

## TCP Congestion Control

$SWS = \min(B \times D, \text{"round"})$   
 dynamic  $\lambda = \frac{SWS}{RTT}$   $\leftarrow$  flow control  
 efficiency RTT?  
 estimate BxD  $\Rightarrow$  congestion  
 overestimate  $\Rightarrow$  underutilization  
 underestimate  $\Rightarrow$  underutilization  
 "cwnd"  
 Thing input (output)  
 Goodput (rate of useful data)  
 "cliff"  
 congestion collapse  
 exponential times  
 "M/M/1"  
 average pkt delay  
 $= \frac{1}{C - \lambda}$   
 $\lambda$  (incl. transmission)  
 feedback system  
 "open loop" C  
 "closed loop" C  
 source (TCP) transmission  
 "duplication" retransmission  
 $\rho = \frac{\lambda}{C} < 1$   
 "traffic intensity"  
 saturation  
 $\frac{1}{2}$

Matta @ BUCS - Transport 1-58

58

## TCP Congestion Control

$\text{cwnd} \uparrow$  if no congestion  
 $\text{cwnd} \downarrow$  if congestion  
 $\frac{\text{RTT}}{\text{RTT}} \Rightarrow$  lost packet  
 $\Rightarrow$  buffer overflow  
 $\Rightarrow$  congestion  
 - AI:  $\text{cwnd} \leftarrow \text{cwnd} + 1$  every RTT  
 MD:  $\text{cwnd} \leftarrow \frac{\text{cwnd}}{2}$  upon loss  
 AI:  $\text{cwnd} \leftarrow \text{cwnd} + \frac{1}{\text{cwnd}}$  every ACK  
 packets  
 bytes  $\rightarrow \text{cwnd} \leftarrow \text{cwnd} + \frac{\text{MSS}}{\left(\frac{\text{cwnd}}{\text{MSS}}\right)}$  every ACK

Matta @ BUCS - Transport 1-59

59

## TCP Congestion Control

### Additive Increase/Multiplicative Decrease

- ❑ Objective: adjust to changes in the available capacity
- ❑ New state variable per connection: **CongestionWindow**
  - limits how much data source has in transit

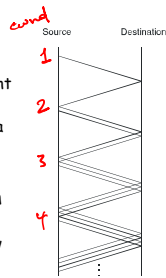
**MaxWin = MIN(CongestionWindow, AdvertisedWindow)**

- ❑ **Idea:**
  - increase **CongestionWindow** when congestion goes down
  - decrease **CongestionWindow** when congestion goes up
- ❑ **Question:** how does the source determine whether or not the network is congested?
- ❑ **Answer:** a timeout occurs
  - timeout signals that a segment was lost
  - segments are seldom lost due to transmission error
  - lost segment implies congestion

Matta @ BUCS - Transport 1-60

## AIMD

- Algorithm:
  - increment **CongestionWindow** by one segment per RTT (*linear increase*)
  - divide **CongestionWindow** by two whenever a timeout occurs (*multiplicative decrease*)
- In practice: increment a little for each ACK
  - Increment =  $(MSS \times MSS) / \text{CongestionWindow}$
  - CongestionWindow += Increment

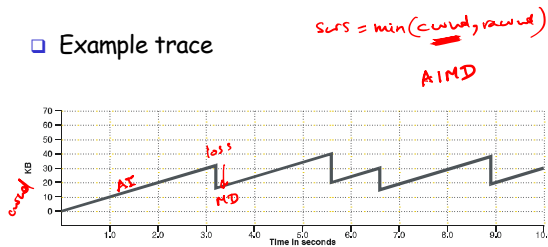


Matta @ BUCS - Transport 1-61

61

## Sawtooth behavior

- Example trace



Matta @ BUCS - Transport 1-62

62

## Slow Start

SS:  $cwnd \leftarrow cwnd + cwnd$  every RTT  
 (exponential increase)  
 $cwnd \leftarrow cwnd + 1$  every ACK

Matta @ BUCS - Transport 1-63

63

## Slow Start

- Objective: determine the available capacity in the first place
  - when first starting connection
  - when connection recovers after a timeout
- Idea:
  - begin with **CongestionWindow** = 1 segment
  - double **CongestionWindow** each RTT (increment by 1 segment for each ACK)

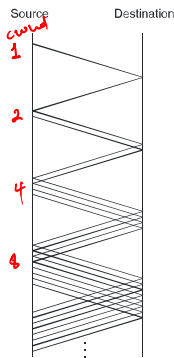
Matta @ BUCS - Transport 1-64

64

## Slow Start (cont'd)

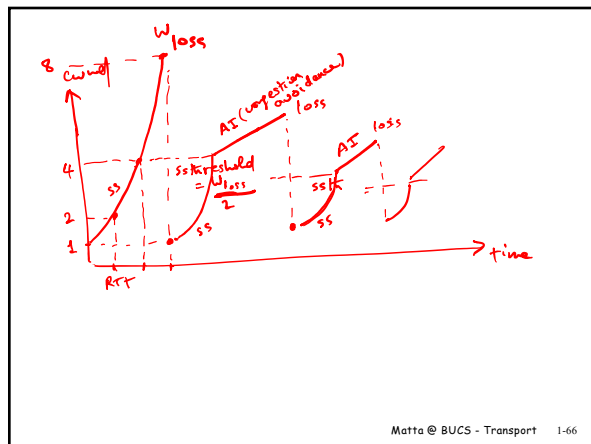
- Exponential growth, but slower than all in one blast

To reach a window  $W$   
 $\log_2 W$  RTTs



Matta @ BUCS - Transport 1-65

65



Matta @ BUCS - Transport 1-66

66

## TCP Congestion Algorithm

On a timeout, half the current window size is recorded in **ssthresh**

```

if (cwnd < ssthresh) // if we're still doing
    // slow-start, open window exponentially
    cwnd += 1
else // otherwise do Congestion Avoidance
    // linear increase
    cwnd += 1/cwnd
    
```

Matta @ BUCS - Transport 1-67

67

---

---

---

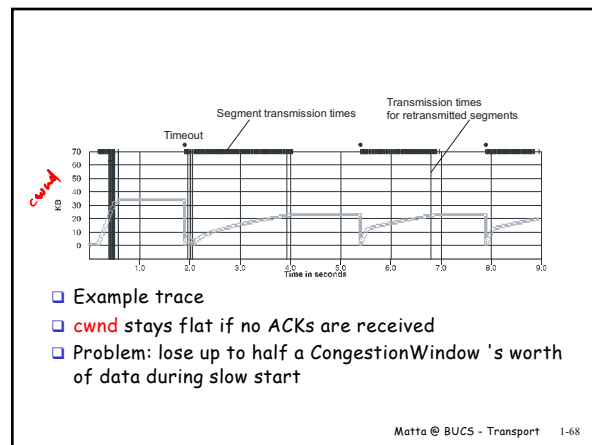
---

---

---

---

---



Matta @ BUCS - Transport 1-68

68

---

---

---

---

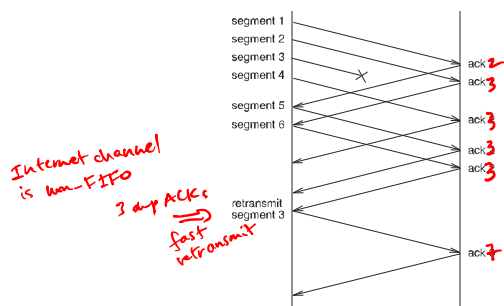
---

---

---

---

## Fast Retransmit and Fast Recovery



Matta @ BUCS - Transport 1-69

69

---

---

---

---

---

---

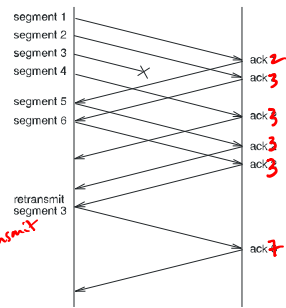
---

---

## Fast Retransmit and Fast Recovery

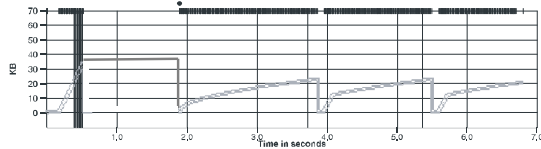
- Problem: coarse-grain TCP timeouts lead to idle periods
- Fast retransmit: use duplicate ACKs to trigger retransmission

*Internet channel is non-FIFO  
3 dup ACKs ⇒ fast retransmit*



Matta @ BUCS - Transport 1-70

70



- Long periods during which **cwnd** stays flat are eliminated

Matta @ BUCS - Transport 1-71

71