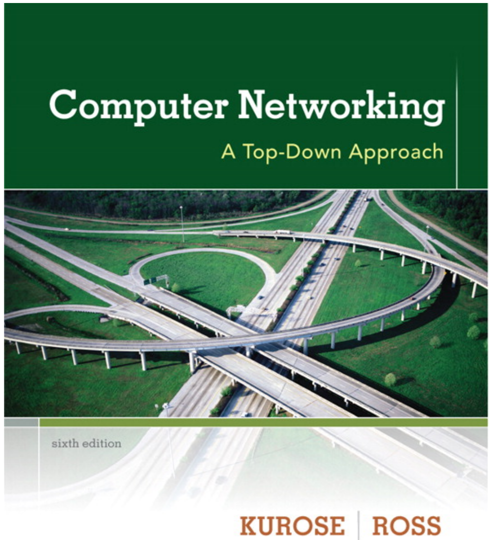
Computer Networking

谢 逸

中山大学・数据科学与计算机学院

2018. Spring

Chapter 4
Network Layer



Computer Networking: A Top Down Approach
6th edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012

A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Assignments (ver6，中文/英文):

- ch4: 1, 7, 9, 11, 13, 16, 19, 26, 30, 37, 39, 42, 52, 53

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)
  - broadcast, multicast
- instantiation, implementation in the Internet

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

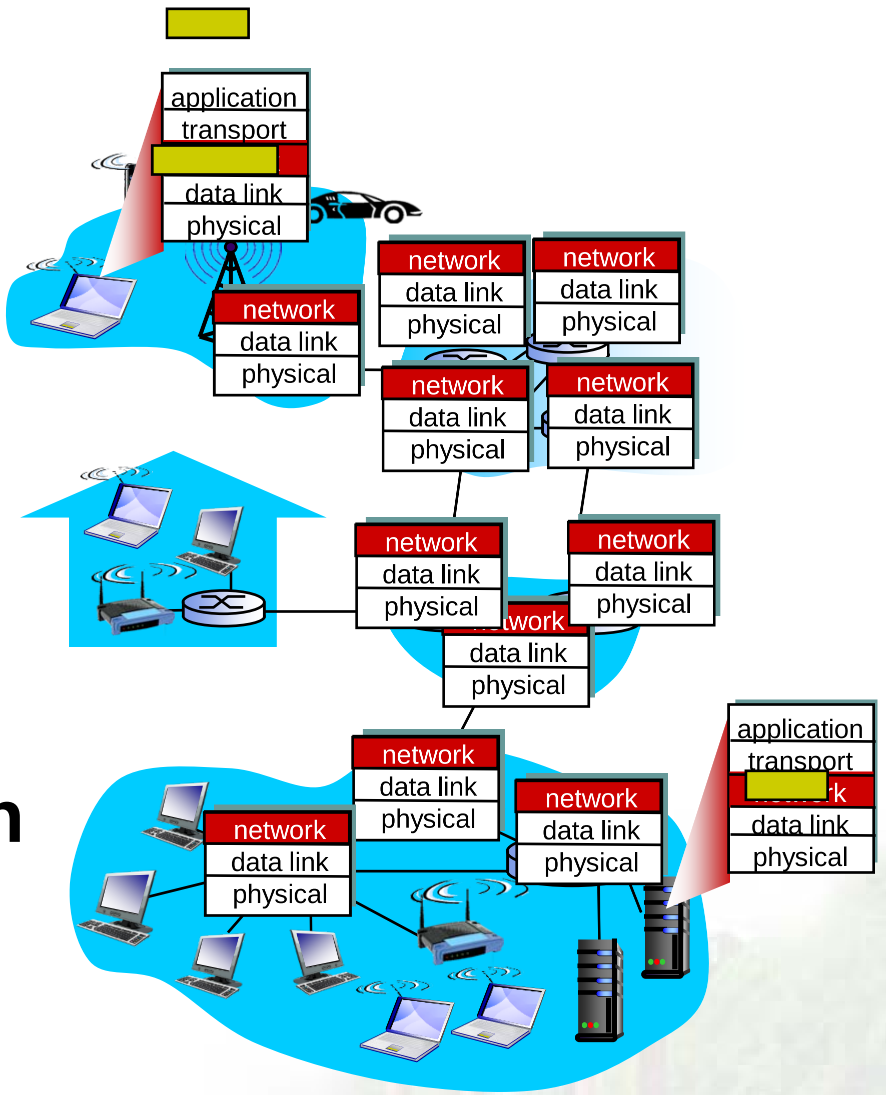
4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

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



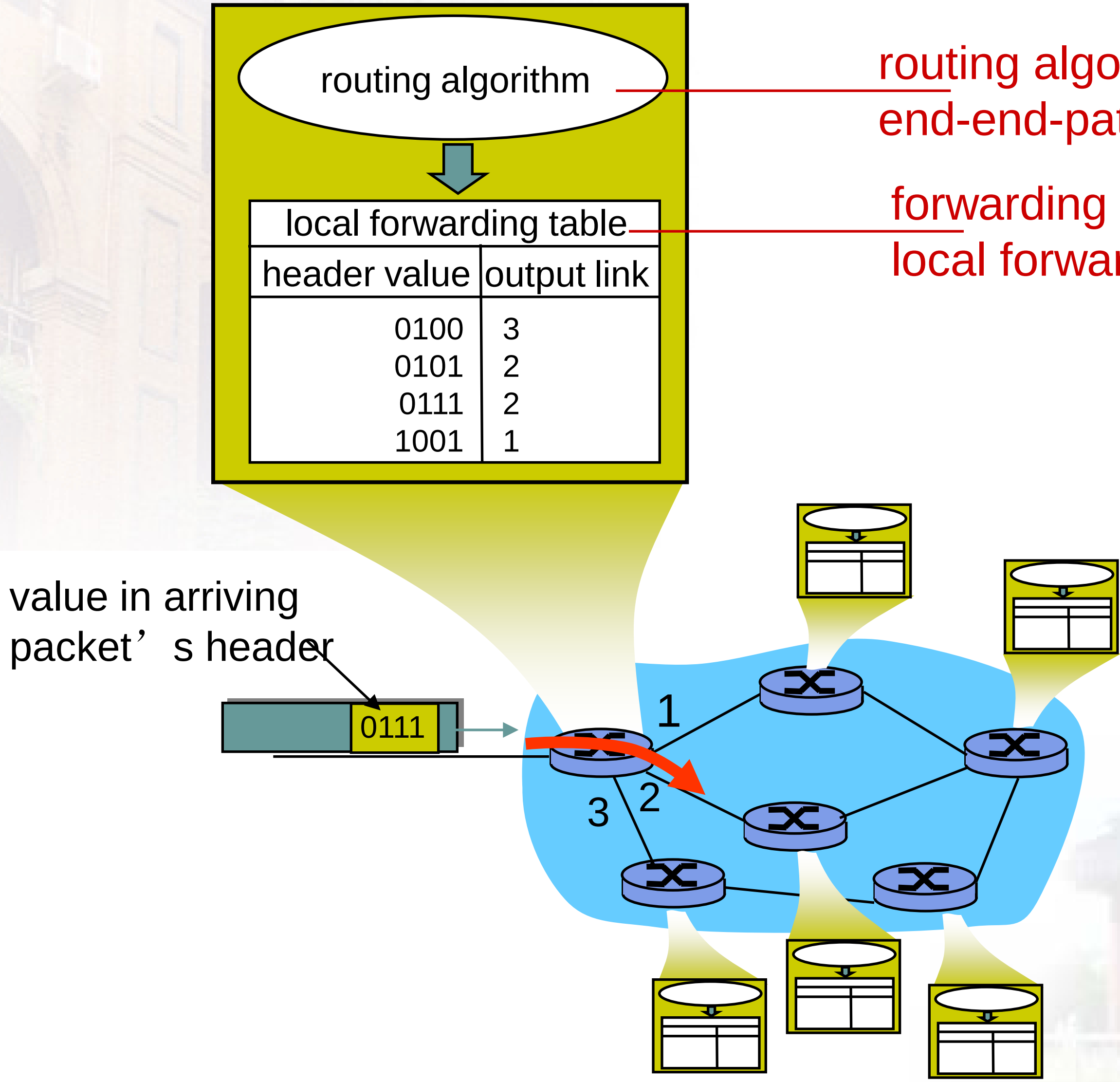
Two key network-layer functions

- ***routing:*** **determine route taken by packets from source to dest.**
    - ***routing algorithms***
- ***forwarding:*** **move packets from router' s input to appropriate router output**
- **Connection Setup**
    - **Not included in IP**

analogy:

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

Interplay between routing and forwarding

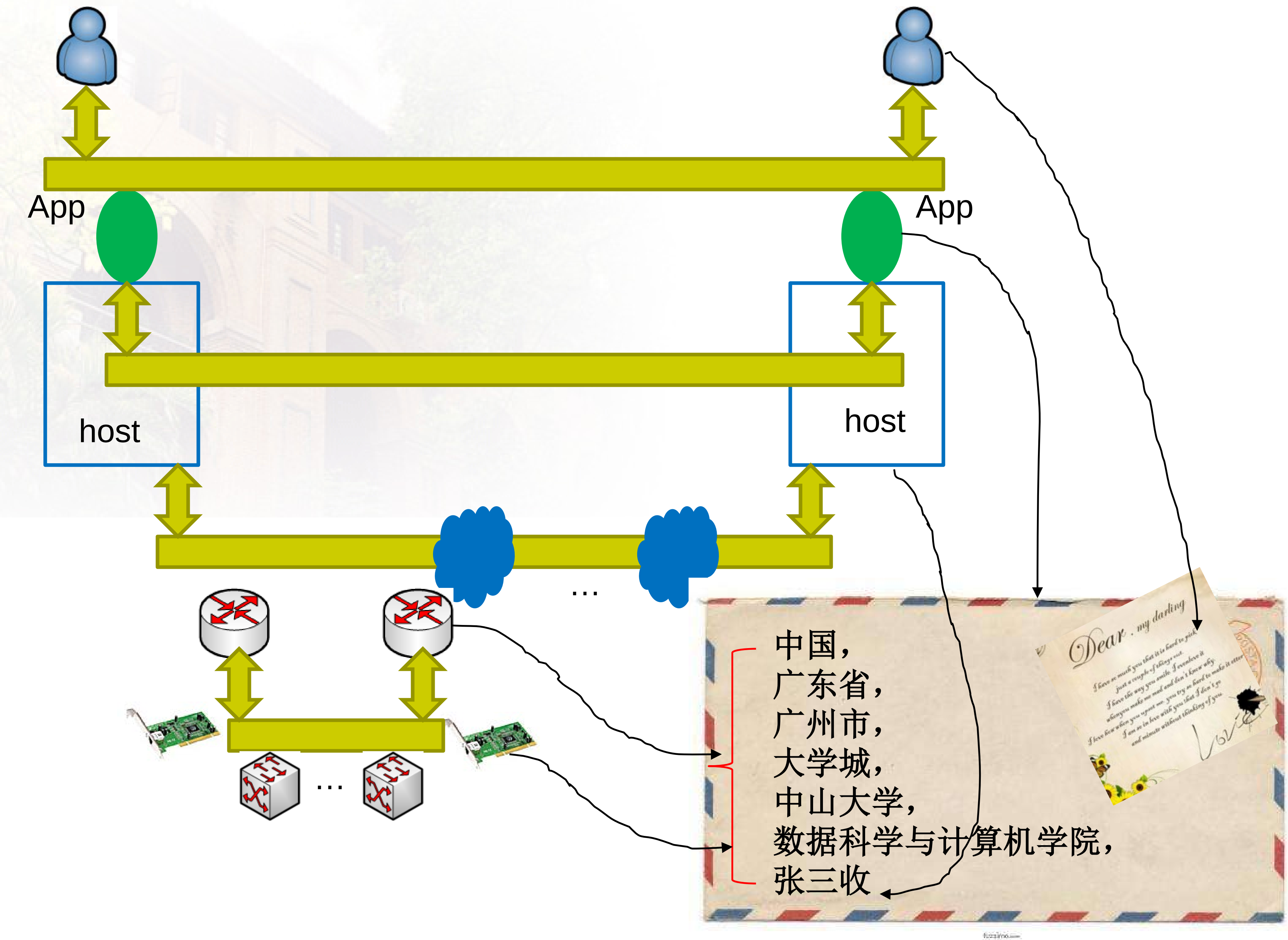


routing algorithm determines end-end-path through network

forwarding table determines local forwarding at this router

Connection setup

- 3rd important function in *some* network architectures:
  - ATM, frame relay, X.25
- before datagrams flow, two end hosts *and* intervening routers establish virtual connection
  - routers get involved
- network vs transport layer connection service:
  - *network:* between two hosts (may also involve intervening routers in case of VCs)
  - *transport:* between two processes



App
App
host
host
...
...
中国，
广东省，
广州市，
大学城，
中山大学，
数据科学与计算机学院，
张三收
Dear, my darling

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 ? Bandwidth	Loss	Order	Timing	Congestion feedback
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

UBR：Unspecified Bit Rate

CBR：Constant Bit Rate

VBR：Variable Bit Rate

ABR：Available Bit Rate

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

Connection, connection-less service

- ***datagram*** **network provides network-layer** ***connectionless*** **service**
- ***virtual-circuit*** **network provides network-layer** ***connection*** **service**
- **analogous to TCP/UDP connecton-oriented / connectionless 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)

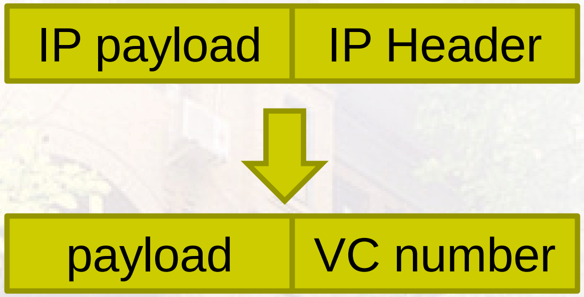
VC implementation

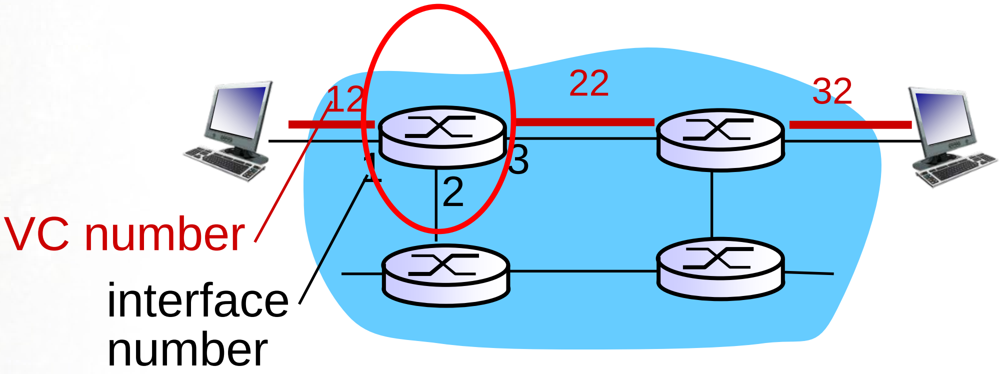
a VC consists of:

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**

VC forwarding table





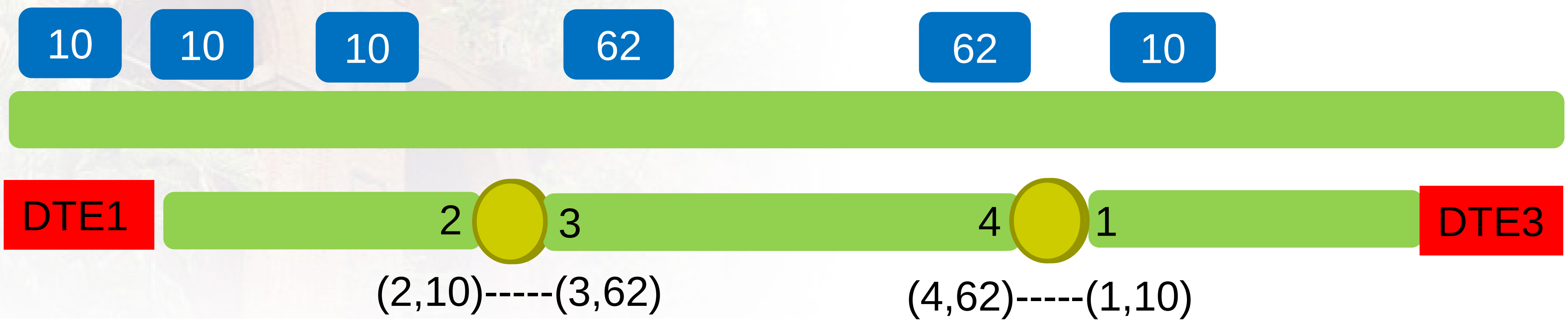
forwarding table in northwest router:

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

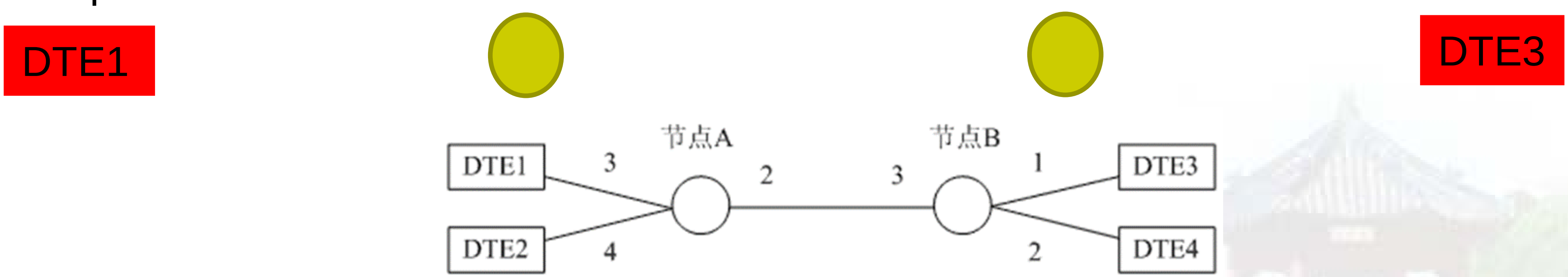
VC routers maintain connection state information!

Step1:

Step2:



Step3:



节点A路由表

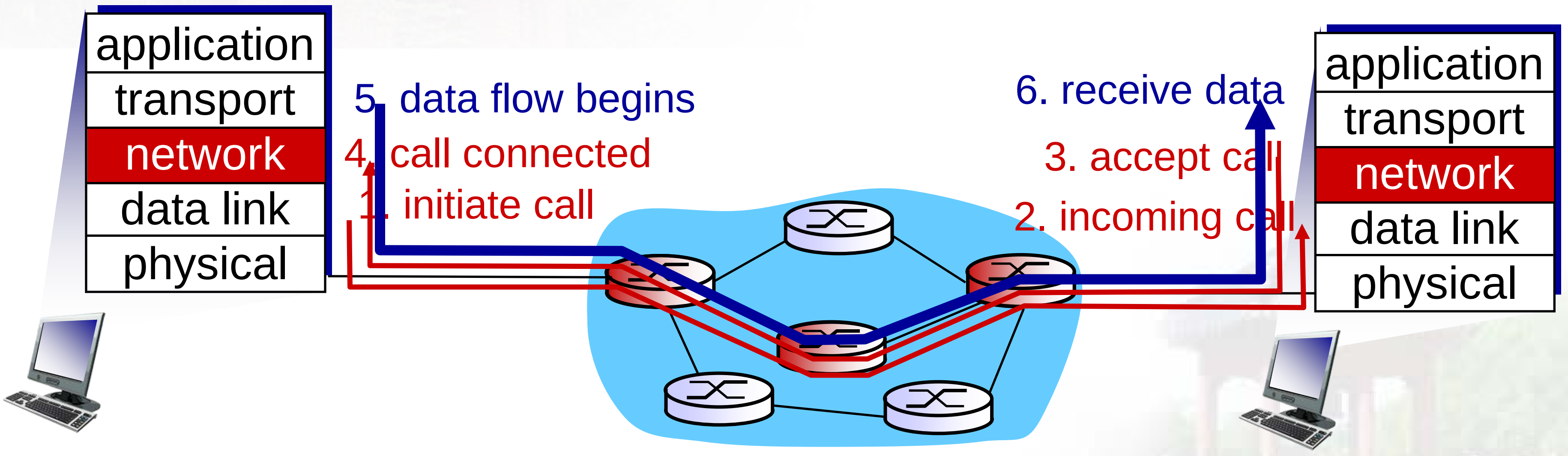
呼叫	入		出	
	端口号	LCN	端口号	LCN
1	3	10	2	62
2	4	12	2	63

节点B路由表

呼叫	入		出	
	端口号	LCN	端口号	LCN
1	3	62	1	22
2	3	63	2	23

Virtual circuits: signaling protocols

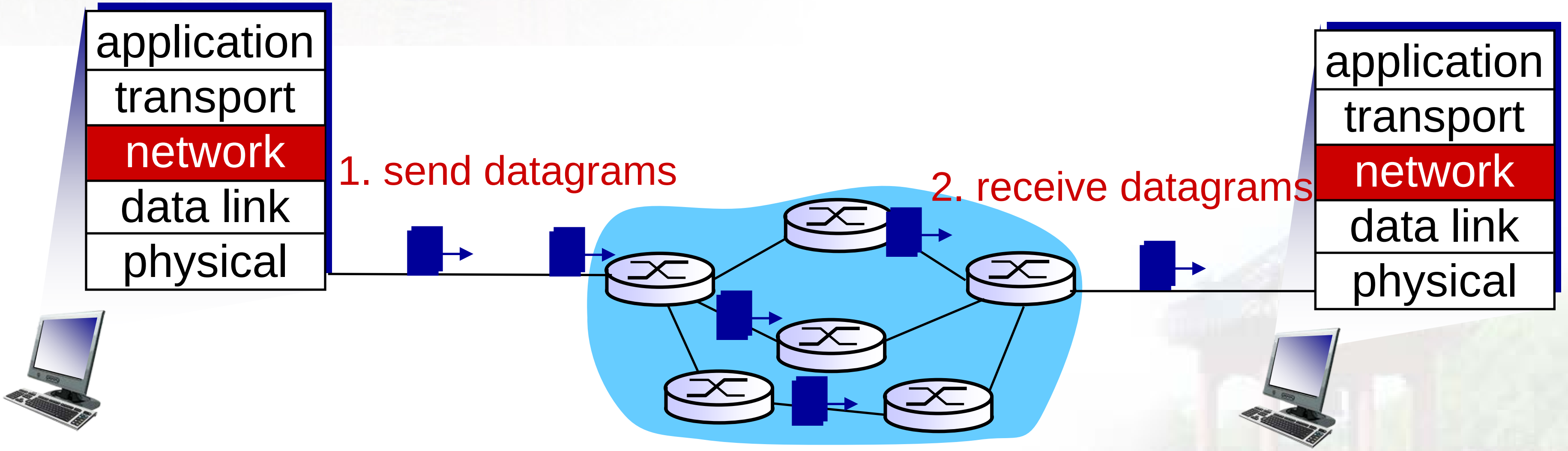
- **used to setup, maintain teardown VC**
- **used in ATM, frame-relay, X.25**
- **not used in today's Internet**



Create forwarding table for each router

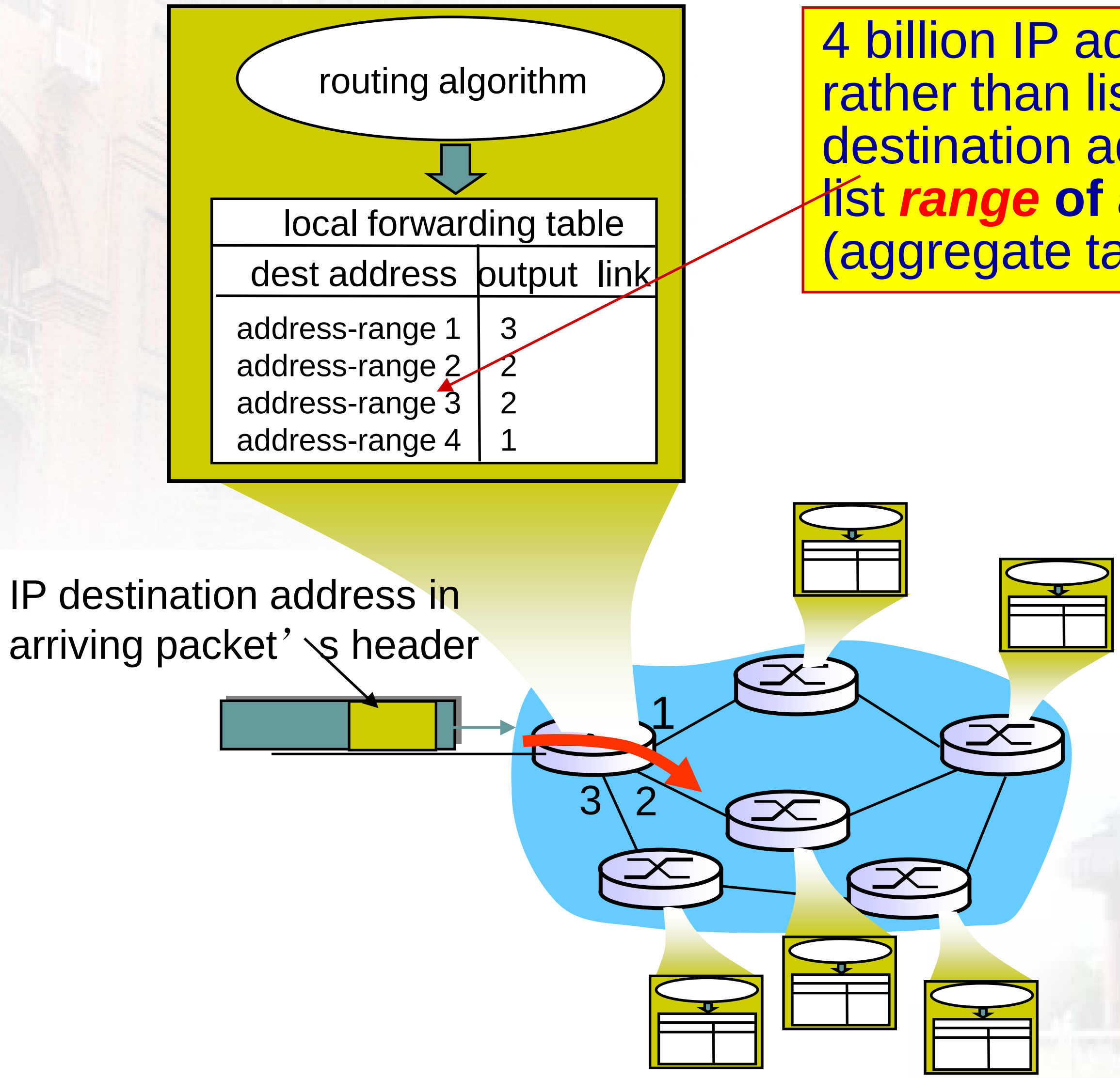
Datagram networks

- 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



Datagram forwarding table

4 billion IP addresses, so rather than list individual destination address list ***range* of addresses** (aggregate table entries)



Datagram forwarding table

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

Q: **but what happens if ranges don't divide up so nicely?**

Longest prefix matching

longest prefix matching

when looking for forwarding table entry for given destination address, use ***longest*** address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

examples:

DA: 11001000 00010111 00010110 10100001 — which interface?

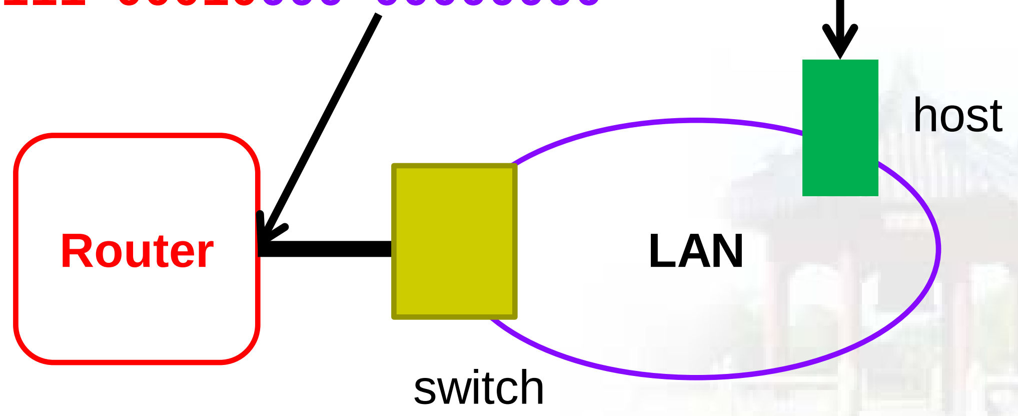
DA: 11001000 00010111 00011000 10101010 — which interface?

Destination Address Range	Link interface
11001000 00010111 00010000 00000000 through **11001000 00010111 00010111 11111111**	0

Destination Address Range	Link interface
11001000 00010111 00010* *********	0

Host: 11001000 00010111 00010010 01001001

router: 11001000 00010111 00010000 00000000



Datagram or VC network: why?

Internet (datagram)

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

ATM (VC)

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

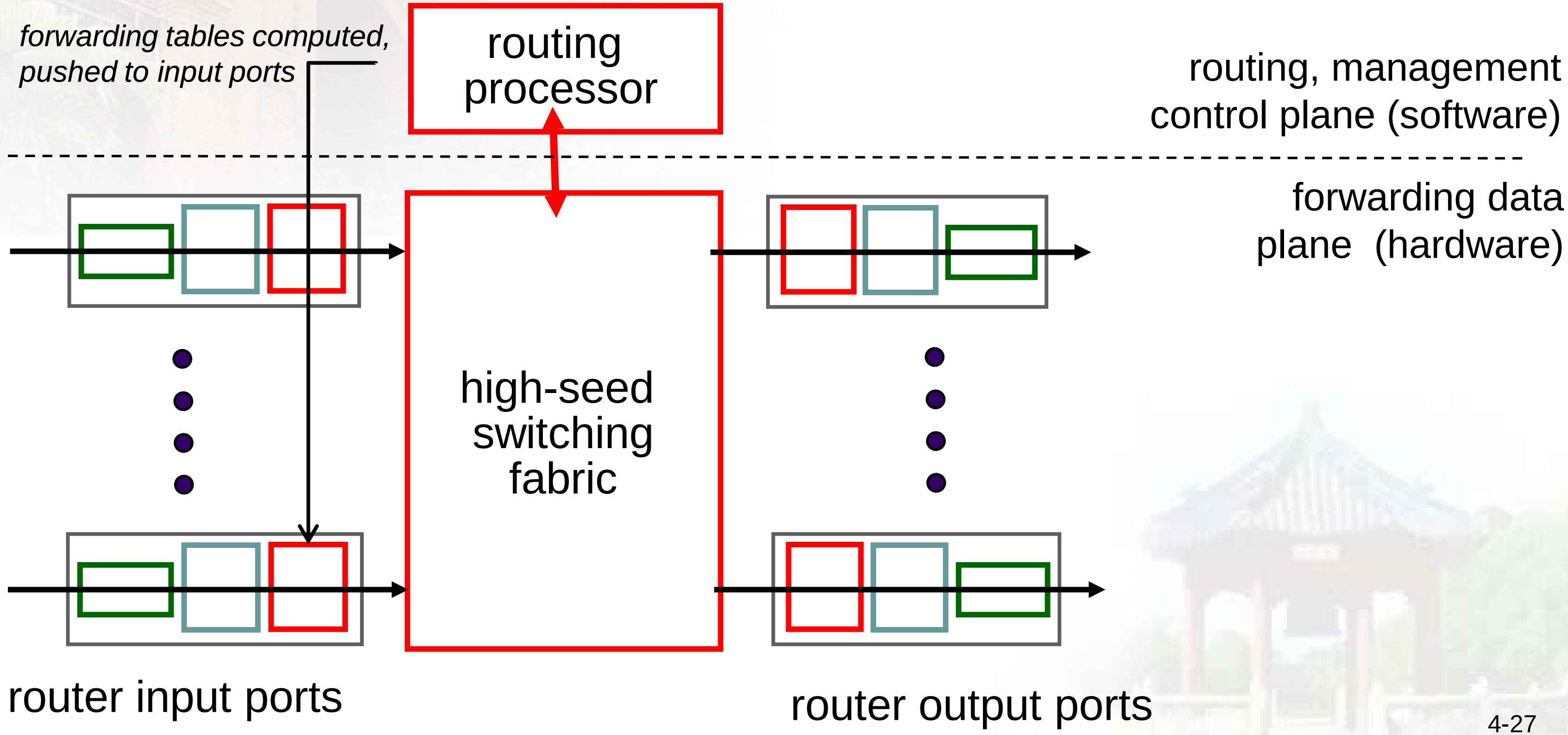
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

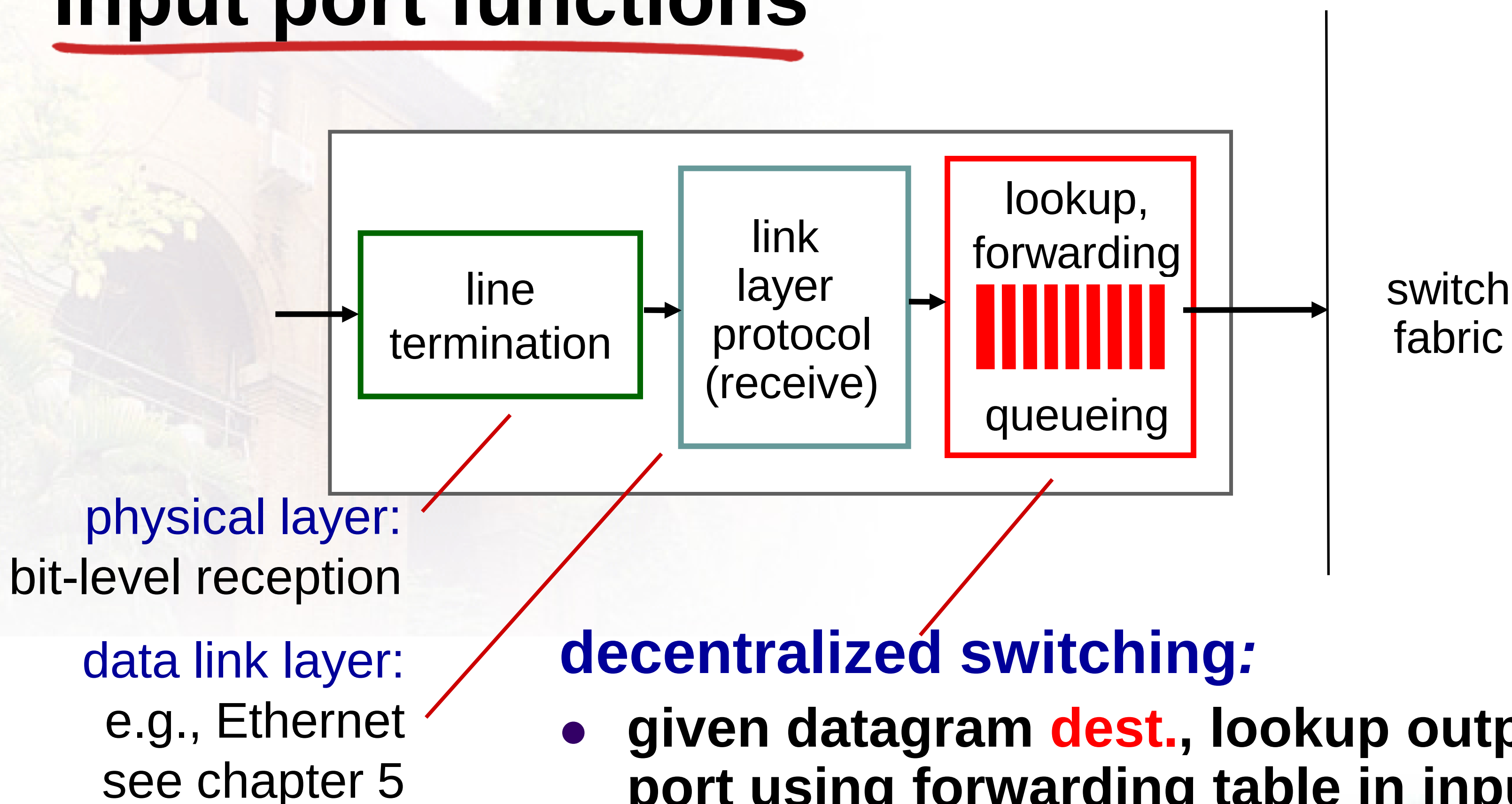
Router architecture overview

two key router functions:

- run **routing** algorithms/protocol (RIP, OSPF, BGP)
- ***forwarding*** datagrams from incoming to outgoing link



Input port functions

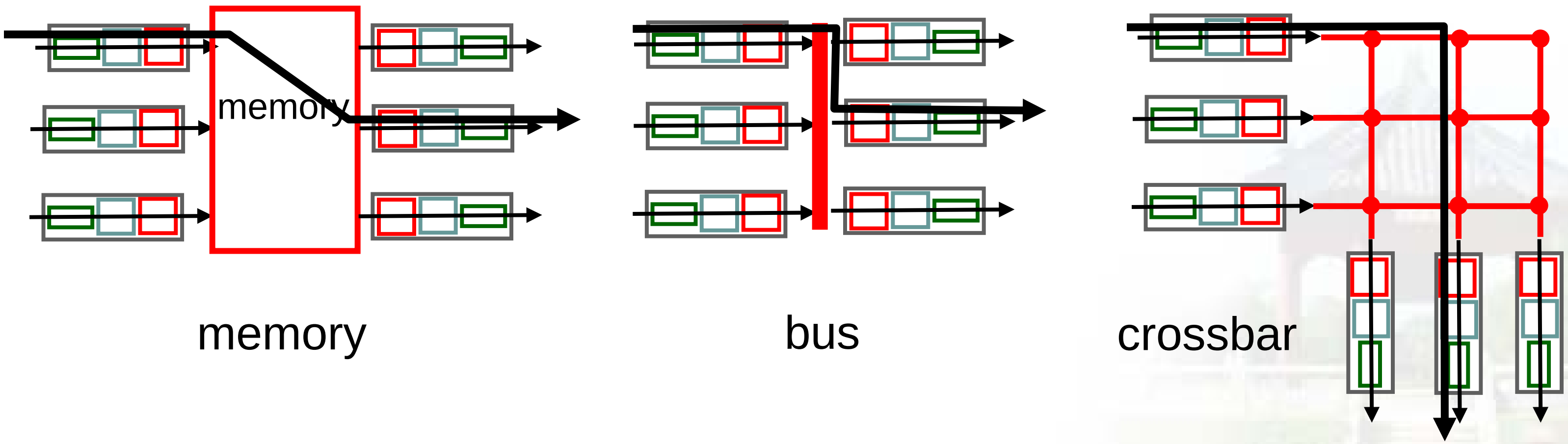


decentralized switching*:*

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

Switching fabrics

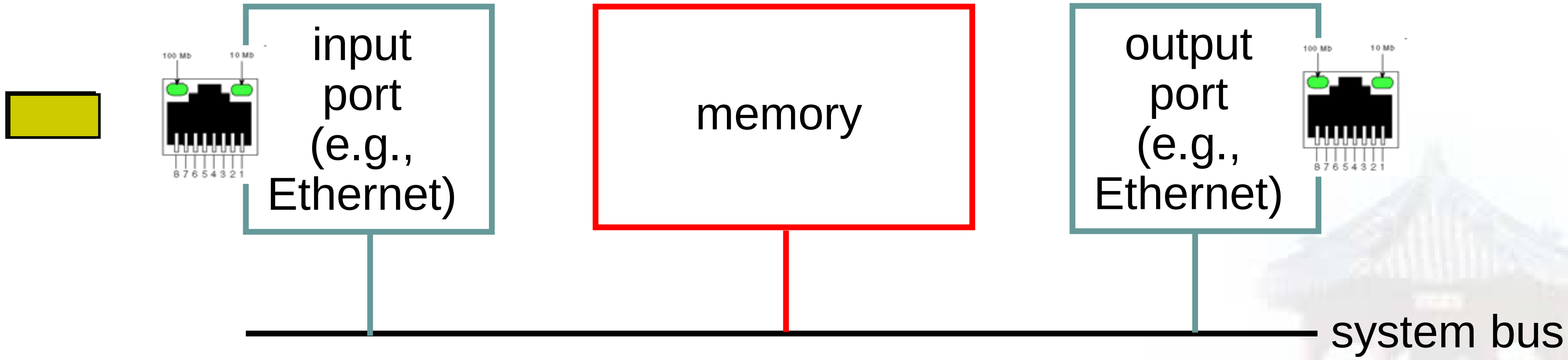
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Switching via memory

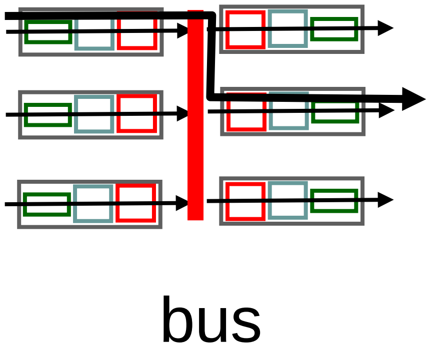
first generation routers:

- **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)**



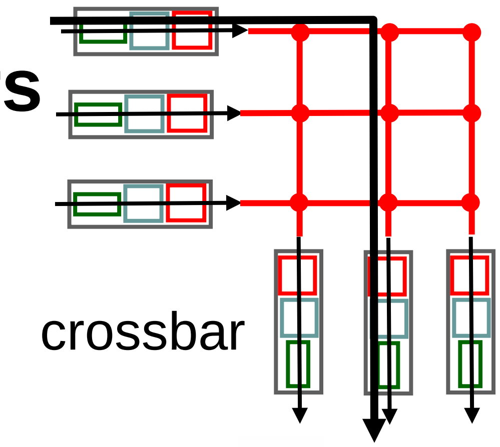
Switching via a bus

- 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



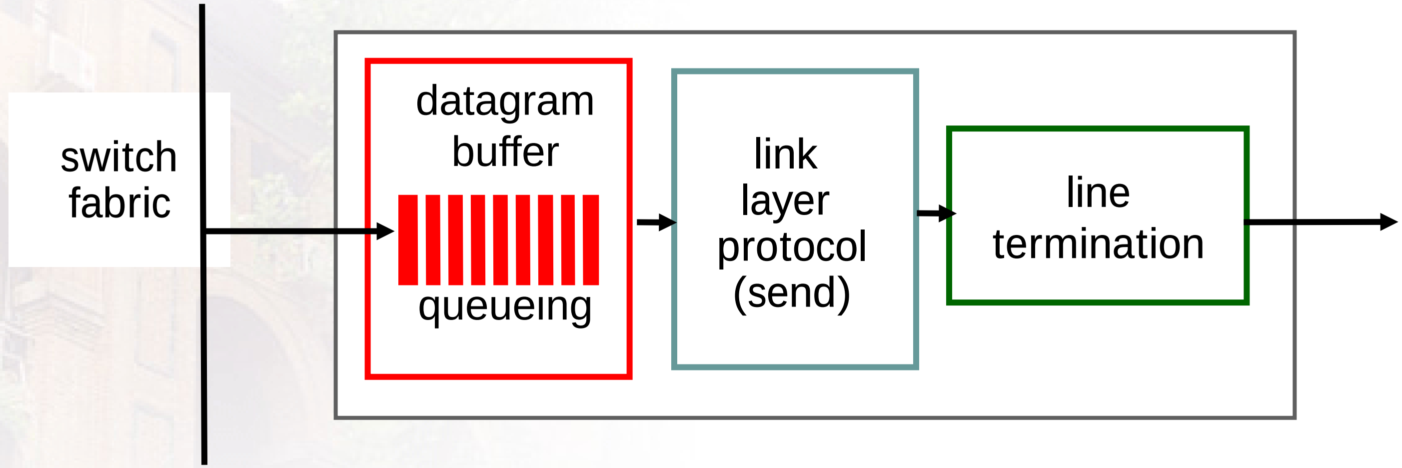
Switching via interconnection network

- overcome bus bandwidth limitations
- banyan networks, crossbar, 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



Output ports

This slide in HUGELY important!



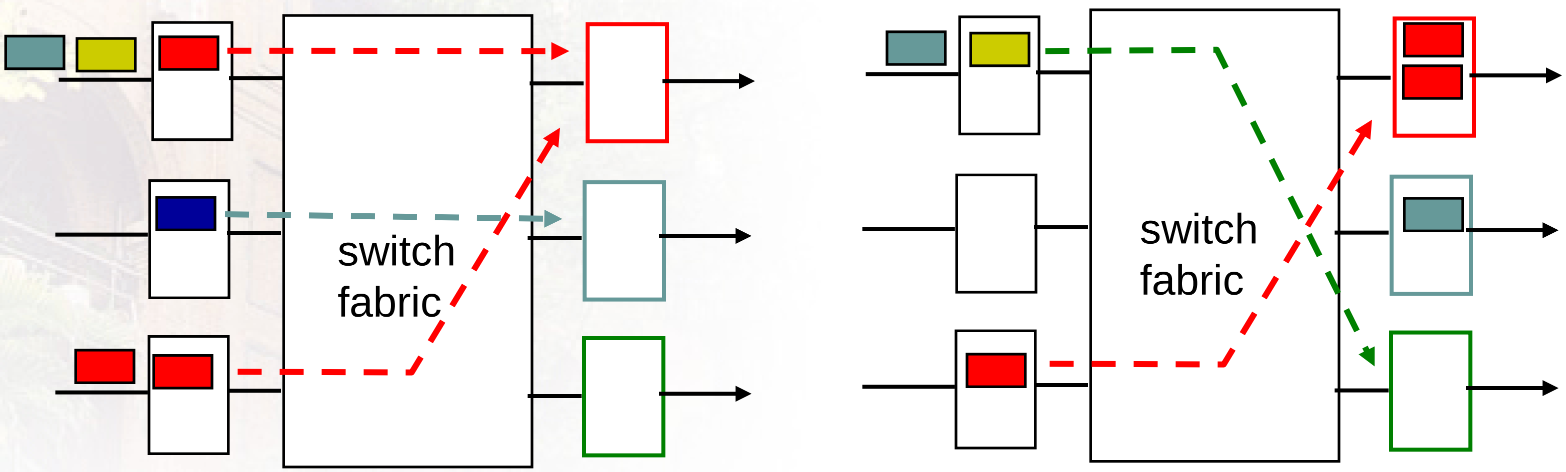
- ***buffering*** **required from fabric faster**

Datagram (packets) can be lost due to congestion, lack of buffers

- ***scheduling discipline*** **chooses among queued datagrams for transmission**

Priority scheduling – who gets best performance, network neutrality

Output port queueing



at *t*, packets more from input to output

one packet time later

- **buffering when arrival rate via switch exceeds output line speed**
- ***queueing (delay) and loss due to output port buffer overflow!***

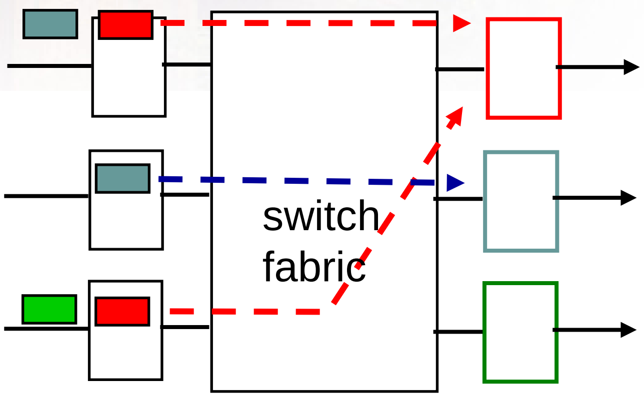
How much buffering?

- RFC 3439 rule of thumb: average buffering equal to "typical" RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gpbs link: 2.5 Gbit buffer
- recent recommendation: with *N* flows, buffering equal to

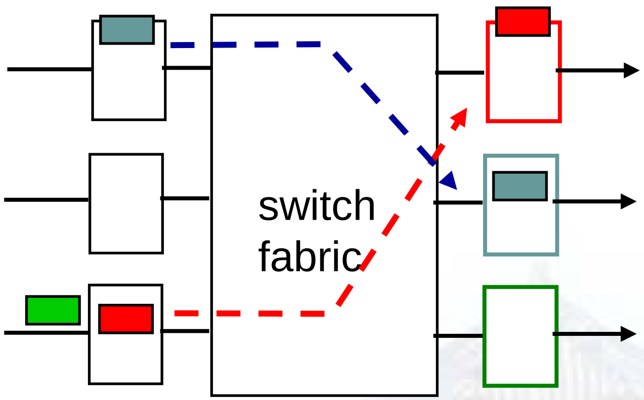
$$\frac{RTT \cdot C}{\sqrt{N}}$$

Input port queuing

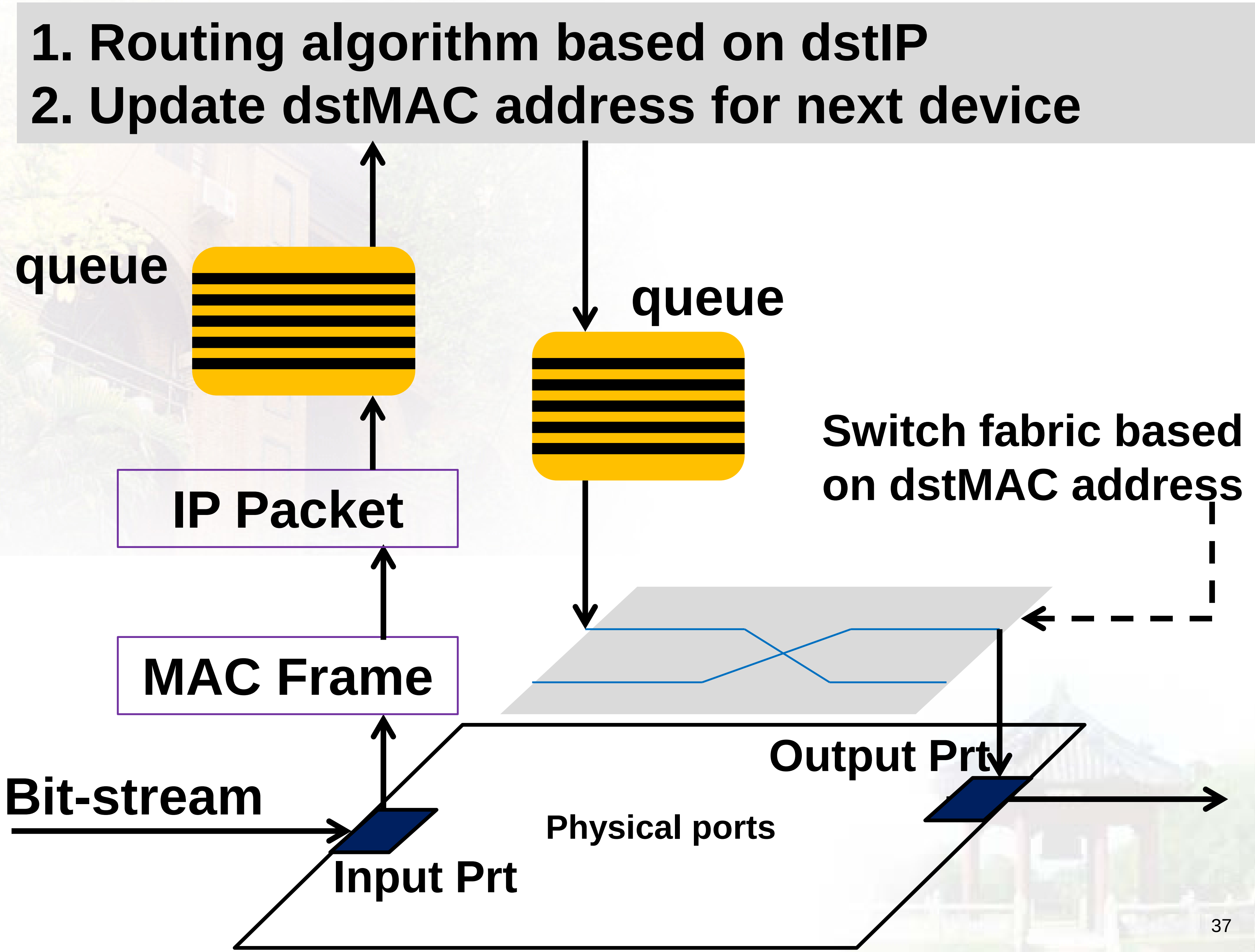
- **fabric slower than input ports combined -> queueing may occur at input queues**
  - ***queueing delay and loss due to input buffer overflow!***
- **Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward**



output port contention:
only one red datagram can be transferred.
lower red packet is blocked



one packet time later: green packet experiences HOL blocking



1. Routing algorithm based on dstIP
2. Update dstMAC address for next device
queue
queue
Switch fabric based on dstMAC address
IP Packet
MAC Frame
Output Prt
Bit-stream
Physical ports
Input Prt

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

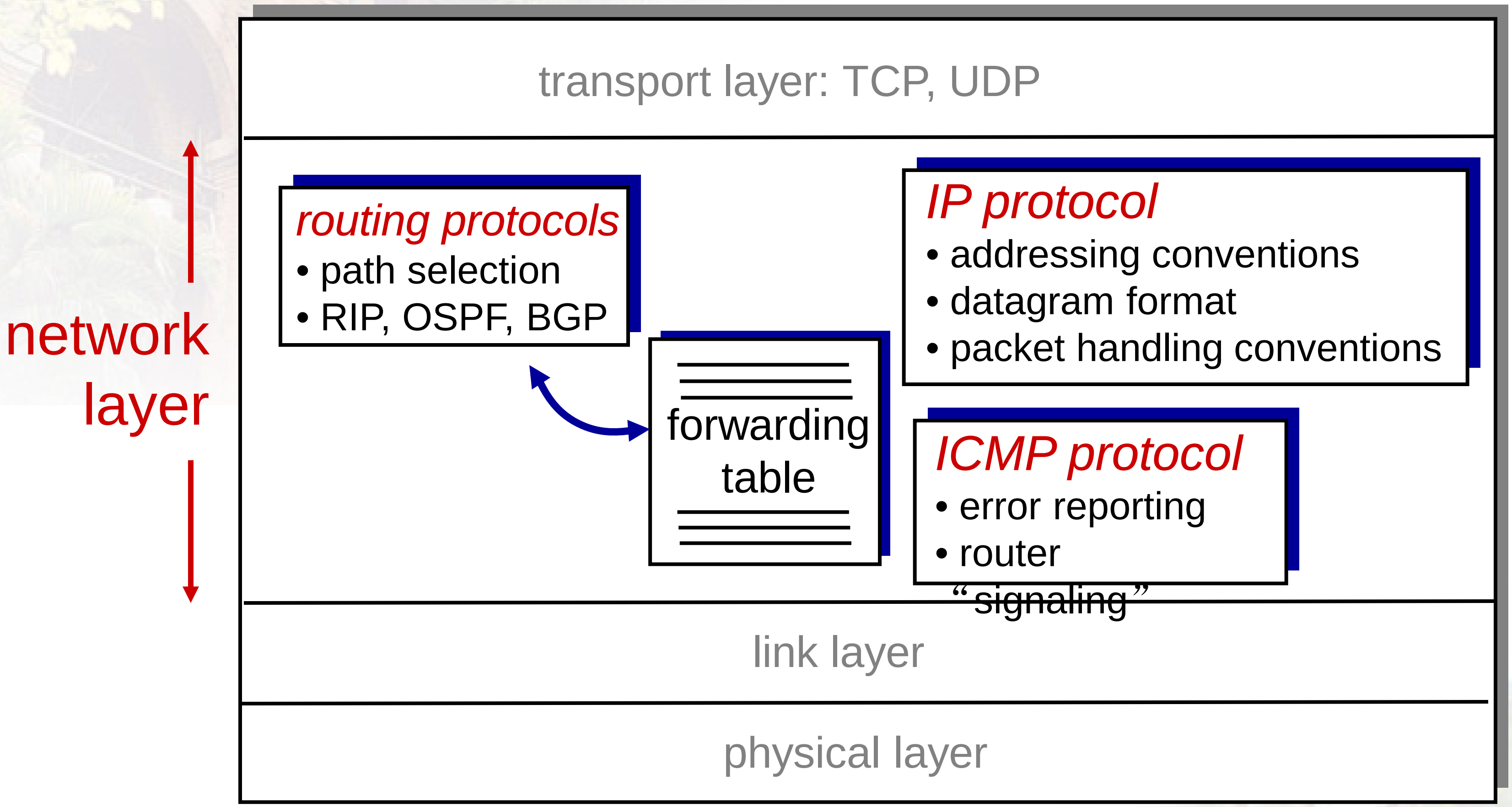
4.6 routing in the Internet

- RIP
- OSPF
- BGP

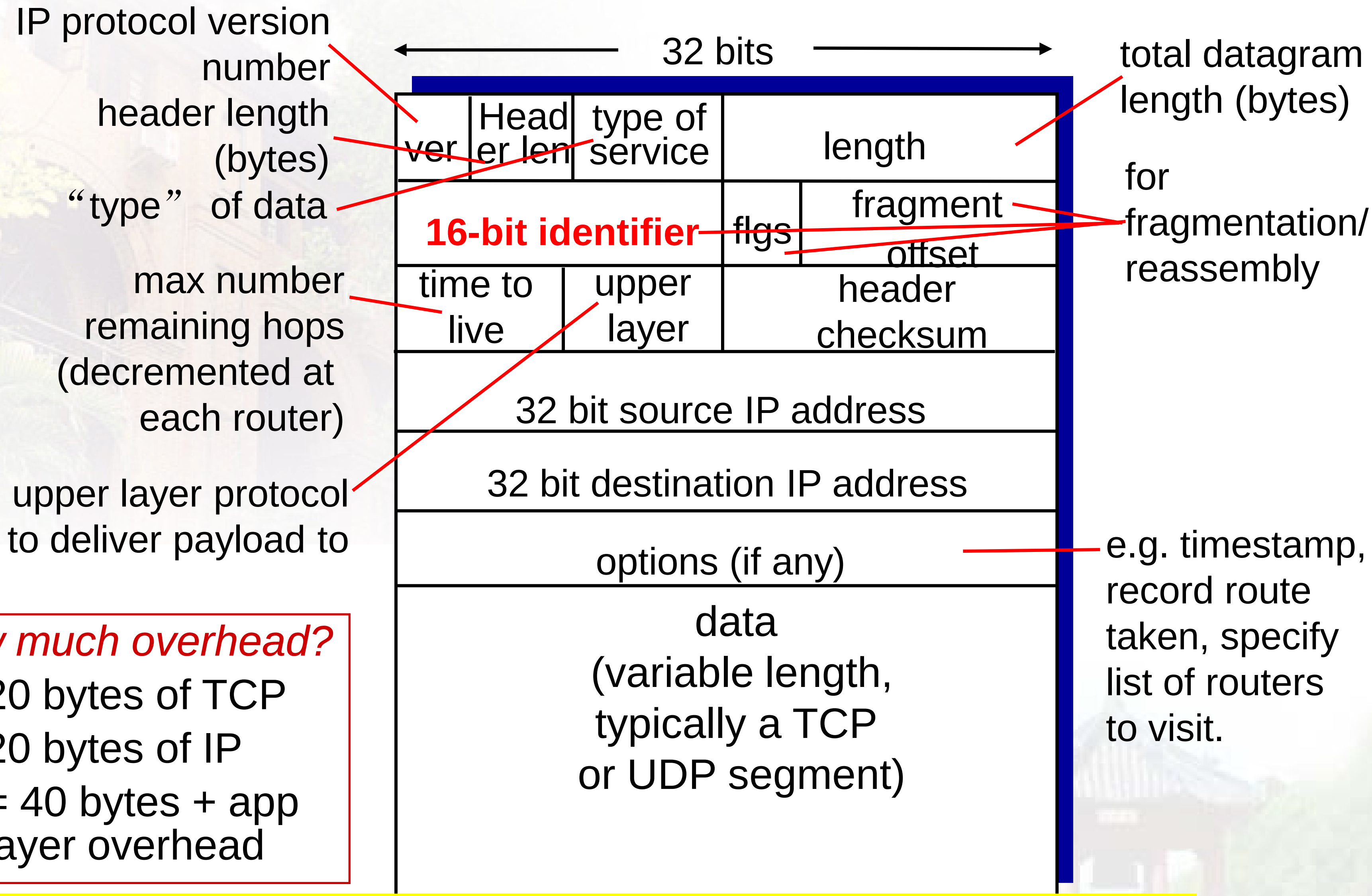
4.7 broadcast and multicast routing

The Internet network layer

host, router network layer functions:



IP datagram format



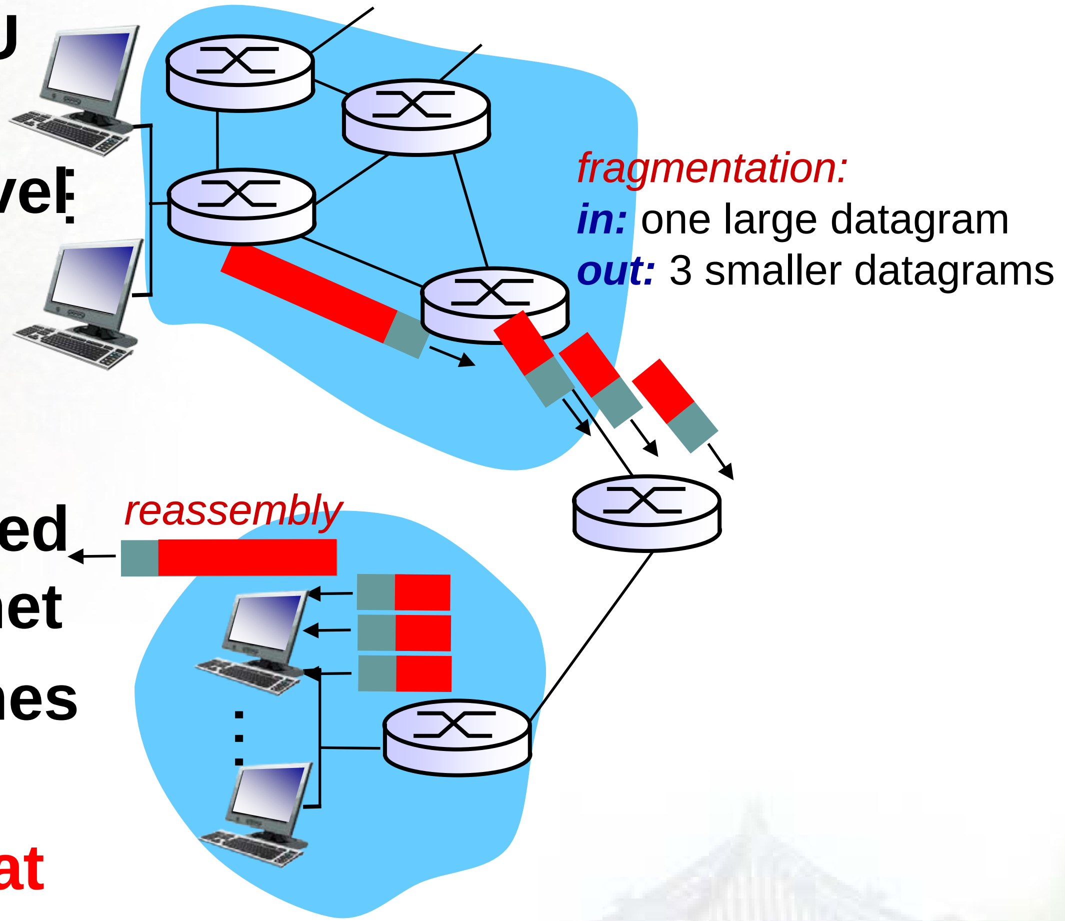
how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

"16-bit identifier" is NOT a number for reordering
No reordering in Net Layer

IP fragmentation, reassembly

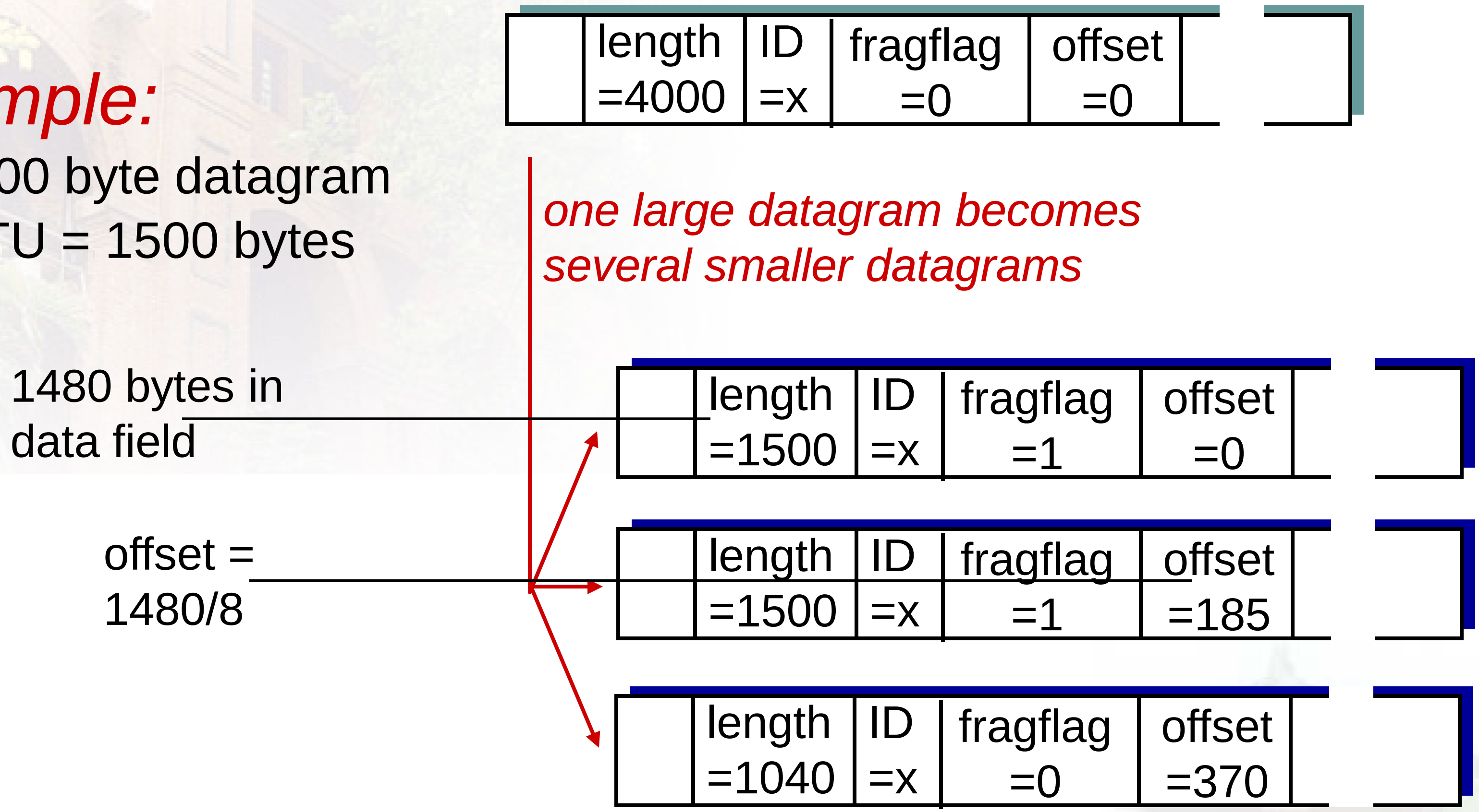
- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order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



IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



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

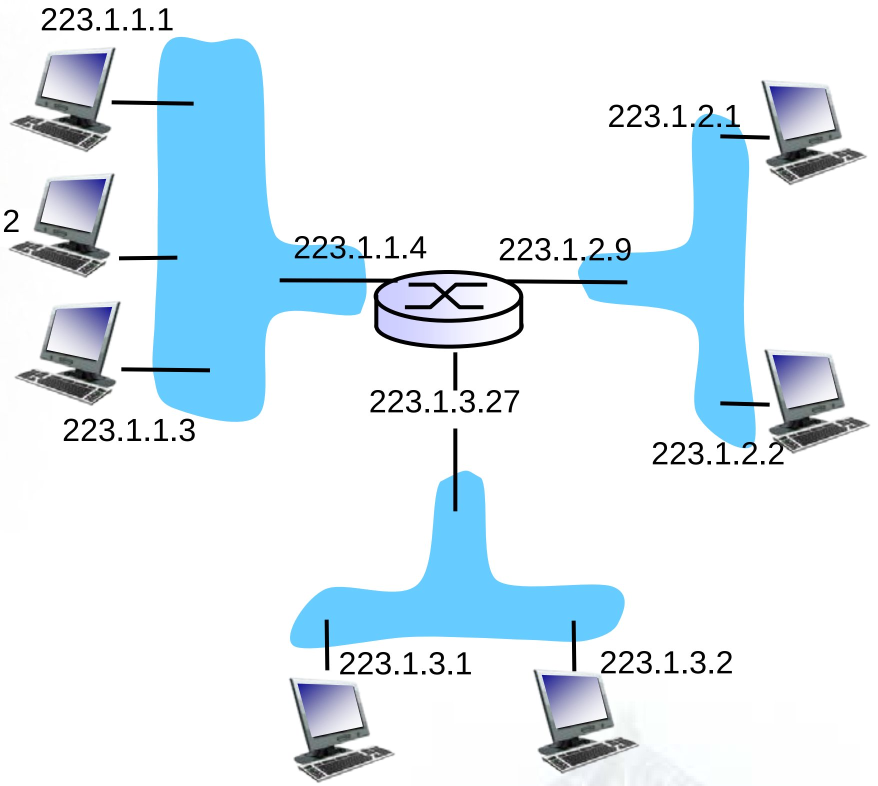
4.6 routing in the Internet

- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

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 or two interfaces (e.g., wired Ethernet, wireless 802.11)**
- ***IP addresses associated with each interface***

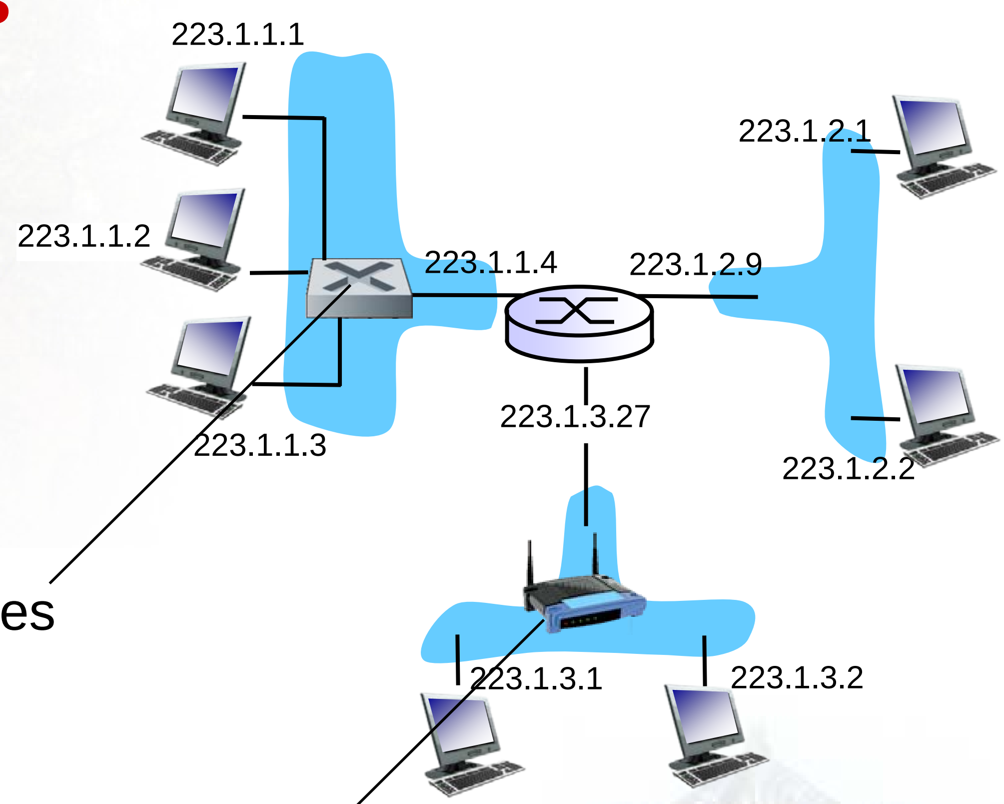


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in chapter 5, 6.



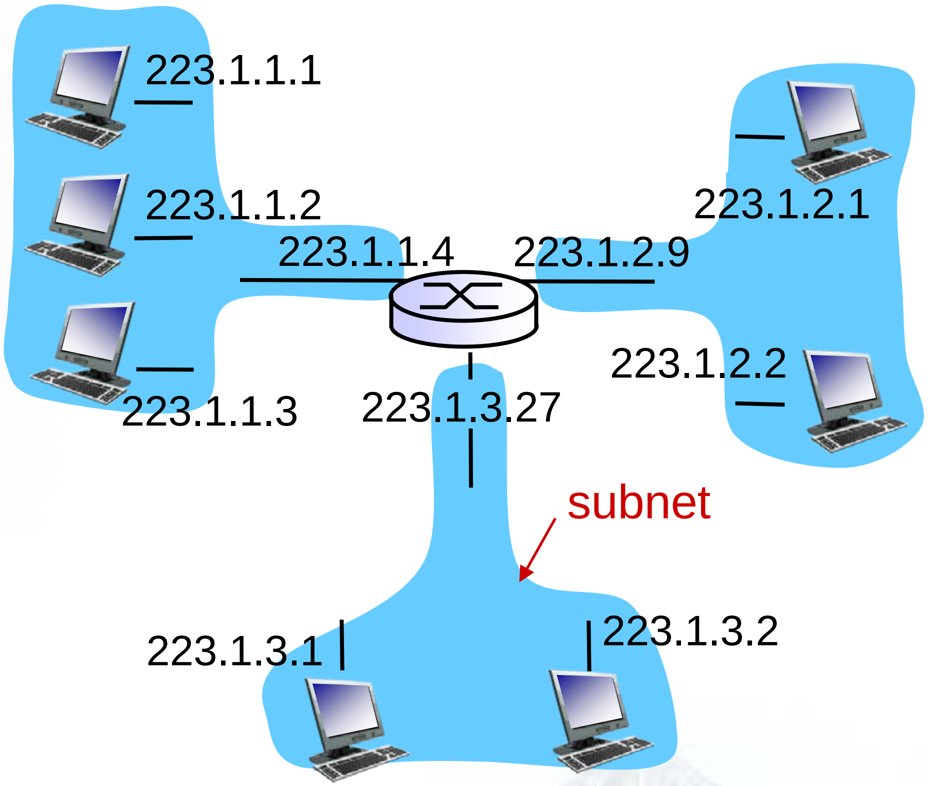
A: wired Ethernet interfaces connected by Ethernet switches

For now: don't need to worry about how one interface is connected to another (with no intervening router)

A: wireless WiFi interfaces connected by WiFi base station

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***

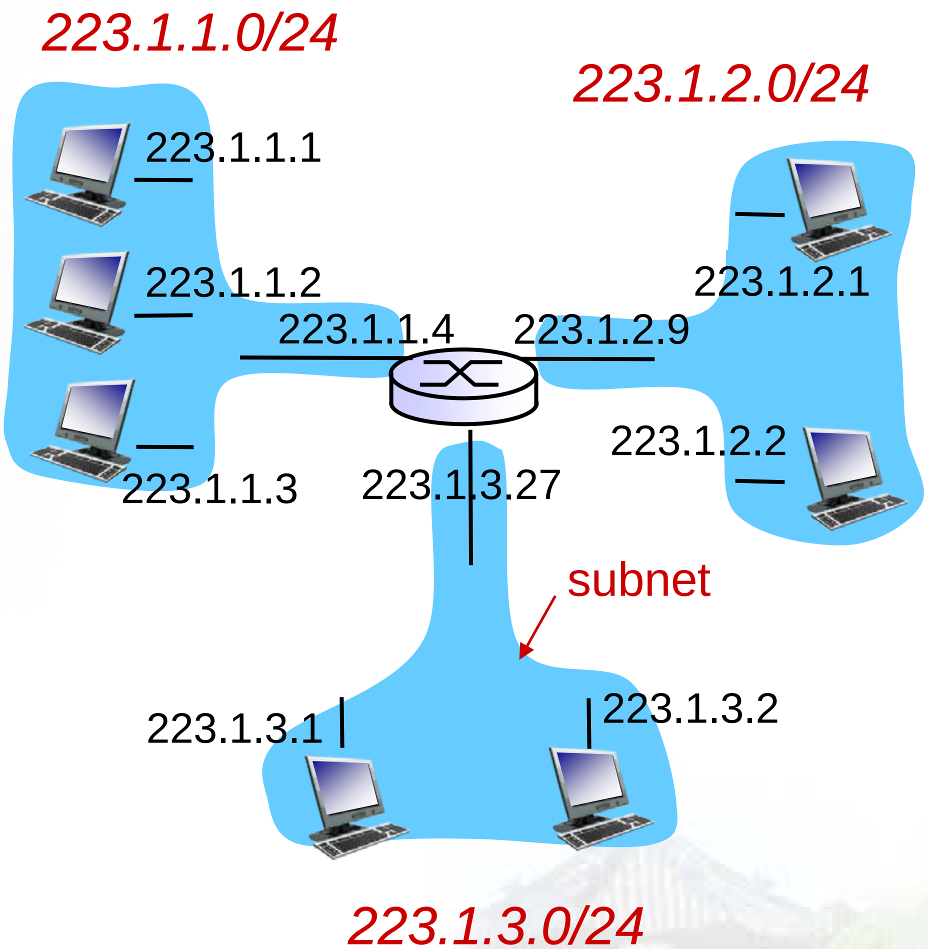


network consisting of 3 subnets

Subnets

recipe

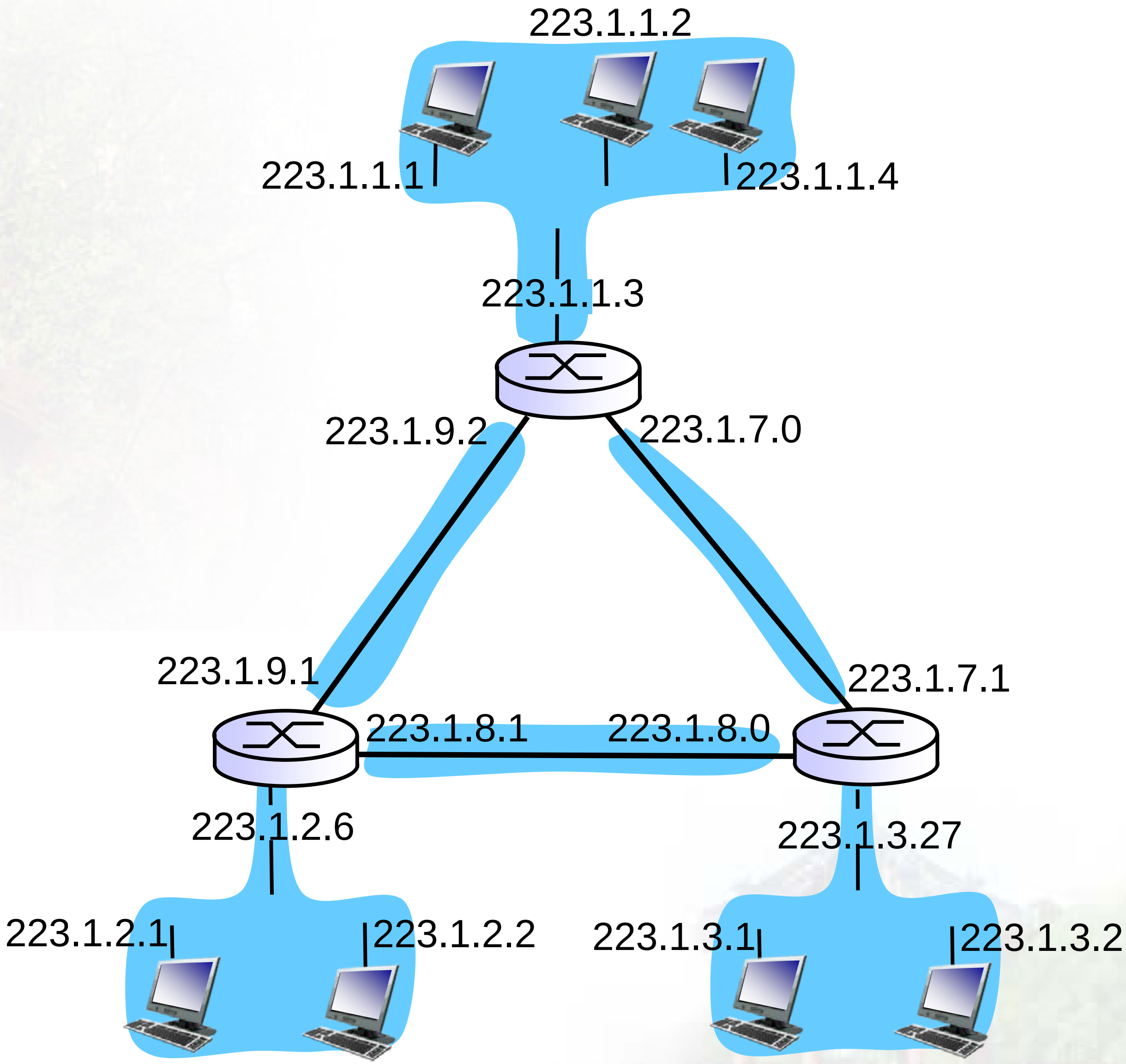
- **to determine the subnets, detach each interface from its host or router, creating islands of isolated networks**
- **each isolated network is called a *subnet***



subnet mask: /24

Subnets

how many?



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

$\underbrace{\text{11001000 00010111 0001000}}_{\text{23bits}}\text{0 00000000}$

23bits 200.23.16.0/23

How to route an IP?

Router gets an IP with dstIP 11001000 00010111 00010110 10100001

Match it with forwarding table: 11001000 00010111 00010000 00000000/21
11001000 10010111 00010000 00000000/22
11001000 10010111 00010000 00000000/15
…

forwarding

IP addresses: how to get one?

Q: 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: D**ynamic **H**ost **C**onfiguration **P**rotocol: dynamically get address from as server
  - "plug-and-play"

DHCP: Dynamic Host Configuration Protocol

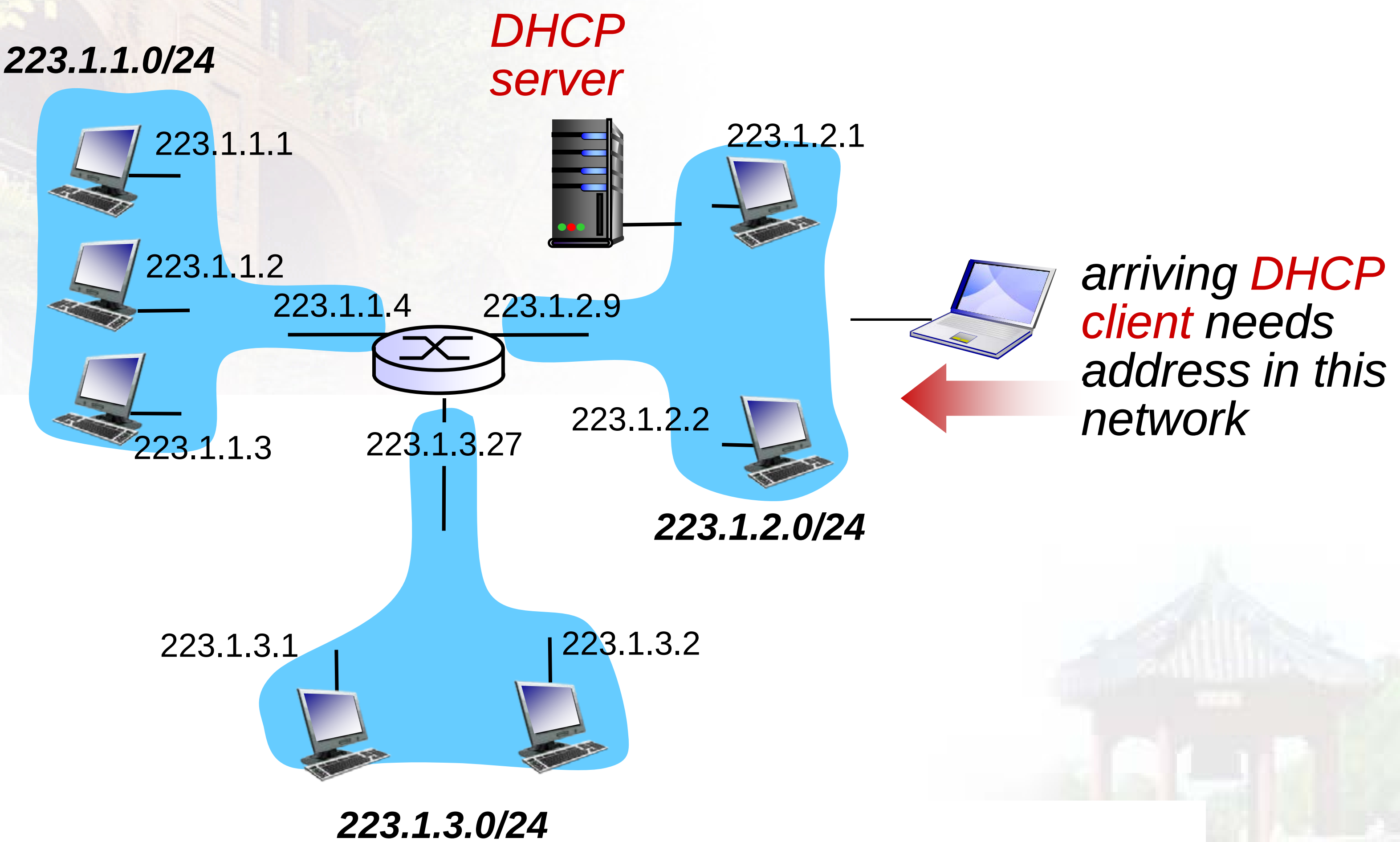
goal: **allow host to *dynamically* obtain its IP address from network server when it joins network**

- **can renew its lease on address in use**
- **allows reuse of addresses (only hold address while connected/"on")**
- **support for mobile users who want to join network (more shortly)**

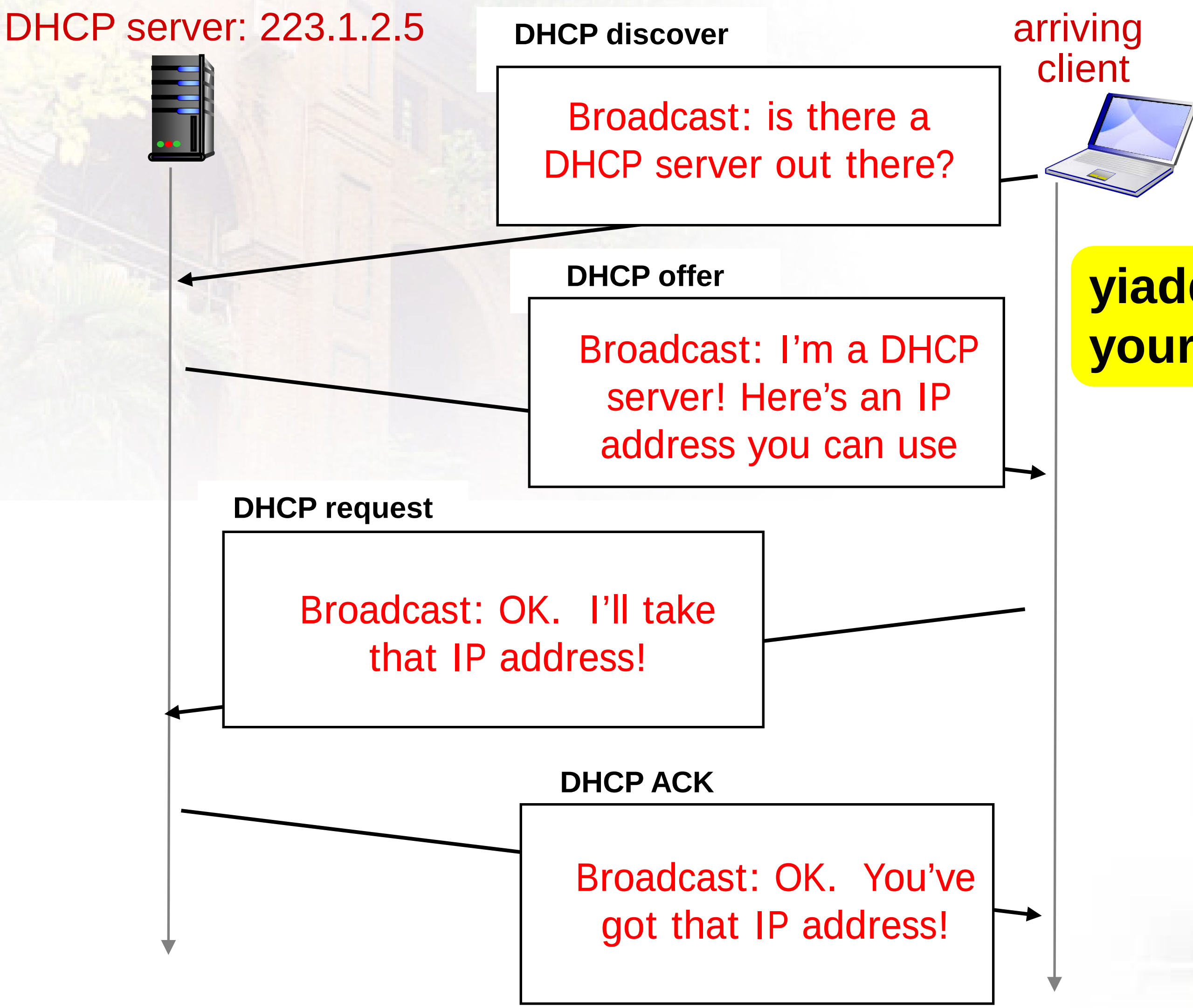
DHCP overview:

- **host broadcasts "DHCP discover" msg [optional]**
- **DHCP server responds with "DHCP offer" msg [optional]**
- **host requests IP address: "DHCP request" msg**
- **DHCP server sends address: "DHCP ack" msg**

DHCP client-server scenario



DHCP client-server scenario



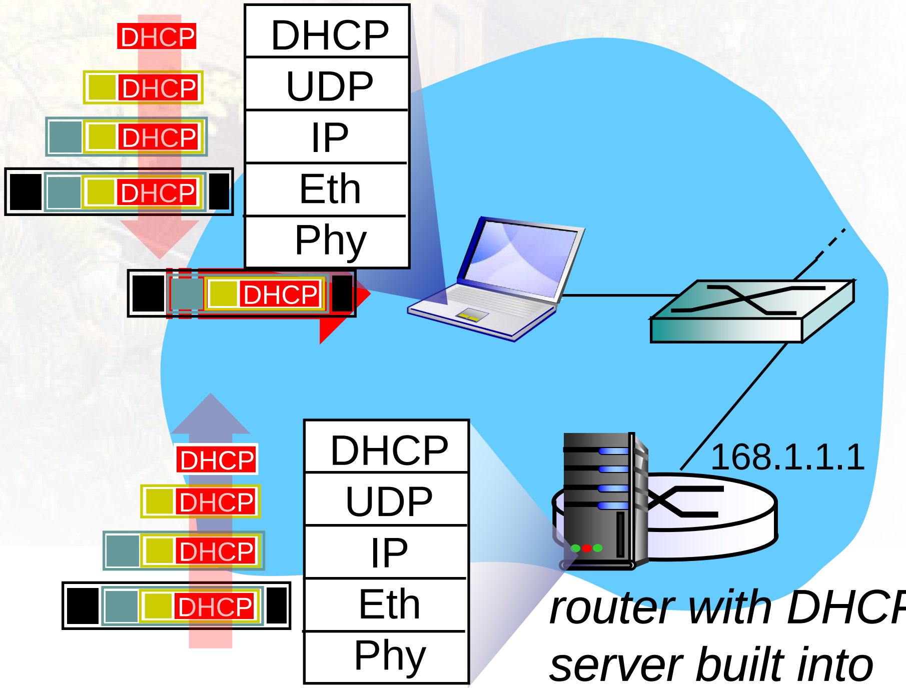
yiaddr:
your Internet address

DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

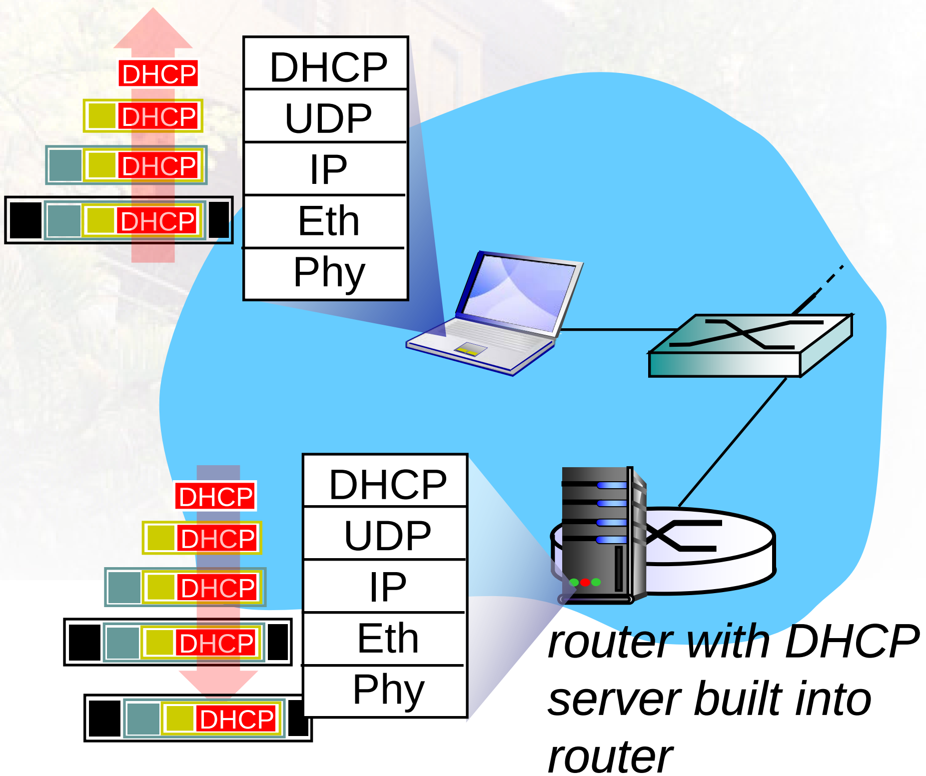
DHCP: example



router with DHCP server built into router

- **connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP**
- DHCP request encapsulated in **UDP**, encapsulated in IP, encapsulated in 802.1 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

DHCP: example



- **DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server**
- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

DHCP: Wireshark output (home LAN)

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
    Length: 7; Value: 010016D323688A;
    Hardware type: Ethernet
    Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
    Length: 11; Value: 010F03062C2E2F1F21F92B
    1 = Subnet Mask; 15 = Domain Name
    3 = Router; 6 = Domain Name Server
    44 = NetBIOS over TCP/IP Name Server
    ……

request

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
    Length: 12; Value: 445747E2445749F244574092;
    IP Address: 68.87.71.226;
    IP Address: 68.87.73.242;
    IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

reply

IP addresses: how to get one?

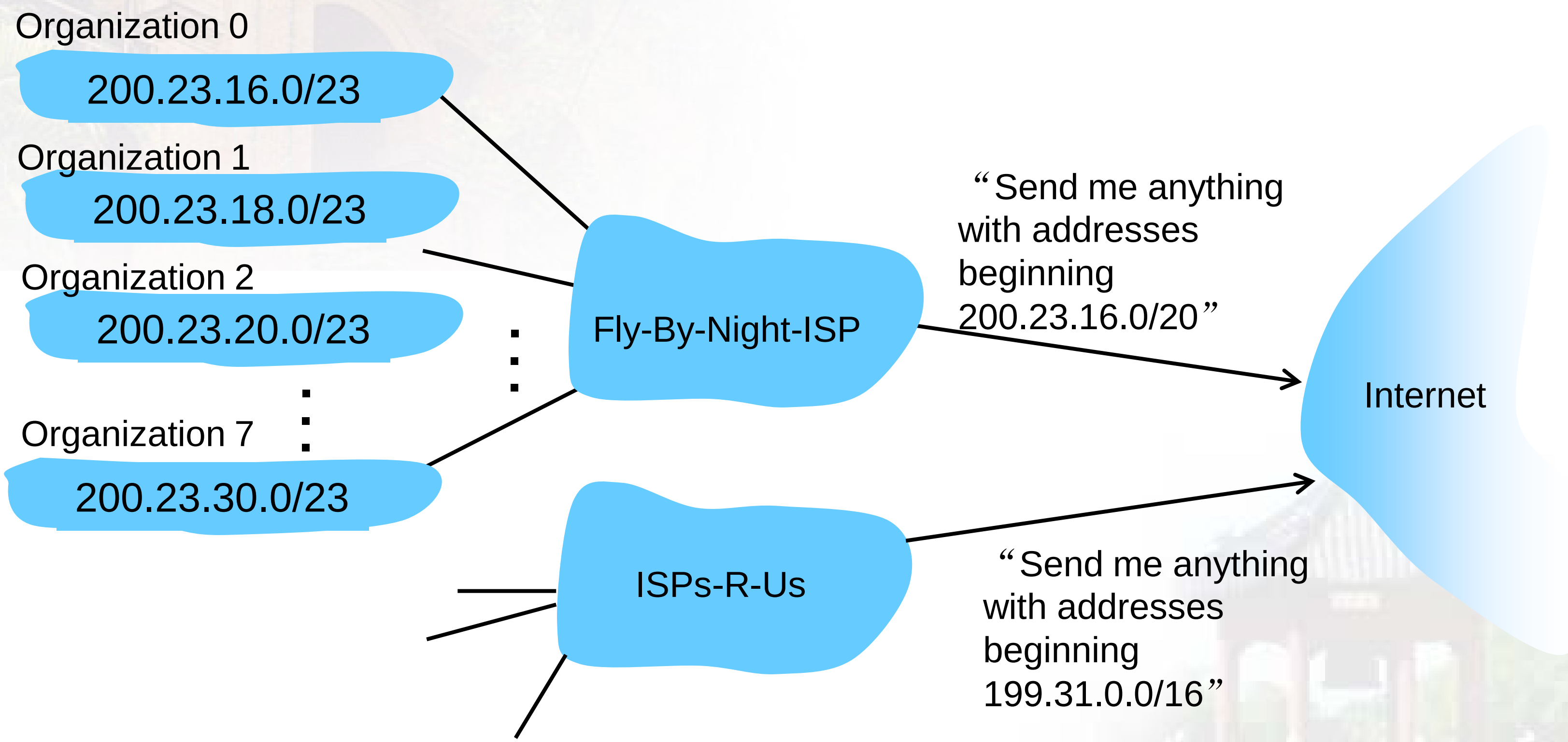
Q: how does *network* get subnet part of IP addr?

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

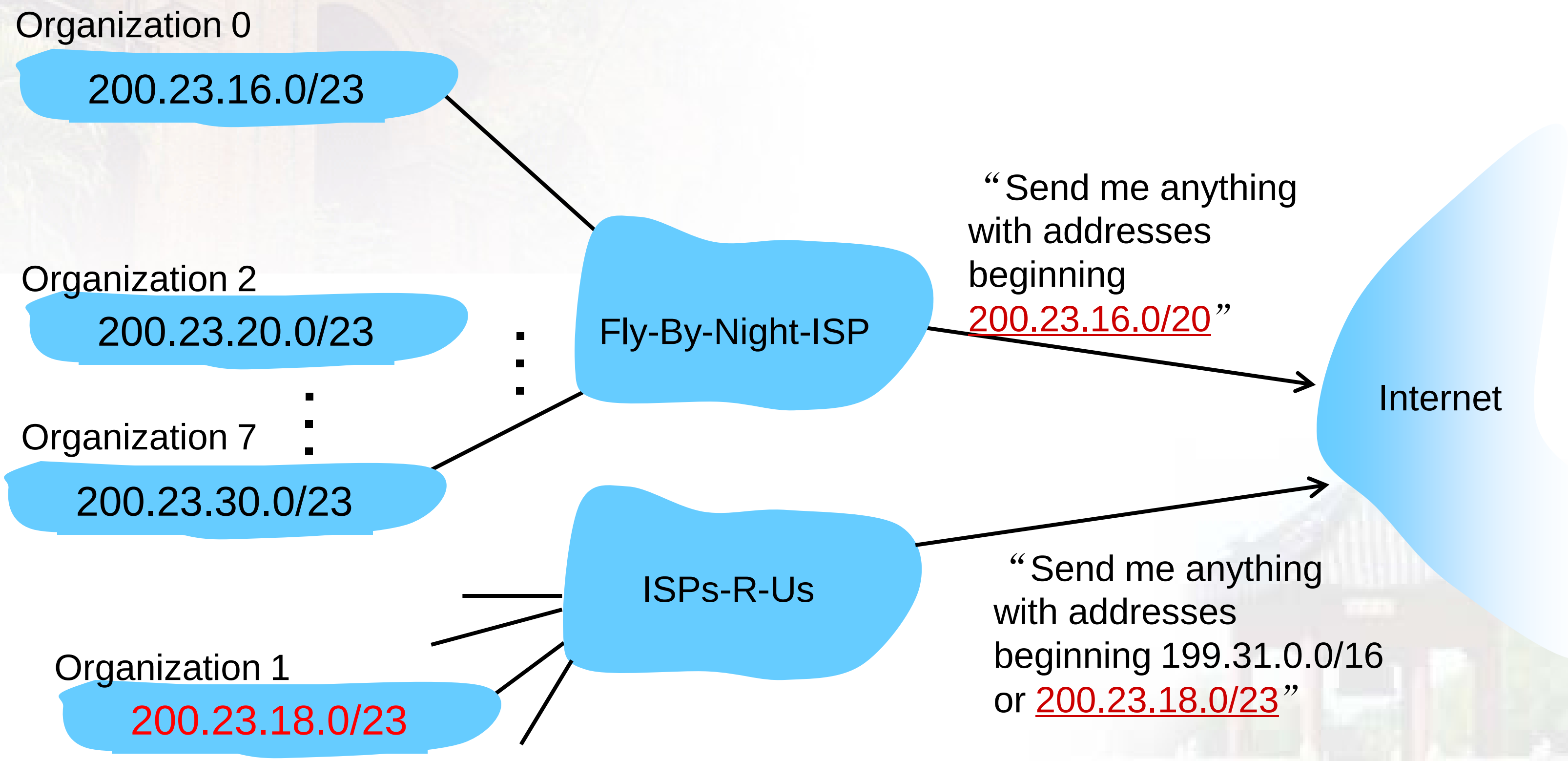
Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



Hierarchical addressing: more specific routes

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



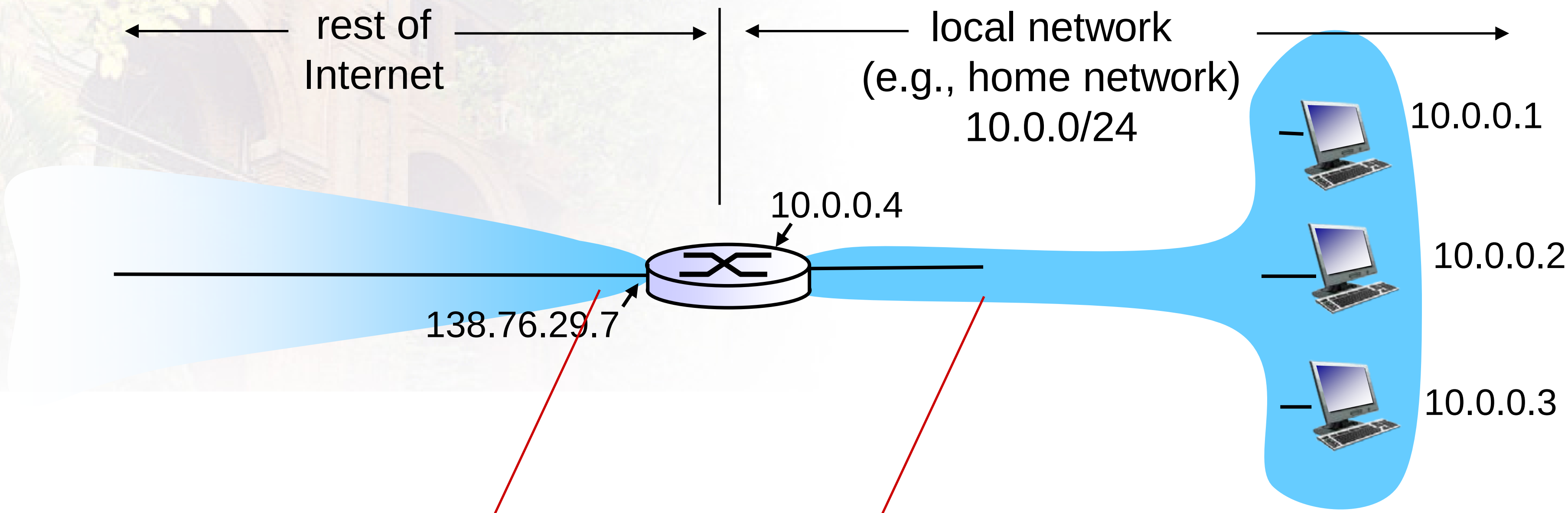
IP addressing: the last word...

Q: **how does an ISP get block of addresses?**

A: **ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/**

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

NAT: network address translation



all datagrams *leaving* local network have *same* single source NAT IP address: 138.76.29.7,different source port numbers

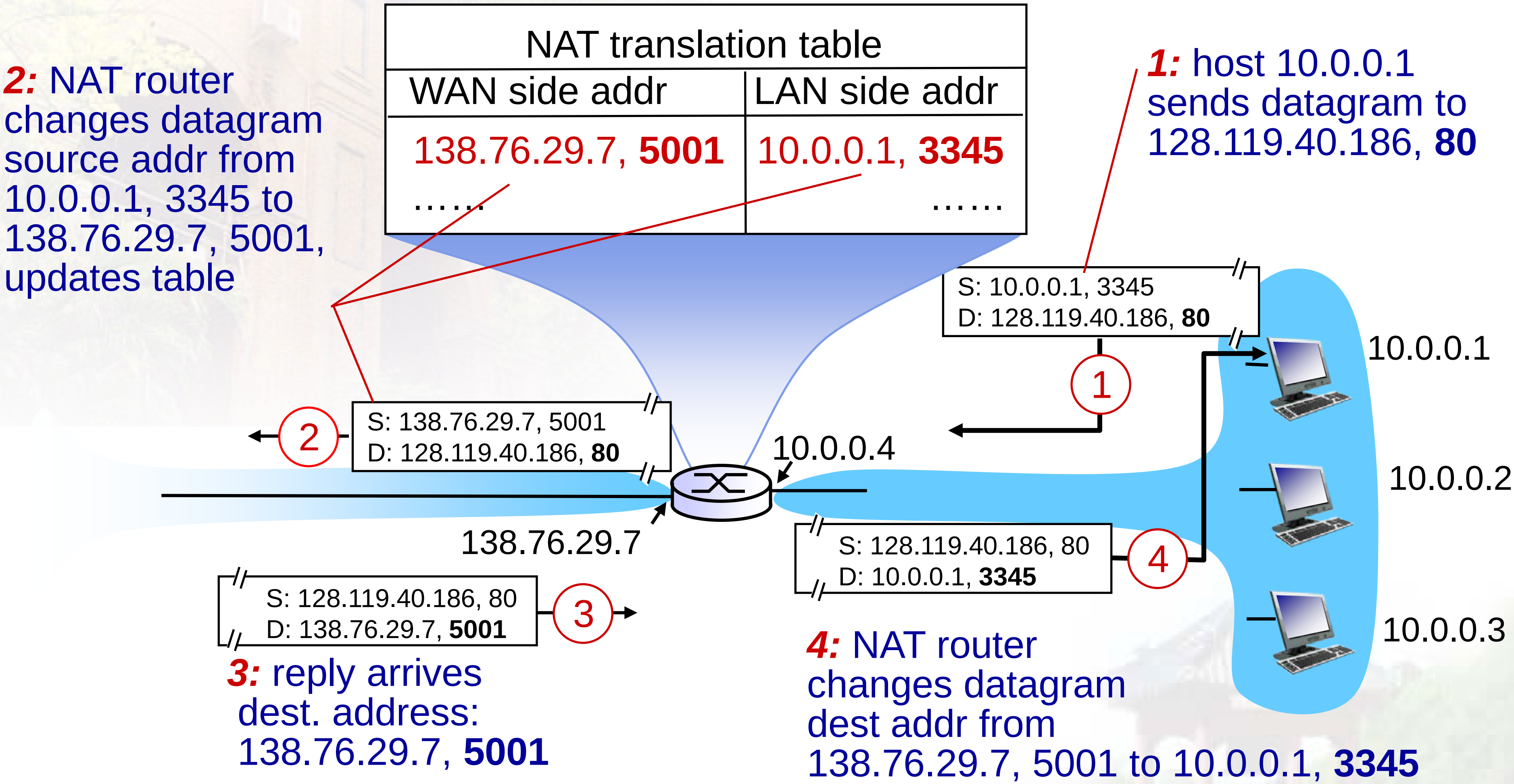
datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: network address translation

motivation: local network uses just one IP address as far as outside world is concerned:

- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network
- devices inside local net not explicitly addressable, visible by outside world (a security plus)

NAT: network address translation



NAT: network address translation

implementation: NAT router must:

- ***outgoing datagrams: replace*** **(source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)**

  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr

- ***remember (in NAT translation table)*** **every (source IP address, port #) to (NAT IP address, new port #) translation pair**

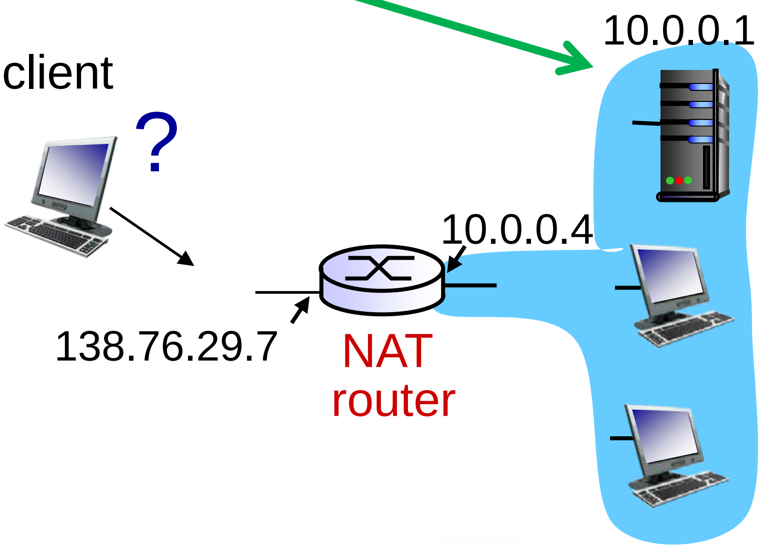
- ***incoming datagrams: replace*** **(NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table**

NAT: network address translation

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should only process up to layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - address shortage should instead be solved by IPv6

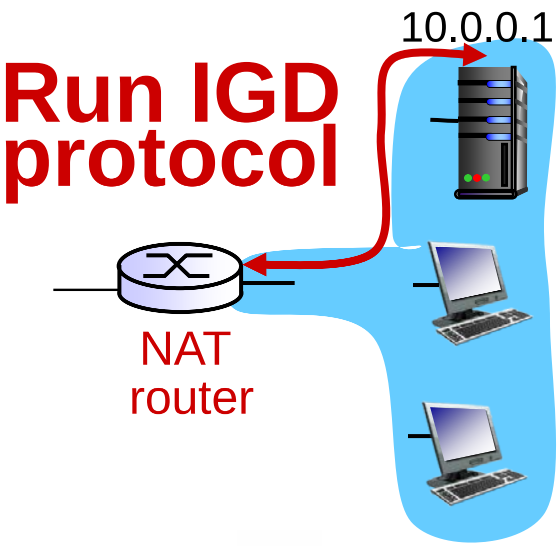
NAT traversal problem

- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATed address: 138.76.29.7
- *solution1:* statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000



NAT traversal problem

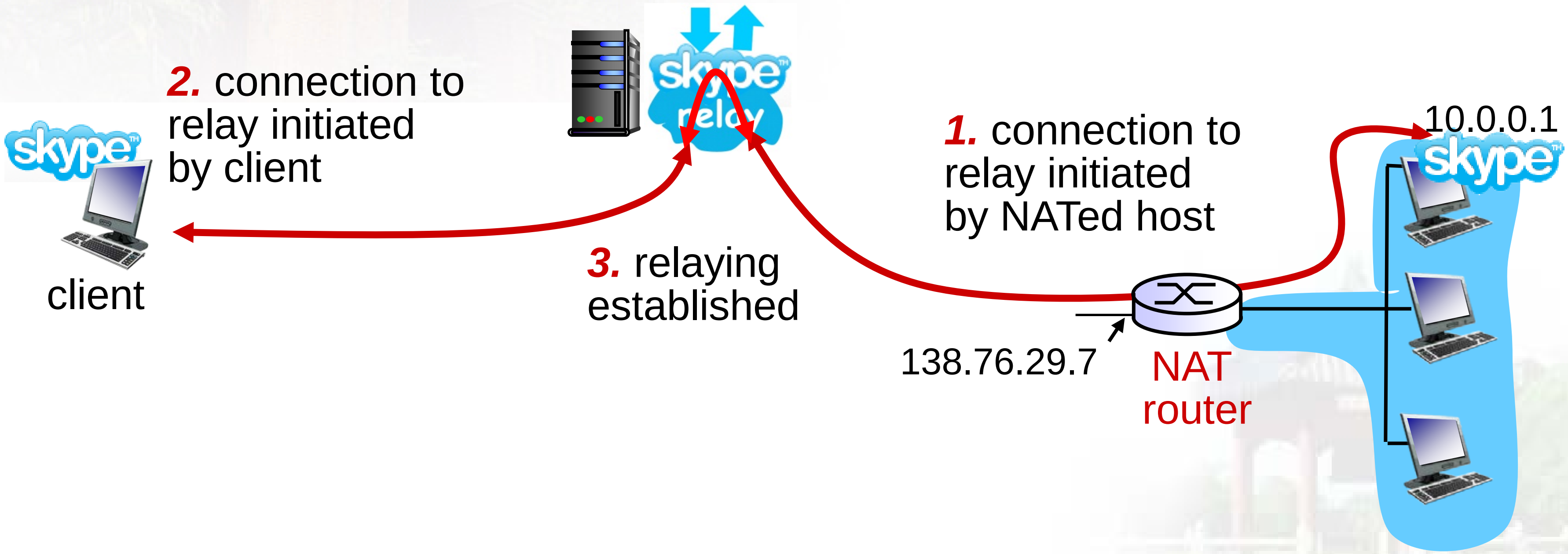
- *solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - learn public IP address (138.76.29.7)
  - add/remove port mappings (with lease times)

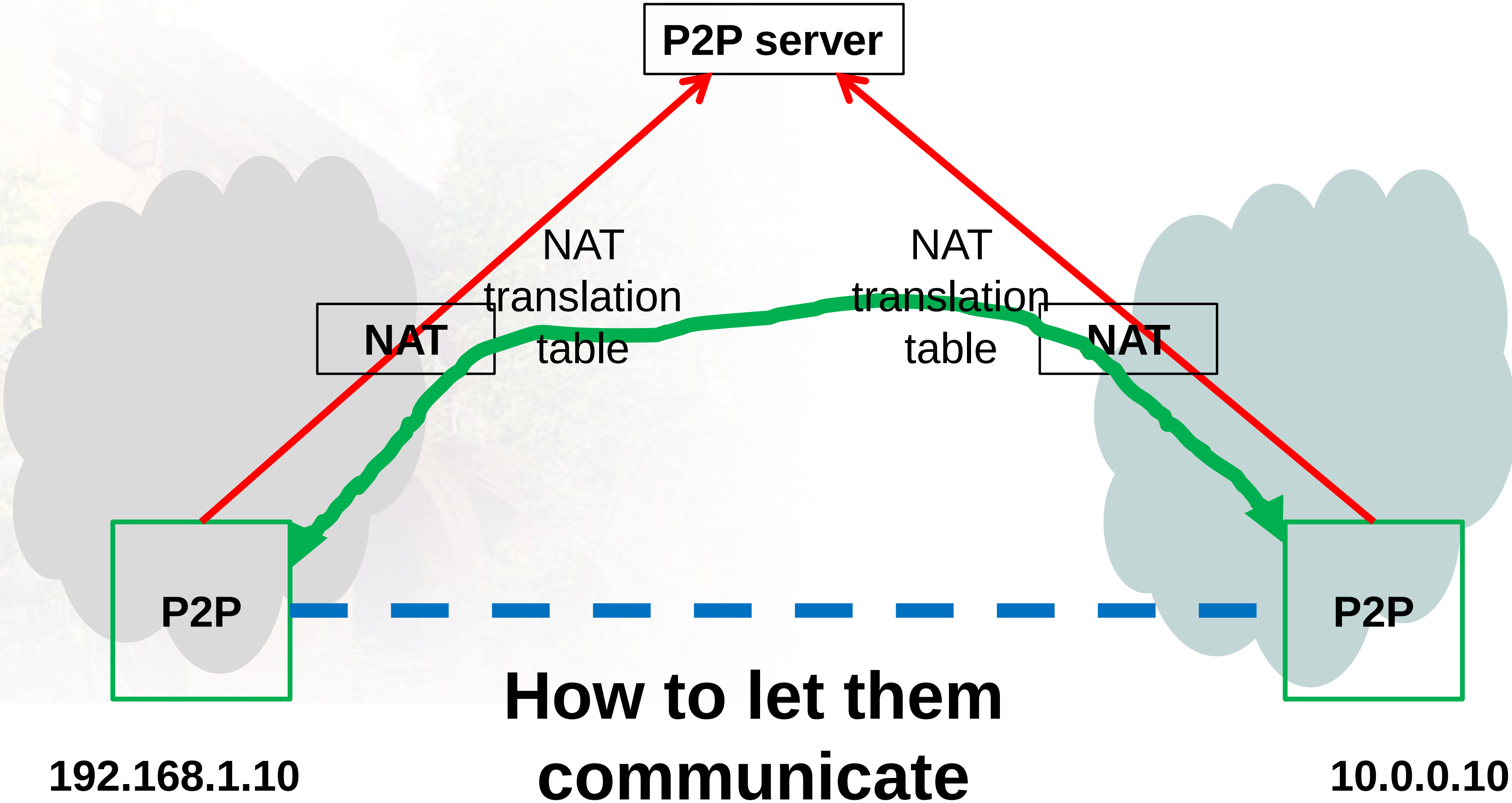


i.e., automate static NAT port map configuration

NAT traversal problem

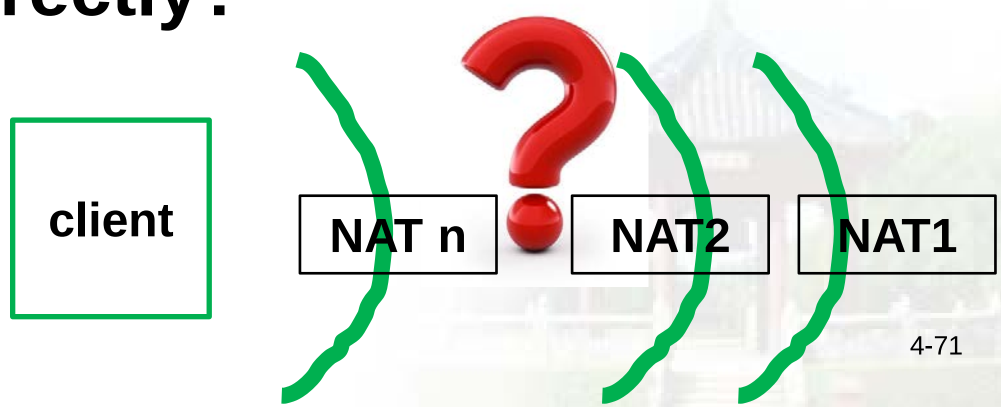
- *solution 3:* relaying (used in Skype)
  - NATed client establishes connection to relay
  - external client connects to relay
  - relay bridges packets between to connections





How to let them communicate directly?

UDP hole punching
TCP hole punching



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

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

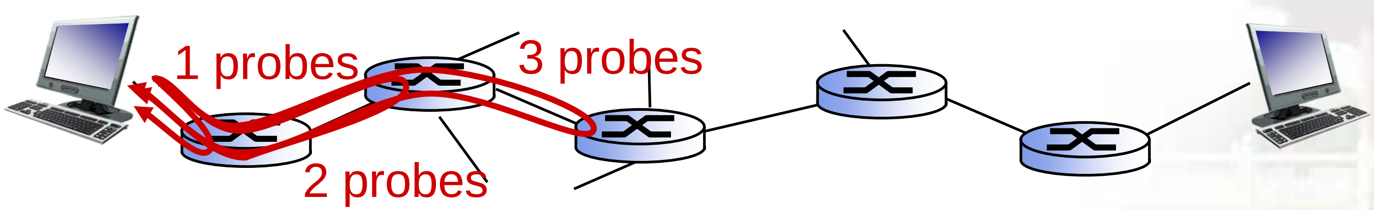
Traceroute and ICMP

- **source sends series of UDP segments to dest**
  - **first set has TTL =1**
  - **second set has TTL=2, etc.**
  - **unlikely port number**
- **when *n*th set of datagrams arrives to nth router:**
  - **router discards datagrams**
  - **and sends source ICMP messages (type 11, code 0)**
  - **ICMP messages includes name of router & IP address**

- **when ICMP messages arrives, source records RTTs**

stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops



IPv6: motivation

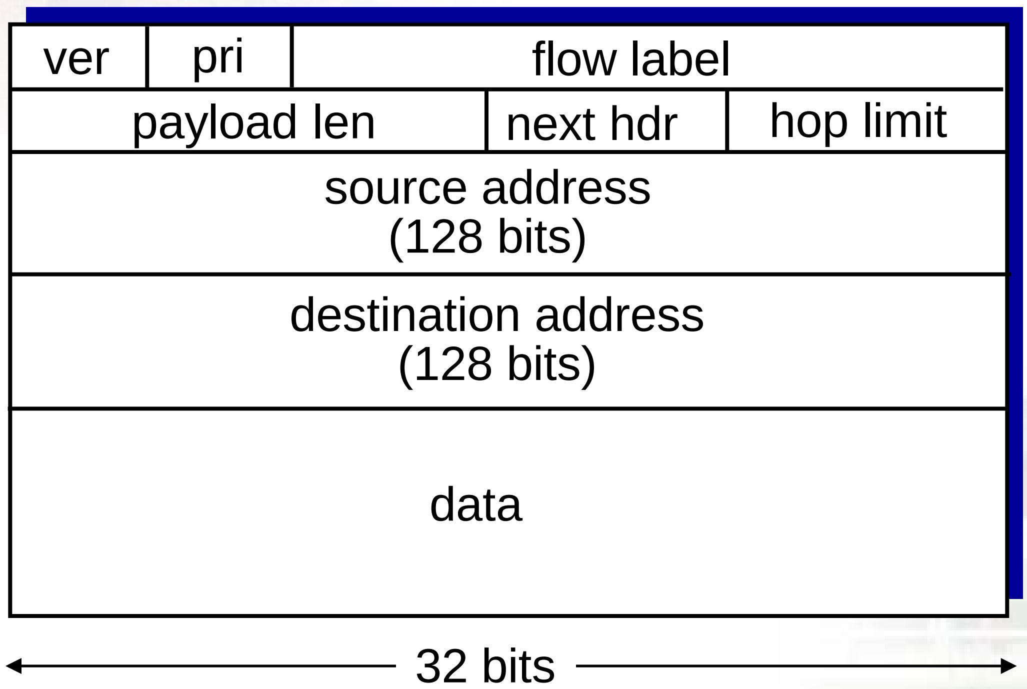
- ***initial motivation:*** **32-bit address space soon to be completely allocated.**
- **additional motivation:**
  - **header format helps speed processing/forwarding**
  - **header changes to facilitate QoS**

IPv6 datagram format:

- **fixed-length 40 byte header**
- **no fragmentation allowed**

IPv6 datagram format

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

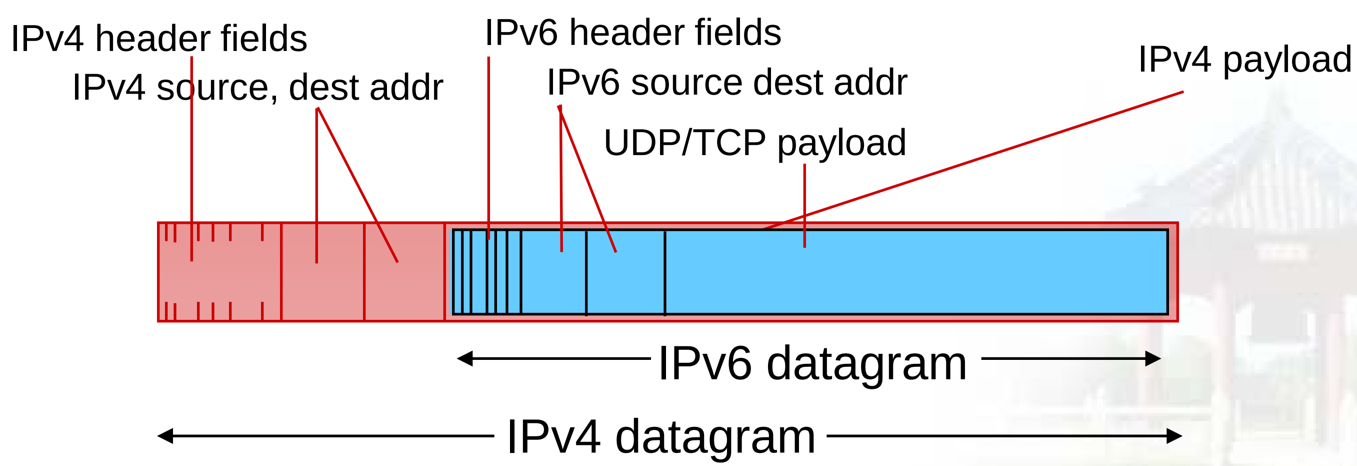


Other changes from IPv4

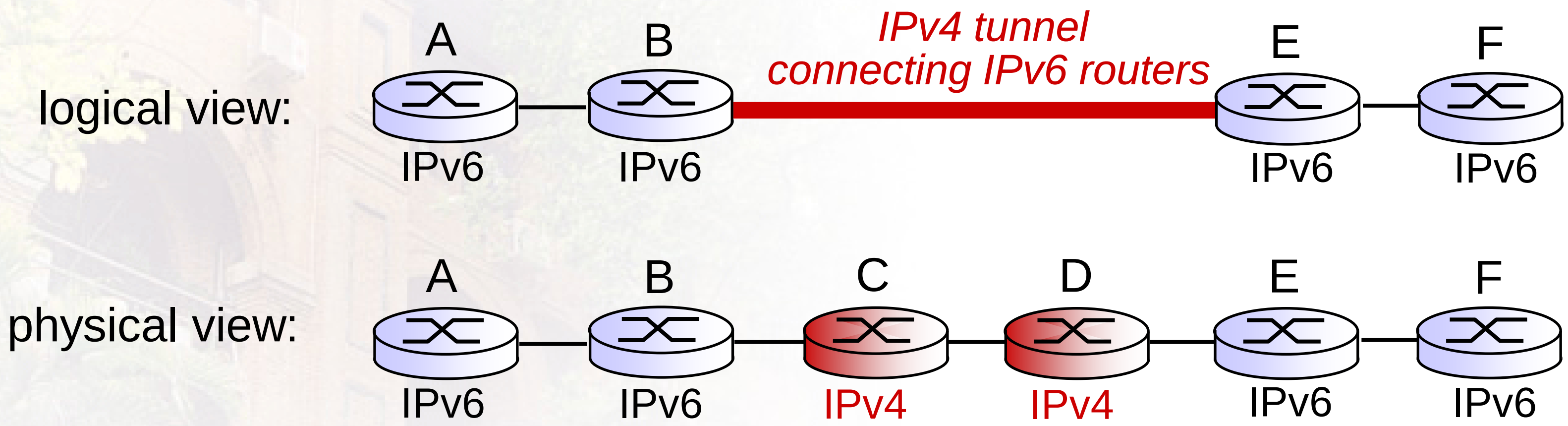
- *checksum*: removed entirely to reduce processing time at each hop
- *options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition from IPv4 to IPv6

- not all routers can be upgraded simultaneously
  - no "flag days"
  - how will network operate with mixed IPv4 and IPv6 routers?
- ***tunneling:*** **IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers**

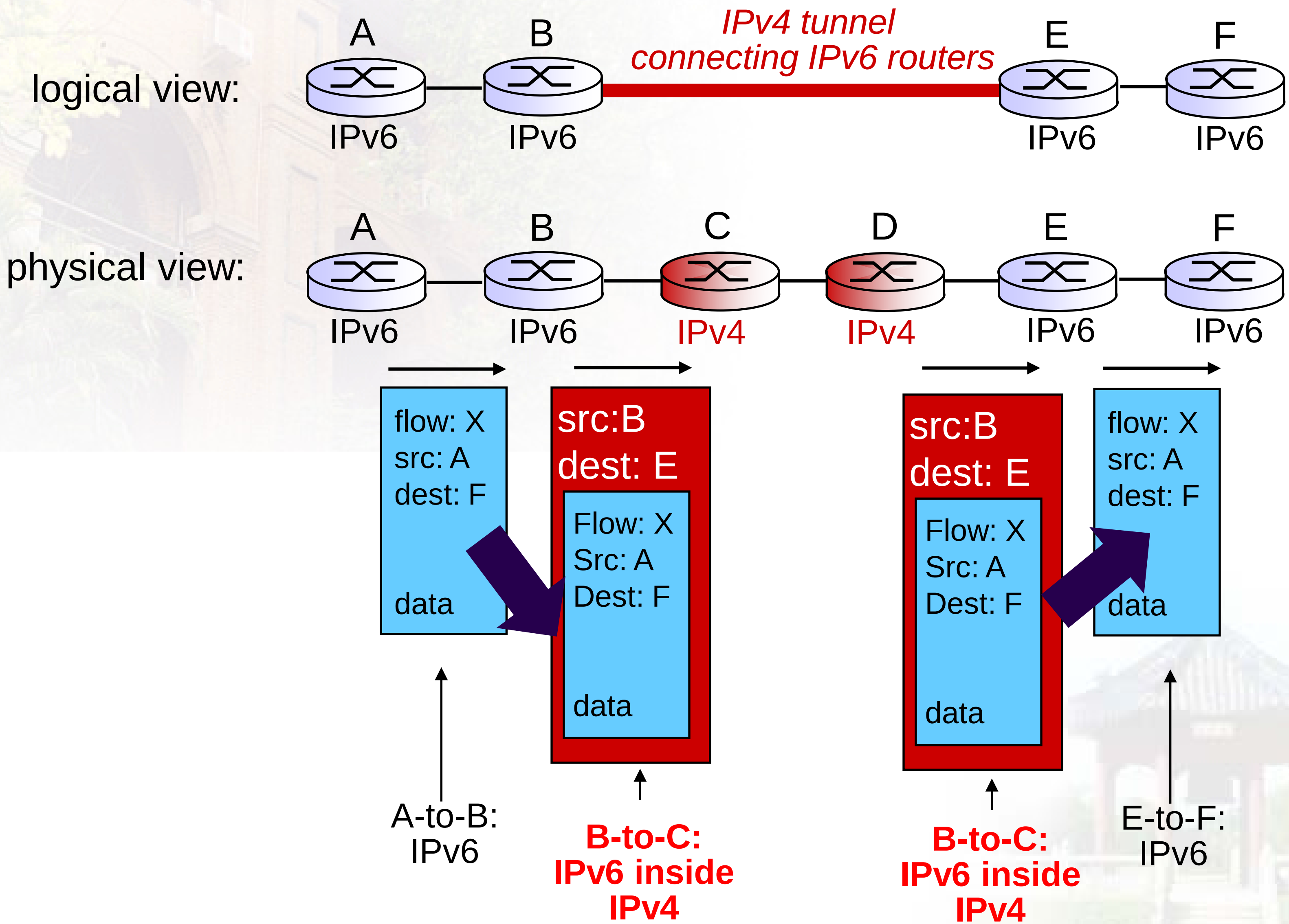


Tunneling



logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6

Tunneling



IPv4 tunnel
connecting IPv6 routers
logical view:
A
B
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

IPv6: adoption

- US National Institutes of Standards estimate [2013]:
  - ~3% of industry IP routers
  - ~11% of US gov't routers
- *Long (long!) time for deployment, use*
  - 20 years and counting!
  - think of application-level changes in last 20 years: WWW, Facebook, …
  - *Why?*

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

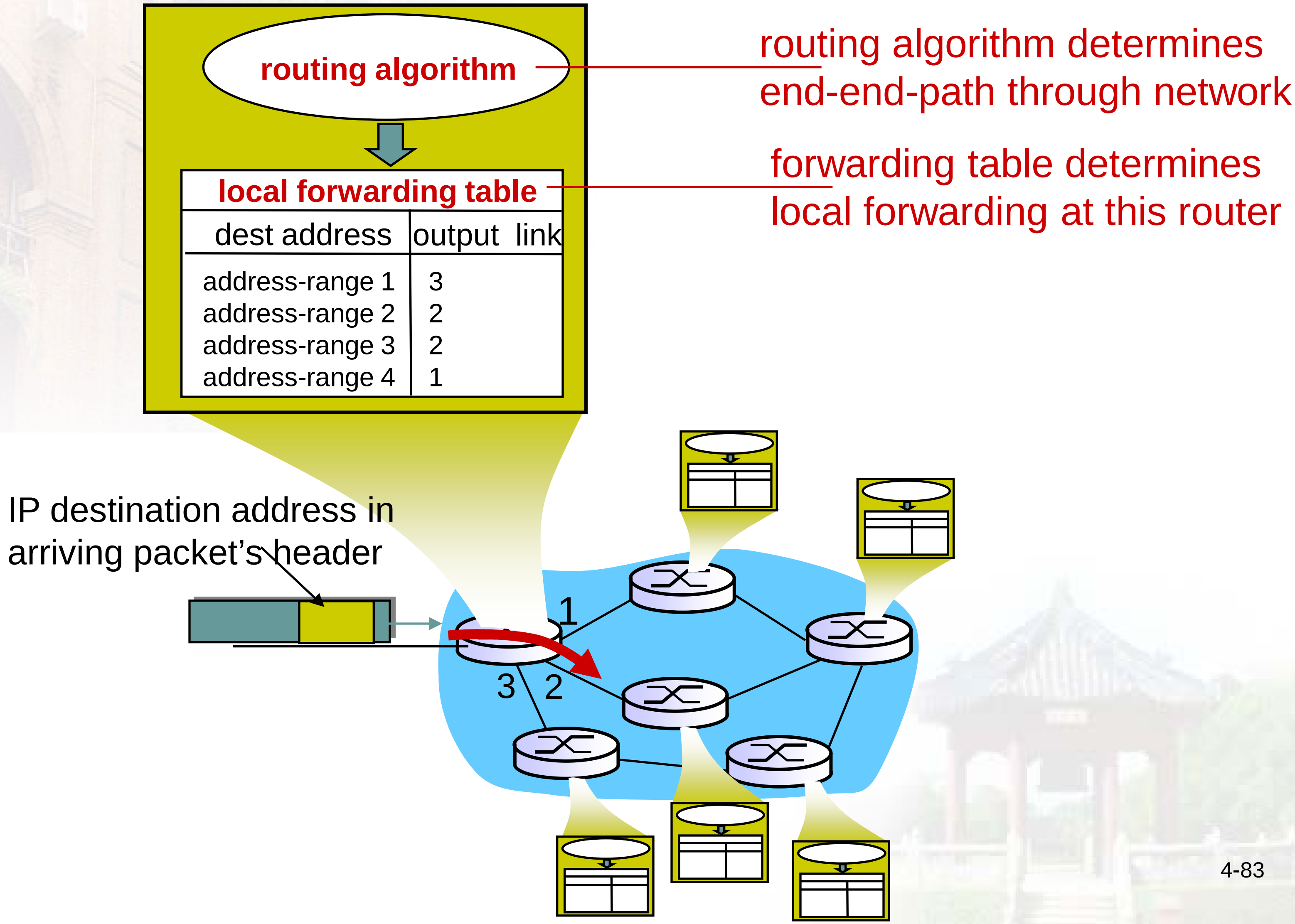
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet

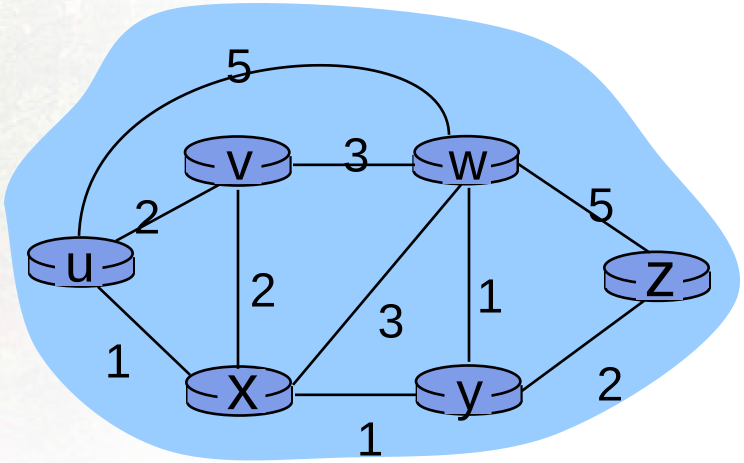
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Interplay between routing, forwarding



Graph abstraction



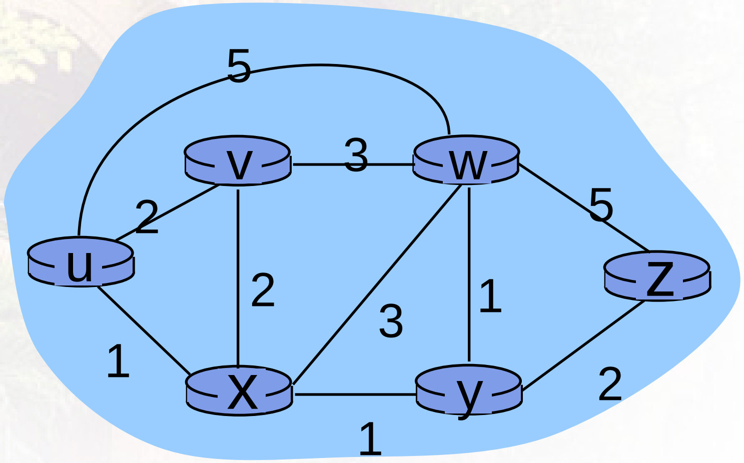
graph: G = (N,E)

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) }

aside: graph abstraction is useful in other network contexts, e.g., 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 $(x_1, x_2, x_3,\ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that finds that least cost path

Routing algorithm classification

Q: 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

Q: static or dynamic?

static:

- routes change slowly over time

dynamic:

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

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

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

Dijsktra’s Algorithm

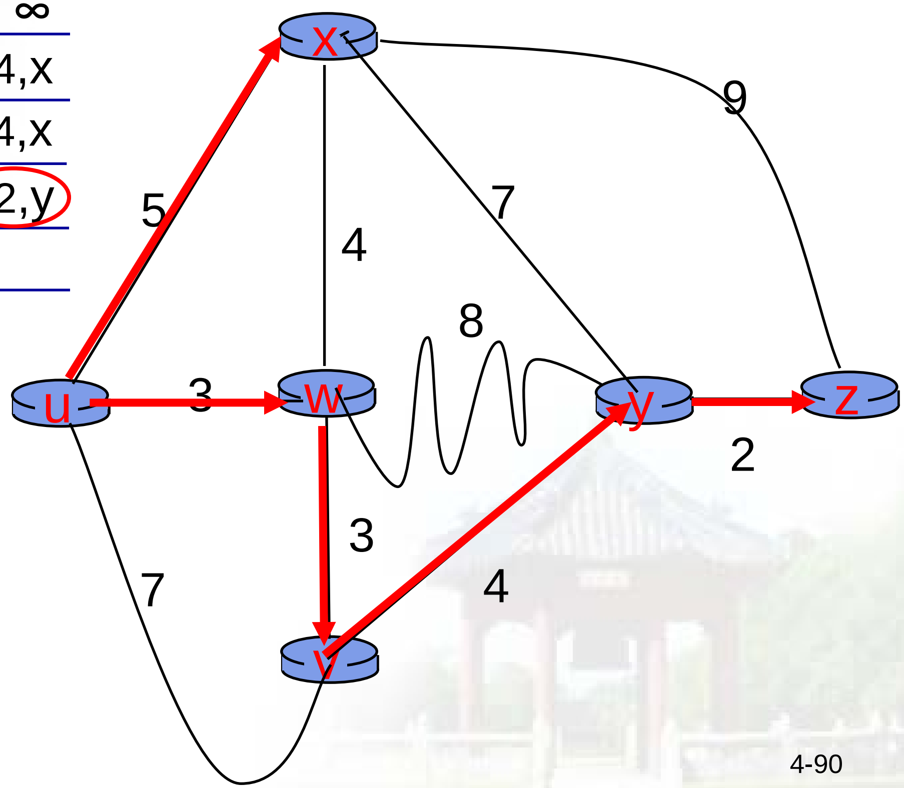
```
Initialization:
  N' = {u}
  for all nodes v
    if v adjacent to u
        then D(v) = c(u,v)
    else D(v) = ∞

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

Dijkstra's algorithm: example

Find the least cost paths between u and other nodes

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

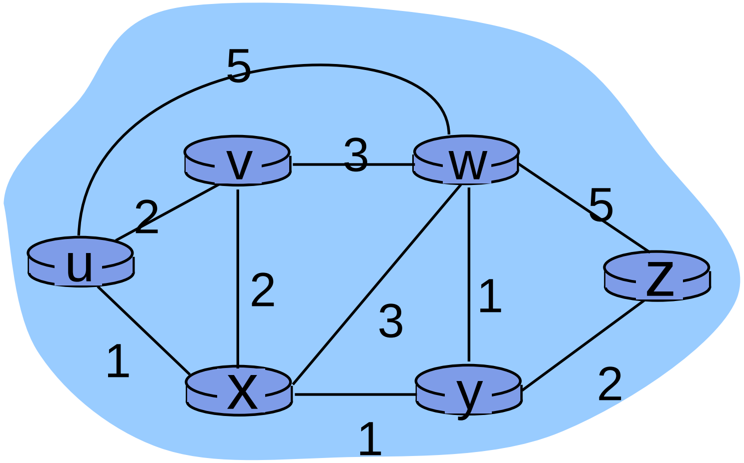


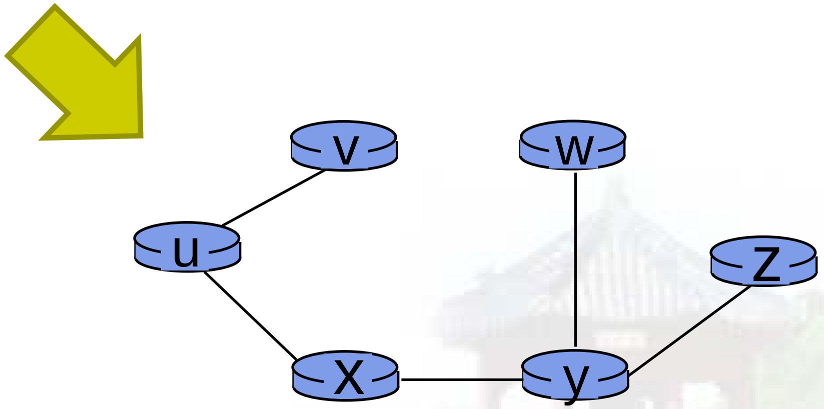
notes:

- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)

Dijkstra's algorithm: another example

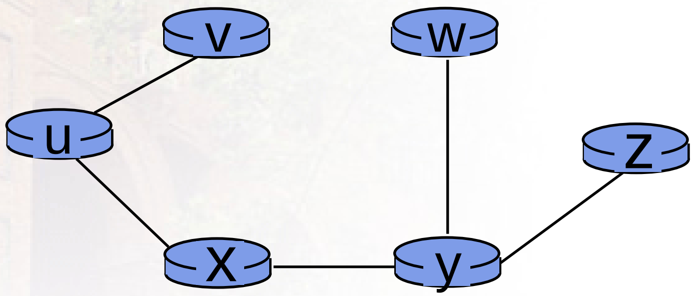
Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					





Dijkstra’s algorithm: example (2)

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)

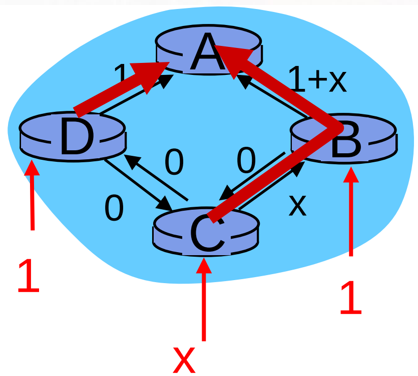
Dijkstra’s algorithm, discussion

algorithm complexity: **n nodes**

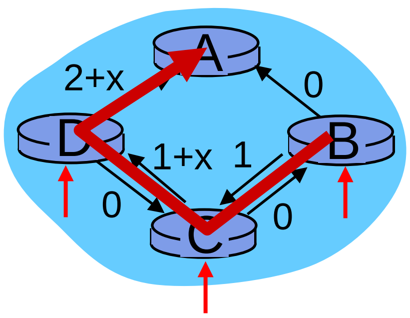
- **each iteration: need to check all nodes, w, not in N**
- **n(n+1)/2 comparisons: $O(n^2)$**
- **more efficient implementations possible: O(nlogn)**

oscillations possible:

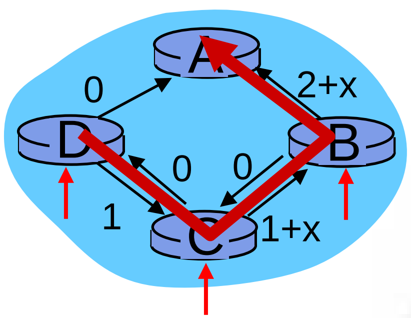
- **e.g., support link cost equals amount of carried traffic:**



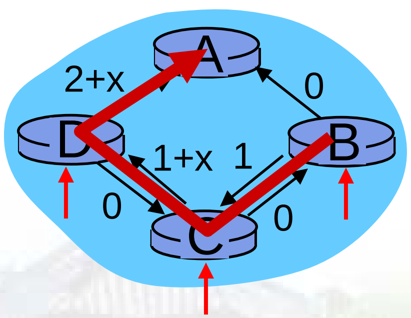
initially



given these costs, find new routing…. resulting in new costs



given these costs, find new routing…. resulting in new costs



given these costs, find new routing…. resulting in new costs

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

Distance vector algorithm

- 邻居能去到的，自己一定能通过邻居到达；
- 多个邻居能到的，找一个总代价最小的邻居作为下一跳；

Bellman-Ford equation (dynamic programming)

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

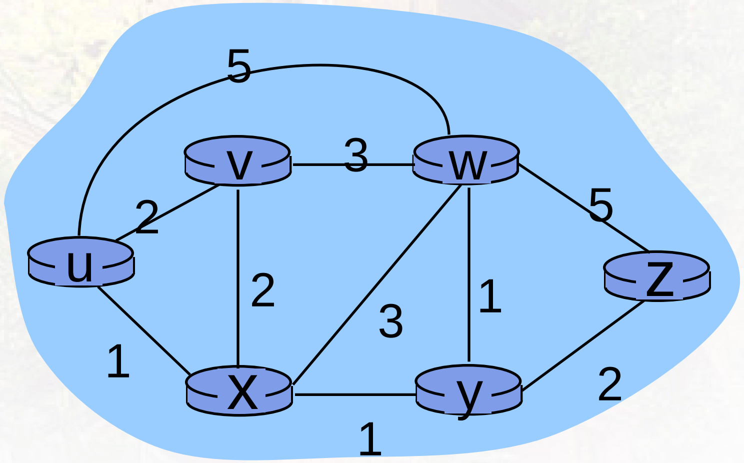
then $d_x(y) = \min_v \{c(x,v) + d_v(y)\}$

cost from neighbor v to destination y

cost to **neighbor** v

min taken over all **neighbor**s v of x

Bellman-Ford example



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

B-F equation says:

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

node achieving minimum is next
hop in shortest path, used in forwarding table

Distance vector algorithm

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

Distance vector algorithm

key idea:

- **from time-to-time, each node sends its own distance vector estimate to neighbors**
- **when 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

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

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

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

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

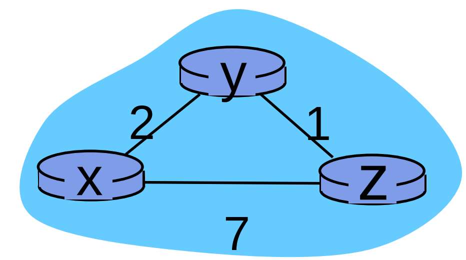
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



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

from \ 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 \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

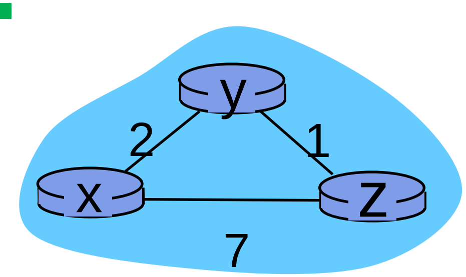
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

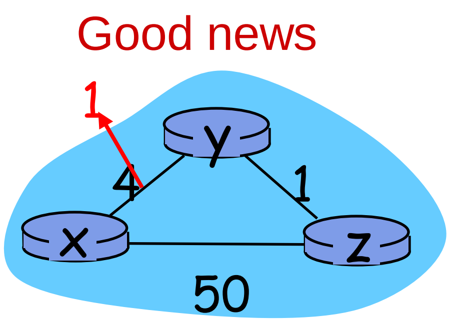


time

Distance vector: link cost changes

link cost changes:

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



"good news travels fast"

t_0 : *y* detects link-cost change, updates its DV, informs its neighbors.

t_1 : *z* receives update from *y*, updates its table, computes new least cost to *x* , sends its neighbors its DV.

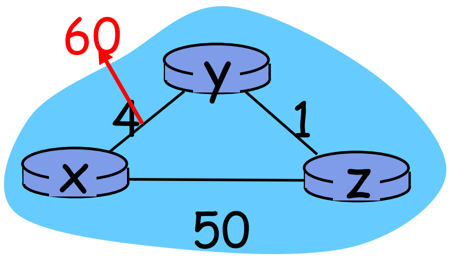
t_2 : *y* receives *z*'s update, updates its distance table. *Y*'s least costs do *not* change, so *y* does *not* send a message to *z*.

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: see text

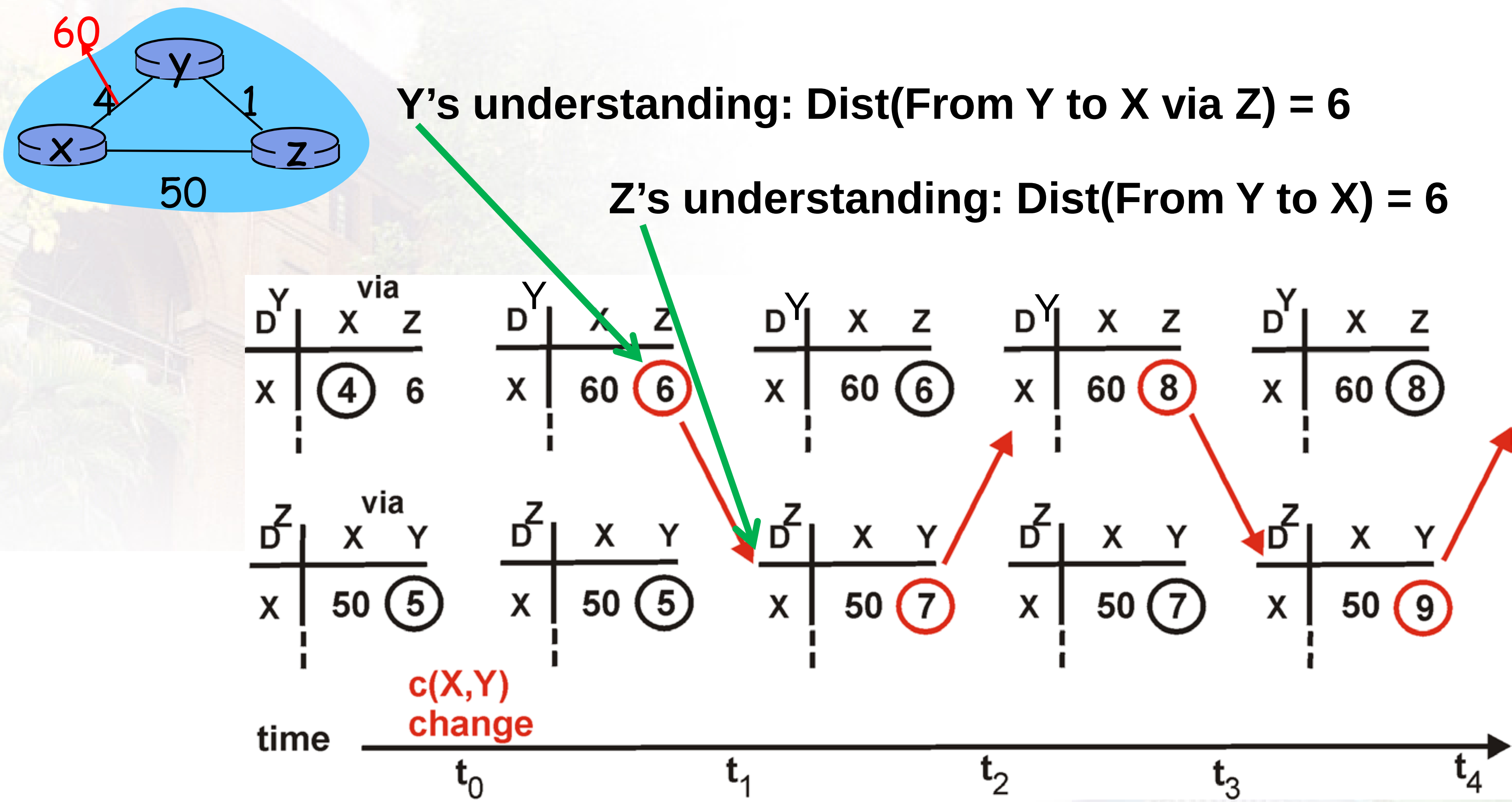
Bad news



poisoned reverse:

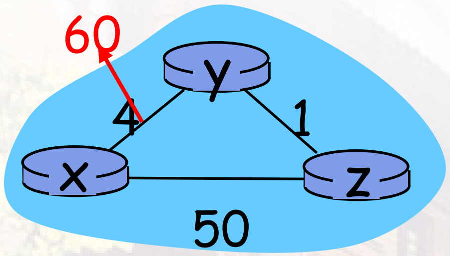
- If Z routes through Y to get to X :
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- will this completely solve count to infinity problem?

Bad news



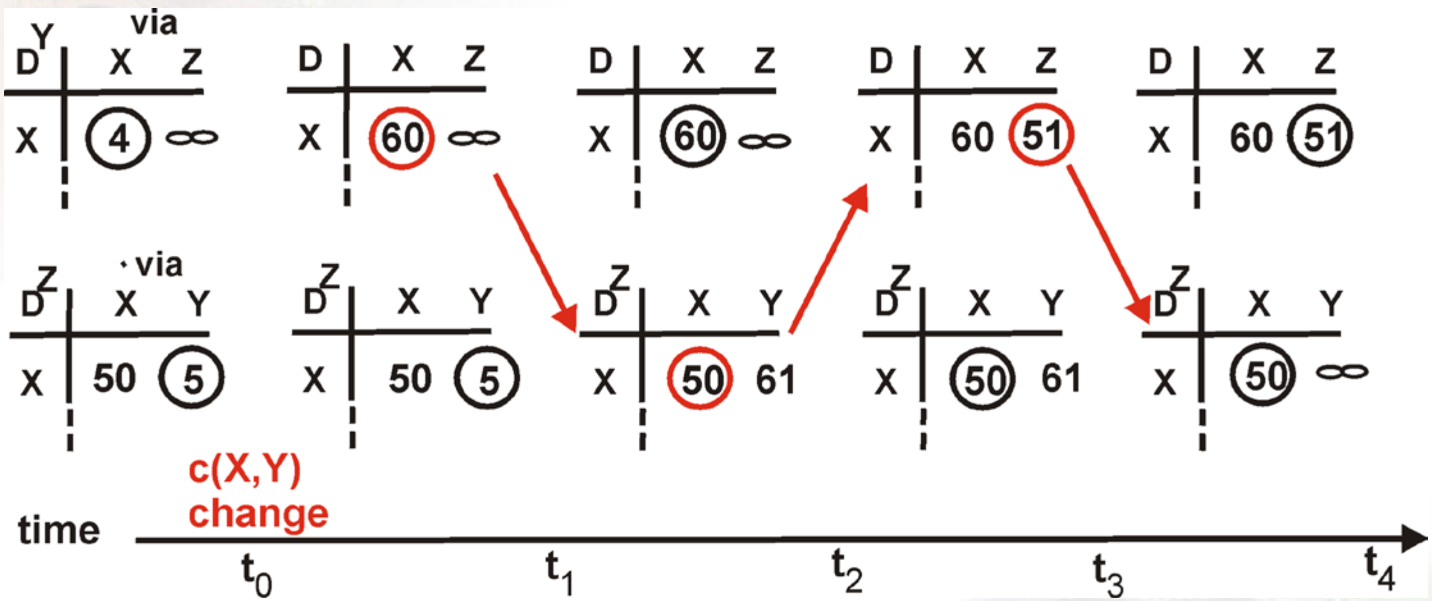
关键原因：Y没有告诉Z它是通过Z到达X

Bad news



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?



关键原因：Z知道自己通过Y到达X，因此如果遇到Y让Z转发给X的，就可以拒绝。

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**

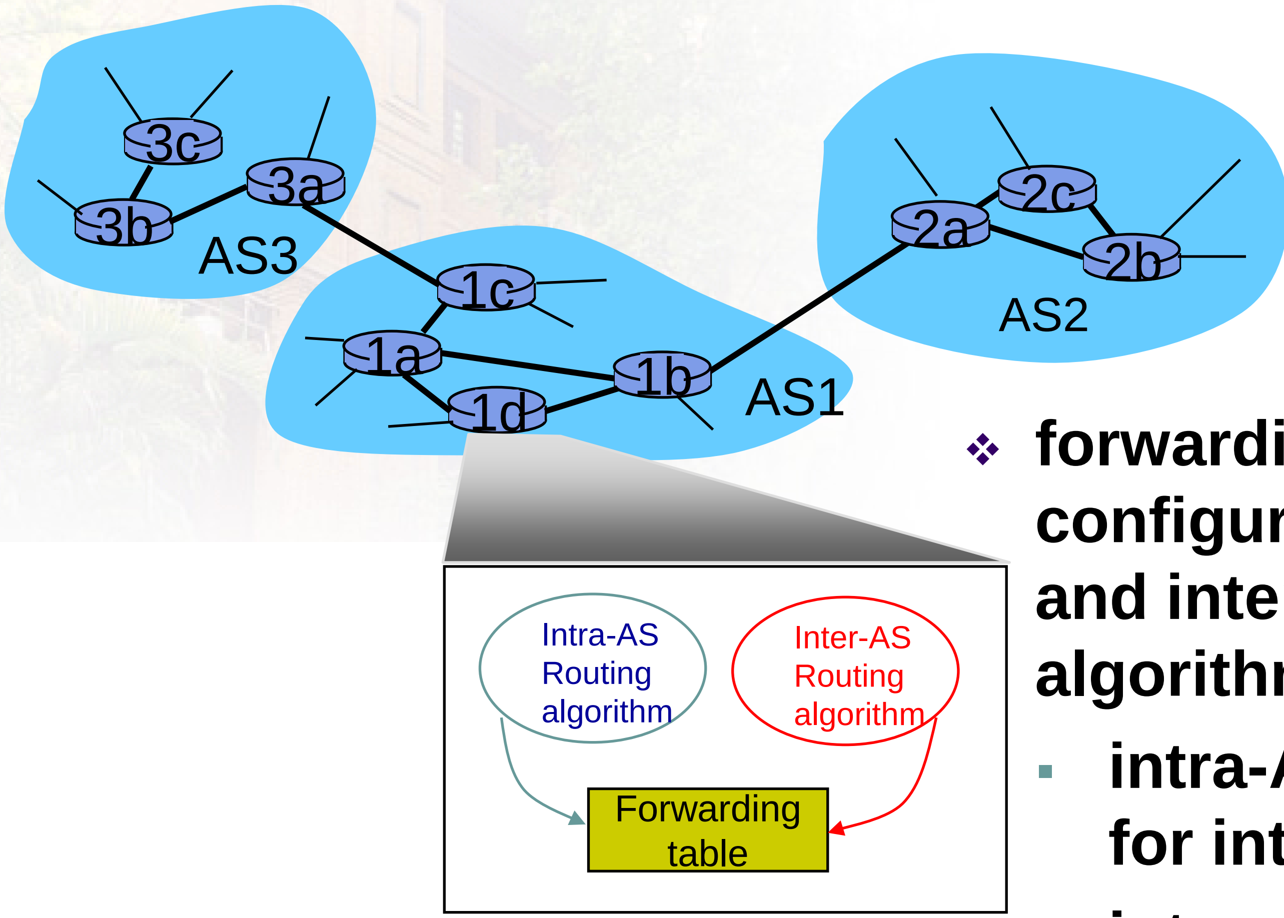
Hierarchical routing

- 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

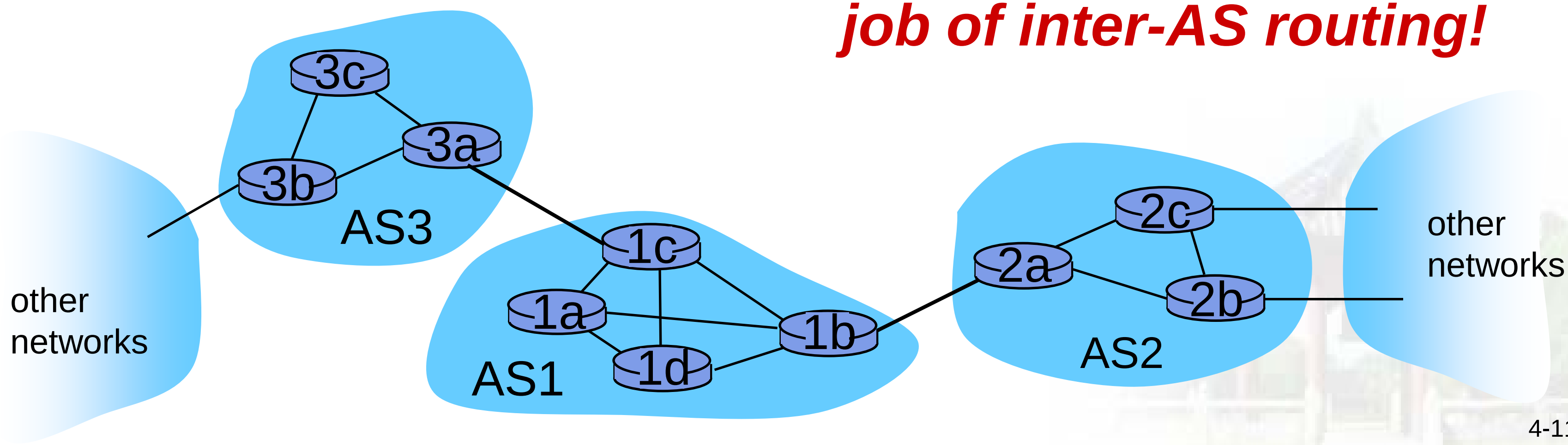
Inter-AS tasks

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

AS1 must:

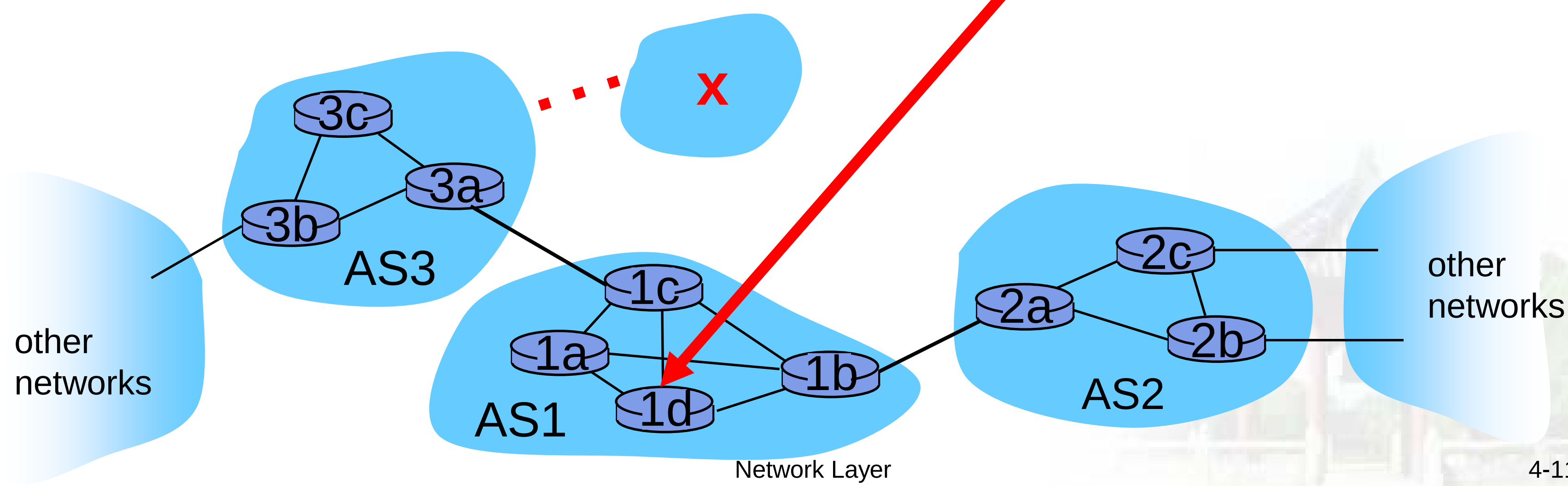
1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

job of inter-AS routing!



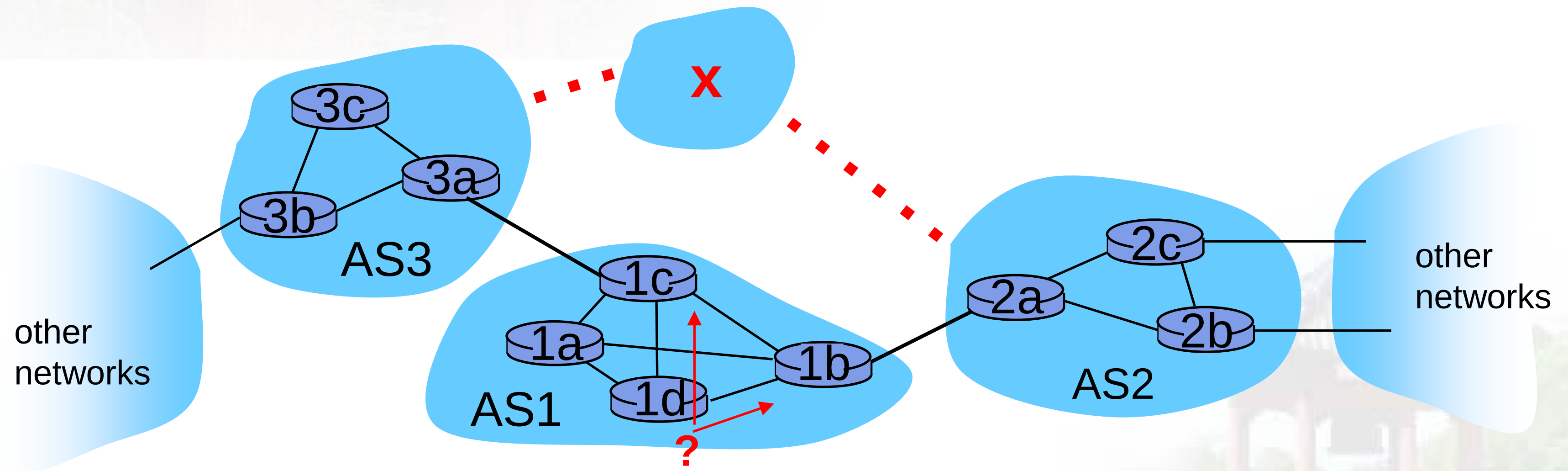
Example: setting forwarding table in router 1d

- **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!



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: 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

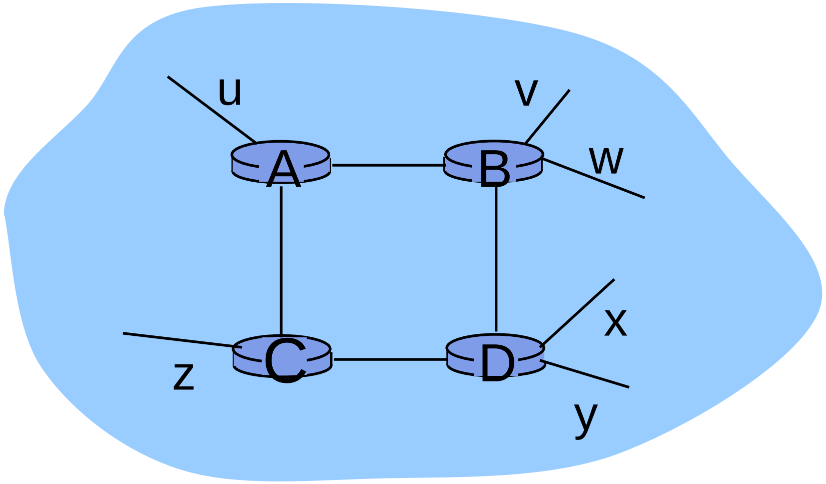
4.7 broadcast and multicast routing

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)

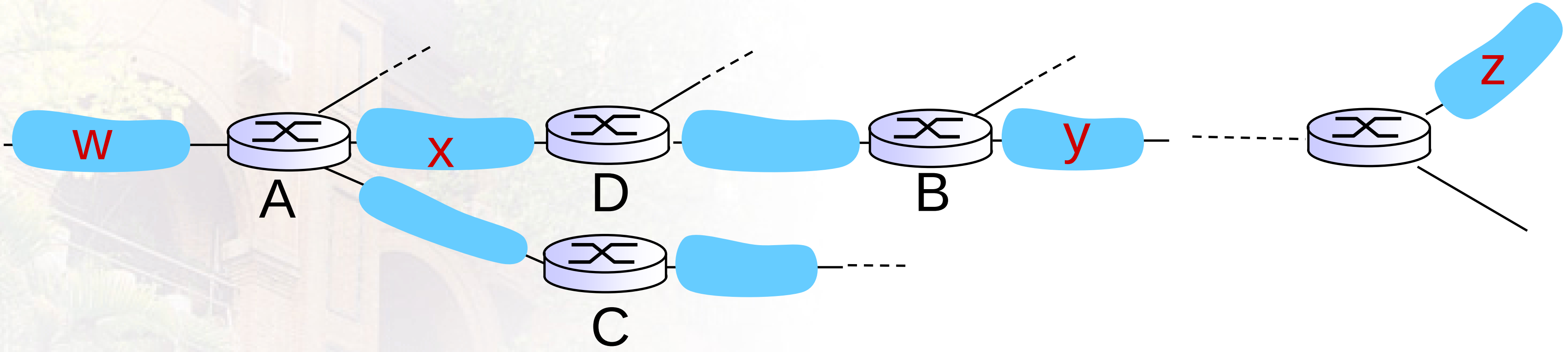
- **included in BSD-UNIX distribution in 1982**
- **distance vector algorithm**
  - **distance metric: # hops (max = 15 hops), each link has cost 1**
  - **DVs exchanged with neighbors every 30 sec in response message (aka advertisement)**
  - **each advertisement: list of up to 25 destination *subnets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



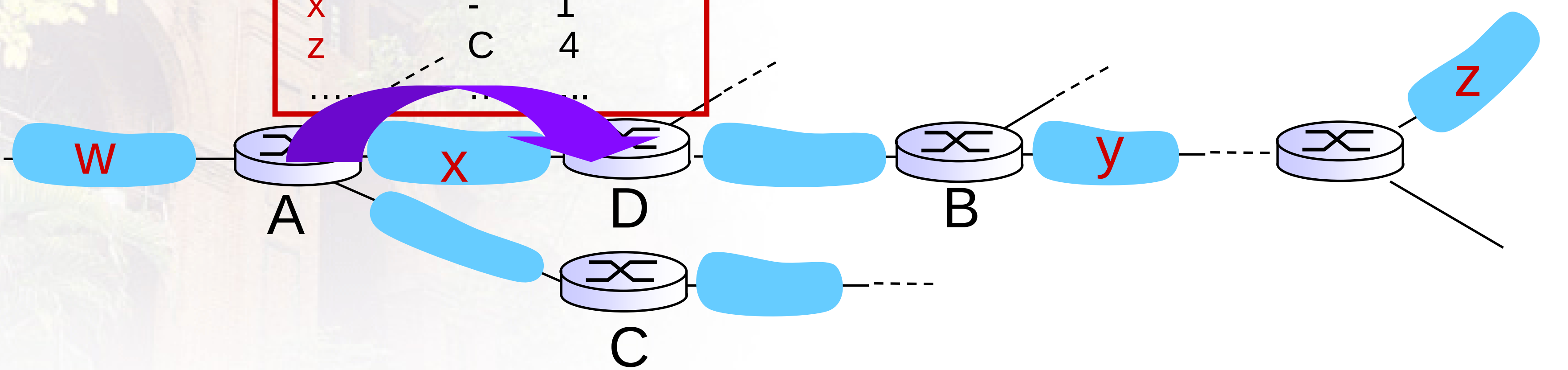
routing table in **router D**

destination subnet	next router	# hops to dest	
w	A	2	(D+A)
y	B	2	(D+B)
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
….	….	….

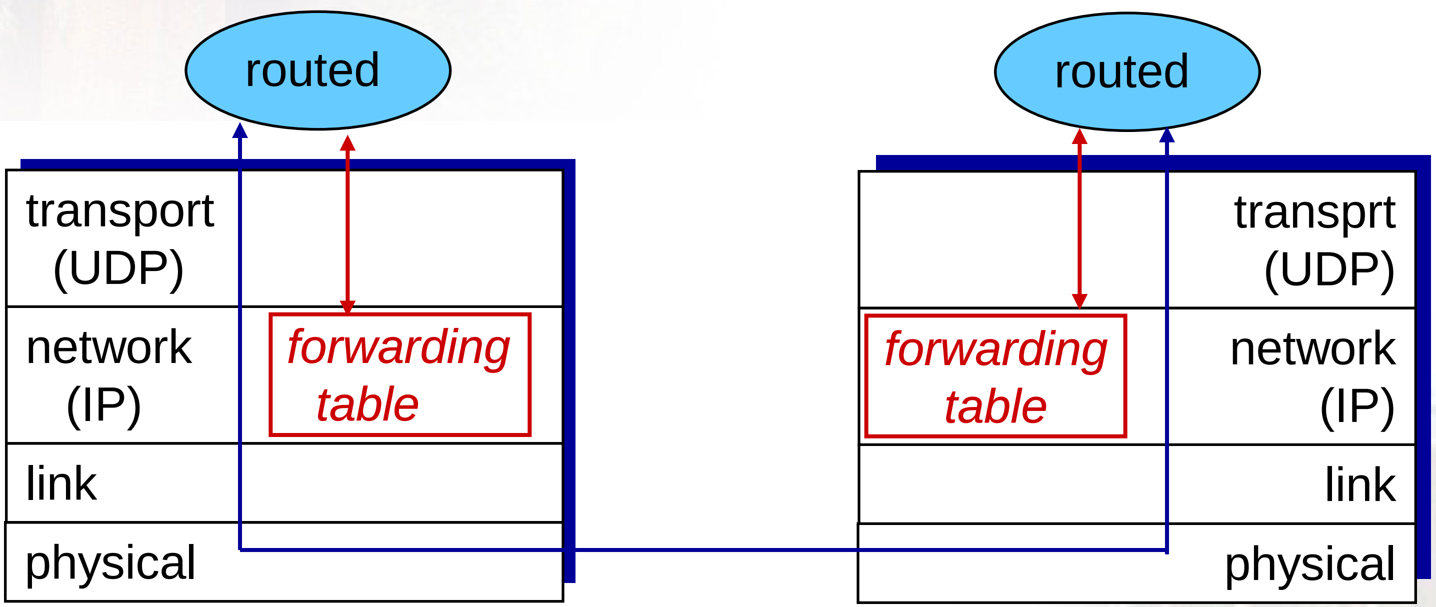
RIP: link failure, recovery

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



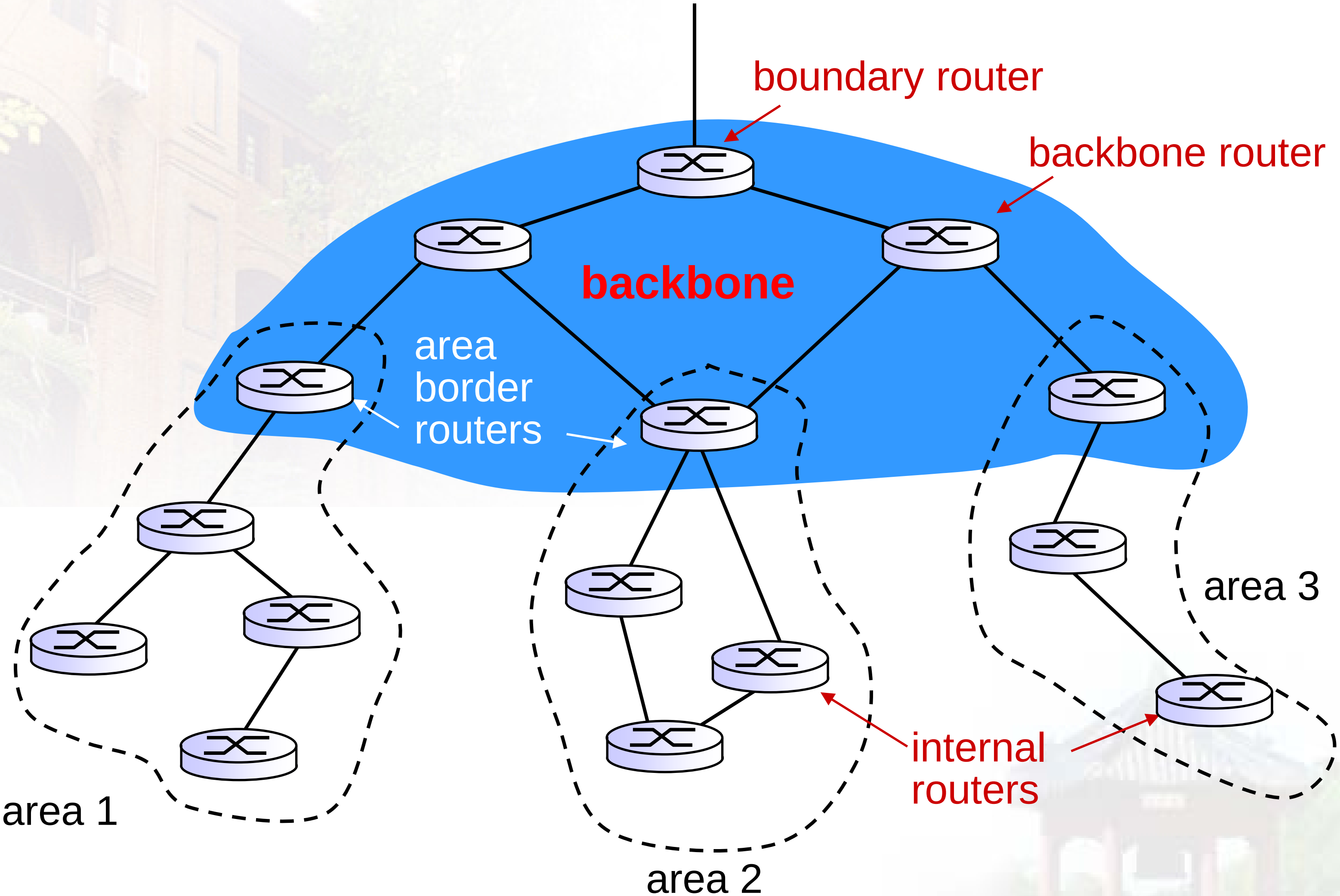
OSPF (Open Shortest Path First)

- "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.

Hierarchical OSPF



Hierarchical OSPF

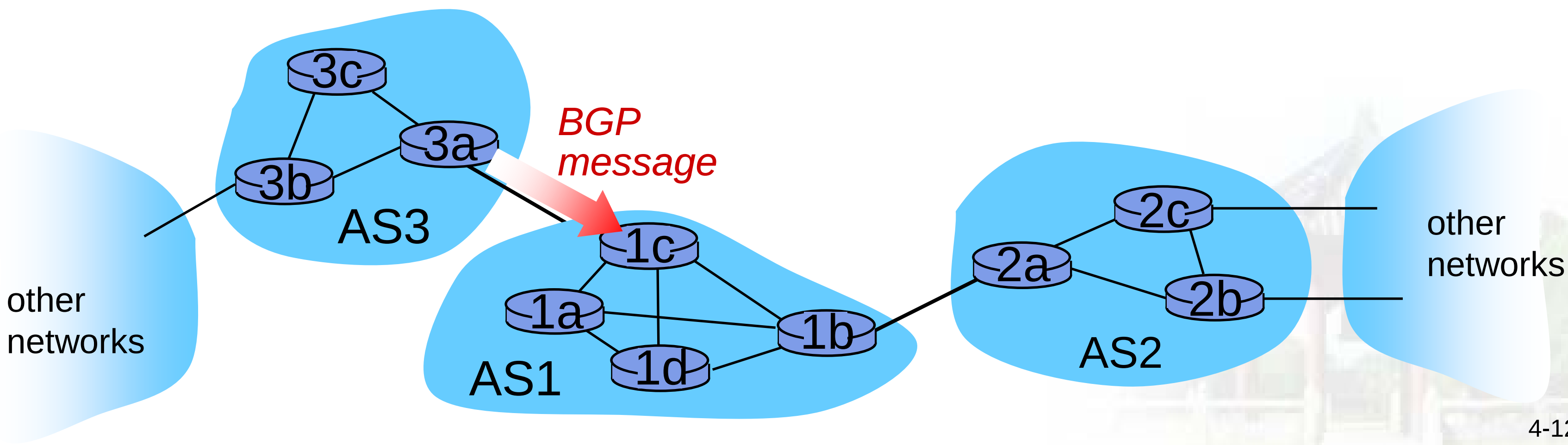
- *two-level hierarchy:* local area, backbone.
  - link-state advertisements only in area
  - each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.
- *area border routers:* "summarize" distances to nets in own area, advertise to other Area Border routers.
- *backbone routers:* run OSPF routing limited to backbone.
- *boundary routers:* connect to other AS's.

Internet inter-AS routing: BGP

- **BGP (Border Gateway Protocol):** ***the*** **de facto inter-domain routing protocol**
  - **"glue that holds the Internet together"**
- **BGP provides each AS a means to:**
  - **eBGP: obtain subnet reachability information from neighboring ASs.**
  - **iBGP: propagate AS reachability information to all AS-internal routers.**
  - **determine "good" routes to other networks based on reachability information and policy.**
- **allows subnet to advertise its existence to rest of Internet: *"I am here"***

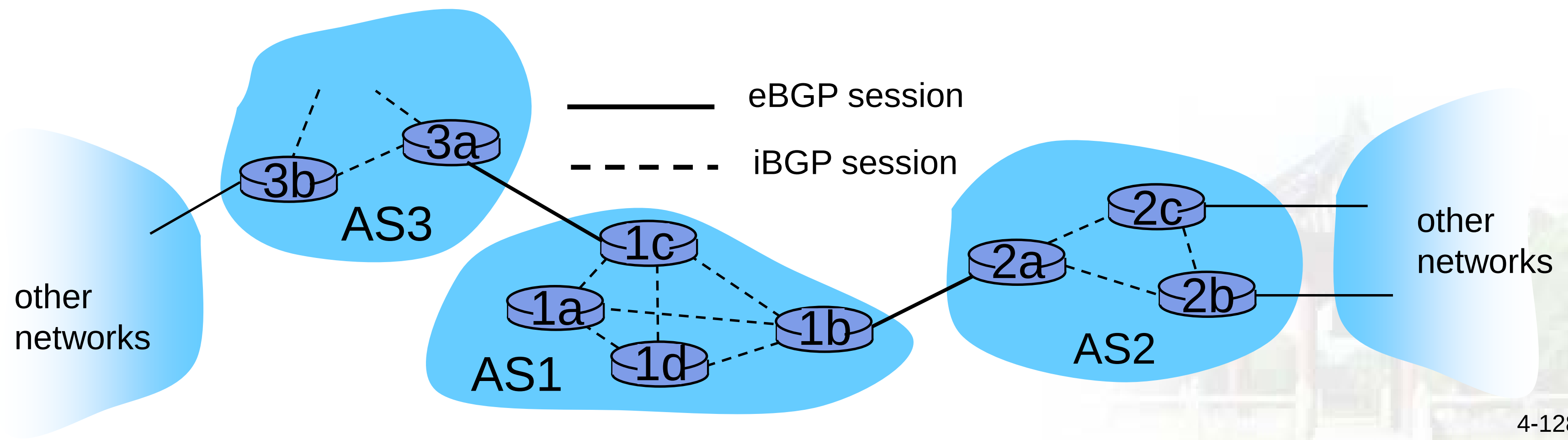
BGP basics

- **BGP session:** two BGP routers (“peers”) exchange BGP messages:
  - advertising *paths* to different destination network prefixes (“path vector” protocol)
  - exchanged over semi-permanent TCP connections
- when AS3 advertises a prefix to AS1:
  - AS3 *promises* it will forward datagrams towards that prefix
  - AS3 can aggregate prefixes in its advertisement



BGP basics: distributing path information

- **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 and 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)
- gateway router receiving route advertisement uses import policy to accept/decline
  - e.g., never route through AS x
  - *policy-based* routing

BGP route selection

- router may learn about more than 1 route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

BGP messages

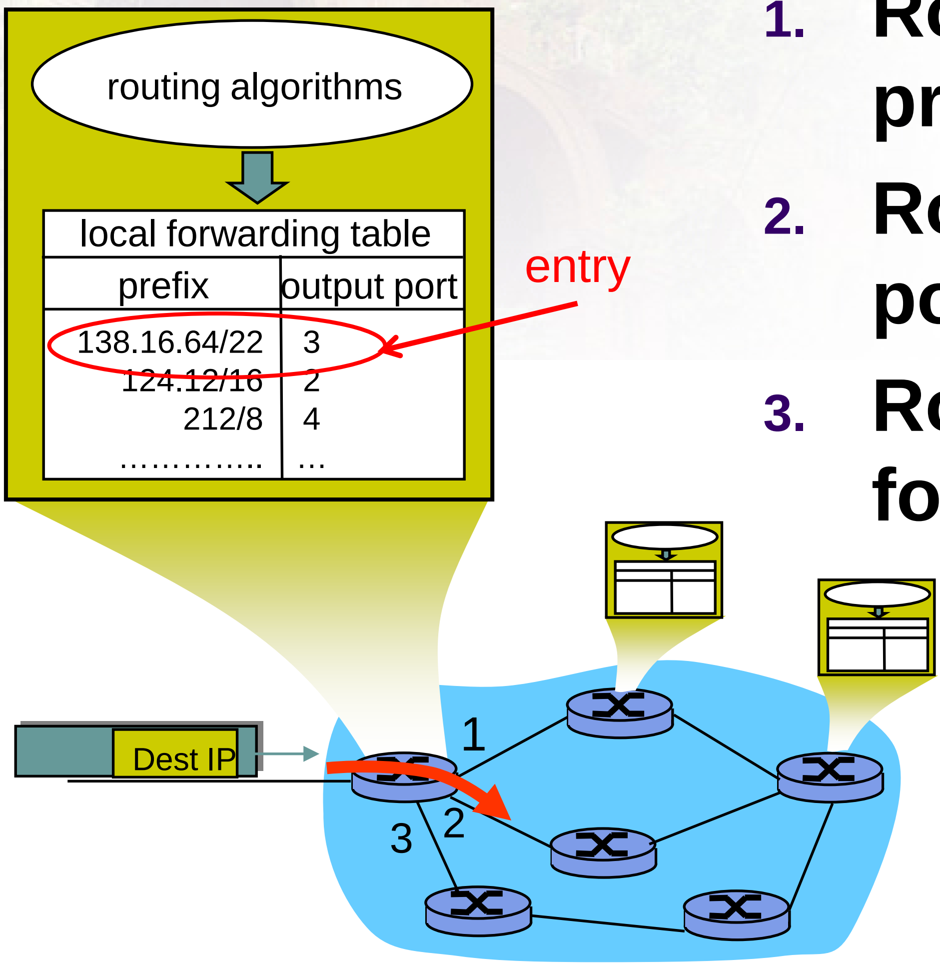
- BGP messages exchanged between peers over TCP connection
- BGP messages:
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE: keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors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

Putting it Altogether:
How Does an Entry Get Into a Router's Forwarding Table?

- **Answer is complicated!**
- **Ties together hierarchical routing (Section 4.5.3) with BGP (4.6.3) and OSPF (4.6.2).**
- **Provides nice overview of BGP!**

How does entry get in forwarding table?

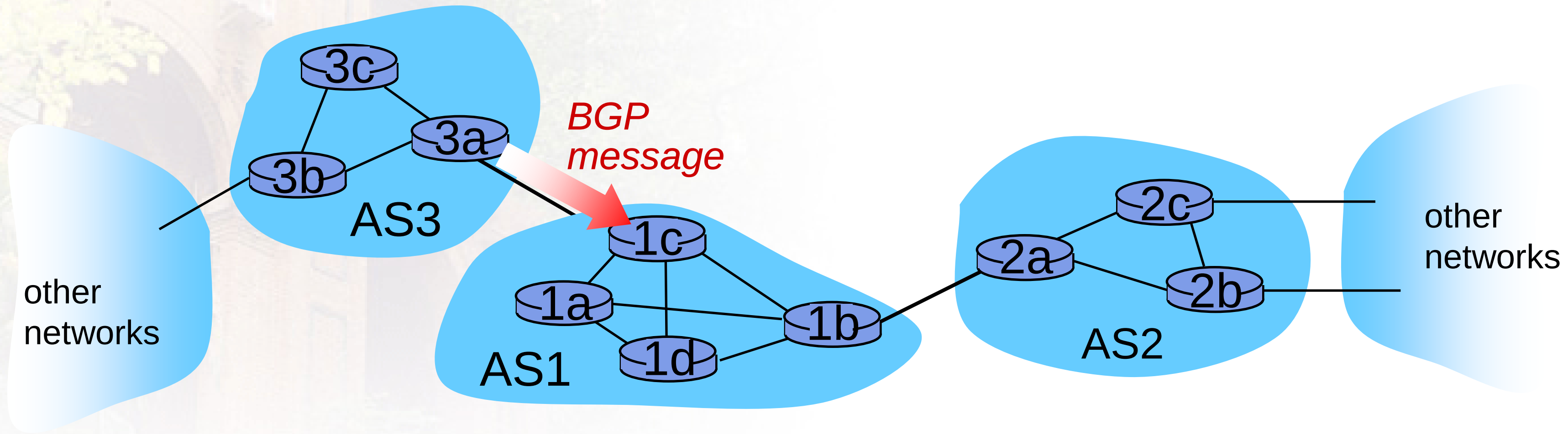
Assume prefix is in another AS.



High-level overview

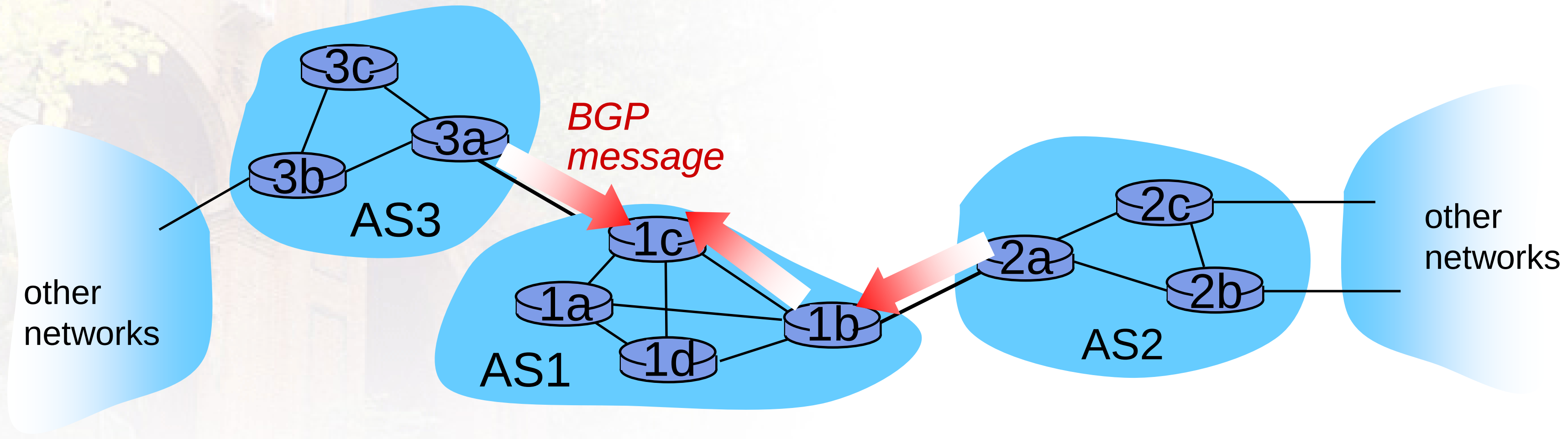
1. **Router becomes aware of prefix**
2. **Router determines output port for prefix**
3. **Router enters prefix-port in forwarding table**

Step1: Router becomes aware of prefix



- BGP message contains "routes"
- "route" is a prefix and attributes: AS-PATH, NEXT-HOP,…
- Example: route:
  - Prefix:138.16.64/22 ; AS-PATH: AS3 AS131 ; NEXT-HOP: 201.44.13.125

Step2:Router may receive multiple routes



- Router may receive multiple routes for <u>same</u> prefix
- Has to select one route

Step3:Select best BGP route to prefix

- **Router selects route based on shortest AS-PATH**

- Example:

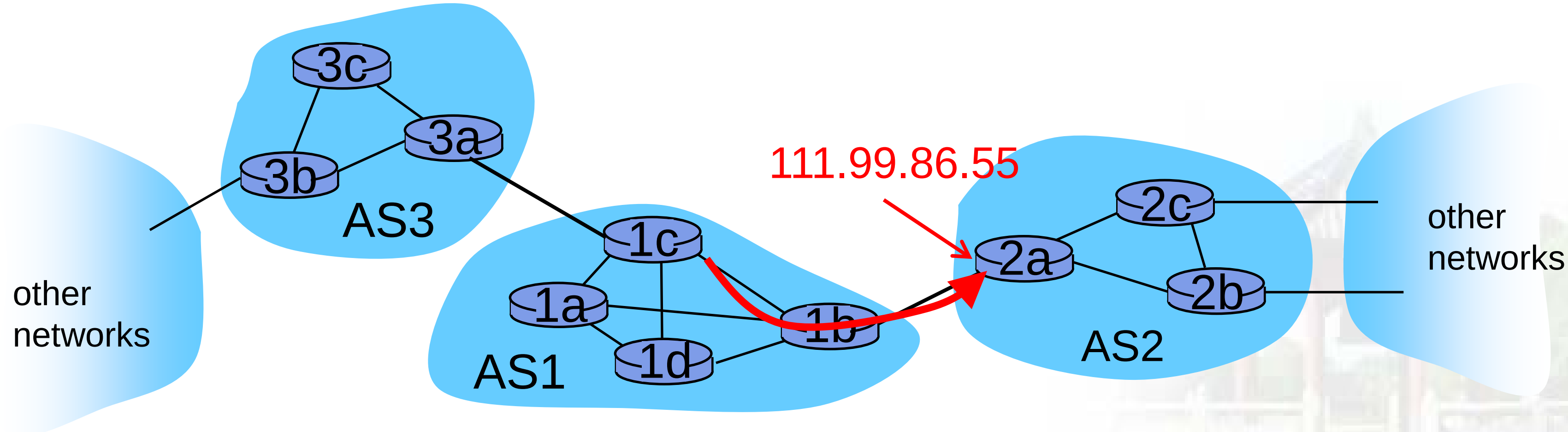
  select

  - **AS2 AS17 to 138.16.64/22**
  - **AS3 AS131 AS201 to 138.16.64/22**

- What if there is a tie? We'll come back to that!

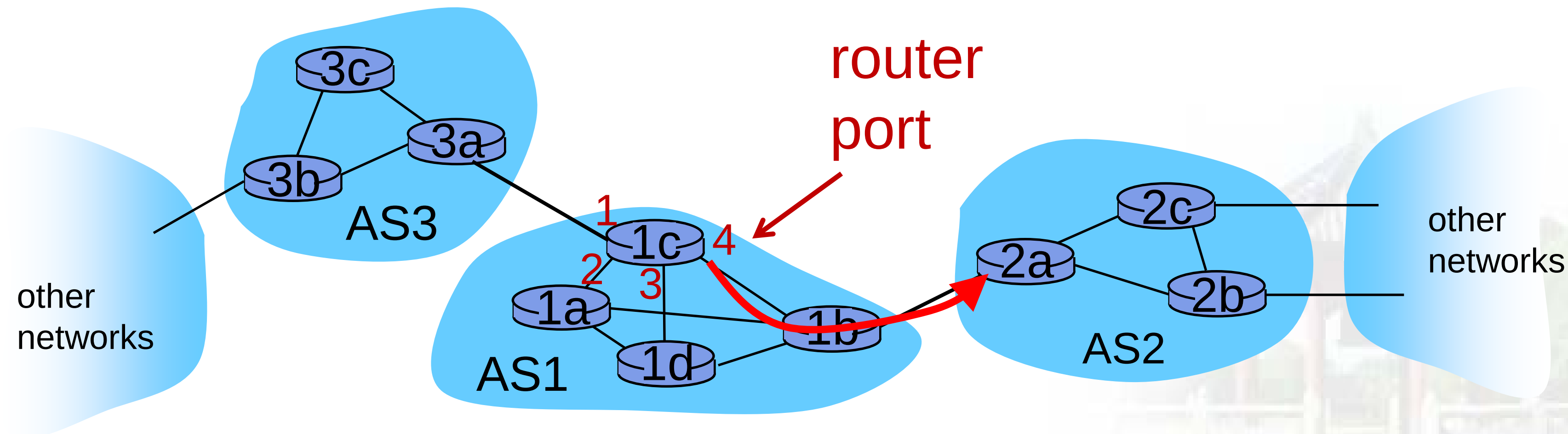
Step4:Find best intra-route to BGP route

- Use selected route's NEXT-HOP attribute
  - Route's NEXT-HOP attribute is the IP address of the router interface that begins the AS PATH.
- Example:
  - AS-PATH: AS2 AS17 ; NEXT-HOP: 111.99.86.55
- Router uses OSPF to find shortest path from 1c to 111.99.86.55



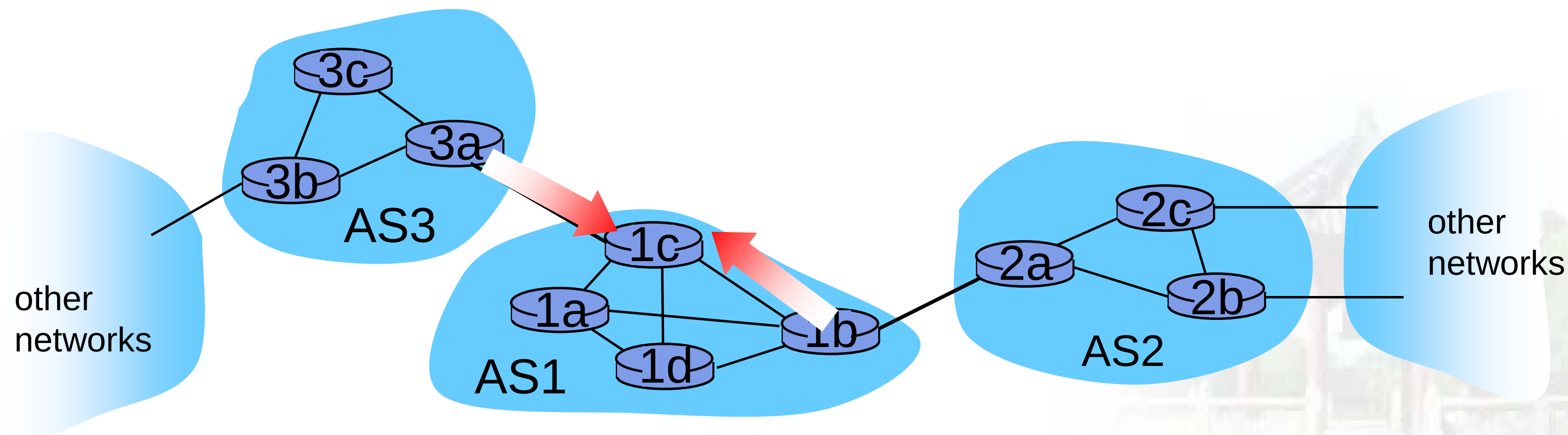
Step5:Router identifies port for route

- **Identifies port along the OSPF shortest path**
- **Adds prefix-port entry to its forwarding table:**
  - **(138.16.64/22 , port 4)**



Hot Potato Routing

- **Suppose there two or more best inter-routes.**
- **Then choose route with closest NEXT-HOP**
  - **Use OSPF to determine which gateway is closest**
  - **Q: From 1c, chose AS3 AS131 or AS2 AS17?**
  - **A: route AS3 AS201 since it is closer**

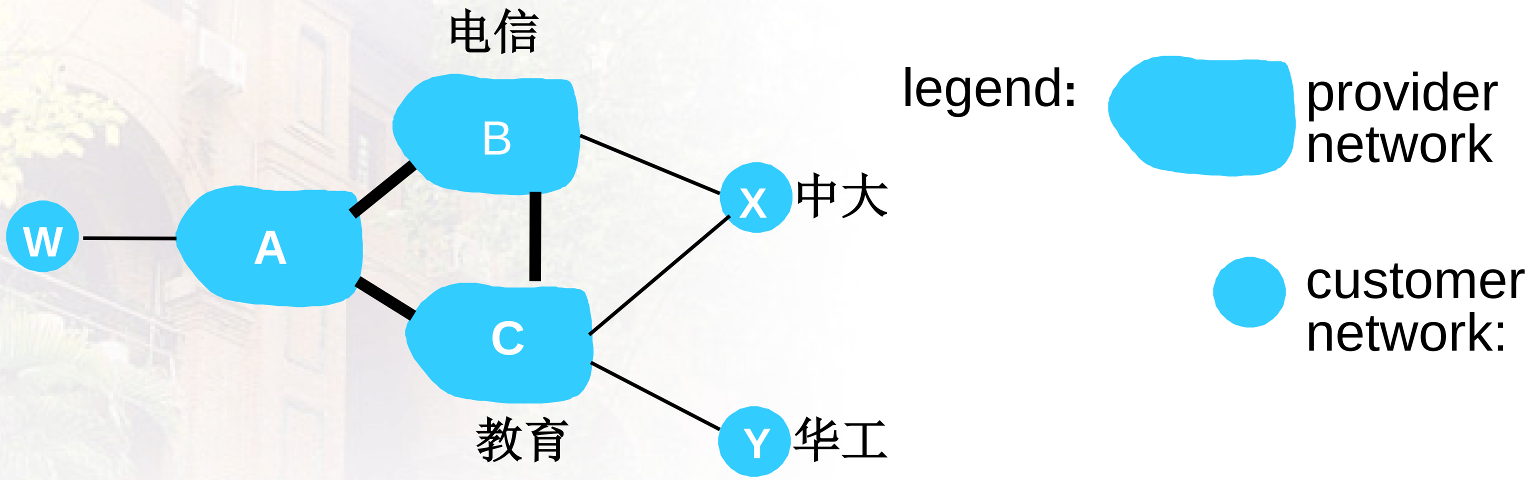


How does entry get in forwarding table?

Summary

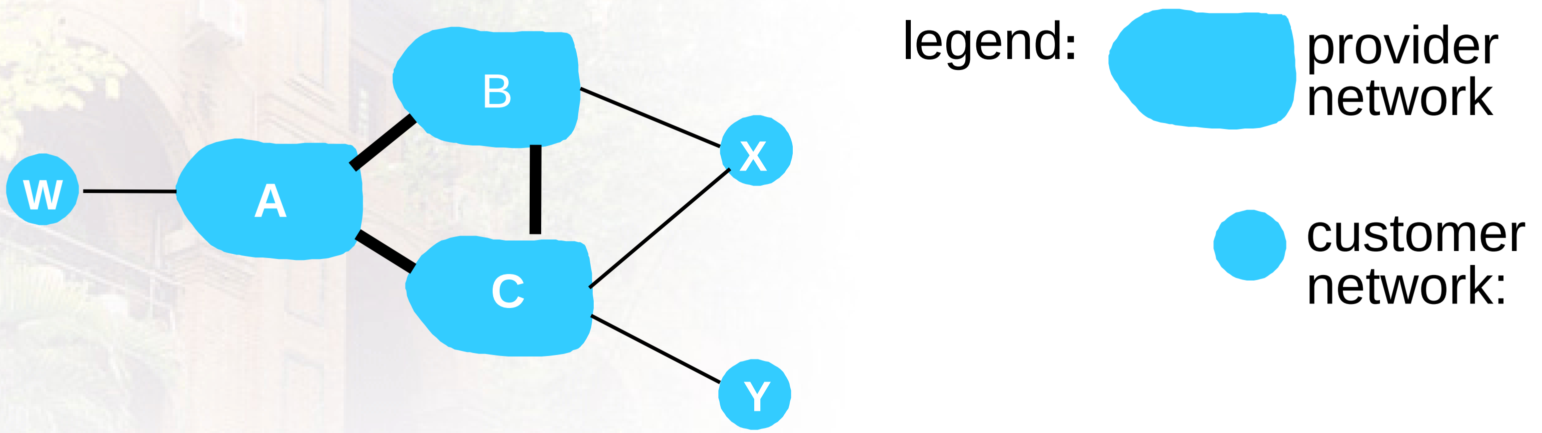
1. **Router becomes aware of prefix**
   - via BGP route advertisements from other routers
2. **Determine router output port for prefix**
   - Use BGP route selection to find best inter-AS route
   - Use OSPF to find best intra-AS route leading to best inter-AS route
   - Router identifies router port for that best route
3. **Enter prefix-port entry in forwarding table**

BGP routing policy (1)



- 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)



- 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-, 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: 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

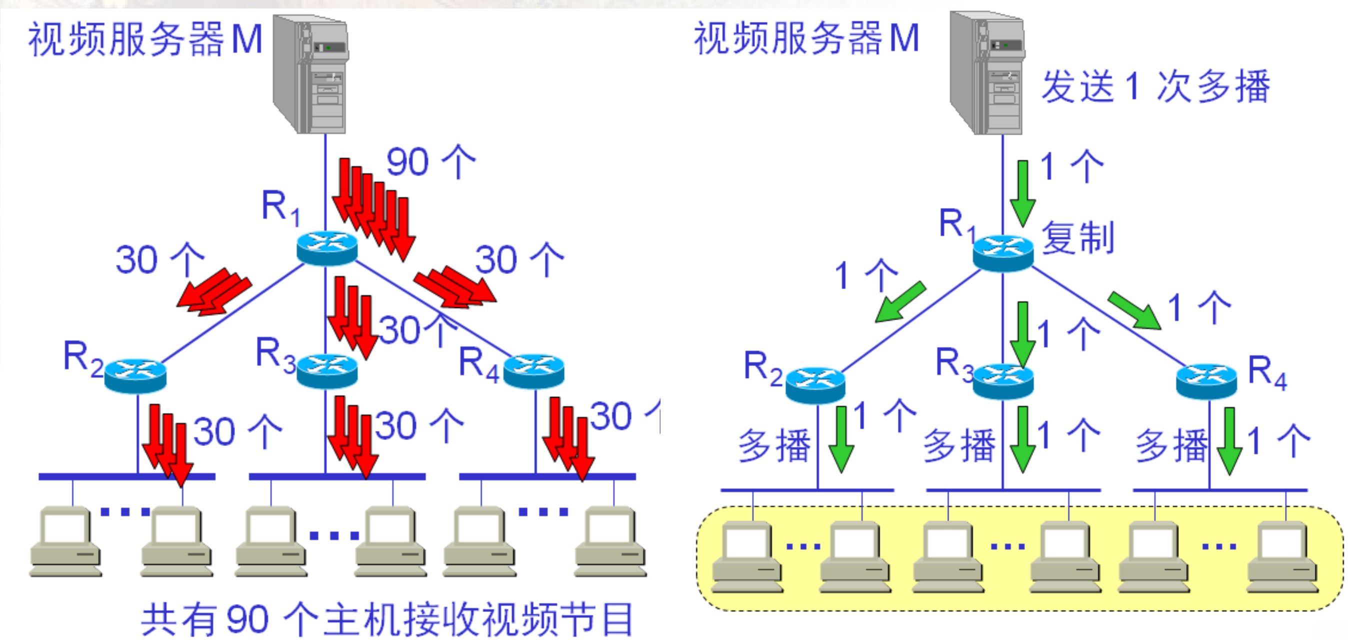
4.6 routing in the Internet

- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Broadcast routing

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



One-to-many communication: video conference, Stock market, news,…

- source duplication: how does source (R1) determine recipient addresses (R3,R4) ?

IP 多播的一些特点

(1) 多播使用组地址—— **IP** 使用 **D** 类地址支持多播。多播地址只能用于目的地址，而不能用于源地址。

(2) 永久组地址——由因特网号码指派管理局 **IANA** 负责指派。

(3) 动态的组成员

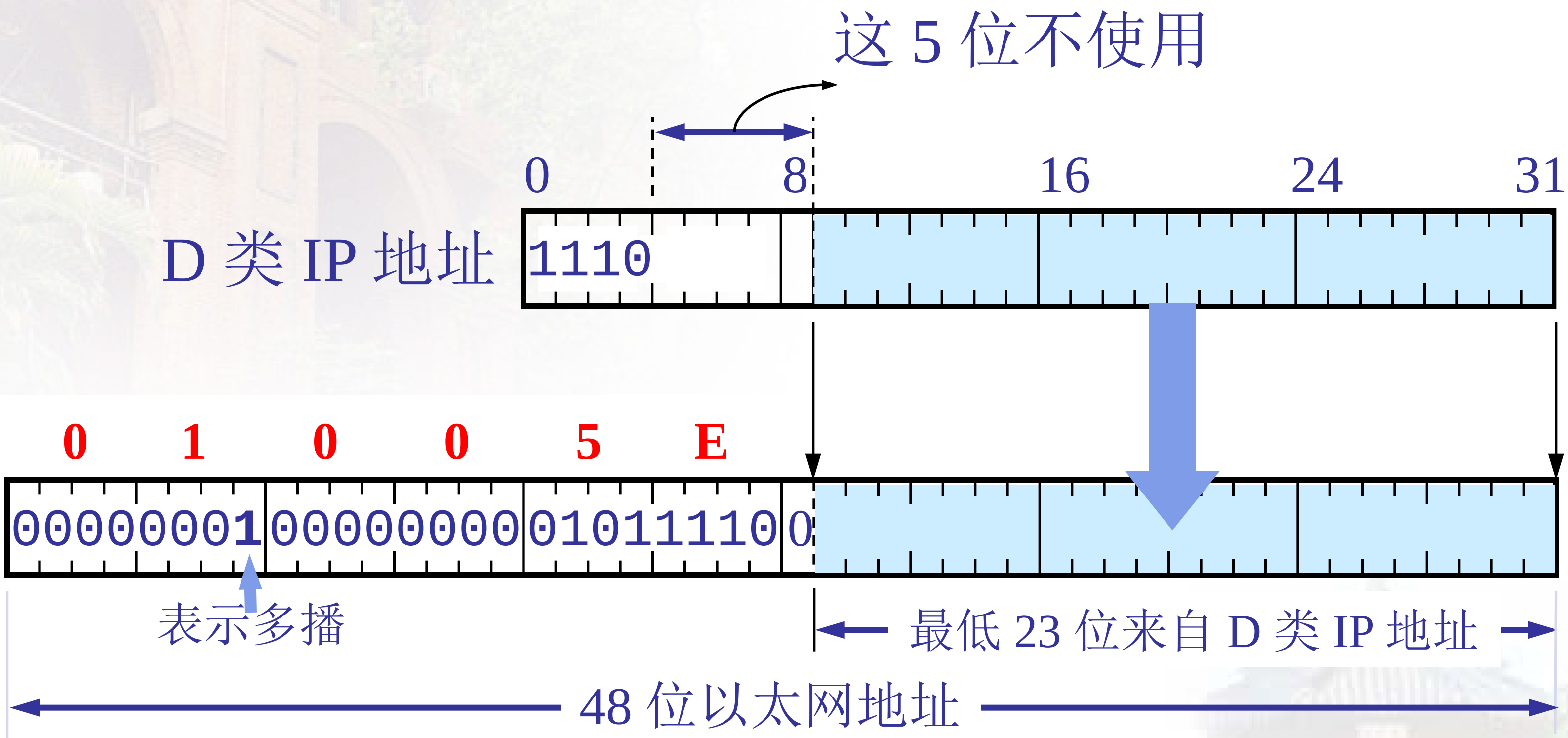
(4) 使用硬件进行多播

在局域网上进行硬件多播

- 因特网号码指派管理局 IANA 拥有的以太网地址块的高 24 位为 00-00-5E。
- 因此 TCP/IP 协议使用的以太网多播地址块的范围是：从 00-00-5E-00-00-00

  到 00-00-5E-FF-FF-FF
- D 类 IP 地址可供分配的有 28 位，在这 28 位中的前 5 位不能用来构成以太网硬件地址。

D类IP地址
与以太网多播地址的映射关系

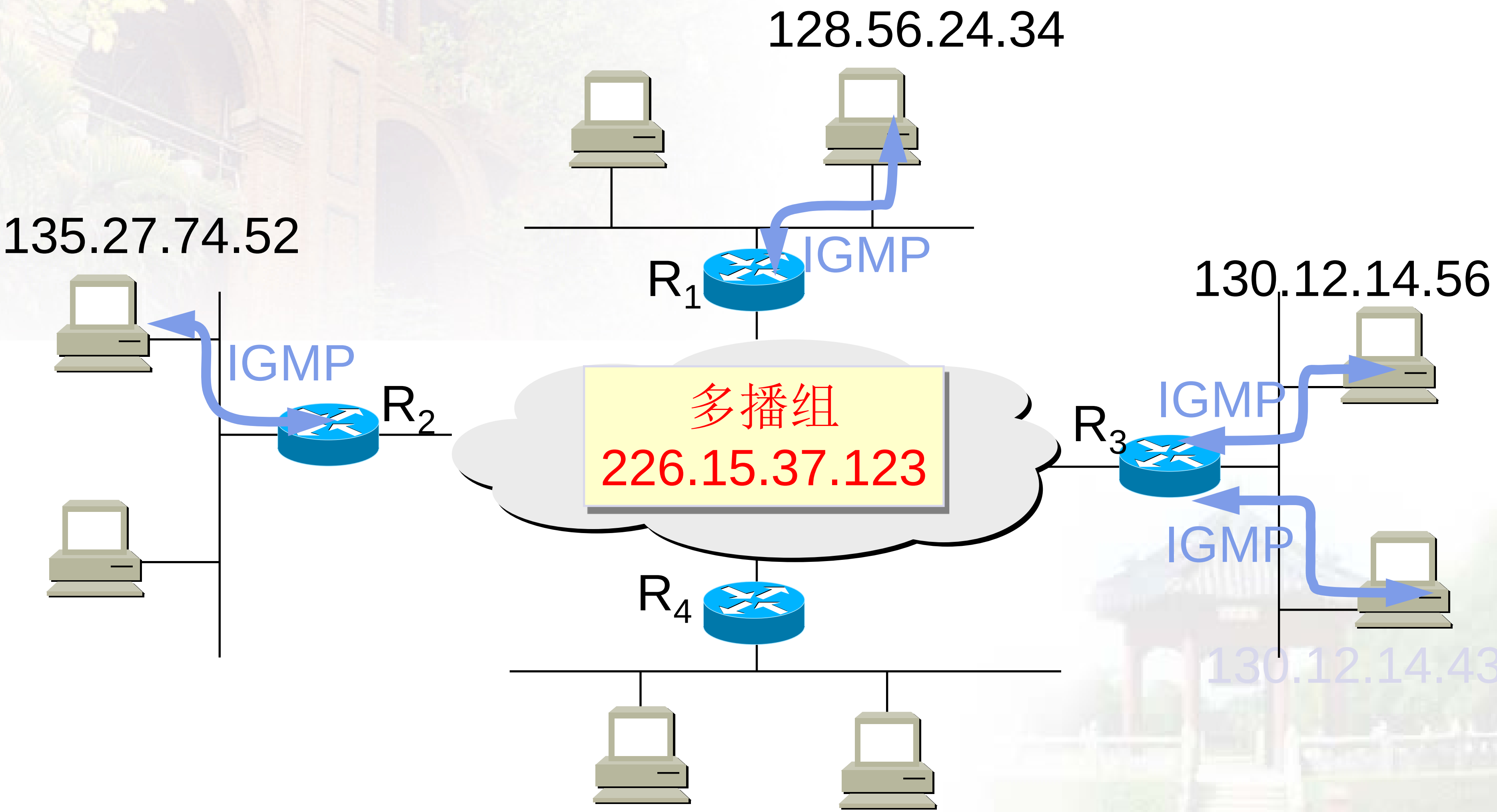


网际组管理协议 IGMP 和多播路由选择协议

1. IP多播需要两种协议

- 为了使路由器知道多播组成员的信息，需要利用网际组管理协议 IGMP (Internet Group Management Protocol)。
- 连接在局域网上的多播路由器还必须和因特网上的其他多播路由器协同工作，以便把多播数据报用最小代价传送给所有的组成员。这就需要使用多播路由选择协议。

IGMP 使多播路由器知道多播组成员信息



IGMP 的本地使用范围

- IGMP 并非在因特网范围内对所有多播组成员进行管理的协议。
- IGMP 不知道 IP 多播组包含的成员数，也不知道这些成员都分布在哪些网络上。
- IGMP 协议是让连接在本地局域网上的多播路由器知道本局域网上是否有主机（严格讲，是主机上的某个进程）参加或退出了某个多播组。

多播路由选择协议
比单播路由选择协议复杂得多

- 多播转发必须动态地适应多播组成员的变化（这时网络拓扑并未发生变化）。请注意，单播路由选择通常是在网络拓扑发生变化时才需要更新路由。
- 多播路由器在转发多播数据报时，不能仅仅根据多播数据报中的目的地址，而是还要考虑这个多播数据报从什么地方来和要到什么地方去。
- 多播数据报可以由没有加入多播组的主机发出，也可以通过没有组成员接入的网络。

2. 网际组管理协议 IGMP

- 1989 年公布的 RFC 1112（IGMPv1）早已成为了因特网的标准协议。
- 1997 年公布的 RFC 2236（IGMPv2，建议标准）对 IGMPv1 进行了更新。
- 2002 年 10 月公布了 RFC 3376（IGMPv3，建议标准），宣布 RFC 2236（IGMPv2）是陈旧的。

IGMP 是整个网际协议 IP 的一个组成部分

- 和 ICMP 相似，IGMP 使用 IP 数据报传递其报文（即 IGMP 报文加上 IP 首部构成 IP 数据报），但它也向 IP 提供服务。
- 因此，我们不把 IGMP 看成是一个单独的协议，而是属于整个网际协议 IP 的一个组成部分。

IGMP 可分为两个阶段

- 第一阶段：当某个主机加入新的多播组时，该主机应向多播组的多播地址发送IGMP 报文，声明自己要成为该组的成员。本地的多播路由器收到 IGMP 报文后，将组成员关系转发给因特网上的其他多播路由器。

IGMP 可分为两个阶段

- 第二阶段：因为组成员关系是动态的，因此本地多播路由器要周期性地探询本地局域网上的主机，以便知道这些主机是否还继续是组的成员。
- 只要对某个组有一个主机响应，那么多播路由器就认为这个组是活跃的。
- 但一个组在经过几次的探询后仍然没有一个主机响应，则不再将该组的成员关系转发给其他的多播路由器。

IGMP 采用的一些具体措施

- 在主机和多播路由器之间的所有通信都是使用 IP 多播。
- 多播路由器在探询组成员关系时，只需要对所有的组发送一个请求信息的询问报文，而不需要对每一个组发送一个询问报文。默认的询问速率是每 125 秒发送一次。
- 当同一个网络上连接有几个多播路由器时，它们能够迅速和有效地选择其中的一个来探询主机的成员关系。

IGMP 采用的一些具体措施（续）

- 在 IGMP 的询问报文中有一个数值 *N*，它指明一个最长响应时间（默认值为 10秒）。当收到询问时，主机在 0 到 *N* 之间随机选择发送响应所需经过的时延。对应于最小时延的响应最先发送。
- 同一个组内的每一个主机都要监听响应，只要有本组的其他主机先发送了响应，自己就可以不再发送响应了。

3. 多播路由选择

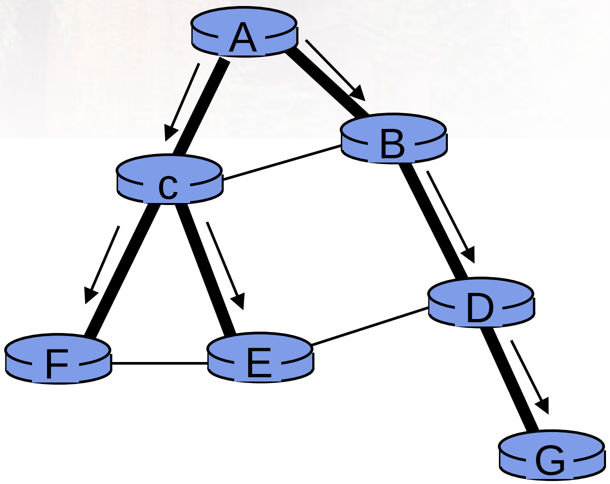
- 多播路由选择协议尚未标准化。
- 一个多播组中的成员是动态变化的，随时会有主机加入或离开这个多播组。
- 多播路由选择实际上就是要找出以源主机为根结点的多播转发树。
- 在多播转发树上的路由器不会收到重复的多播数据报。
- 对不同的多播组对应于不同的多播转发树。同一个多播组，对不同的源点也会有不同的多播转发树。

In-network duplication

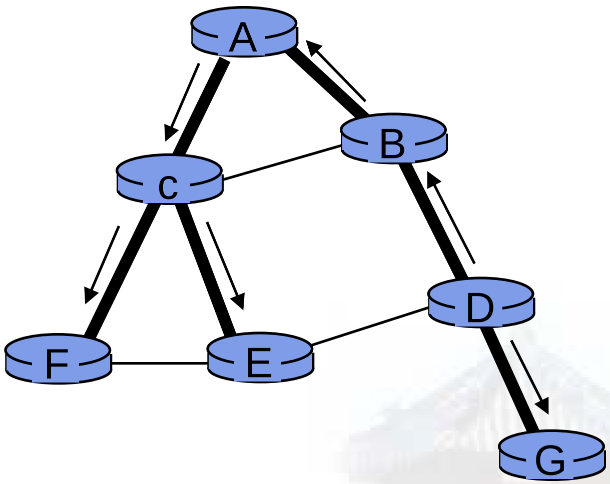
- ***flooding:*** **when node receives broadcast packet, sends copy to all neighbors**
  - **problems: cycles & broadcast storm**
- ***controlled flooding:*** **node only broadcasts pkt if it hasn't broadcast same packet before**
  - **node keeps track of packet ids already broadacsted**
  - **or reverse path forwarding (RPF): only forward packet 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 then forward/make copies only along spanning tree



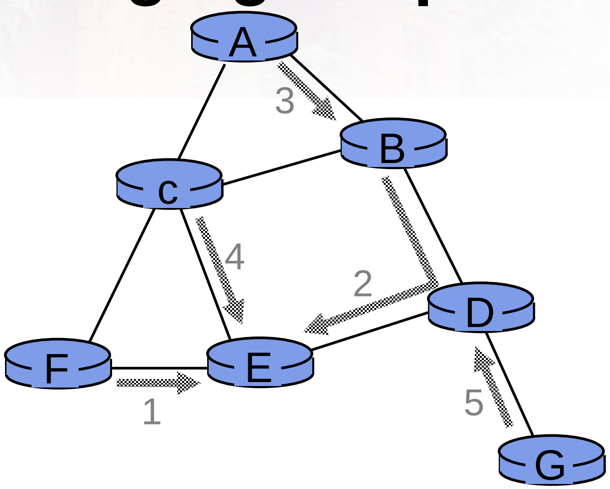
(a) broadcast initiated at A



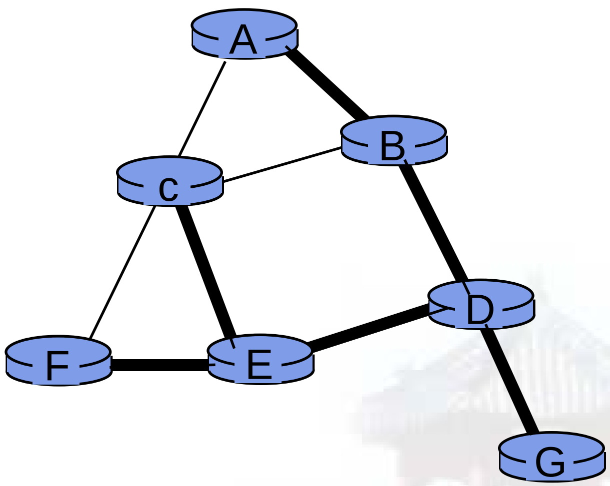
(b) broadcast initiated at D

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 (center: E)

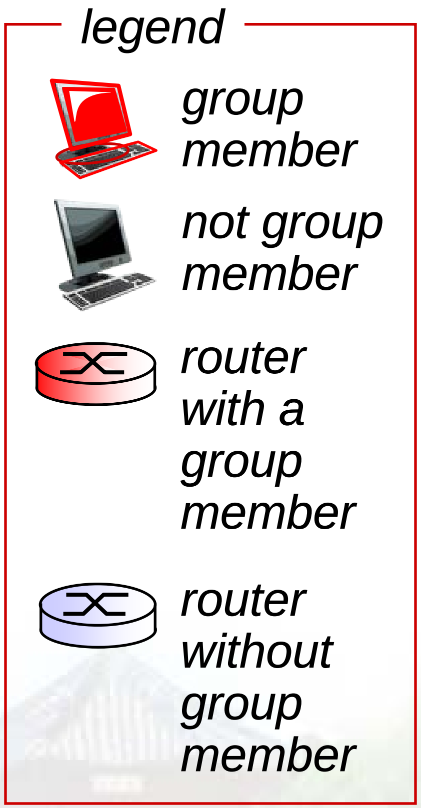


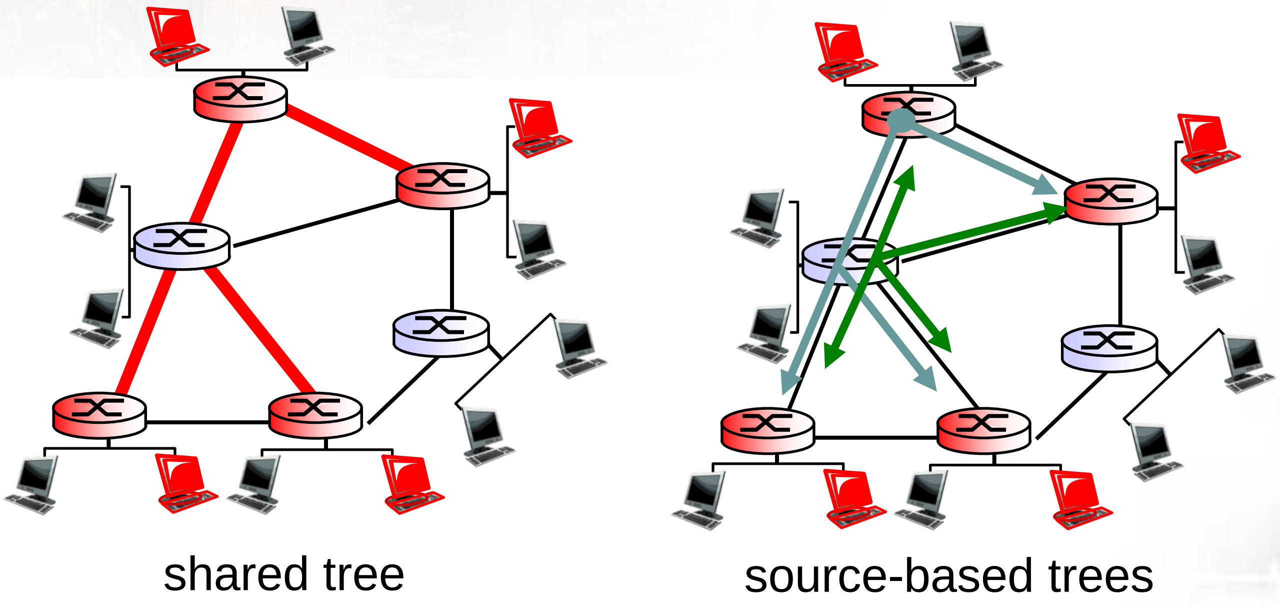
(b) constructed spanning tree

Multicast routing: problem statement

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

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





Approaches for building mcast trees

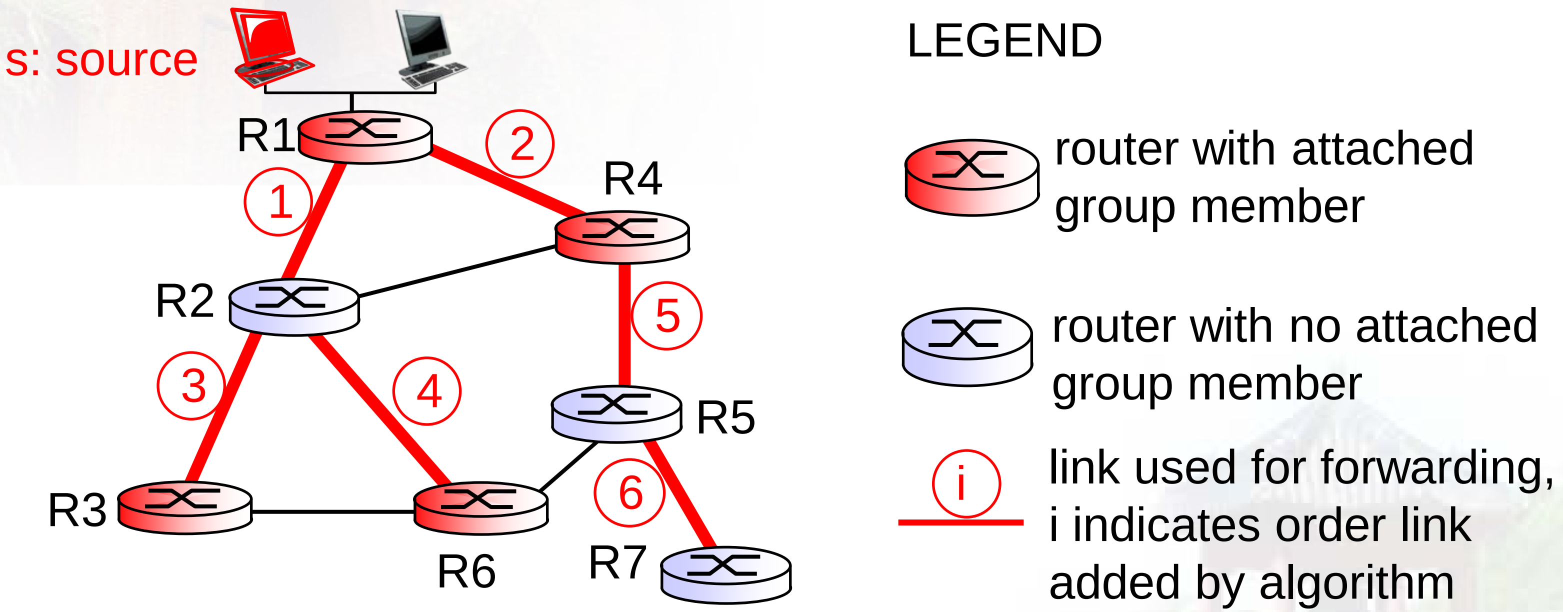
approaches:

- ***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**



Reverse path forwarding

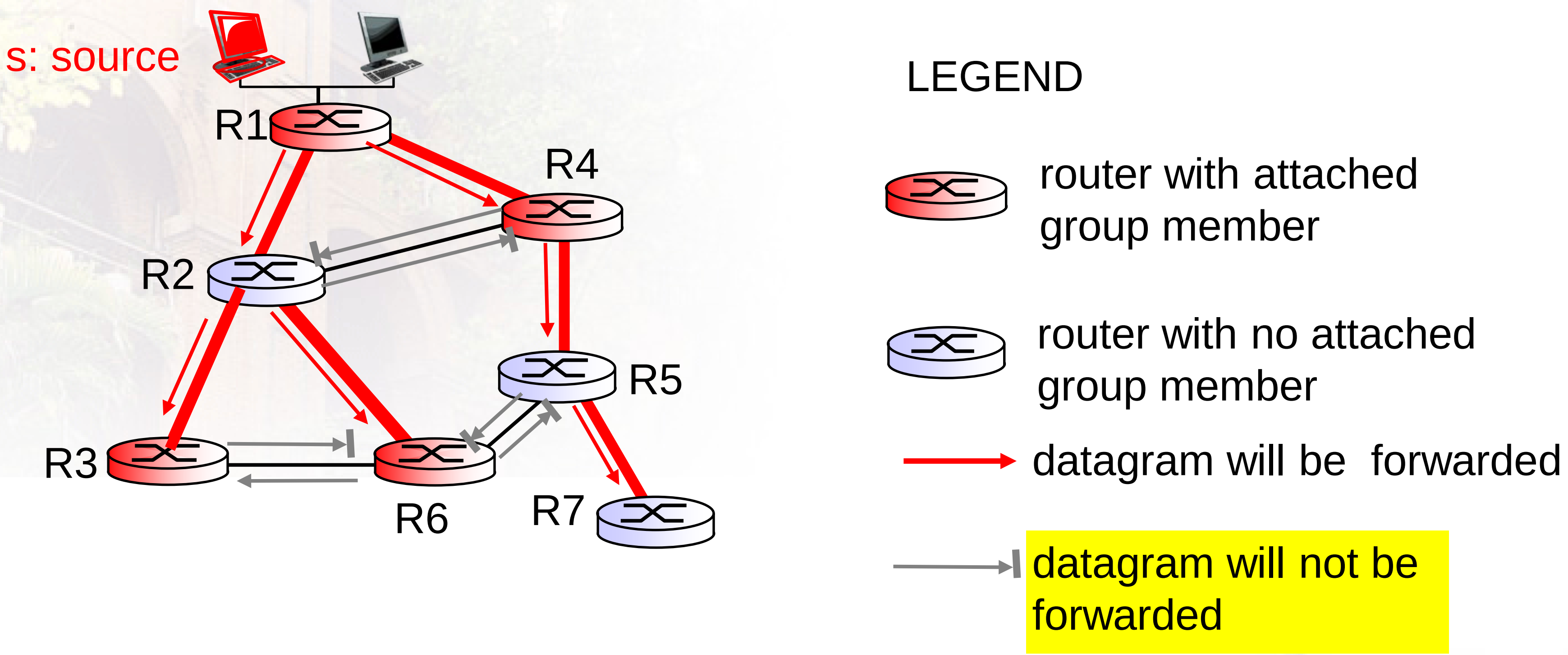
- rely on router's knowledge of unicast shortest path from it to sender
- each router has simple forwarding behavior:

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

then **flood datagram onto all outgoing links**

else **ignore datagram**

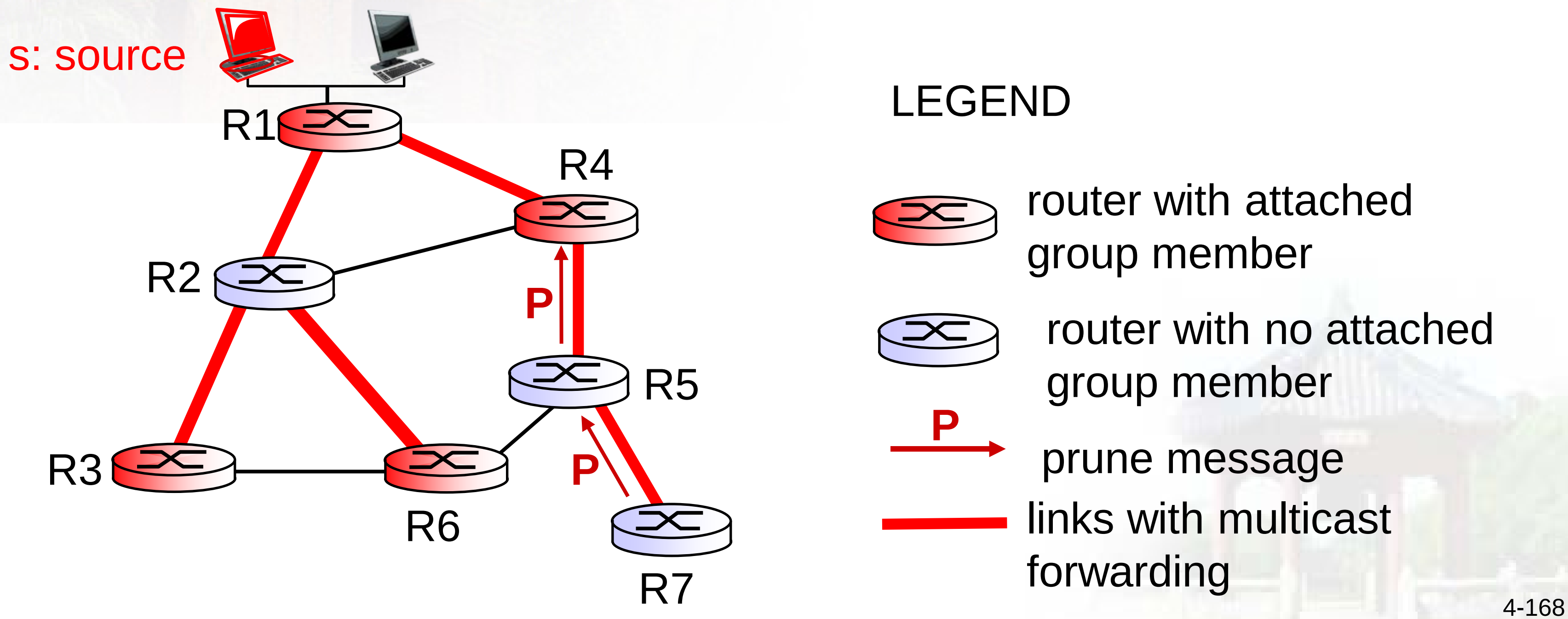
Reverse path forwarding: example



- 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



Shared-tree: steiner tree

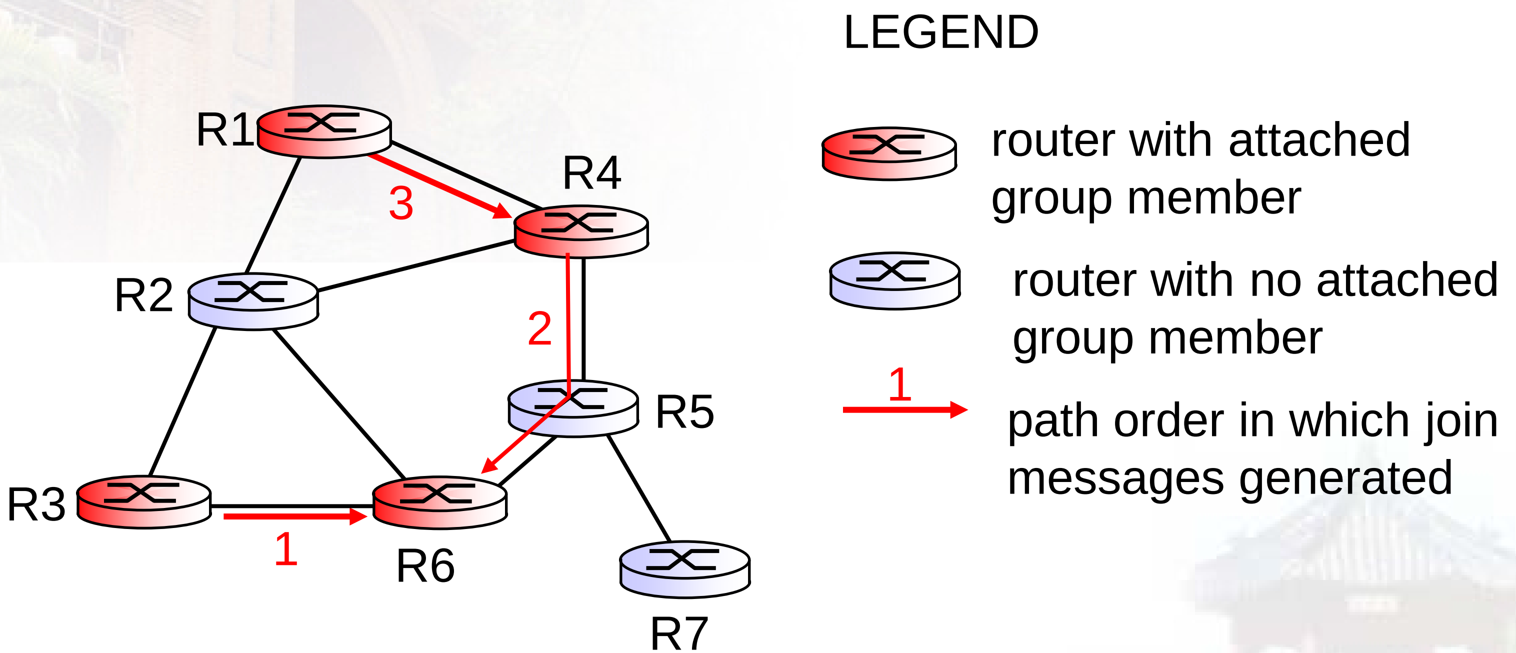
- ***steiner tree:*** **minimum cost tree connecting all routers with attached group members**
- **problem is NP-complete**
- **excellent heuristics exists**
- **not used in practice:**
  - **computational complexity**
  - **information about entire network needed**
  - **monolithic: rerun whenever a 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

Center-based trees: example

suppose R6 chosen as center:



Internet Multicasting Routing: DVMRP

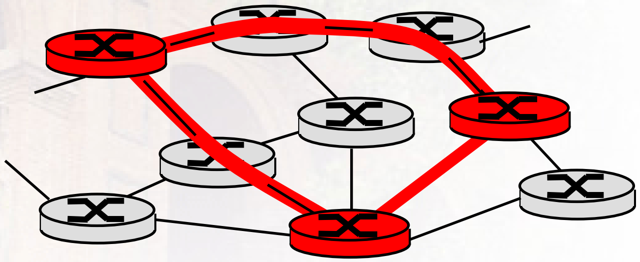
- **DVMRP:** distance vector multicast routing protocol, RFC1075
- ***flood and prune:*** reverse path forwarding, source-based tree
  - RPF tree based on DVMRP's own routing tables constructed by communicating DVMRP routers
  - no assumptions about underlying unicast
  - initial datagram to mcast group flooded everywhere via RPF
  - routers not wanting group: send upstream prune msgs

DVMRP: continued…

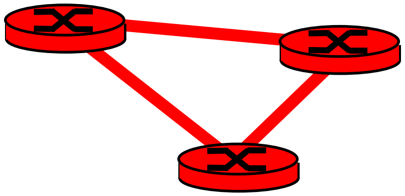
- *soft state:* DVMRP router periodically (1 min.) "forgets" branches are pruned:
  - mcast data again flows down unpruned branch
  - downstream router: reprune or else continue to receive data
- routers can quickly regraft to tree
  - following IGMP join at leaf
- odds and ends
  - commonly implemented in commercial router

Tunneling

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 (recall IPv6 inside IPv4 tunneling)
- 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***

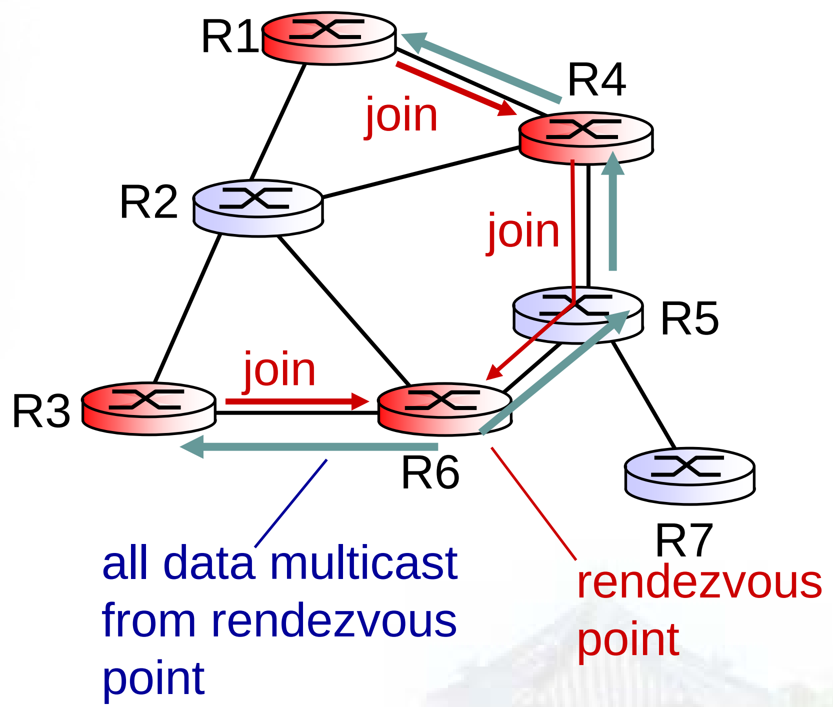
PIM- dense mode

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

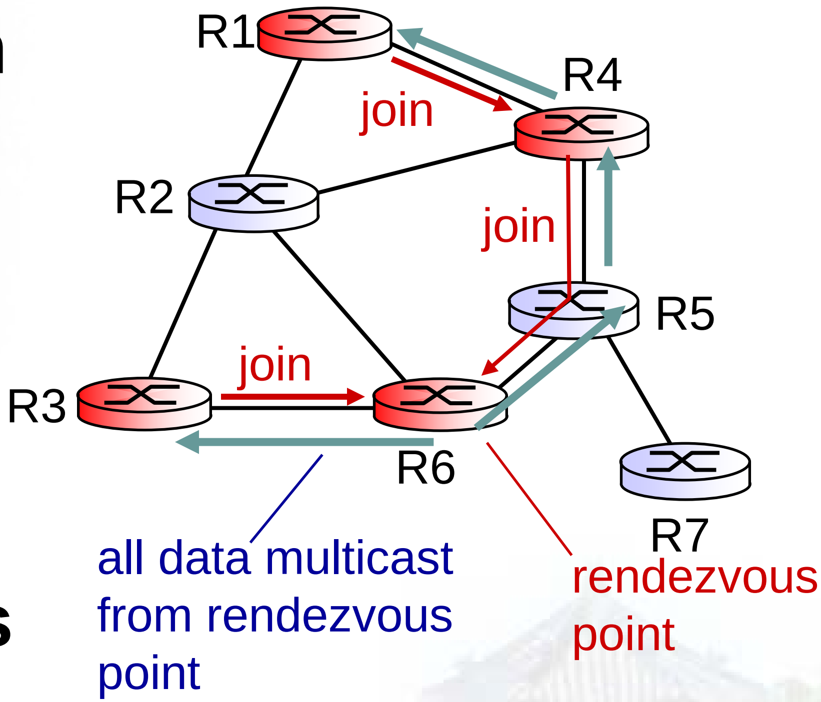
- center-based approach
- router sends *join* msg to rendezvous point (RP)
  - intermediate routers update state and forward *join*
- after joining via RP, router can switch to source-specific tree
  - increased performance: less concentration, shorter paths



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!"**



Chapter 4: *done!*

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

- understand principles behind network layer services:
  - network layer service models, forwarding versus routing how a router works, routing (path selection), broadcast, multicast
- instantiation, implementation in the Internet

Quiz for Chapter 4

- Brief describe the A Link-State Routing Algorithm procedure, and give a practical example of Internet protocol using it.
- What's the difference of source-based tree and group-shared tree? What's the approaches for building these mcast trees?
- What's BGP? How it works?
- What's the difference of broadcast vs multicast routing?
- What's CIDR?
- Please compare the LS and DV algorithms

- What's the difference of Datagram or VC network? Use Internet and ATM as examples
- What's DHCP? Brief describe how it works.
- How does the Graph theory apply to computer networks? Point out 2 graph examples in computer networks.
- What's Head-of-the-Line (HOL) blocking? Where does it occur?
- What's the meaning of hierarchical addressing: route aggregation?
- Give the fragmentation of a IP datagram, with size of 5000 bytes and the MTU = 1500 bytes.
- What‘s the changes (or advantages) of IPv6, compare to IPv4?

- What's the key network-layer functions
- What's longest prefix matching? Give an example.
- What's NAT, and how it works?
- How to solve the NAT traversal problem? Give 2 solutions
- What service model for “channel” transporting datagrams from sender to receiver?
- What's the difference of network and transport layer connection service?
- What's PIM, how does PIM sparse mode work?
- What‘s RPF, and how does it work?

- **Talk about the three types of switching fabrics inside routers, focus on their advantages and disadvantages**
- **How does Traceroute program works?**
- **Talk about the Transition from IPv4 to IPv6. How will the network operate with mixed IPv4 and IPv6 routers?**
- **What does a VC consist of?**
- **What's RIP? What algorithm does it use and how it works?**
- **Why different Intra- and Inter-AS routing?**

The End of Chapter 4

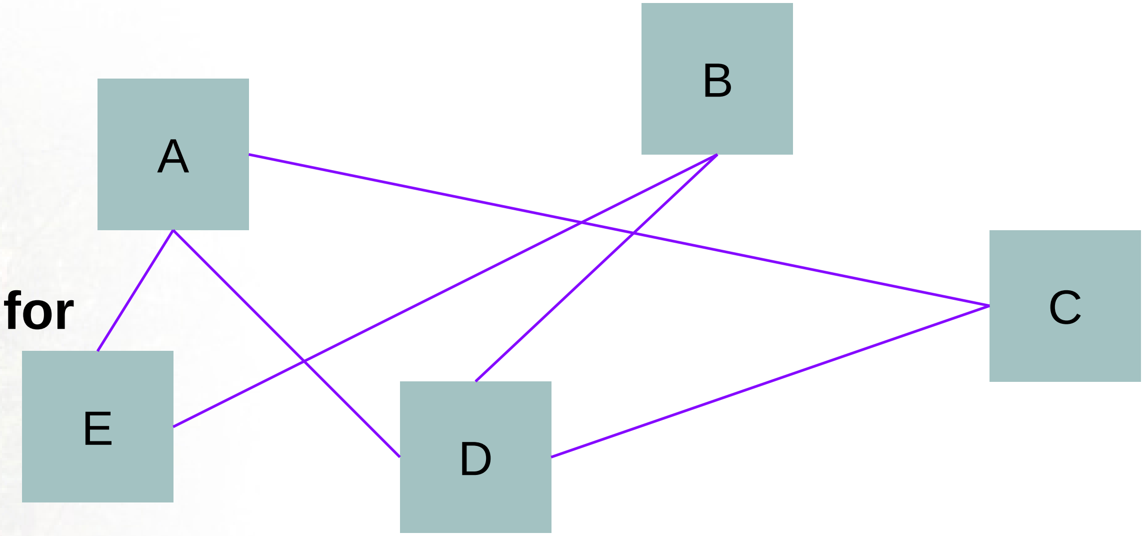
- **Task-2: virtual routing**
  (Application-layer routing)
  - **self-organized routing**
    - **Select a virtual topo for members' computers**
    - **Build virtual connection between computers according to the virtual topo, define the cost of links;**
    - **Each computer acts as both client and router.**
    - **Each computer exchanges and updates routing table periodically.**
    - **A computer can send message to other computers,**

Hint:

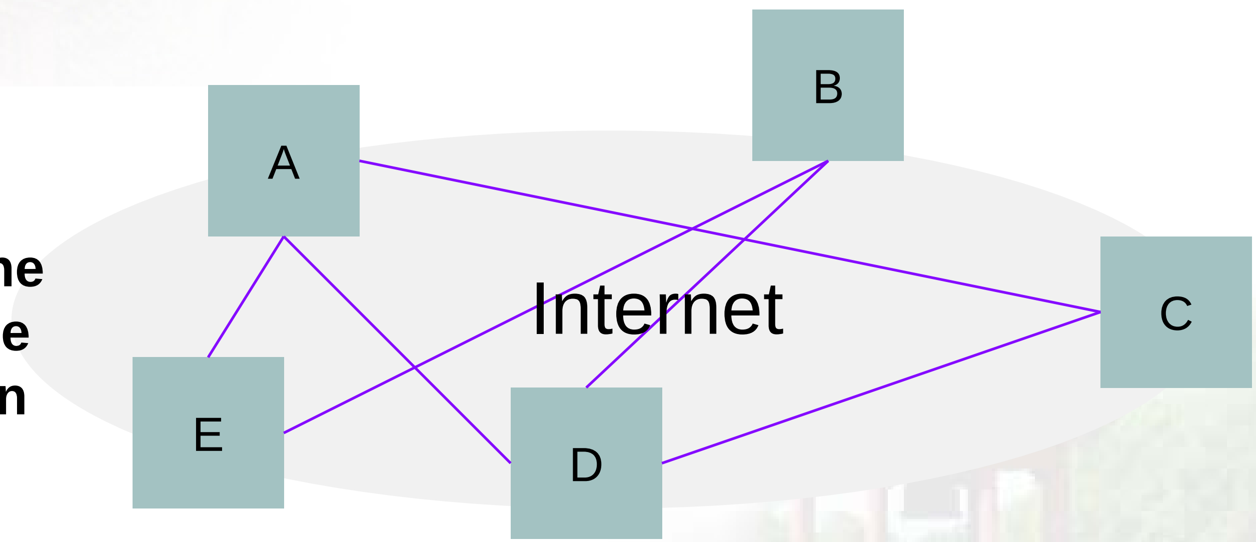
- **IP-in-IP (IP-layer virtual routing) or**
- **use sock directly (Application-layer routing)**
- **Use TCP or UDP**

Step 1:

- Design the virtual topo (link cost)
- Each node has two ports for receiving and sending: Prt_I, Prt_O.



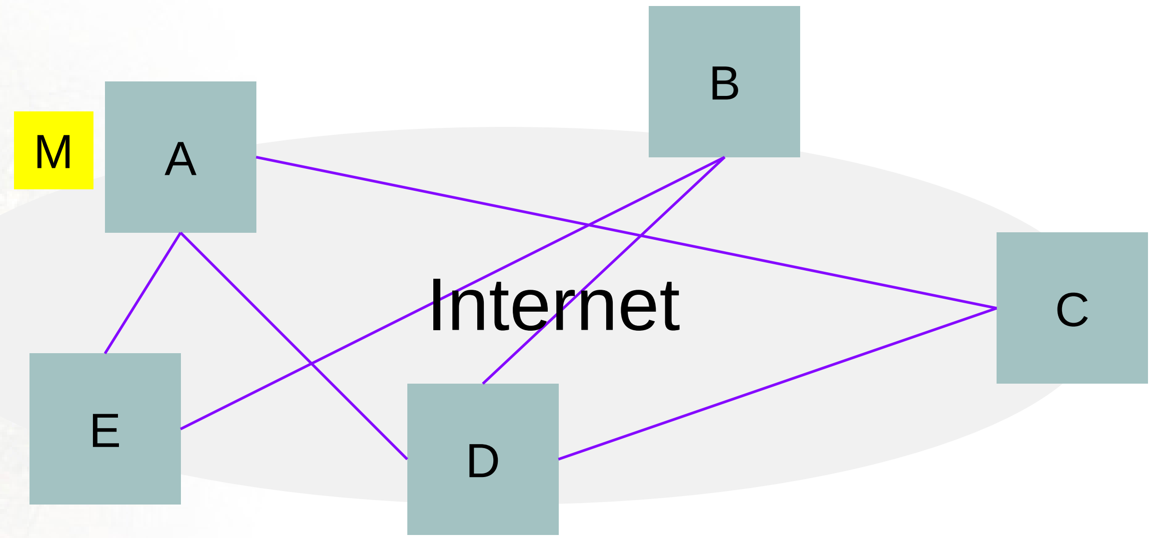
Step 2:



Build the virtual Topo over Internet, define the cost of links; exchange the routing information periodically

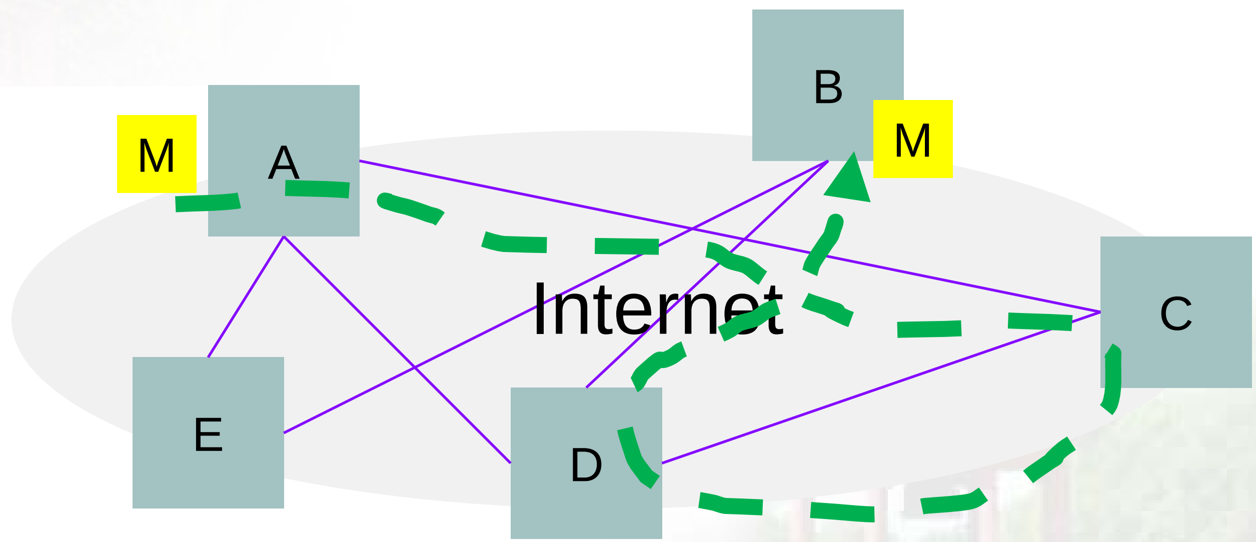
Step 3:

Simulate the routing and forwarding. For example: A sends M to B. Which path is better?
A →E → B? or
A →D → B?
A → C → D → B?



Step 4:

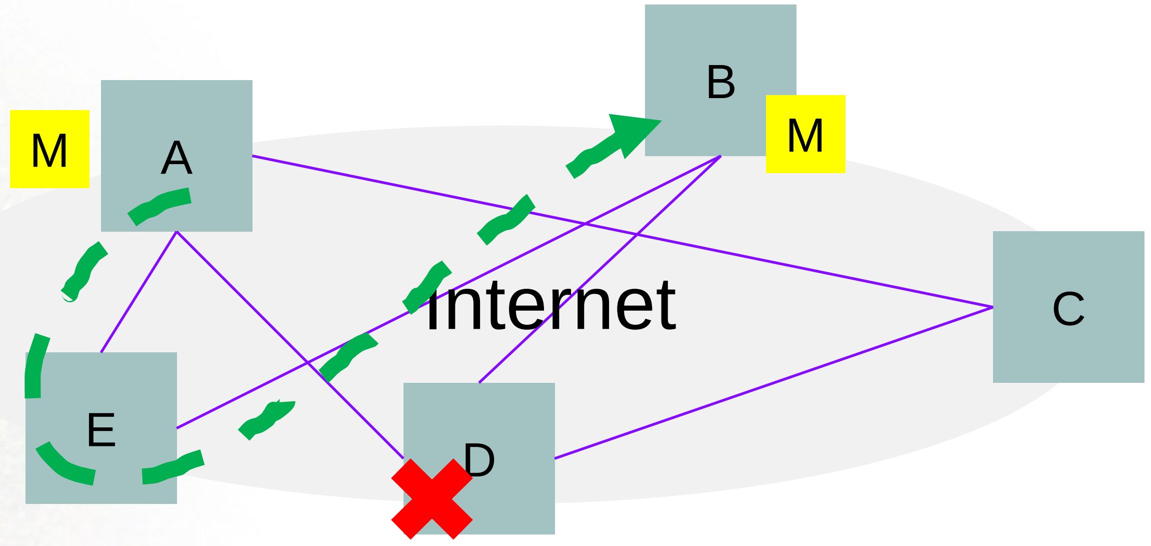
Transmit data M via the best path, e.g., A → C → D → B



Please try different topo and different routing algorithms (LS & DV).

Step 5:

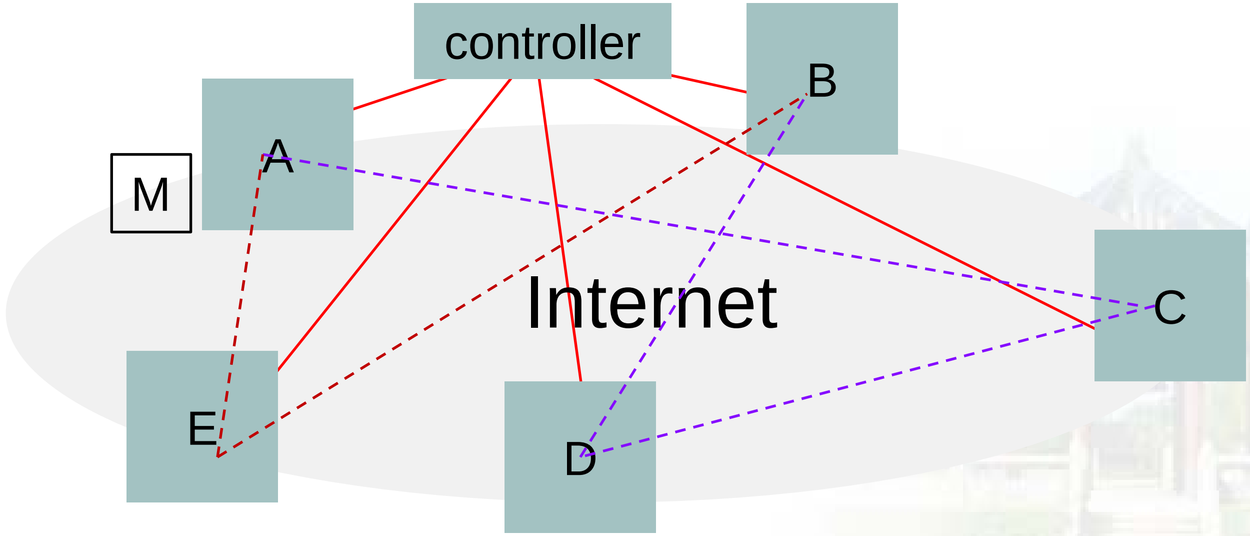
A node is down.
e.g., D



Please try different topo and different routing algorithms (LS & DV).

- **Task-2: virtual routing**
  - **centralized routing**
    - **Like the above self-organized routing**
    - **Controller determines and distributes routing policy (routing table) to each member**

Example: A sends M to B. Which path is better? A →E → B? or A → C → D → B?



- **Submit**
  - **PPTs + demo video**
  - **Source code (and the compiled executable files)**
  - **The project report documents (including introduction, design, setup and deploy, and result, project management records)**
  - **The individual report of each team members (your contributions, and anything else you want to talk about)**
  - **votes of the top 5 teams (based on their presentations and your observations, give comments of 2-3 sentences)**
  - **A list that shows each member’s contribution and grade.**

- **Put all file into a package and name it as:**
  A_B_C.rar,
  A: the student ID of group leader;
  B: the name of group leader;
  C: task1 or task2
  example: 1500001_张三_task1.rar

- **Group leader submit it to the given FTP server.**

- **Basic points**
  - **Protocol design. (10 points)**
  - **Finish basic function correctly (w/o error). (60 points)**
  - **On time (WEEK 15). (10 points)**
  - **Documents, codes, presentation. (20 points)**
  - **votes**
  - **in-group assessment**