FLOW CONTROL IN TCP

W. Stallings, High-Speed Networks, TCP/IP and ATM Design Principles, Prentice-Hall, 1998, Sections 10.1-10.2

- Based on window mechanism
- Aims at sharing the bandwidth fairly between the users
- Timeout and retransmission
  - measurement of the round trip time (RTT)
  - estimating variance of RTT (Jacobson's algorithm)
  - exponential backoff of RTO
  - Karn's algorithm
- Slow start
- Dynamic window control
- Fast retransmit
- Fast recovery
- Selective acknowledgement, SACK (an optional addition)

Implementation of the mechanisms in different versions of TCP

Mechanism	RFC 1122	TCP Tahoe	TCP Reno	NewReno
Estimation of variance of RTT	✓	✓	✓	✓
Exponential backoff of RTO	✓	✓	✓	✓
Karn's algorithm	✓	✓	✓	✓
Slow start	✓	✓	✓	✓
Dynamic window control	✓	✓	✓	✓
Fast retransmit		✓	✓	✓
Fast recovery			✓	✓

- NewReno (RFC 2582, April 1999) is currently the most popular version of TCP
- Includes an improved fast transmit mechanism for the case of multiple packet drops

Timeout and retransmission

- For each sent segment a retransmission time-out RTO counter is set up
- The purpose of the counter is to differentiate the cases where
  - acknowledgment is delayed to due random delay fluctuations
  - network is congested and the sent segment has been lost

$$\mathrm{RTO} = \mathrm{SRTT} + f \times \mathrm{SDEV}$$

$$\begin{cases} \mathrm{RTO} & = \text{ retransmission timeout} \\ \mathrm{SRTT} & = \text{ smoothed roundtrip estimate} \\ \mathrm{SDEV} & = \text{ smoothed roundtrip standard deviation estimate} \end{cases}$$

  - for the coefficient f the value $f = 4$ is often used (original recommendation $f = 2$)

Estimation of the round trip time and its variation (Jacobson's algorithm)

- For each sent segment, the time RTT to the arrival of the acknowledgment is measured
- The difference of the measured round trip time and the current smoothed estimate SRTT is calculated

$$\text{SERR} = \text{RTT} - \text{SRTT}$$

- The smoothed estimate SRTT is updated (exponential averaging)

$$\text{SRTT} \leftarrow (1-g) \times \text{SRTT} + g \times \text{RTT}$$

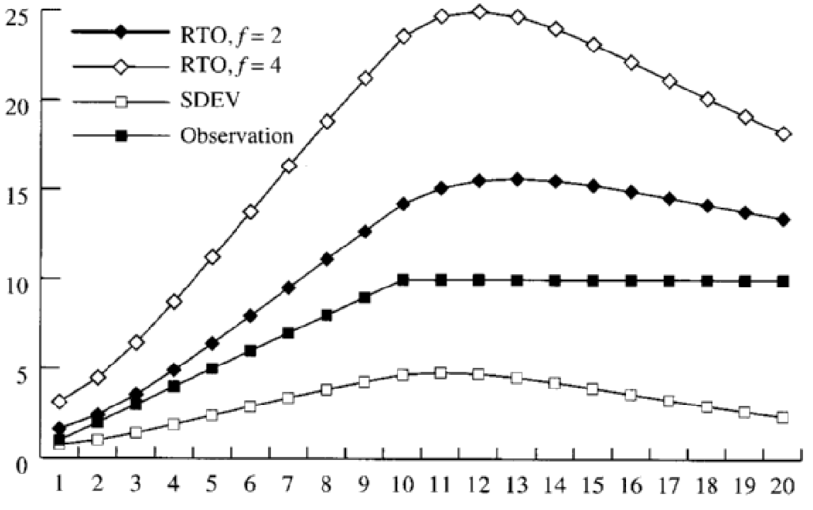
- Similarly, the estimate for the delay variance SDEV is updated

$$\text{SDEV} \leftarrow (1-h) \times \text{SDEV} + h \times |\text{SERR}|$$

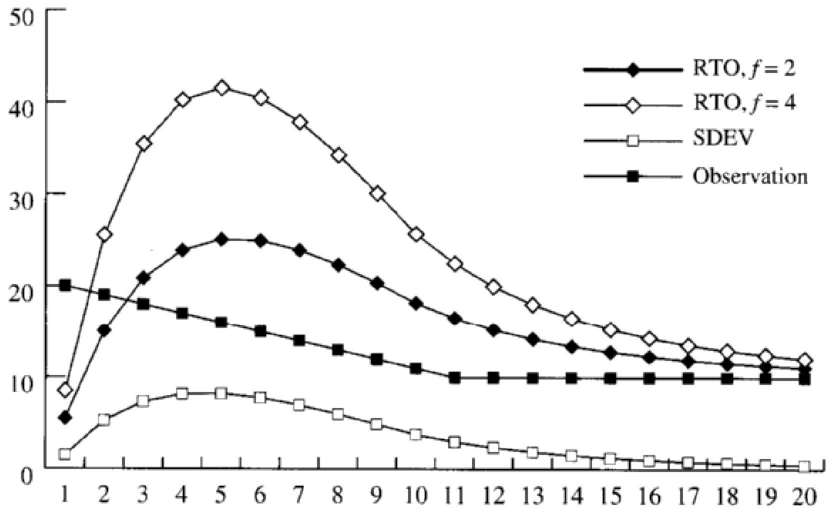
  – the recommended values for the coefficients g and h are

$$\begin{cases} g \;=\; 1/8 \;= 0.125 & \text{'average of the last 8 measurements'} \\ h \;=\; 1/4 \;= 0.25 & \text{'average of the last 4 measurements'} \end{cases}$$

Example of the evolution of the estimates[1]



(a) Increasing function



(b) Decreasing function

[1]from W. Stallings *High Speed Networks: TCP/IP and ATM Design Principles*, Prentice Hall, 1997.

Exponential RTO backoff of

- When the timer RTO expires the segment is retransmitted
  - indication that the network is congested
  - using the same RTO for the new segment would not make sense as it could easily lead to new expiration of the timer
  - RTO is increased by a factor q

$$\text{RTO} \leftarrow q \times \text{RTO}$$

- If the timer still expires, one continues in the same way (RTO grows exponentially up to some set limit (e.g. 64 s)
- Often the value $q = 2$ is used
  - binary exponential backoff
  - as in the CSMA/CD protocol of the Ethernet
- Retransmissions are tried up to a given limit (e.g. 9 min)

Karn's algorithm

- When finally an acknowledgment is received after one or several retransmissions, one cannot know for sure whether it is the acknowledgment of the original segment or one of the retransmitted segments
- One cannot make a reliable measurement of RTT
- Karn's algorithm defines the procedure in such a case:
    1. SRTT and SDEV are not updated
    2. In the case of a retransmission, RTO is increased by factor q
    3. For the following segments the same value of RTO is used
    4. First, when a acknowledgment is received for a non-retransmitted segment, the normal procedure is resumed

Slow start

- The size of the allowed transmission window *awnd* is defined in segments (not in octets)
- It is governed by two factors: $awnd = \min(credit,\ cwnd)$

$awnd$	=	allowed transmission window in segments
$cwnd$	=	congestion window of the TCP flow control
$credit$	=	the receivers advertised allowed transmission window; computed from the value of the *window* field in a TCP segment coming from the receiver transformed into segments: $credit = window\ /\ segment\ size$

- In the setup of a TCP connection one sets $cwnd = 1$
- For each received acknowledgment *cwnd* is incremented by one up to a given maximum

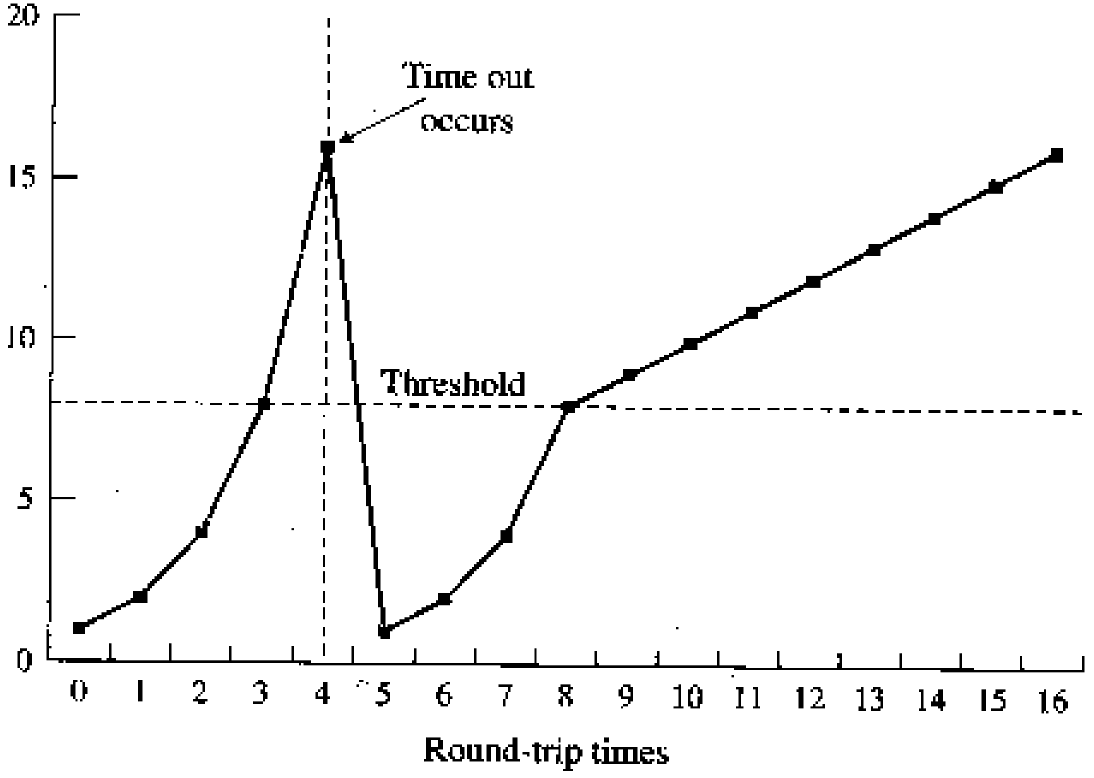
$$cwnd \leftarrow cwnd + 1$$

- In fact, the start is not slow at all but <u>exponential</u>
  - for each acknowledged full window of segments, the size of the window is doubled
  - in the case of a 'greedy' source, the size is doubled every RTT

Dynamic window control in the case of congestion

- The expiration of the RTO timeout counter is an indication of the congestion
  - the segment has been either lost or badly delayed
- Then one goes back to the initial state of the slow start
  - set $cwnd = 1$
  - increment the window by one for each received acknowledgment
- The exponential growth of the window may be too aggressive in a congested network
- A more cautious, congestion avoidance method is as follows:
  1. Set threshold $ssthresh$ to one half of the current congestion window $ssthresh = cwnd/2$
  2. Set $cwnd = 1$ and continue according to the slow start procedure until $cwnd = ssthresh$
  3. From that point on (when $cwnd \geq ssthresh$) $cwnd$ is incremented by one for each full round trip time. This can be achieved for instance by updating $cwnd$ for each acknowledgment as follows: $cwnd \leftarrow cwnd + 1/cwnd$.

‘Slow start’ and dynamic window control[2]



[2]from W. Stallings *High Speed Networks: TCP/IP and ATM Design Principles*, Prentice Hall, 1997.

Fast retransmission

- In practice, the retransmission timeout value RTO is much greater then the real RTT
- This is well justified since for many reasons a reliable measurement of the round trip time is difficult
- This, however, means that retransmissions based on the expiration of the timer may be slow
- Fast retransmissions exploit the fact that the TCP always acknowledges the last segment successfully received in correct order
- Repeated acknowledgments of the same segment indicate a missing, possibly lost segment
- To exclude the possibility that the segment is just delayed for some reason one waits until three acknowledgments are received for the same segment
- In fast retransmission, this is interpreted to signify a real loss of the segment and the segment is retransmitted

Fast recovery

- Also in the case of fast retransmission, the size of the congestion window has to be reduced in order to alleviate the congestion
- The received acknowledgments, however, indicate that some traffic is going through the network, and the congestion may not be as severe as in the case of expiration of the timeout counter
- Therefore, the window is not reduced to 1, but in the 'fast recovery' the procedure is the following (upon receipt of the third acknowledgment of the same segment):
  1. Set $ssthresh = cwnd/2$
  2. Retransmit the missing segment
  3. Set $cwnd = ssthresh + 3$ (as three segments have been received)
  4. If still more acknowledgments are received for the same segment, each time $cwnd$ is incremented by one
  5. When finally a non-retransmitted segment is acknowledged (which at the same time acknowledges all missing and later segments), one sets $cwnd = ssthresh$

Selective acknowledgement, SACK

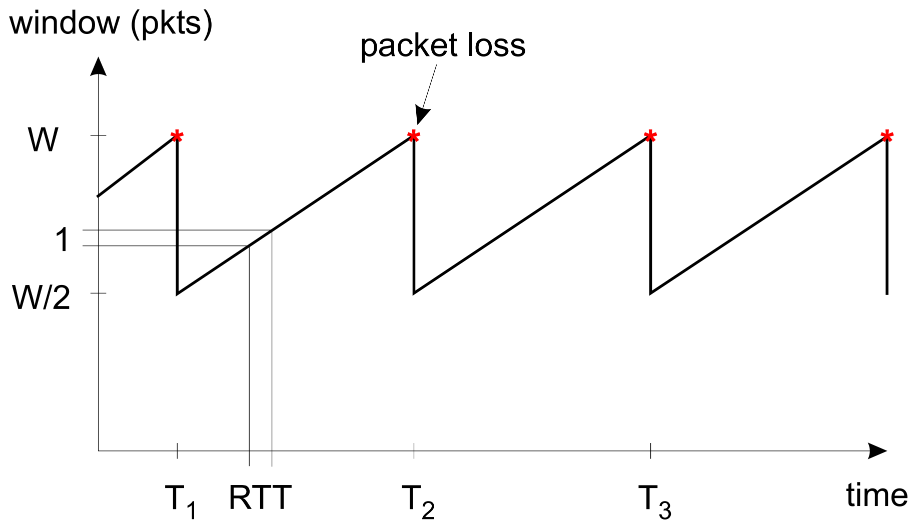
- An optional mechanism (RFC 2018, October 1996)
- Allows the receiver to acknowledge packets out of order
- Requires support from both ends of a TCP connection
  - the connection functions without SACK if supported only by one of the hosts

TCP Vegas

- A new experimental TCP version
- Employs three techniques to increase throughput and decrease losses
  - the first technique results in a more timely decision to retransmit a dropped segment
  - the second technique gives TCP the ability to anticipate congestion and adjust its transmission rate accordingly
  - the third technique modifies slow start mechanism so as to avoid packet losses while trying to find available bandwidth

Throughput analysis of TCP

- Simple model by Floyd and Fall[3]
  - stationary model (long flows) for TCP operating in the congestion avoidance mode
  - average packet drop probability p and the RTT determine the throughput
  - naïve assumption of periodicity: every $1/p$ th packet is lost



[3] S. Floyd and K. Fall, *Promoting the use of end-to-end congestion control in the Internet*, IEEE/ACM Trans. on Networking, 1999.

Throughput analysis of TCP (continued)

- Number of packets sent between two dropped packets (e.g., between T_1 and T_2)

$$\frac{W}{2} + (\frac{W}{2} + 1) + \cdots + W \approx \frac{3}{8} W^2$$

- There is one dropped packet per $(3/8)W^2$ sent packets. Thus

$$p = \frac{8}{3W^2} \qquad \Rightarrow \qquad W = \sqrt{\frac{8}{3p}}$$

- The time between two packet drops is $W/2$ round trip times. We get the throughput T

$$T = \frac{(3/8)W^2 B}{(W/2)RTT} = \frac{3}{4}\,\frac{W \cdot B}{RTT} = \frac{\sqrt{3/2}\,B}{RTT\sqrt{p}} \approx \frac{1.22\,B}{RTT\sqrt{p}}$$

  where B is the size of one segment

Throughput analysis of TCP (continued)

- A more refined model has been presented by Padhye et al.[4]
- Takes into account retransmit timeouts and delayed acks

$$T \approx \min\left\{\frac{W_m B}{RTT}, \frac{B}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\{1, 3\frac{3bp}{8}\}\, p\,(1+32p^2)}\right\},$$

where

- W_m is the maximum size of the congestion window set by the receiver
- b is the number of packets acknowledged by a received ACK
- T_0 is the length of the first timeout period

[4] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, *Modeling TCP throughput: a simple model and its empirical validation*, in Proc. of ACM SIGCOMM, 1998.