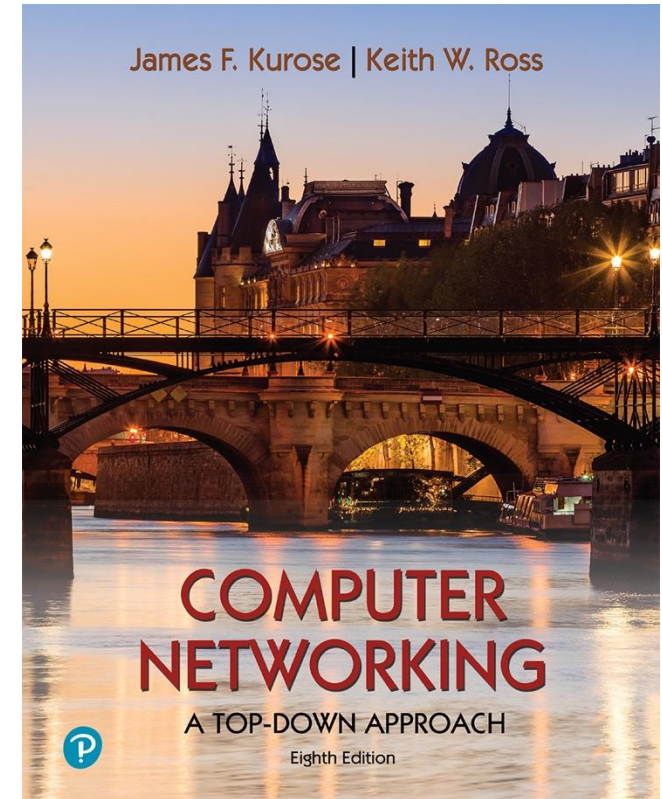# Computer Networks

Amir Mahdi Sadeghzadeh, Ph.D.

Chapter 4
Network Layer:
Data Plane

Computer Networking: A Top-Down Approach

8th edition
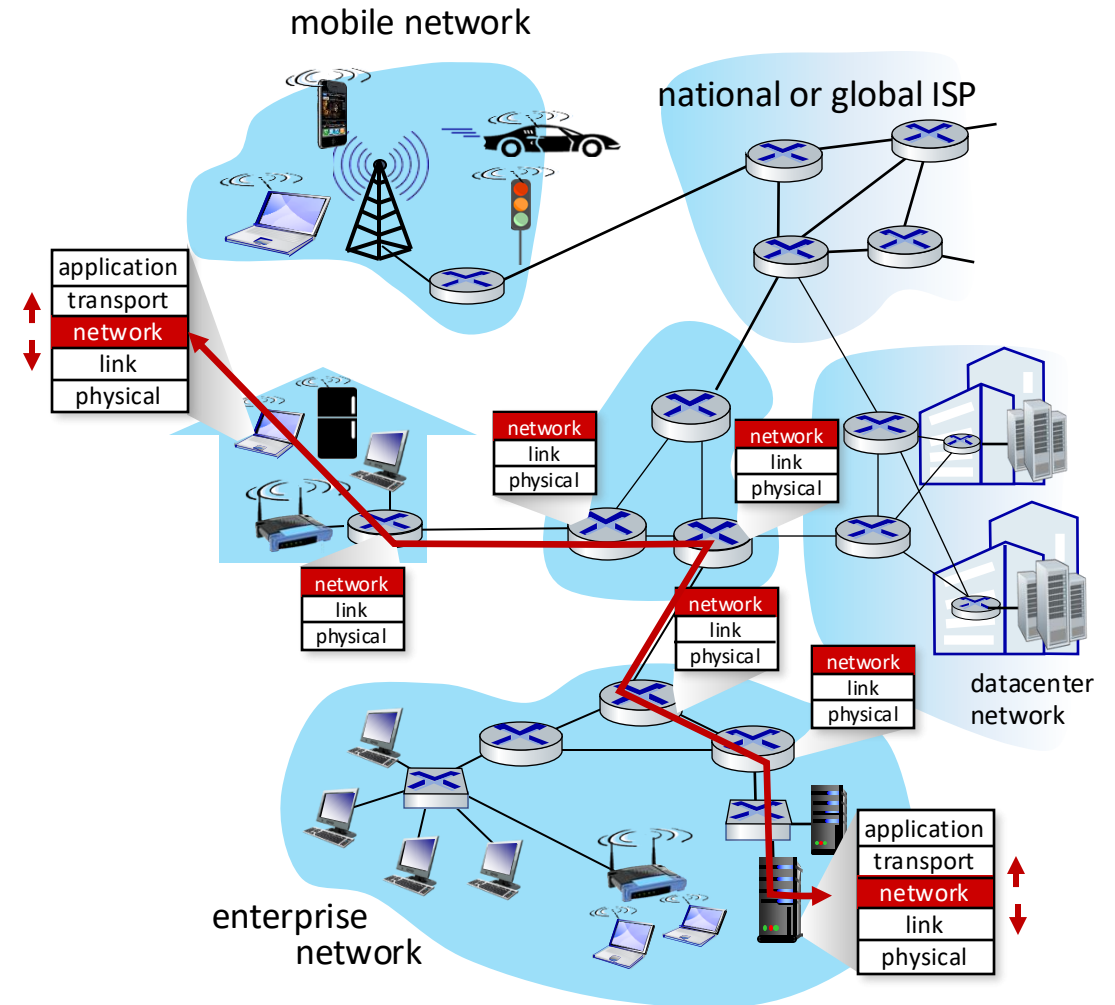Jim Kurose, Keith Ross
Pearson, 2020

Network layer: "data plane" roadmap

- **Network layer: overview**
  - data plane
  - control plane

- **What's inside a router**
  - input ports, switching, output ports
  - buffer management, scheduling

- **IP: the Internet Protocol**
  - datagram format
  - addressing
  - network address translation
  - IPv6

- **Generalized Forwarding, SDN**
  - Match+action
  - OpenFlow: match+action in action

- **Middleboxes**

Network-layer services and protocols

- **transport segment from sending to receiving host**
  - sender: encapsulates segments into datagrams, passes to link layer
  - receiver: delivers segments to transport layer protocol
- **network layer protocols in *every Internet device*: hosts, routers**
- **routers:**
  - examines header fields in all IP datagrams passing through it
  - moves datagrams from input ports to output ports to transfer datagrams along end-end path

mobile network

national or global ISP

| application |
| transport |
| network |
| link |
| physical |

| network |
| link |
| physical |

| network |
| link |
| physical |

| network |
| link |
| physical |

| network |
| link |
| physical |

| network |
| link |
| physical |

datacenter network

| application |
| transport |
| network |
| link |
| physical |

enterprise network

Two key network-layer functions

network-layer functions:

- *forwarding:* move packets from a router's input link to appropriate router output link

- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*

analogy: taking a trip

- *forwarding:* process of getting through single interchange

- *routing:* process of planning trip from source to destination
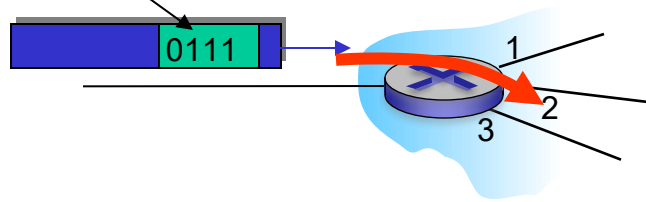

forwarding

routing

Network layer: data plane, control plane

Data plane:

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

values in arriving packet header

0111

1
2
3

Control plane

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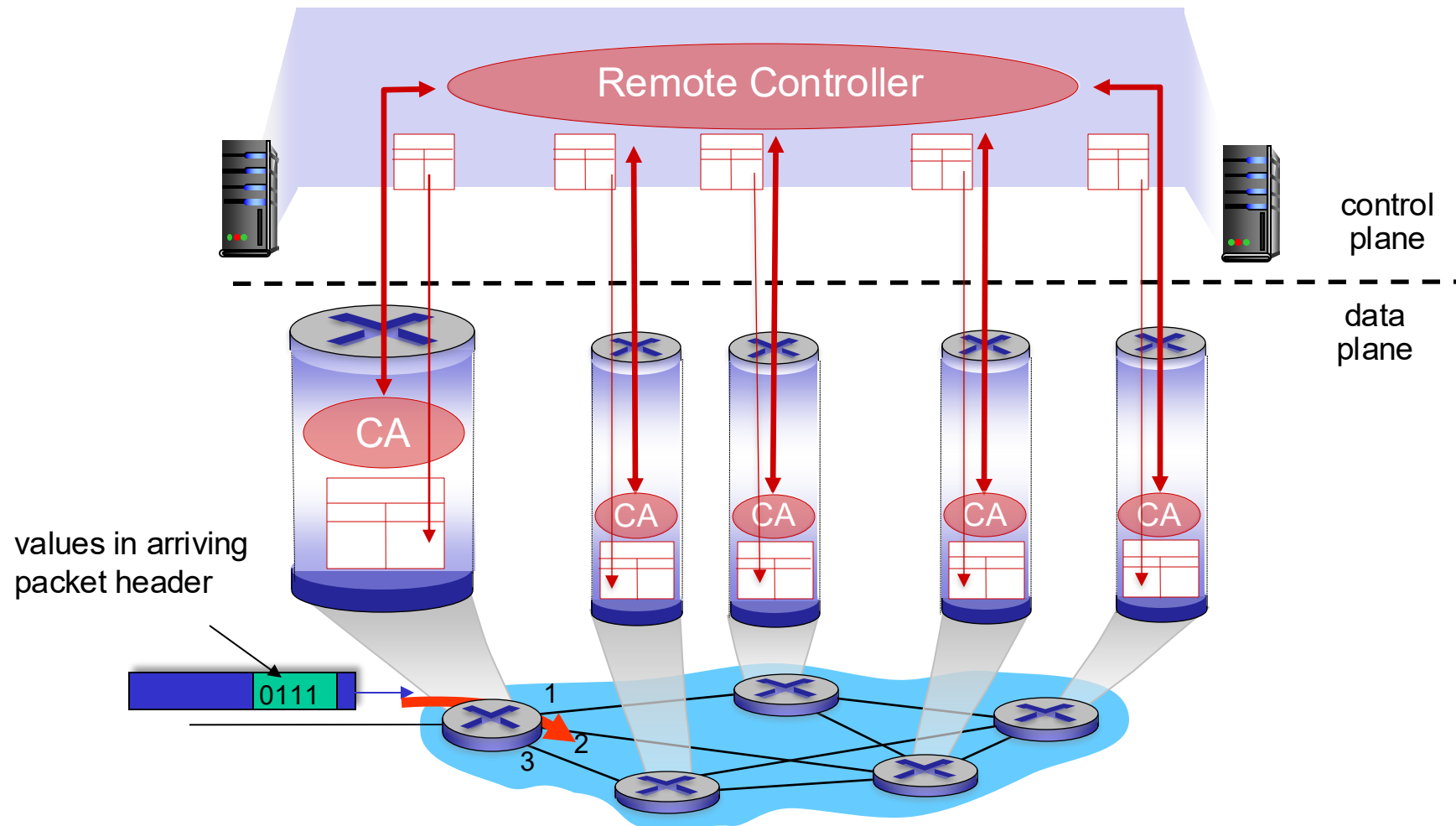 - *software-defined networking (SDN)*: implemented in (remote) servers

Per-router control plane

Individual routing algorithm components *in each and every router* interact in the control plane
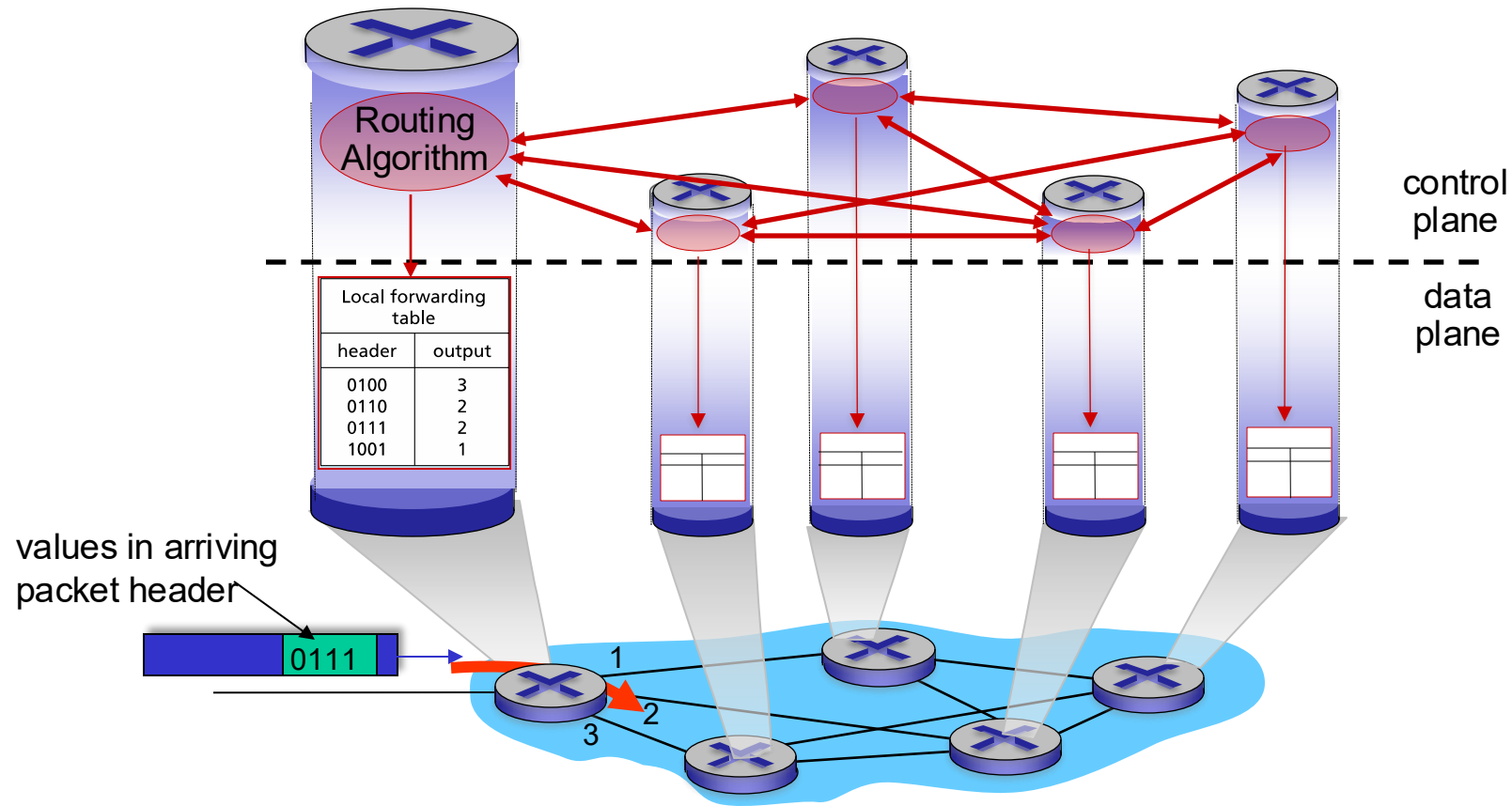
Software-Defined Networking (SDN) control plane

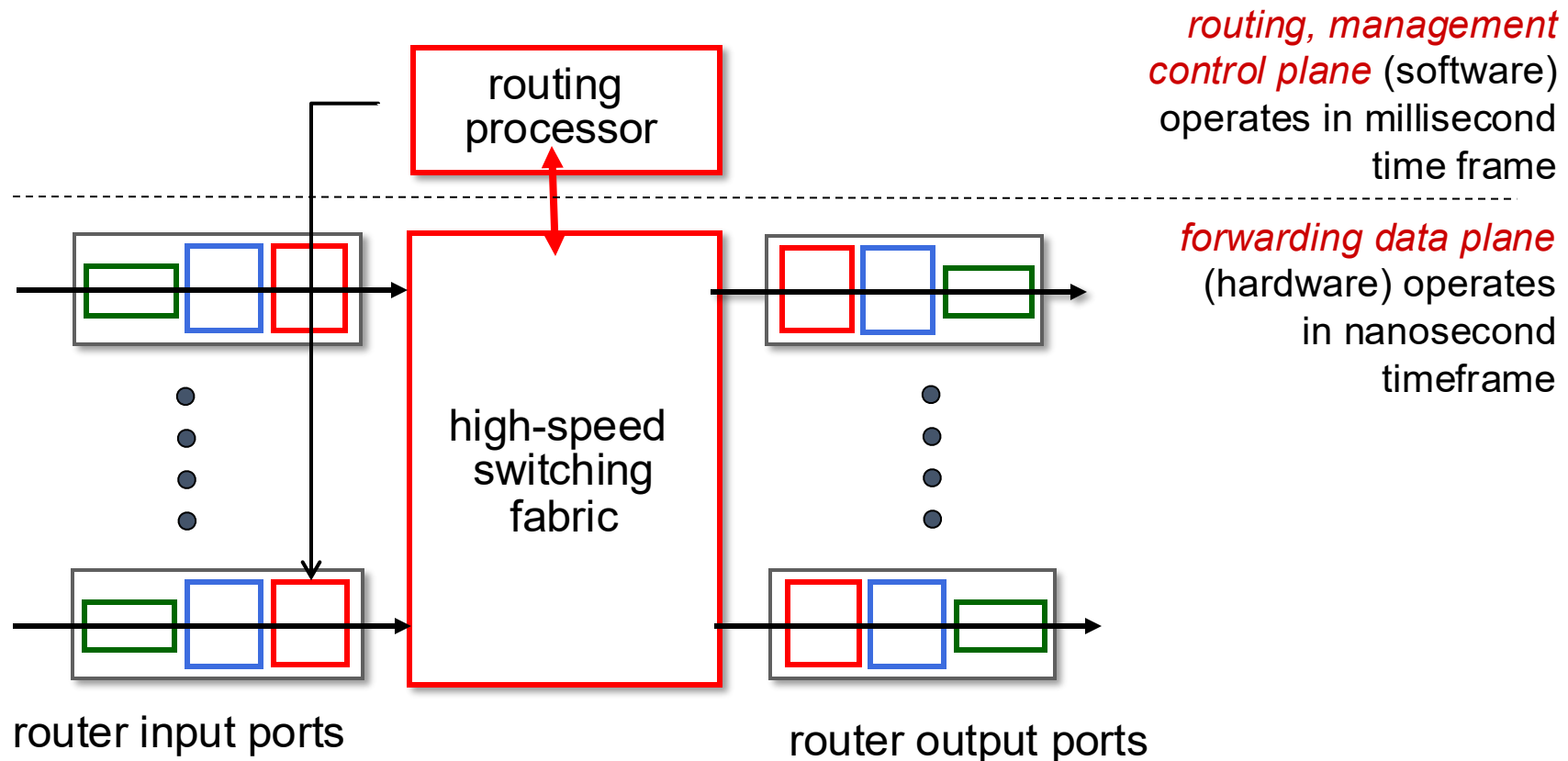Remote controller computes, installs forwarding tables in routers

Network layer: "data plane" roadmap

- Network layer: overview
  - data plane
  - control plane

- **What's inside a router**
  - input ports, switching, output ports
  - buffer management, scheduling

- IP: the Internet Protocol
  - datagram format
  - addressing
  - network address translation
  - IPv6

- Generalized Forwarding, SDN
  - Match+action
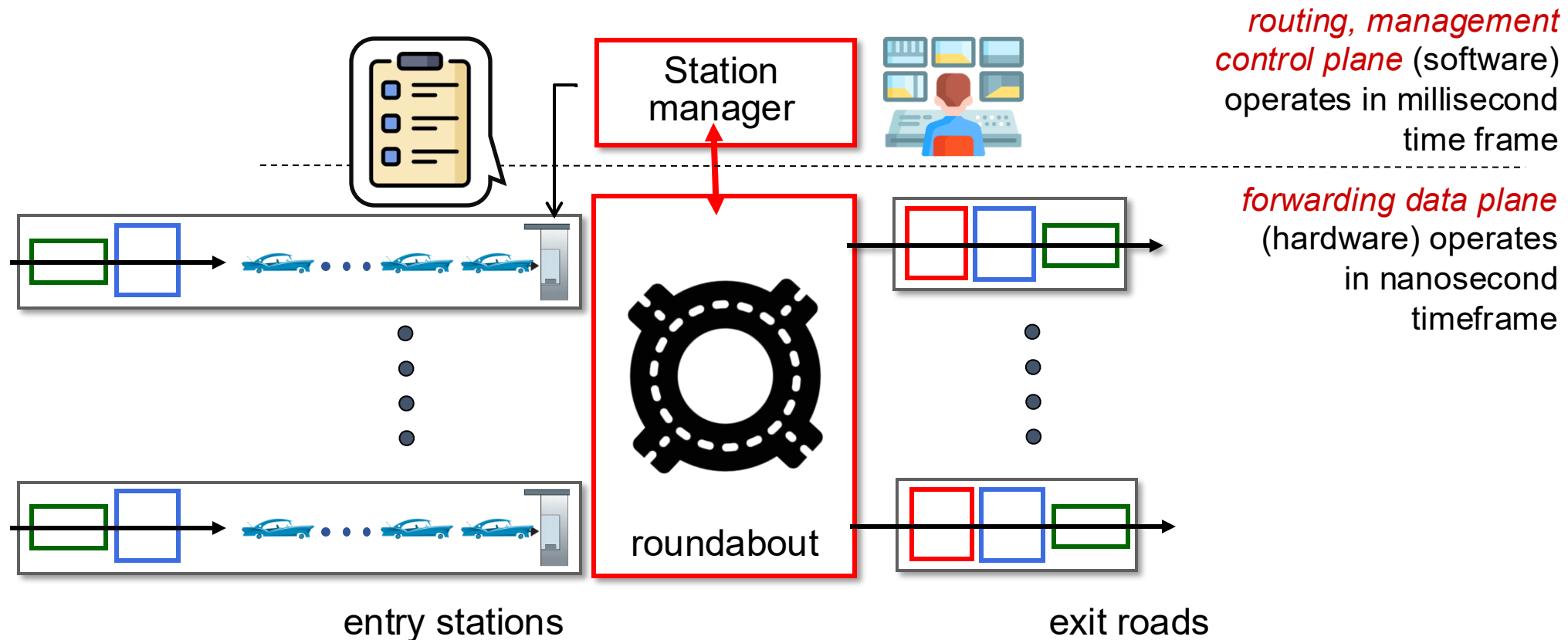 - OpenFlow: match+action in action

- Middleboxes

Router architecture overview

high-level view of generic router architecture:
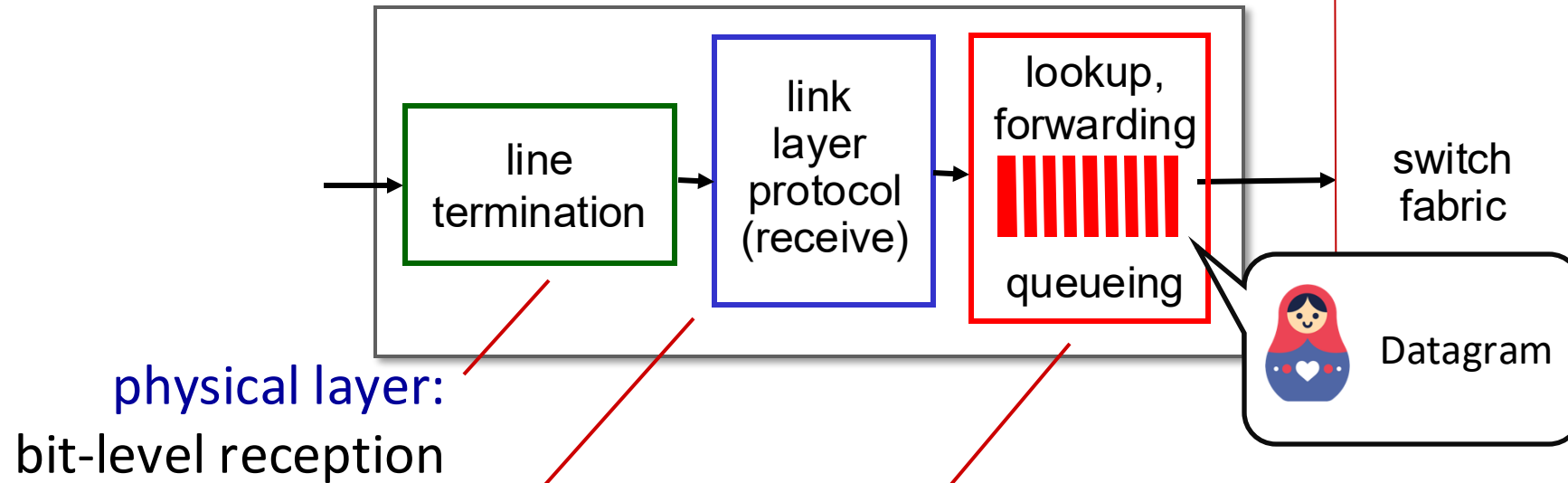

routing, management control plane (software) operates in millisecond time frame

forwarding data plane (hardware) operates in nanosecond timeframe

routing processor

high-speed switching fabric

router input ports

router output ports

Router architecture overview

analogy view of generic router architecture:

routing, management control plane (software) operates in millisecond time frame

forwarding data plane (hardware) operates in nanosecond timeframe

Station manager

roundabout

entry stations

exit roads

Input port functions

physical layer:
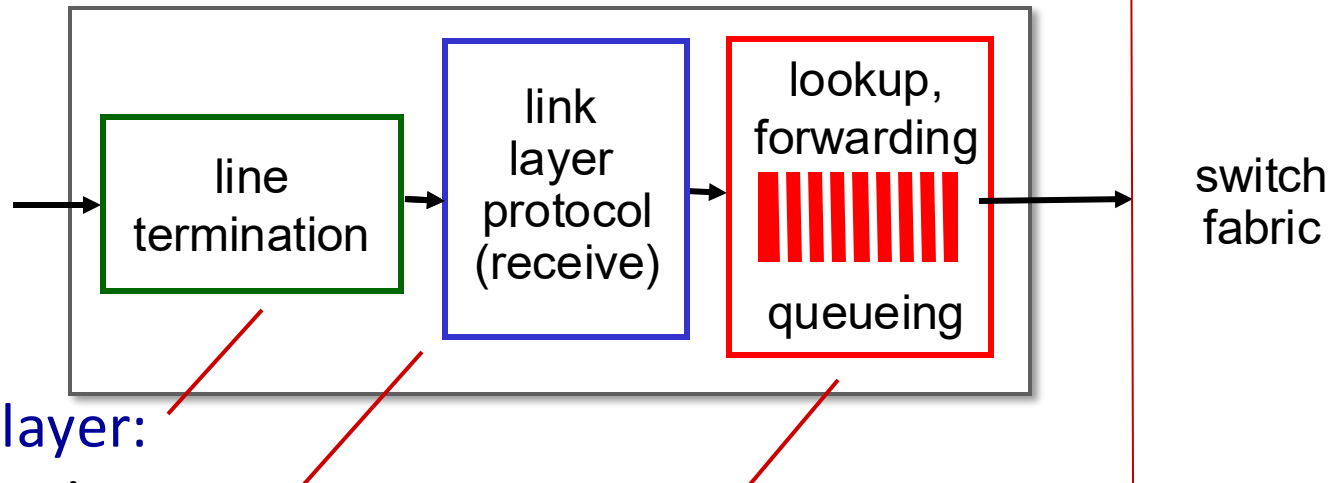bit-level reception

link layer:
e.g., Ethernet
(chapter 6)

Frame

decentralized switching:

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

Input port functions

line termination

link layer protocol (receive)

lookup, forwarding

queueing

switch fabric

physical layer:
bit-level reception

link layer:
e.g., Ethernet
(chapter 6)

decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- destination-based forwarding: forward based only on destination IP address (traditional)
- generalized forwarding: forward based on any set of header field values

Destination-based forwarding

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010000 00000111	0
11001000 00010111 00010000 00000100 through 11001000 00010111 00010000 00000111 11001000 00010111 00011000 11111111	3
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

—

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:

11001000	00010111	00010110	10100001	which interface?
11001000	00010111	00011000	10101010	which interface?

Longest prefix matching

longest prefix match

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

match!

examples:
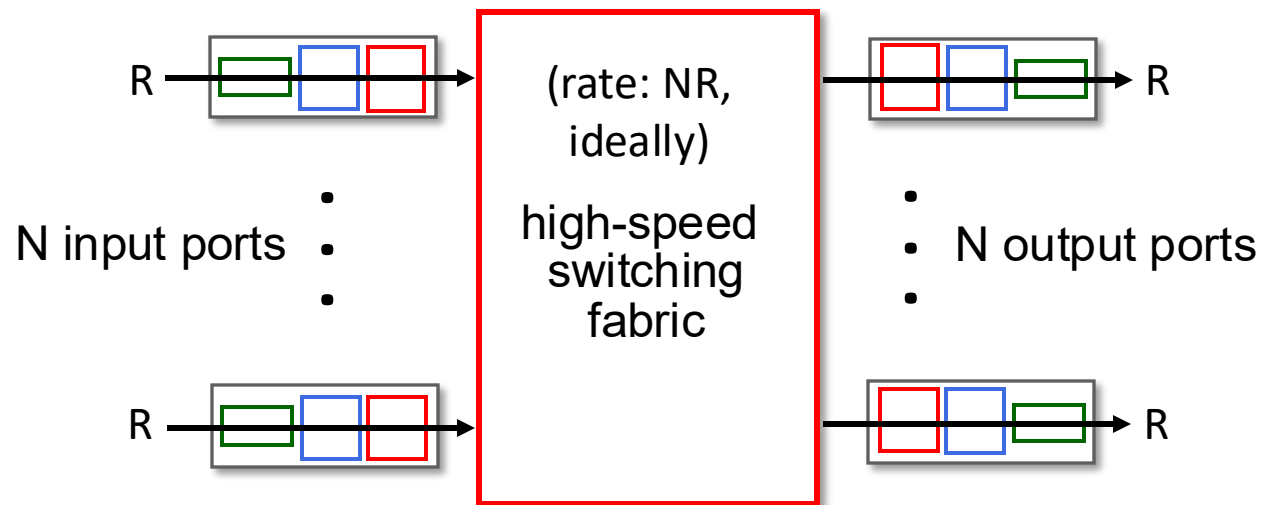
11001000   00010111   00010110   10100001   which interface?

11001000   00010111   00011000   10101010   which interface?

Longest prefix matching

longest prefix match

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

match!

examples:

11001000	00010111	00010110	10100001	which interface?
11001000	00010111	00011000	10101010	which interface?

Longest prefix matching

longest prefix match

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

match!

examples:

11001000	00010111	00010110	10100001	which interface?
11001000	00010111	00011000	10101010	which interface?

Longest prefix matching

- we'll see *why* longest prefix matching is used shortly, when we study addressing

- longest prefix matching: often performed using ternary content addressable memories (TCAMs)
  - *content addressable:* present address to TCAM: retrieve address in one clock cycle, regardless of table size
 - Cisco Catalyst: ~1M routing table entries in TCAM

Switching fabrics

- transfer packet from input link to appropriate output link
- 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

R → (rate: NR, ideally) high-speed switching fabric → R

N input ports

N output ports

R → high-speed switching fabric → R

Switching fabrics

- transfer packet from input link to appropriate output link
- 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 major types of switching fabrics:

memory

bus

interconnection network
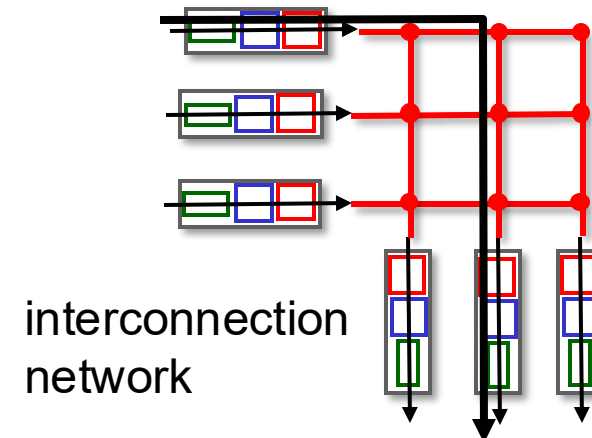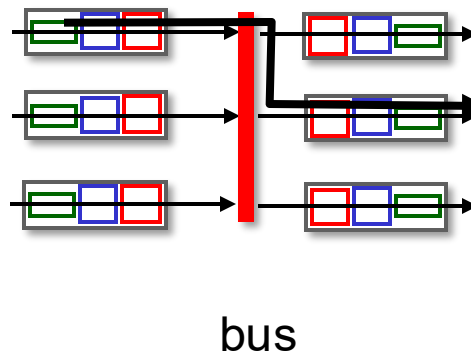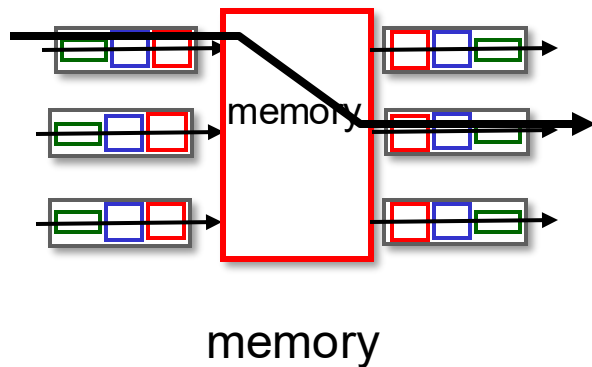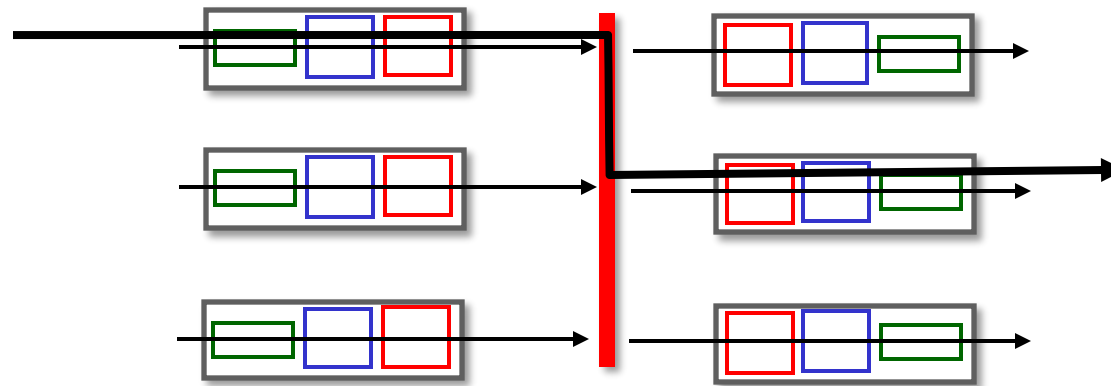
Switching via memory

first generation routers:

- traditional computers with switching under direct control of CPU

- packet copied to system's memory

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

input port (e.g., Ethernet) — memory — output port (e.g., Ethernet)

system bus
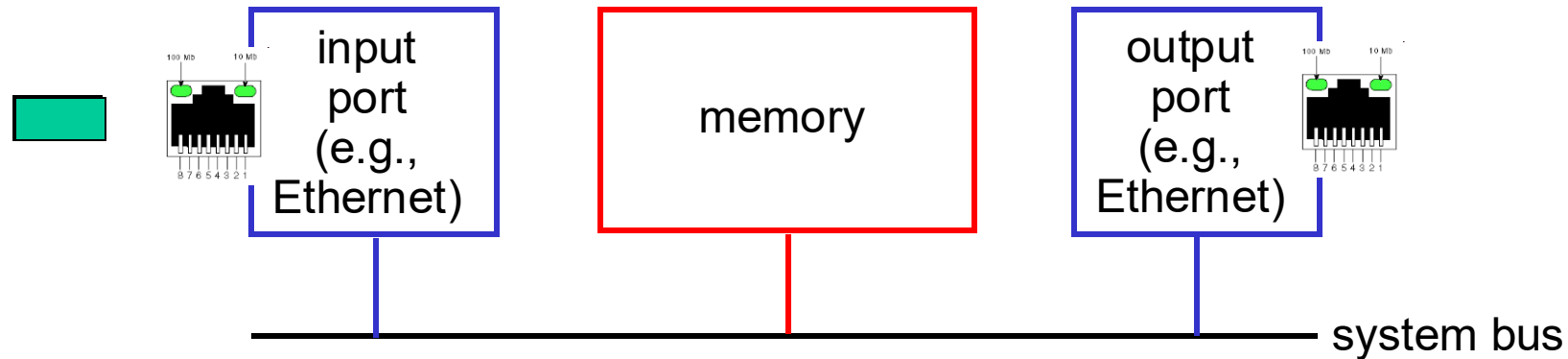
Switching via a bus

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

- *bus contention:* switching speed limited by bus bandwidth

- 32 Gbps bus, Cisco 5600: sufficient speed for access routers

Switching via interconnection network

- Crossbar, Clos networks, other interconnection nets initially developed to connect processors in multiprocessor


3x3 crossbar
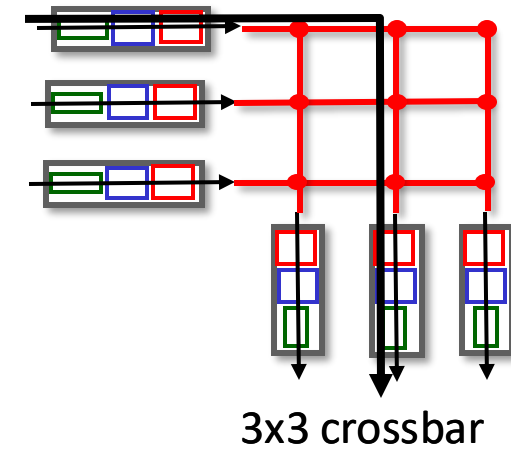
Input port queuing

- If switch 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
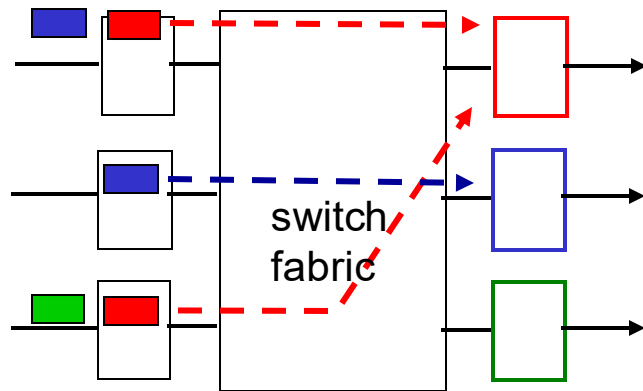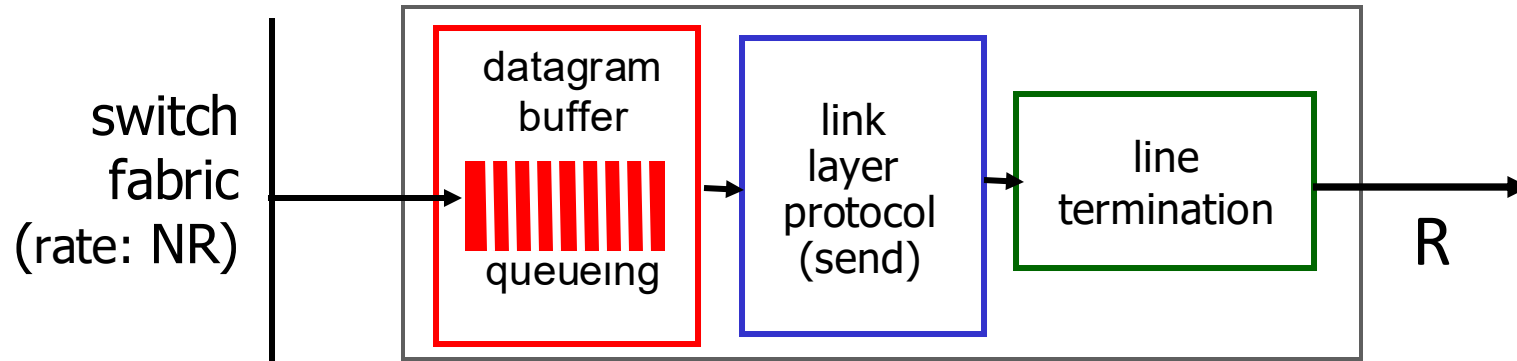

output port contention: only one red datagram can be transferred. lower red packet is *blocked*
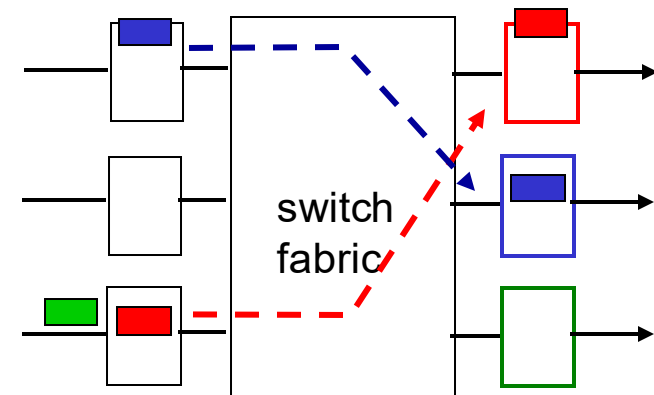
one packet time later: green packet experiences HOL blocking

Output port queuing

- *Buffering* required when datagrams arrive from fabric faster than link transmission rate. *Drop policy:* which datagrams to drop if no free buffers?
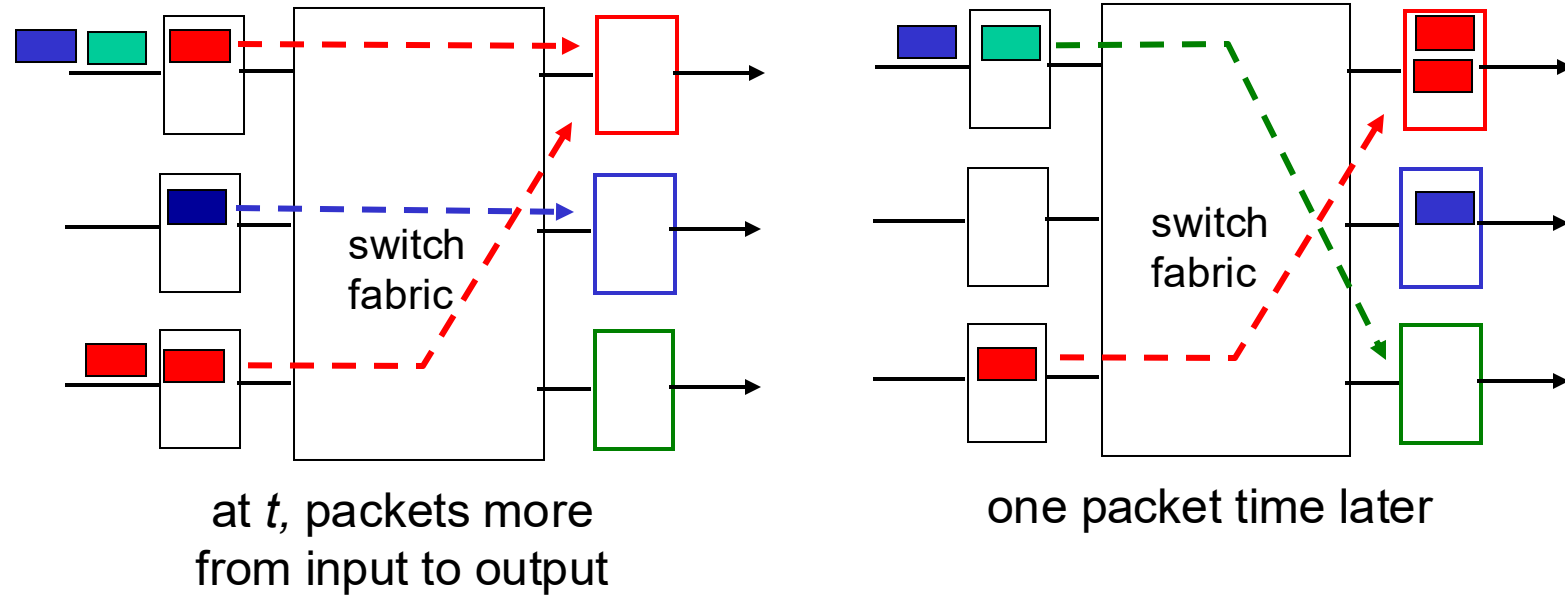
→ Datagrams can be lost due to congestion, lack of buffers

- *Scheduling discipline* chooses among queued datagrams for transmission

→ Priority scheduling – who gets best performance, network neutrality

Output port queuing

at *t*, packets more
from input to output

one packet time later

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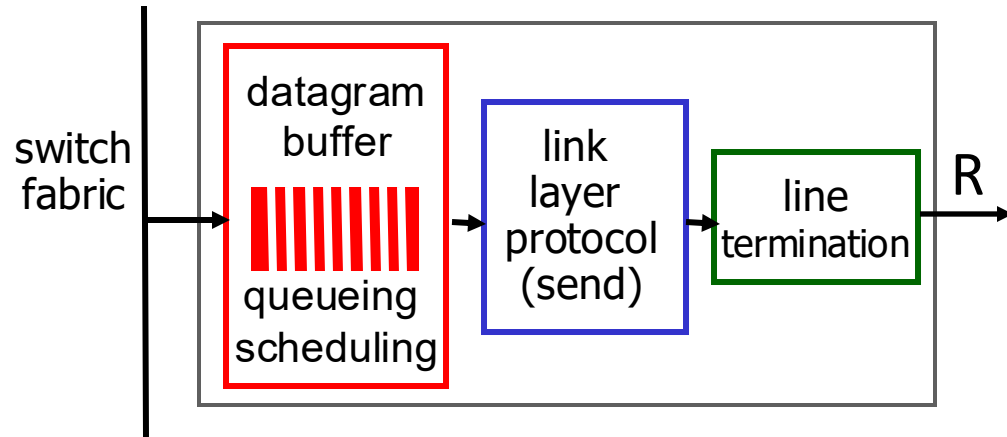- *queueing (delay) and loss due to output port buffer overflow!*

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 Gbps link: 2.5 Gbit buffer

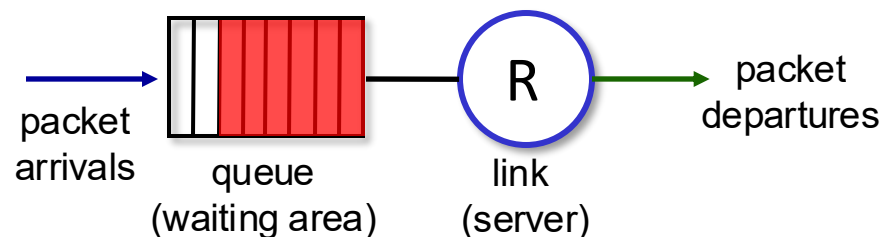- more recent recommendation: with *N* flows, buffering equal to

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

- but *too* much buffering can increase delays (particularly in home routers)
  - long RTTs: poor performance for real-time apps, sluggish TCP response
  - recall delay-based congestion control: "keep bottleneck link just full enough (busy) but no fuller"

Buffer Management

switch fabric →

datagram buffer
queueing scheduling

→ link layer protocol (send) → line termination → R

Abstraction: queue

packet arrivals →

queue (waiting area)

link (server) R → packet departures

buffer management:

- **drop:** which packet to add, drop when buffers are full
  - tail drop: drop arriving packet
  - priority: drop/remove on priority basis

- **marking:** which packets to mark to signal congestion (ECN, RED)
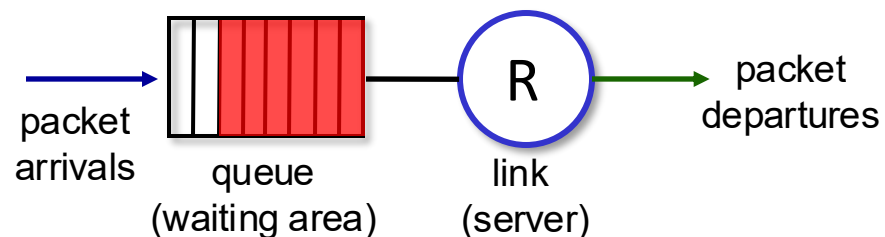
Packet Scheduling: FCFS

packet scheduling: deciding which packet to send next on link
- first come, first served
- priority
- round robin
- weighted fair queueing

FCFS: packets transmitted in order of arrival to output port
- also known as: First-in-first-out (FIFO)
- real world examples?

Abstraction: queue

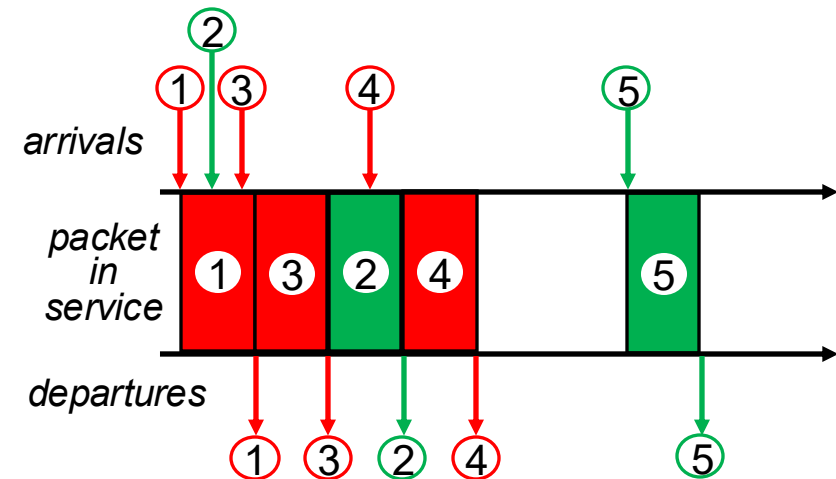packet arrivals

queue (waiting area)

link (server)

packet departures

Scheduling policies: priority

Priority scheduling:

- **arriving traffic classified, queued by class**
  - any header fields can be used for classification

- **send packet from highest priority queue that has buffered packets**
  - FCFS within priority class
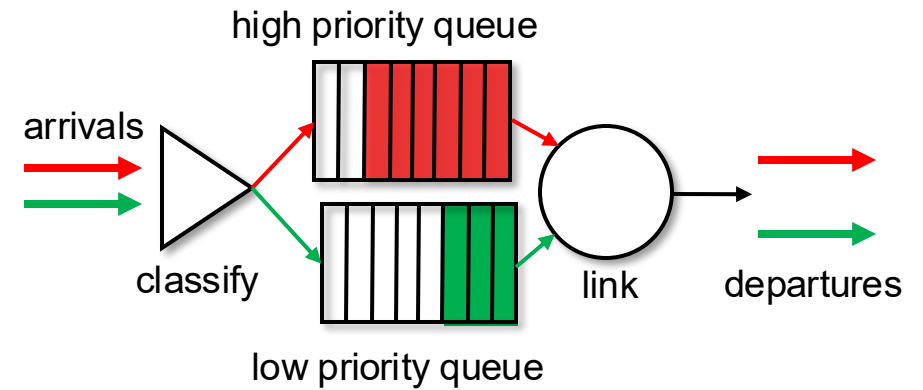
Scheduling policies: round robin

Round Robin (RR) scheduling:

- arriving traffic classified, queued by class
  - any header fields can be used for classification

- server cyclically, repeatedly scans class queues, sending one complete packet from each class (if available) in turn

classify arrivals

R

link   departures
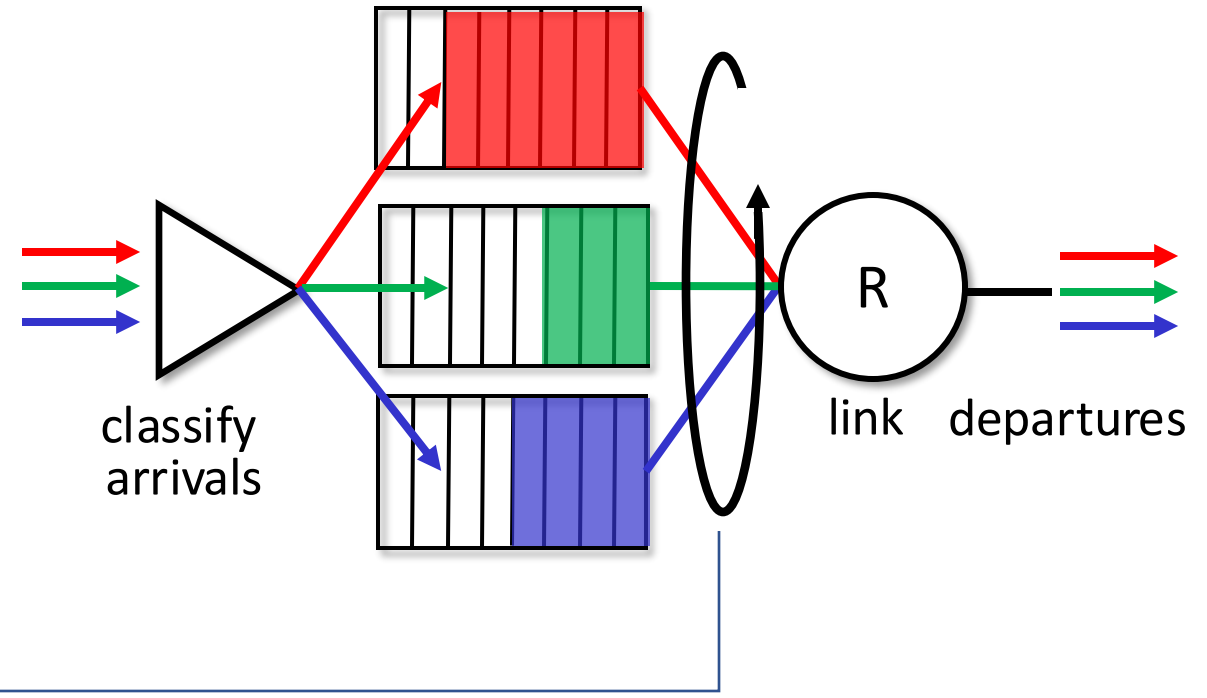
Scheduling policies: weighted fair queueing

Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class, *i,* has weight, w_i, and gets weighted amount of service in each cycle:

$$\frac{w_i}{\Sigma_j w_j}$$

- minimum bandwidth guarantee (per-traffic-class)
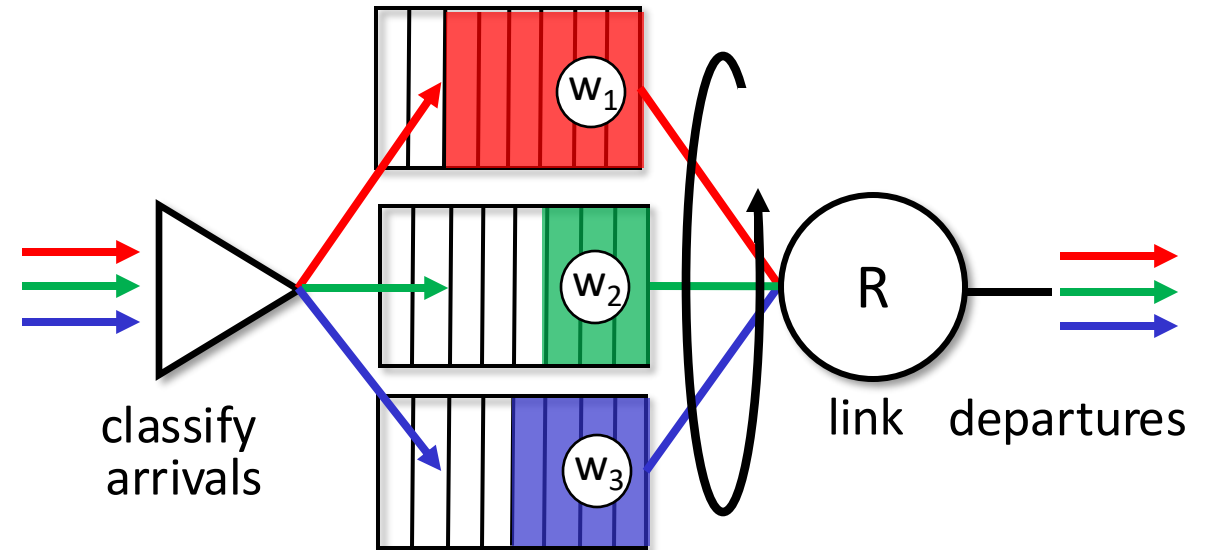
classify arrivals

link   departures

Network layer: "data plane" roadmap

▪ Network layer: overview
  • data plane
  • control plane
▪ What's inside a router
  • input ports, switching, output ports
  • buffer management, scheduling

▪ **IP: the Internet Protocol**
  • datagram format
  • addressing
  • network address translation
  • IPv6

▪ Generalized Forwarding, SDN
  • match+action
  • OpenFlow: match+action in action
▪ Middleboxes

Network Layer: Internet

host, router network layer functions:

transport layer: TCP, UDP

network layer

Path-selection algorithms:
implemented in
- routing protocols (OSPF, BGP)
- SDN controller

forwarding table

IP protocol
- datagram format
- addressing
- packet handling conventions

ICMP protocol
- error reporting
- router "signaling"

link layer

physical layer

IP Datagram format

32 bits

IP protocol version number

header length(bytes)

"type" of service:
- diffserv (0:5)
- ECN (6:7)

TTL: remaining max hops
(decremented at each router)

upper layer protocol (e.g., TCP or UDP)

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

total datagram length (bytes)

fragmentation/reassembly

header checksum

32-bit source IP address

Maximum length: 64K bytes
Typically: 1500 bytes or less

e.g., timestamp, record route taken

overhead

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