6.263 Data Communication Networks
MIT, Fall 2005

November 3, 2005

**Solutions to Problem Set 6**

**Problem 6.1**

(a) The propagation delay along a single link is the time it takes to transmit a bit from one end to the other. The *end-to-end propagation delay* is the sum of the propagation delay on all links along the path. This includes both the links along the forward path and the reverse path (i.e., the links traversed by data and those traversed by acks).

The *round trip time* (RTT) is the time to send a packet and receive its acknowledgement. This includes both the end-to-end propagation delay and the queueing delay along the path. As the queue builds up, the propagation delay stays the same whereas the RTT increases.

The *congestion window* (cwnd) is a TCP variable that limits the maximum number of packets that a sender can keep in flight (i.e., the maximum unacknowledged packets).

(b) Yes. Recall that the sending rate = cwnd/RTT. Assume that at time $t$ the rate was capacity+$\epsilon$. Assume also that the RTT at time $T$ is $x$, and the congestion window at time $T$ is $w$ and is kept constant. Since the rate is larger than the capacity, the queue will build up. As a result, the RTT will increase. After some time $t + dt$, the RTT will be $x + dx$. The new rate at $t + dt$, will be $w/(x + dx)$. Thus, the new rate is smaller. The queue will continue to build up until the rate becomes equal to capacity. Thus, even if the sender does nothing and keeps the cwnd at its current value, the RTT will increase, resulting in a reduction of the sending rate. Obviously this will not happen if the sender increases its cwnd, or if the buffer size is too small such that the RTT cannot grow enough.

(c) The sender might receive 3 dupacks because of substantial reordering (even if there is no drop). For example, if the receiver receives packets: 15, 17, 18, 16 in this order, it will send 3 dupacks to the sender, triggering a retransmission of packet 16.

When the congestion window is too small, i.e., less than 4 packets, there are not enough packets in flight to generate 3 dupacks. For example, assume cwnd=3 packets. Packet 1 is dropped. Packet 2 and 3, when received, generate 2 dupacks but there are not more packets in the pipe. The sender will timeout. Similarly, even when the congestion window is large, it is possible that there are too many dropped packets from the same cwnd such that there are not enough packets in the pipe to trigger 3 dupacks. (Some of these problem were addressed by a TCP version called NewReno).

(d) With MIMD, it is possible to approach efficiency but not fairness (try to check the phase plot).

(e) We want sender 1 to simulate 2 TCP senders. The simplest approach is to make sender 1's increase-decrease rules as follow:

- If no loss, cwnd = cwnd + 2 (per RTT).
- Whenever there is a loss, cwnd = cwnd - (1/4)cwnd.

(f) The TCP window will oscillate between $W_{max}$ and $0.5W_{max}$. The maximum window size is bounded by "the number of packets TCP can fit in the queue" + "the numver of packets in the pipe". The maximum number of packets in the pipe is $CD$. Thus, $W_{max} = B + CD$, where $B$ is the buffer size. If we want the utilization to stay optimal (i.e., 100%) even when cwnd $= 0.5W_{max}$, then we need to the minimum cwnd to be at least as large as the pipe. Thus, $0.5W_{max} \geq CD$. By substituting $W_{max}$, we get $B \geq 0.5CD$. The number $C \times D$ is usually referred to as the delay-bandwidth product. A common practice in router design is to make the buffer size equal to the delay-bandwidth product.

**Problem 6.2**

(a) Because max_thresh is compared against the average queue, not the instantaneous queue size.

(b) To cope with bursty traffic. In particular, a queue that builds up and drains in less than RTT is not a sign of congestion. Only a persistent queue, i.e., a queue that persists for longer than RTT is a sign of congestion.

(c) In particular, the drop probability in RED is a linear function in the average queue, $p = b \cdot q_{avg}$, where $b$ is a constant. The TCP throughput is inversely proportional to the square root of the drop probability, $\lambda = a/\sqrt{p} = k/\sqrt{q_{avg}}$, where $a$ and $k$ are constants. Assume that $n$ TCPs share the link, and also that the capacity is $C$. Then in equilibrium, we want $C = n \cdot k/\sqrt{q_{avg}}$. Thus, with fixed $C$, the average queue size will increase in proportion to $n^2$.

**Problem 6.3**

Alissa Hacker has an idea for improving TCP's performance. Alissa has heard that using packet drops as a sign of congestion works badly in very high bandwidth environments. Further, drops waste network resources and do not explicitly tell the sender the severity of congestion. Alissa thinks that TCP should use increased delay as a signal of congestion. In particular, she argues that as long as the peak input traffic at the bottleneck does not exceed its capacity, there will be no queues and the RTT of the sender should stay close to the propagation delay, which is constant. If the input rate at the bottleneck exceeds its capacity, the RTT will grow due to queuing delay.

(a) Alissa designs a scheme in which the sender keeps increasing its congestion window as long as the RTT stays constant and slows down once the RTT starts increasing. The sender tries to figure out the maximum sending rate at which the RTT stays constant. Assume that there is only one source of traffic and that it follows Alissa's protocol. Ben argues that Alissa's scheme may not work if the sender is bursty and that the sender needs to pace its traffic. Explain why Ben is right.

**A:** A bursty source would likely cause the queues of routers along the path to build, particularly for slower links. As these queues build and drain, the RTT will vary accordingly. The RTT may increase even when the average input traffic is less than the capacity of the link. If Alissa's protocol causes the sender to close its congestion window when the RTT deviates from constant, the path would be underutilized.

Alissa agrees with Ben and makes the sender pace its traffic to ensure that the instantaneous send rate is close to the average over an RTT. Alissa replaces the AIMD update rules in TCP

with a new `cwnd` update rule. She doesn't change slow start, and you should ignore slow start for this problem. The sender keeps a measure of the minimum RTT seen so far, called BaseRTT. This BaseRTT is used as an estimate of the propagation delay. The congestion window is computed in packets while RTTs are measured in seconds. Assume that round trip times are always less than 100ms.

Every T=100ms, the sender computes the average RTT seen in the last T, and:

$$cwnd = cwnd\left(\frac{BaseRTT}{RTT}\right) + \alpha$$

where $\alpha > 0$ is a constant. Alissa's protocol does not reduce cwnd in response to lost packets.

(b) Why it is important to have a positive $\alpha$? If $\alpha$ were zero, what would go wrong?

**A:** For any send rate below the capacity of the system, the average RTT will be exactly BaseRTT. Thus, if $\alpha$ is zero, the congestion window will remain constant, even if the network is underutilized.

(c) If Alissa's TCP shared the bottleneck link with a standard AIMD TCP that uses dropped packets as the congestion signal, which one do you expect to get more throughput? Why? Assume a shared drop tail queue and a very large queue limit.

**A:** We would expect standard AIMD TCP to "win" and achieve more throughput. Since AIMD TCP uses loss as a congestion signal, it will continue increasing its congestion window until there is a loss, i.e. after the queue is filled and a router along the path is dropping packets. As the queues fill because of this standard AIMD TCP source, Alissa's TCP will slow, by closing the congestion window, in proportion to the increase in RTT. Thus, AIMD TCP gains the throughput that Alissa's TCP is giving up due to the increased RTT.

(d) Assume the network contains only Alissa's TCPs. Assume also that everyone has the same propagation delay. Is Alissa's TCP fair? Explain. Assume that no packets are dropped.

**A:** Yes, Alissa's TCP is fair. There are several ways to see this; a simple argument can be made via phase plots as per Chiu and Jain. First, note that Alissa's protocol does AIMD in a single step, with an adaptive decrease factor (The decrease factor is $1 - \frac{BaseRTT}{RTT}$). Assume that both sources begin beneath the efficiency line ($x1 + x2 = C$). They will therefore see a constant RTT and will increase their congestion window by $\alpha$ in each 100ms period. Thus, as in AIMD, both sources increase by the same amount and therefore move in the phase plot at a 45 degree angle. They will do this until they pass over the efficiency line. At this point, they will both see the same increase in RTT. Therefore each source decreases proportionally to its congestion window, effectively moving them along the line that passes through the origin. An interesting effect of Alissa's protocol is that the decrease is proportional to the increase in RTT. Thus, in contrast to TCP AIMD, the sources are less likely to overshoot their decrease and end up at a point closer to the efficiency line. In sum total, just as in TCP AIMD, they are pushed from any point toward efficiency and fairness.

(e) If the network contained only UDP and Alissa's TCPs, would Fair Queuing be useful? Explain.

**A:** Yes, fair queuing would be useful. Neither AIMD TCP nor Alissa's TCP are immune to the effects of non-TCP friendly sources. Fair queuing provides flow isolation and would therefore be useful.

(f) Alissa says that, theoretically, the congestion window in her protocol is less likely to show oscillations as wide as TCP's sawtooth. Explain why this is true.

**A:** This is true by a similar argument made as to why the sources don't overshoot their decrease in the phase plots (in question D). Since sources are no longer multiplicatively decreasing by 1/2, but rather decreasing in proportion to the increase in RTT, the congestion window is less likely to show the sawtooth oscillations we see in AIMD.

(g) Alissa claims that since her TCP does not use drops as a sign of congestion, it will work well in both low-speed and very high-speed networks in which a single flow could send at gigabits/second.

(a) Explain why current TCP has difficulties in very high bandwidth networks.

**A:** To fully utilize a high bandwidth network, TCP must have a very large congestion window. Many improvements to TCP have been added, for instance the window scaling option, SACK, etc., however the fundamental problem remains that TCP requires an unrealistically low loss rate to maintain, on average, a large enough window for high bandwidth paths. Given that packets will always be lost because of errors, the loss rate can never be zero. Further, after a loss TCP ramps up by 1 packet per RTT. As a result, it takes a very long time to build up the window to its value before the loss.

(b) Ben argues that, though Alissa's protocol reacts to delay rather than drops, its cwnd update rules are not that different from AIMD. The increase constant is $\alpha$ per 100ms rather than one packet per RTT. Explain this statement.

**A:** Again, this answer comes directly from the phase plot argument in question D. Sources increase linearly in $\alpha$ and decrease multiplicatively in proportion to the increase in the RTT. Thus, the congestion window update rule is similar to AIMD's.

(c) Explain to Alissa why a single value of $\alpha$ won't work well in both low- and very high-bandwidth networks.

**A:** If $\alpha$ is too low, Alissa's protocol will take an unacceptably long time to open its congestion window large enough to utilize very high-bandwidth paths. If $\alpha$ is too high, when the RTT is constant, the source will increase its congestion window too much and overshoot the available link bandwidth in low-bandwidth networks. Therefore a single value of $\alpha$ doesn't work well for both high and low-bandwidth networks.