# ECE 356/COMPSI 356
## Computer Network Architecture

## Queuing and Congestion Avoidance

Monday November 11th, 2019





## Recap

- Previous lecture: TCP congestion control
- Readings for this lecture: **PD 6.1, 6.2, 6.4**

# TCP Congestion Control: A Quick Recap

- Network congestion is problematic. It leads to:
  - Delays
  - Segment losses
  - Wasted work of the network
- TCP employs window-based congestion control
  - Maximum number of bytes in transit: *min(CongestionWindow, AdvertisedWindow)*
  - Sender *probes the network* by injecting more and more data in it
  - Backs off when encountering losses

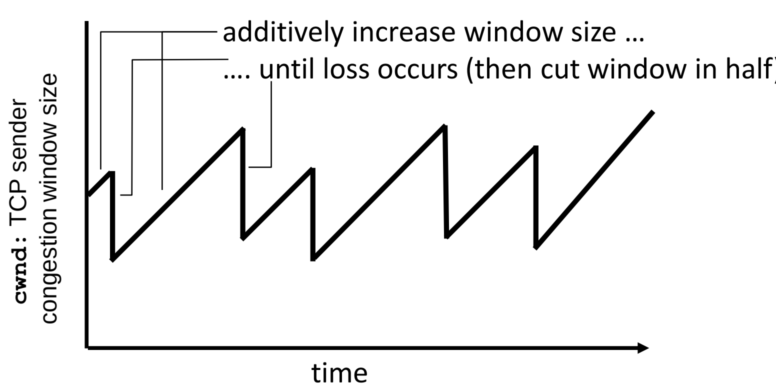
3

Duke UNIVERSITY

3

# TCP Congestion Control: AIMD

AIMD "*sawtooth behavior*": probing for bandwidth




4

Duke UNIVERSITY

4

# Multiple Flavors of TCP

- TCP Tahoe, Reno, Vegas, BBR, CUBIC, …
- Different feedback signals
- Different specifics of sawtooth patterns



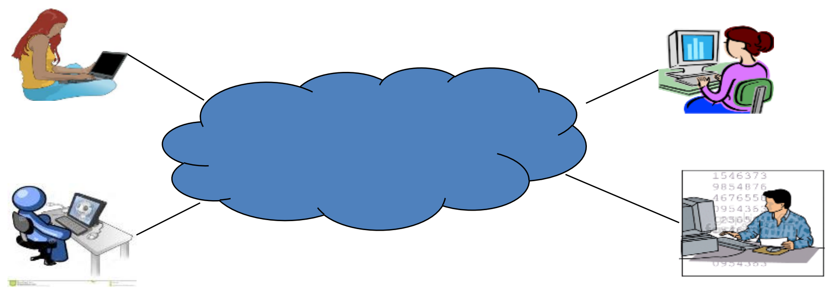




# Lecture Outline

- **Issues in resource allocation**
- Queuing disciplines
- Congestion avoidance: an overview
- Router-based congestion avoidance schemes: DECbit, RED, ECN
- Source-based congestion avoidance schemes: general approaches, TCP Vegas

# Resource Allocation



- A fundamental question of networking: who gets to send at what speed?

Duke UNIVERSITY

7

# Resource Allocation vs. Congestion Control

- **Resource allocation**: the process by which network elements try to meet the competing demands that applications have for network resources
  - Bandwidth and buffer space

- **Congestion control**: efforts made only by network nodes to prevent or respond to overload conditions

Duke UNIVERSITY

8

# Network Model

- Packet switched
- Connectionless flows
  - Flow: a sequence of packets sent between a source host and a destination host
- Service model
  - Best-effort
  - Quality of Service

Duke UNIVERSITY

9

# Design Space for Resource Allocation

- Router-centric vs. host-centric
- Reservation-based vs. feedback-based
- Window-based vs. rate-based

Duke UNIVERSITY

10

# Evaluation Criteria

- Performance and fairness
  - Performance: high throughput, low latency
  - Fairness: Chiu-Jain fairness index

$$F(x) = \frac{(\Sigma x_i)^2}{n(\Sigma x_i^2)}.$$



11

# Jain Fairness Index: An Example

$$F(x) = \frac{(\Sigma x_i)^2}{n(\Sigma x_i^2)}.$$

- 2 flows, total BW=10
- [5,5]:
  - F(x) = (10)^2/(2*(25+25)) = 100/100= 1
- [4,6]:
  - F(x) = (10)^2/(2*(16+36))= 100/104 = 0.96
- [1,9]:
  - F(x) = (10)^2/(2*(1+81))= 100/164 = 0.61
- [0.1, 9.9]
  - F(x) = (10)^2/(2*(0.01+98.01)) = 100/196.04 = 0.51



12

# Lecture Outline

- Issues in resource allocation
- **Queuing disciplines**
- Congestion avoidance: an overview
- Router-based congestion avoidance schemes: DECbit, RED, ECN
- Source-based congestion avoidance schemes: general approaches, TCP Vegas

Duke UNIVERSITY 13

13

# Queuing Mechanisms

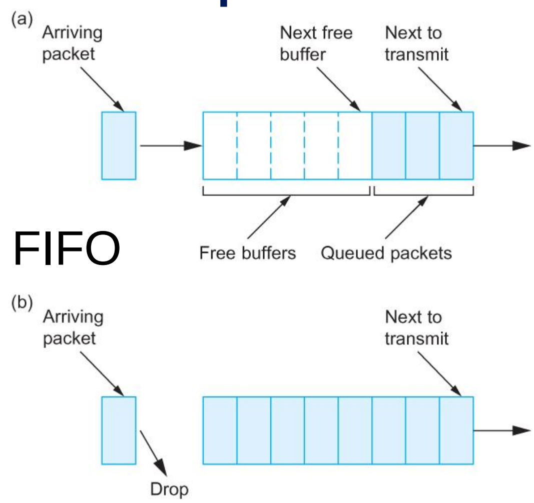
- Router-enforced resource allocation
  - Scheduling policy: which packet gets sent
  - Drop policy: which packet gets dropped

Duke UNIVERSITY

14

# Default: FIFO with Drop Tail

- Scheduling policy: first come first serve (FIFO)
- Drop policy: tail drop
- Simple, widely used
- No congestion control, resource allocation included




15

Duke UNIVERSITY

15

# A Variation: Priority Queuing

- Mark packets with priority bits
- Multiple FIFO queues, each for one priority
- Transmit packets out of highest priority queues
- Limitation: may starve low priority packets
  - Users cannot set their priority bits
  - Could potentially charge users more for sending higher-priority traffic
- **Routing messages get high priority**

Duke UNIVERSITY

16

# Fair Queuing

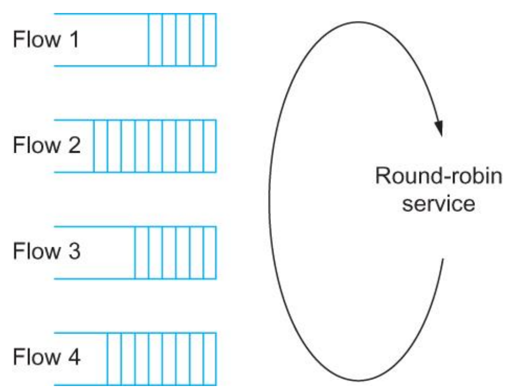
- FIFO is not concerned with which packets belong to which flows
- Alternative approach:
  - **Fair queuing**: a queuing algorithm that aims to "fairly" allocate buffer, bandwidth, latency among competing users





# Round-robin Service of Flows

- Maintain separate queues per flow
- Service different flows in a round-robin fashion
- A source cannot get more service at the expense of others
- Implementations take into account that packets are not the same length




Example: service of 4 flows by a router

# Resource Allocation in Fair Queuing

- The link is not idle if there is at least one packet in the queue
  - ***Work conserving*** technique
- With *n* flows sending data, no source can use more than *1/n*th of the link bandwidth
- Bandwidth available to a flow changes depending on the number of flows served by a link
- But, available bandwidth is always shared fairly between competing flows





# Weighted Fair Queuing

- Assign a weight to each flow
  - E.g., flows with weights 1,2,3: the first one gets 1/6$^{th}$ of the bandwidth, the second one gets 1/3$^{rd}$, the third one gets ½
- Can be implemented on *classes* of traffic
- Weak resource reservation: actual bandwidth allocated to a flow depends on other flows and their priorities

# Queuing Disciplines: Key Points to Remember

- Default queueing approach: FIFO with drop tail
- Priority queuing: multiple FIFO queues for packets with different priority levels
  - May starve low-priority packets
- Fair queuing: a queue for each flow
  - Shares available bandwidth fairly between the flows

Duke UNIVERSITY 21

21

# Lecture Outline

- Issues in resource allocation
- Queuing disciplines
- **Congestion avoidance: an overview**
- Router-based congestion avoidance schemes: DECbit, RED, ECN
- Source-based congestion avoidance schemes: general approaches, TCP Vegas

Duke UNIVERSITY 22

22

## TCP: Controls Congestion Once It Happens

- TCP reacts to congestion after it takes place
- The data rate changes rapidly and the system is barely stable (or is already unstable)
- Can we *predict* when congestion is about to happen and avoid it?
  - E.g., delays are increasing
  - Queues are getting long

Duke UNIVERSITY

23

## Congestion Avoidance Schemes (1/2)

- **Router-based** congestion avoidance
  - DECbit: routers explicitly notify sources about congestion
  - Random Early Detection (RED)
    - Routers implicitly notify sources by dropping packets
    - RED drops packets at random, as a function of the level of congestion

Duke UNIVERSITY

24

## Congestion Avoidance Schemes (2/2)

- Host-based congestion avoidance
  - ➢Source monitors changes in RTT to detect onset of congestion
  - ➢Or changes in effective throughput





## Lecture Outline

- Issues in resource allocation
- Queuing disciplines
- Congestion avoidance: an overview
- **Router-based congestion avoidance schemes**: DECbit, RED, ECN
- Source-based congestion avoidance schemes: general approaches, TCP Vegas

# DECbit

- Add a congestion bit to a packet header
- A router sets the bit if its average queue length is non-zero
- If less than 50% of packets in one window do not have the bit set
  - A host increases its congest window by 1 packet
- Otherwise
  - Decreases by 0.875x
- AIMD

Duke UNIVERSITY

27

# Random Early Detection (RED)

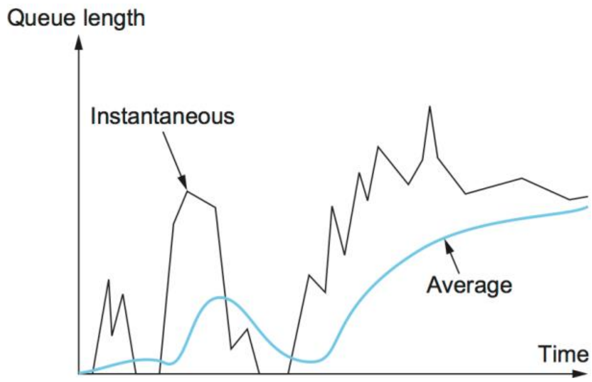
- Also known as *random early discard* or *random early drop*
- Pre-emptively drop packets before a buffer becomes full
  - Implicitly notifies sender by dropping packets
  - Works with standard TCP mechanisms
- Drop probability is increasing as the *average* queue length increases
  - Exponential weighted averaging algorithm for queue length estimation

$$AvgLen_{n+1} = (1-\alpha) \times AvgLen_n + \alpha \times Length_n$$

Duke UNIVERSITY

28

# Queue Length: Instantaneous vs. Average




- Side note: tail drop can be seen as using instantaneous queue length as a signal
- Tail drop is unfair

$$AvgLen_{n+1} = (1-\alpha) \times AvgLen_n + \alpha \times Length_n$$

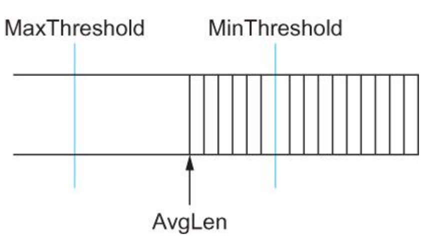
29

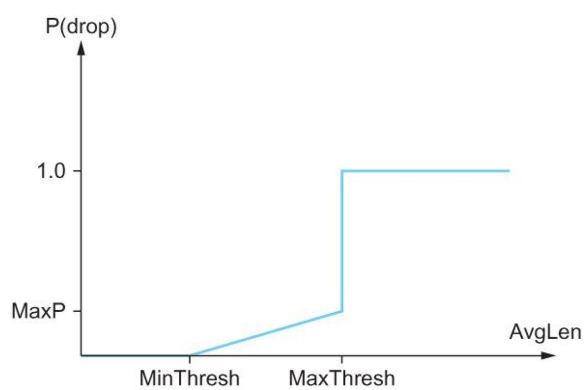
Duke UNIVERSITY

29

# RED Algorithm

- Two thresholds for different packet drop policies




Drop probability function for RED

Duke UNIVERSITY

30

# Fairness in RED

- Packets dropped at random → probability to drop flow's packet is ~ proportional to the flow's share of bandwidth
- Does not possess a bias against bursty traffic that uses only a small portion of the bandwidth

31

Duke UNIVERSITY

31

# RED: Evening Out Packet Drops (1/2)

- Caveat: do not want to drop a packet immediately after a previous drop
  - Happens readily with purely random drop settings
  - Serves no purpose: one packet drop per RTT is sufficient to reduce congestion window size
  - Multiple drops could cause a slow start
- Spaced-out drops are more likely to affect different connection, when traffic is bursty

Duke UNIVERSITY

32

# RED: Evening Out Packet Drops (2/2)

- Approach: make drop probability additionally dependent on the time since the last packet drop
- TempP = MaxP x (AvgLen – MinThreshold)/(MaxThreshold-MinThreshold)
- P = TempP / (1 – count * TempP)
- Count
  - Keeps track of how many newly arriving packets have been queued when min < Avglen < max
  - It keeps drop evenly distributed over time, even if packets arrive in burst
  - Reset to zero after a drop

Duke UNIVERSITY

33

# Explicit Congestion Notification (ECN) (1/2)

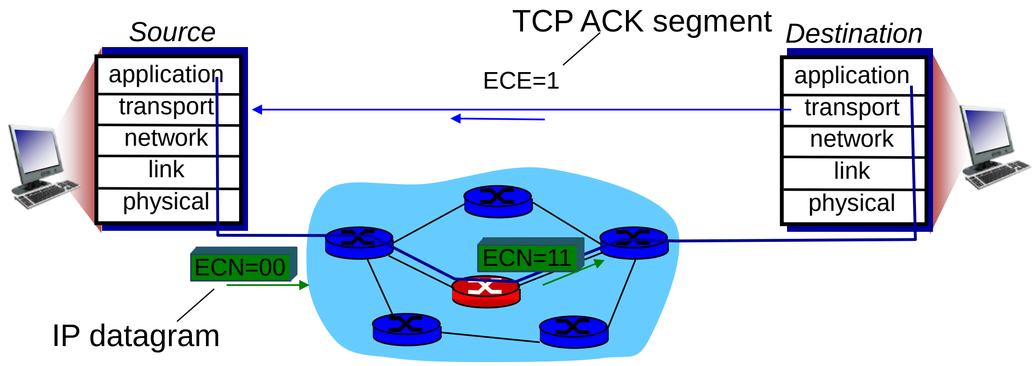
- RED can be used in conjunction with ECN
- **Explicit notification** instead of packet dropping
- Extension of IP and TCP standards
- Two bits in IP header (ToS field) marked by network router to indicate congestion
- Congestion indication carried to receiving host
- Receiver (seeing congestion indication in IP datagram) sets ECE bit on receiver-to-sender TCP ACK segment to notify sender of congestion

3-34

Duke UNIVERSITY

34

# Explicit Congestion Notification (ECN) (2/2)




- Used in datacenter networking

Duke UNIVERSITY

35

# Router-based Congestion Avoidance Schemes: Key Points to Remember

- Routers implicitly or explicitly notify sources of their state
  - Implicitly: by pre-emptively dropping packets
    - Working with existing TCP mechanisms
  - Explicitly: by reporting congestion via setting flags on packets in transit
- For reporting state
  - Use average, rather than instantaneous, queue length
  - Space out packet drops/notifications

3-36

Duke UNIVERSITY

36

# Lecture Outline

- Issues in resource allocation
- Queuing disciplines
- Congestion avoidance: an overview
- Router-based congestion avoidance schemes: DECbit, RED, ECN
- **Source-based congestion avoidance schemes**: general approaches, TCP Vegas





# Source-based Congestion Avoidance

- General idea: watch, at the source, for a sign of upcoming congestion
  - Some router's queue is building up
- Reduce congestion window pre-emptively

# Source-based Congestion Avoidance: Reacting to Increasing RTT (1/2)

- Use standard TCP window increase and decrease mechanisms
- Every two RTTs, checks to see if the current RTT is greater than the average of the minimum and maximum RTTs seen so far
- If it is, then the algorithm decreases the congestion window by one-eighth

Duke UNIVERSITY

39

# Source-based Congestion Avoidance: Reacting to Increasing RTT (2/2)

- Another approach
- Every two RTTs, calculate
  - (CurrentWindow − OldWindow)×(CurrentRTT − OldRTT)
- Positive: the source decreases the window size by one-eighth
- Negative or 0: the source increases the window by one maximum packet size
- Window changes during every adjustment
  - Oscillates around its optimal point

Duke UNIVERSITY

40

# Source-based Congestion Avoidance: TCP Vegas (1/2)

- General mechanism:
  - Detect increase in queueing delay
  - Reduce sending rate

Duke UNIVERSITY

41

# Source-based Congestion Avoidance: TCP Vegas (2/2)

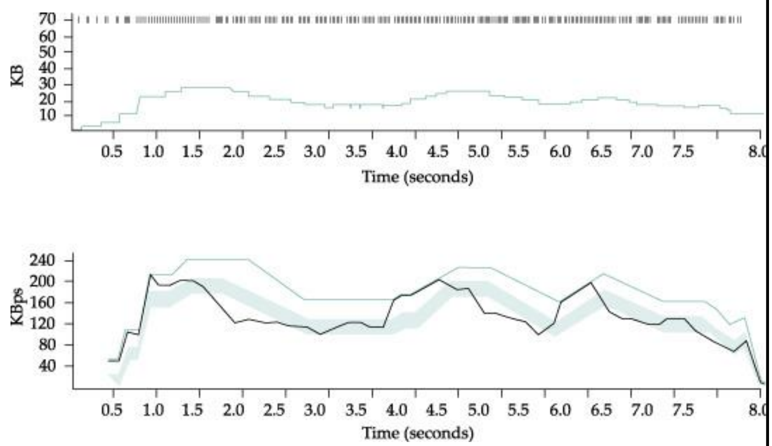
- Record baseRTT (minimum seen)
- Compute ExpectedRate = cwnd/baseRTT
- Diff = ExpectedRate - ActualRate
  - Diff is positive by definition
- When Diff < α, increase cwnd linearly
- When Diff > β, decrease cwnd linearly
  - α < β
  - When timeout occurs, decreases multiplicatively

Duke UNIVERSITY

42

# TCP Vegas: An Example

- Top: congestion window
- Bottom:
  - Blue: expected throughput
  - Black: actual throughput
  - Shaded, top: α away from expected
  - Shaded, bottom: β away from expected




Duke UNIVERSITY

43

# TCP Vegas Co-Existence With Other TCP Flavors

- Vegas backs off before other TCP variants do
  - Able to do it because it detects congestion early
- Ends up giving greater bandwidth to co-existing flows running e.g., TCP Reno

44

Duke UNIVERSITY

44

# Source-Based Congestion Avoidance Schemes: Key Points to Remember

- Watching, at the source, for signs of arising congestion
  - Typically increasing delays
- In TCP Vegas, compare expected throughput with achieved throughput
  - Back off when the throughput is far from expected

3-45

Duke UNIVERSITY

45

# Lecture Summary

- Issues in resource allocation
- Queuing disciplines
- Congestion avoidance: an overview
- Router-based congestion avoidance schemes: DECbit, RED, ECN
- Source-based congestion avoidance schemes: general approaches, TCP Vegas

Duke UNIVERSITY 46

46

# Next Lecture

- Quality of Service



Duke UNIVERSITY

47