50.012 Networks

Lecture 12: Network Layer Overview

2021 Term 6

Assoc. Prof. CHEN Binbin

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Learning Objectives

- Understand principles behind network layer services:
  - network layer service models
  - forwarding versus routing
  - how a router works
  - generalized forwarding
- Instantiation, implementation in the Internet

Read textbook Section 4.1, 4.2

Outline

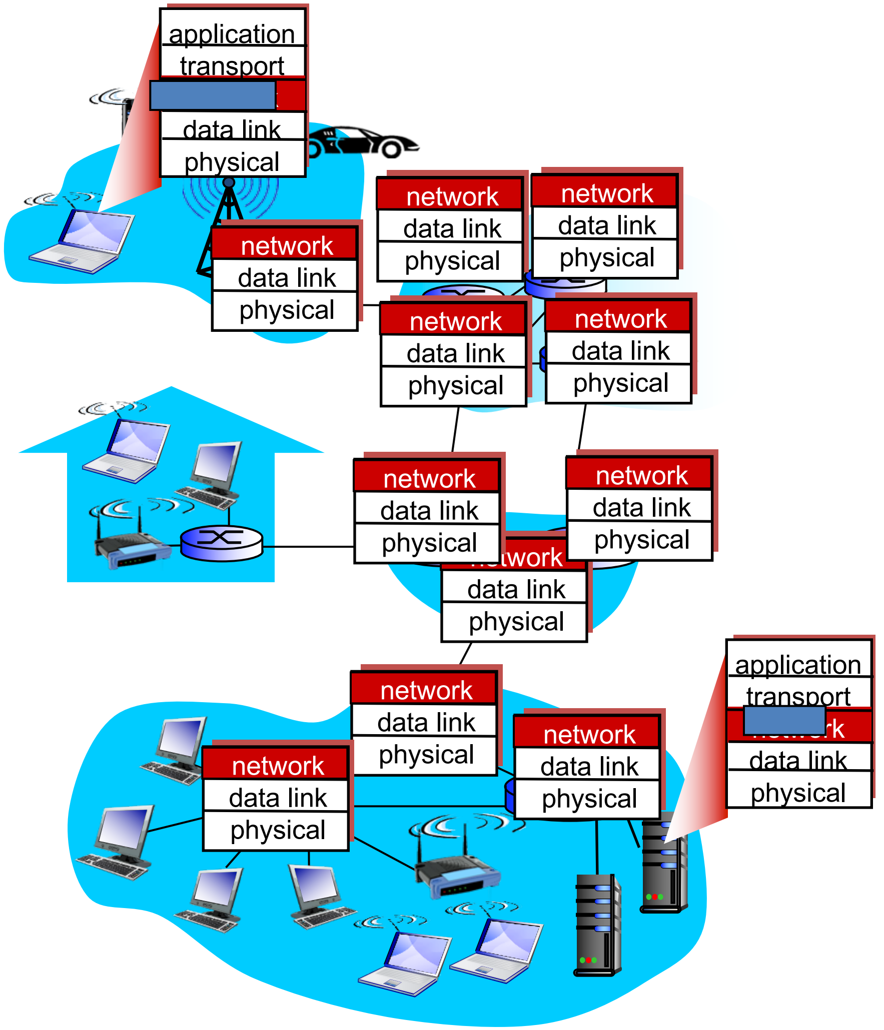
Overview of Network layer

- data plane
- control plane

What's inside a router

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

network-layer functions:

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

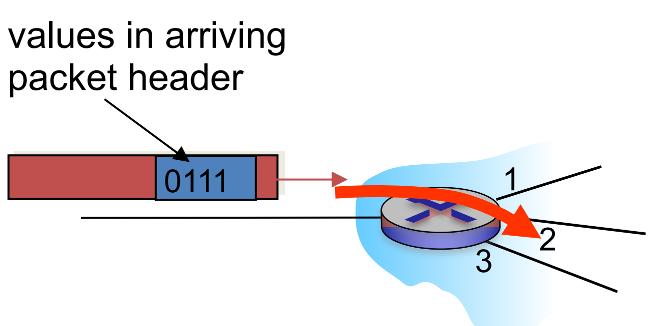
analogy: taking a trip

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

Network layer: data plane, control plane

Data plane

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

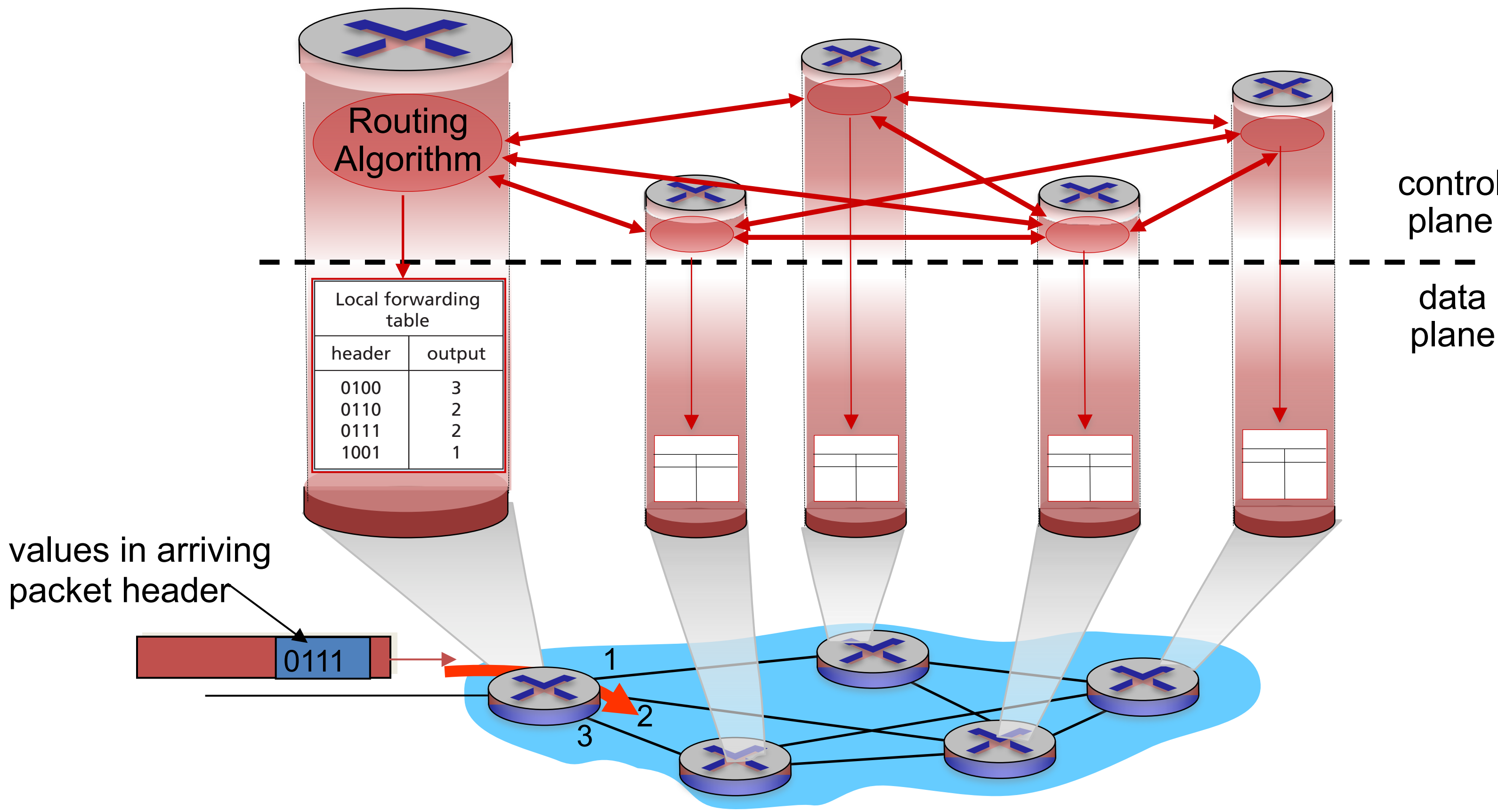


Control plane

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

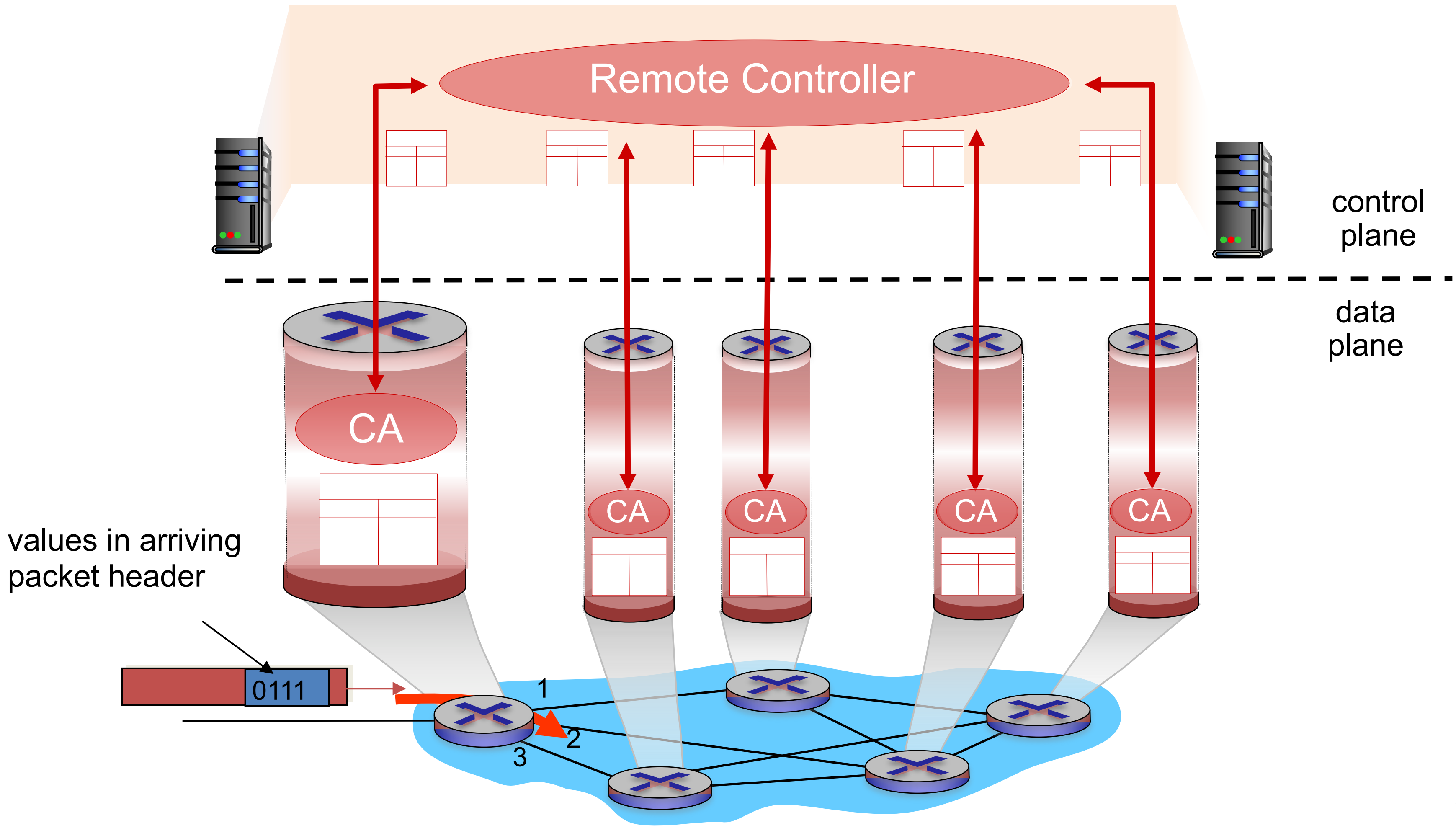
Per-router control plane

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



Logically centralized control plane

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



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
Internet	best effort	none	no	no	no

Outline

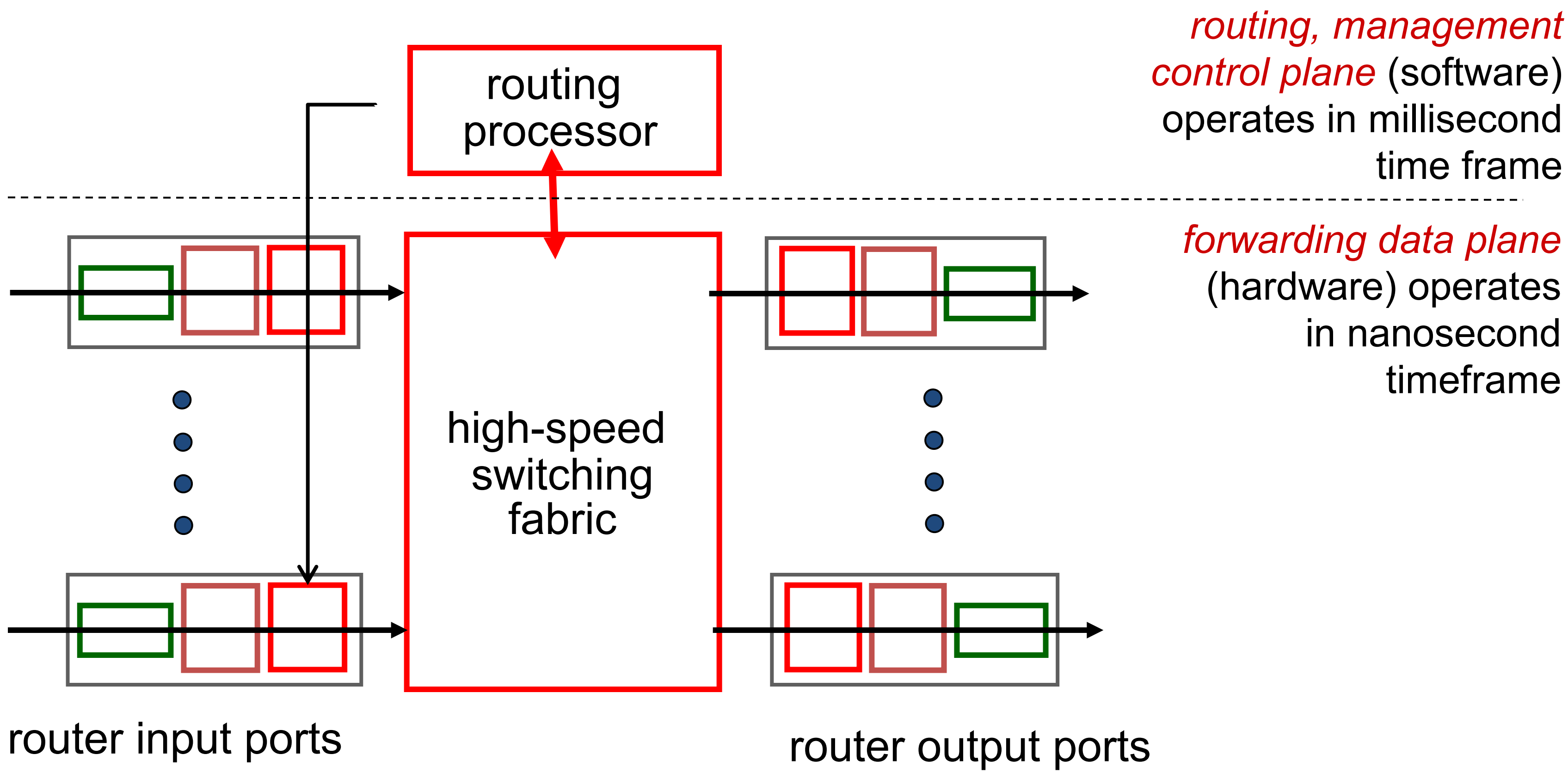
Overview of Network layer

- data plane
- control plane

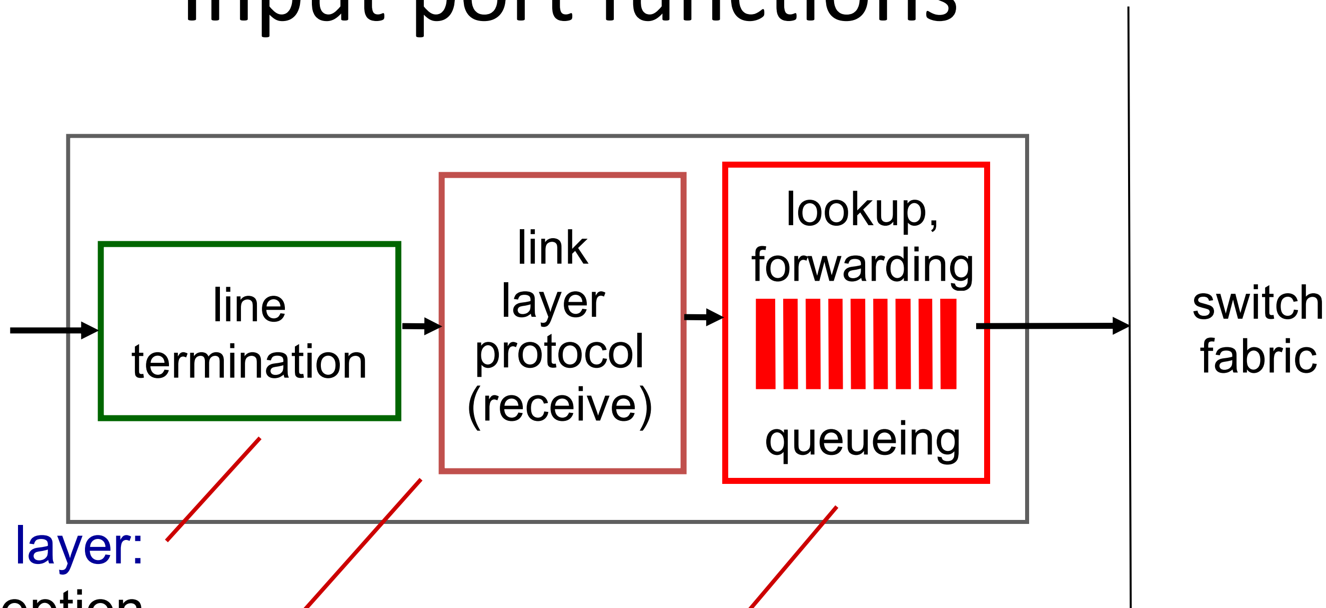
What's inside a router

Router architecture overview

- high-level view of generic router architecture:



Input port functions



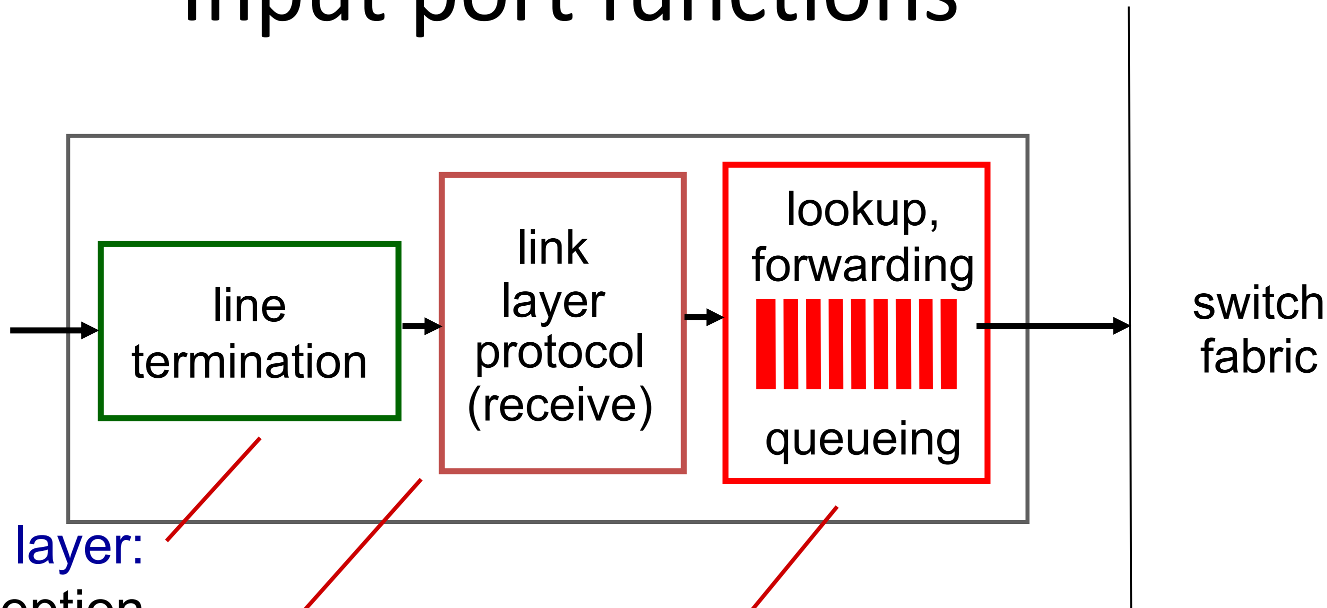
physical layer:
bit-level reception

data link layer:
e.g., Ethernet

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'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Input port functions



physical layer:
bit-level reception

data link layer:
e.g., Ethernet

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

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?

Longest prefix matching

- we'll see *why* longest prefix matching is used, when we study addressing
- longest prefix matching: often performed using ternary content addressable memories (TCAMs)
  - *content addressable:* present data to TCAM, retrieve address in one access cycle, regardless of table size
  - Cisco Catalyst: up to ~1M routing table entries in TCAM

TCAM

Adapted from TCAM Slides by Tom Edsall, Cisco

- Ternary: can match ‘0’, ’1’, or ‘X’
- Great for partial match
  - Longest prefix

- The magic does not come free:
  - 6X more power than SRAM
  - 7X more area than SRAM
  - 4X higher latency than SRAM

0	1	0	1	1	0	1
1	1	0	1	1	X	X
2	1	0	1	X	X	X
3	1	1	0	1	X	X
4	0	0	1	0	X	X
5	X	0	0	X	0	0
6	X	0	0	X	1	0
7	X	X	X	X	X	X

TCAM examples

1	0	1	1	0	0

0	1	0	1	1	0	1
1	1	0	1	1	X	X
2	1	0	1	X	X	X
3	1	1	0	1	X	X
4	0	0	1	0	X	X
5	X	0	0	X	0	0
6	X	0	0	X	1	0
7	X	X	X	X	X	X

1

1	0	0	1	1	0

0	1	0	1	1	0	1
1	1	0	1	1	X	X
2	1	0	1	X	X	X
3	1	1	0	1	X	X
4	0	0	1	0	X	X
5	X	0	0	X	0	0
6	X	0	0	X	1	0
7	X	X	X	X	X	X

6

TCAM examples

1	0	1	0	0	1

0	1	0	1	1	0	1
1	1	0	1	1	X	X
2	1	0	1	X	X	X
3	1	1	0	1	X	X
4	0	0	1	0	X	X
5	X	0	0	X	0	0
6	X	0	0	X	1	0
7	X	X	X	X	X	X

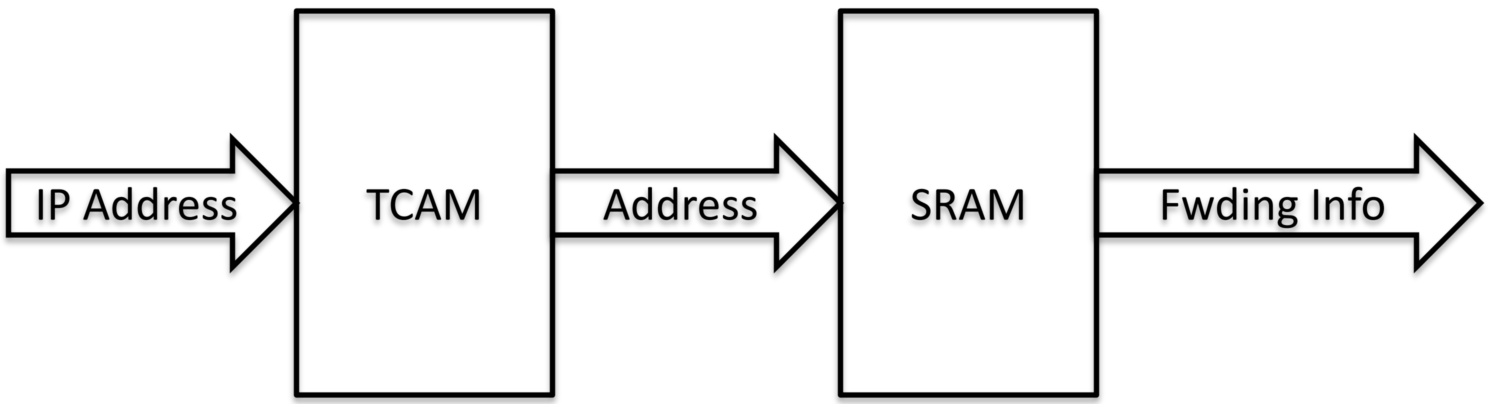
2

1	1	0	0	0	1

0	1	0	1	1	0	1
1	1	0	1	1	X	X
2	1	0	1	X	X	X
3	1	1	0	1	X	X
4	0	0	1	0	X	X
5	X	0	0	X	0	0
6	X	0	0	X	1	0
7	X	X	X	X	X	X

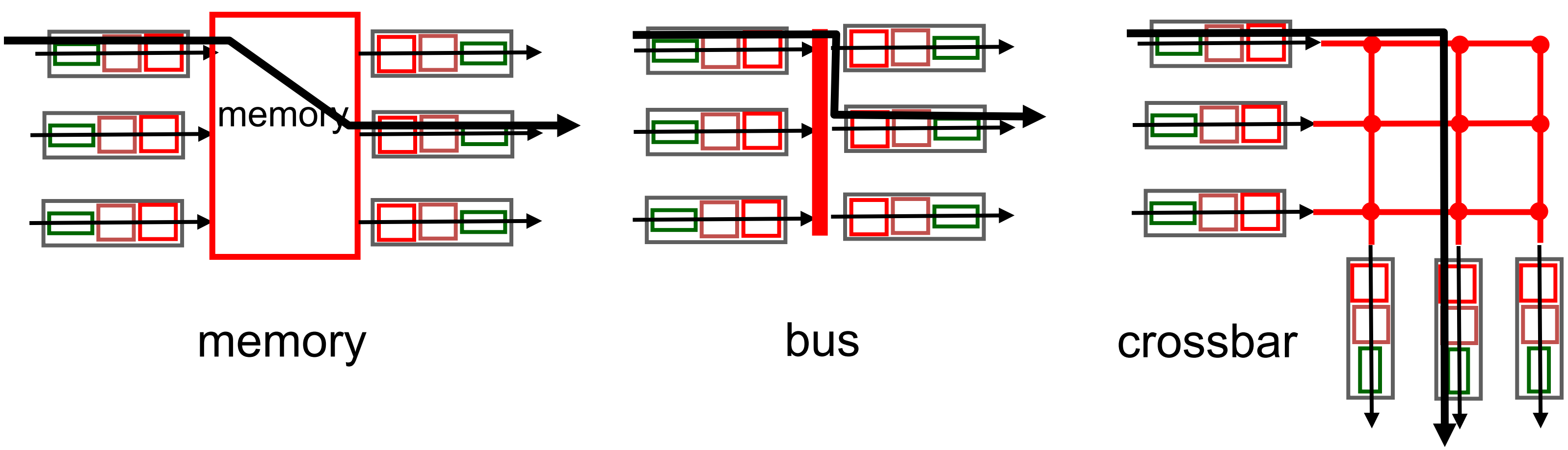
7

Typical Forwarding Diagram



Switching fabrics

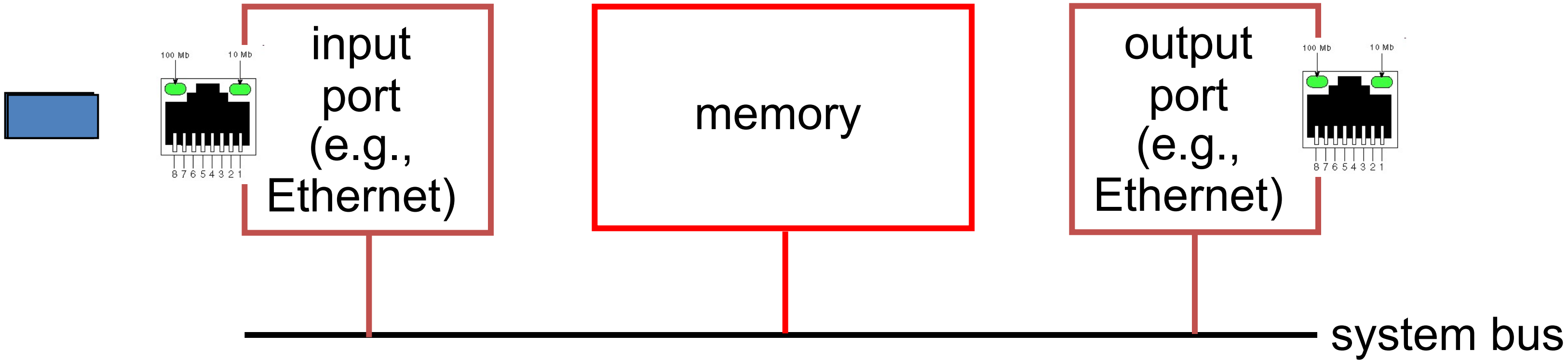
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transferred from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Switching via memory

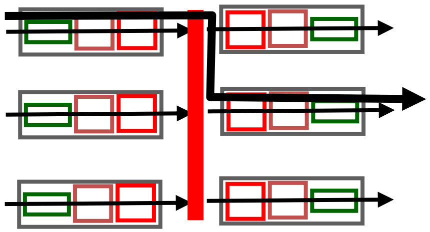
first generation routers:

- 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

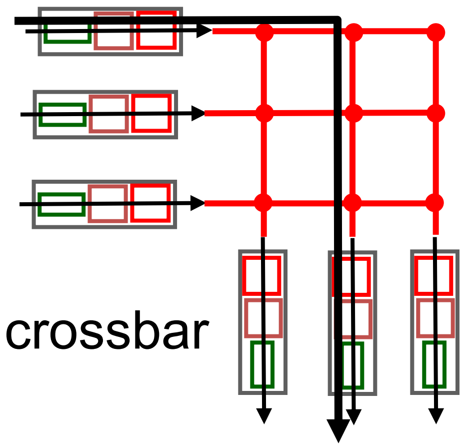


bus

Switching via interconnection network

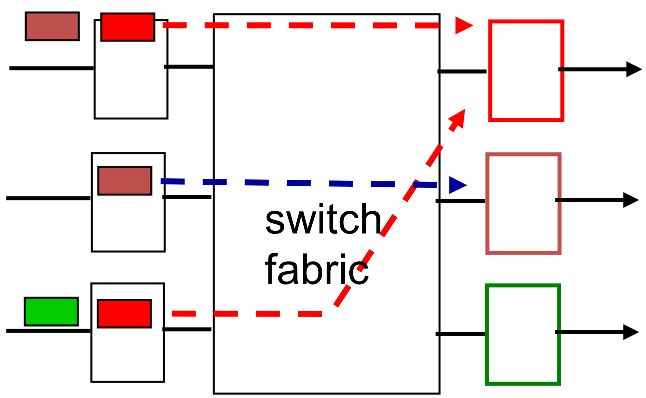
- overcome bus bandwidth limitations

- Crossbar (and other interconnection nets) initially developed to connect processors in multiprocessor

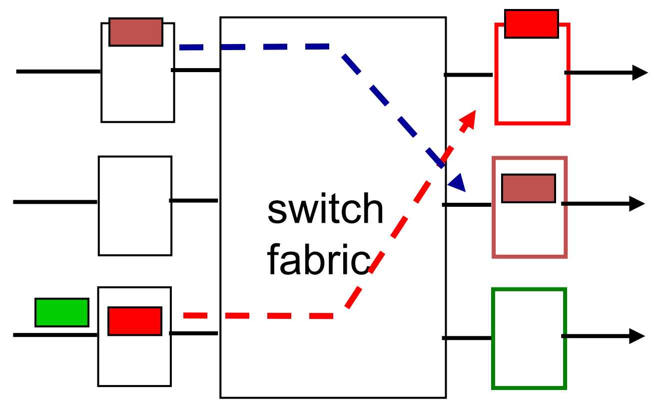


Input port queuing

- 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

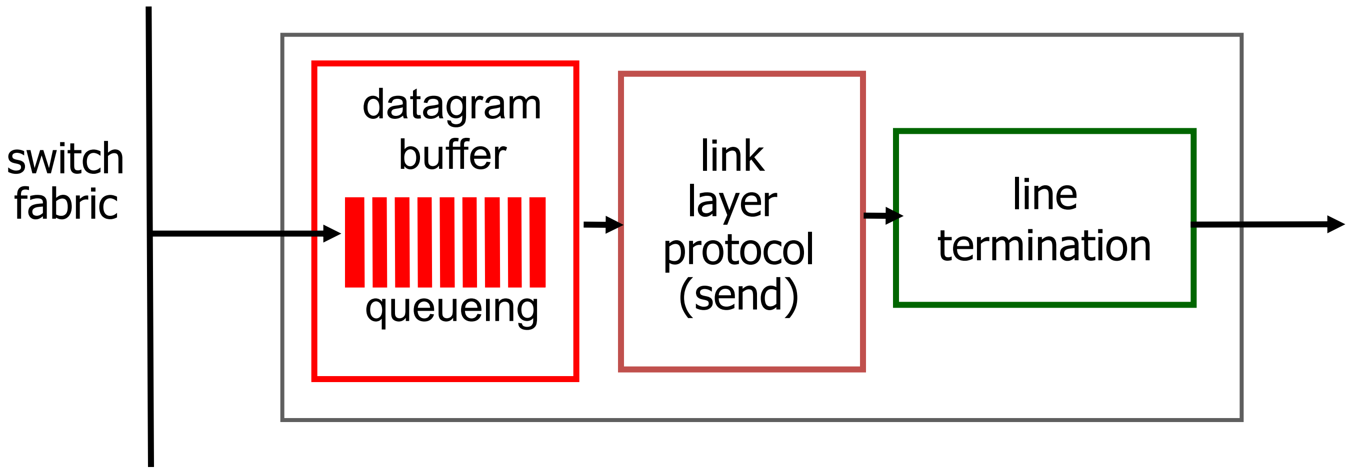


output port contention:
only one red datagram can be transferred.
lower red packet is blocked



one packet time later:
green packet experiences HOL blocking

Output ports

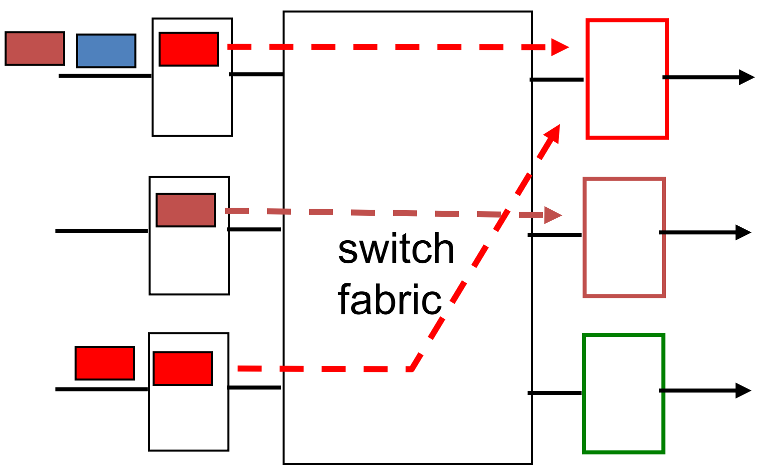


- *buffering* required when datagrams arrive from fabric faster than the transmission rate
- *scheduling discipline* chooses among queued datagrams for transmission

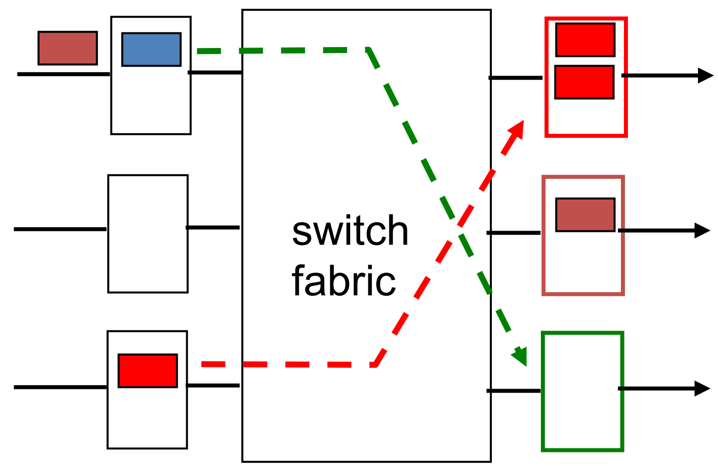
Datagram (packets) can be lost due to congestion, lack of buffers

Priority scheduling – who gets best performance

Output port queueing



at *t,* multiple packets destined for the same outgoing port and switch fabric is faster than line speed

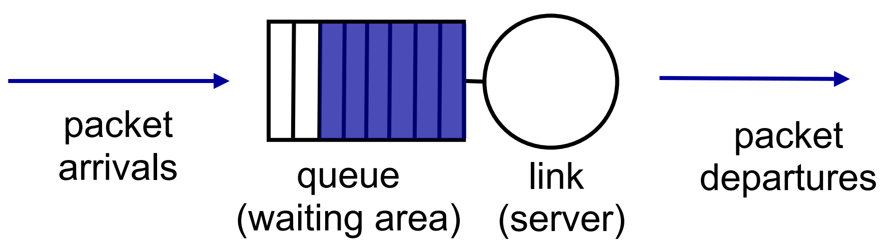


one packet time later

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

Scheduling mechanisms

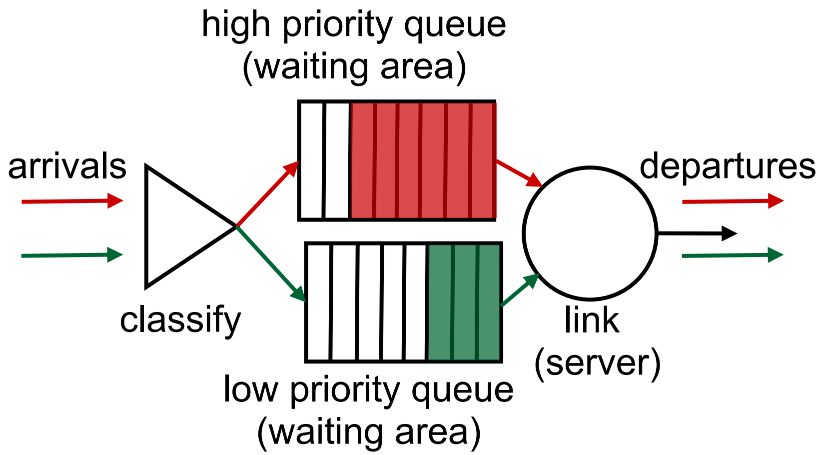
- *scheduling:* choose next packet to send on link
- *FIFO (first in first out) scheduling:* send in order of arrival to queue
  - *discard policy:* if packet arrives to full queue: who to discard?
    - *tail drop:* drop arriving packet
    - *priority:* drop/remove on priority basis
    - *random:* drop/remove randomly

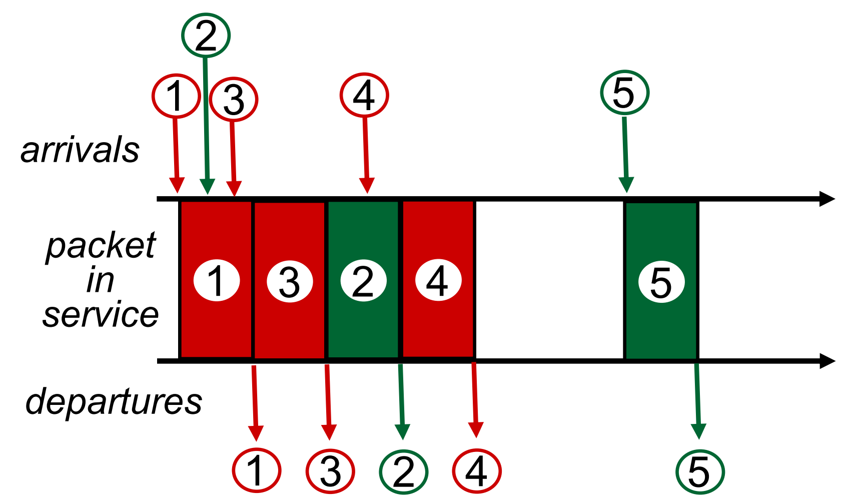


Scheduling policies: priority

priority scheduling: send highest priority queued packet

- multiple *classes*, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.

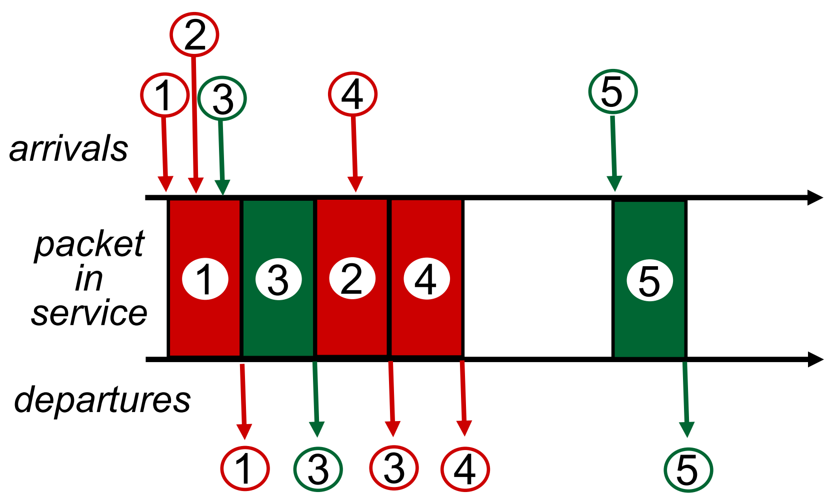




Scheduling policies: still more

Round Robin (RR) scheduling:

- multiple classes
- cyclically scan class queues, sending one complete packet from each class (if available)



Scheduling policies: still more

Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class gets weighted amount of service in each cycle