Networked Systems, COMPGZ01, 2012
Answer TWO questions from Part ONE on the answer booklet containing lined writing paper, and answer ALL questions in Part TWO on the multiple-choice question answer sheet.
Marks for each part of each question are indicated in square brackets
Calculators are permitted

Part ONE

1. **Error-Control Coding and Framing with DNA**

   In this question we'll explore the error correcting code and framing mechanism that cells use to synthesize amino acids from DNA, the molecule that carries information in most living things. In order to focus on the information-theoretic aspects, we'll slightly simplify our description.

   A strand of DNA is made of an ordered sequence of *nucleotides*. A nucleotide contains one of four possible *bases*: *cytosine* (*C*), *guanine* (*G*), *adenine* (*A*), and *thymine* (*T*). Consequently, each nucleotide conveys **information**.

   a. Assuming each of the four bases occurs with equal probability, how many bits of information does a nucleotide contain?

   [4 marks]

   A strand of DNA is turned into a sequence of *amino acids* (which in turn are the building blocks of proteins) through the process of *synthesis*, which we focus on here. In synthesis, groups of three nucleotides (called *codons*) are mapped to one of 20 possible amino acids according to Table 1.

   b. What is the rate of this code?

   [6 marks]

   To understand the error correcting and framing properties of this code, let's define the *nucleotide error rate* (NER) as the probability that any **single** nucleotide is errored. Assume that when a nucleotide error occurs, the nucleotide in question changes to one of the other three nucleotides at random and with equal probability of changing to each.

   For the following questions, assume an NER of 10%.

   c. What is the probability that a codon coding for the amino acid Thr has at least one nucleotide error?

   [3 marks]

   d. What is the probability that a codon coding for the amino acid Thr is synthesised to a different amino acid or (stop)?

   [6 marks]

First nucleotide base of DNA codon	Second nucleotide base of DNA codon			
	T	**C**	**A**	**G**
T	TTT = Phe TTC = Phe TTA = Leu TTG = Leu	TC– = Ser	TAT = Tyr TAC = Tyr TAA = (stop) TAG = (stop)	TGT = Cys TGC = Cys TGA = (stop) TGG = Trp
C	CT– = Leu	CC– = Pro	CAT = His CAC = His CAA = Gln CAG = Gln	CG– = Arg
A	ATT = Ile ATC = Ile ATA = Ile ATG = Met	AC– = Thr	AAT = Asn AAC = Asn AAA = Lys AAG = Lys	AGT = Ser AGC = Ser AGA = Arg AGG = Arg
G	GT– = Val	GC– = Ala	GAT = Asp GAC = Asp GAA = Glu GAG = Glu	CG– = Gly

Table 1: Codon to three-letter amino acid abbreviation mapping for the DNA code. An en-dash (–) indicates any nucleotide base. (stop) indicates a codon that frames the end of the protein instead of translating to an amino acid.

e. What is the probability that the codon GAT coding for the amino acid Asp is synthesised to a different amino acid or (stop)?

[6 marks]

f. Which is/are the codon(s) that is/are most likely to be mistaken for (stop) and what is the likelihood of those codons being confused for (stop)? How does the framing error you have just computed compare with the NER?

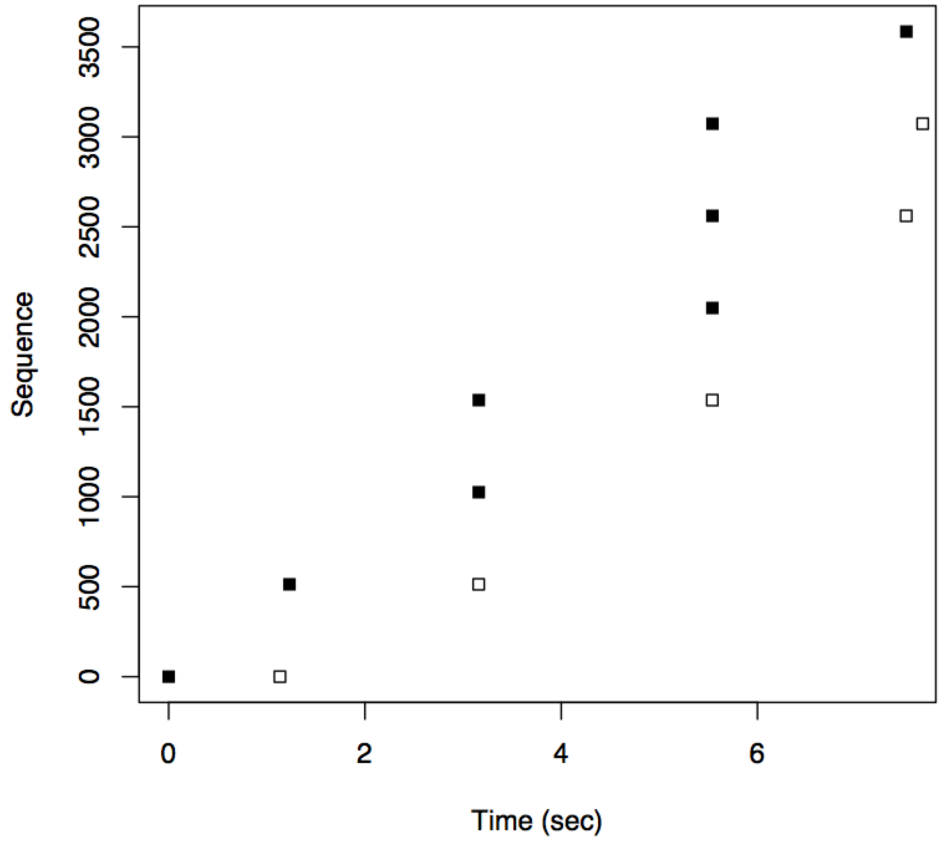
[8 marks]

[Question 1: Total 33 marks]

2. **TCP and Time-Sequence Plots**

Below follows a series of Transmission Control Protocol (TCP) time-sequence plots *taken at the TCP sender*, as discussed in lecture. In the plots, data packets are shown as solid rectangles and ACKs are shown as hollow rectangles. Answer the following questions about these plots.

a. Time-sequence plot A:



i. In plot A above, what is the approximate packet size used by the sender?

[3 marks]

ii. In plot A above, what is the approximate round-trip time (RTT) for the first exchange in the 3-way TCP handshake?

[3 marks]

iii. In plot A above, what is the approximate RTT for data packets once the connection is established?

[3 marks]

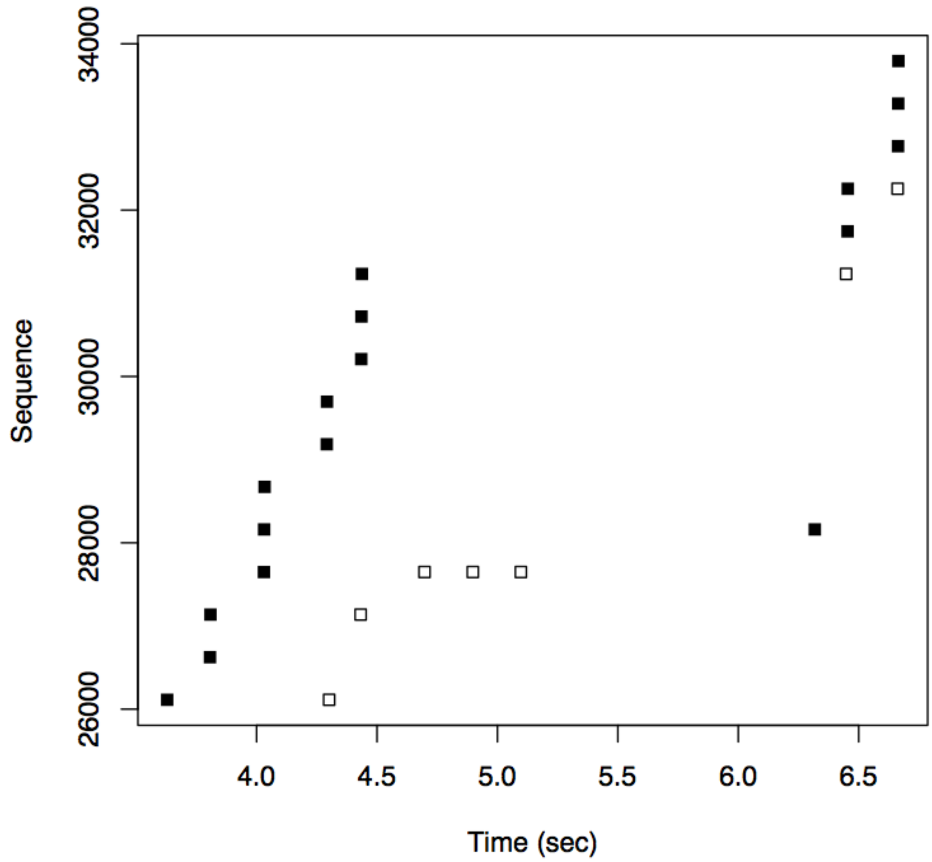
iv. In plot A above, what, *measured in number of packets,* is the value of the congestion window just before time = 6 seconds?

[3 marks]

v. In plot A above, does the receiver use delayed ACKs? Justify your answer with evidence from the plot.

[3 marks]

b. Time-sequence plot B:



i. In plot B above (again, recorded at the sender), what, *measured in number of packets,* is the value of the congestion window after the arrival of the ACK recorded just before time = 4.5 seconds?

[3 marks]

ii. In plot B above, there is a retransmitted packet. What is the approximate highest sequence number contained in the retransmitted packet?

[3 marks]

iii. For the same retransmitted packet in plot B above, was this packet retransmitted using a retransmit timeout or fast retransmit? Give specific evidence from the plot to support your answer.

[3 marks]

iv. For the same retransmitted packet in plot B above, was only that one packet lost, or were others in flight at the same time also lost? Give specific evidence from the plot to support your answer.

[3 marks]

v. After the same retransmitted packet in plot B above has been acknowledged, what, in *number of packets* is the congestion window size?

[3 marks]

vi. Assuming no further packet loss after the retransmission in plot B above, how many round-trip times will it take for the sender's congestion window size to return to its value before the retransmission? Justify your answer by referring to relevant details of the TCP protocol.

[3 marks]

[Question 2: Total 33 marks]

3. **Domain Name System (DNS), Firewalls, and Tunneling**

   John Henry, your classmate from 3035/GZ01, doesn't like being without Internet access while he travels, yet doesn't like paying WiFi providers, such as those found in airports and cafes. He wants to design a scheme that will let him obtain Internet access for free through an airport's pay-for-access WiFi service.

   Typical pay-for-service WiFi systems initially firewall clients on the WiFi network from the Internet (blocking outbound and inbound TCP traffic from/to that client). They only remove the firewalling for a client if the client *pays* for Internet access. These pay-for-service WiFi systems place a local caching DNS server on the same WiFi LAN as the client (on the same side of the firewall as the client). When a non-paying client looks up a DNS name for a web server elsewhere on the Internet (*e.g.,* `www.cnn.com`), the local caching DNS server returns the *correct* IP address for `www.cnn.com`. But the firewall then intercepts the client's HTTP GET request for `http://www.cnn.com/` and redirects the client (using an HTTP redirect) to a payment web server run by the WiFi service provider, which sends the client a page asking for payment for Internet service. If the client pays (typically by entering a credit card number into a form on this page), the payment web server reconfigures the firewall to allow TCP traffic between the client and the Internet thereafter, and to stop responding to the client's HTTP requests with redirects to the payment web server.

   While sitting in a cafe with pay-for-service WiFi, John opens his laptop and enables WiFi. He finds that his laptop associates with a WiFi access point and successfully obtains IPv4 address 10.1.1.37 (with a netmask of 255.255.255.0) through the Dynamic Host Configuration Protocol (DHCP). He sees that as part of the DHCP configuration provided by the network, his laptop has begun using the local caching Domain Name System (DNS) server with IPv4 address 10.1.1.10.

   Throughout this question, assume that all DNS traffic is sent using UDP.

   a. John's initial idea is that the firewall between the WiFi LAN and the Internet may block outbound and inbound TCP traffic for clients who have not paid, but may still allow clients on the WiFi LAN who haven't paid to send DNS queries *directly* to the Internet (*i.e.,* not through the local caching DNS server, 10.1.1.10), and receive responses to them. John finds that DNS queries from his laptop to the local caching DNS server work properly—that is, for every valid Internet hostname he looks up with `dig` via the local caching nameserver, he receives a correct DNS response.

      i. How can John use `dig` to determine whether the firewall allows non-paying clients to send DNS queries *directly* to the Internet from the WiFi LAN and receive replies? Be specific in your description of any features of `dig` John must use.

      [3 marks]

Suppose that John determines using `dig` that the firewall indeed *does* allow non-paying clients to send DNS queries directly to the Internet, and receive replies to them. John excitedly hypothesizes that to obtain free Internet access through this WiFi service, he can simply tunnel IP within UDP to get through the firewall. Before leaving home, John wrote a UDP tunnel endpoint server and left it running on a host elsewhere on the Internet. John hopes that the firewall only inspects the UDP header, and not the payload of the UDP packets.

ii. Assuming the firewall doesn't inspect the payload of these UDP packets, but only their headers, what header values will outbound and inbound UDP packets need to have in order to appear to the firewall as DNS-like traffic?

[2 marks]

iii. What does it mean to tunnel IP within UDP? Describe specifically what actions John's laptop would need to take to tunnel IP within UDP, and what actions the UDP tunnel endpoint server John runs elsewhere on the Internet would need to take to serve as the remote endpoint of that tunnel.

[5 marks]

b. Suppose that when John attempts to tunnel IP over UDP, he finds that the firewall does *not* allow these packets through: neither outbound nor inbound. Yet tests with `dig` reveal that the firewall *does* consistently allow through DNS queries and responses sent directly to/from all nameservers John tries elsewhere on the Internet.

i. Based on the behavior John observes in this part of this question, what *additional* step(s) apart from looking at the header is the firewall most likely using to decide whether to block or forward UDP packets?

[3 marks]

ii. Despite the firewall's blocking of "naive" IP-over-UDP tunneling, based on his observation that he can directly reach Internet name servers and receive replies from them, John believes he can tunnel IP over *actual DNS query and reply packets* (sent over UDP). To do so, the tunneling software on his laptop and tunnel endpoint server he runs elsewhere on the Internet will need to send correctly formatted DNS queries and correctly formatted DNS replies, respectively. Based on your knowledge of the DNS protocol, describe roughly how John's laptop can encode IP packets to be sent to the Internet as DNS queries, and how the the tunnel endpoint server can encode IP packets to be sent to John's laptop as DNS responses. In your answer, describe how to handle the following challenges:

- Hostnames in DNS queries may only contain letters in the Roman alphabet, numerals, and dashes. (Take this as an assumption.)
- DNS packets (both queries and replies) are limited to 512 bytes of UDP payload, but IP packets may be up to the link Maximum Transfer Unit (typically 1500 bytes) in size.
- DNS queries have a maximum hostname length of 255 characters.
- DNS queries can only include a single question.

[9 marks]

iii. In which direction, laptop-to-Internet, or Internet-to-laptop, do you expect the achievable throughput to be greater, and why? Assume that the DNS tunneling system only has one packet in flight at a time in each direction.

[3 marks]

iv. An address (A) RR contains only 4 bytes of IPv4 address, making it an inefficient encoding of IP packet data from the tunnel endpoint server to John's laptop (even when multiple A records are included in a reply). How can the tunnel endpoint server more efficiently encode replies to A record queries from John's laptop?

[3 marks]

c. Now suppose that when John investigates with `dig`, he determines that the firewall does *not* allow non-paying clients to send *any* UDP traffic *directly* to the Internet—not even DNS queries and responses. Yet John's laptop's DNS queries via the local caching nameserver all complete correctly.

What additional mechanism(s) will John need in place for his DNS tunneling system to work correctly in this setting?

[5 marks]

[Question 3: Total 33 marks]