

**ECEU-628: Computer Communications Networks
Final Examination Spring 2004: Solutions**

Prof. A. Bruce McDonald[†]

mcdonald@ece.neu.edu

**[†]Department of Electrical and Computer Engineering
Northeastern University**

April 7, 2004

Problem One: Short-Answer

(A) ASCII files often contain control characters, for example, CR and LF. The file transfer protocol Kermit, however, cannot send such characters because they are not printable and may cause unpredictable actions by the communications hardware or driver software at a receiving host. Explain how Kermit handles the transmission and reception of ASCII control characters within the text of a file?

Solution In the header fields of Kermit packet the designers of the protocol could easily require that only 'printable' ASCII characters be used. In some control fields this requires **characterization**, which simply involved adding 20h (32) to **promote** the raw number, for example a length, into a value between 20h and FEh. The receiver **demotes** the field by subtracting 20h. This is simple because it is an a priori rule that applies to certain fixed fields. The contents of the file, however, are not known a priori. Moreover, the designers of Kermit could not presume to limit the values of any bit sequence as they could with control fields. The technique they developed is known as **controlification**. It involves flipping the 7th bit of the character and sending a prefix character that is negotiated at the start of the session. As an example consider the CR LF combination—the flip operator is the bitwise XOR with 40h: $\text{ctrl}(\text{CR}) = \text{CR} \text{ XOR } 40\text{h} = 'M'$ and $\text{ctrl}(\text{LF}) = \text{LF} \text{ XOR } 40\text{h} = 'L'$; let the prefix encoding character be '#', 'CR LF' is then transmitted as '#M#L'. The ctrl operation undoes itself, hence, whenever the receiver sees the prefix character it drops it from the incoming data (same principle as bit stuffing) and repeats the ctrl operation to retrieve the original character. Hence '#M' becomes $\text{ctrl}('M') = \text{CR}$.

(B) Let the SNR for a given transmission channel be 30.1dB. According to Shannon's Capacity Theorem approximate the minimum bandwidth in Hz required to support one digital PCM voice stream (in one direction)? How does this compare to the bandwidth of the original analog signal?

Solution The maximum **theoretical** information rate of a channel is limited by the signal-to-noise ratio and the channel bandwidth—all channels act as filters. If C is the information rate (in bits-per-sec), B is the bandwidth (in Hertz) and $\frac{S}{N}$ is the signal-to-noise ratio (without units— S and N are the received voltage levels—not power), then $C = B \log_2(\frac{S}{N} + 1)$. Since voltage levels are being used the signal-to-noise ratio in dB is: $\text{SNR}(\text{dB}) = 10 \log_{10}(\frac{S}{N})$. Hence, 30.1dB implies $\frac{S}{N} \approx 1000$. (exact: $\frac{S}{N} = 1023$). Since PCM Voice is 64000 and $\log_2(\frac{S}{N} + 1) = 10$, then $B = \frac{64000}{10} = 6400$ Hz, whereas the original analog source is band-limited 4000 Hz signal. Thus, the increase in bandwidth required is greater than 50%.

(C) Given an IP address: 155.224.128.27 with a netmask 255.255.240.0 answer the following: What class network does this address come from? Identify the class 'X' network ID the total number of hosts that can be supported. How does this compare to the number supported in a class 'X' network without subnets?

Solution To find the network class examine the first octet in binary: $155 = 10011011$, hence it is a class B network which has a network mask 255.255.0.0, thus, the class B network ID is obtained by bitwise ANDing the address with the network mask = 155.224.0.0. To determine the number of hosts that can be supported we must first determine the number of subnets that the class B network has been divided into. Since the subnet mask is 240 there are 4 bits allocated to the subnet ID. Hence, we are permitted $2^4 - 2 = 14$ (all ones and all zeros are not permissible). There are 12 bits allocated to the host ID on each subnet (assuming no deeper levels of subnet!). Hence, each subnet can support up to $2^{12} - 2 = 4094$. The aggregate number of host IDs is $14 * 4094 = 57316$. A flat class B network supports $2^{16} - 2 = 65534$ hosts. Thus, 8218 host IDs are 'given up' due to the two infeasible subnets and the extra pair of infeasible host IDs on each extra subnet.

(D: Bonus Problem) What is the expected value of the maximum backoff-time in CSMA/CD Ethernet? What is the conceptual basis for exponential backoff and why or why not is the scheme fair?

Solution *The Ethernet backoff scheme includes selection of a random number of slot times to wait before a retransmission attempt after a collision. The range of slot time increases exponentially after each collision of the same frame for 10 attempts. The backoff window then remains fixed for the next 5 attempts after which a failure is indicated if the transmission does not succeed, The largest backoff window size is this $2^{10} * \text{slot_time} = 1024 * 51.2\mu\text{s} = 52.4\text{ms}$. Since the actual backoff-time is chosen uniformly from the backoff window, the average maximum back will be on half the maximum backoff: 26.2ms.*

The conceptual basis is spread out the retransmission attempts in a random manner in order to avoid repeat collisions. The two goals are to respond rapidly—resend as soon as possible, but to avoid further collisions. Seeing repeat collisions indicates that with high probability there are multiple contenders and they need to be spread out more in time. This is the fundamental tradeoff. The scheme has something of a LIFO effect, thus, at first glance appears unfair as it is biased to favor faster retransmission for nodes that have not been contending as long. On a sample instance this is true, however, in the long run a node will be equally likely to find itself in the disadvantaged position of waiting longer or in the advantaged position of being the last node to join the contention and be involved in a collision. Thus, contrary to popular wisdom I contend the scheme will exhibit long-term fairness.

Problem Two: Go-Back-N ARQ Efficiency

Give an approximate analysis of the *effective-transmission-rate* for continuous-ARQ (go-back-n) algorithm ¹. Assume that data flow is in one direction only—from Node-A to Node-B. In the analysis do not put a fixed restriction on the number of error-ed frames or the number of times a give frame can be received in error. In your analysis you must account for the following system parameters for which details will be given as needed:

- The number of overhead bits associated with the DLC of each data frame in the forward direction.
- The length of the acknowledgment frames, both positive and negative, in the reverse direction.
- The processing times at the sender and the receiver.
- The propagation delay between the sender and the receiver.

Define the effective transmission rate in a similar manner to efficiency as follows:

$$R_e = \frac{\text{Total number of received bits accepted by DLC}}{\text{Elapsed time to transfer those bits and accept them.}}$$

Based on this definition it should be apparent that the link efficiency can be computed as follows, where R is the link's transmission rate in bits-per-second:

$$\eta = \frac{R_e}{R}$$

For a continuous-ARQ scheme without any transmission errors it is easy to show that the denominator in the definition of R_e is merely K/R which is the time between the acceptance of successive data blocks. (Convince yourself that this is the case). Given a more realistic scenario in which tranmission errors do occur, the problem is to compute the number of frames, indeed including the extra frames due to the go-back-n protocol, to get a frame accepted by the receiving DLC. Let N be the size of the transmission window, and P be the probability a frame is received in error. Assume that frame errors are independent and occur with probability P .

- Let $p = \text{bit error rate (BER)}$, the probability of a bit being in error for the link.
- Assume that bit errors are independent.
- Assume that the frame-length is fixed and equal to K bits inclusive of DLC overhead bits.

(A) The probability of N retransmissions is given by $P(1 - P)$. This can be seen by noting that one frame must be received correctly once to be accepted. The probability of a frame being received correctly is $(1 - P)$, which is independent of the probability of its rejection due to transmission error which is given by P . Retransmission of the errant frame will cause the retransmission of the subsequent $(N - 1)$ frames as well in a go-back-n ARQ protocol. If the frame is received in error a second time there will be $2N$ retransmission. This event occurs with probability $P^2(1 - P)$. Based on this line of reasoning, show that the expected number of **total transmissions** before an arbitrary frame is received correctly and accepted by the receiving DLC is given by the following expression:

$$E[\text{Number of transmissions}] = \frac{1 + (N - 1)P}{1 - P}$$

¹Be sure to read all the question *very* carefully, including what *isn't said directly* and consider all the assumptions.

Solution It is stated above that a frame is received without error with probability $(1-P)$, however with probability P it will incur an error. Based directly on the specific problem formulation stated in the previous paragraph: **Retransmission of the errant frame will cause the retransmission of the subsequent $(N-1)$ frames as well.** Following the same reasoning used in the analysis of the stop-and-wait algorithm and the statement above, independence leads to a probability of $P(1-P)$ that there will be N 'retransmissions,' and, hence, a total of $(N+1)$ transmissions to receive a frame. The statement further elaborates that **if the frame is received in error a second time there will be $2N$ retransmission. This event occurs with probability $P^2(1-P)$.** Hence, a total of $(2N+1)$ transmissions are needed to receive the frame with probability $P^2(1-P)$, which accounts for the two independent transmission failures and one success—the fate of the other $(N-1)$ frames is irrelevant in go-back- N .

Following the reasoning laid out it is easy to see that the probability of j retransmissions, or $(j+1)$ total transmissions is $P^j(1-P)$. The expected number of transmissions (per successful frame transmission) is thus:

$$\begin{aligned}
E[\text{Number of transmissions}] &= 1 + NP(1-P) + NP^2(1-P) + NP^3(1-P) + \dots \\
&= 1 + \sum_{j=1}^{\infty} NjP^j(1-P) \\
&= 1 + N(1-P) \sum_{j=1}^{\infty} jP^j \\
&= 1 + NP(1-P) \sum_{j=1}^{\infty} jP^{j-1} \\
&= 1 + NP(1-P) \sum_{j=1}^{\infty} \frac{d}{dj} P^j \\
&= 1 + NP(1-P) \frac{d}{dj} \left(\sum_{j=1}^{\infty} P^j \right) \\
&= 1 + NP(1-P) \frac{d}{dj} \left(\sum_{j=0}^{\infty} P^j - 1 \right) \\
&= 1 + NP(1-P) \frac{d}{dj} \left(\frac{1}{1-P} - \frac{1-P}{1-P} \right) \\
&= 1 + NP(1-P) \frac{d}{dj} \left(\frac{P}{1-P} \right) \\
&= 1 + NP(1-P) \frac{1}{1-P^2} \\
&= \frac{1-P}{1-P} + \frac{NP}{1-P} \\
&= \frac{1+(N-1)P}{1-P}
\end{aligned}$$

The previous expression provides only the mean number of total frame transmissions in order to receive a frame correctly; however, it fails to account for the additional overhead due to DLC and transmission, propagation and processing latency. Make the following additional assumptions:

- Assume that the error detecting algorithm is ideal in that it always catches frames that are in error, and never detects an error in an error-free frame.
- Assume that acknowledgment frames (both positive and negative) are very small relative to the data frames. Hence, given reasonable line error rates the *probability of receiving an ACK or NAK in error is negligible*.
- Let n_h represent a fixed number of DLC overhead bits in each K bit frame.
- Let n_a represent a fixed number of bits in both ACK and NAK frames, where, $n_a \ll K$.
- Assume there is a fixed processing delay equal to t_{ta} that effectively acts as a “line-turn-around” time in the stop-and-wait ARQ.
- Let t_p represent the propagation delay of the link.
- Let R be the transmission rate of the link in bits-per-second.

(B) Based on the given assumptions and system parameters, along with the mean number of total transmissions required to have a frame accepted, show that the following expression holds for R_e ²:

$$R_e = \frac{(1 - P)(K - n_h)R}{(1 + (N - 1)P)K}$$

Solution In the previous part we showed that the average number of frame transmissions required to correctly receive one frame is: $\frac{1+(N-1)P}{1-P}$. The total time spent is equal to the transmission time of a single frame times the number of frames: $\frac{1+(N-1)P}{1-P} \frac{K}{R}$. The effective transmission rate is equal to the amount of data received which is the frame length minus the overhead: $K - n_h$ divided by the time required:

$$R_e = \frac{K - n_h}{\frac{1+(N-1)P}{1-P} \frac{K}{R}} = \frac{(1 - P)(K - n_h)R}{(1 + (N - 1)P)K}$$

²Pay close attention to the total elapsed for the acceptance of the given number of bits.

Problem Three: Voice-over-IP

Consider a packet voice system similar to the one on the problem given in the midterm. The system consists of multiple *packet voice* (VoIP) channels that are multiplexed over a single T-1 line. To maintain voice quality it is required that the end-to-end delay be maintained ≤ 100 ms. We are interested in how many voice channels can be supported.

(A) The transmission rate of a T-1 line is 1.5444 Mbps; however there are two factors that limit the amount of usable bandwidth: (1) framing and (2) signaling overhead. As such, without *clear-channel* service the effective transmission rate is limited to 1.344 Mbps. Explain why only 7 of the 8 bits per time-slot can be used?

Solution *Clear-channel service removes the in-band signaling system that was developed to support E&M signals for TDM voice. The technique involved robbing the low-order bit of each time-slot in every sixth frame—a 'superframe' structure was created to enable synchronization of the frame number so the receiver would know which frames to find signalling information. Robbing the low order bit one every 750 μ s has no effect on voice quality but is ineffective for carrying data. Without clear-channel service we cannot predict what the system will do to these bits, hence, they are not used. To avoid the timing and synchronization complexity that would be needed to only use 7 bits on the 'robbed' frames the choice was to only allow user access the 7 higher order bits in each 'time-slot'. Note, however, that the time-slots are effectively concatenated into a continuous bit-stream for non-TDM applications. However, only 7 of every 8 bits are used. The aggregate rate is thus 7 bits/slot *24 slots/frame *8000 frames/sec which is 1.344 Mbps.*

(B) Current VoIP systems packetize voice in fixed 30 ms blocks. The packet length may vary depending on the encoding and compression schemes being used. Assume standard PCM is used to convert the voice from analog to digital and that no compression is being used. Assume the total protocol overhead per-packet (due to IP etc.) is 40 octets (bytes). What is the resulting fixed packet length L and how long does it take to *transmit* a single packet?

Solution *PCM Voice samples the signal once ever 125 μ s. Hence, 30 ms requires $\frac{30}{0.125} = 240$ 8 bit samples. To this is added the 40 bytes of overhead which comes to 280 bytes or $L = 2240$ bits. At 1.344 Mbps the total transmission time for one packet is $\frac{2240}{1344000} = 1.67$ ms.*

(C) In order to maximize the system's utilization the designers noted that by staggering the start-times for sampling each successive channel they had more control over the queuing delay. Specifically, if the system was not fully loaded and they staggered the timing by the (fixed) transmission time of the previous packet they could eliminate queuing delay altogether. Assuming a 250 μ s propagation delay and that processing delays are negligible what is the end-to-end delay for a single VoIP channel?

Solution *The end-to-end delay for each voice channel based on this strategy would include the following components: sampling/quantizing/packetization delay + transmission time + propagation delay. The waiting times for the individual samples balance out between the sending and receiving sides. For example, the first sample taken must wait for the next 239 samples (239 * 125 μ s) before it can be sent, whereas on the receive side it is played out first—taking 125 μ s. Hence, the total delay from sampling and playback is 30ms for regardless of the sample 'rank' in the packet. The total delay is thus:*

$$D = 30 + 1.67 + 0.250 = 31.92 \text{ ms}$$

(D) Based on the strategy outlined in part c above and your answer to the question, how many VoIP channels can be supported without any queuing delay?

Solution *The optimal staggering scheme to avoid any queuing delays requires one packet for each channel to be completed at the same instant the previous packet transmission has completed. Moreover, the system must come around again to pick up the subsequent packet from each channel within 30ms. Hence, the maximum number of channels that can be supported is the floor of the sampling/quantizing/packetization delay divided by the packet transmission time:*

$$N_{max} = \frac{30}{1.67} = 17$$

(E: Bonus Problem) Based on the strategy outlined in part c above and your answer to the question, what is the maximum number of VoIP channels can be supported without exceeding the voice quality constraint of 100 ms end-to-end delay? What will be the average queuing delay? Will the long term average delay differ among the channels or will it be the same for all channels?

Solution *In the previous problem we were asked to stagger the channel sampling times in order to maximize the number of channels supported subject to the constraint that the queuing delay is zero for all packets. In this problem the constraint is on end-to-end delay. However, the only component of delay that is not fixed is the queuing delay. The first step, therefore, is to transform the constraint into a maximum acceptable delay in the queue. The delay without waiting in queue is 31.92 ms; the max queuing delay $W_q = 100 - 31.92 = 68.08$ ms. The next step is to determine the equivalent queue length $N_q = \frac{68.08}{1.67} = 40$. Hence, we can support $17 + 40 = 57$ channels. Both the packet interarrival times and packet service times are deterministic and equal (1.67ms), thus,*

*the utilization is 100%. Because the times are fixed and the arrival rate **never** exceeds the service rate the system is stable. However, it builds up to a steady-state and fixed queue length. The initial packets from the first 17 channels pass through the system with no delay in the queue. However, rather than service the second packet from the first channel 40 additional channels are serviced—again, due to the staggered sampling they will all pass through the system without any queuing delay. However, by the time the 57th channel's packet completes its transmission a queue will have built up to 40 packets. From this point forward one packet will arrive for every packet that is transmitted. Furthermore, every packet will wait in the queue precisely $40 * 1.67 = 66.8$ ms. Thus, the average is the same as the fixed delay $W_q = 66.8$ ms for all channels. The total end-to-end delay for each channel will be $66.8 + 31.92 = 98.72 < 100$ ms. If an additional channel were added the delay would increase to 100.39ms which exceeds the constraint.*

Problem Four: Wireless LANs

(A) Identify and briefly explain three reasons why CSMA/CD cannot be used for a wireless LAN? Hints: Is there some difference between guided and unguided media? Do the relative positions of the nodes (topography) cause problems—if so what?

Solution

1. Collisions occur at the receiver and do not propagate back to the sender, hence, depending on the relative location(s) of the interfering transmitter(s) it may not be possible to even detect a collision at the sender.
2. The transceiver would typically be saturated by its own transmission, hence, could not reliably sense a local collision if it is transmitting unless it has a second transceiver dedicated to that task, which is not an economical solution at this time.
3. A station may transmit to a station that is hidden with respect to another node involved with a transmission that causes a collision at the sender. The collision at the sender has no effect on the received signal, hence, even if it were possible to detect the collision it is not destructive interference.

(B) In the lecture and the book an analysis is presented for the maximum capacity of ALOHA and SLOTTED ALOHA. Justify the use of the Poisson arrival process in the analysis? Illustrate a time line with Poisson packet arrivals—identify the inter-arrival times and what the distribution of inter-arrival times is?

Solution It was explained on several occasions that the Poisson process is often used to model the packet arrival process in a data network because it possesses from very useful mathematical and probabilistic properties that make it possible to find closed-form solutions to many interesting performance problems. However, assuming Poisson without any justification is poor science. Using arrivals to a switch I showed that when a large number of independent random arrival processes are merged in a network the aggregate process can be modeled reasonably well as a Poisson process. This is the Kleinrock Independence Assumption. By modeling a large number of independent sources in the Aloha network the same aggregation effect is achieved. The interarrival times are independent and exponentially distributed—the interarrival time is the time from the arrival instant of packet k to the arrival instant packet $k + 1$.

(C) In IEEE 802.11 there are a number of inter-frame-space (IFS) timers. Rank the following timers: DIFS, SIFS and EIFS in terms of their relative durations and illustrate by example how they would be use in the RTS/CTS + ACK scheme?

Solution $SIFS < DIFS < EIFS$; DIFS is the distributed IFS and is used by stations that have sensed the network idle and have a frame ready to send. Since stations sense the medium idle at different times due to propagation delays, hidden terminals, deferral and random backoff times, the DIFS gives a chance to sense a transmission that may have been initiated already but has not been sensed yet. This helps to avoid excess collisions. SIFS is the short IFS and is used to provide a higher priority to the sender. This is necessary when sending CTS frames, data frames after receiving a CTS and ACK frames. They Since all these frames are part of an ongoing exchange it is desirable to complete that 'transaction' without a collision. SIFS gives an advantage to the senders of those frames. EIFS is the extended IFS and is the longest timer. It is used after a station that has data to send detects a collision. The purpose is to give the station sending the frame involved in the collision a higher priority in its re-transmission. The idea is that seeing the collision implies there is higher probability that there will be a re-transmission. EIFS reduces the probability of causing yet another collision.

(D: Bonus Problem) Why do the control packets (RTS,CTS,ACK) and IFS contribute a larger ratio of overhead as the data rate increases under 802.11 b?

Solution *To allow backward compatibility with older devices that do not support higher speeds, and, more importantly to enable transmission rate adaptation based on signal strength and quality it is possible that even a small LAN will have devices transmitting data frames at different rates: 1,2,5 or 11 Mbps at the same time. Inter-operability requires that all stations can interpret the control frames and use the same timers. Hence, control frames are always transmitted at 1 Mbps and the timers are fixed. Furthermore, the NAB must specify the deferral in terms of time rather than frame length. This could present problems if a node uses an NAB based on a higher speed but can no longer transmit reliably at that rate. Clearly, since the control signals are all sent at the lowest transmission rate they represent an 11 fold increase in relative overhead when transmitting at 11 Mbps.*