Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Networked Systems
ETH Zürich — seit 2015

Communication Networks

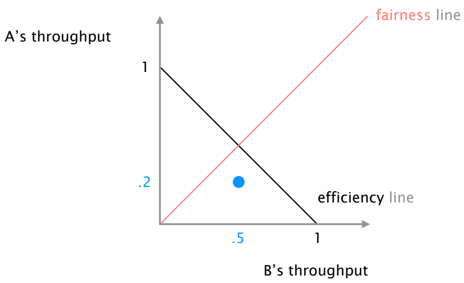
Prof. Laurent Vanbever

Solution: Exercises week 11 – TCP Congestion Control

Fairness

Consider the situation in which two hosts, A and B, are concurrently using a 1 Mbps link with a Maximum Segment Size (MSS) of 100 kb.

Assuming that B starts with 500 kbps and A with 200 kbps (see left picture). Describe the evolution of the throughput of the two hosts when:



Are you getting a fair share?

a) A and B rely on Additive Increase Multiplicate Decrease (AIMD).

Solution: By drawing on the left picture, you can show that AIMD eventually converges to the fairness state. For instance, assuming that both sender increase their CWND by 1 MSS when there is no congestion and divide it by 2 upon congestion, the following points can be drawn:

- (.2, .5)
- (.3, .6)
- (.4, .7) > congestion!
- (.2, .35)
- (.3, .45)
- (.4, .55)
- (.5, .65) > congestion!
- (.25, .325)
- ...

It can be seen that, because of its bigger share, B looses more than A because of the halving, eventually the system evolves along the fairness line.

b) A and B rely on Multiplicative Increase Additive Decrease (MIAD).

Solution: Again, by drawing on the left picture, you can show that MIAD does not converge to the fairness state at all, but rather to one in which A is completely shut down. For instance, assuming that both sender double their CWND when there is no congestion and decrease it by 1 MSS upon congestion, the following points can be drawn:

- (.2, .5)
- (.4, 1) > congestion!
- (.3, .9) > congestion!
- (.2, .8)
- (.4, 1.6) > congestion!
- (.3, 1.5) > congestion!
- (.2, 1.4) > congestion!
- (.1, 1.3) > congestion!
- (0, 1.2) > congestion!
- (0, 1.1) > congestion!
- (0, 1)
- ...

It can be seen that the sender which benefits from a bigger initial share will end up using the entire link.

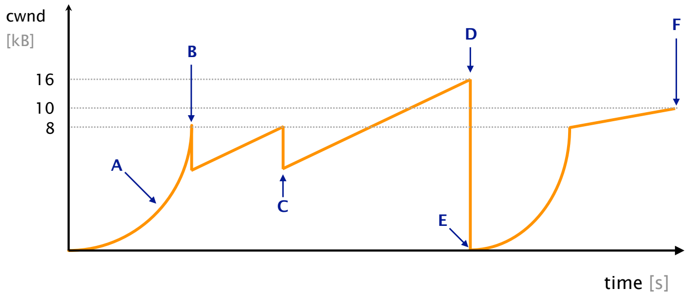
Assume now that only A is malicious, and wants to cheat congestion control to get more throughput. Describe two distinct ways A could do so and what would be the net effect on B's throughput.

Solution: There are many ways A could cheat congestion control. Two simple ones would be: *i)* increase its congestion window faster than what is prescribed, e.g. instead of increasing its CWND by 1 per RTT, it could increase it by 2 or 3 per RTT; *ii)* opening up many connections and therefore profit from more overall throughput as TCP tends to allocate throughput equally across all connections.

It is worth nothing that detecting cheating is hard as controlling all end-points in the Internet is rather hopeless. Yet, the Internet continues to work and we haven't faced a new congestion collapse event since the 80s.

Congestion Window

Consider the following plot which depicts the evolution of the size of the TCP congestion window of the sender.



What kind of network conditions is this flow seeing?

Describe briefly:

a) What happens at point B?
Solution: Triple duplicate ACKs.

b) Does the event happening at point B require the network to discard packets? Why or why not?
Solution: No. This could be caused by packet reordering (e.g., due to queuing or asymmetric paths).

c) What happens at point D?
Solution: A timeout event which causes the sender to decrease its window.

d) Does the event happening at point D require the network to discard packets? Why or why not?
Solution: No. Congestion (queuing delay) in the forward or backward direction could cause the round-trip time to be higher than the retransmission timeout.

Consider that the Maximum Segment Size (MSS) of the connection is 1 kB and the Round-Trip Time (RTT) between the two end points is 100 milliseconds. The sender opens the connection at time $t = 0$. Transmission delay in this network is negligible, so you should only consider the propagation delay in the following.

a) How much time has elapsed at point B?
Solution: Two solutions would be accepted here depending on when you consider B has received the triple duplicate ACKs. Total time: 1 RTT for the TCP handshake + 3–4 RTT in slow-start (1, 2, 4, (8) MSS) = 4 or 5 RTT, i.e. 400 or 500 ms.

b) How much time has elapsed *between* point C and D?
Solution: The congestion window grows from 4 MSS to 16 MSS linearly at a growth of 1 MSS per RTT, translating to 12 periods of RTT or 1.2 s.

c) How much time has elapsed *between* point E and point F?

Solution: From E, we start with slow start phase to a 8K window size which takes 4 RTT (1, 2, 4, 8 MSS). After that we flip to congestion avoidance and follow an additive increase from a 8 to a 10 MSS window size (8, 9, (10) MSS) which takes 1 or 2 RTTs depending on when you consider F happens. We would therefore accept either 5 or 6 RTT, i.e. 500 or 600 ms.

Briefly explain how come point D is higher than point B. Would you expect this to happen often?

Solution: D's height can vary as a consequence of other concurrent flows being sent along the same link. This is therefore likely to happen all the time.