Rechnernetze - Computer Networks

Lecture 11: End-to-end Protocols
TCP Congestion Control

Prof. Dr.-Ing. Markus Fidler

Institute of Communications Technology
Leibniz Universität Hannover

June 30, 2024

Outline

Congestion control overview

TCP congestion control

Fairness

Throughput models

Congestion control

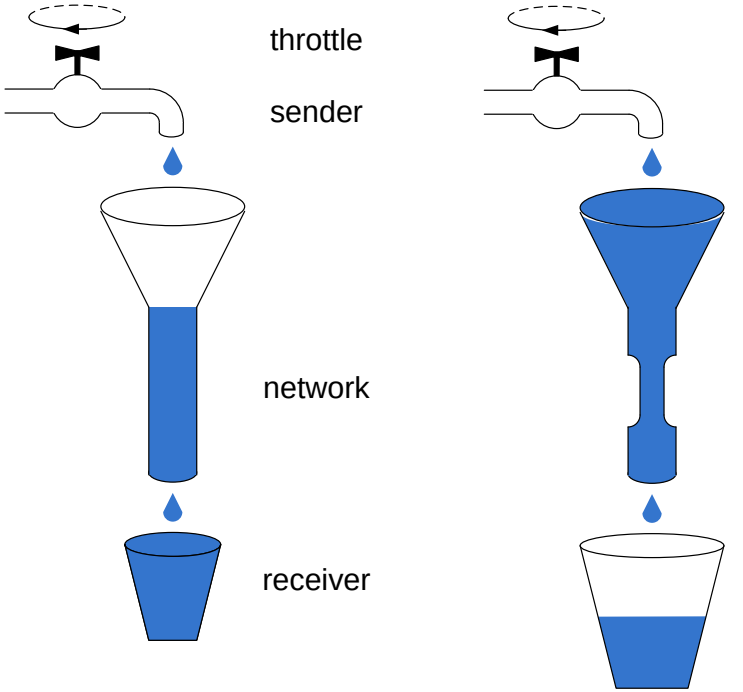
Packet loss in the network is mostly due to buffer overflow if the network is congested (too many sources sending at too high rate)

- ARQ can fix the symptoms of congestion but
- ARQ cannot fix the cause of congestion

Congestion control throttles senders to adjust the data rate to

- bottleneck bandwidth
- variations in bandwidth
- share bandwidth fairly

Congestion control vs. flow control



Throttle sender to

- avoid flooding the receiver: flow control
- avoid flooding the network: congestion control

Approaches to congestion control

End-to-end congestion control

- no explicit support from the network
- end systems must infer congestion from packet delays and loss
- RFC 2581 TCP congestion control

Network-assisted congestion control

- routers provide explicit feedback about their congestion state
  - direct feedback: congested routers send choke packets to the senders
  - indirect feedback: congested routers mark a field in packet flowing from sender to receiver; the receiver then notifies the sender, e.g. using a field in the acknowledgement to indicate congestion
- RFC 3168 adds explicit congestion notification (ECN) to IP

Overview of TCP congestion control

TCP congestion control is window-based

- TCP senders have at most w segments in the pipe
  i.e. w transmitted but not yet acknowledged segments
- TCP does not compute a target data rate explicitly

TCP adapts the window size dynamically

- it starts cautiously with a small w
- whenever segments are acknowledged it cautiously increases w
- if it has to retransmit a segment it drastically reduces w
  (retransmissions indicate segment loss due to congestion)

Congestion window

TCP senders comply with two windows that control the send rate

- receive window: used for flow control
  - maintained and advertised by the receiver
- congestion window: used for congestion control
  - maintained by the sender

A TCP sender may send as long as

$$\text{last byte sent} - \text{last byte acked} \leq \min(\text{congestion window}, \text{receive window})$$

Here, assume that receive window $>$ congestion window.

Bandwidth-delay-product

Recall how go-back-N improves stop-and-wait ARQ.

Given

- a window size of w segments
- a maximum segment size MSS
- a pipe with capacity C
- round trip time $RTT = T_t + 2T_p$

TCP can transmit $w \cdot MSS$ bytes per RTT, i.e. the achievable throughput is

$$T_{put} = \frac{w \cdot MSS}{RTT}$$

In order to fill the pipe with capacity C, i.e., $T_{put} = C$, a congestion window of $C \cdot RTT = w \cdot MSS$ bytes is needed.

$C \cdot RTT$ is referred to as the bandwidth delay product of the pipe. Smaller windows can be used to throttle the sender.

Window size adaptation

Selecting an appropriate TCP congestion window size w is crucial. Consider a greedy TCP sender

- if $w \cdot MSS < C \cdot RTT$ the pipe cannot be fully utilized and resources are wasted
- otherwise $w \cdot MSS = C \cdot RTT$ and the pipe is fully utilized
  - there exists a minimal $w_{\min}$ which achieves full utilization
  - increasing w beyond $w_{\min}$ results in buffering and possibly segment loss at routers
    queuing delays add to the RTT such that $w \cdot MSS = C \cdot RTT$
- the case $w \cdot MSS > C \cdot RTT$ cannot occur

- TCP does, however, not know the minimal $w_{\min}$ in advance; moreover, $w_{\min}$ may be highly variable over time
- TCP increases the window size cautiously;
  if loss occurs it reduces w drastically

Slow start

TCP's procedure for adapting the congestion window consists of two phases

- slow start
- congestion avoidance

Phase 1: slow start

- initially the congestion window is set to 1 MSS
- for each segment that is acknowledged 1 MSS is added

During slow start the congestion window is increased exponentially

- the first segment is acked after RTT
  the congestion window is set to 2 MSS
- two segments are sent and acked after another RTT
  the congestion window is set to 4 MSS
- the congestion window is doubled each RTT

Congestion avoidance

Once the window exceeds a threshold TCP enters congestion avoidance.

Phase 2: congestion avoidance

- the congestion window size w is increased by one MSS after reception of w acknowledgements
- if a timeout occurs a segment is retransmitted and
  - the threshold value is set to half the congestion window size
  - slow start is invoked ($w = 1$ MSS)

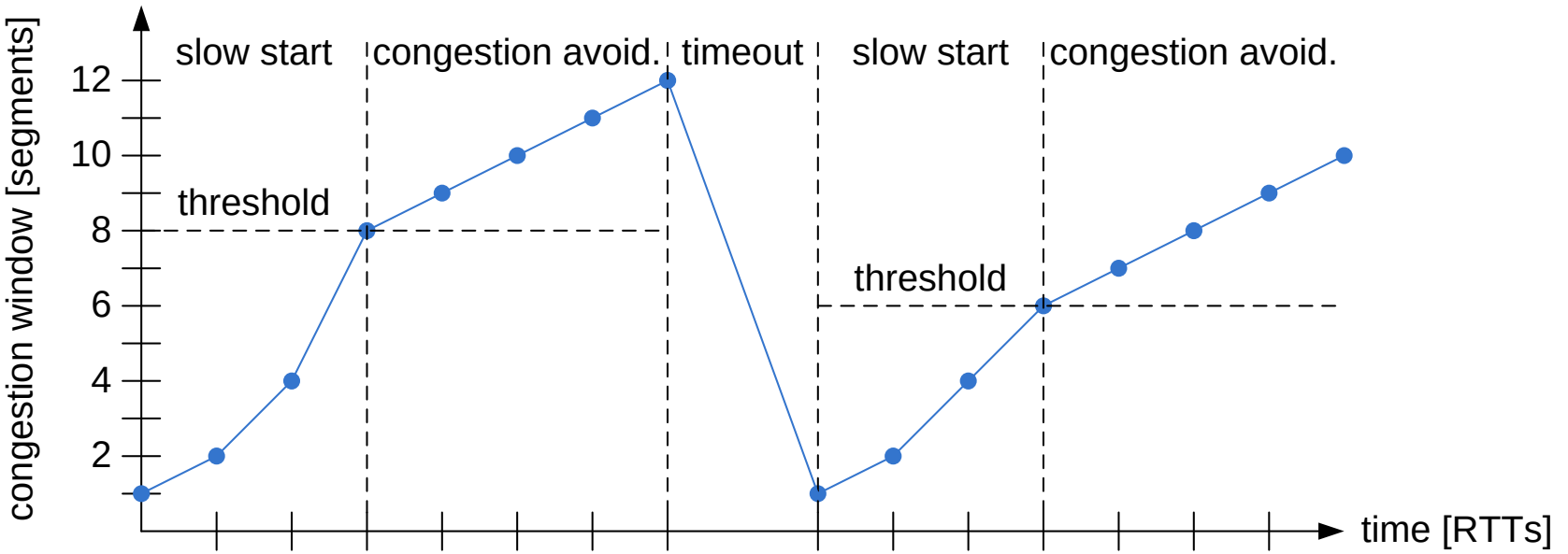
During congestion avoidance the window is increased additively

- the congestion window is increased by one MSS each RTT

The procedure is referred to as additive increase multiplicative decrease (AIMD).

Saw-tooth characteristic

- below threshold the congestion window grows exponentially
- above threshold the congestion window grows linearly
- in case of a timeout
  - the threshold is set to one-half of the congestion window
  - the congestion window is set to one maximum segment size



TCP versions

- **Tahoe**: the version just described
  - if a segment is lost long waiting times until the timeout occurs
- **Reno**: basic version widely used today
  - adds fast retransmit in case of three duplicate acknowledgements
  - in case of a fast retransmit the slow start phase is canceled, called fast recovery ($w =$ threshold)
- **Vegas**: uses a congestion avoidance algorithm
  - congestion is inferred before segments get lost
  - increasing RTTs are used as an early indicator for upcoming congestion
- **SACK**: adds selective acknowledgements
- many more

Outline

Congestion control overview

TCP congestion control

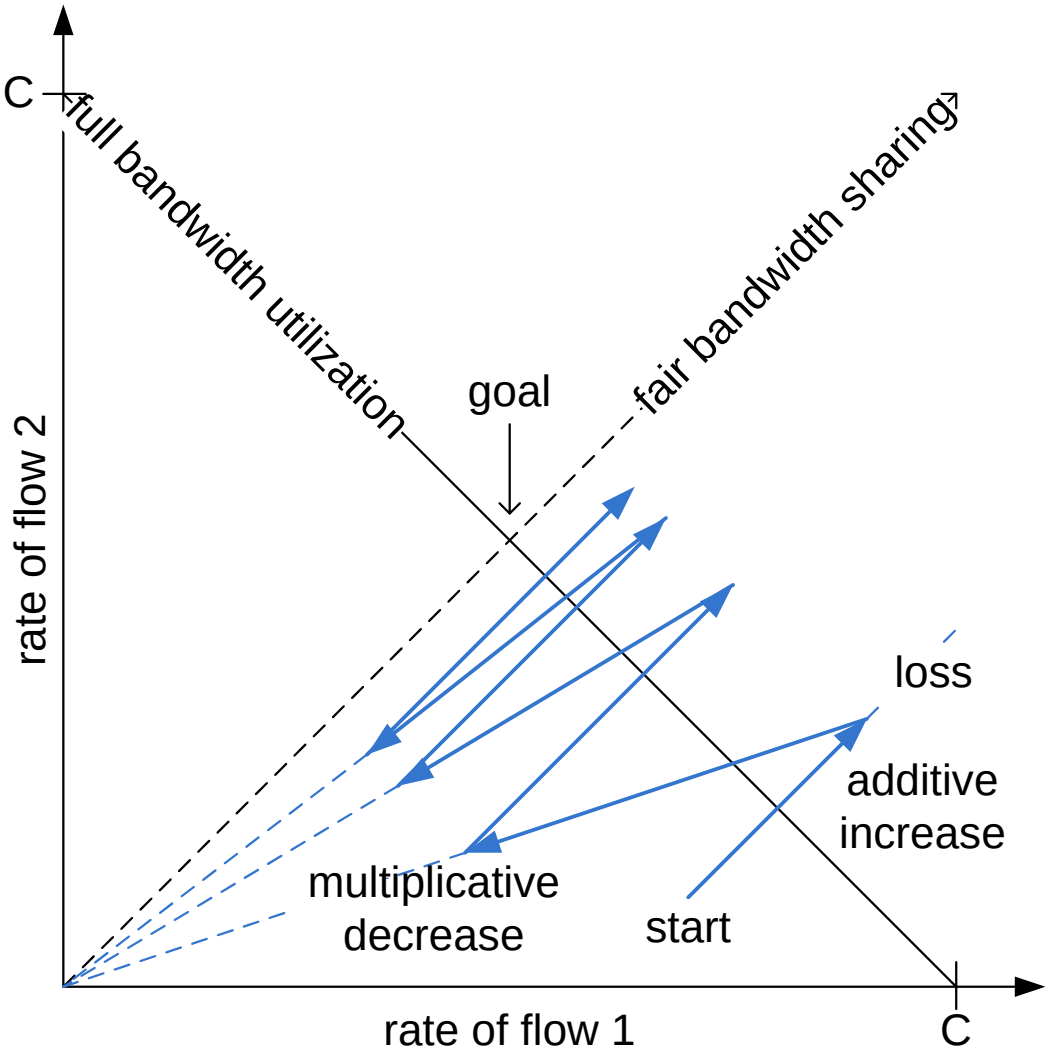
Fairness

Throughput models

Fairness of AIMD

AIMD

- converges iteratively
- achieves fairness
- and high utilization



A simple macroscopic model

Given the saw-tooth characteristics of the TCP congestion window what is the throughput of a TCP connection?

For simplicity neglect the initial slow start phase and assume fast retransmit and fast recovery afterwards.

Denote W the window size at which loss occurs. The TCP throughput during congestion avoidance thus ranges from

$$\frac{0.5 \cdot W \cdot MSS}{RTT} \cdots \frac{W \cdot MSS}{RTT}$$

Since the window size is increased linearly we have

$$\text{average throughput} = \frac{0.75 \cdot W \cdot MSS}{RTT}$$

A refined macroscopic model

Now assume the congestion window size at which loss occurs W is not known but the probability that a segment is lost p_{loss} is given.

During congestion avoidance it takes $W/2$ $RTTs$ until the window reaches W and a loss occurs. Thus

$$\frac{3W \cdot MSS}{4RTT} \cdot \frac{W \cdot RTT}{2} = \frac{3}{8} W^2 MSS$$

bytes are transmitted until a segment is lost respectively

$$\frac{3}{8} W^2$$

segments are transmitted until a segment is lost.

A refined macroscopic model continued

Given segment losses occur independently with probability p_{loss}. The probability that segments $1 \ldots k-1$ are not lost and k is lost is geometrically distributed, i.e.

$$\mathsf{P}[K = k] = (1 - p_{\text{loss}})^{k-1} p_{\text{loss}}$$

The average number of segments transmitted until the first loss occurs is given by the expected value (see next slide)

$$\mathsf{E}[K] = \sum_{k=1}^{\infty} k \cdot \mathsf{P}[K = k] = \frac{1}{p_{\text{loss}}}$$

Equating the two results

$$\frac{3}{8} W^2 = \frac{1}{p_{\text{loss}}} \Rightarrow W = \sqrt{\frac{8}{3 p_{\text{loss}}}}$$

Expected value of a geometric random variable

Need to solve

$$\mathsf{E}[K] = \sum_{k=1}^{\infty} k \cdot \mathsf{P}[K = k] = \sum_{k=0}^{\infty} k \cdot (1 - p_{\text{loss}})^{k-1} p_{\text{loss}}$$

Geometric sum

$$\sum_{k=0}^{\infty} q^k = \frac{1}{1-q} \text{ if } q < 1$$

Rewriting with $q = 1 - p_{\text{loss}}$

$$\mathsf{E}[K] = p_{\text{loss}} \sum_{k=0}^{\infty} k q^{k-1} = p_{\text{loss}} \sum_{k=0}^{\infty} \frac{\partial}{\partial q} q^k = p_{\text{loss}} \frac{\partial}{\partial q} \sum_{k=0}^{\infty} q^k$$

allows using the geometric sum such that

$$\mathsf{E}[K] = p_{\text{loss}} \frac{\partial}{\partial q} (1-q)^{-1} = p_{\text{loss}} \frac{(-1) \cdot (-1)}{(1-q)^2} = \frac{1}{p_{\text{loss}}}$$

A refined macroscopic model continued

The average throughput follows from

$$\text{average throughput} = \frac{3W \cdot MSS}{4RTT}$$

by insertion of $W = \sqrt{\frac{8}{3 \cdot p_{\text{loss}}}}$ as

$$\text{average throughput} = \frac{MSS}{RTT \cdot \sqrt{\frac{2}{3} p_{\text{loss}}}}$$

Considering short message transfer additional delays occur due to slow start etc. that are not considered here.

Note that the dependence on RTT can cause significant unfairness in case flows have paths with different RTTs.