CSE123 Spring 2015 Homework #4
Instructor: Stefan Savage

Due Friday 6/5 **at the beginning of class**
(Please TYPE your answers… this will allow our TAs to be much more efficient)

1. TCP performance

a) Consider a TCP-based application in which a sender tries to transfer 100MB to a receiver. The bottleneck link between the sender and receiver is 10Mbps and there is no competing traffic. The round-trip time is 10ms and the packet size is 1KB. The receiving application reads 100KB of data every second and has 1MB of buffer. Will this application by limited by congestion control or flow control? (explain your answer)

This application is limited by flow control. The application is only reading at 100KB per second. The 1MB buffer will fill fairly quickly after which the steady state draw of the application will be 100KB/sec (i.e., 800Kbps). For a router with minimal buffering congestion control will not cause backoffs until the congestion window is over 1250 packets, and even if cwnd were reduce in half at that point (625 packets) this is considerably more than 100KB/sec.

b) Consider two TCP flows: one between hosts A and B with a RTT of 10ms and a bottleneck of 10Mbps and the other between hosts C and D with a RTT of 100ms and a bottleneck of 100Mbps. Suppose you need to transfer a 1MB file between each pair of hosts (from A to B and from C to D). How fast will each transfer take and which will complete first (or will they complete at the same time?)

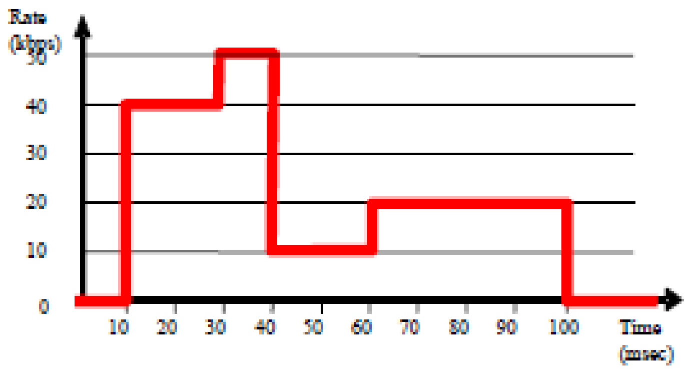
The flow between A& B will complete first. Using TCP congestion control, the 1MB file will be covered by k windows (the first window will hold 1 packet and will double every RTT due to slow start). The bottleneck capacity of the link, even at 10Mbps, is higher than the total size so we will never lose a packet. So how big is k? If we have 1500 byte packets, its $\log_2$(1MB/1.5KB) which is ~9.3 (round up to 10) RTTs or 100ms. With a 100ms RTT the flow between C&D will take 10 times longer (1second approximately).

c) In class we've explained that the TCP receiver sends an ACK in response to each packet it receives. However, this is not true in practice. For historical reasons, TCP implements a "delayed ACK" policy at the receiver, in which acknowledgements may be delayed up to 500ms in the hope that another packet will be received and a single ACK can be sent for both (TCP guarantees that the second in-sequence packet will always generate an ACK). For a large transfer (e.g., 100MB) does this optimization make TCP send more quickly, less quickly or the same? (explain your answer)

Delayed ACKs will make TCP send more slowly because the ACKs implicitly control the window size at the sender. Send ACKs less frequently will cause the congestion window to open more slowly and thus it will take longer for TCP to reach its maximum throughput.

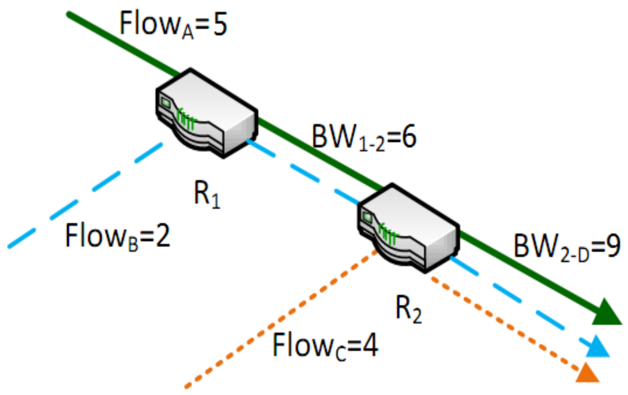
2. Quality of service

a) The following graph shows the rate at which data enters a router implementing a token bucket-based traffic policing policy with **r**=15kbps. What is the *minimum* size (in bits) for the token bucket depth, **b**, to ensure that *none* of this traffic will be dropped?



Min B = ((40-15) *20) + ((50-15)*10) – ((15-10) * 20) + ((20-15)*40) = 950bits

b) In the diagram below, routers R1 and R2 implement fair queuing. Flow_A transmits at a rate of 5 Kbps at router R1. Flow_B transmits at a rate of 2 Kbps at R_1. Both are competing for the same outgoing link at R_1 and that link has 6 Kbps of available bandwidth (represented by BW_{1-2} in the figure). Flow_C transmits at a rate of 4 Kbps at R_2, where Flow_A is competing with Flow_B and Flow_C. How much bandwidth will each flow be allocated when exiting R_2?



R1 output link: FlowA=4Kbps, FlowB=2Kbps
R2 output link: Flow A= 3.5Kbps, Flow C = 3.5Kbps, FlowB = 2Kbps