Network Systems (201600146/201600197), Test 1

February 15, 2019, 13:45–15:15

Answers

1. Drexit voting information

1 pt (a) A.

The second-most popular choice was B, but packet-switching by itself is not more reliable than circuit-switching; in fact, with the possibilities of packet drops due to e.g. queue overflows or routing errors, one could argue it is *less* reliable.

3 pt (b) Use the formula: $H = 0.49 \log_2 \frac{1}{0.49} + 0.50 \log_2 \frac{1}{0.50} + 0.01 \log_2 \frac{1}{0.01} = 1.0707$ bits per word.

3 pt (c) Correct answers are, respectively: G, C, G, G, B.

MC02: this code is impossible: since 0 and 1 on their own are already valid codewords, there are no possibilities left to make a longer codeword; e.g., an extra codeword 01 would be indistinguishable from the codeword 0 followed by the codeword 1.
MC04: no choice satisfies the stated requirement that the average message length must be less than 1.6 bits.
MC05: same reason as MC04.

1 pt (d) G.

All information would still get across (since every vote is still encoded in a uniquely decodeable way), but the average number of bits needed would be higher. An optimal code will assign a longer codeword to the unlikely `I don't care` vote, and this longer codeword will now be used much more often than expected.

1 pt (e) C.

The best code you could find in question (c) for encoding the votes separately, needs about 1.5 bits per message. That's much more than the 1.07 bits calculated in question (b). The way to get closer to the theoretical limit from (b), is to take messages together. E.g., if you take 2 messages together, you have a total of 9 "combined messages" (Remain+Remain, Remain+Leave, and so on), each with their own probability (0.49×0.49, 0.49×0.50, and so on. The total Shannon information of these 9 possibilities will be exactly double what was calculated in (b), but it is now possible to choose 9 bit patterns of which the average length gets closer to this.

3 pt (f) First, calculate the channel capacity: $C = 2000 \cdot (1 - 0.98 \log_2 \frac{1}{0.98} - 0.02 \log_2 \frac{1}{0.02}) = 1717.1$ bits per second.
Divide this by the entropy from question (b) to find the answer: 1603.7 votes per second.

Very many students only calculated the channel capacity, but forgot to compute the actual number of messages per second which the question asked for.

1 pt (g) C.

Many chose A, but that's not correct. The error probability can be made arbitrarily small if the information rate is *less* than the channel capacity, but the question asked what would happen if one tries to send *more*.

Continued on next page...

2. Protocols and performance

2 pt (a) Propagation time = $\frac{\text{distance}}{\text{speed}} = \frac{800}{200\,000}$ = 0.004 s = 4 ms.

2 pt (b) Transmission time = $\frac{\text{number of bits}}{\text{bit rate}} = \frac{10000}{1000\,000\,000}$ = 0.000 01 s = 0.01 ms

2 pt (c) Each packet takes 1 transmission time (0.01 ms), then a one-way propagation time (4 ms). Then an ack is sent (assuming it's a short packet, its transmission time is negligible), followed by again propagation time (4 ms). So 10000 bits per 8.01 ms, that's 1.25 Mbit/s.
(or 125 packets/second)

2 pt (d) To achieve this, one needs to be able to transmit continuously during an entire RTT; number of packets fitting in 1 RTT is
$\frac{\text{RTT}}{\text{transmission time of 1 packet}} = \frac{8}{0.01} = 800$ packets.
So after the first packet is sent, another 800 packets are sent before an ack for the first can be received. Thus the send window must be at least 801 packets.

800 packets was also accepted as correct, or the delay bandwidth product which is $8 \cdot 10^6$ bits.

1 pt (e) D.

We've learnt, partially in the book and partially in the tutorial class, that confusion between earlier and later packets with the same sequence number can occur if RWS+SWS > number of possible sequence numbers. In this question, that is not the case, so it will work correctly.

1 pt (f) D.

Same reason as (e).

1 pt (g) C.

This time, RWS+SWS > number of possible sequence numbers, so indeed, problems can occur. We've seen example time-sequence diagrams illustrating how things can go wrong then: a retransmission of an old packet being delivered in duplicate, taking the place of a newer one with the same sequence number.

---

3. Application protocols

1 pt (a) G.

HT in HTTP stands for HyperText, and hypertext means documents (texts) referring to other documents which the reader can easily navigate to (i.e., click on a link in a web page).

2 pt (b) B.

Using this header, the provider of e.g. advertisements can see which page the user fetching the advertisement is currently reading. That's a breach of privacy, especially if advertisements of this same provider are on many different websites so the advertiser can track the user's interests.

2 pt (c) A.

The parallel connections allow the client to fetch both images at the same time, so this speeds up 1.1 compared to 1.0, regardless of whether the files are on the same or a different server.
Thus, the answer must be A, regardless of whether persistent connections speeds up version 1.1 even further (that depends on where the images are hosted).

Continued on next page...

4. Client-server and peer-to-peer file distribution

2 pt (a) D.

With client-server, the server must sent a separate copy to each of the clients. The data rate in this case is limited by the server's uplink, 10 Mbit/s. So this takes 650 (number of clients) $\times$ $500 \cdot 2^{20}$ (bytes per file) $\times$ 8 (bits per byte) / $10 \cdot 10^6$ (bits per second) $\approx 2.72 \cdot 10^5$ seconds. (Or 260000 seconds if computed with Mbyte = 10^6 bytes)

2 pt (b) C.

The four expressions within the $\max\{\dots\}$ describe the four possible limiting factors:

- The uplink speed u_s of the British file server, which has to send the file to two Deuropean file servers.
- The download speed d_{s2} of each of the Deuropean servers, through which the file needs to go once.
- The uplink speed $2u_{s2}$ of the two Deuropean servers together; the file needs to be sent N times through this, namely once to each of the N members of parliament.
- The download speed d_{mp} of each of the members of parliament; the file needs to once through each of them.

Many students chose option G, which would be correct if the file is first transferred compltely to both Deuropean servers (first $\max\{\dots\}$), and *after* that has completed it is sent from the Deuropean servers to the MPs (second $\max\{\dots\}$). However, the text clearly says that the Deuropean servers immediately start sending out the data after they have received the first bit, so they don't wait until they have received the file completely.

2 pt (c) This is essentially a P2P network just like discussed in the K&R text, with $N + 2$ peers, namely N Deuropean MPs and 2 Deuropean servers, which have different upload and download rates. Substituting this into equation (2.2) from the K&R text gives:

$$D_{P2P} \geq \max\left\{\frac{F}{u_s}, \frac{F}{\min(d_{mp}, d_{s2})}, \frac{(N+2)F}{u_s + Nu_{mp} + 2u_{s2}}\right\}$$