EECS 122: Introduction to Communication Networks

Homework 9

(6 points)

Due: 1999-Nov-05-Fri (in class, or 467 Cory by 2pm)

Problem 1. TCP is fair if we consider storage in the network to be the resource being shared, because competing TCP connections tend to get the same window size. But if we consider bandwidth to be the shared resource, then TCP is unfair because connections with longer round-trip times get lower throughput.

a) (2 points) If we wanted fairness in terms of bandwidth, what simple and crude modification could we make to TCP (on all hosts) to achieve this? (Hint: How do web browsers increase their throughput?)

b) (food-for-thought) How could the same idea be applied less crudely, by adjusting the parameters of the congestion control algorithms?

Problem 2. If TCP's retransmission timeout is too small, a slow packet will be misinterpreted as being lost, and an unnecessary retransmission will be sent. Before the introduction of congestion control in 1988, this was a minor annoyance—if 5% of the packets were mistakenly thought to be lost, then the source would send at 105% of the rate it needed to. But with congestion control, every window of packets that experiences at least one loss causes the congestion window to be halved. This will cause a connection that mistakenly thinks 5% of its packets are lost to have an average window size of about four packets, even if the network is uncongested. (For a packet loss probability of p, the average window size will be roughly $\sqrt{1/p}$ packets.) This is why the addition of congestion control was accompanied by an improvement in round-trip time estimation. The old method used an exponentially weighted moving average of the mean only, and the retransmission timeout was set to a multiple of the estimated mean. RFC 793 suggested a factor between 1.3 and 2.

a) (1 point) Suppose a TCP connection experiences round-trip times of 10 ms for 80% of its packets, and 100 ms for 20% of its packets (perhaps because some other source is bursty, causing a router queue to fluctuate between being empty and mostly full). Suppose no packets actually get lost. If the estimated mean is close to the true mean, and the timeout is set to 2 times the estimated mean, then what fraction of the packets will TCP mistakenly believe are lost?

b) (1 point) The new method estimates both the mean and the mean deviation (the mean deviation is the average absolute distance of samples from the mean), and sets

the timeout to the mean plus a multiple of the mean deviation (let's say the factor 4 is used). If the connection from part (a) uses this new method, and the estimated mean and mean deviation are close to their true values, what fraction of the packets are mistakenly believed to be lost?

c) (food-for-thought) The code for updating the timeout given a new round-trip time sample looks something like this:

```
err = samp - mean;
mean += a * err; (0 ≤ a ≤ 1)
dev += b * (abs(err) - dev); (0 ≤ b ≤ 1)
timeout = mean + c * dev; (c ≥ 0)
```

It might be nice to guarantee that the timeout is never less than the the latest sample (because if it happens once, it could easily happen again). Derive a necessary and sufficient condition on a, b, and c to make this guarantee. TCP uses $a = \frac{1}{8}$, $b = \frac{1}{4}$, $c = 4$. Does it make this guarantee?

Problem 3. (2 points) A standard analog television signal contains frequencies up to 5 MHz. If we sample the signal losslessly, and quantize the samples so that the signal-to-quantization-noise ratio is about 48 dB, then the resulting bit rate is at least what?