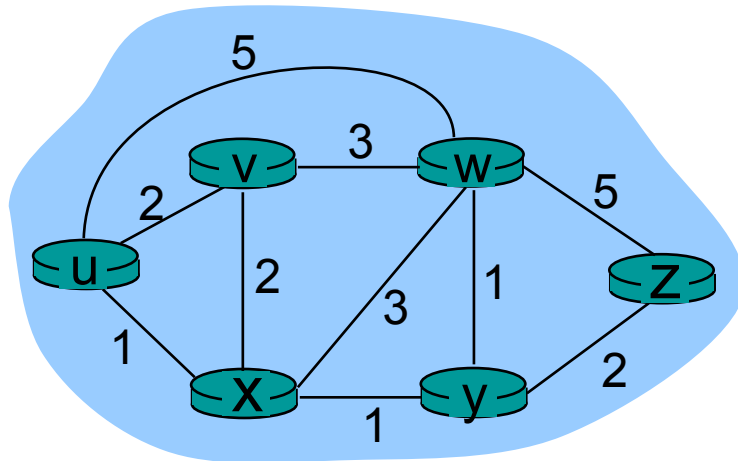
Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

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

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

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 $\mathbf{D}_x = [D_x(y): y \in N]$

❑ Node x also maintains its neighbors' distance vectors

▪ For each neighbor v, x maintains
$\mathbf{D}_v = [D_v(y): y \in N]$

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* $d_x(y)$
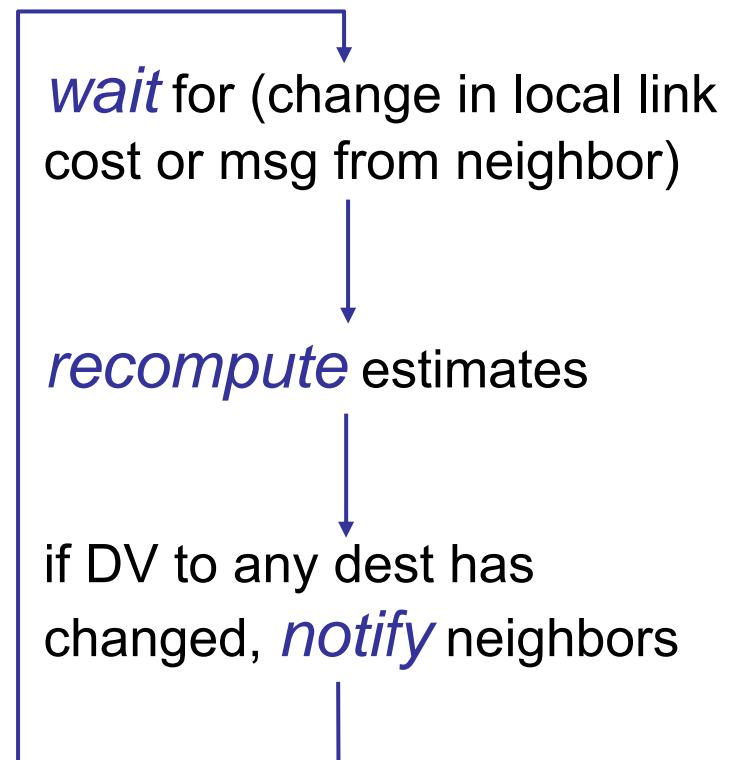
Distance Vector Algorithm (5)

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

node x table

cost to

	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from

cost to

	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

node y table

cost to

	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from

node z table

cost to

	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from

time

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0, 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1, 7+0\} = 3$

node x table

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

cost to

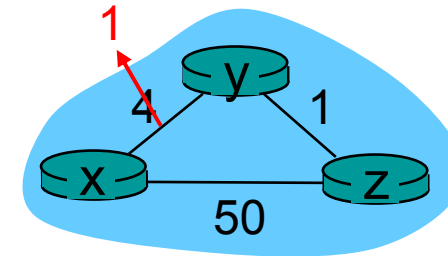| from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

time

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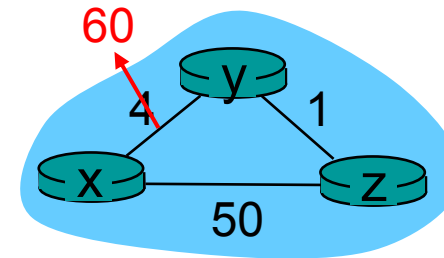
At time t_2, y receives z's update and updates its distance table. y's least costs do not change and hence y does *not* send any message to z.

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?

60

y

4   1

x   z

50

Comparison of LS and DV algorithms

Message complexity

- <u>LS:</u> with n nodes, E links, O(nE) msgs sent
- <u>DV:</u> exchange between neighbors only
  - convergence time varies

Speed of Convergence

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

Robustness: what happens if router malfunctions?

<u>LS:</u>
- node can advertise incorrect *link* cost
- each node computes only its *own* table

<u>DV:</u>
- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol

- Datagram format
- IPv4 addressing
- ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

❑ NAT

Part 3

❑ **Routing algorithms**

- Link state
- Distance Vector
- **Hierarchical routing**

❑ Routing in the Internet

- RIP
- OSPF
- BGP

❑ Broadcast and multicast routing

Hierarchical Routing

Our routing study thus far - idealization

- ❑ all routers identical
- ❑ network "flat"

… not true in practice

scale: with 200 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

- Direct link to router in another AS

Interconnected ASes

AS3

3c
3a
3b

1c
1a
1d
1b
AS1

2a
2c
2b
AS2

Intra-AS
Routing
algorithm

Inter-AS
Routing
algorithm
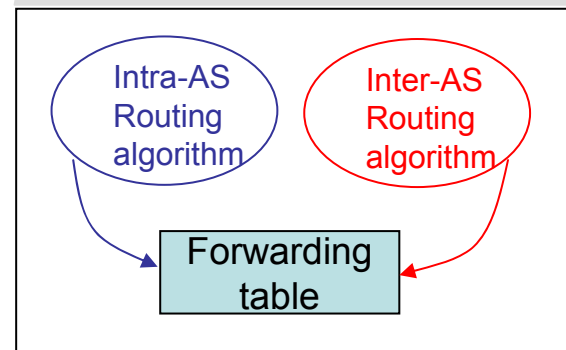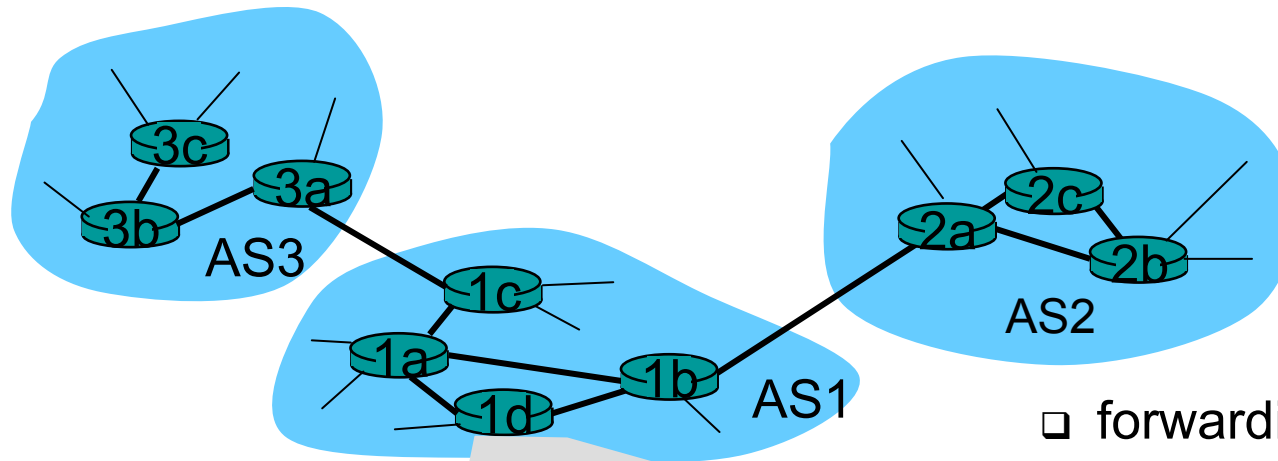
Forwarding
table

❑ 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

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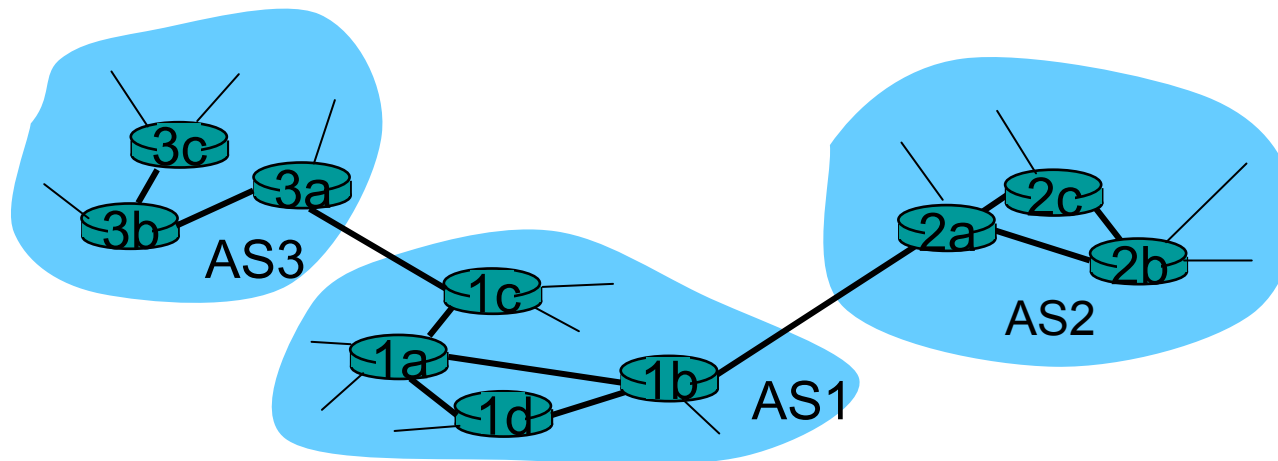
- ❑ suppose AS1 learns (via inter-AS protocol) that subnet *x* reachable via AS3 (gateway 1c) but not via AS2.
- ❑ inter-AS protocol propagates reachability info to all internal routers.
- ❑ router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c.
  - ▪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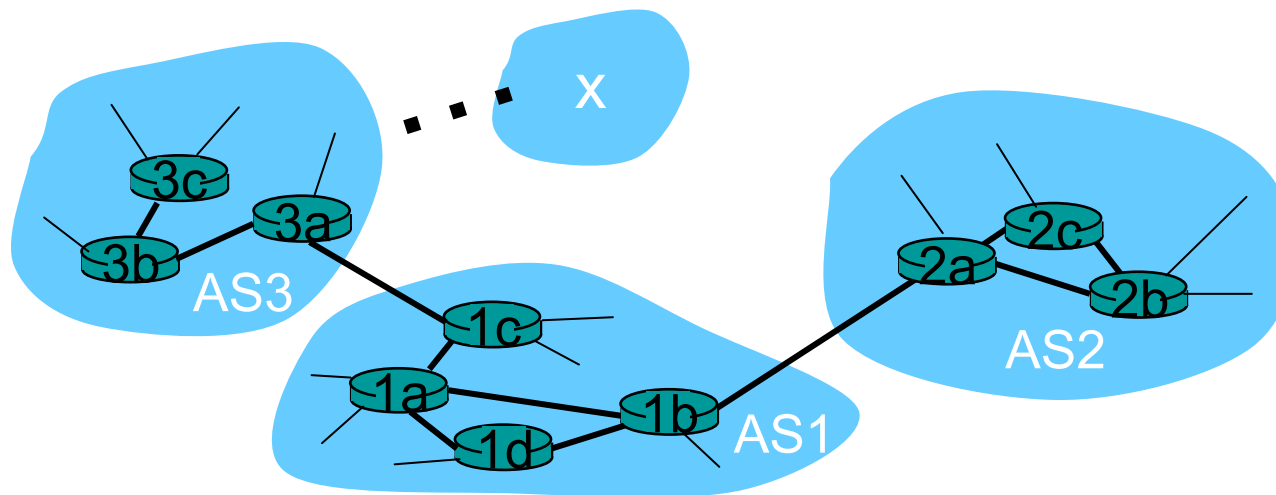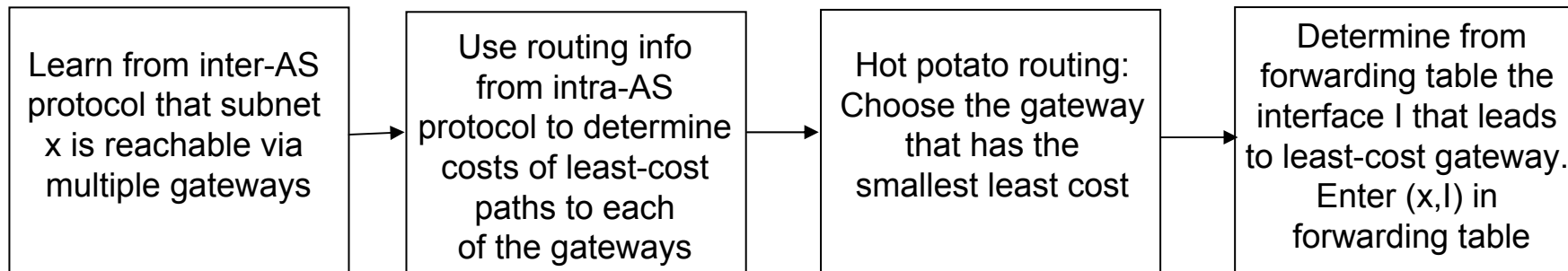
Example: Choosing among multiple ASes

❑ 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 towards which gateway it should forward packets for dest *x*.

- ▪ this is also job of inter-AS routing protocol!
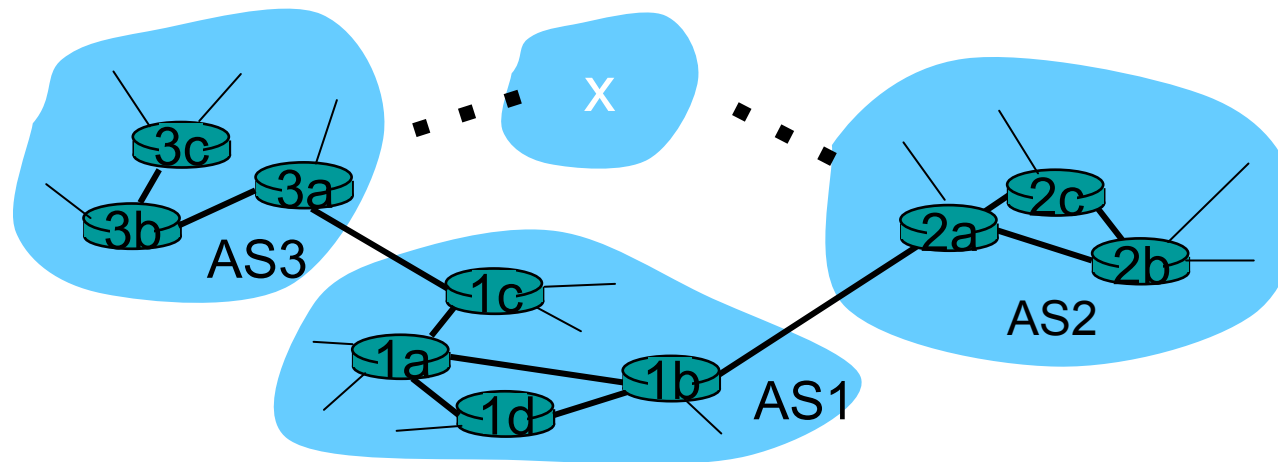
Example: Choosing among multiple ASes

❑ 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 towards which gateway it should forward packets for dest x.

  ▪ this is also job of inter-AS routing protocol!

❑ hot potato routing: send packet towards closest of two routers.

Learn from inter-AS protocol that subnet x is reachable via multiple gateways	Use routing info from intra-AS protocol to determine costs of least-cost paths to each of the gateways	Hot potato routing: Choose the gateway that has the smallest least cost	Determine from forwarding table the interface I that leads to least-cost gateway. Enter (x,I) in forwarding table

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- Virtual circuit and datagram networks
- What's inside a router
- NAT

Part 3

- **Routing algorithms**
  - Link state
  - Distance Vector
  - **Hierarchical routing**
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 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)

Chapter 4: Network Layer

Part 1

❏ Introduction

❏ IP: Internet Protocol

- Datagram format
- IPv4 addressing
- ICMP

Part 2

❏ IPv6

❏ Virtual circuit and datagram networks

❏ What's inside a router

❏ NAT

Part 3

❏ Routing algorithms

- Link state
- Distance Vector
- Hierarchical routing

❏ **Routing in the Internet**

- **RIP**
- OSPF
- BGP

❏ Broadcast and multicast routing

RIP (Routing Information Protocol)

- distance vector algorithm
- included in BSD-UNIX Distribution in 1982
- distance metric: # of hops (max = 15 hops)
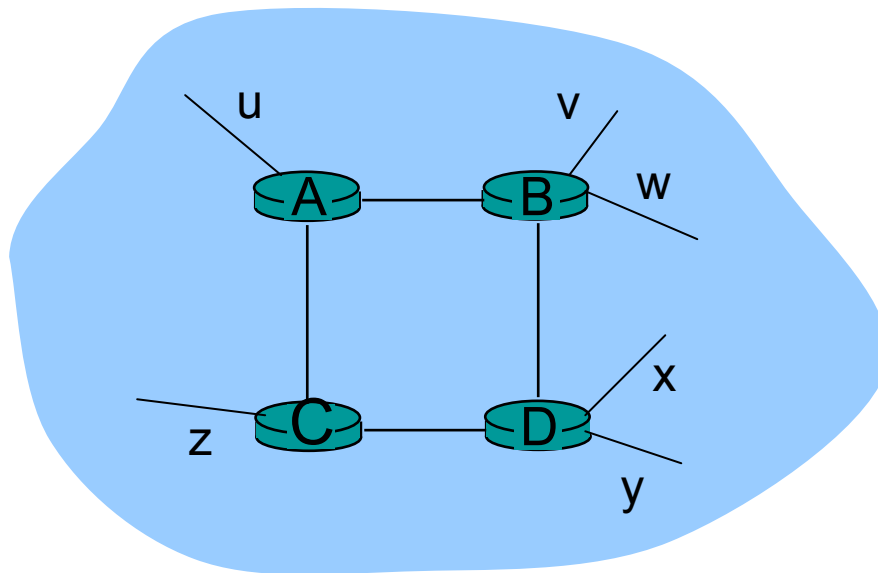
From router A to subnets:

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

RIP advertisements

❑ *distance vectors:* exchanged among neighbors every 30 sec via Response Message (also called advertisement)

❑ each advertisement: list of up to 25 destination subnets within AS

w   A   x   D   B   y   z   C
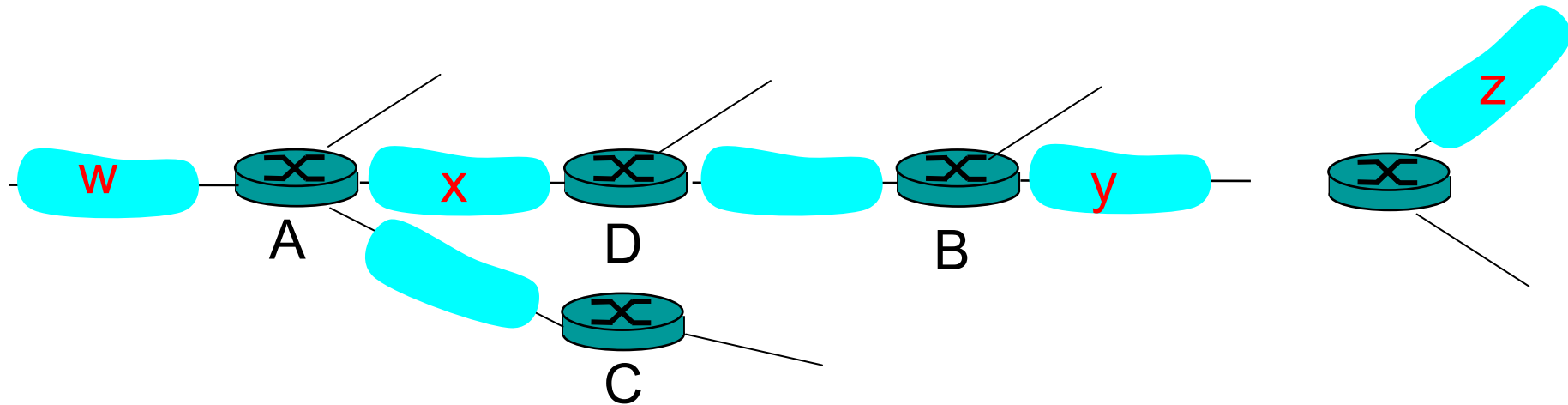
Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	B	7
x	--	1
....	....	....

Routing/Forwarding table in D

Dest	Next	hops
w	-	1
x	-	1
z	C	4
....	...	...

Advertisement from A to D
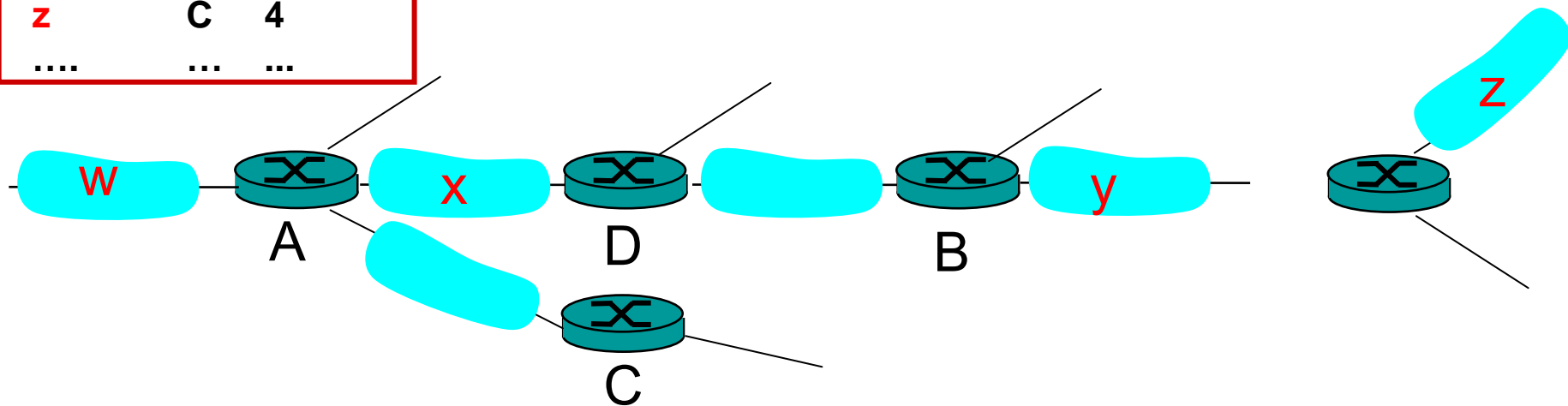
Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
....	....	....

Routing/Forwarding table in D

RIP: Link Failure and 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
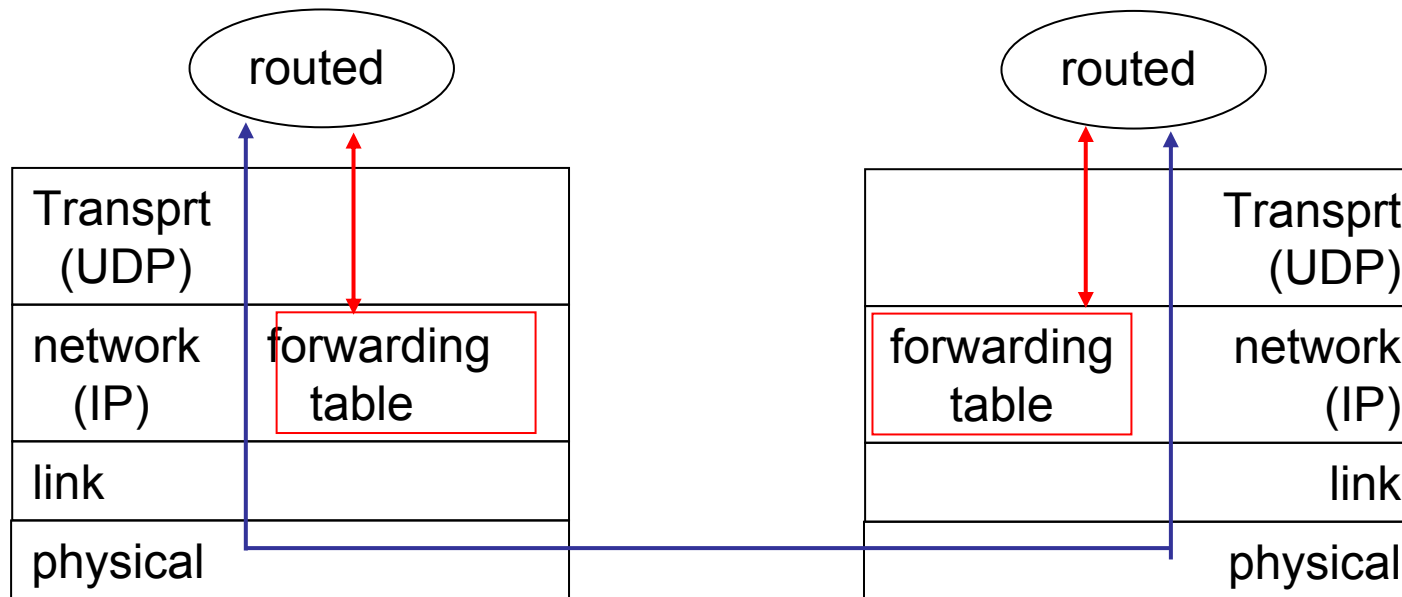
Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol

- Datagram format
- IPv4 addressing
- ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

❑ NAT

Part 3

❑ Routing algorithms

- Link state
- Distance Vector
- Hierarchical routing

❑ **Routing in the Internet**

- RIP
- **OSPF**
- BGP

❑ Broadcast and multicast routing

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 router
- ❑ advertisements disseminated to entire AS (via flooding)
  - ▪ carried in OSPF messages directly over IP (rather than TCP or UDP

OSPF "advanced" features (not in RIP)

- **security:** all OSPF messages authenticated (to prevent malicious intrusion)
- **multi**ple same-cost **path**s 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; high for real time)
- integrated uni- and **multicast** support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- **hierarchical** OSPF in large domains.

boundary router
backbone router
Backbone
area border routers
internal routers
Area 1
Area 2
Area 3
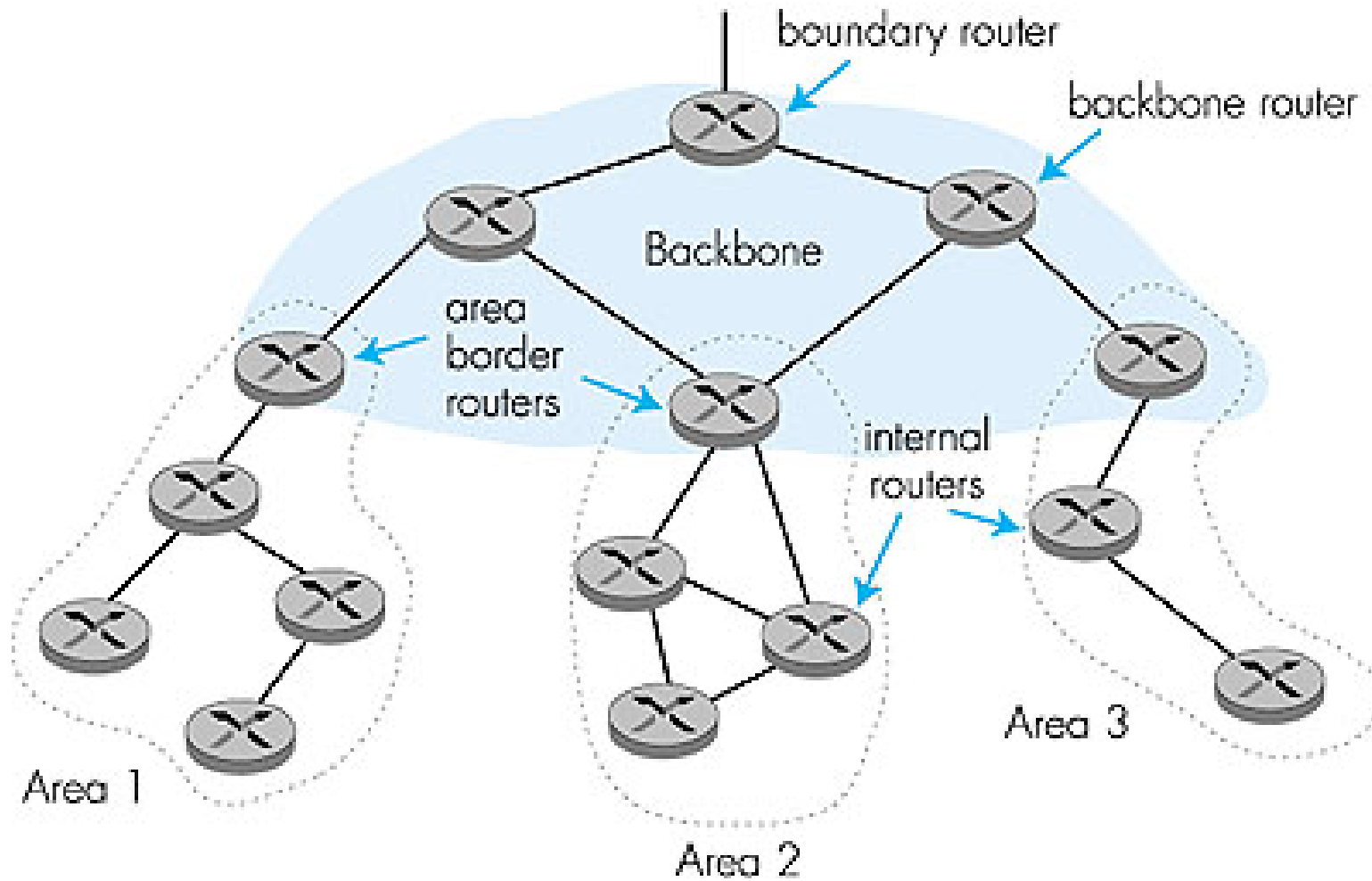
Hierarchical OSPF

❑ **two-level hierarchy:** local area, backbone.

- ▪ Link-state advertisements only in area

- ▪ each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.

❑ *area border routers:* "summarize" distances to nets in own area, advertise to other Area Border routers.

❑ *backbone routers:* run OSPF routing limited to backbone.

❑ *boundary routers:* connect to other AS's.

Chapter 4: Network Layer

Part 1

❑ Introduction

❑ IP: Internet Protocol

  ▪ Datagram format

  ▪ IPv4 addressing

  ▪ ICMP

Part 2

❑ IPv6

❑ Virtual circuit and datagram networks

❑ What's inside a router

❑ NAT

Part 3

❑ Routing algorithms

  ▪ Link state

  ▪ Distance Vector

  ▪ Hierarchical routing

❑ **Routing in the Internet**

  ▪ RIP

  ▪ OSPF

  ▪ **BGP**

❑ Broadcast and multicast routing

Internet inter-AS routing: BGP

❑ **BGP (Border Gateway Protocol):** *the* de facto standard

❑ BGP provides each AS a means to:

1. Obtain subnet reachability information from neighboring ASs.

2. Propagate reachability information to all AS-internal routers.

3. Determine "good" routes to subnets based on reachability information and policy.

❑ allows subnet to advertise its existence to rest of Internet: *"I am here"*

BGP basics

- pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections: BGP sessions
    - BGP sessions need not correspond to physical links.
- when AS2 advertises a prefix to AS1:
    - AS2 *promises* it will forward datagrams towards that prefix.
    - AS2 can aggregate prefixes in its advertisement
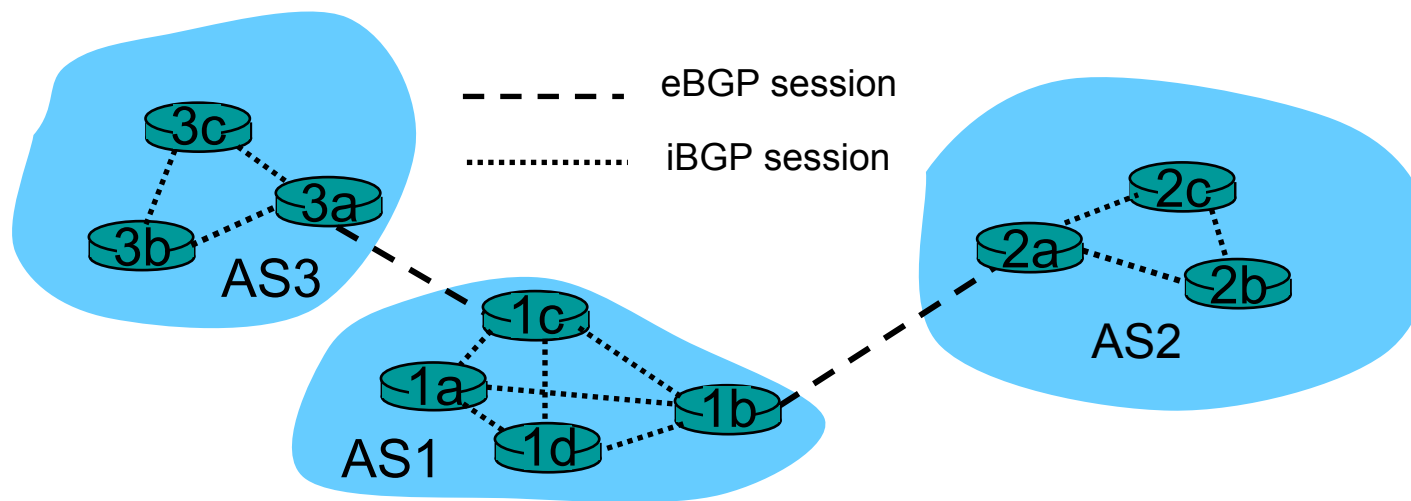
Distributing reachability info

- using eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
  - 1c can then use iBGP do distribute new prefix info to all routers in AS1
  - 1b can then re-advertise new reachability info to AS2 over 1b-to-2a eBGP session
- when router learns of new prefix, it creates entry for prefix in its forwarding table.
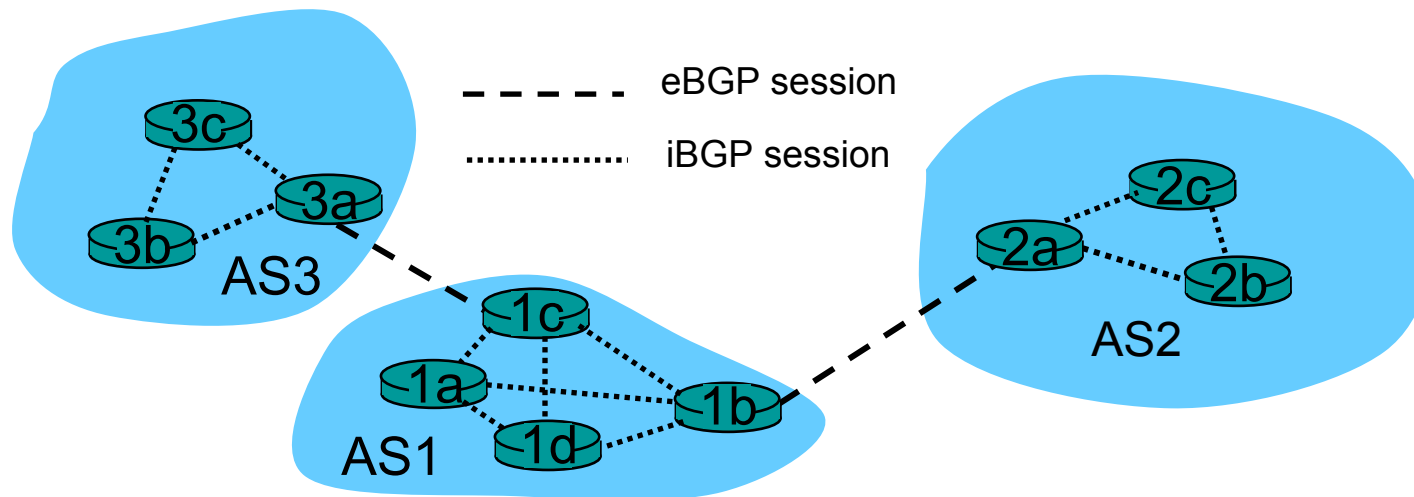
Path attributes & BGP routes

- advertised prefix includes BGP attributes.
  - prefix + attributes = "route"
- two important attributes:
  - **AS-PATH:** contains ASs through which prefix advertisement has passed: e.g, AS 67, AS 17
  - **NEXT-HOP:** indicates specific internal-AS router to next-hop AS. (may be multiple links from current AS to next-hop-AS)
- when gateway router receives route advertisement, uses import policy to accept/decline.

BGP route selection

❑   router may learn about more than 1 route to some prefix. Router must select route.

❑   elimination rules:

1. local preference value attribute: policy decision
2. shortest AS-PATH
3. closest NEXT-HOP router: hot potato routing
4. additional criteria

BGP messages

- BGP messages exchanged using TCP.
- BGP messages:
  - **OPEN:** opens TCP connection to peer and authenticates sender
  - **UPDATE:** advertises new path (or withdraws old)
  - **KEEPALIVE** keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - **NOTIFICATION:** reports errors in previous msg; also used to close connection

legend: provider network
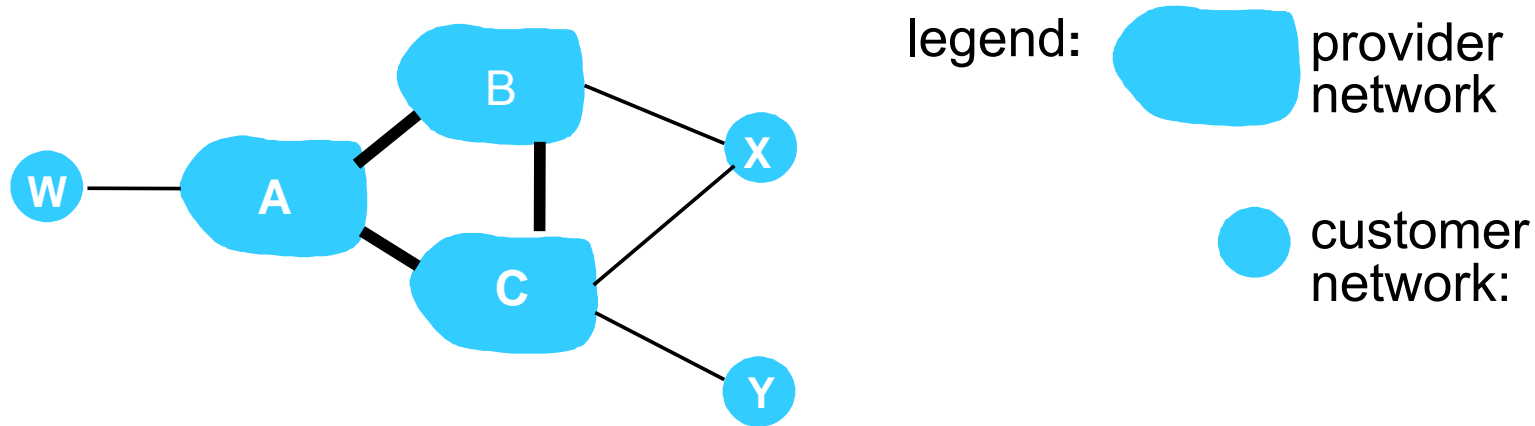
customer network:

- A,B,C are provider networks
- X,W,Y are customer (of provider networks)
- X is dual-homed: attached to two networks
  - X does not want to route from B via X to C
  - .. so X will not advertise to B a route to C

BGP routing policy (2)

legend:

provider network

customer network:
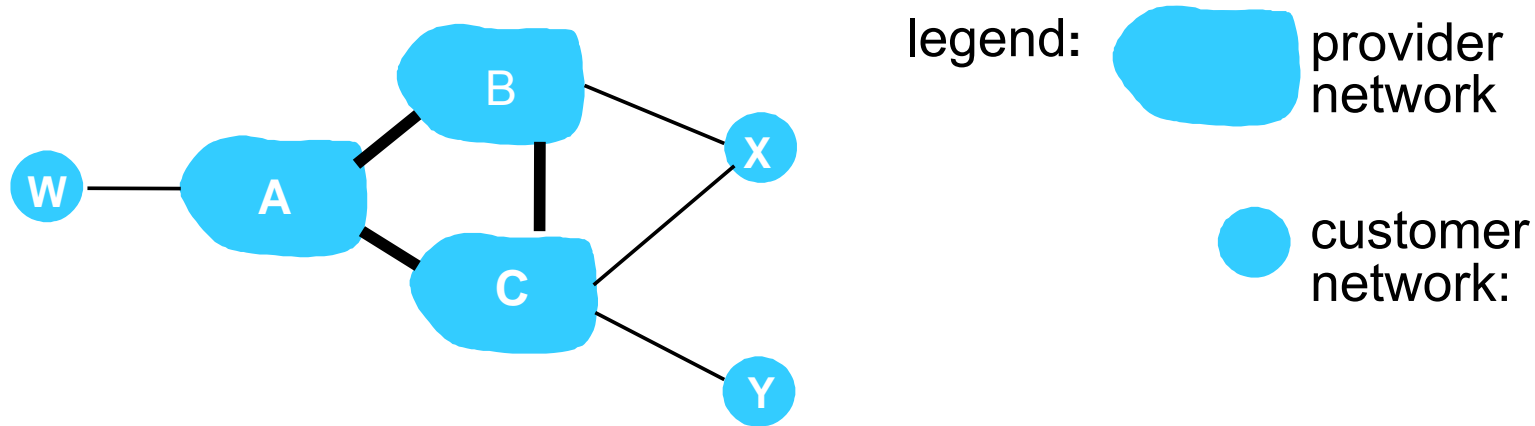
- ❏ A advertises path AW to B

- ❏ B advertises path BAW to X

- ❏ Should B advertise path BAW to C?

  - ▪ No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers

  - ▪ B wants to force C to route to w via A

  - ▪ B wants to route *only* to/from its customers!

Why different Intra- and Inter-AS routing?

Policy:

❑ Inter-AS: admin wants control over how its traffic routed, who routes through its net.

❑ Intra-AS: single admin, so no policy decisions needed

Scale:

❑ hierarchical routing saves table size, reduced update traffic

Performance:

❑ Intra-AS: can focus on performance

❑ Inter-AS: policy may dominate over performance

Chapter 4: Network Layer

Part 1

- ❑ Introduction
- ❑ IP: Internet Protocol
  - ▪ Datagram format
  - ▪ IPv4 addressing
  - ▪ ICMP

Part 2

- ❑ IPv6
- ❑ Virtual circuit and datagram networks
- ❑ What's inside a router
- ❑ NAT

Part 3

- ❑ Routing algorithms
  - ▪ Link state
  - ▪ Distance Vector
  - ▪ Hierarchical routing
- ❑ Routing in the Internet
  - ▪ RIP
  - ▪ OSPF
  - ▪ BGP
- ❑ **Broadcast and multicast routing**

Broadcast Routing

- deliver packets from source to all other nodes
- source duplication is inefficient:

source
duplication

in-network
duplication
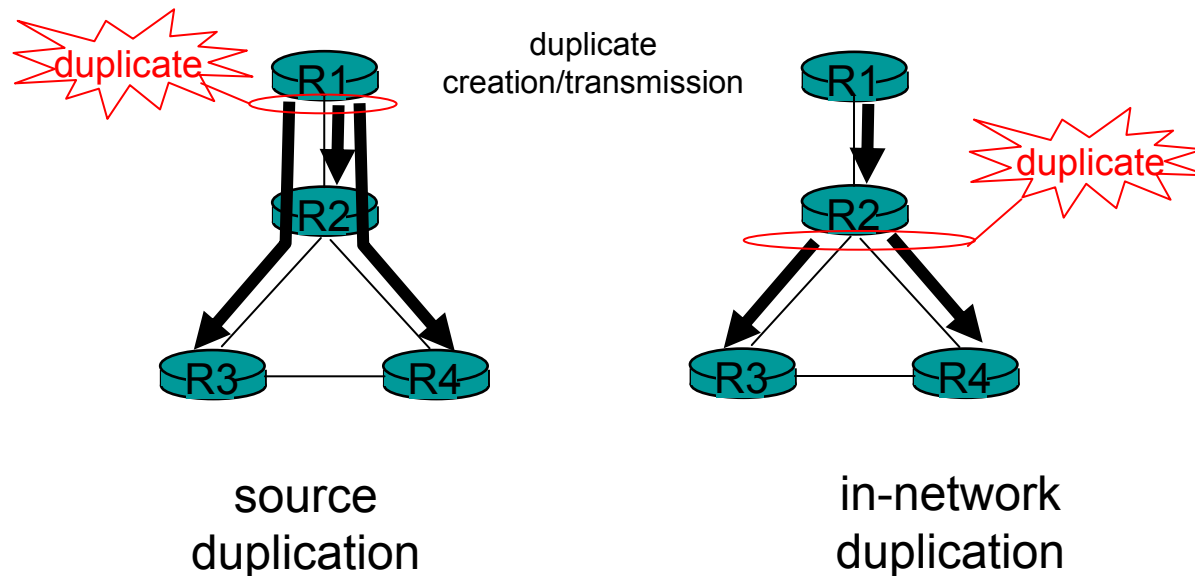
- source duplication: how does source determine recipient addresses?

In-network duplication

- flooding: when node receives brdcst pckt, sends copy to all neighbors
  - Problems: cycles & broadcast storm
- controlled flooding: node only brdcsts pkt if it hasn't brdcst same packet before
  - Node keeps track of pckt ids already brdcsted
  - Or reverse path forwarding (RPF): only forward pckt if it arrived on shortest path between node and source
- spanning tree
  - No redundant packets received by any node

Spanning Tree

□ First construct a spanning tree

□ Nodes forward copies only along spanning tree
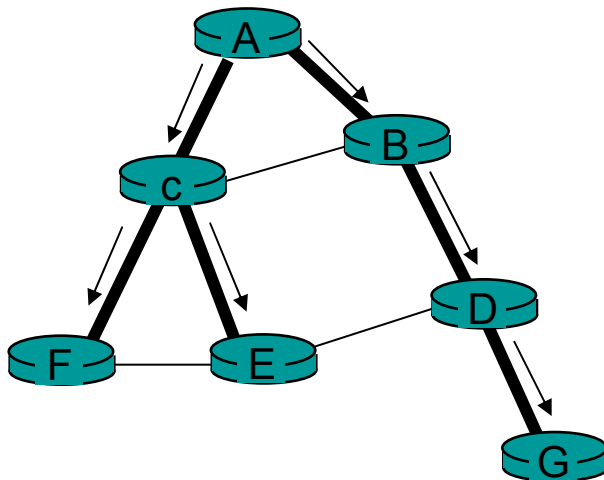
(a) Broadcast initiated at A
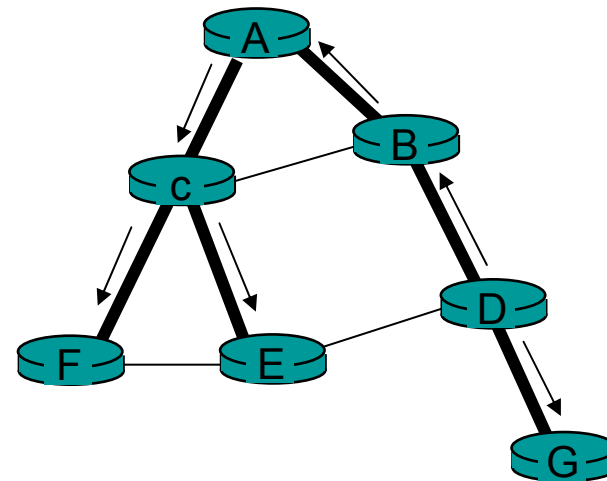
(b) Broadcast initiated at D

Spanning Tree: Creation

- ❑ Center node
- ❑ Each node sends unicast join message to center node
  - ▪ Message forwarded until it arrives at a node already belonging to spanning tree
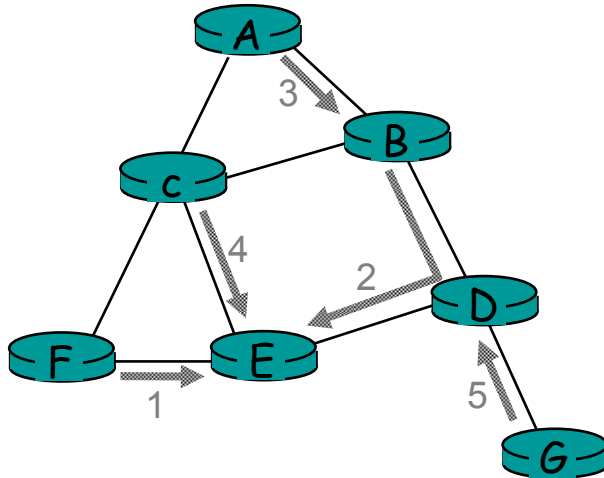
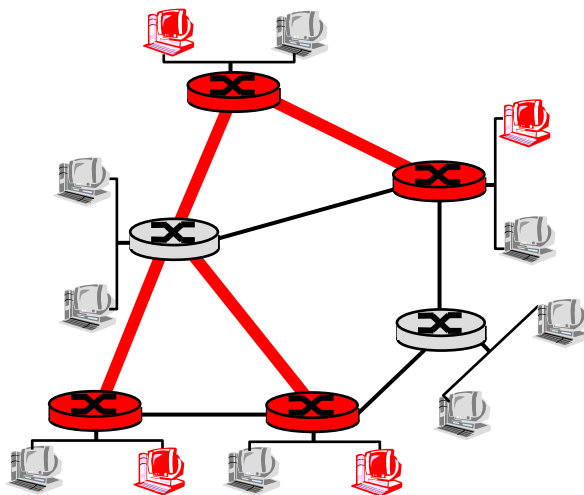(a) Stepwise construction of spanning tree
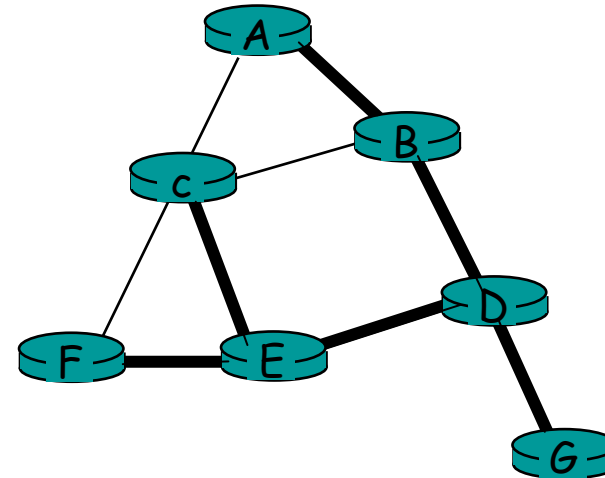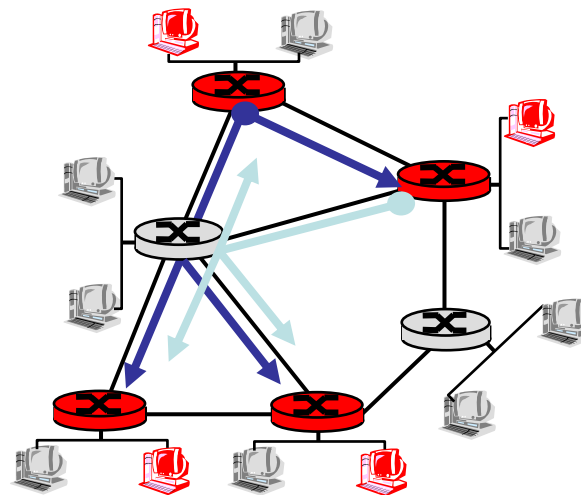
(b) Constructed spanning tree

Multicast Routing: Problem Statement

❑ ***Goal:*** find a tree (or trees) connecting routers having local mcast group members

- ▪ *tree:* not all paths between routers used
- ▪ *source-based:* different tree from each sender to rcvrs
- ▪ *shared-tree:* same tree used by all group members

Shared tree   Source-based trees

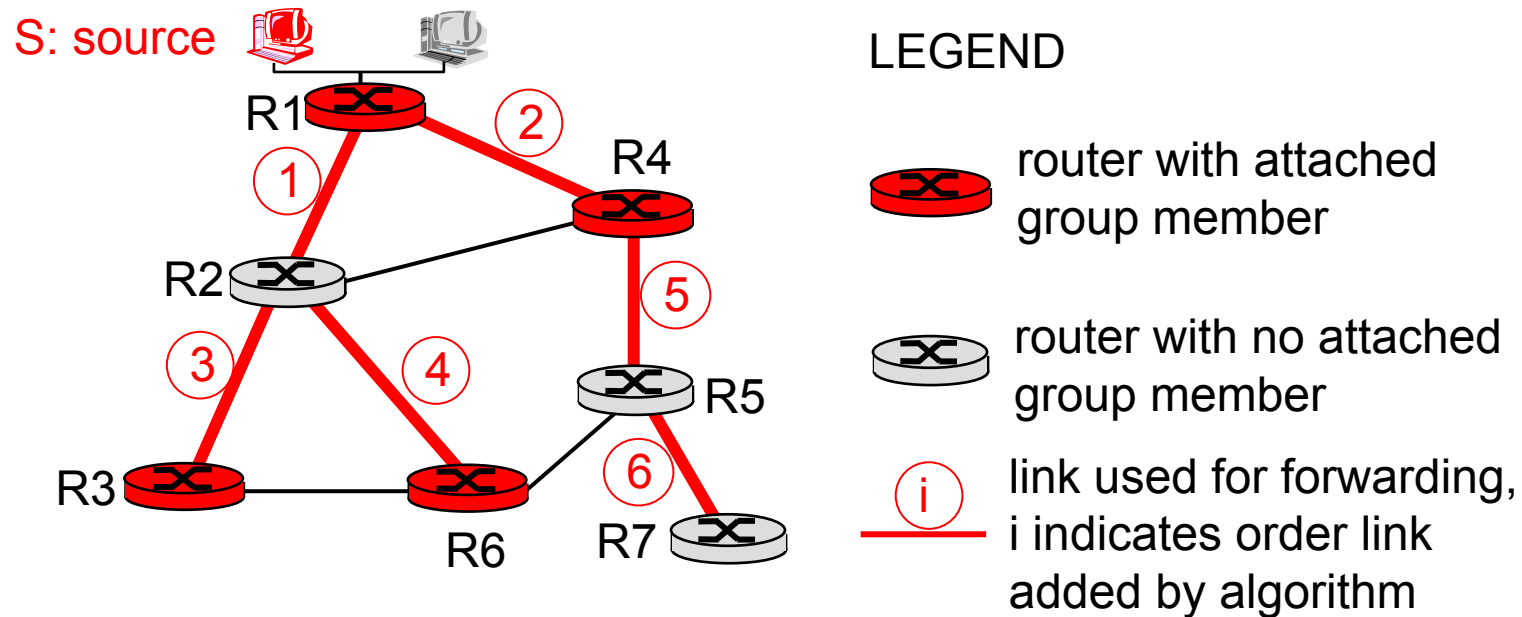# Approaches for building mcast trees

Approaches:

❑ source-based tree: one tree per source

- shortest path trees
- reverse path forwarding

❑ group-shared tree: group uses one tree

- minimal spanning (Steiner)
- center-based trees

…we first look at basic approaches, then specific protocols adopting these approaches

Shortest Path Tree

❑ mcast forwarding tree: tree of shortest path routes from source to all receivers
  ▪ Dijkstra's algorithm

S: source

LEGEND

R1
R2
R3
R4
R5
R6
R7

1
2
3
4
5
6

router with attached group member

router with no attached group member

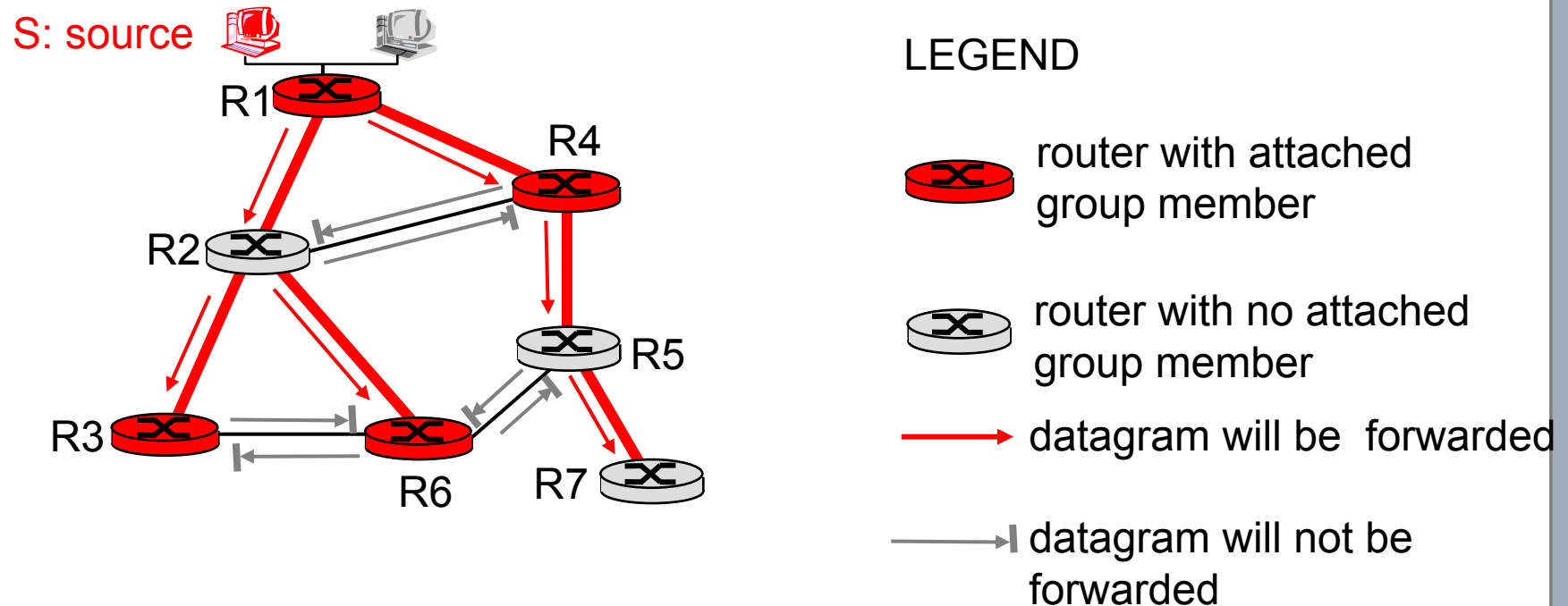(i) link used for forwarding, i indicates order link added by algorithm

Reverse Path Forwarding

❑ rely on router's knowledge of unicast shortest path from it to sender

❑ each router has simple forwarding behavior:

if (mcast datagram received on incoming link on shortest path back to center)

    then flood datagram onto all outgoing links

    else ignore datagram

Reverse Path Forwarding: example

S: source

R1

R4

R2

R3

R5

R6

R7

LEGEND

router with attached group member

router with no attached group member

→ datagram will be forwarded

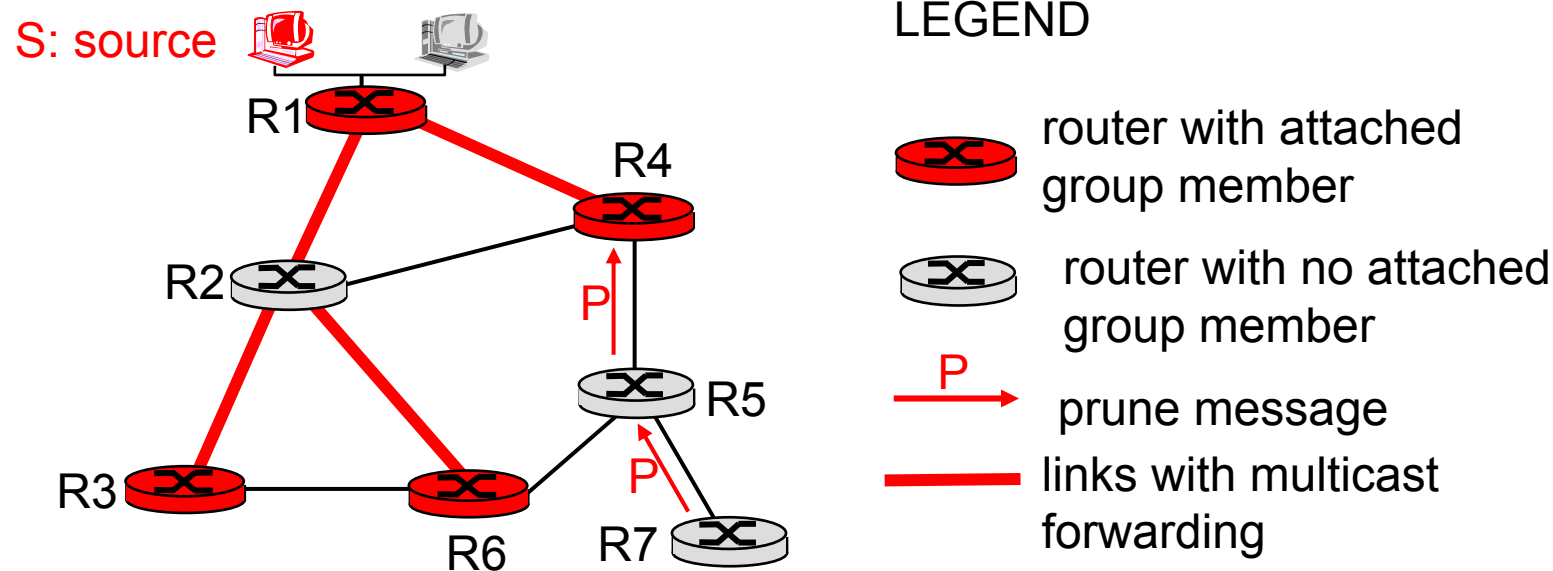→| datagram will not be forwarded

- result is a source-specific *reverse* SPT
  - may be a bad choice with asymmetric links

Reverse Path Forwarding: pruning

- ❑ forwarding tree contains subtrees with no mcast group members
  - ▪ no need to forward datagrams down subtree
  - ▪ "prune" msgs sent upstream by router with no downstream group members

S: source

R1

R2

R3

R4

R5

R6

R7

P

P

LEGEND

router with attached group member

router with no attached group member

P → prune message

━━━ links with multicast forwarding

Shared-Tree: Steiner Tree

❑ Steiner Tree: minimum cost tree connecting all routers with attached group members

❑ problem is NP-complete

❑ excellent heuristics exists

❑ not used in practice:

   ▪ computational complexity

   ▪ information about entire network needed

   ▪ monolithic: rerun whenever a router needs to join/leave

Center-based trees

❑ single delivery tree shared by all

❑ one router identified as *"center"* of tree

❑ to join:

- edge router sends unicast *join-msg* addressed to center router

- *join-msg* "processed" by intermediate routers and forwarded towards center

- *join-msg* either hits existing tree branch for this center, or arrives at center

- path taken by *join-msg* becomes new branch of tree for this router

Suppose R6 chosen as center:
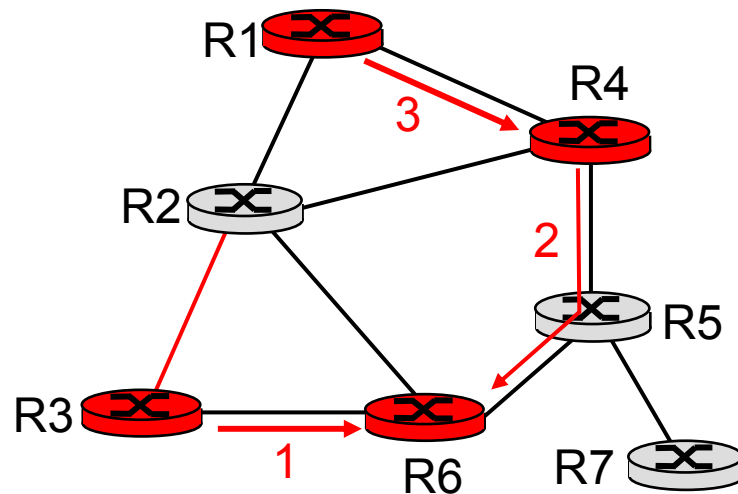
LEGEND

router with attached group member

router with no attached group member

1 → path order in which join messages generated

Internet Multicasting Routing: DVMRP

❑ DVMRP: distance vector multicast routing protocol, RFC1075

❑ *flood and prune:* reverse path forwarding, source-based tree

- ▪ RPF tree based on DVMRP's own routing tables constructed by communicating DVMRP routers
- ▪ no assumptions about underlying unicast
- ▪ initial datagram to mcast group flooded everywhere via RPF
- ▪ routers not wanting group: send upstream prune msgs

DVMRP: continued…

- *soft state:* DVMRP router periodically (1 min.) "forgets" branches are pruned:
    - mcast data again flows down unpruned branch
    - downstream router: reprune or else continue to receive data
- routers can quickly regraft to tree
    - following IGMP join at leaf
- odds and ends
    - commonly implemented in commercial routers
    - Mbone routing done using DVMRP
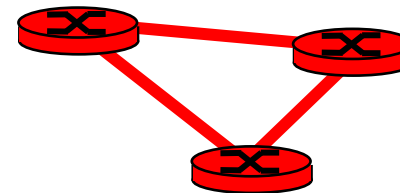
Q: How to connect "islands" of multicast routers in a "sea" of unicast routers?
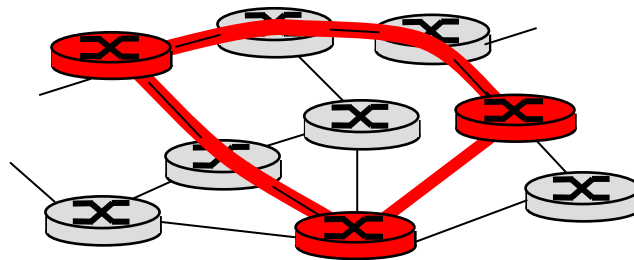
physical topology   logical topology

- ❏ mcast datagram encapsulated inside "normal" (non-multicast-addressed) datagram

- ❏ normal IP datagram sent thru "tunnel" via regular IP unicast to receiving mcast router

- ❏ receiving mcast router unencapsulates to get mcast datagram

PIM: Protocol Independent Multicast

❑ not dependent on any specific underlying unicast routing algorithm (works with all)

❑ two different multicast distribution scenarios :

Dense:

❑ group members densely packed, in "close" proximity.

❑ bandwidth more plentiful

Sparse:

❑ # networks with group members small wrt # interconnected networks

❑ group members "widely dispersed"

❑ bandwidth not plentiful

Consequences of Sparse-Dense Dichotomy:

Dense

- group membership by routers *assumed* until routers explicitly prune
- *data-driven* construction on mcast tree (e.g., RPF)
- bandwidth and non-group-router processing *profligate*

Sparse:

- no membership until routers explicitly join
- *receiver- driven* construction of mcast tree (e.g., center-based)
- bandwidth and non-group-router processing *conservative*

PIM- Dense Mode

flood-and-prune RPF, similar to DVMRP but

- ❑ underlying unicast protocol provides RPF info for incoming datagram

- ❑ less complicated (less efficient) downstream flood than DVMRP reduces reliance on underlying routing algorithm

- ❑ has protocol mechanism for router to detect it is a leaf-node router

PIM - Sparse Mode

- center-based approach
- router sends *join* msg to rendezvous point (RP)
  - intermediate routers update state and forward *join*
- after joining via RP, router can switch to source-specific tree
  - increased performance: less concentration, shorter paths

R1

R4

R2

join

join

R5

R3

join

R7

R6

all data multicast from rendezvous point
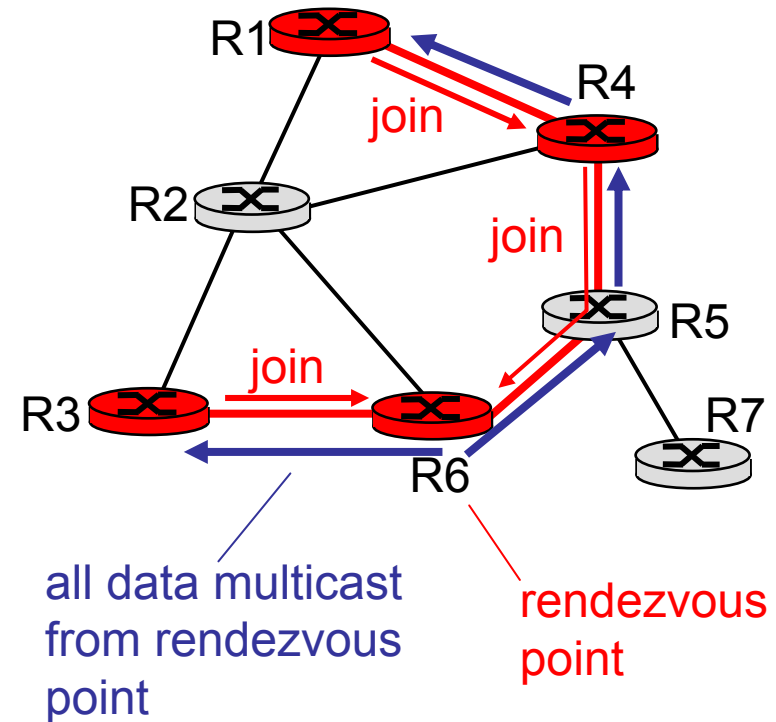
rendezvous point
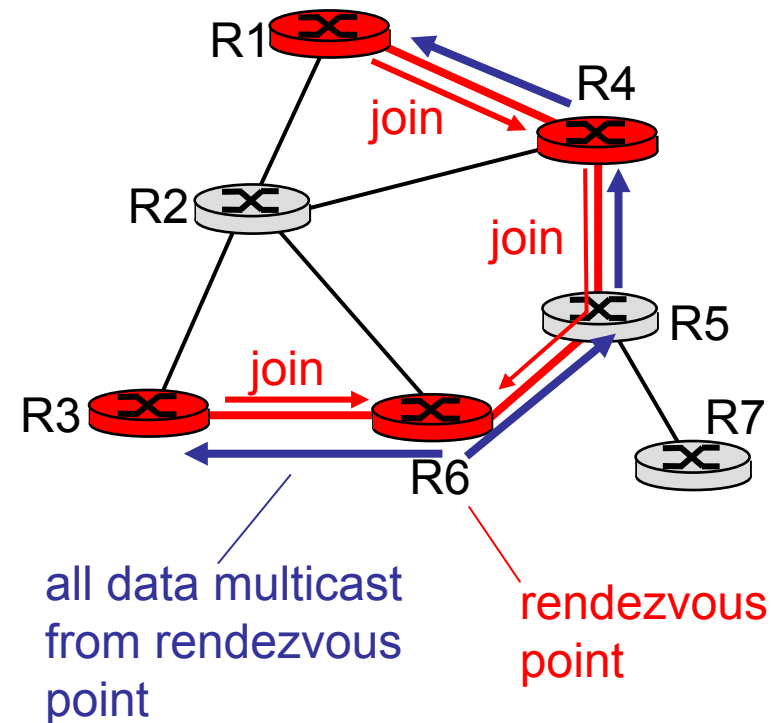
PIM - Sparse Mode

sender(s):

- ❑ unicast data to RP, which distributes down RP-rooted tree
- ❑ RP can extend mcast tree upstream to source
- ❑ RP can send *stop* msg if no attached receivers
  - ▪ "no one is listening!"

R1

R4

join

R2

join

R5

join

R3

join

R7

R6

all data multicast from rendezvous point

rendezvous point

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- Virtual circuit and datagram networks
- What's inside a router
- NAT

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP
- Broadcast and multicast routing