EECS 122: Introduction to Communication Networks

Homework 7 Solutions

Solution 1. Recall that a constant bit rate stream is said to conform to the parameters PCR and CDVT if we can imagine that the stream is copied into a leaky bucket with capacity 1 cell $+$ CDVT $\cdot$ PCR that drains at rate PCR, and the bucket never overflows. Ten cells arrive in each frame, so cells arrive at an average rate of

$$\frac{10 \text{ cells}}{\left(10 \text{ cells} \cdot \frac{53 \text{ bytes}}{\text{cell}} + 95 \text{ bytes}\right) \cdot \frac{8 \text{ bits}}{\text{byte}} \cdot \frac{\text{second}}{40 \times 10^6 \text{ bits}}} = 80,000 \text{ cells/second}$$

If the bucket drains any slower than that, it will eventually overflow, so let's suppose our bucket drains at that rate, and see if we can find a bucket large enough that it won't overflow. We can assume the bucket is empty when the first cell arrives. The second cell will arrive one cell-time later, where a cell-time is

$$53 \text{ bytes} \cdot \frac{8 \text{ bits}}{\text{byte}} \cdot \frac{\text{second}}{40 \times 10^6 \text{ bits}} = 10.6 \ \mu\text{s}$$

The first cell is not fully drained when the second arrives, so the bucket is more full just after the second cell arrives than just after the first arrived. Similarly, it will be even more full just after the third cell arrives, and so on until the last cell in the frame (the tenth) arrives. During the nine cell-times between the arrival of the first cell and the arrival of the tenth, ten cells enter the bucket, while the number that leave is

$$9 \cdot 10.6 \times 10^{-6} \text{ seconds} \cdot \frac{80,000 \text{ cells}}{\text{second}} = 7.632 \text{ cells}$$

Therefore 2.368 cells must still be in the bucket. The gap between the last cell of the first frame and the first cell of the second frame is $(53+95)$ bytes instead of the usual 53 bytes (because of the trailer and header). Not coincidentally, this provides just enough time for those 2.368 cells to drain from the bucket (because we let the bucket drain at the average arrival rate), so the whole process repeats for each frame. The most full the bucket ever gets is 2.368 cells, so we set that equal to 1 cell $+$ CDVT $\cdot$ PCR, and find that

$$\text{CDVT} = 1.368 \text{ cells} \cdot \frac{\text{second}}{80,000 \text{ cells}} = 17.1 \ \mu\text{s}$$

Solution 2. In the alternating bit protocol, the source sends a message (perhaps repeatedly) until it receives an acknowledgement for that message, then it moves on to the next message. So the throughput is one packet per round-trip time. When the data rate is C, the round-trip time is

$$\rho = \frac{440 \text{ bytes}}{C} + 0.005 \text{ second} + \frac{60 \text{ bytes}}{C} + 0.005 \text{ second} = \frac{500 \text{ bytes}}{C} + 0.01 \text{ seconds}$$

The time to transmit a data packet is

$$\tau = \frac{440 \text{ bytes}}{C}$$

The efficiency is the throughput (1 packet$/\rho$) divided by the channel's packet rate (1 packet$/\tau$), which is τ/ρ. Plugging in the various values for C (and remembering the factor of 8 bits per byte), we get:

data rate	throughput	efficiency
1 kbps	0.249 packets/second	87.8%
100 kbps	20.0 packets/second	70.4%
10 Mbps	96.2 packets/second	3.38%
1 Gbps	99.960 packets/second	0.0352%

Solution 3. One solution is, instead of having the source repeatedly send a message until it gets an ack, have the sink repeatedly request a message until it gets the data. So the code for the source and sink is roughly swapped:

```
sink:   R ← 0
        while true
          send R,req
          wait up to T for message
          if received message then
            deliver message
            R ← not R
          endif
        endwhile
```

```
source: S ← 0
        M ← first message
        while true
          wait for any seq,req
          if seq = not S then
            M ← next message
            S ← not S
          endif
          send M
        endwhile
```