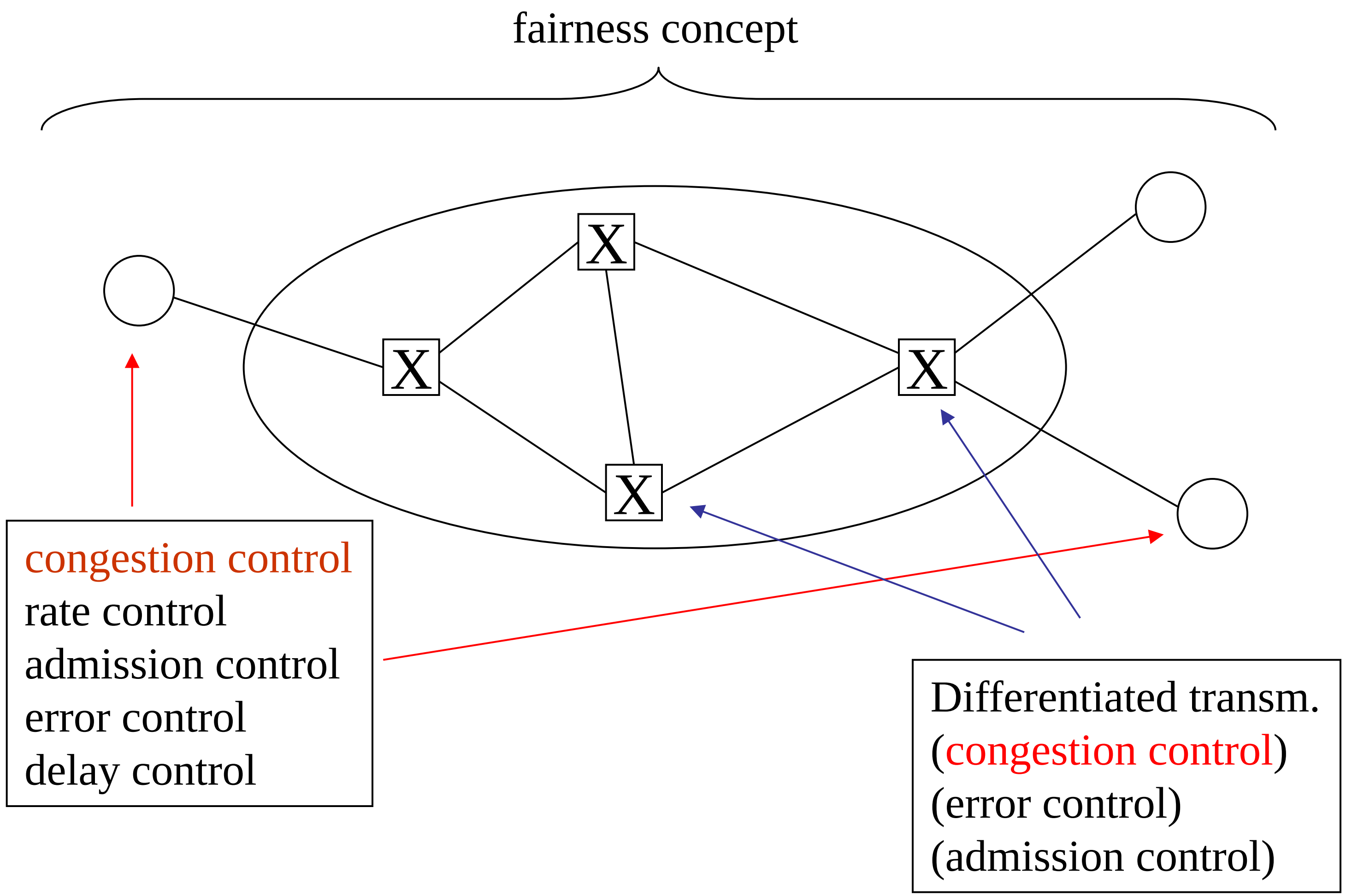
EP2200
Performance analysis of Communication networks

Topic 3
Congestion and rate control

Congestion, rate and error control

- Lecture material:
  - Bertsekas, Gallager, *Data networks*, 6.1-2
  - I. Kay, *Stochastic modeling*, 3.3.4
  - J-Y Le Boudec, “Rate adaptation, congestion control and fairness: a tutorial,” Nov. 2005

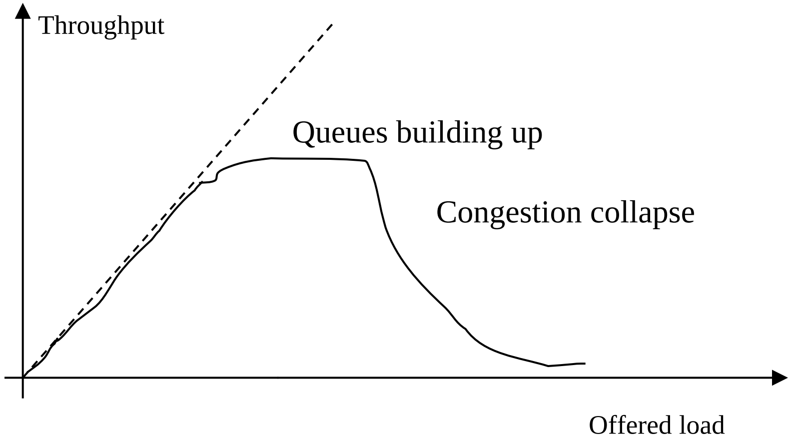
Control functions in communication networks



Congestion control

- To regulate the packet population in the network
- To share resources (link bandwidth, buffer space)
- *Flow control: between two users for speed matching,* sometimes means congestion control in the literature
- The main objectives of congestion control
  - Efficiency
    - high utilization (from the network provider's perspective)
    - high per flow throughput, low delay (from users' perspective)
  - Fairness: fair allocation of resources

Congestion control



- What happens if the incoming traffic is not restricted?
  - Bottleneck links: the offered traffic is higher than the link transmission capacity: temporarily (bursts arriving) or permanently
- What happens at bottleneck links?
  - queue sizes grow, end-to-end delays increase
  - queue space fills up, packets get dropped
  - packets are retransmitted by the applications, further increasing the load
- Congestion collapse: the network throughput decreases and delays become excessive

Congestion control techniques

- Should depend on the service requirements of the application
  - voice, video (streaming): minimum bandwidth requirement, some loss is allowed, delay sensitive
  - data (elastic): lossless end-to-end transmission required

- Call blocking at the network edge + rate control
  - calls blocked if resources are not available
  - the rate of accepted calls is controlled
- Packet discarding at a network node
  - at buffer overflow or earlier
  - discarding policy - fairness, service differentiation
- Packet blocking at the network edge
  - packet waits in a queue outside the network

Group work:

- select solutions for streaming and for elastic flows
- find examples for the techniques above

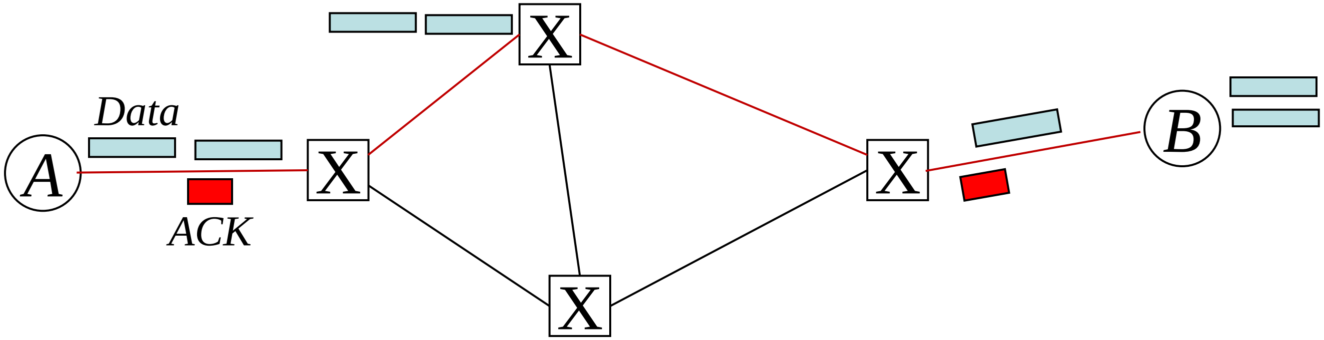
Congestion control techniques – examples

- Call blocking at the network edge + rate control
  - PSNT, mobile voice, IntServ (planned for Internet)
- Packet discarding at a network node
  - active queue management (e.g., Random Early Detection)
- Packet blocking at the network edge
  - window based congestion control
    (e.g., TCP, Transmission Control Protocol)

Congestion control techniques

- Should depend on the service requirements of the application
  - voice, video: minimum bandwidth requirement, delay sensitive
  - data: requires strict error control
- Call blocking at the network edge + rate control
  - calls blocked if resources are not available
  - the rate of accepted calls is controlled
- Packet discarding at a network node
  - buffer overflow or active queue management
  - discarding policy - fairness, service differentiation
- Packet blocking at the network edge
  - packet waits in a queue outside the network

End-to-end window based congestion control



- Basic idea:
- Window: Upper bound on the number of packets transmitted by A and not yet acknowledged by B
- Input rate reduced if acknowledgements arrive slowly – achieves congestion control

Fixed window based congestion control

- Congestion is indicated by increased round-trip time (RTT)
- Simple case:
  - ACK after each packet reception
  - Window size W is
    * decremented after each transmission
    * incremented if ACK arrives
  - Transmission, if W > 0
  - No lost Data packet or lost ACK
  - Parameters
    - constant maximum window size (W_m)
    - constant packet transmission time (X: packet size/link rate [sec])
    - round-trip time (rtt: propagation, **transmission** and queuing delays)
  - To calculate:
    maximum packet transmission rate (r: [packet/sec])

$$r = \min\left(\frac{1}{X}, \frac{W_m}{rtt}\right)$$

Fixed window based congestion control

- Maximum packet transmission rate (r: [packet/sec])
  - Parameters
    - constant maximum window size (W_m)
    - constant packet transmission time (X: packet size/link rate [sec])
    - round-trip delay (rtt: propagation, **transmission** and queuing delays)

$$r = \min\left(\frac{1}{X}, \frac{W_m}{rtt}\right)$$

- Rate inversely proportional to round-trip delay
- Congestion control: large queuing delays – large rtt – lower rate
  - Large W_m allows higher rate if there is no congestion
  - Reacts to congestion in W_m packets transmission time – large window means slow reaction to congestion
  - What rtt value means congestion? When should the congestion control "be activated"? ($rtt=W_mX$)?

Window control - packet delay

- End-to-end packet delay (T)
  - Delay: propagation, transmission and waiting at the queues in the multihop path
  - Queuing analysis in "steady state", for given network load
  - Model the network as a black box, apply Little theorem ($N = \lambda T$)
  - Assumption:
    - Constant arrival intensity (per node rate)
    - Infinite buffers, no loss

$$N = \sum_{i=1}^{n} const_i W_i$$

$$T = \frac{N}{\lambda} = \frac{\sum_{i=1}^{n} const_i W_i}{\lambda}$$

n: active flows

W_i : max window size for flow *i*

N: number of packets in the network

λ: aggregate throughput of all flows (no loss – infinite buffers!)

- For constant aggregate throughput (λ)
- Delay is proportional to the number of active sessions and to the window sizes
- Large W_m - large delay

Window control

- How to select the maximum window size?
  - – Small window: OK packet delay, but low throughput (congestion control starts with low rtt)
  - – Large window: OK throughput, but high packet delay and long reaction time

$$r = \min\left\{\frac{1}{X}, \frac{W}{rtt}\right\}$$

$$T = \frac{\sum_{i=1}^{n} const_i W_i}{\lambda}$$

- Dynamic maximum window sizes are necessary to follow the network load
  - — small window if the network is congested
  - — large window if the network is low loaded
  - — congestion is controlled by the window size – not by the *rtt*
  - — *But what indicates the congestion: packet drop, increased rtt, ...?*

Congestion control-dynamic window size

- How to select the window size?
  - Small window: OK packet delay, but low throughput
  - Large window: OK throughput, but high packet delay
- Dynamic window sizes are necessary to follow the network load
  - Congestion is controlled by the window size – not by the *rtt*
- TCP
  - Performance with dynamic window sizes - comes now
  - More realistic models for TCP performance – reading assignment

TCP congestion control

- Window based congestion control
- Dynamic window size:
  - decrease max window size if congestion is detected
  - congestion indicated by packet loss
  - increase max window size if current rate does not cause congestion (e.g., no loss)
- How to increase and decrease the window sizes?
  - Additive-increase, multiplicative decrease (AIMD)
  - Efficient and fair (if all users get the same immediate indication of congestion): Chiu and Jain, 1989
  - Probing - increase phase in each rtt: $w_{i+1} = w_i + b, 0 < b << w_{max}$
  - Congestion - decrease phase: $w_{i+1} = aw_i, 0 < a < 1$

- TCP: a=0.5, b=1

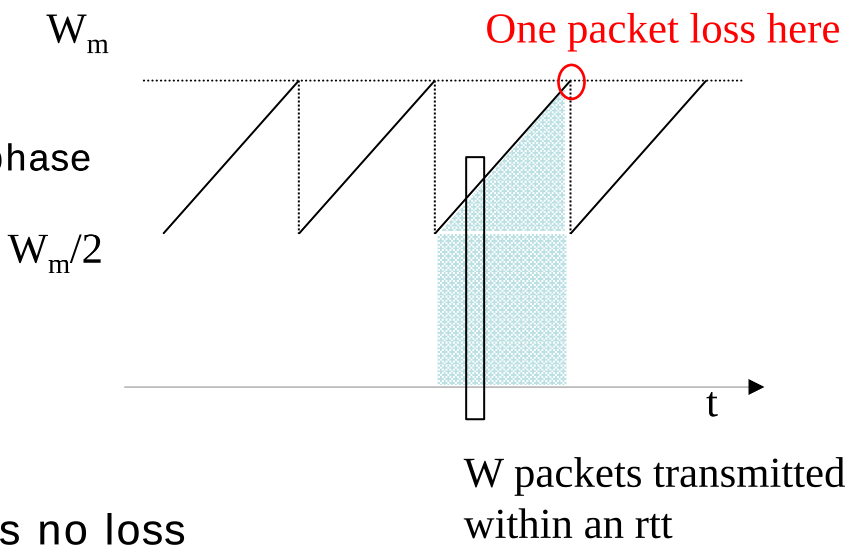
- TCP additional phases: slow start, fast recovery – not considered now

Analysis – AIMD model

- Congestion indicator: packet loss (full buffer on the path)
- Question: How does the throughput depend on the loss rate?

- Assumptions – for a very simplified case
  - saturated source (always has packet to send)
  - constant background traffic (all other traffic), loss at the same window size
  - loss due to congestion only
  - loss is the only congestion indicator
  - transients negligible (long flows)

  $\Rightarrow$ we model a static congestion avoidance phase
  - constant round trip time (rtt)
  - rtt >> transmission time
  - constant packet size L
  - low loss probability
    (to simplify calculations)



- W increased to W+1 after rtt if there was no loss
- W decreased to W/2 after rtt if there was a loss

AIMD throughput vs. packet loss

- *Th* as a function of *p* and *rtt*

$$Th = \frac{(N-1)L}{T_0}$$

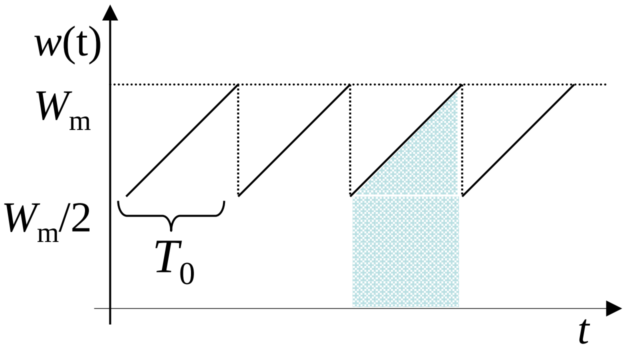
$$p = \frac{1}{N}, \quad N = \frac{1}{p}$$

$$T_0 = \frac{W_m}{2} rtt$$

$$N = \left(\frac{W_m}{2}\right)^2 + \frac{1}{2}\left(\frac{W_m}{2}\right)^2 = \frac{3}{8} W_m^2$$

$$W_m = \sqrt{\frac{8}{3}}\sqrt{N} = 2\sqrt{\frac{2}{3}}\frac{1}{\sqrt{p}}$$

$$Th = \frac{(N-1)L}{T_0} = \frac{\left(\frac{1}{p}-1\right)L}{\sqrt{\frac{2}{3}}\frac{1}{\sqrt{p}} rtt} = \sqrt{\frac{3}{2}}\frac{L}{rtt}\left(\frac{1-p}{p}\right)\sqrt{p} \approx \sqrt{\frac{3}{2}}\frac{L}{rtt}\frac{1}{\sqrt{p}}$$



rtt: round trip time (s)
p: packet loss probability
L: packet length (bit)

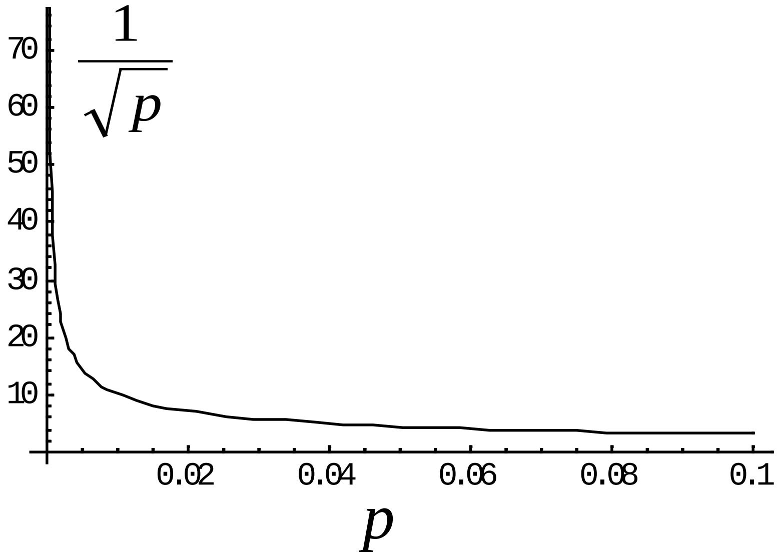
Th: throughput (bit/s)

W_m: max achieved window (packets)
T_0: cycle (s) – time between losses
N: packets transmitted in one cycle

TCP throughput vs. packet loss

- Statement: throughput in steady state
  - varies as the inverse-square-root of the loss rate, and
  - is inversely proportional to the round trip time (this we have seen for fixed window already!)

$$Th \approx \sqrt{\frac{3}{2}}\frac{L}{rtt}\frac{1}{\sqrt{p}}$$



Summary

- Congestion control – definition and possible solutions
  - End-to-end congestion control based on packet blocking at the network edge
    - Fixed window size schemes – throughput, delay conflict
    - Dynamic window congestion control
    - Simple AIMD scheme and throughput analysis – first steps towards evaluating TCP
- Home reading:
  - Bertsekas, Gallager, Data Networks, pp 500-505, 510.
  - J. Padhye, F. Firoiu, D. Towsley, J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," Sigcomm, 1998

Recall and outline for today

- Last lecture:
  - Congestion control – definition and possible solutions
  - End-to-end congestion control based on packet blocking at the network edge
  - Fixed window size schemes – throughput, delay conflict
  - Dynamic window congestion control
  - Simple AIMD scheme and throughput analysis – first steps towards evaluating TCP

- Todays lecture:
  - Intro for the TCP modeling home reading
  - TCP friendly rate control for delay sensitive traffic
  - Rate control for guaranteed service

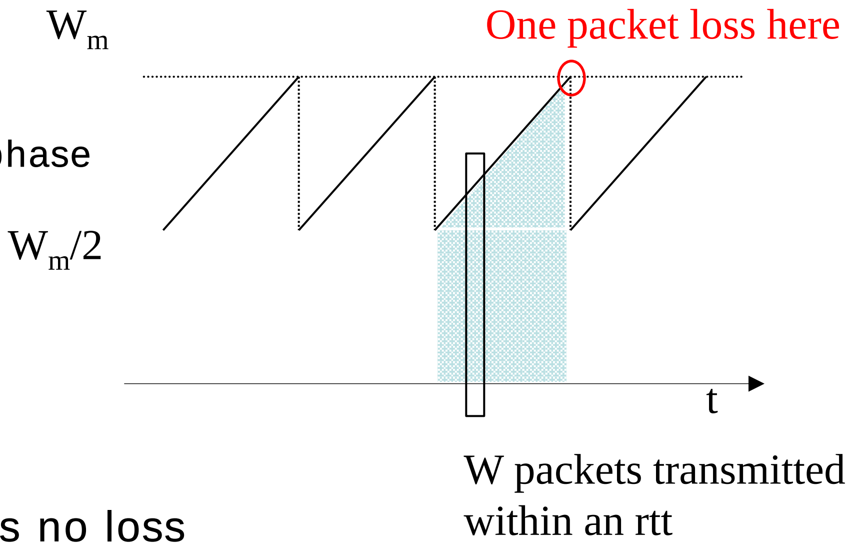
- Home reading: J. Padhye, F. Firoiu, D. Towsley, J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," Sigcomm, 1998

Analysis – AIMD model

- Congestion indicator: packet loss (full buffer on the path)
- Question: How does the throughput depend on the loss rate?

- Assumptions – for a very simplified case
  - saturated source (always has packet to send)
  - constant background traffic (all other traffic), loss at the same window size
  - loss due to congestion only
  - loss is the only congestion indicator
  - transients negligible (long flows)

  ⇒ we model a static congestion avoidance phase
  - constant round trip time (rtt)
  - rtt >> transmission time
  - constant packet size L
  - low loss probability
    (to simplify calculations)



- W increased to W+1 after rtt if there was no loss
- W decreased to W/2 after rtt if there was a loss

AIMD throughput vs. packet loss

- *Th* as a function of *p* and *rtt*

$$Th = \frac{(N-1)L}{T_0}$$

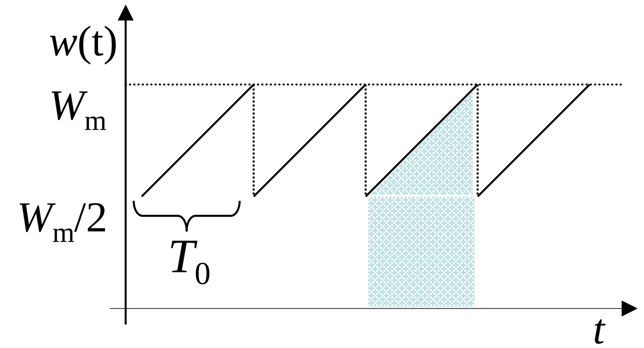
$$p = \frac{1}{N}, \quad N = \frac{1}{p}$$

$$T_0 = \frac{W_m}{2} rtt$$

$$N = \left(\frac{W_m}{2}\right)^2 + \frac{1}{2}\left(\frac{W_m}{2}\right)^2 = \frac{3}{8} W_m^2$$

$$W_m = \sqrt{\frac{8}{3}}\sqrt{N} = 2\sqrt{\frac{2}{3}}\frac{1}{\sqrt{p}}$$

$$Th = \frac{(N-1)L}{T_0} = \frac{\left(\frac{1}{p}-1\right)L}{\sqrt{\frac{2}{3}}\frac{1}{\sqrt{p}}rtt} = \sqrt{\frac{3}{2}}\frac{L}{rtt}\left(\frac{1-p}{p}\right)\sqrt{p} \approx \sqrt{\frac{3}{2}}\frac{L}{rtt}\frac{1}{\sqrt{p}}$$



rtt: round trip time (s)
p: packet loss probability
L: packet length (bit)

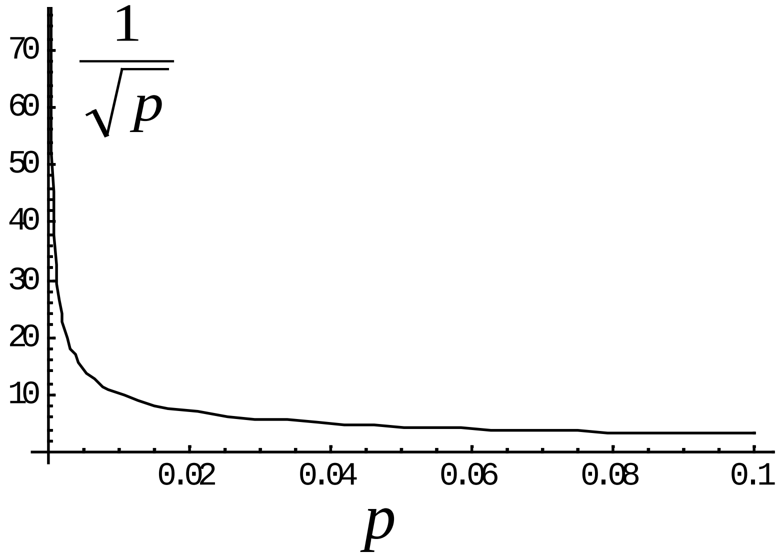
Th: throughput (bit/s)

W_m: max achieved window (packets)
T_0: cycle (s) – time between losses
N: packets transmitted in one cycle

TCP throughput vs. packet loss

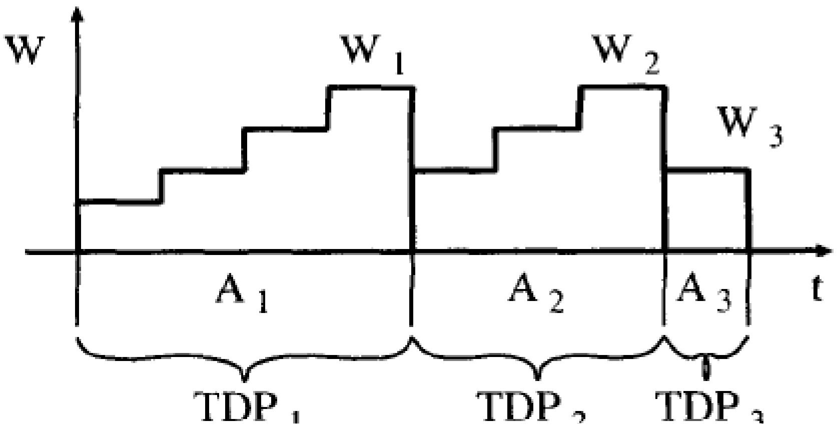
- Statement: throughput in steady state
  - varies as the inverse-square-root of the loss rate, and
  - is inversely proportional to the round trip time (this we have seen for fixed window already!)

$$Th \approx \sqrt{\frac{3}{2}}\frac{L}{rtt}\frac{1}{\sqrt{p}}$$

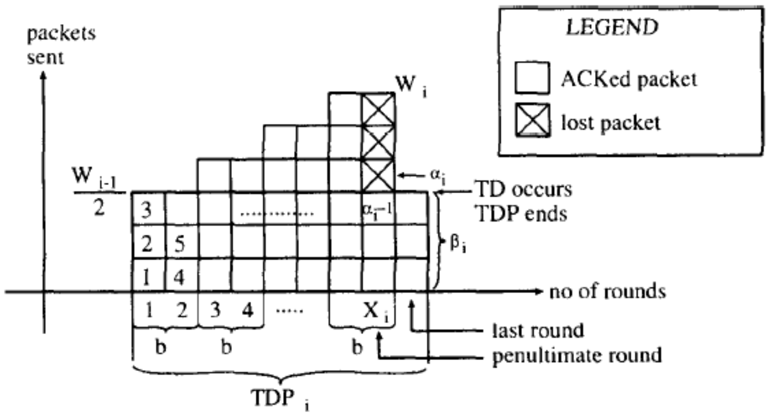


TCP congestion control model

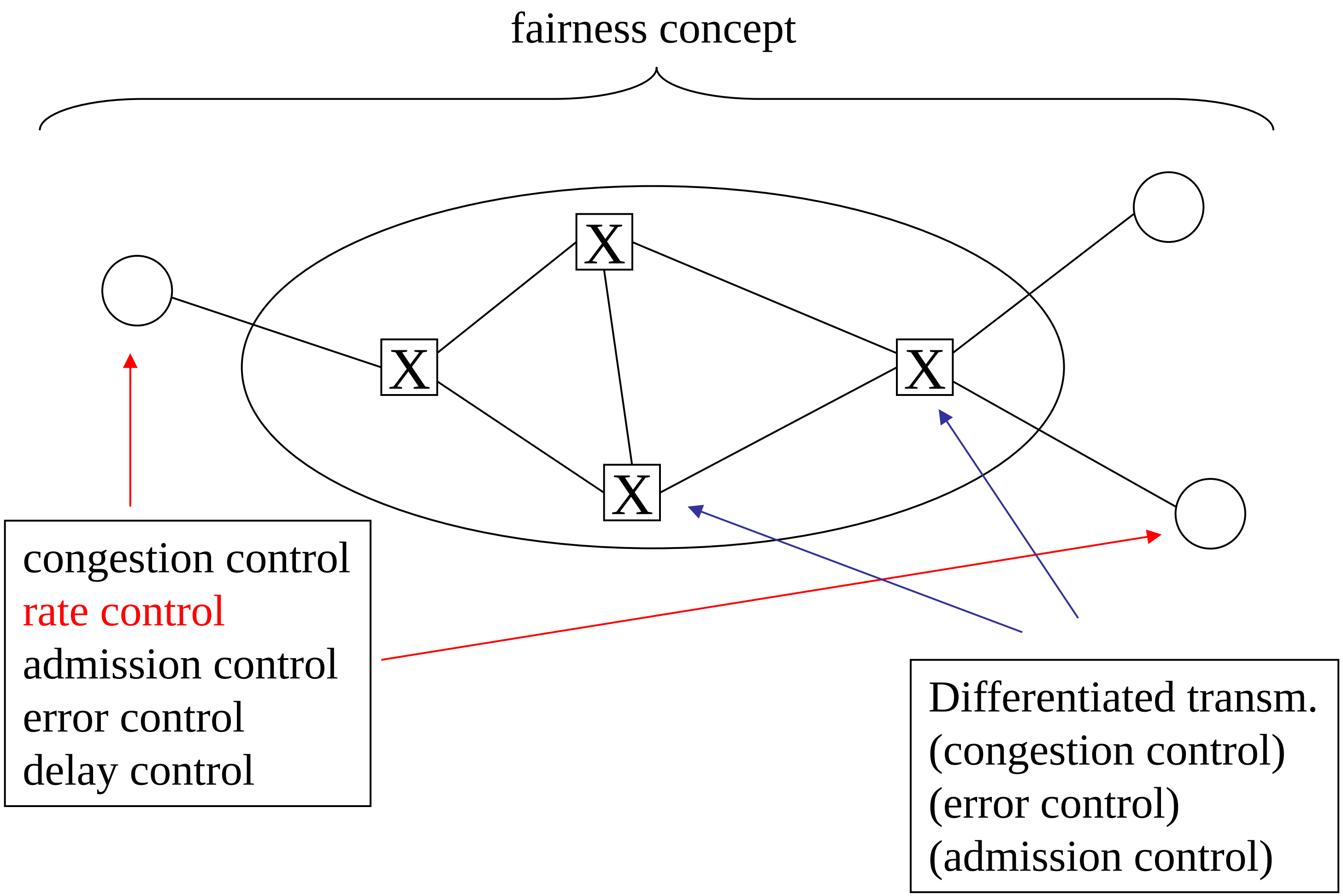
- Padhye, et al, “Modeling TCP Reno….”
  - Tripple-Duplicate ACKs
  - And Time-Outs
  - With limited max window size
  - Trace analysis
  - Discussion of the model



- Discuss the differences from and similarities to the model on the lecture
  - AIMD details
  - “First loss” process
  - Loss correlation
  - RTT



Control functions in communication networks



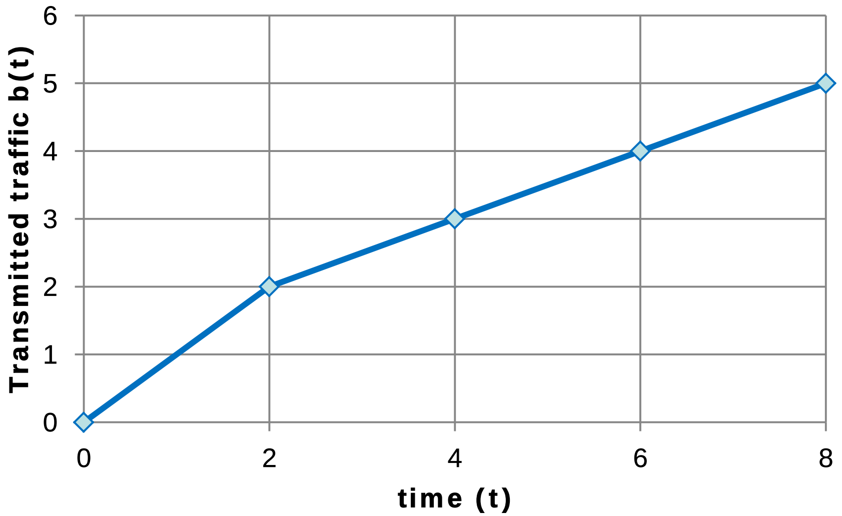
Rate control and guaranteed service

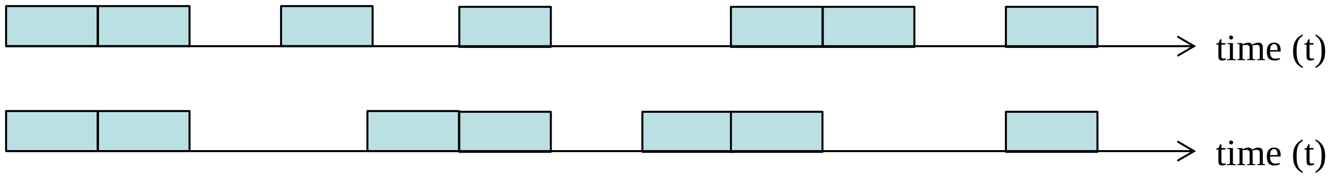
- Contract between the user and the network
  - If the user satisfies a traffic constraint,
  - Then the network provides delay, delay jitter (and loss) limits
- Rate control during the connection time
  - Peak rate (packets can not be closer in time than…)
  - Average rate (in some longer time interval)
  - Burstiness (e.g., max. number of packets transmitted without time gap)

- Window based control (r,T) – for average rate
  - Average rate r ensured in window T
    (n maximum number of packets per T: $n=rT$)
  - Jumping or sliding window versions
  - Problem:
    - Burstiness (packets transmitted back to back):
      jumping: $b=2n=2rT$, sliding: $b=n=rT$
    - Max burst size can not be controlled independently from r and T

Rate control and guaranteed service

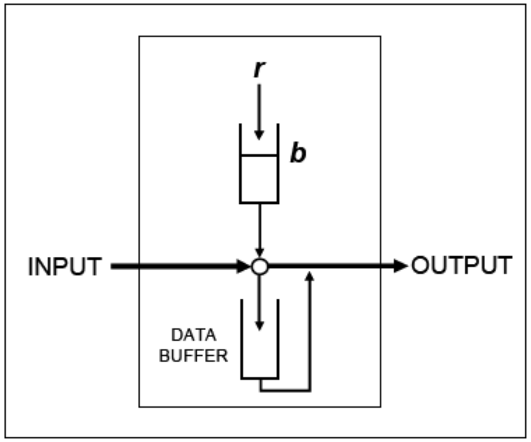
- Traffic envelope
  - max transmitted data within any time interval is given by a function b(t)
  - Are the traffic flows below accepted by the traffic envelope?





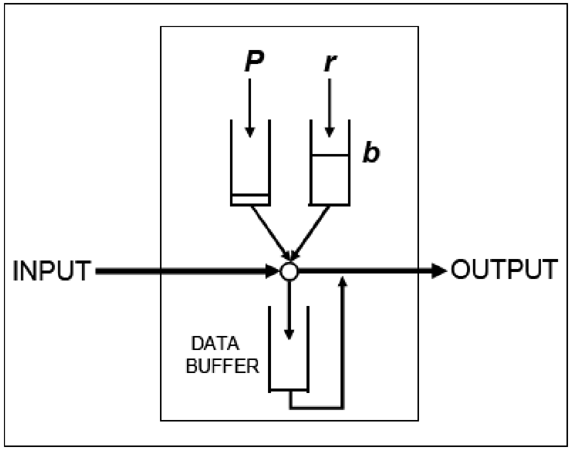
Rate control – Leaky-bucket

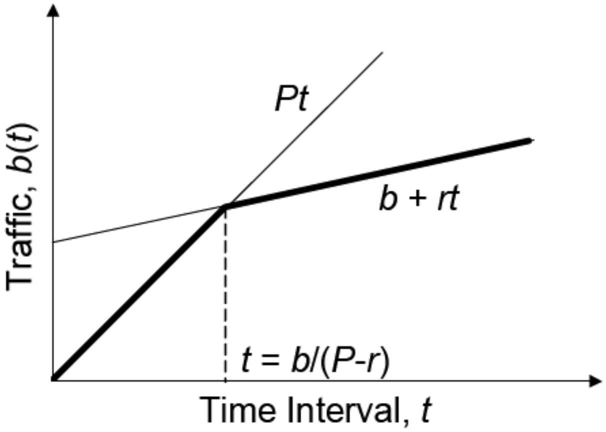
- Leaky-bucket scheme
  - Version 1: average rate and max burst size
    - Average rate for given time interval
    - Burst size: packets accepted at the same time
  - token buffer size (b)
  - token generation rate (r)
  - packets from "input" enter data buffer, waiting for service
  - packet transmitted to "output" if there is a token in the token buffer
  - token removed at packet transmission
  - Result:
    - traffic envelope: b(t)=b+rt
    - average rate: (b+rt)/t
  - Note: from "output" the packets enter a transmission buffer (MAC)



Rate control – Leaky-bucket

- Leaky-bucket scheme
  - Version 2: peak rate, average rate, burstiness
  - average rate buffer, size b
  - average token generation rate (r)
  - peak rate buffer size 1
  - peak token generation rate (P>r)
  - packets from "input" enter data buffer, waiting for service
  - packet transmitted to "output" if there is at least one token in both token buffers
  - tokens removed at packet transmission
  - Result:
    - traffic envelope b(t)





Summary

- Congestion control
  - to limit the traffic in the network (avoid loss or large delays)
  - window based congestion control
  - AIMD schemes and TCP

- Rate control
  - to control the average rate, burstiness or peak rate of traffic injected to the network
  - with the goal of providing service guarantees
  - Solutions
    - window based – problem with rate and burstiness coupling
    - traffic envelope based – leaky bucket

- At home
  - Check the matlab emulator for the Leaky Bucket!