SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

L7 – The Network Layer

50.012 Networks

Jit Biswas

Cohort 1: TT7&8 (1.409-10)

Cohort 2: TT24&25 (2.503-4)

Introduction

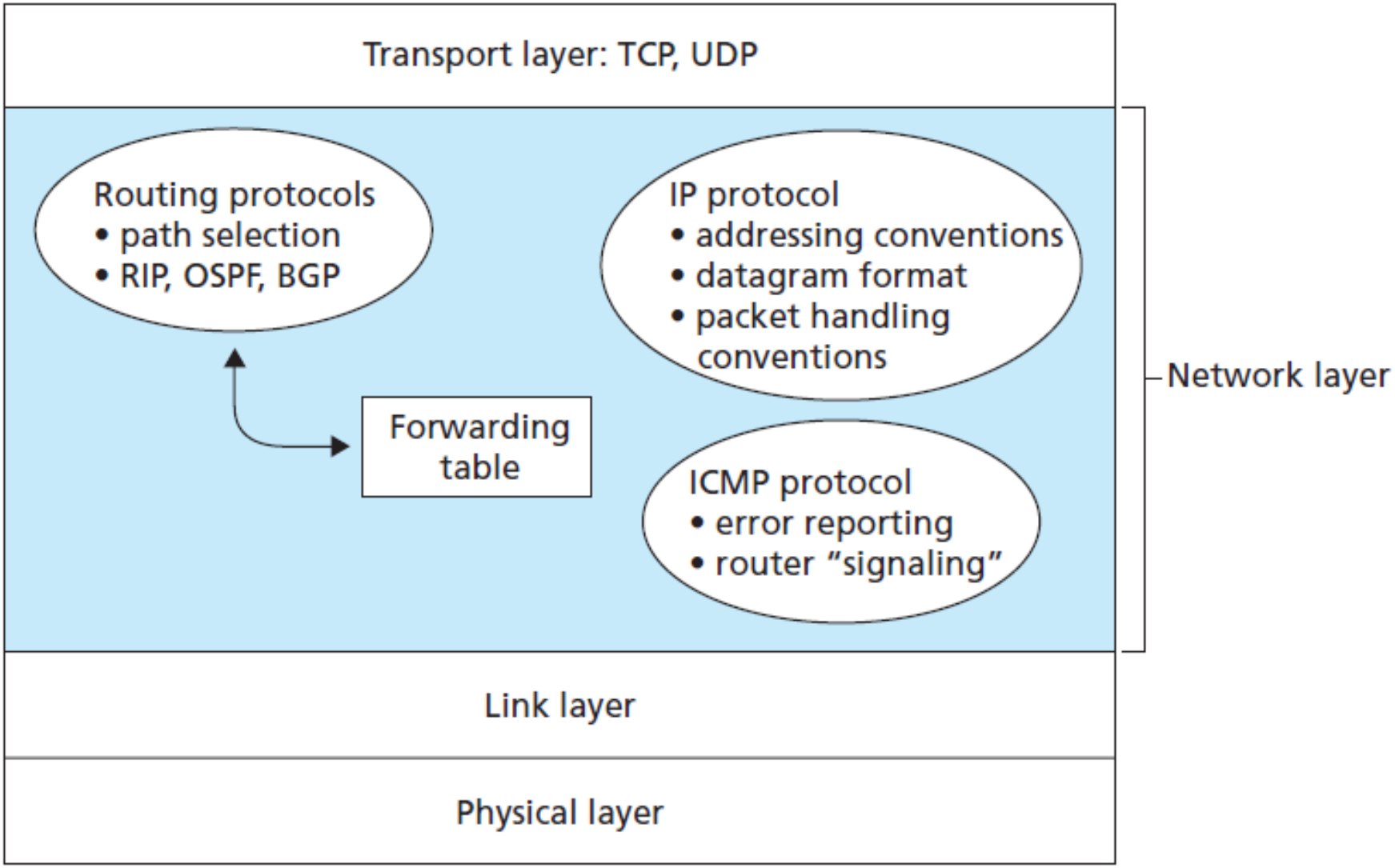
- Todays lecture:
  - Introduction to the Network layer
  - Forwarding
  - Switching fabric
  - The routing problem
  - Dijkstra's algorithm
- Note: parts of this slide set are based on Kurose & Ross chapter 4 slide set

The Network Layer

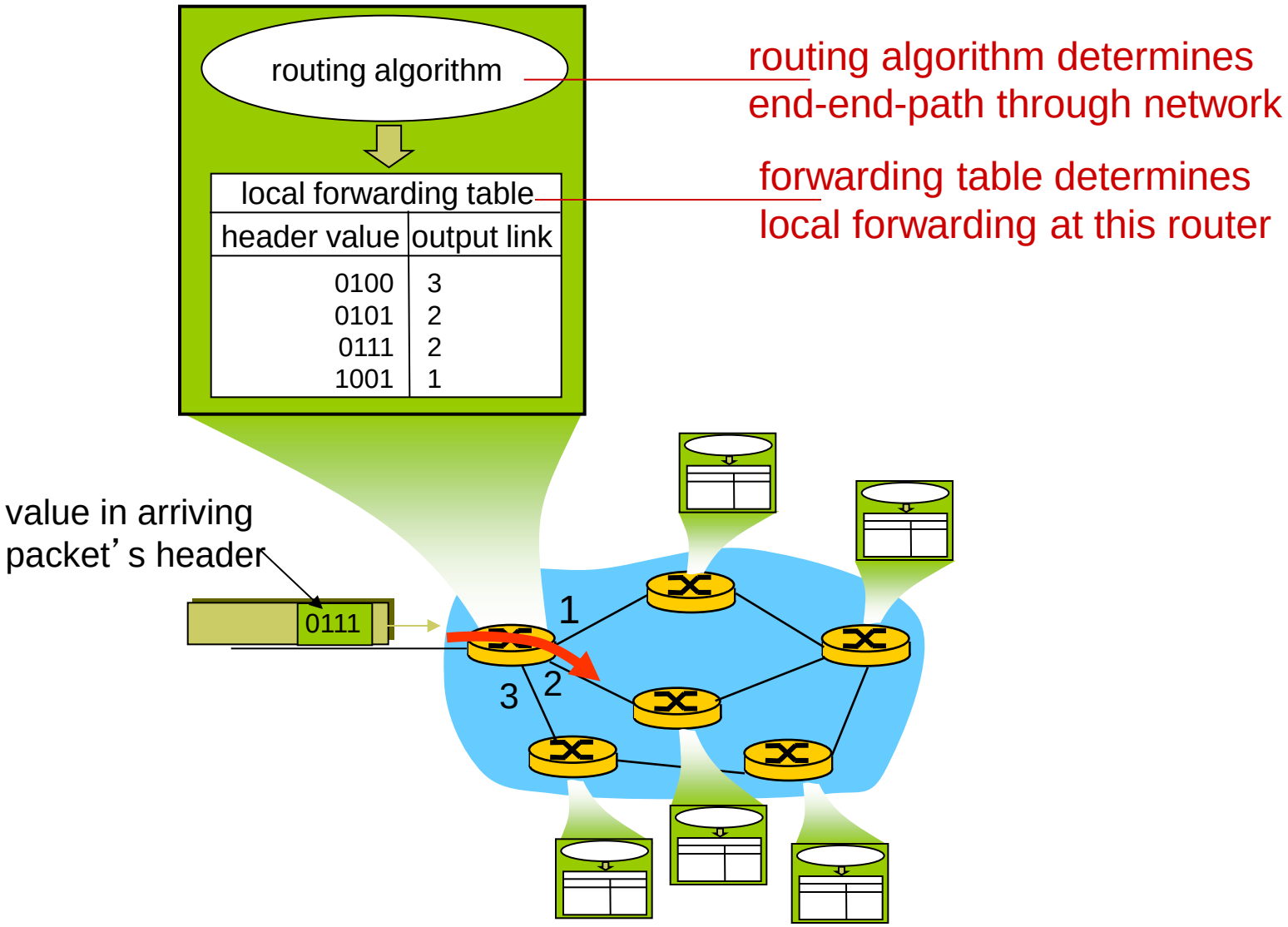
The Network Layer

- The Network layer provides logical connection between hosts
  - Protocols: IP, ICMP, IGMP
  - Addressing is based on IP addresses
- Remember:
  - App layer: connection between applications
  - Transport layer: connection between processes
  - Link layer: single-hop connections between interfaces
- The network layer is domain of routers and Internet backbone
- Main services of Network layer:
  - Routing: find best path for datagram from hostA to hostB
  - Forwarding: move datagrams from input to output interface

The Network Layer – what's inside

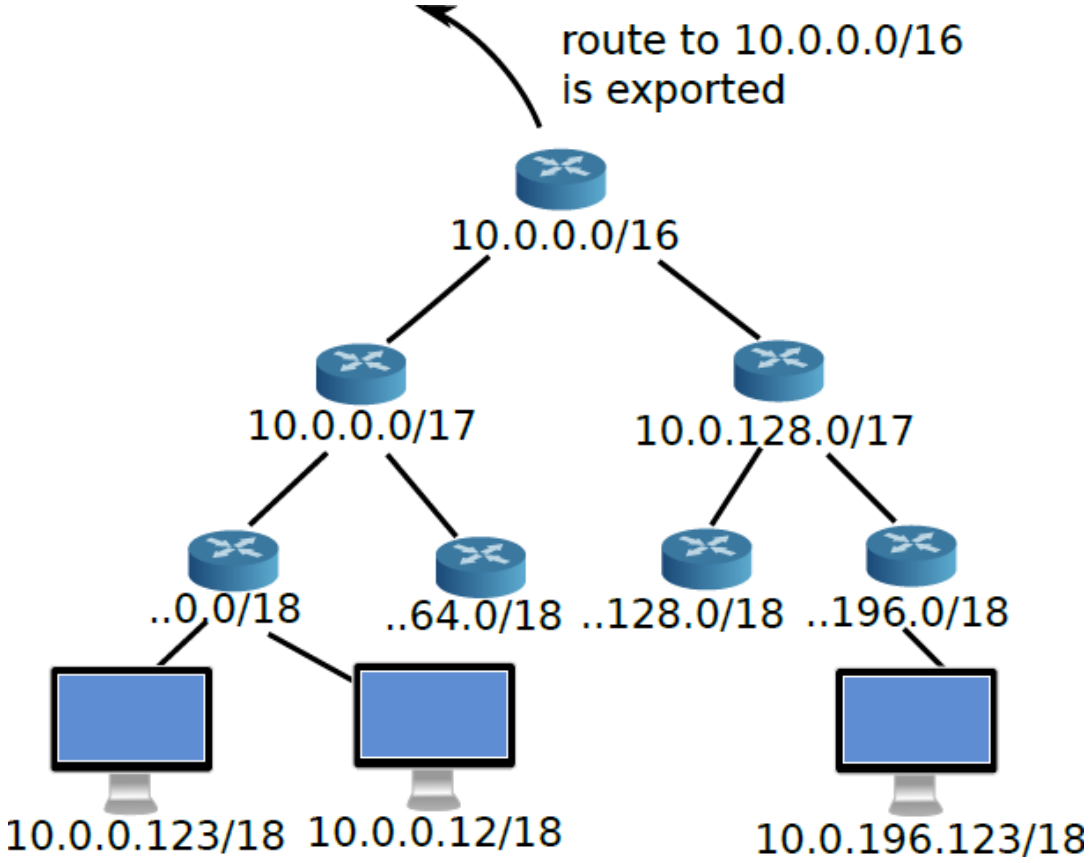


Interplay between routing and forwarding

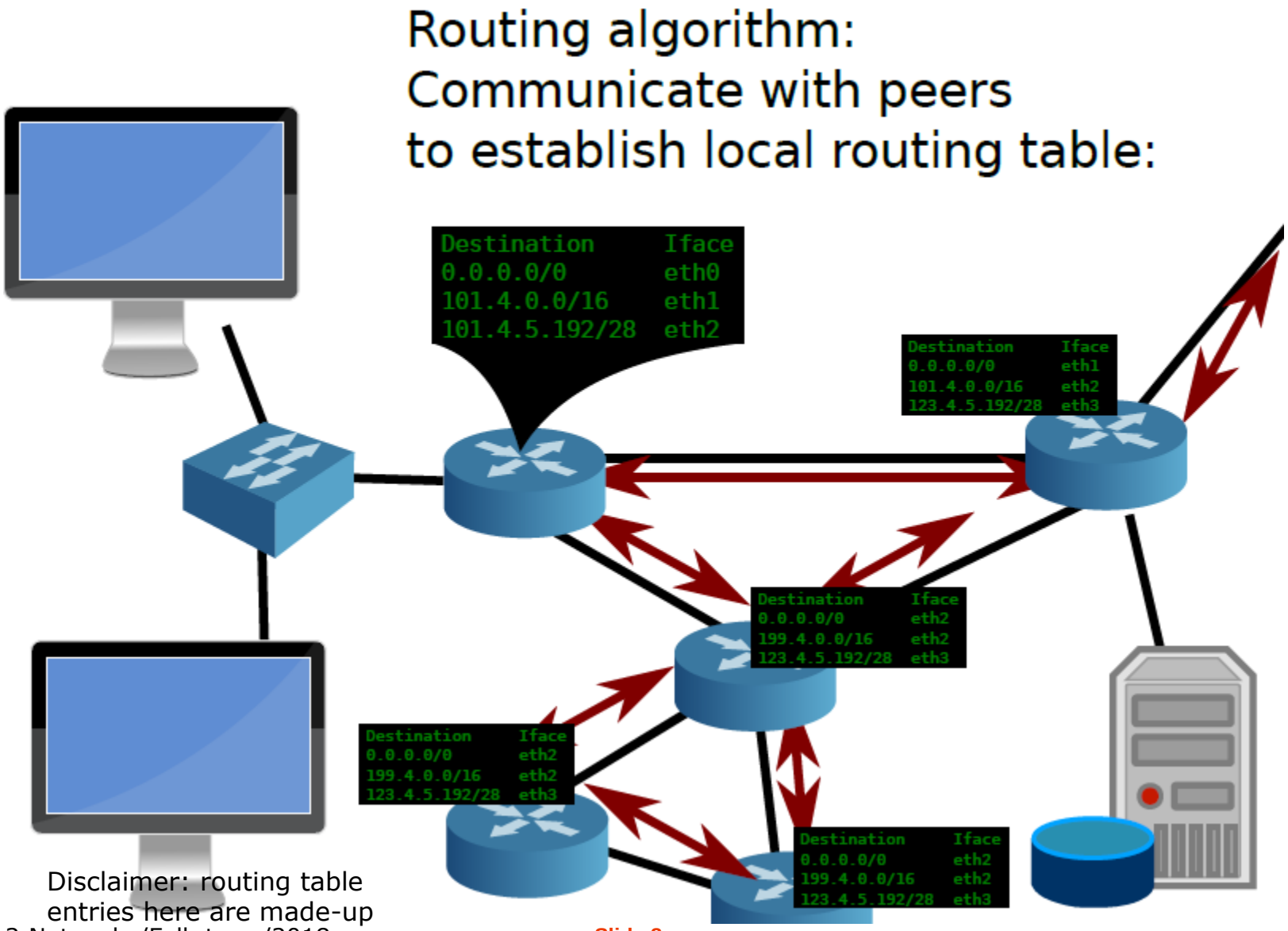


Subnet Masks and CIDR

- Recap: What are subnetworks used for again?
- From client perspective: *know when to send via GW*
- From routing perspective: enable hierarchical routing

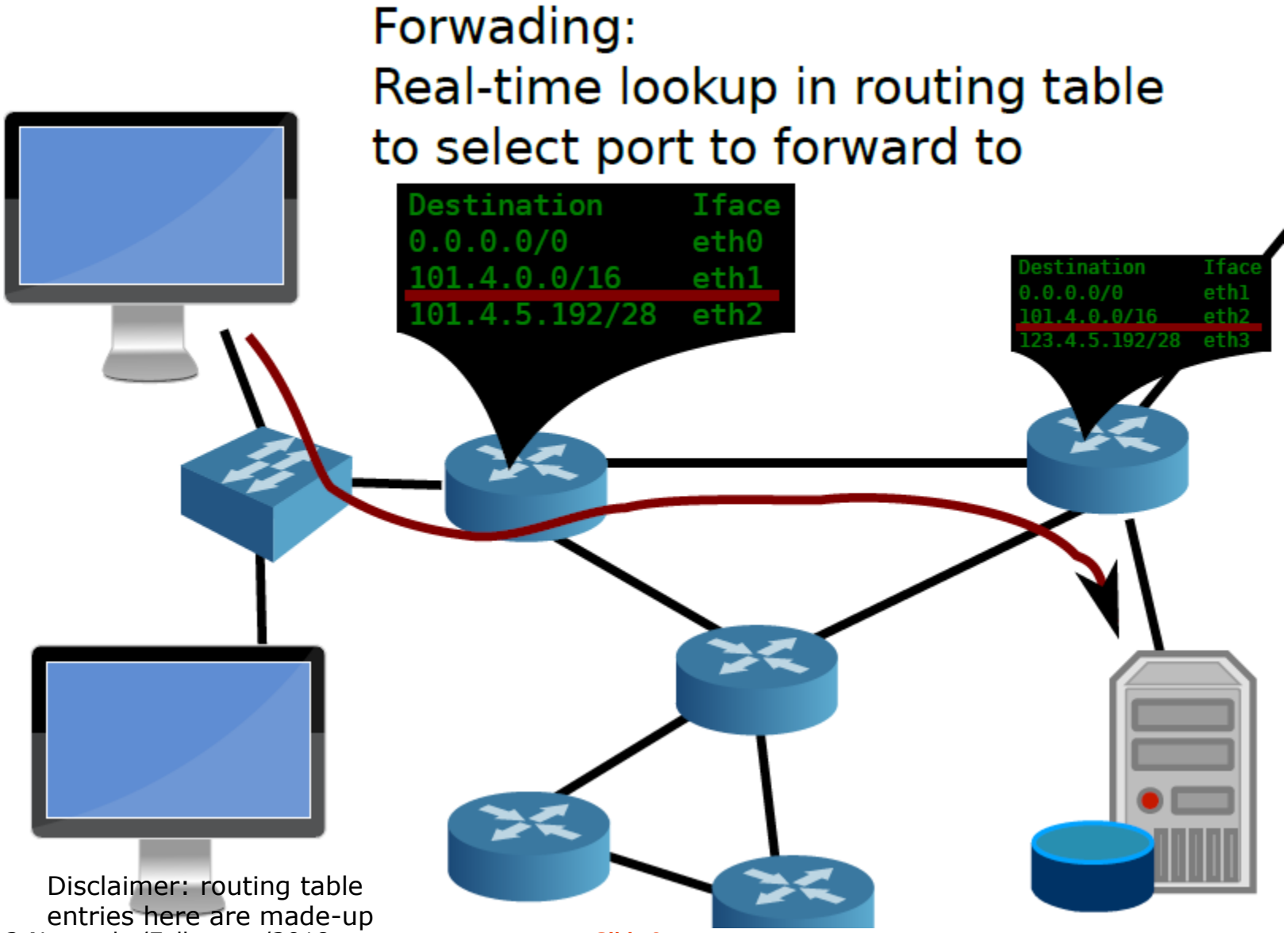


Routing vs Forwarding



Disclaimer: routing table entries here are made-up

Routing vs Forwarding



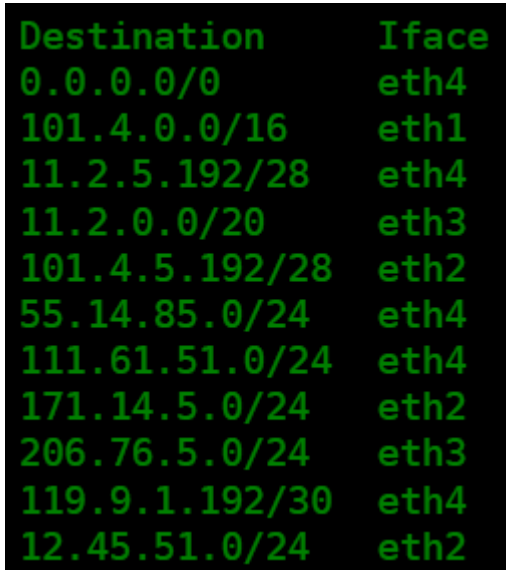
Disclaimer: routing table entries here are made-up

Routing is de-centralized

- Remember: Internet is based on packet switching
  - No permanent circuits are established
  - Packet route are dynamic and can change during connection
- Internet is also de-centralized
  - No central authority that keeps all optimal routes
- Routing decision are made by each router on the way
  - Each router selects next router to forward based on destination address

How are routing decisions made?

- Routing decisions have to be taken fast
  - One for every datagram, millions of datagrams per second
- Basic approach: Routing table lookup
  - Each row contains target CIDR network and interface
  - Target IP is matched to one or more rows
  - Most specific entry is used



Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

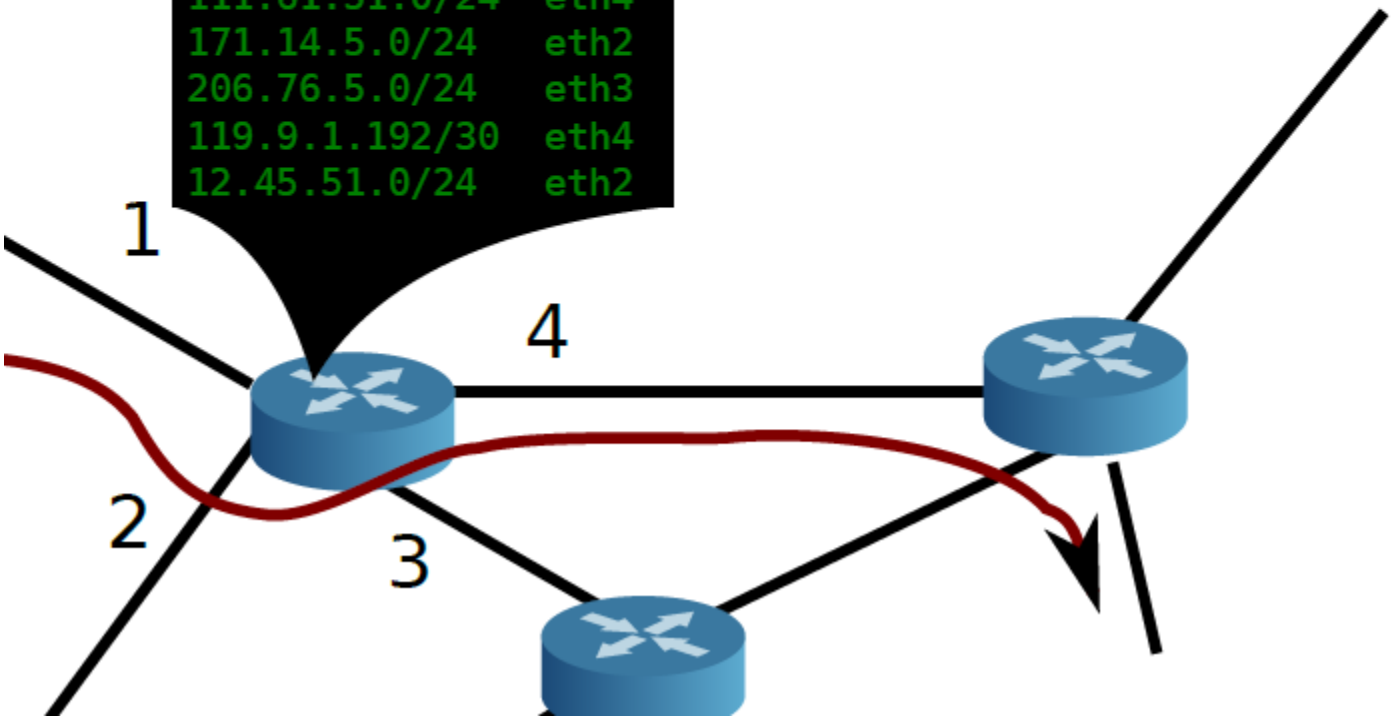
An example routing table

Example routing decisions

Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

What interface to take to:

- 11.2.5.201
- 11.2.6.201
- 77.132.5.100
- 172.16.54.200

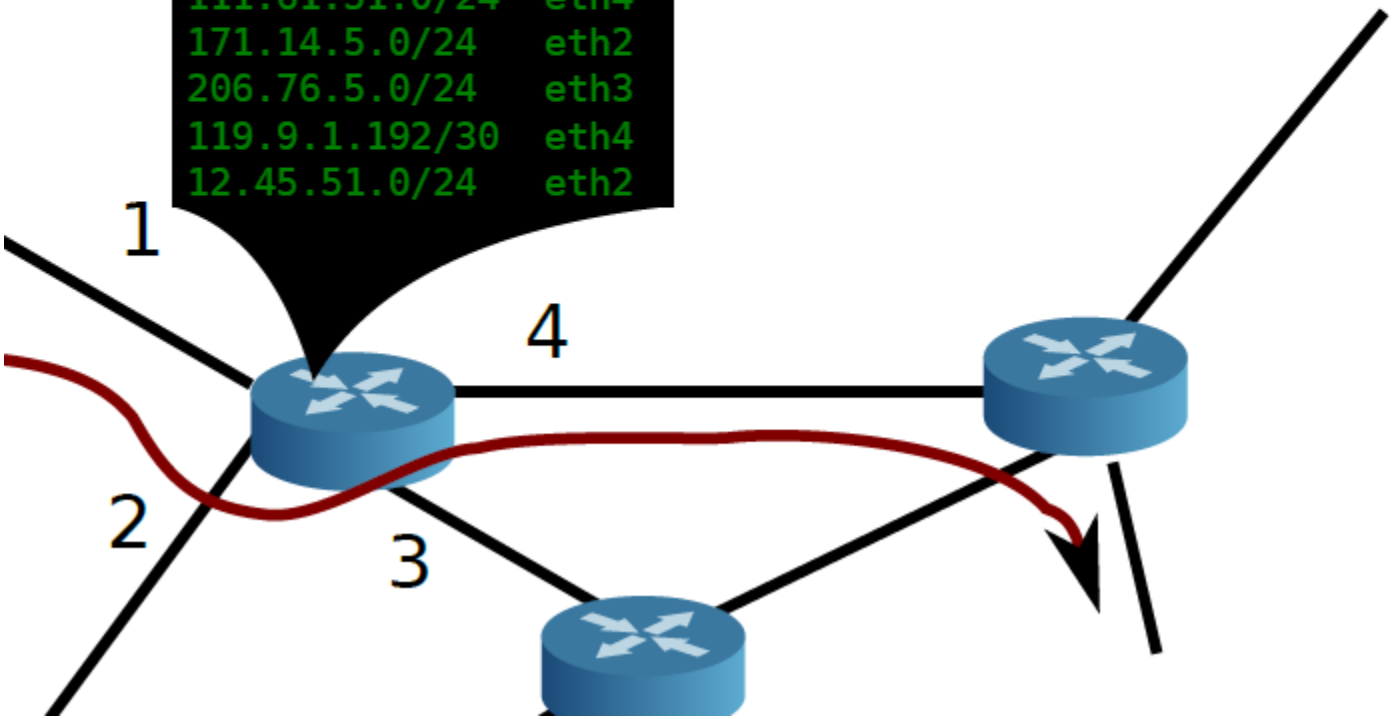


Example routing decisions

Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

What interface to take to:

- 11.2.5.201 - eth4
- 11.2.6.201
- 77.132.5.100
- 172.16.54.200

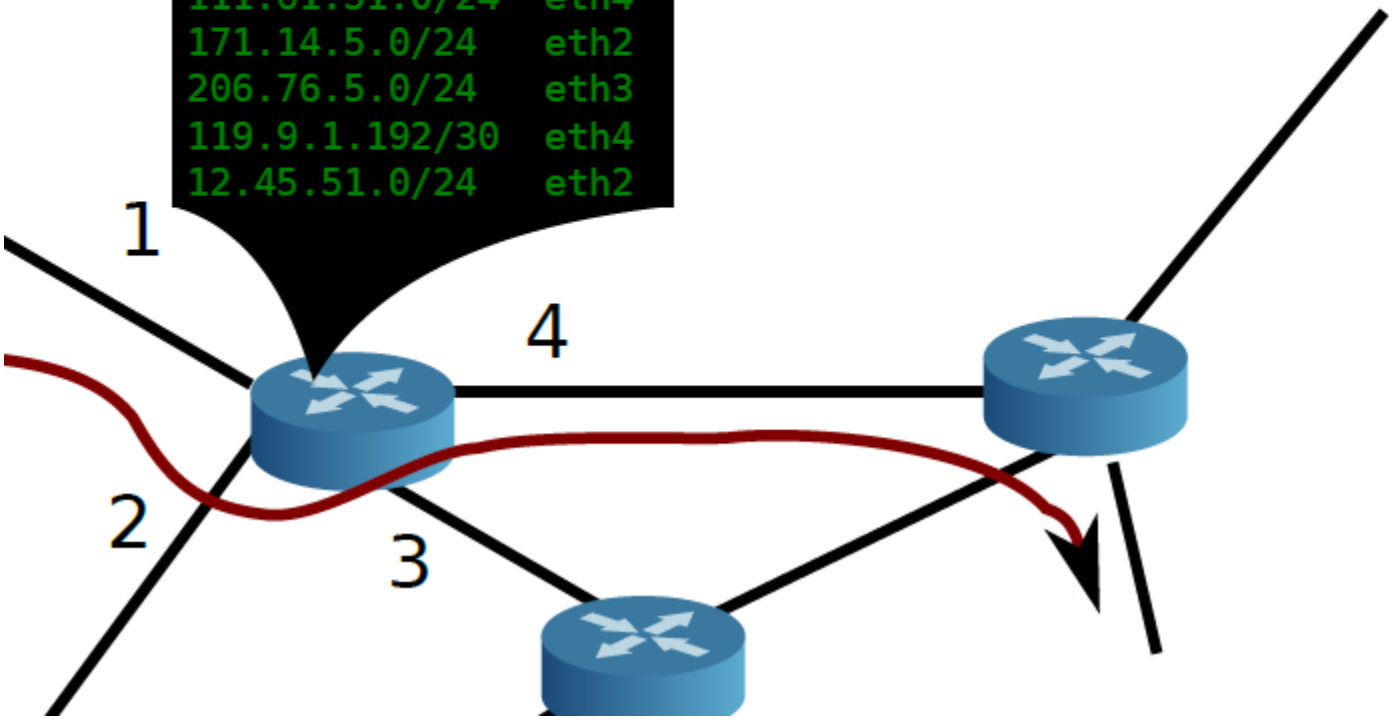


Example routing decisions

Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

What interface to take to:

- 11.2.5.201 - eth4
- 11.2.6.201 - eth3
- 77.132.5.100
- 172.16.54.200

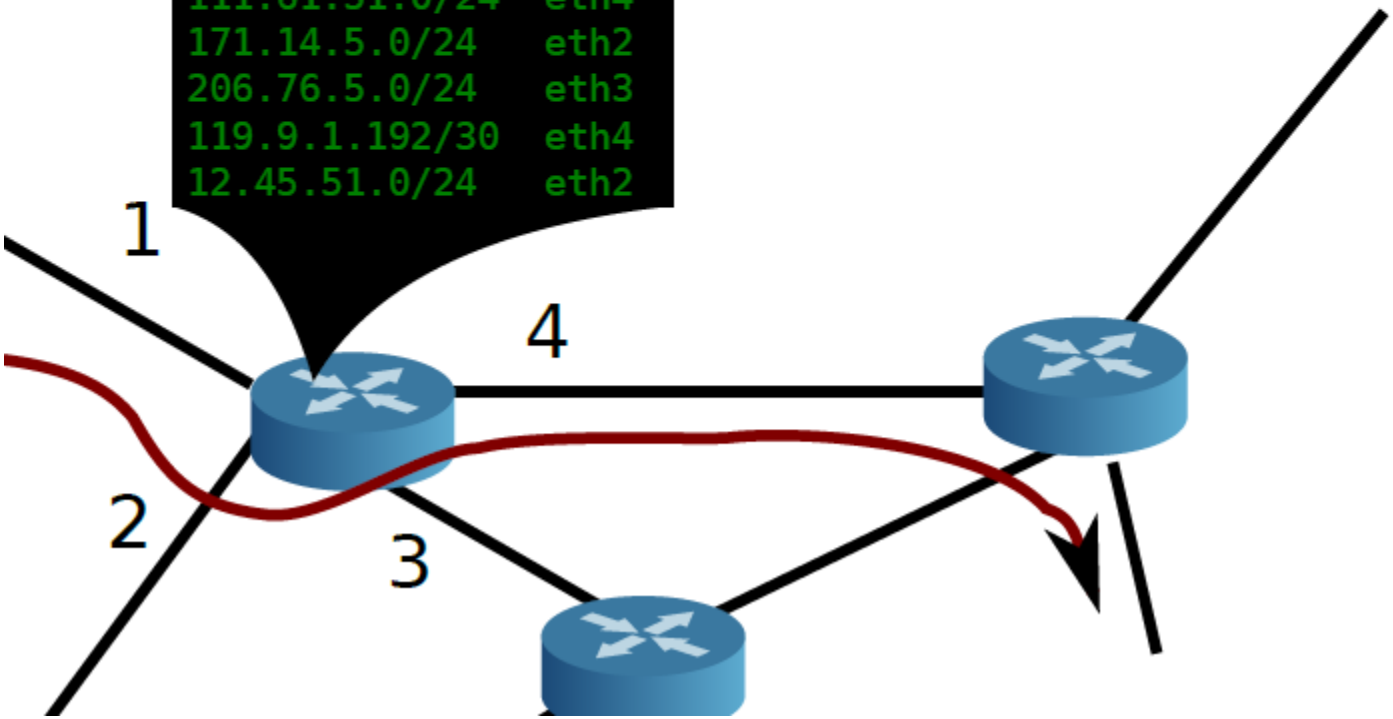


Example routing decisions

Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

What interface to take to:

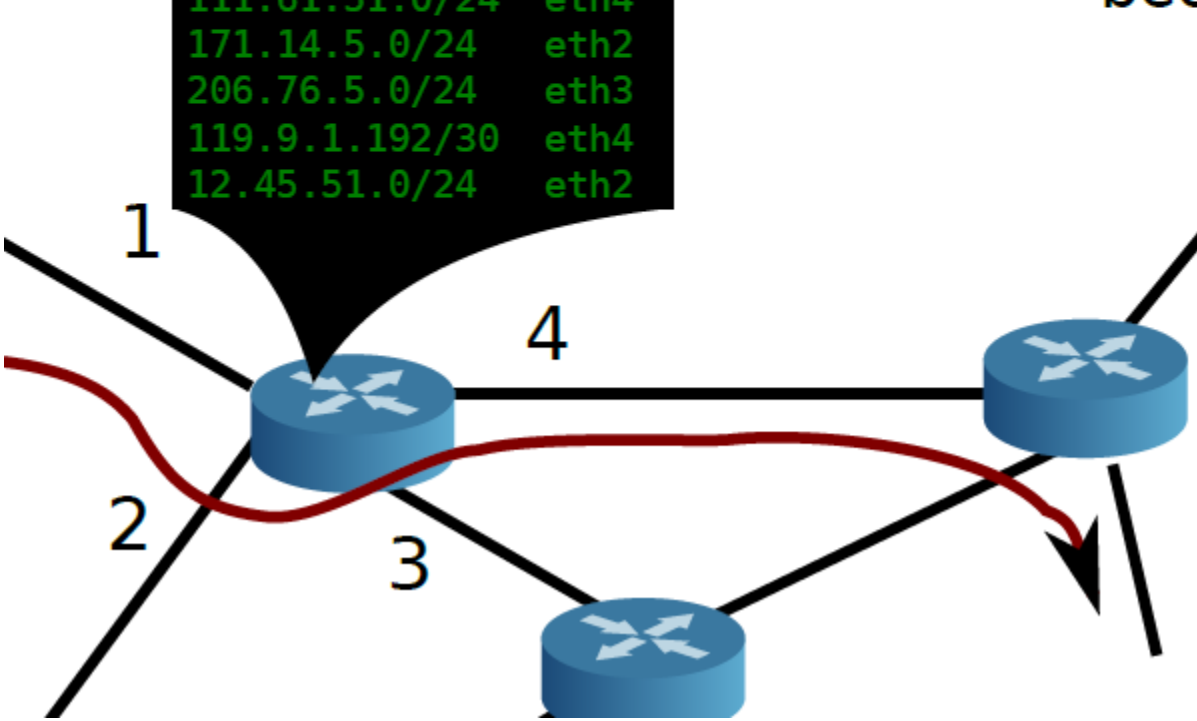
- 11.2.5.201 - eth4
- 11.2.6.201 - eth3
- 77.132.5.100 - eth4
- 172.16.54.200



Example routing decisions

Destination	Iface
0.0.0.0/0	eth4
101.4.0.0/16	eth1
11.2.5.192/28	eth4
11.2.0.0/20	eth3
101.4.5.192/28	eth2
55.14.85.0/24	eth4
111.61.51.0/24	eth4
171.14.5.0/24	eth2
206.76.5.0/24	eth3
119.9.1.192/30	eth4
12.45.51.0/24	eth2

What interface to take to:
- 11.2.5.201 - eth4
- 11.2.6.201 - eth3
- 77.132.5.100 - eth4
- 172.16.54.200 - dropped because private!



Activity 7: Packet Forwarding

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Destination prefix	Interface
128.75.43.0 / 24	Eth0
128.75.43.0 / 25	Eth1
192.12.16.0 / 23	Eth2
Default	Eth3

The routing table of a router is shown above:

On which interface will the router forward packets addressed to destinations 128.75.43.16 and 192.12.17.138 respectively ?

(A) Eth1 and Eth2

(B) Eth0 and Eth2

(C) Eth0 and Eth3

(D) Eth1 and Eth3

Homework 4: Prefix range and subnetting

Q1. Consider a scenario where host addresses are 16 bits long. A router uses longest prefix matching and has the following entries in its forwarding table:

Prefix Match	Interface
1100	Eth0
11001	Eth1
111	Eth2
Default	Eth3

For each of the four interfaces, give the associated range of destination host addresses and the number of addresses in the range.

Q2. Consider a subnet with prefix 10.20.15.128/26. Give an example of an IP address (of the form xxx.xxx.xxx.xxx) that can be assigned to this network. Suppose an ISP owns the block of addresses of the form 10.20.12.0/22. Suppose it wants to create two subnets from this block, with each block having the same number of IP addresses. What are the prefixes (of the form a.b.c.d/x) for the two subnets?

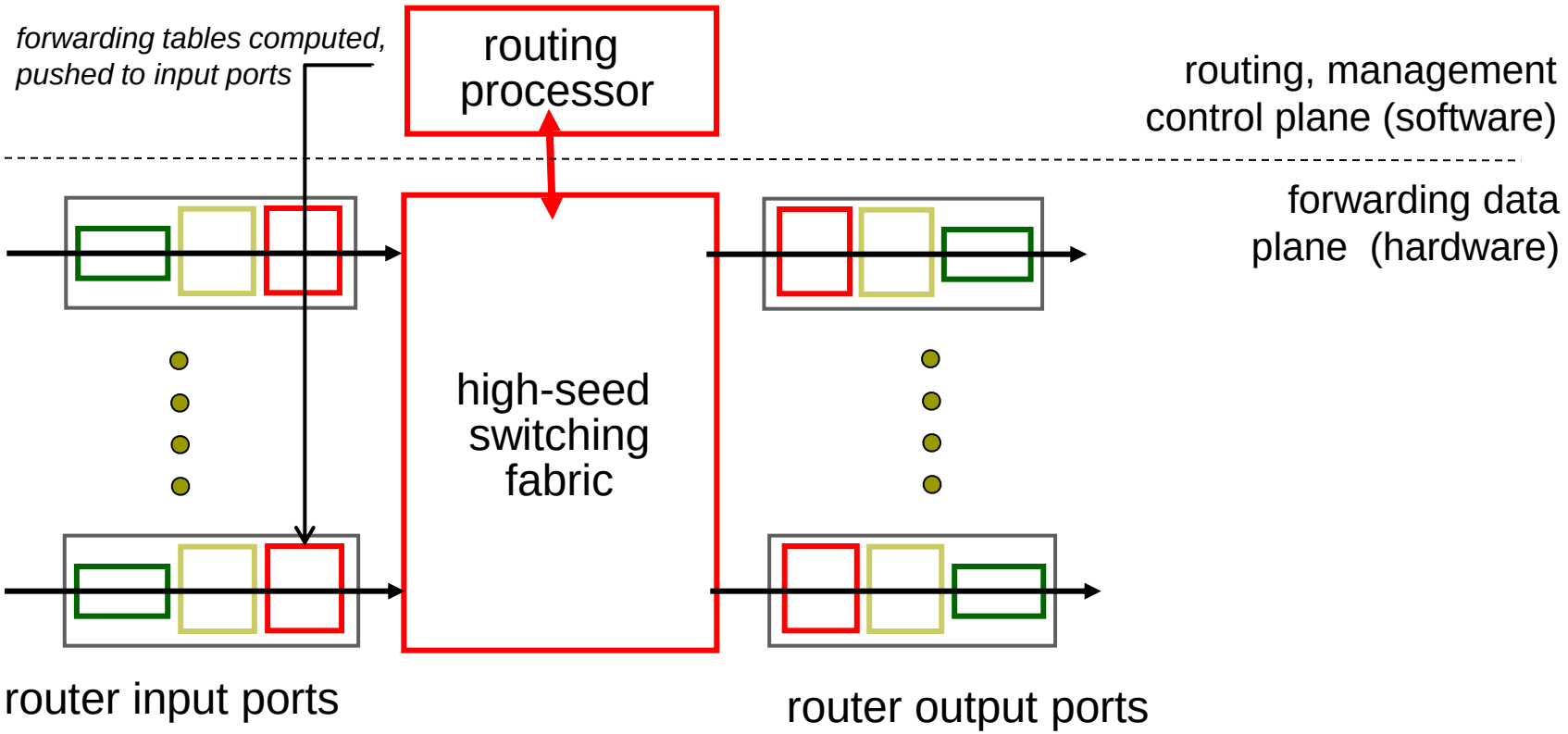
Creation of the routing tables

- So, where do these routing tables come from?
- They are not created on-the-fly:
  - Some routes are learned from other routers
  - Some routes are manually set by administrators
  - Existing routes are often updated over time
- They are exchanged via protocols such as
  - BGP (between different autonomous systems)
  - OSPF (within the same autonomous system)
- More on algorithms and protocols used later

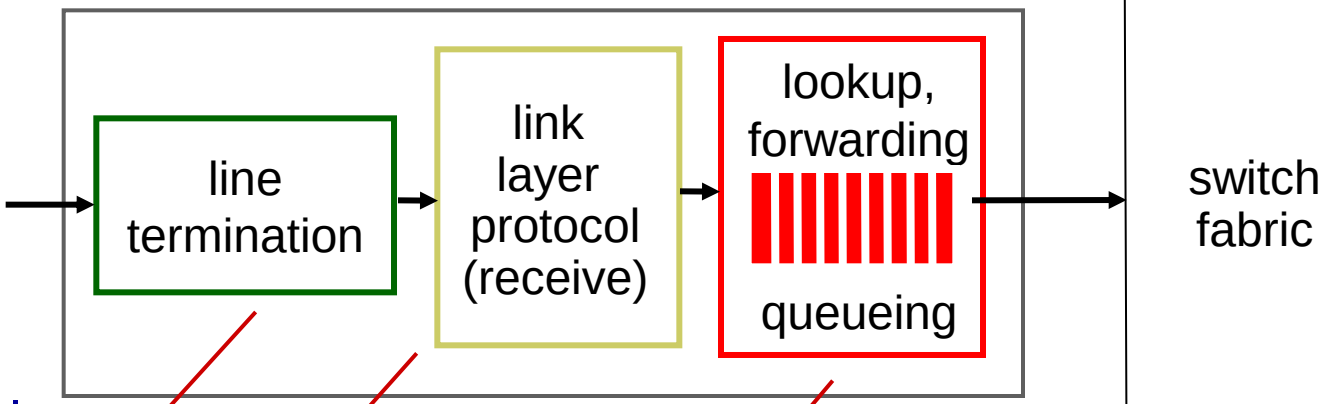
Router architecture overview

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

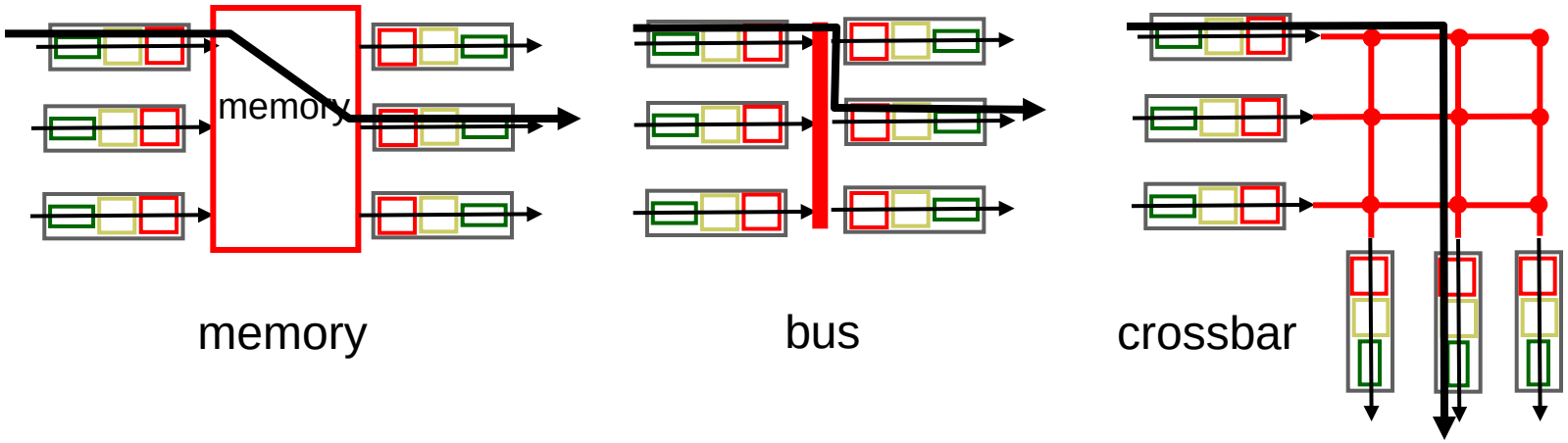
data link layer:
e.g., Ethernet
see chapter 5

decentralized switching:

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

Switching fabrics

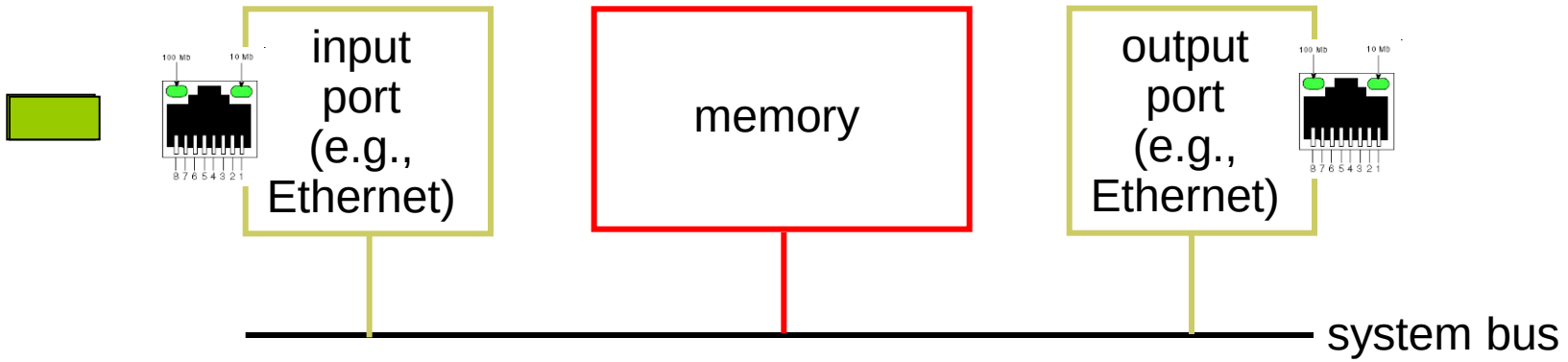
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Switching via memory

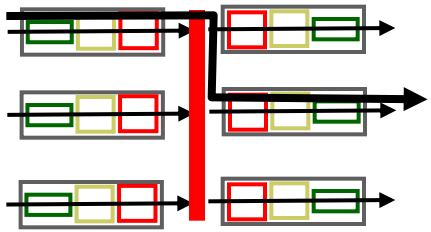
first generation routers:

- 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

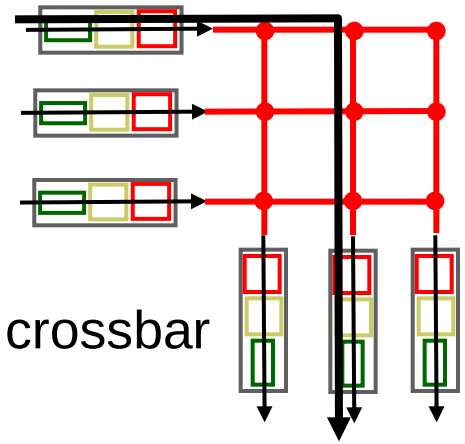
- datagram from input port memory to output port memory via a shared bus
- *bus contention:* switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers



bus

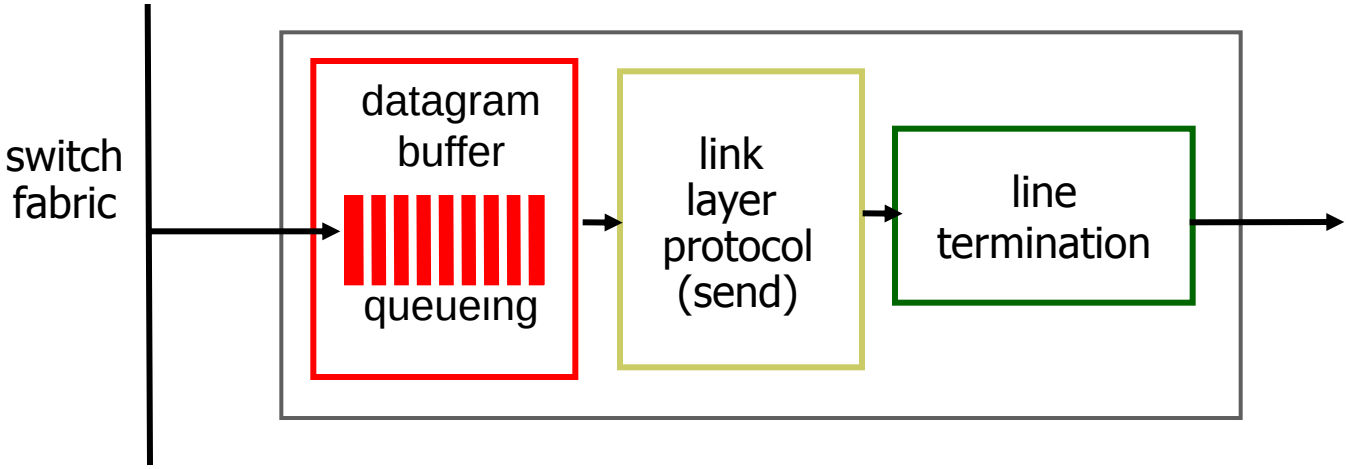
Switching via interconnection network

- 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!

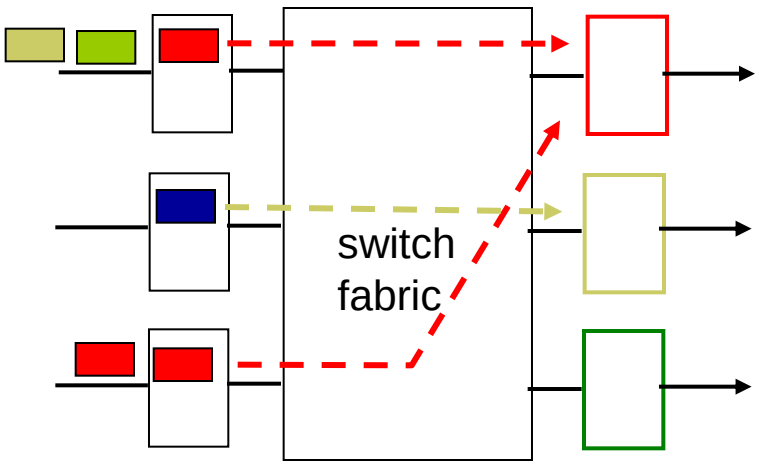


- *buffering* required when datag
  transmission rate
- *scheduling discipline* chooses
  transmission

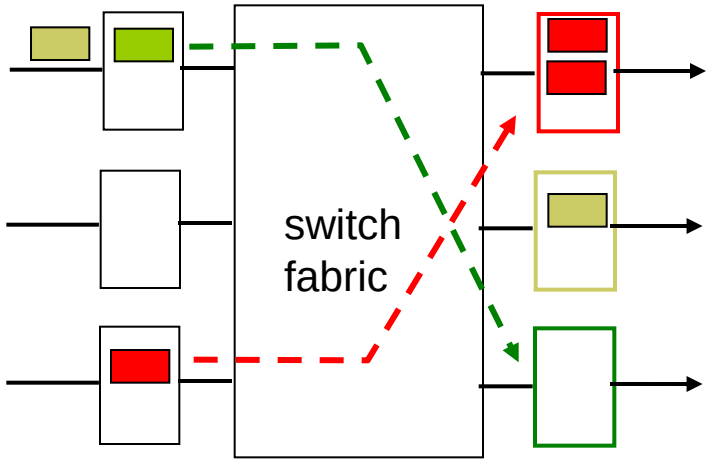
Datagram (packets) can be lost due to congestion, lack of buffers

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?

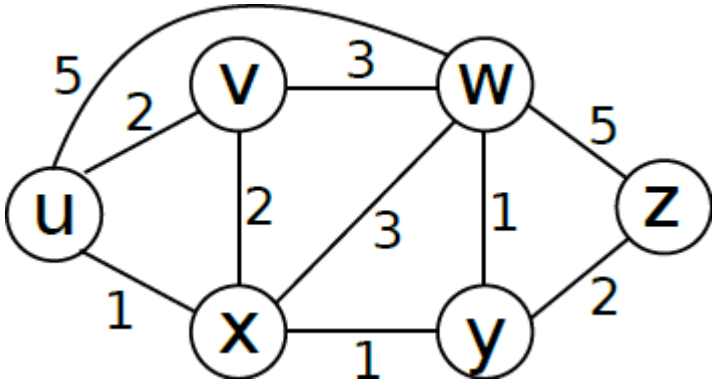
- RFC 3439 rule of thumb: average buffering equal to “typical” RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gpbs link: 2.5 Gbit buffer
- recent recommendation: with *N* flows, buffering equal to

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

Solving the Routing Problem

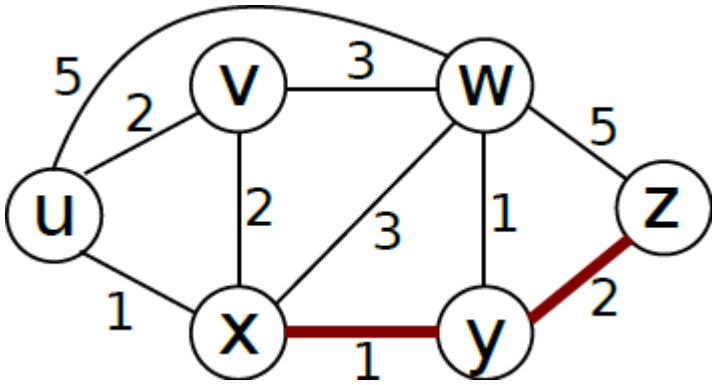
Graphs

- Quick recap on graph data structures:
  - G={N,E} (N=Nodes and E=Edges)
- When using in a network context:
  - Nodes are simply our hosts (routers, PCs, servers)
  - Network links are undirected edges with weights
  - E={(x,y),(x,z),(y,z)}
  - Weights are costs of that link, a label to each tuple
- Graphs can also be used to represent more abstract links
  - TCP streams
  - Overlay networks



Costs of links and paths

- A link is an edge in the graph, represented by 2-tuple
- A path is a sequence of one or more edges, an n-tuple
- A cost function for a network will yield cost of link or path
  - Cost can be related to delay, inversely to bandwidth, or congestion
- Usually, the cost of links can be added up:
  - If c(x,y)=1 and c(y,z)=2, then c(x,z)=3 (if this is shortest path)
- The routing algorithms will try to find a minimal cost path between two hosts



Routing Algorithms

Two classes of algorithms

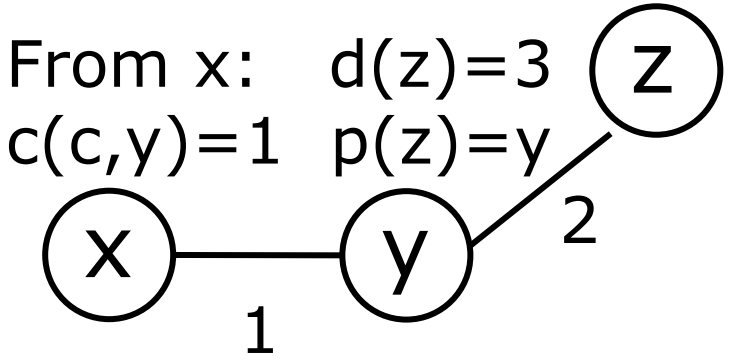
- Global view: shared global info on topology state, cost
  - called link state algorithms
- Local view: every host has own view, must talk to neighbors
  - Only topology and cost to neighbors is known
  - Algorithms typically converge over time
  - called distance vector algorithms
- path vector algorithms mix both:
  - Routers announce routes throughout the network
  - Intermediate routers forward shortest routes and add themselves
- The system can be considered static or dynamic
  - In the latter case, partial updates of routes should be possible

The shortest path problem

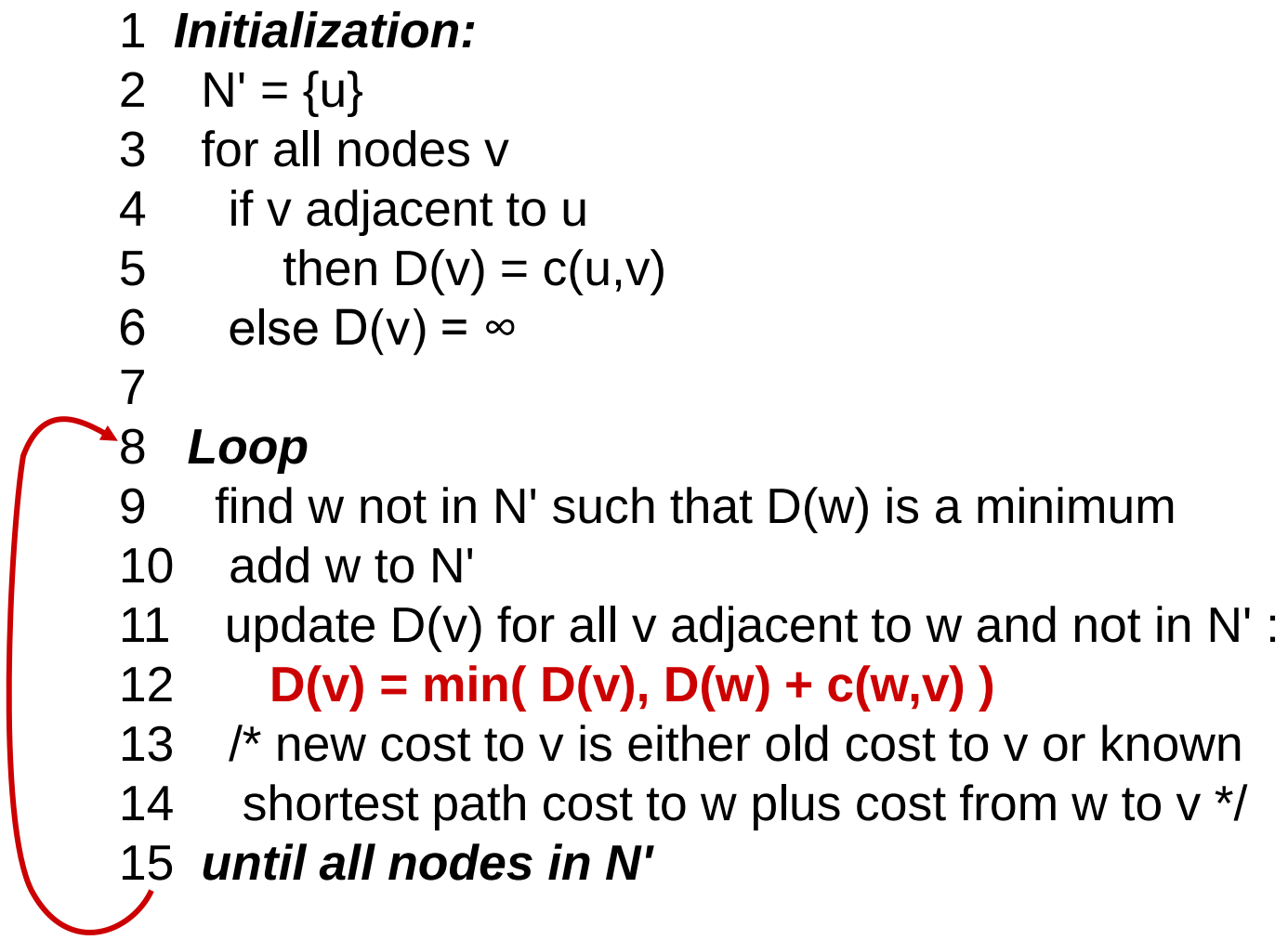
- The single-pair shortest path problem :
- Find the shortest path from x to y
  - Edges have weights
  - All nodes are connected to graph
- Related problems:
  - Single source shortest path (from x to each other node)
  - Single destination shortest path (to y from each other node)
  - All-pairs shortest path: single-pair for all possible pairs
- Cost for single-pair: O(E + V log V) (Dijkstra's algorithm)

Dijkstra's algorithm setup

- Link state algorithm, complete network has to be known
- Iterative algorithm that builds a set of shortest paths
- *d(v)* is the currently known cost of the shortest path from *x* to *v*
- *p(v)* is the predecessor of v on the currently shortest path from *x* to *v*
- *N':* set of nodes with known absolute shortest path



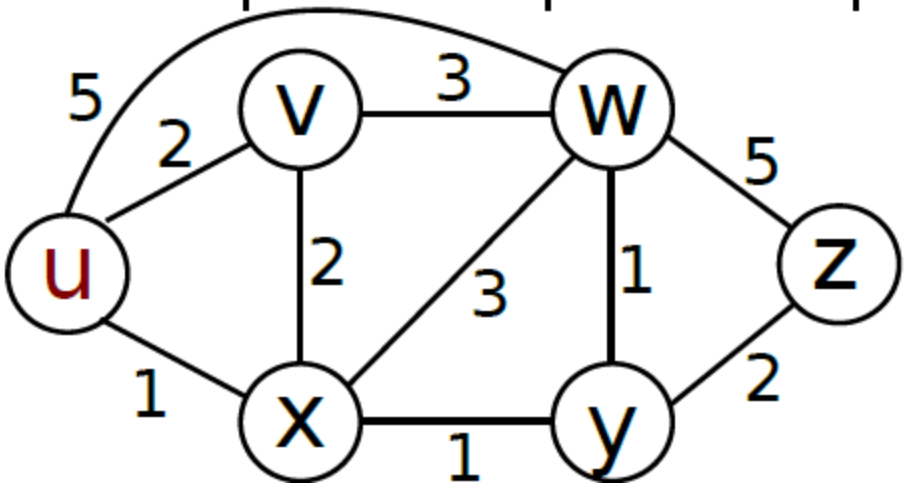
Dijkstra's algorithm



```
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'
```

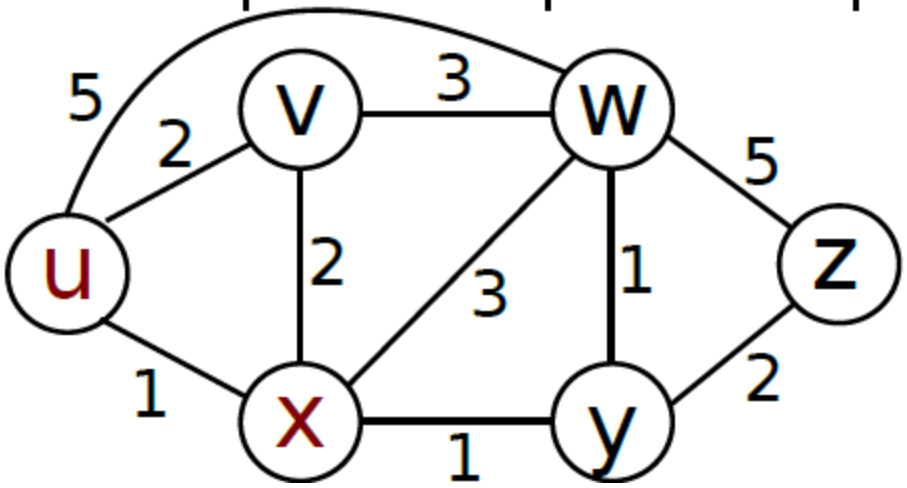
Dijkstra example

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	infty	infty



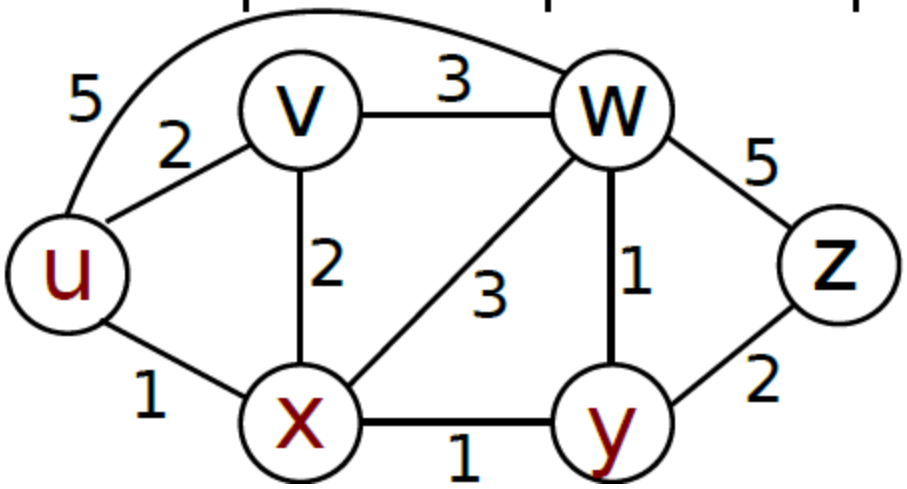
Dijkstra example

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	infty	infty
1	ux	2,u	4,x		2,x	infty



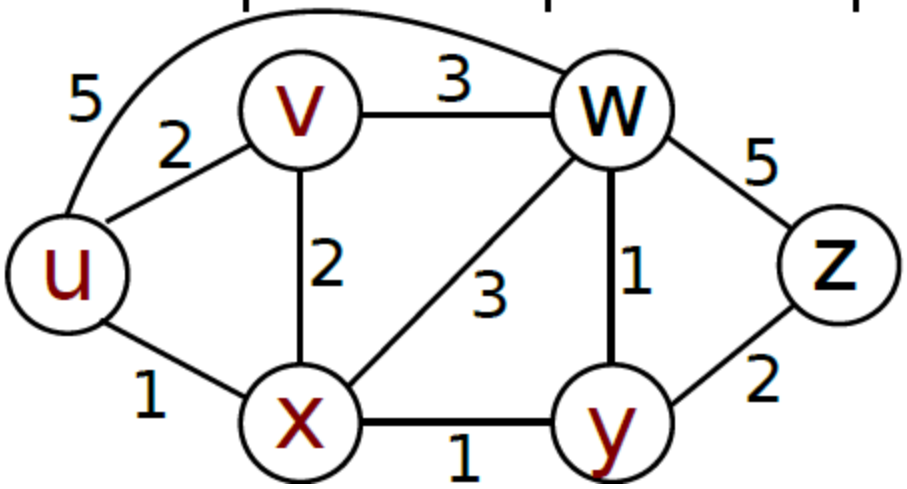
Dijkstra example

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	infty	infty
1	ux	2,u	4,x		2,x	infty
2	uxy	2,u	3,y			4,y



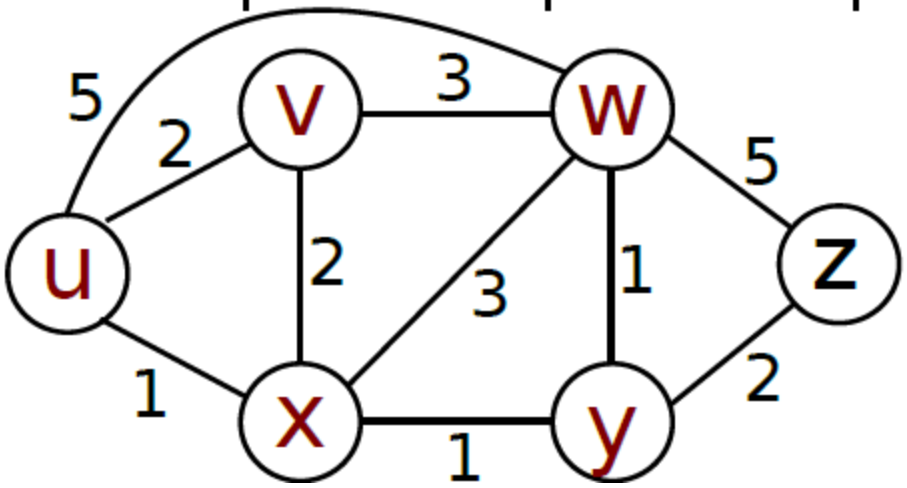
Dijkstra example

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	infty	infty
1	ux	2,u	4,x		2,x	infty
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y



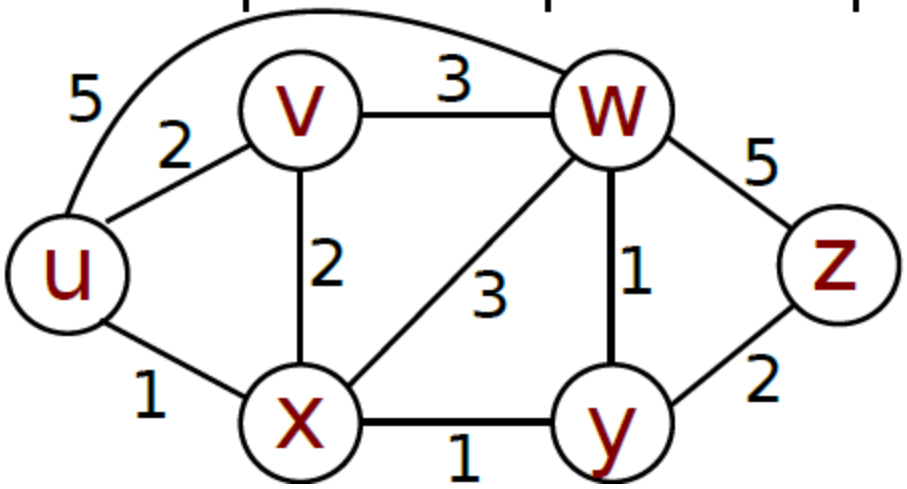
Dijkstra example

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	infty	infty
1	ux	2,u	4,x		2,x	infty
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y



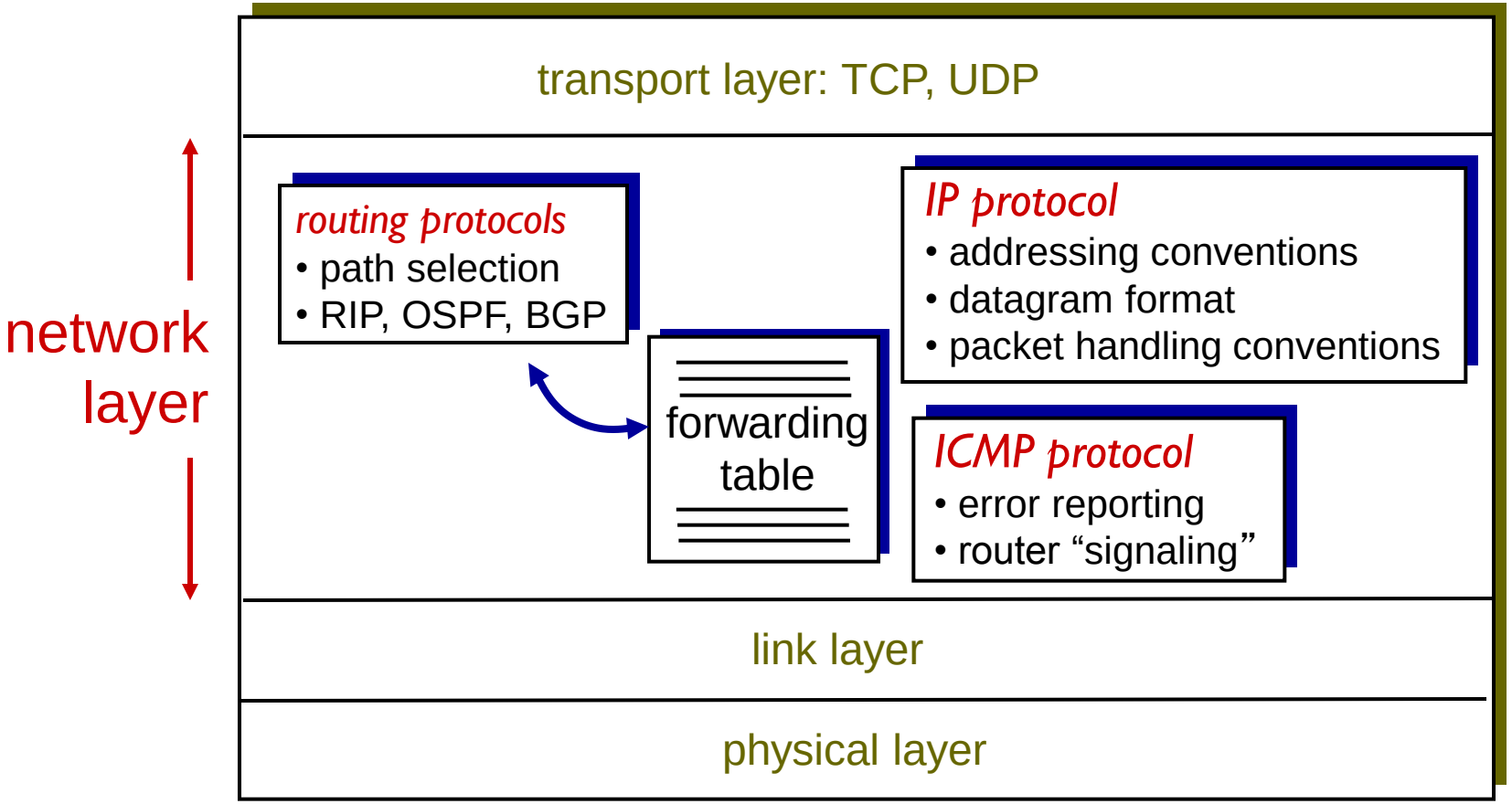
Dijkstra example

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	infty	infty
1	ux	2,u	4,x		2,x	infty
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



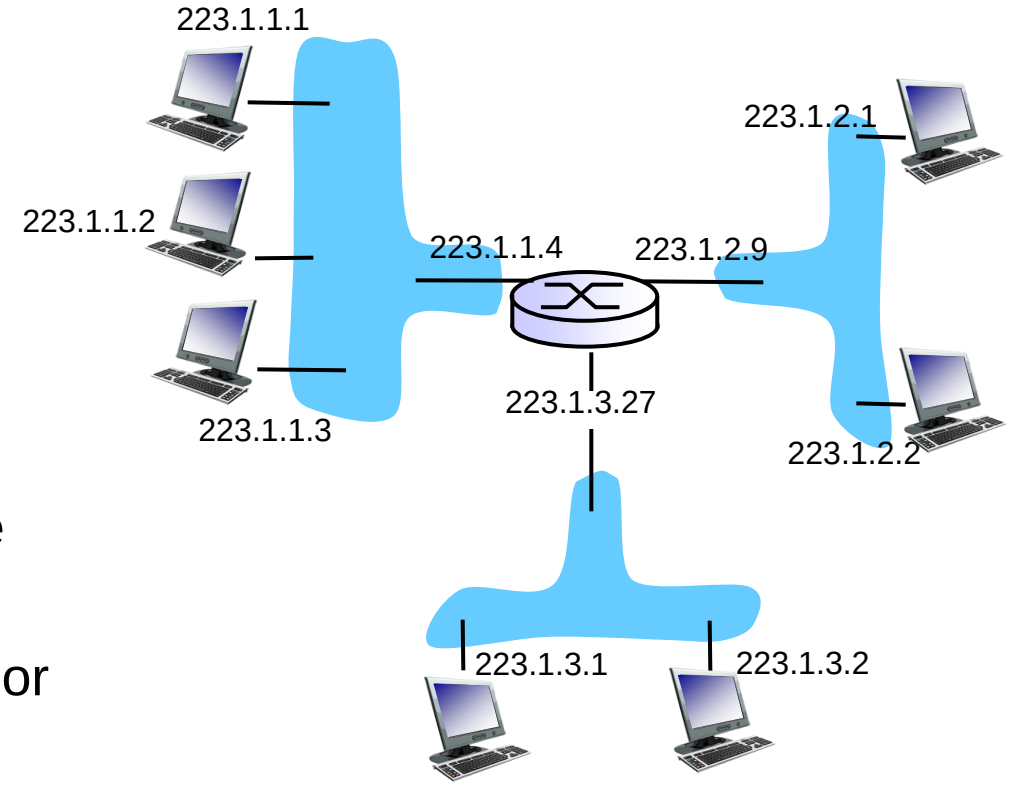
The Internet network layer

host, router network layer functions:



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*



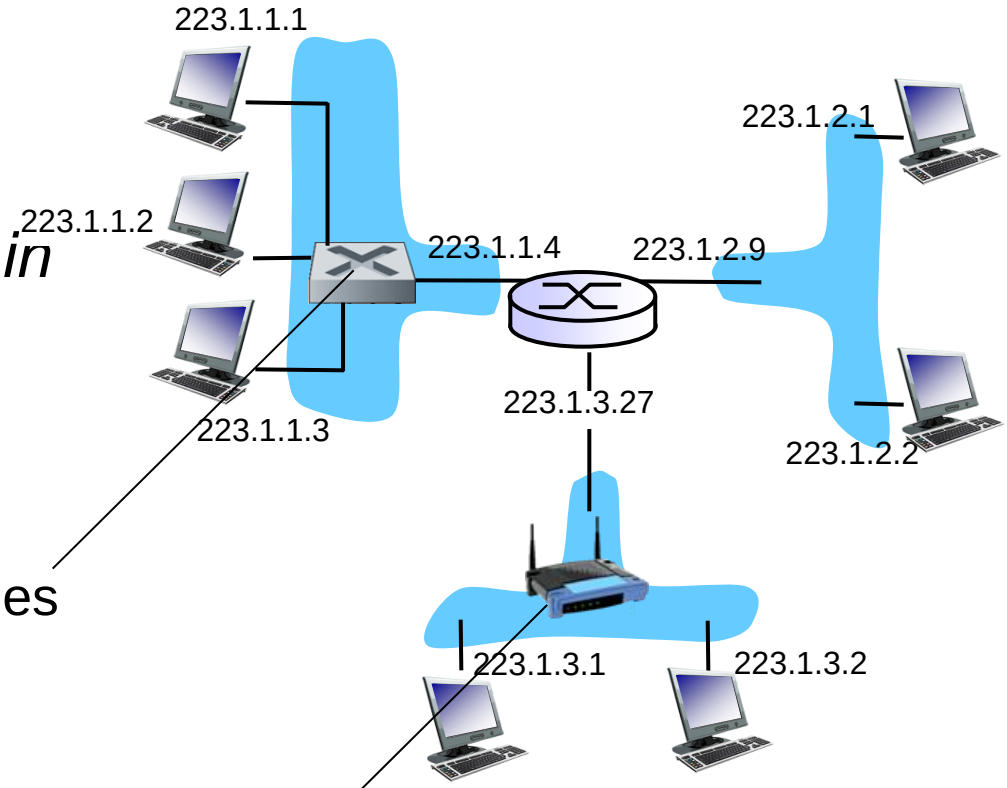
223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

IP addressing: introduction

Q: how are interfaces actually connected?

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



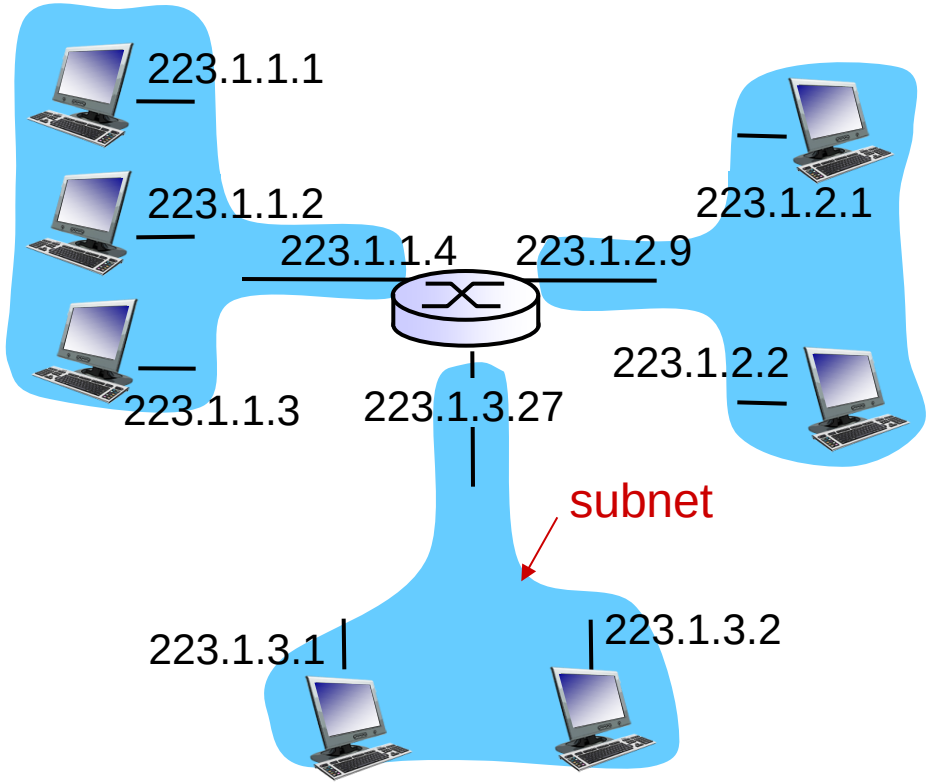
A: wired Ethernet interfaces connected by Ethernet switches

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*

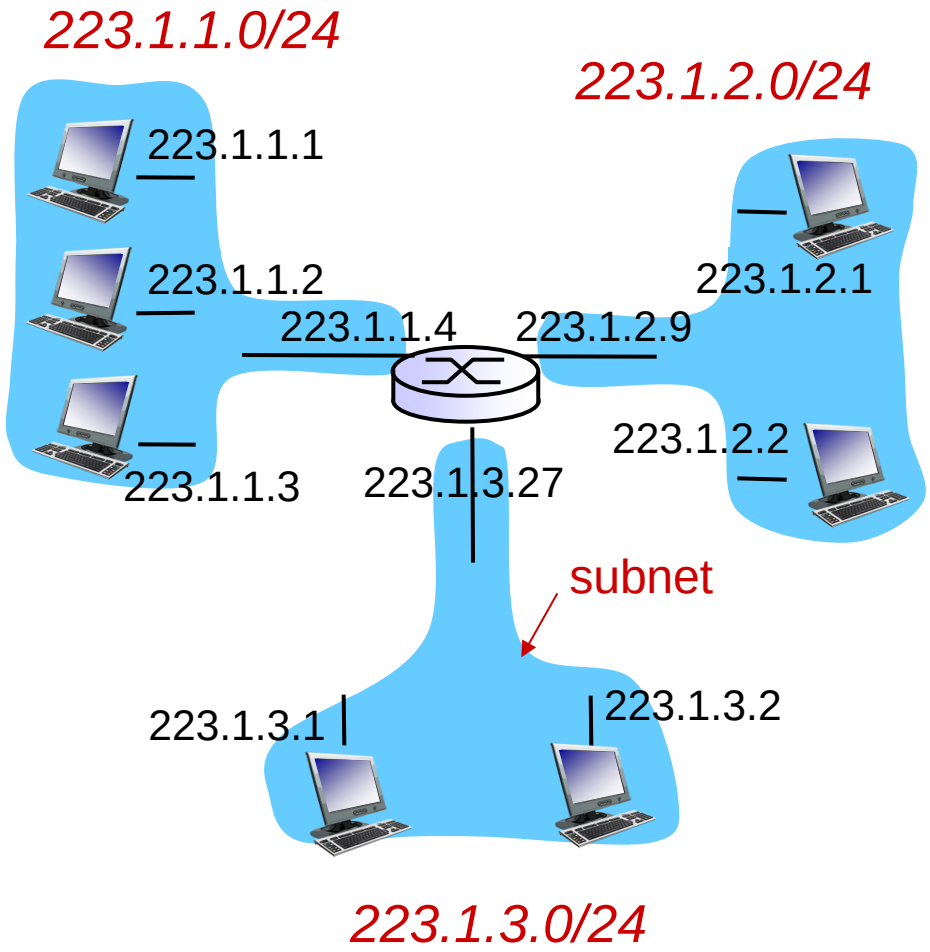


network consisting of 3 subnets

Subnets

recipe

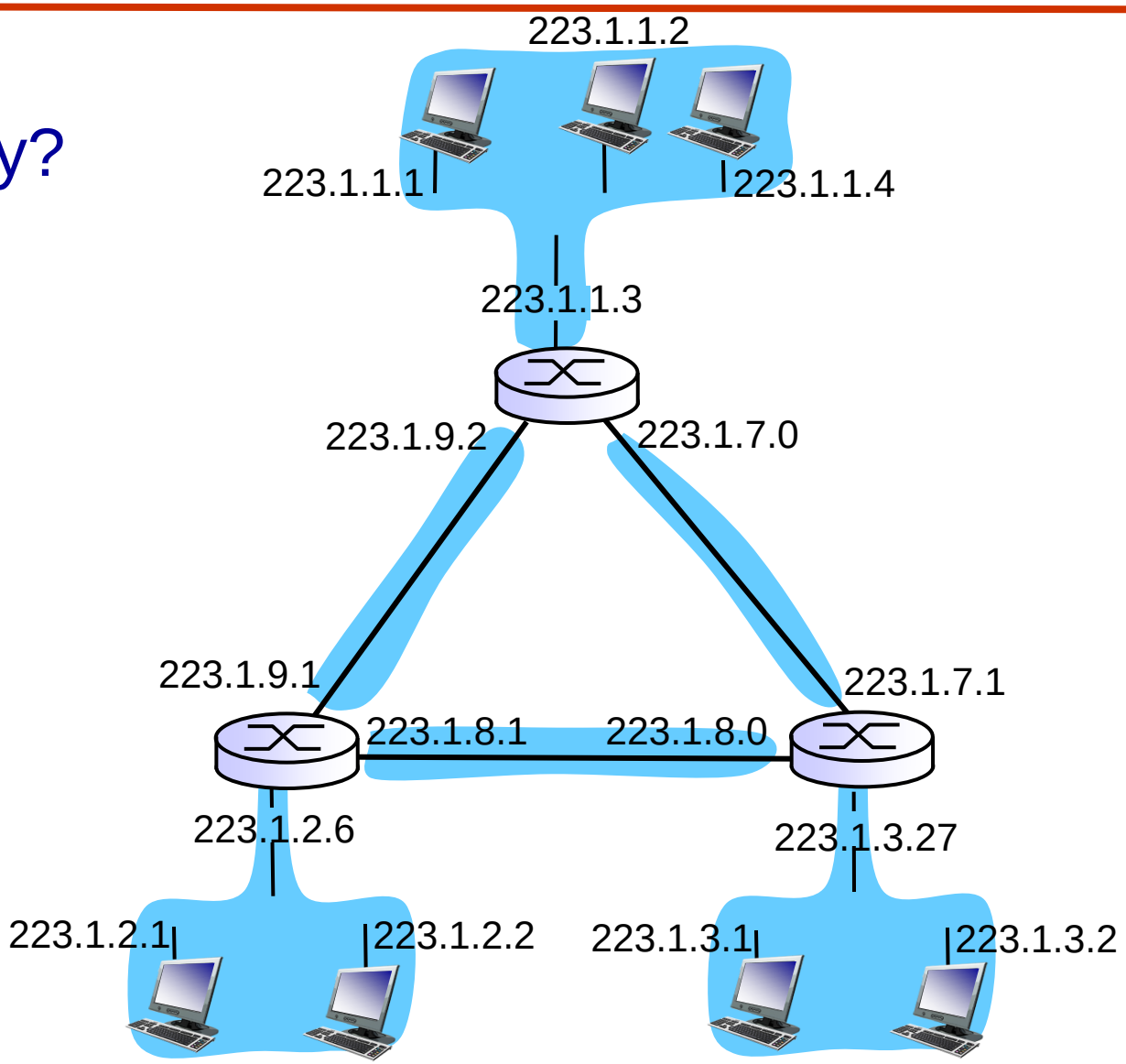
- to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- each isolated network is called a *subnet*



subnet mask: /24

Subnets

how many?



Conclusion

- We discussed the two main functions of the Network layer:
  - Forwarding within switches
  - Routing between switches on the Internet
- Forwarding within a switch is again a networking problem
  - With static topology and real-time requirements
- Routing is a field with rich theoretical work
  - We started with Dijkstra's algorithm today, more in next lecture