# Principles of congestion control

*Computer Networking: A Top Down Approach*
6th edition
Jim Kurose, Keith Ross
Addison-Wesley

# Chapter 3 outline

3.1 Transport-layer services

3.2 Multiplexing and demultiplexing

3.3 Connectionless transport: UDP

3.4 Principles of reliable data transfer

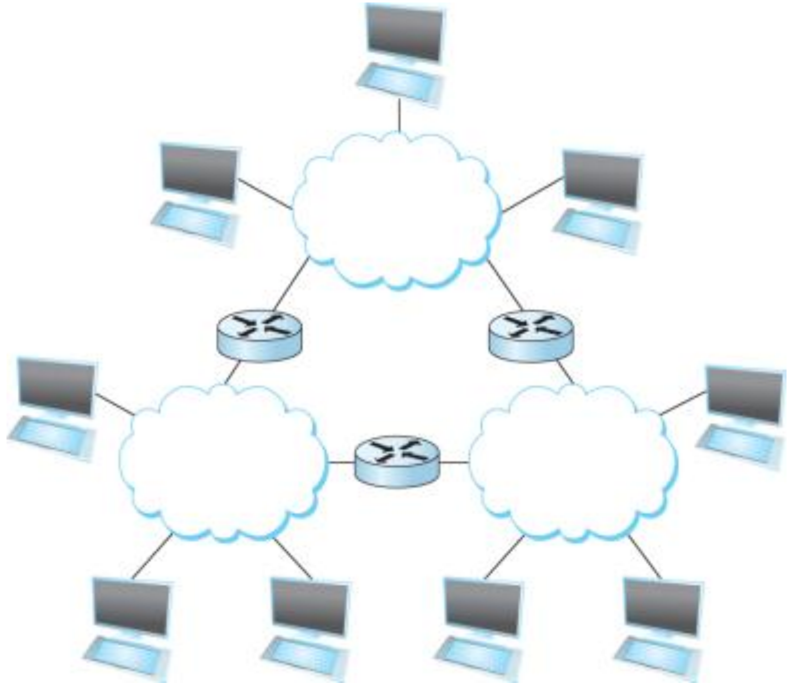
3.5 Connection-oriented transport: TCP
– Segment structure
– Reliable data transfer
– Flow control
– Connection management

3.6 Principles of congestion control

3.7 TCP congestion control

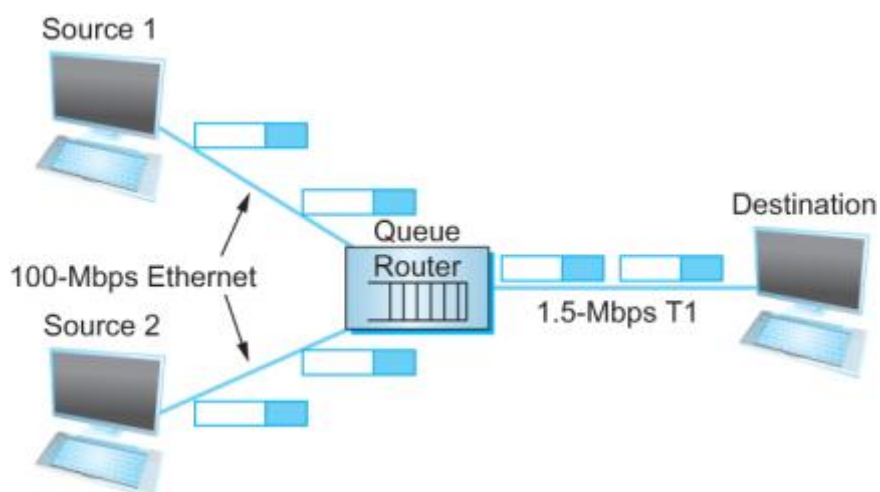
# IP best-effort network

- Best-effort model
  - Everybody can send
  - Network does the best it can to deliver
  - Delivery not guaranteed, some traffic may be dropped



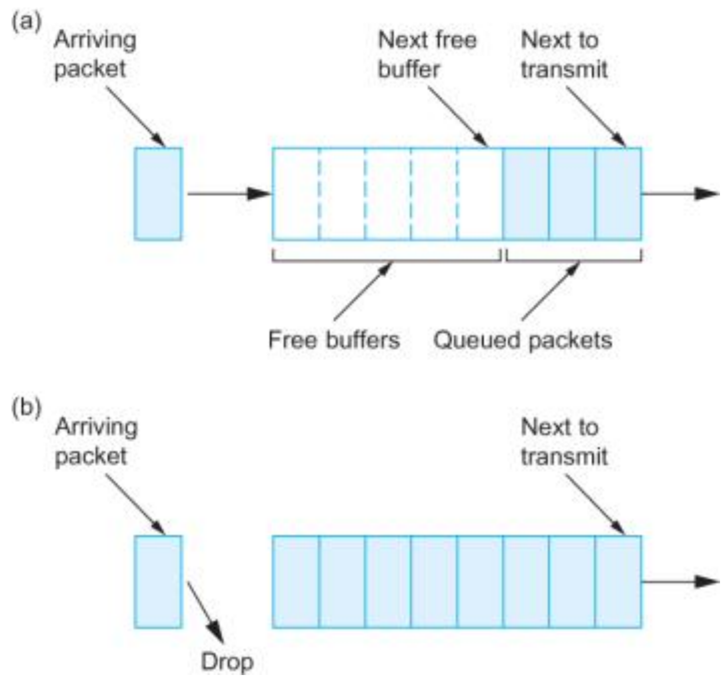
# Congestion unavoidable

- Multiple packets arrive at same time
  - Router can only transmit one
  - Router has to buffer remaining
- If too many arrive in a short time window
  - Buffer may overflow
  - Router has to choose some packets to drop



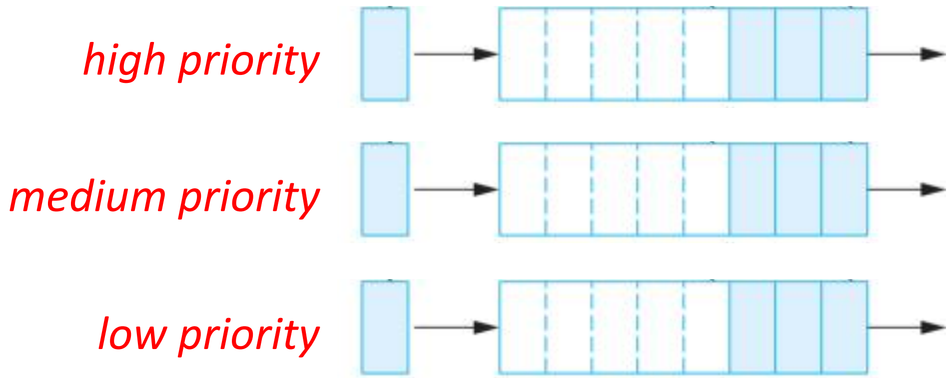
# What routers do

- Too many packets arrive too quickly
  - Which packets should we drop?
- First-in first-out (FIFO) with tail drop
  - Simple, drop the new guy that doesn't fit in your buffer



# Queuing disciplines

- Priority queuing
  - Packets marked with priority in header
  - Multiple FIFO queues, one for each priority class
  - Transmit high priority queues first
  - Who is allowed to set priority bit?



# Principles of congestion control

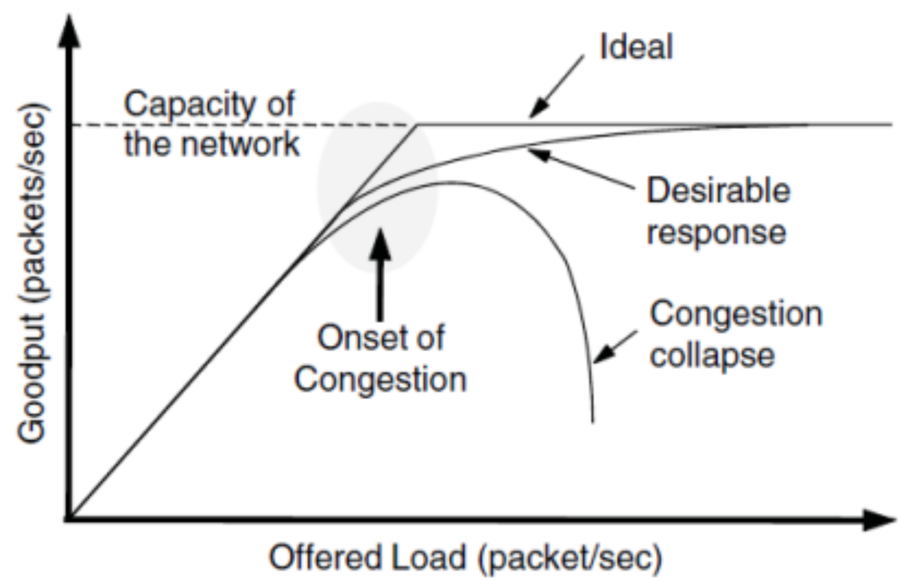
*Congestion*:

- Informally: "too many sources sending too much data too fast for *network* to handle"
- Different from flow control!
- Manifestations:
  - Lost packets (buffer overflow at routers)
  - Long delays (queueing in router buffers)
- A top-10 problem!

# Congestion collapse

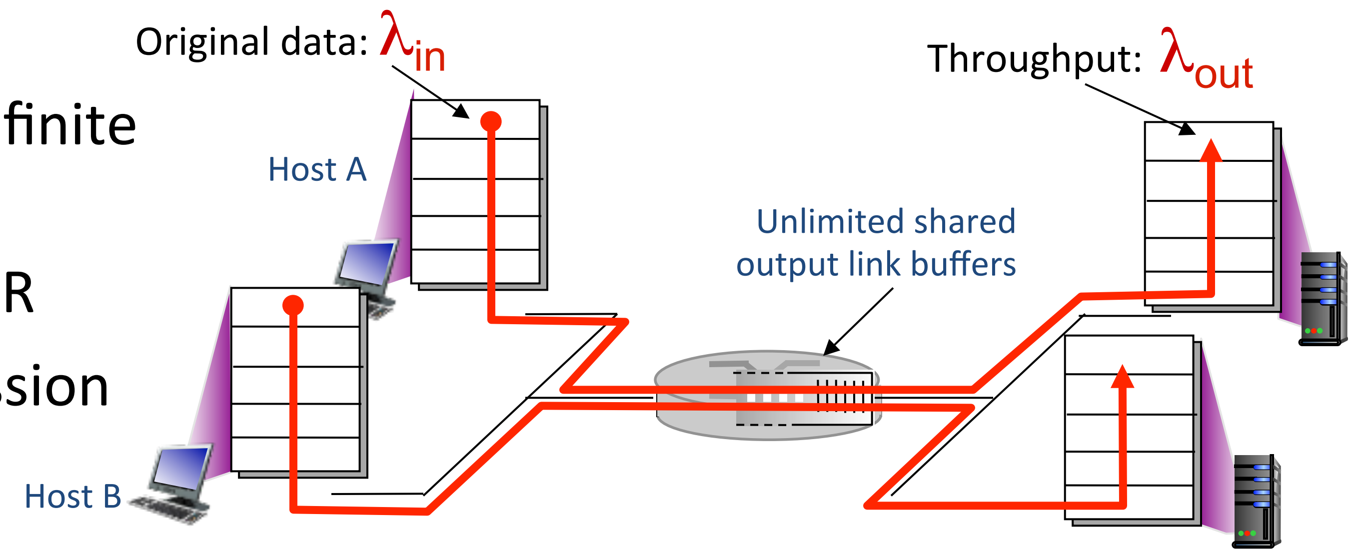
- Congestion collapse
  - 1986, NSF backbone dropped from 32 kbps to 40 bps
    - Hosts send packets as fast as advertised window allowed
    - When packets dropped, hosts retransmit causing more congestion
  - Goodput = useful bits delivered per unit time
    - Excludes header overhead, retransmissions, etc.

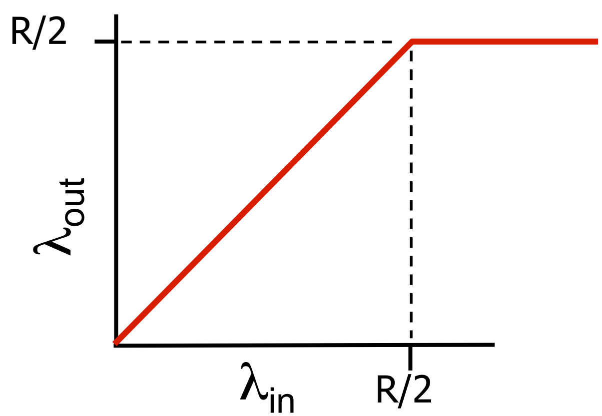
> "In October of '86, the Internet had the first of what became a series of 'congestion collapses'. During this period, the data throughput from LBL to UC Berkeley (sites separated by 400 yards and two IMP hops) dropped from 32 Kbps to 40 bps. We were fascinated by this sudden factor-of-thousand drop in bandwidth and embarked on an investigation of why things had gotten so bad." –Van Jacobson

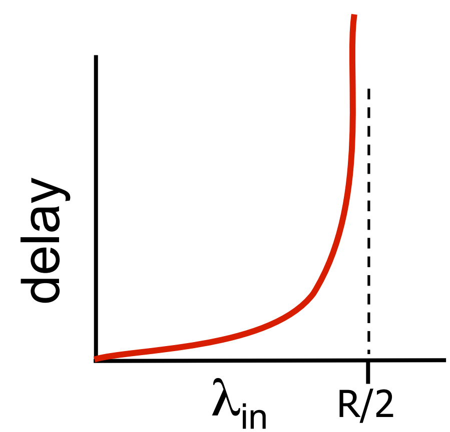


# Causes/costs of congestion: scenario 1

- Two senders, two receivers
- One router, infinite buffers
- Link capacity: R
- No retransmission



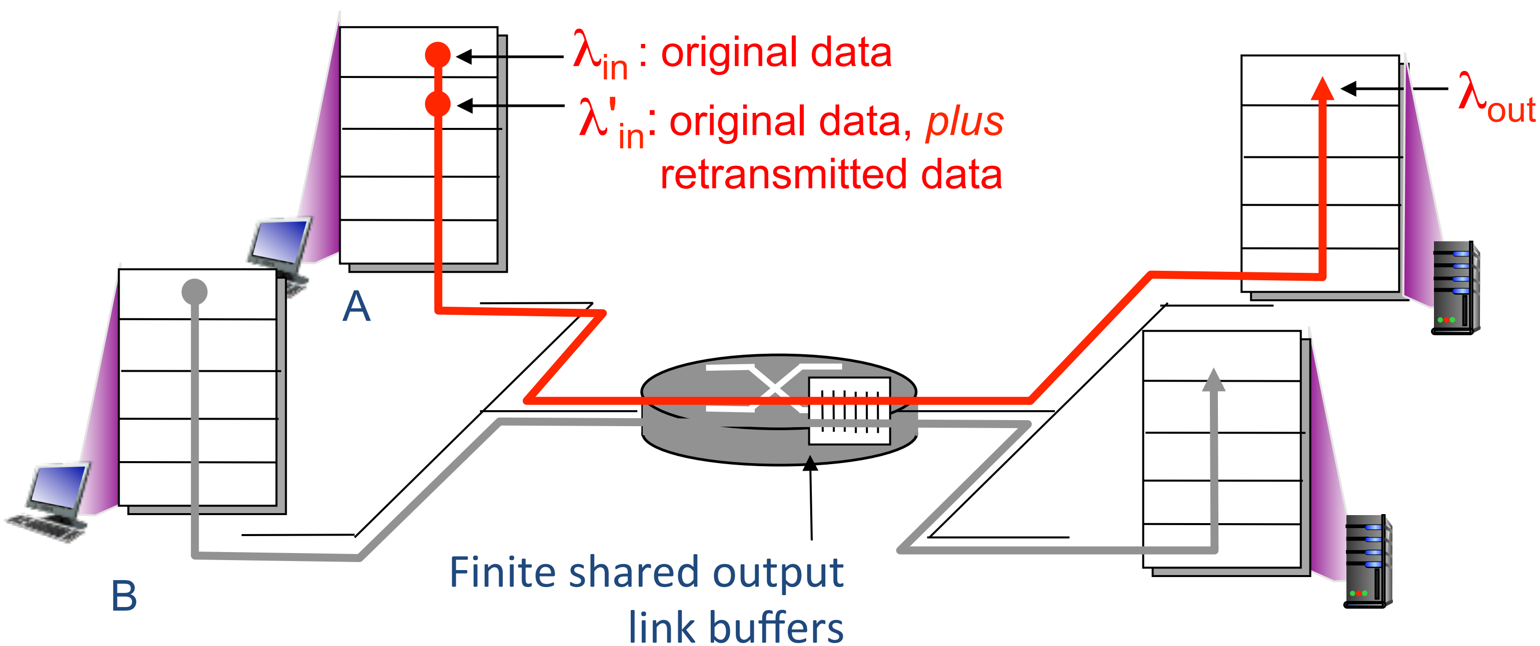




- Maximum per-connection throughput: R/2
- Large delays as arrival rate, $\lambda_{in}$, approaches capacity

# Causes/costs of congestion: scenario 2

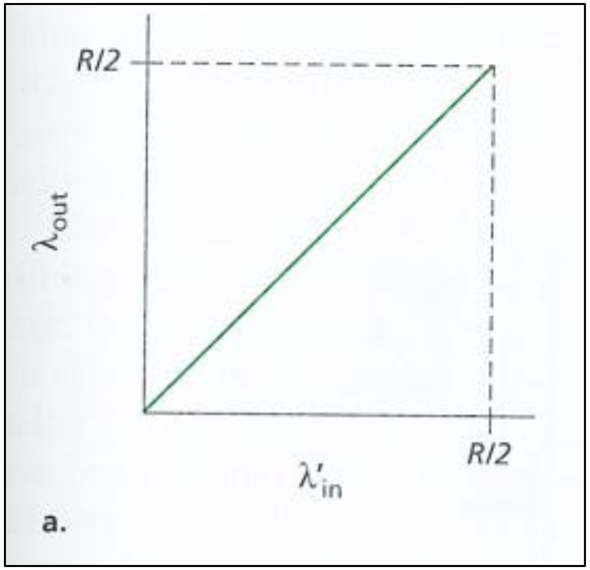
- One router, *finite* buffers, reliable connection
- Sender retransmission of timed-out packet
  - Application-layer input = application-layer output: $\lambda_{in} = \lambda_{out}$
  - Transport-layer input includes *retransmissions* : $\lambda'_{in} \geq \lambda_{in}$
  - $\lambda'_{in}$ = *offered load*

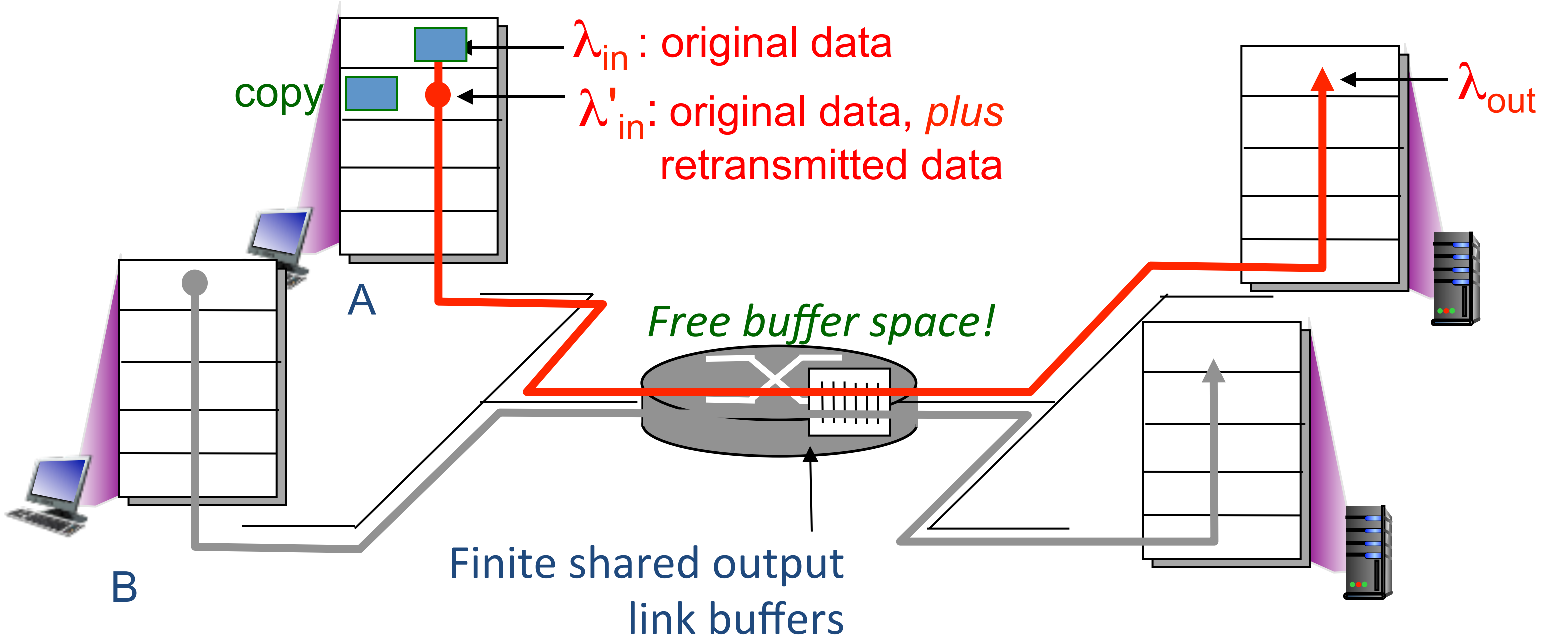


# Congestion scenario 2a: ideal case

## Idealization: *perfect knowledge*

- Sender magically sends only when router buffers available
- No loss, $\lambda'_{in} = \lambda_{in}$
- Hosts won't send faster than R/2

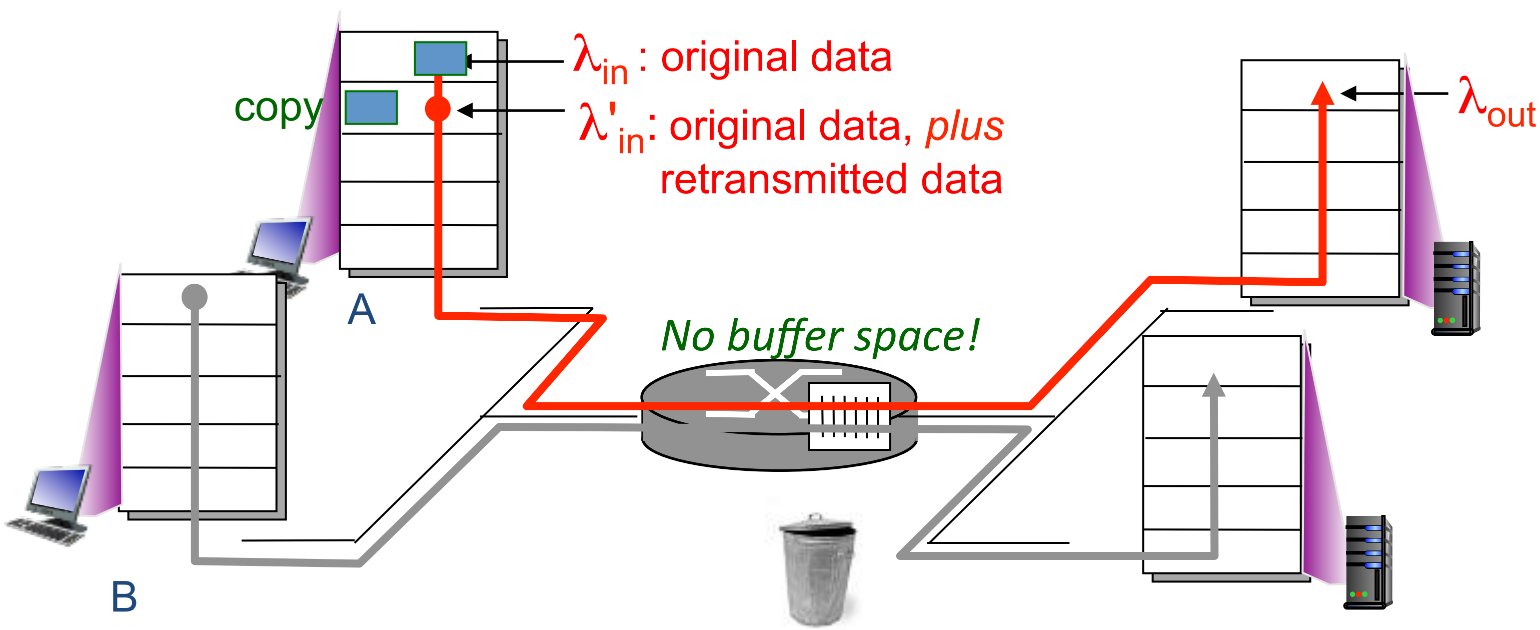


# Congestion scenario 2b: known loss

## Idealization: *known loss*

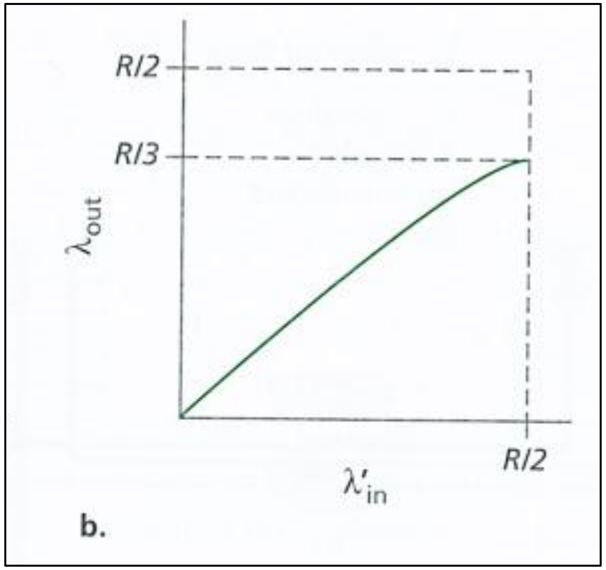
- Packets can be lost, dropped at router due  to full buffers
- Sender only resends if packet *known* to be lost

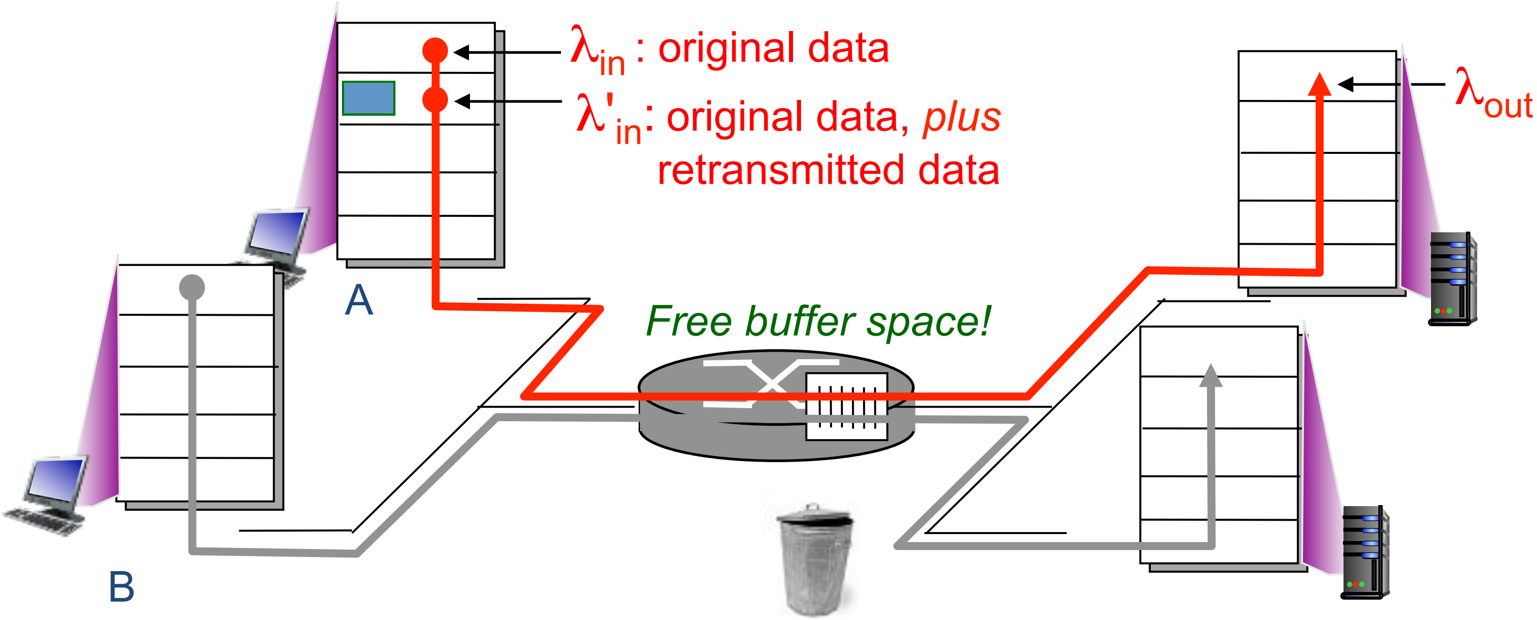


# Congestion scenario 2b: known loss

## Idealization: *known loss*

- Packets can be lost, dropped at router due to full buffers
- Sender only resends if packet *known* to be lost

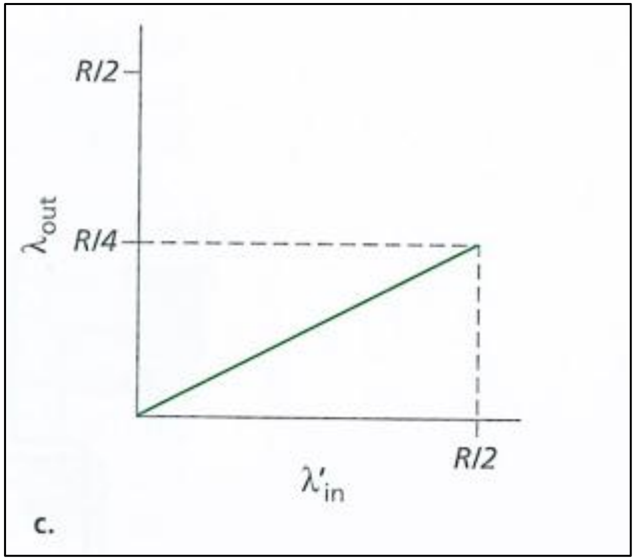


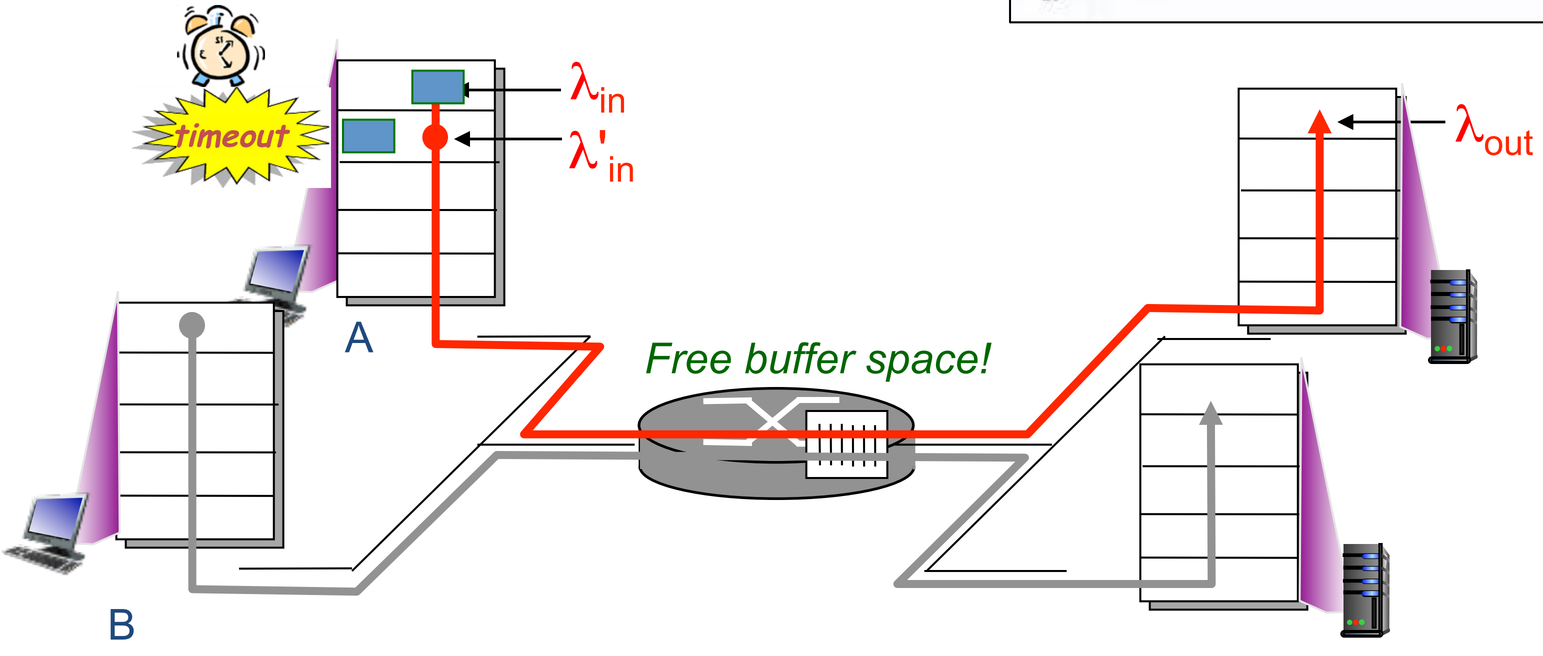



# Congestion scenario 2c: duplicates

## Realistic: *duplicates*

- Packets can be lost, dropped at router due to full buffers
- Sender times out prematurely, sending *two* copies, both of which are delivered

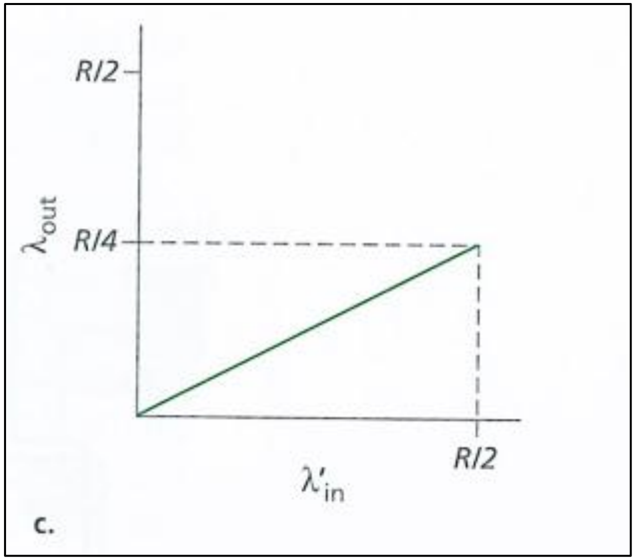


# Congestion scenario 2c: duplicates

## Realistic: *duplicates*

- Packets can be lost, dropped at router due to full buffers
- Sender times out prematurely, sending *two* copies, both of which are delivered




## Costs of congestion:

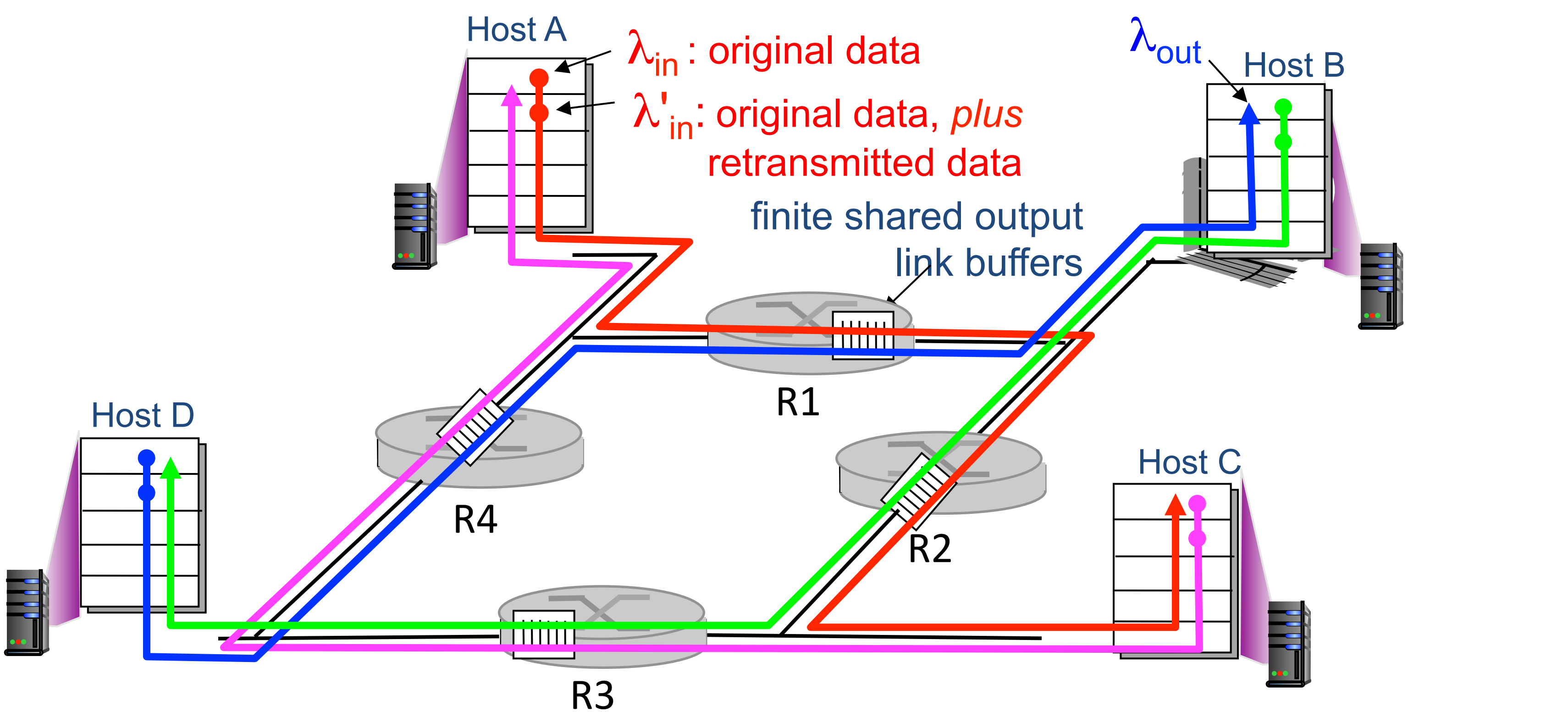
- More work (retransmissions) for given goodput
- Unneeded retransmissions
  - Link carries multiple copies of packet
  - Decreases goodput

# Causes/costs of congestion: scenario 3

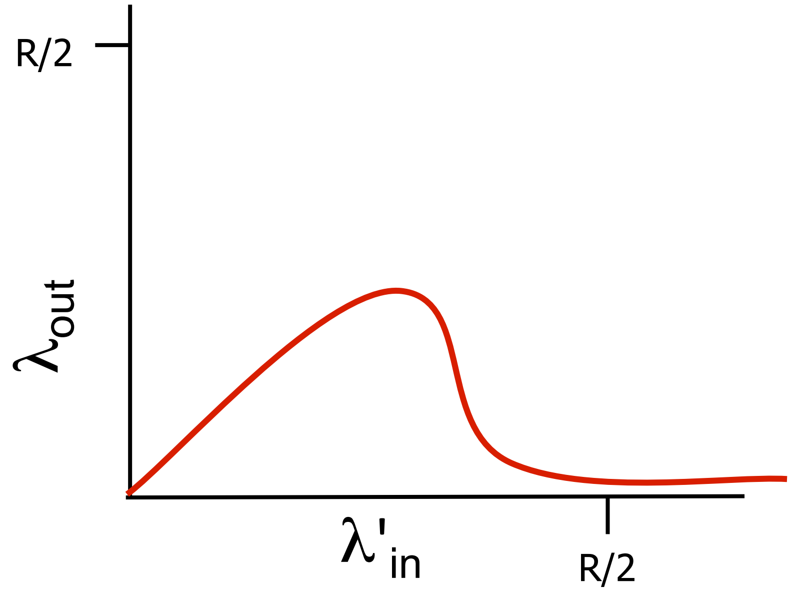
- Four senders
- Multihop paths
- Timeout/retransmit

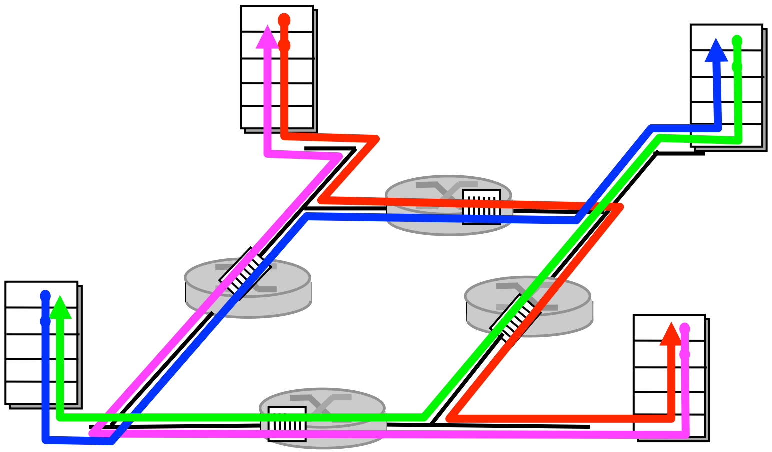
**Q:** What happens as $\lambda_{in}$ and $\lambda'_{in}$ increase?

**A:** As red $\lambda'_{in}$ increases, all arriving blue pkts at upper queue are dropped, blue throughput → 0



# Causes/costs of congestion: scenario 3



## Another cost of congestion:

- When packet dropped, any upstream transmission capacity used for that packet was wasted!

# Approaches to congestion control

Two broad approaches towards congestion control:

## End-end:

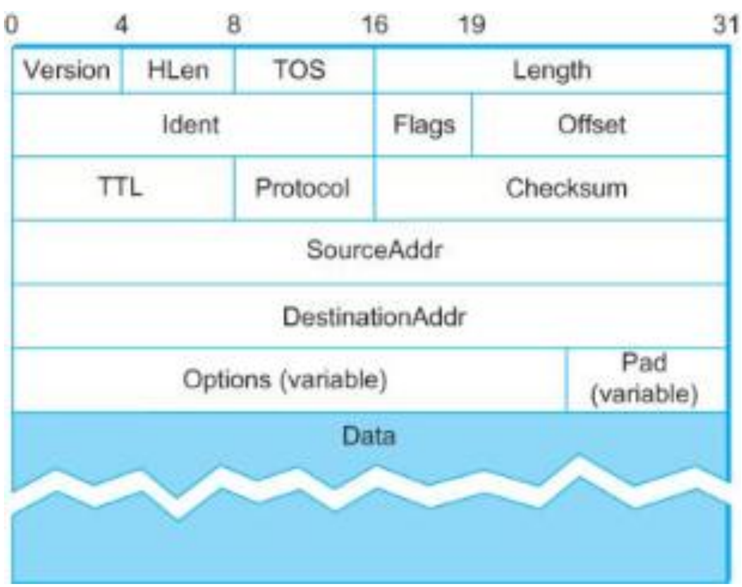
- No explicit feedback from network
- Congestion inferred from end-system observed loss, delay
- Approach taken by TCP

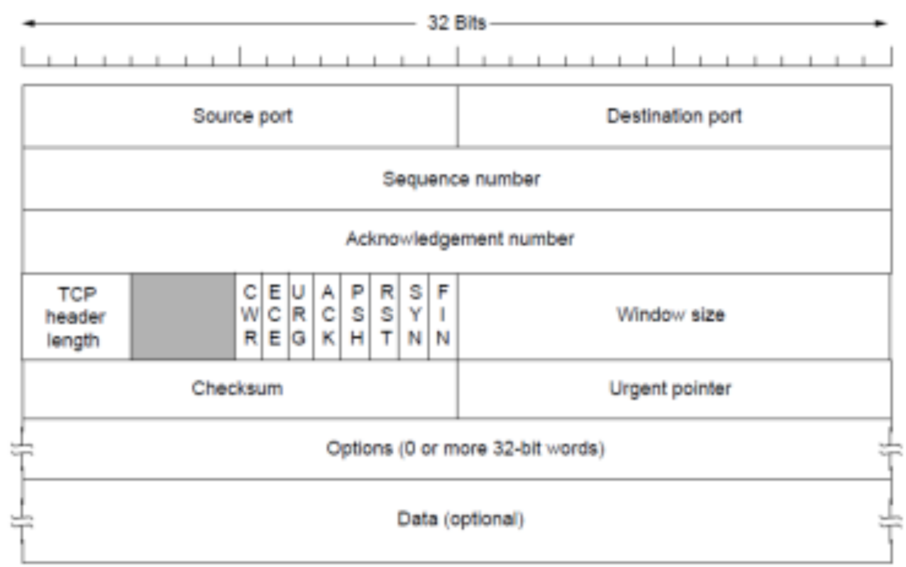
## Network-assisted:

- Routers provide feedback to end systems
  - Single bit indicating congestion (SNA, DECbit, TCP/IP ECN, ATM)
  - Explicit feedback to sender via *choke packet*

# Router signaling

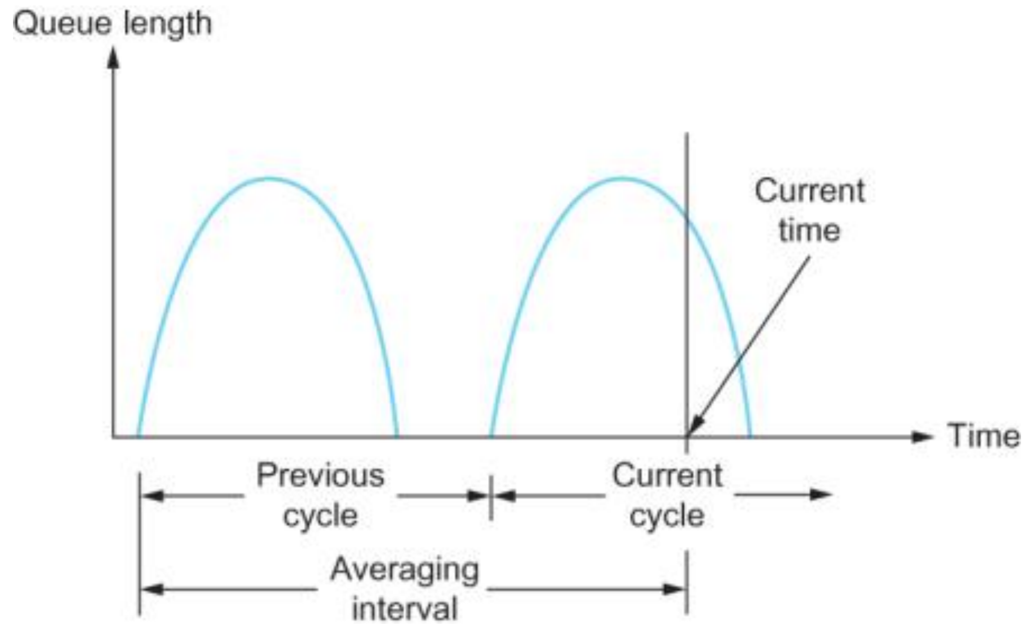
- Explicit Congestion Notification (ECN)
  - Sender sets TOS IP header bit saying it supports ECN
  - If ECN-aware router is congested, marks another TOS bit
  - TCP receiver sees IP congestion bit, informs sender via TCP segment ECN-Echo (ECE) bit
  - TCP sender confirms receipt of ECE with Congestion Window Reduced (CWR) bit



# Router signaling

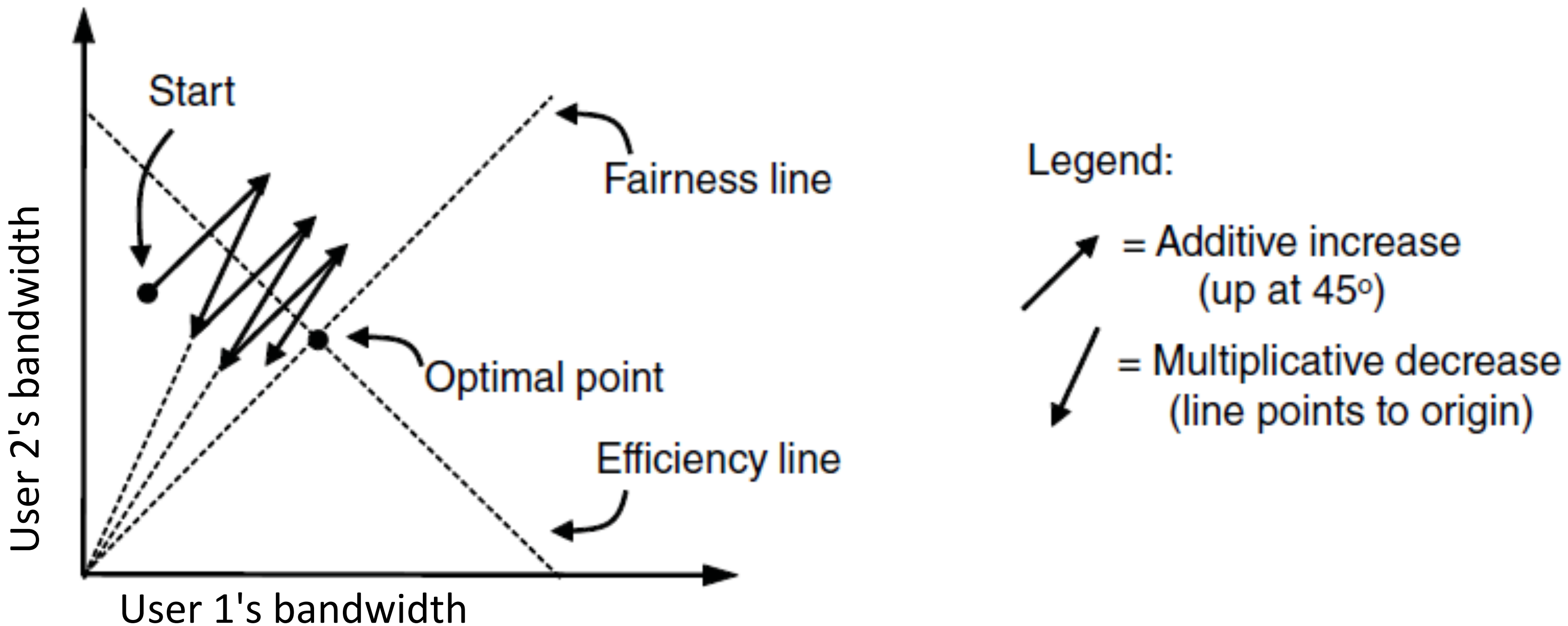
- How does router determine congestion?
  - Checks avg. queue length spanning last busy + idle cycle + current cycle




- What does TCP sender do with congestion signals?
  - Checks fraction of packets having congestion bit set
  - If < 50%, increase congestion window by one packet
  - If > 50%, decrease congestion window by 0.875

# AIMD principle

- Additive increase, multiplicative decrease (AIMD)
  - Additive increase: On success of last packet, increase number of packets in-flight by one
  - Multiplicative decrease: On loss of packet, divide number of allowed in-flight packets in half



# Summary

- Principles of congestion control
  - Too many senders can lead to congestion collapse
    - Links between routers have limited bandwidth
    - Router queues are finite
    - Traffic patterns are unpredictable
  - Goodput = useful bits delivered per unit time
  - Broad approaches
    - End-to-end, no information from routers
    - Network assisted, routers warn when congestion occurring (or about to)
  - AIMD principle
    - Two competing senders achieve efficiency & fairness