Question 1 (25 points): Comparing persistent vs. non-persistent HTTP

Suppose your browser downloads a webpage. The webpage html object is 7 Kbits in length and additionally contains 5 embedded images, each 2.5 Kbits in length. The webpage and the 5 images are all stored on the same server. We will abstract the network path between your browser and the web server as a 50 Mbps link. Assume your browser has a 100 ms RTT to the server, which includes propagation, processing, and queuing delays. Assume also that the time to transmit a GET message is zero. You should, however, account for the time needed to setup up a TCP connection (i.e, 1 RTT) and the time to transmit the html file and the embedded images.

(a) 5 points. Assume one non-persistent HTTP connection. What is the response time? I.e., the time from when the user requests the webpage to the time when the webpage and its embedded images are displayed? Make sure you describe the various components that contribute to the response time.

Solution:

First, compute some useful quantities.

$$\begin{aligned} d_{trans}^{(w)} &= \text{Webpage transmission delay} \\ &= \text{7Kb/50Mbps} \\ &= 0.14 \text{ ms} \end{aligned}$$

$$\begin{aligned} d_{trans}^{(i)} &= \text{Image transmission delay} \\ &= \text{2.5Kb/50Mbps} \\ &= 0.05 \text{ ms} \end{aligned}$$

$$\begin{aligned} d_{tcp} &= \text{TCP connection setup delay} \\ &= 100 \text{ ms} \end{aligned}$$

$$\begin{aligned} d_{RTT} &= \text{Time to send HTTP Request and receive HTTP Response} \\ &= 100 \text{ ms} \end{aligned}$$

Then we compute the response time as the sum of the time to get the webpage html object and the time to get each of the images.

$$\begin{aligned} \text{Response Time} &= d_{tcp} + d_{trans}^{(w)} + d_{RTT} + 5[d_{tcp} + d_{trans}^{(i)} + d_{RTT}] \\ &= 100 \text{ ms} + 0.14 \text{ ms} + 100 \text{ ms} + 5[100 \text{ ms} + 0.05 \text{ ms} + 100 \text{ ms}] \\ &= 1200.39 \text{ ms} \end{aligned}$$

(b) 5 points. Assume non-persistent HTTP and that the browser can open two parallel TCP connections to the server. What is the response time?

Solution:

We have a total of six objects to get, which we can get in parallel over two connections. In order to know that we need to get the images, we must first get the HTML object. Once we have the HTML object, we can split getting the images over two connections: two images on one connection and three images on the other. The connection getting three images will take longer, so that is the connection delay we use in the equation.

$$\begin{aligned}\text{Response Time} &= d_{tcp} + d_{trans}^{(w)} + d_{RTT} + 3[d_{tcp} + d_{trans}^{(i)} + d_{RTT}] \\ &= 100 \text{ ms} + 0.14 \text{ ms} + 100 \text{ ms} + 3[100 \text{ ms} + 0.05 \text{ ms} + 100 \text{ ms}] \\ &= 800.29 \text{ ms}\end{aligned}$$

(c) 5 points. Assume one persistent HTTP connection with no pipelining. What is the response time?

Solution:

If there is no pipelining, this means we must wait the full RTT to receive an object before sending a request for another object. Since we have a persistent HTTP connection, this means we only need to set up one TCP connection.

$$\begin{aligned}\text{Response Time} &= d_{tcp} + d_{trans}^{(w)} + d_{RTT} + 5[d_{trans}^{(i)} + d_{RTT}] \\ &= 100 \text{ ms} + 0.14 \text{ ms} + 100 \text{ ms} + 5[0.05 \text{ ms} + 100 \text{ ms}] \\ &= 700.39 \text{ ms}\end{aligned}$$

(d) 5 points. Assume one persistent HTTP connection with pipelining. What is the response time?

Solution:

With pipelining, we can send requests for all of the objects without waiting a full RTT to receive an object before sending the next request. However, we must first receive the HTML webpage before requesting any other objects.

$$\begin{aligned}\text{Response Time} &= d_{tcp} + d_{trans}^{(w)} + d_{RTT} + 5d_{trans}^{(i)} + d_{RTT} \\ &= 100 \text{ ms} + 0.14 \text{ ms} + 100 \text{ ms} + 5[0.05 \text{ ms}] + 100 \text{ ms} \\ &= 300.39 \text{ ms}\end{aligned}$$

(e) 5 points. Are non-persistent HTTP connections ever preferable to persistent HTTP connections?

Solution:

From the user perspective, non-persistent HTTP connections are never preferable since they result in larger delays than do persistent HTTP connections. This assumes that both non-persistent and persistent connections are allowed parallel connections.

Question 2 (15 points): DNS and `dig`

This question will use the command-line tool `dig` to explore the DNS hierarchy of servers. To learn more about how `dig` works, type the command `man dig` in a terminal. Starting with a root DNS servers (e.g., choose one from `https://www.iana.org/domains/root/servers`), you will initiate a sequence of non-recursive DNS look-ups to resolve a hostname using dig. To help you get started, an example first command might be the following, using `a.root-servers.net` as the name server. Note the use of the `@` sign in the command.

```
dig +norecurse @a.root-servers.net any www.wesleyan.edu
```

From this command you'll get a list of next name servers. Choose one of the name servers and execute another `dig` command. Repeat this process until the address is resolved.

(a) 10 points. Give the list of all name servers you use to resolve `www.wesleyan.edu`. List the name servers in the order that you accessed them.

Solution:

The following name servers were used to resolve `www.wesleyan.edu`.

```
a.root-servers.net
f.edu-servers.net
ns3.wesleyan.edu
```

First command: `dig +norecurse @a.root-servers.net any www.wesleyan.edu`

First response contained the following name servers:

```
;; AUTHORITY SECTION:
edu. 172800 IN NS f.edu-servers.net.
edu. 172800 IN NS a.edu-servers.net.
edu. 172800 IN NS g.edu-servers.net.
edu. 172800 IN NS l.edu-servers.net.
edu. 172800 IN NS c.edu-servers.net.
edu. 172800 IN NS d.edu-servers.net.
```

Second command: `dig +norecurse @f.edu-servers.net any www.wesleyan.edu`

Second response contained the following name servers:

```
;; AUTHORITY SECTION:
wesleyan.edu. 172800 IN NS ns3.wesleyan.edu.
wesleyan.edu. 172800 IN NS ns4.wesleyan.edu.
wesleyan.edu. 172800 IN NS planc.wesleyan.edu.
```

Third command: `dig +norecurse @ns3.wesleyan.edu any www.wesleyan.edu`

Third response contained the IP address for `www.wesleyan.edu`.

```
;; ANSWER SECTION:
www.wesleyan.edu. 300 IN A 129.133.7.68
```

(b) 5 points. Give the list of all name servers you use to resolve the host `www.csail.mit.edu`. List the name servers in the order that you accessed them.

Solution:

```
dig +norecurse @a.root-servers.net any csail.mit.edu
dig +norecurse @f.edu-servers.net any csail.mit.edu
dig +norecurse @usw2.akam.net any csail.mit.edu
dig +norecurse @auth-ns3.csail.mit.edu any csail.mit.edu
csail.mit.edu. 14400 IN A 128.30.2.121
```

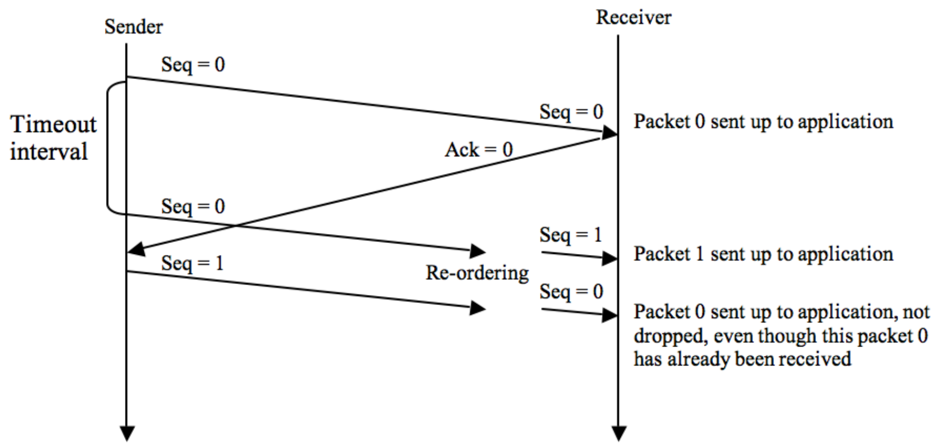

Figure 1: Question 3(a) solution: packet is delivered to the application layer twice.

Question 3 (20 points): Stop-and-wait Protocol

Suppose the channel can reorder packets: i.e., if the sender sends packet i followed by packet j, packet j may arrive before packet i. We know that the stop-and-wait protocol uses a window size of 1. Now assume there are only 2 sequence numbers, 0 and 1. Show using a timeline that the final stop-and-wait protocol we discussed (and analyzed) in class can result in each of the following two scenarios.

(a) 10 points. A packet is delivered to the receiver-side application layer twice.

Solution: See Figure 1.

(b) 10 points. A packet is never delivered to the receiver-side application.

Solution: See Figure 2.

Question 4 (15 points): Selective repeat and Go-Back-N

Answer true or false to the following questions and briefly justify your answer. Assume a window size of three.

(a) 10 points. For the selective repeat protocol, is it possible for the sender to receive an ACK for a packet that falls outside of its current window?

Solution:

True. The intuition is that if the packet sent wasn't lost and the sender times out too early and retransmits then the sender will receive two acks for the same packet sequence number. When the first ack is received the window will move forward. When the second ack is received, it will be outside of the window.

More carefully: suppose the sender has a window size of 3 and sends packets 1, 2, 3 at time t_0. At time t_1, where $t_1 > t_0$, the receiver ACKS 1, 2, 3. At time t_2, where $t_2 > t_1$ the sender times out and resends 1, 2, 3. At time t_3 the receiver receives the duplicates and re-acknowledges 1, 2, 3. At time t_4 the sender receives the ACKs that the receiver sent at time t_1 and advances its window to 4, 5, 6. At time t_5 the sender receives the ACKs 1, 2, 3 the receiver sent at time t_2 . These ACKs are outside of its window.

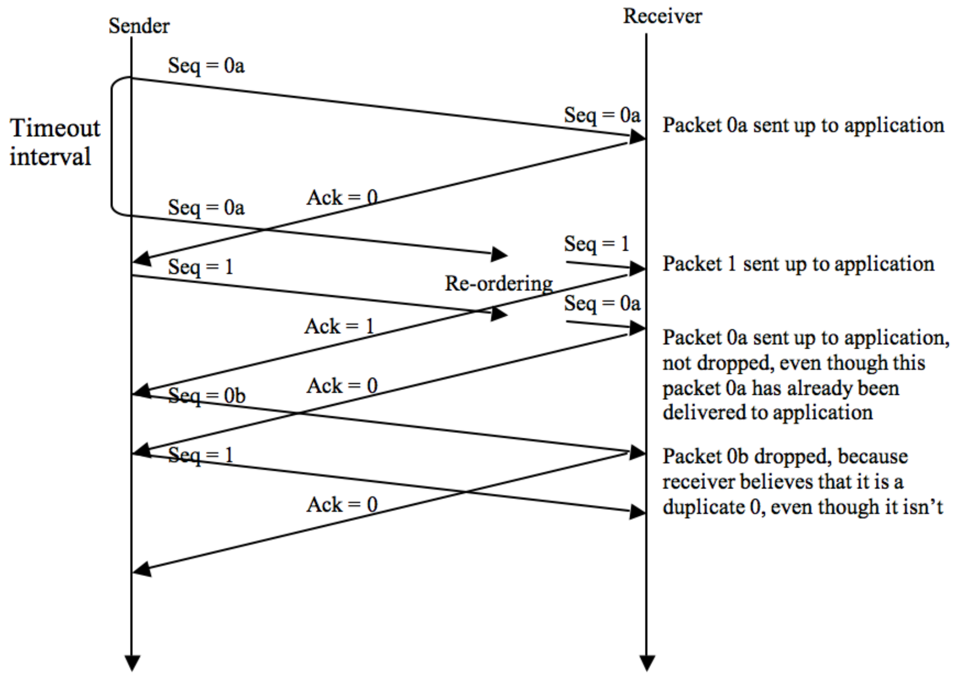

Figure 2: Question 3(b) solution: packet is never delivered to the receiver-side application (packet 0b).

(b) 5 points. For the Go-Back-N protocol, is it possible for the sender to receive an ACK for a packet that falls outside of its current window?

Solution:

True. By essentially the same scenario as in (a).

Question 5 (25 points): TCP sequence numbers and acknowledgements

Hosts A and B are communicating over a TCP connection. Host B has already received from A all bytes up through byte 248. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 40 and 60 bytes of data respectively. In the first segment, the sequence number is 249, the source port number is 503, and the destination port number is 80. Host B sends an acknowledgement whenever it receives a segment from Host A.

(a) 5 points. In the second segment sent from Host A to Host B, what are the sequence number, source port number, and destination port number?

Solution:

The sequence number is 289, the source port number is 503 and the destination port number is 80.

(b) 5 points. Suppose the first segment arrives before the second segment. Host B sends an acknowledgement in response to the first segment: what is the acknowledgment number, the source port number, and the destination port number?

Solution:

The acknowledgement number is 289, the source port number is 80 and the destination port number is 503.

(c) 5 points. Suppose the second segment arrives before the first segment. Host B sends an acknowledgement in response to the first segment. What is the acknowledgement number?

Solution:

The acknowledgement number is 249, indicating that Host B is still waiting for bytes 249 and onwards.

(d) 10 points. Suppose the two segments sent by A arrive in order at B. The first acknowledgement is lost and the second acknowledgement arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgements sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes of data. For each acknowledgement that you add, provide the acknowledgement number.

Solution: See Figure 3.

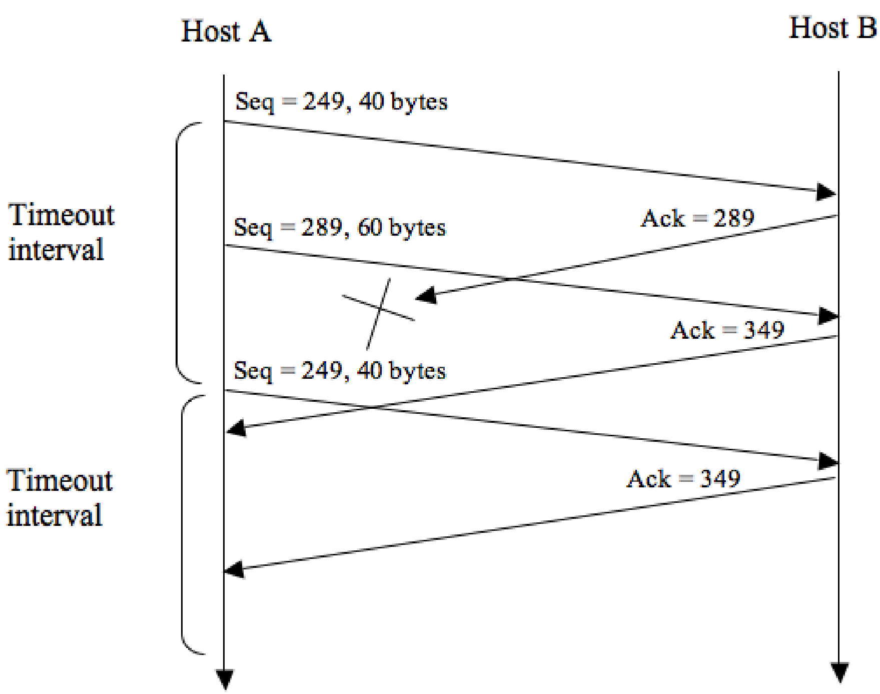

Figure 3: Question 5(d) solution.

Extra credit (15 points): TCP estimation of RTT

Consider the TCP procedure for estimating RTT. Suppose that $\alpha = 0.1$. Let r_1 be the most recent sample RTT, let r_2 be the 2nd most recent sample RTT, and so on.

(a) 5 points. Suppose four acknowledgements have been returned for a TCP connection, with the following sample RTTs: r_4, r_3, r_2, and r_1. Express the estimated RTT in terms of the four sample RTTs.

Solution:

Let $RTT^{(n)}$ be the estimate after the nth sample.

$$RTT^{(1)} = r_1$$

$$\begin{aligned}
RTT^{(2)} &= \alpha r_1 + (1-\alpha)RTT^{(1)} \\
&= \alpha r_1 + (1-\alpha)r_2
\end{aligned}$$

$$\begin{aligned}
RTT^{(3)} &= \alpha r_1 + (1-\alpha)RTT^{(2)} \\
&= \alpha r_1 + (1-\alpha)[\alpha r_2 + (1-\alpha)r_3] \\
&= \alpha r_1 + (1-\alpha)\alpha r_2 + (1-\alpha)^2 r_3
\end{aligned}$$

$$\begin{aligned}
RTT^{(4)} &= \alpha r_1 + (1-\alpha)RTT^{(3)} \\
&= \alpha r_1 + (1-\alpha)[\alpha r_2 + (1-\alpha)\alpha r_3 + (1-\alpha)^2 r_4] \\
&= \alpha r_1 + (1-\alpha)\alpha r_2 + (1-\alpha)^2 \alpha r_3 + (1-\alpha)^3 r_4
\end{aligned}$$

(b) 5 points. Generalize your formula for n sample RTTs.

Solution:

$$RTT^{(n)} = (1-\alpha)^{n-1} r_n + \sum_{i=1}^{n-1} \alpha(1-\alpha)^{i-1} r_i$$

(c) 5 points. For the formula in part (b), let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.

Solution:

$$\begin{aligned}
RTT^{(\infty)} &= (1-\alpha)^{\infty} r_{\infty} + \sum_{i=1}^{\infty} \alpha(1-\alpha)^{i-1} r_i \\
&= 0 + \sum_{i=1}^{\infty} \alpha(1-\alpha)^{i-1} r_i \\
&= \frac{\alpha}{1-\alpha} \sum_{i=1}^{\infty} (1-\alpha)^i r_i \\
&= \frac{0.1}{0.9} \sum_{i=1}^{\infty} (0.9)^i r_i
\end{aligned}$$

The weight given to past samples decays exponentially.