15-441 Computer Networking

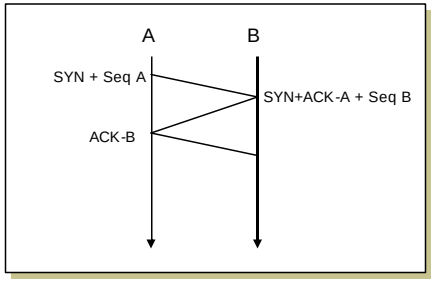
Lecture 8 – TCP & Congestion Control

Outline

- TCP connection setup/data transfer
- TCP Reliability
- Congestion sources and collapse
- Congestion control basics

Connection Establishment

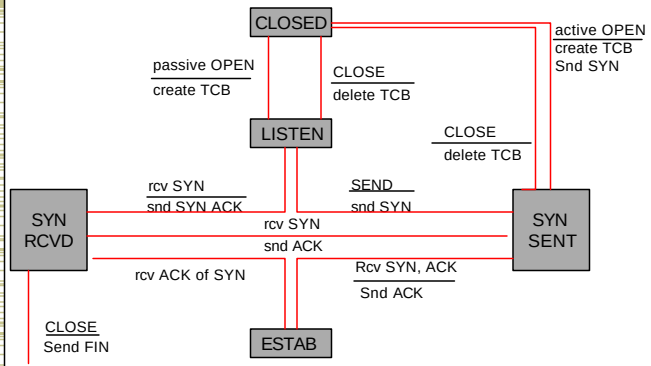
- A and B must agree on initial sequence number selection
  - Use 3-way handshake



Sequence Number Selection

- Why not simply chose 0?
  - Must avoid overlap with earlier incarnation
  - Security issues

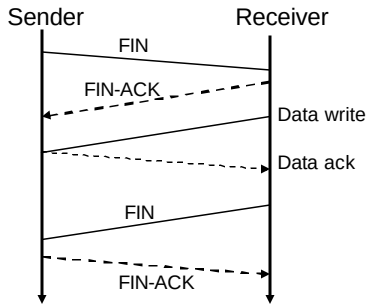
Connection Setup



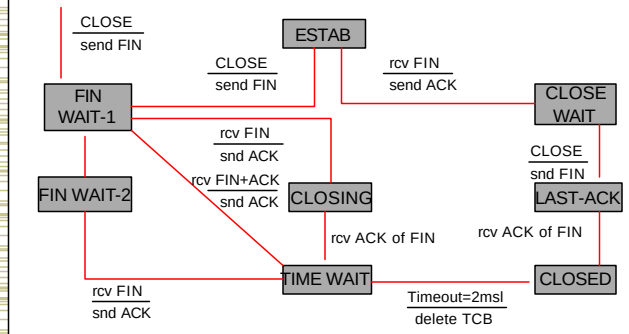
Connection Tear-down

- Normal termination
  - Allow unilateral close
- TCP must continue to receive data even after closing
- Cannot close connection immediately
  - What if a new connection restarts and uses same sequence number?

Tear-down Packet Exchange



Connection Tear-down



Detecting Half-open Connections

	TCP A		TCP B
1.	(CRASH)		(send 300, receive 100)
2.	CLOSED		ESTABLISHED
3.	SYN-SENT → <SEQ=400><CTL=SYN>	→	(??)
4.	(!!)	← <SEQ=300><ACK=100><CTL=ACK>	← ESTABLISHED
5.	SYN-SENT → <SEQ=100><CTL=RST>	→	(Abort!!)
6.	SYN-SENT		CLOSED
7.	SYN-SENT → <SEQ=400><CTL=SYN>	→	

Outline

- TCP connection setup/data transfer
- TCP Reliability
- Congestion sources and collapse
- Congestion control basics

Reliability Challenges

- Congestion related losses
- Variable packet delays
  - What should the timeout be?
- Reordering of packets
  - Ensure sequences numbers are not reused
  - How long do packets live?
    - MSL = 120 seconds based on IP behavior

TCP = Go-Back-N Variant

- Sliding window with cumulative acks
  - Receiver can only return a single “ack” sequence number to the sender.
  - Acknowledges all bytes with a lower sequence number
  - Starting point for retransmission
  - Duplicate acks sent when out-of-order packet received
- But: sender only retransmits a single packet.
  - Reason???
- Error control is based on byte sequences, not packets.
  - Retransmitted packet can be different from the original lost packet – Why?

Round-trip Time Estimation

- Wait at least one RTT before retransmitting
- Importance of accurate RTT estimators:
  - Low RTT → unneeded retransmissions
  - High RTT → poor throughput
- RTT estimator must adapt to change in RTT
  - But not too fast, or too slow!
- Spurious timeouts
  - "Conservation of packets" principle – never more than a window worth of packets in flight

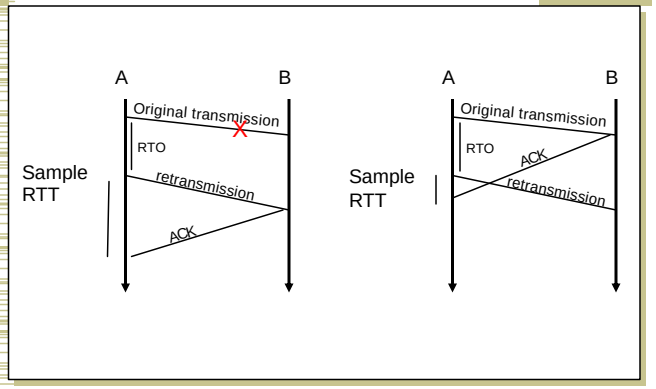
Initial Round-trip Estimator

- Round trip times exponentially averaged:
  - New RTT = α (old RTT) + (1 - α) (new sample)
  - Recommended value for α: 0.8 - 0.9
    - 0.875 for most TCP's
- Retransmit timer set to β RTT, where β = 2
  - Every time timer expires, RTO exponentially backed-off
  - Like Ethernet
- Not good at preventing spurious timeouts
  - Why?

Jacobson's Retransmission Timeout

- Key observation:
  - At high loads round trip variance is high
- Solution:
  - Base RTO on RTT and standard deviation
  - rttvar = χ * dev + (1- χ)rttvar
    - Dev = linear deviation
    - Inappropriately named – actually smoothed linear deviation

Retransmission Ambiguity



Karn's RTT Estimator

- Accounts for retransmission ambiguity
- If a segment has been retransmitted:
  - Don't count RTT sample on ACKs for this segment
  - Keep backed off time-out for next packet
  - Reuse RTT estimate only after one successful transmission

Timestamp Extension

- Used to improve timeout mechanism by more accurate measurement of RTT
- When sending a packet, insert current timestamp into option
  - 4 bytes for timestamp, 4 bytes for echo
- Receiver echoes timestamp in ACK
  - Actually will echo whatever is in timestamp
- Removes retransmission ambiguity
  - Can get RTT sample on any packet

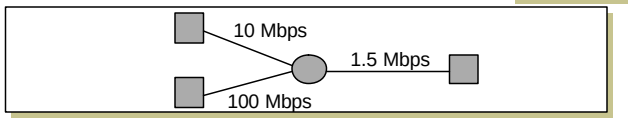
Timer Granularity

- Many TCP implementations set RTO in multiples of 200,500,1000ms
- Why?
  - Avoid spurious timeouts – RTTs can vary quickly due to cross traffic
  - Make timers interrupts efficient
- What happens for the first couple of packets?
  - Pick a very conservative value (seconds)

Outline

- TCP connection setup/data transfer
- TCP Reliability
- Congestion sources and collapse
- Congestion control basics

Congestion

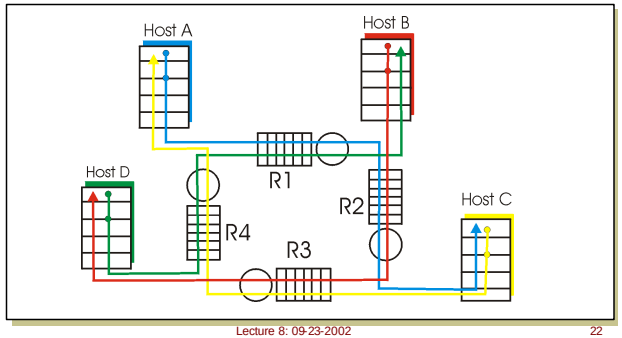


- Different sources compete for resources inside network
- Why is it a problem?
  - Sources are unaware of current state of resource
  - Sources are unaware of each other
- Manifestations:
  - Lost packets (buffer overflow at routers)
  - Long delays (queuing in router buffers)
  - In many situations will result in < 1.5 Mbps of throughput for the above topology (congestion collapse)

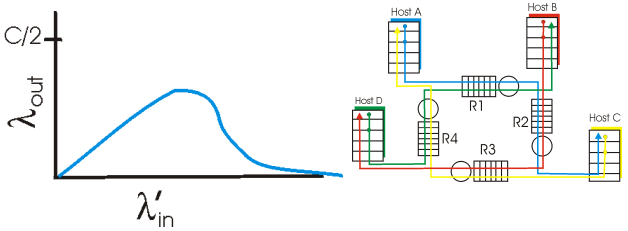
Causes & Costs of Congestion

- Four senders – multihop paths
- Timeout/retransmit

Q: What happens as rate increases?



Causes & Costs of Congestion



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

Congestion Collapse

- Definition: *Increase in network load results in decrease of useful work done*
- Many possible causes
  - Spurious retransmissions of packets still in flight
    - Classical congestion collapse
    - How can this happen with packet conservation
    - Solution: better timers and TCP congestion control
  - Undelivered packets
    - Packets consume resources and are dropped elsewhere in network
    - Solution: congestion control for ALL traffic

Other Congestion Collapse Causes

- Fragments
  - Mismatch of transmission and retransmission units
  - Solutions
    - Make network drop all fragments of a packet (early packet discard in ATM)
    - Do path MTU discovery
- Control traffic
  - Large percentage of traffic is for control
    - Headers, routing messages, DNS, etc.
- Stale or unwanted packets
  - Packets that are delayed on long queues
  - "Push" data that is never used

Congestion Control and Avoidance

- A mechanism which:
  - Uses network resources efficiently
  - Preserves fair network resource allocation
  - Prevents or avoids collapse
- Congestion collapse is not just a theory
  - Has been frequently observed in many networks

Approaches Towards Congestion Control

- Two broad approaches towards congestion control:

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

- Network-assisted congestion control:
  - Routers provide feedback to end systems
    - Single bit indicating congestion (SNA, DECbit, TCP/IP ECN, ATM)
    - Explicit rate sender should send at
  - Problem: makes routers complicated

Example: TCP Congestion Control

- Very simple mechanisms in network
  - FIFO scheduling with shared buffer pool
  - Feedback through packet drops
- TCP interprets packet drops as signs of congestion and slows down
  - This is an assumption: packet drops are not a sign of congestion in all networks
    - E.g. wireless networks
- Periodically probes the network to check whether more bandwidth has become available.

Outline

- TCP connection setup/data transfer
- TCP Reliability
- Congestion sources and collapse
- Congestion control basics

Objectives

- Simple router behavior
- Distributedness
- Efficiency: $X = \Sigma x_i(t)$
- Fairness: $(\Sigma x_i)^2/n(\Sigma x_i^2)$
- Convergence: control system must be stable

Basic Control Model

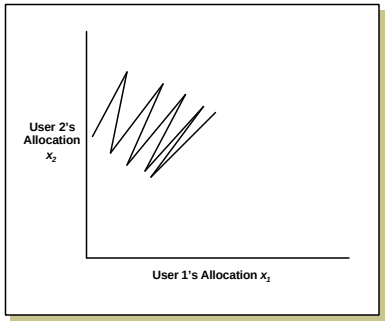
- Reduce speed when congestion is perceived
  - How is congestion signaled?
    - Either mark or drop packets
  - How much to reduce?
- Increase speed otherwise
  - Probe for available bandwidth – how?

Linear Control

- Many different possibilities for reaction to congestion and probing
  - Examine simple linear controls
    - Window(t + 1) = a + b Window(t)
    - Different a_i/b_i for increase and a_d/b_d for decrease
- Supports various reaction to signals
  - Increase/decrease additively
  - Increased/decrease multiplicatively
  - Which of the four combinations is optimal?

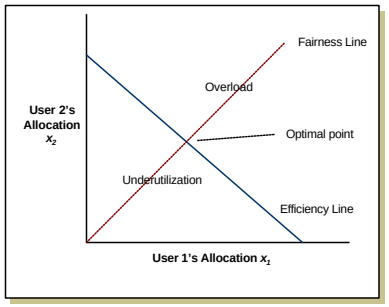
Phase Plots

- Simple way to visualize behavior of competing connections over time



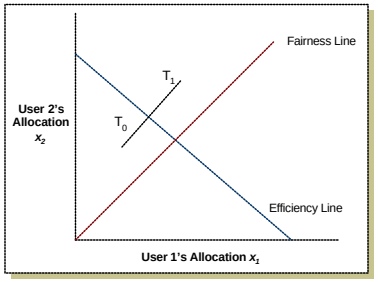
Phase Plots

- What are desirable properties?
- What if flows are not equal?



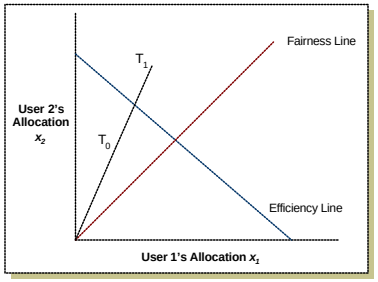
Additive Increase/Decrease

- Both X_1 and X_2 increase/ decrease by the same amount over time
  - Additive increase improves fairness and additive decrease reduces fairness

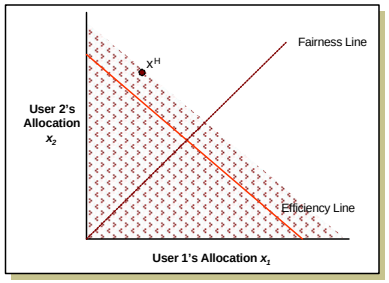


Muliplicative Increase/Decrease

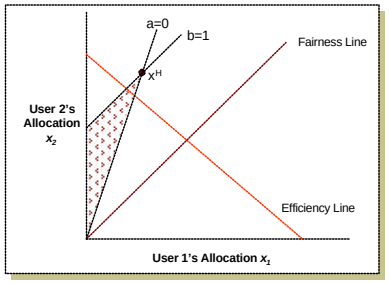
- Both X_1 and X_2 increase by the same factor over time
  - Extension from origin – constant fairness



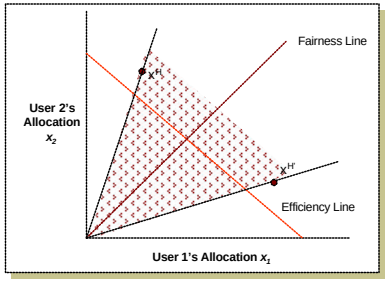
Convergence to Efficiency



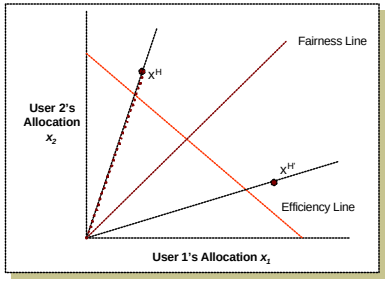
Distributed Convergence to Efficiency



Convergence to Fairness

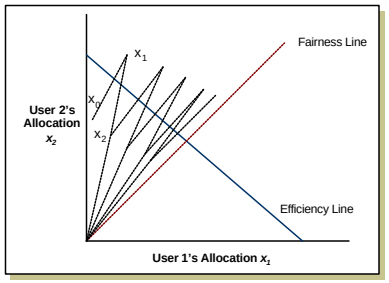


Convergence to Efficiency & Fairness



What is the Right Choice?

- Constraints limit us to AIMD
  - Can have multiplicative term in increase
  - AIMD moves towards optimal point



Next Lecture

- TCP Congestion Control
- TCP Loss Recovery
- TCP Modeling