CS551
TCP Congestion Control

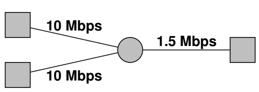
Bill Cheng

http://merlot.usc.edu/cs551-f12

Causes and Costs of Congestion

- Queueing delays in router as packet arrival rate nears link capacity
  - even if routers have infinite buffer space
    - costs: wasting bandwidth to forward unneeded copies
- Retransmissions costs: (routers have finite buffer, so packet get dropped)
  - routers have finite buffer (packets get dropped)
  - retransmitted data eat up bandwidth
  - when a packet is dropped along a path, the transmission capacity that was used at each of the upstream routers to forward that packet was wasted
- The theory behind congestion control
  - stability
  - efficiency

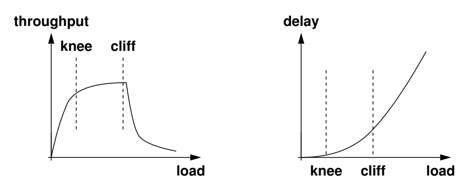
Congestion



- If both sources send at full speed, the router is overwhelmed
  - *congestion collapse:* senders lose data from congestion and they resend, causing *more* congestion (can be self-reinforcing)
- Other forms of congestion collapse:
  - Retransmissions of large packets after loss of a single fragment
  - Non-feedback controlled sources

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



Congestion Control vs. Flow Control

- What does flow control do?
  - avoids overrunning the receiver
- What does congestion control do?
  - avoids overrunning router buffers and saturating the network
- What mechanism do they use?
  - both use windows: *wnd* for flow control and *cwnd* for congestion control, actual window used is *min(wnd,cwnd)*

Congestion Control Goals

- Efficiency (maximize throughput or power [Ramakrishnan90a])
- Fairness [Ramakrishnan90a]
- Stability [Jacobson88a]

Fairness

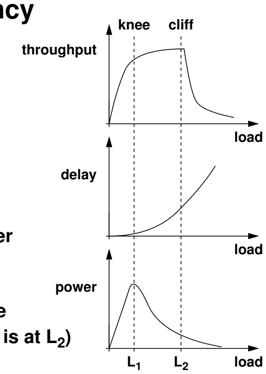
- Should treat all users equally
  - But, defining fairness is hard... what is a user?
    - host, flow, person?
  - *n* flows through a link, each flow should get *1/n* bandwidth
    - what if their needs are different?
- Measuring fair allocations [Ramakrishnan90a]
  - In the absence of knowing requirements, assume a fair allocation means equal allocation
  - Jain and Chiu's fairness index: $(\sum x_i)^2 / n (\sum x_i^2)$
    - x_i = throughput of flow i
    - *Ex:* fairness index = *1* if all x_i are equal
    - *Ex:* fairness index = *k/n* if *k* out of *n* flows are equal and other flows *(n-k)* receives *0* throughput
- Other schemes, e.g., fair queueing [Demers89a]

Efficiency

- Want most throughput with low delay
  - System is most efficient at knee of curve
  - Power [Ramakrishnan90a]

$$power = \frac{throughput^{\alpha}}{delay}$$

    - 0<α<1, α=1 results in power being maximized at the knee of the curve
      - (others may say that the knee of the delay curve is at L_2)



Congestion Control Design

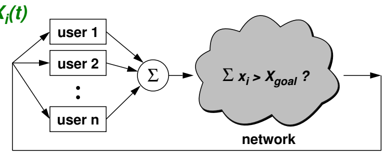
- Avoidance or control?
  - *Avoidance* keeps system at knee of curve
  - But, to do that, need routers to send *accurate signals* (some feedback)
    - this is what ECN tries to accomplish
    - another possibility is to use rate (in the future)
  - *Control* responds to loss after the fact
- Sending host must adjust amount of data it puts in the network based on detected congestion
  - TCP uses its window to do congestion control
    - but also avoidance, sort of
  - But what's the right strategy to increase/decrease window (slow start, congestion avoidance, exponential backoff)

How To Adjust Window in TCP?

- When to increase/decrease cwnd?
- A control theory problem
  - Observe network
  - Reduce window when congestion is perceived
  - Increase window otherwise
- Constraints:
  - Efficiency
  - Fairness
  - Stability or convergence (too much oscillation is bad)
  - Out-of-date information
    - RTT is fundamental limit to how quickly you can react

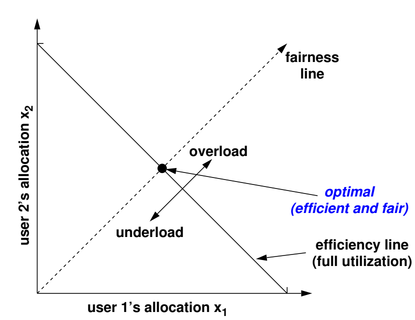
Linear Control

- $X_i(t+1) = a_i(t) + b_i(t)X_i(t)$

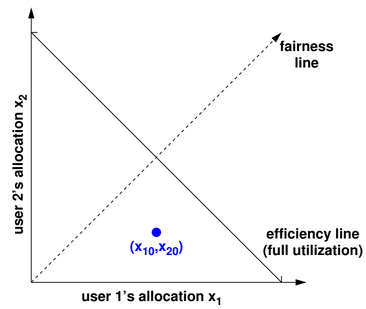


- Formulation allows for the feedback signal:
  - to change additively: $a_i(t)$
  - to change multiplicatively: $b_i(t)$
  - can consider feedback
- What does TCP do?
  - *AIMD: additive increase, multiplicative decrease*
  - maximize stability: slow increase, fast decrease

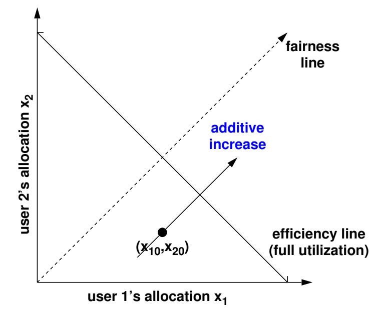
Linear Control Example [Chiu89a]



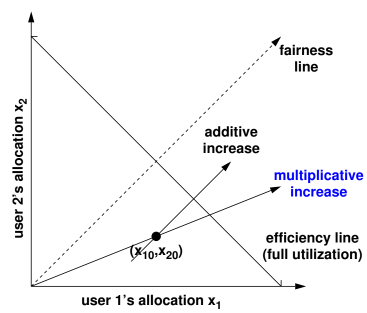
Linear Control Example [Chiu89a]



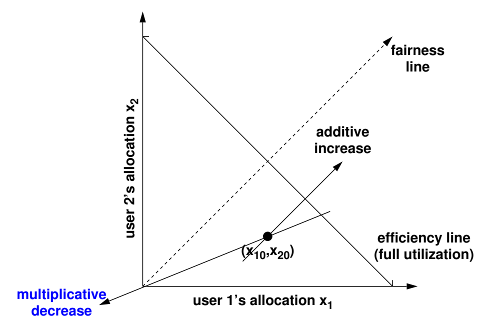
Linear Control Example [Chiu89a]



Linear Control Example [Chiu89a]



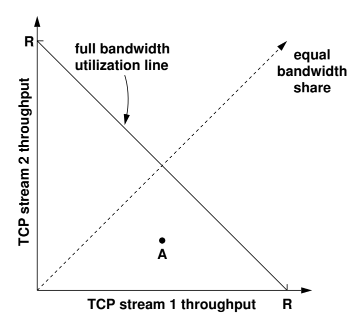
Linear Control Example [Chiu89a]



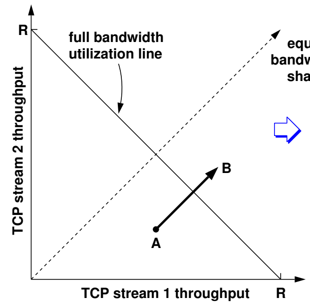
Linear Control Result [Chiu89a]

- Efficiency, fairness and distributedness imply that
  - Decrease must be mutiplicative
  - Increase must be additive and may have a multiplicative factor
    - smoothness implies that multiplicative increase factor must be exactly 1

TCP Equally Shares Bandwidth On A Congested Link

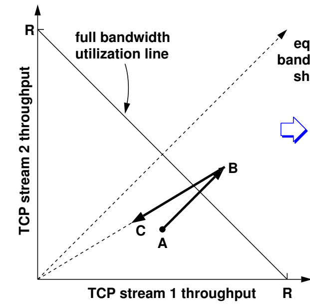


TCP Equally Shares Bandwidth On A Congested Link (Cont...)



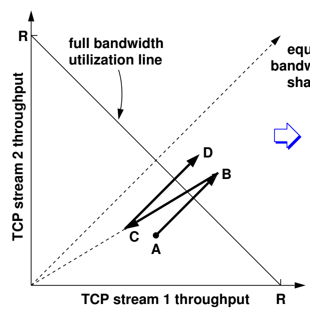
- A → B:
  - no slow start
  - congestion avoidance for both streams
  - paralle to the 45-degree line

TCP Equally Shares Bandwidth On A Congested Link (Cont...)



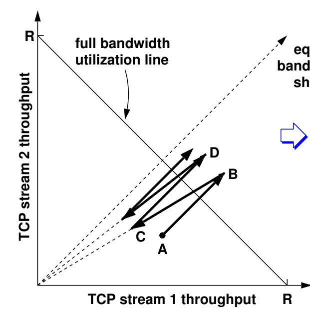
- B → C:
  - packet loss, so ssthresh = cwnd/2
  - C is half way between B and origin

TCP Equally Shares Bandwidth On A Congested Link (Cont...)



- C → D:
  - congestion avoidance

TCP Equally Shares Bandwidth On A Congested Link (Cont...)



- What if B → C does not go towards the origin?
  - unfare?