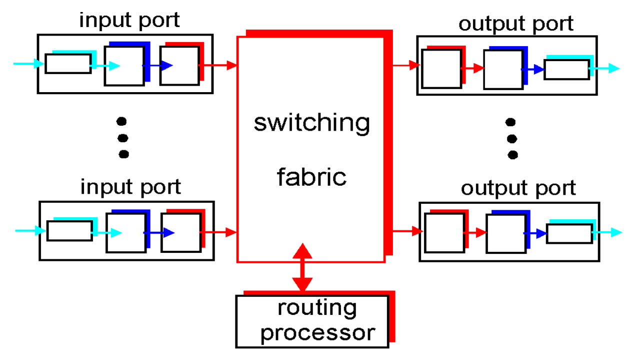
eecs489 Computer Networks

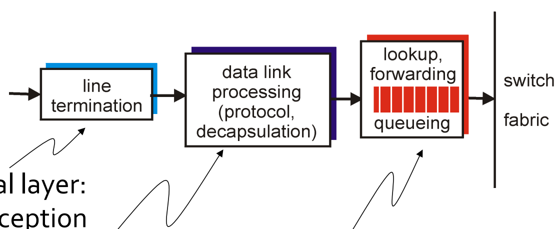
Lecture 35:
Router Architecture, FIFO, RED, ECN

Router Architecture Overview

Two key router functions:
1. run routing algorithms/protocol (RIP, OSPF, BGP)
2. forward packets from incoming to outgoing link



Input Port Functions



Physical layer:
bit-level reception

Data link layer:
e.g., CRC

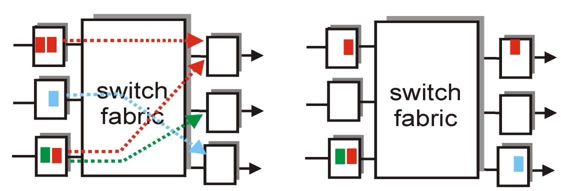
Packet forwarding:
- given packet destination, lookup output port using forwarding table in input port memory
- goal: complete input port processing at "line speed"
- input queueing: if packets arrive faster than forwarding rate into switch fabric

Input Port Queueing

Fabric slower than input ports taken together
⇒ queueing may occur at input queues

Head-of-the-Line (HOL) blocking: queued packet at the front of queue prevents others in queue from moving forward

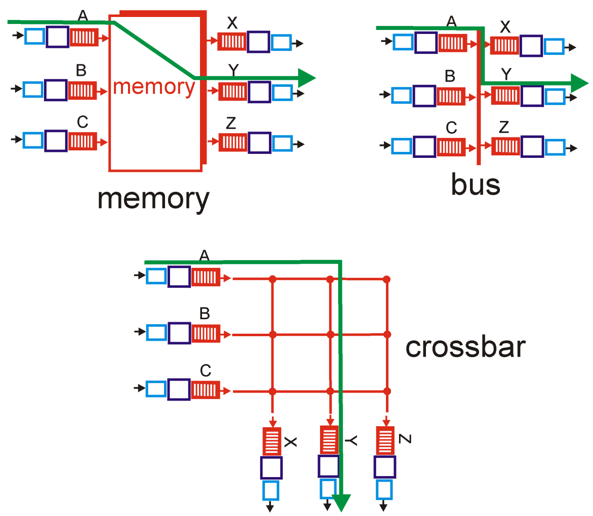
Queueing delay and loss can happen at input port buffer



output port contention at time t - only one red packet can be transferred

green packet experiences HOL blocking

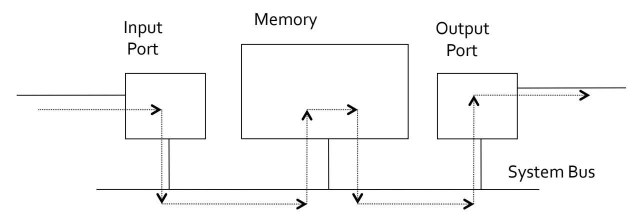
Three Types of Switching Fabrics



Switching Via Memory

First generation routers:

- traditional computers, with switching under direct control of CPU
- packet copied into system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)

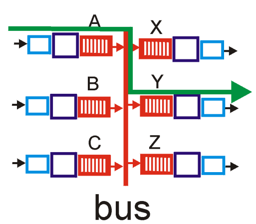


Switching Via a Bus

Packet transferred from input port memory to output port memory via a shared bus

Bus contention: switching speed limited by bus bandwidth

1 Gbps bus (e.g., Cisco 1900): sufficient speed for access and enterprise routers (not regional or backbone)



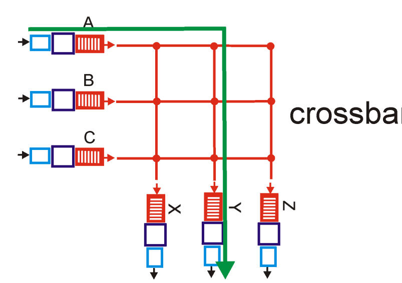
Switching via Interconnect

Overcome bus bandwidth limitations

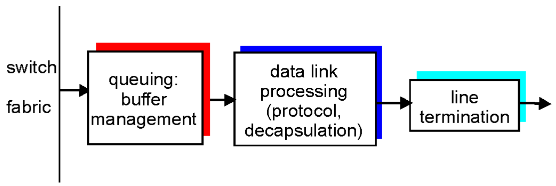
Banyan networks and other interconnection nets initially developed to connect processors in multiprocessor system

Advanced design: fragment datagrams into fixed length cells, switch cells through the fabric

E.g., Cisco 12000: switches Gbps through the interconnect



Output Ports



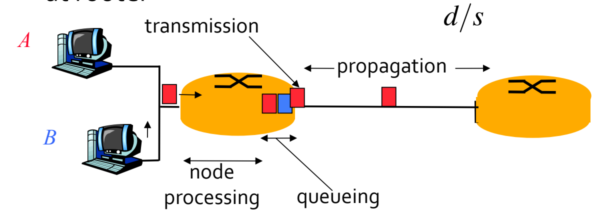
Output queueing: buffering required when packets arrive from fabric faster than the line transmission rate

Scheduling discipline chooses among queued packets for transmission

Queueing delay and loss can also happen at output port buffer

Four Sources of Packet Delay

1. Node processing/ service time:
   - check bit errors
   - determine output link
2. Queueing delay
   - time waiting at output link for transmission
   - depends on congestion level at router
3. Transmission delay (μ):
   - $R =$ link bandwidth (bps)
   - $L =$ packet length (bits)
   - transmission delay $= L/R$
4. Propagation delay (τ):
   - $d =$ length of physical link
   - $s =$ propagation speed in medium (~$2\text{x}10^8$ m/sec)
   - propagation delay (latency) $= d/s$



Note: s and R are very different quantities!

Per-Node Delay

$$d_{\text{per-node}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}}$$

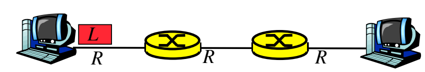
$d_{\text{proc}} =$ processing delay
- typically a few μsecs or less

$d_{\text{queue}} =$ queueing delay
- depends on congestion

$d_{\text{trans}} =$ transmission delay
- $= L/R$, significant for low-speed links

$d_{\text{prop}} =$ propagation delay
- a few μsecs to hundreds of msecs

Packet Switching: Store and Forward



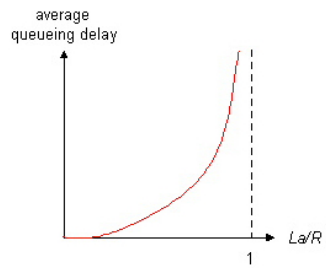
Transmission delay: it takes L/R seconds to transmit packet of L bits onto link of R bps

Entire packet must arrive at a router before it can be transmitted onto the next link: store and forward

Total transmission delay $= 3L/R$

Example:
- $L = 7.5$ Mbits
- $R = 1.5$ Mbps
- total transmission delay $= 15$ sec

Queueing Delay



$L =$ packet length (bits)
$a =$ average packet arrival rate
$R =$ link bandwidth (bps)
Traffic intensity $(I) = La/R$

- $I \sim 0$: average queueing delay small
- $I \sim 1$: delays become large
- $I > 1$: more "work" arriving than can be serviced, average delay infinite!

Queue (buffer) preceding a link has finite capacity

- when packet arrives at full queue, packet is dropped (lost)
- lost packet may be retransmitted by the previous node, by the source, or not at all

Queue Management

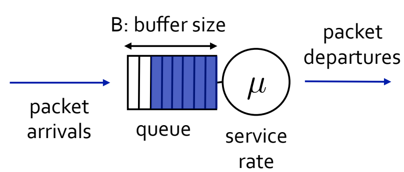
Design issues:

- scheduling discipline:
  - choose next packet to send on link
  - e.g., FIFO, priority queue, fair queue
  - fairness issue
  - delay bound
- drop policy:
  - drop tail: drop newly arriving packet
  - drop head: long queued packet may no longer be of use to receiver
  - priority drop: drop on a priority basis
  - random drop: either on overflow or early drop
- cost of operation

Traditionally: FIFO with drop tail

First In First Out (FIFO)

- queued packets sent in order of arrival
- low cost
- not fair
- no delay bound



Problems with Drop-Tail

Traditionally, FIFO with drop tail, however:

- TCP uses packet drop as congestion signal
- congestion signal is time delayed in reaching sender
- problems:
  - drop-tail queuing leads to bursty losses
  - when a link becomes congested many arriving packets encounter a full queue
  - as a result, many flows divide sending rate in half and, many individual flows lose multiple packets
  - growth of `cwnd`'s across flows can become synchronized
  - unfair to long-RTT connections

Slow Feedback from Drop Tail

Feedback is given only when buffer is completely full
- even though the buffer has been filling up for a while

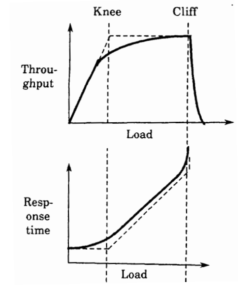
Plus, the filling buffer is increasing RTT
- and RTT variance

May be better to give early feedback
- get 1-2 largest flows to slow down, not all of them
- get these flows to slow down before buffer overflows

Early Drop

Early drop:
- watch the queue and start dropping before the queue is full
- give time for congestion signal to propagate back to source

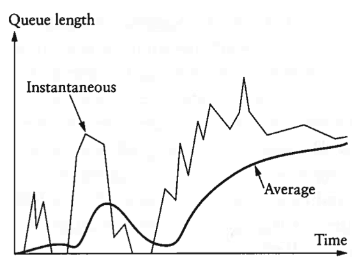
Goals:
- operate router at "knee" capacity: low delay, high throughput
- keep average queue size low
- but allow for fluctuations in actual queue size to accommodate bursty traffic and transient congestion



Jain et al.

Random Drop

Random drop:
- observe: flows with more packets in queue have a higher probability of being dropped
- drop packet randomly if average queue length is above threshold
  - high instantaneous queue length could simply mean transient congestion
  - high average queue length indicates persistent congestion
  - what would be a good way to measure average queue length?



Peterson & Davie

Random Early Drop

Random Early Drop:
- router notices that the queue is getting backlogged
- and randomly drops packets to signal congestion early
  - drop probability increases as queue length increases
  - if buffer is below some level, don't drop anything
  - otherwise, set drop probability as function of queue

Implementation:
- define a minimum (`min_th`) and a maximum (`max_th`) queue occupancy thresholds
- monitor average queue length (`aqlen`)

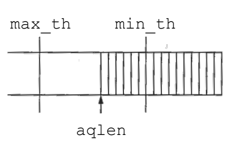


Figure 6.16 RED thresholds on a FIFO queue.

Peterson & Davie

Random Early Drop

- monitor average queue length:
  aqlen ← (1−w) aqlen + w*q,
  where :
  - aqlen: average queue length
  - w: weight
  - q: instantaneous queue length
- provide congestion avoidance by controlling average queue length
  - aqlen < min_th, no packet is dropped
  - aqlen > max_th, all packets are dropped
  - between the two thresholds, each packet has some probability P() ≤ MaxP to be dropped, depending on size and duration of queue occupancy

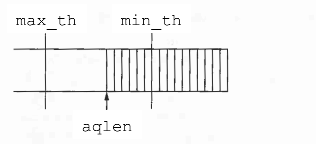


Figure 6.16 RED thresholds on a FIFO queue.

Peterson & Davie

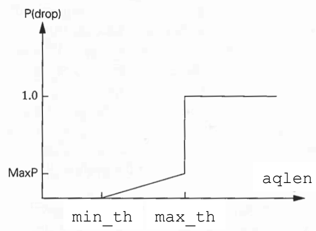


Figure 6.17 Drop probability function for RED.

Properties of RED

Drops packets before queue is full
- in the hope of reducing the rates of some flows

Drops packet in proportion to each flow's rate
- high-rate flows have more packets in queue
- hence, a higher chance of being selected

Drops are spaced out in time
- should help desynchronize the TCP senders

Tolerant of burstiness in the traffic
- by basing the decisions on average queue length

But, hard to get tunable parameters just right
- when parameters are not right, RED doesn't help
- as a result, implemented and deployed but not used

Approaches Towards Congestion Control

End-end congestion control:
- approach taken by TCP
- no explicit feedback from network
- congestion inferred from end-system observed loss, delay

Network-assisted congestion control:
- routers provide feedback to end systems
  - single bit indicating congestion, or
  - specific sending rate

Explicit Congestion Notification

Early dropping of packets
- good: gives early feedback
- bad: has to drop the packet to give the feedback

Explicit Congestion Notification (ECN)
- router marks the packet with an ECN bit
- receiver reflects ECN to sender
- sender interprets it as a sign of congestion and reduce rate

Explicit Congestion Notification

Implementation issues:

- must be supported by both end hosts and routers
- requires 2 bits in the IP header for forward (router to receiver) direction
  - one for the ECN mark; the other to indicate ECN capability
  - solution: borrow 2 Type-Of-Service bits in IPv4 header
- requires another 3 bits in TCP header (the 4 bits after header length that were previously not used) for receiver to signal sender (reverse direction)