Problem 1

Tow hosts, A and B are separated by 30,000 kilometers and are connected by a direct link of $R = 2.5Mbps$. Suppose the propagation speed over the link is $2.5 * 10^8 meters/sec$.

1. Consider sending a file of 800,000 bits from Host A to Host B. Suppose the file is sent continuously as one large message. What is the maximum number of bits that will be in the link at any given time?
2. How long does it take to send the file, assuming it is sent continuously?
3. Suppose now the file is broken up into 20 packets with each packet containing 40,000 bits. Suppose that each packet is acknowledged by the receiver and the transmission time of an acknowledgment packet is negligible. Finally, assume that the sender cannot send a packet until the preceding one is acknowledged. How long does it take to send the file?

1. $P = \frac{3*10^8 meters}{2.5*10^8 meters/sec} = 1.2sec$

   $R * P = 2.5 * 10^6 bits/sec * 1.2sec = 3 * 10^6 bits$

   The maximum number of bits that will be in the link at any given time is 800,000, since $8*10^5 < 3*10^6$

2. $T = \frac{8*10^5 bits}{2.5*10^6 bits/sec} = 3.2 * 10^{-1} sec = 320msec$

   $D = P + T = 1.2 * 10^3 msec + 320msec = 1520msec$

   It will take 1.52 seconds to send.

3. $T_{packet} = \frac{4*10^4 bits}{2.5*10^6 bits/sec} = 1.6 * 10^{-2} sec = 16msec$

   $D_{packet} = T_{packet} + P_{packet} + P_{ack} = 16msec + 120msec + 120msec = 156msec$

   $D_{total} = 20 * D_{packet} = 3120msec$

   It will take 3.12 seconds to send.

Problem 2

Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is cached in your local host, so a DNS look-up is not needed. Suppose that the Web page associated with the link is a small HTML file, consisting only of references to 80 very small objects on the same server. Let RTT_0 denote the RTT between the local host and the server containing the object. How much time elapses (in terms of RTT_0) from when you click on the link until your host receives all of the objects, if you are using:

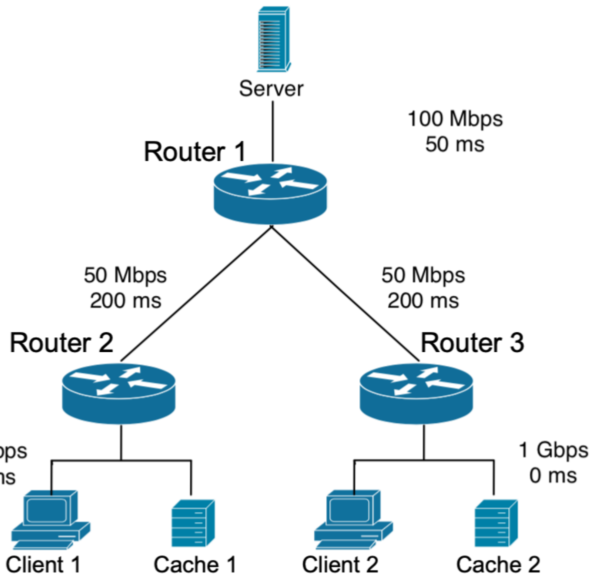
1. HTTP/1.0 without parallel TCP connections?
2. HTTP/1.0 with parallel TCP connections? The number of parallel connections is 20.
3. HTTP/1.1 without parallel connections, but with pipelining?

Ignore any processing, transmission, or queuing delays in your calculation.

1. $T_1 = T_{html} + 80 * T_{obj} = 2RTT_0 + 80 * 2RTT_0 = 162RTT_0$
2. $T_2 = T_{html} + \frac{80*T_{obj}}{20} = 2RTT_0 + 4RTT_0 = 6RTT_0$
3. $T_3 = T_{html} + T_{allobj} = 2RTT_0 + RTT_0 = 3RTT_0$

Problem 3

Consider the scenario shown in Figure below in which a server is connected to the Router1 by a 100 Mbps link with a 50 ms propagation delay. Initially Router1 is also connected to two routers (Router2 and Router3), each over a 50 Mbps link with a 200 ms propagation delay. A 1 Gbps link connects a host and a cache (if present) to each of these routers and we assume that this link has 0 propagation delay. All packets in the network are 20,000 bits long.



1. What is the end-to-end delay from when a packet is transmitted by the server to when it is received by Client1? In this case, we assume there are no caches, there's no queuing delay at the routers, and the packet processing delays at routers and nodes are all 0.
2. Here we assume that client hosts send requests for files directly to the server (caches are not used or off in this case). What is the maximum rate at which the server can deliver data to Client1 if we assume Client2 is not making requests?
3. Now we assume that the caches are ON and behave like HTTP caches. Again, Client2 is not active in this problem. A client's HTTP GET is always first directed to its local cache. 65% of the requests can be satisfied by the local cache. What is the average rate at which Client1 can receive data in this case?
4. Now clients in both LANs are active and the both caches are on. 65% of the requests can be satisfied by the local caches. What is the average rate at which each client can receive data?

Problem 3 continued on next page. . .

1. $D_{total} = D_{Server} + D_{R1} + D_{R2}$
   $D_{total} = (\frac{2*10^4 bits}{1*10^5 bits/msec} + 50) + (\frac{2*10^4 bits}{5*10^4 bits/msec} + 200) + (\frac{2*10^4 bits}{1*10^6 bits/msec} + 0)$
   $D_{total} = 0.2msec + 50msec + 0.4msec + 200msec + .02msec = 250.62msec$

2. We are limited by the speed of the bottleneck link, so the maximum rate at which the server can deliver data to Client1 is 50 Mbps.

3. $U_{avg} = 0.35 * 50Mbps + 0.65 * 1000Mbps$
   $U_{avg} = 17Mbps + 650Mbps = 667Mbps$

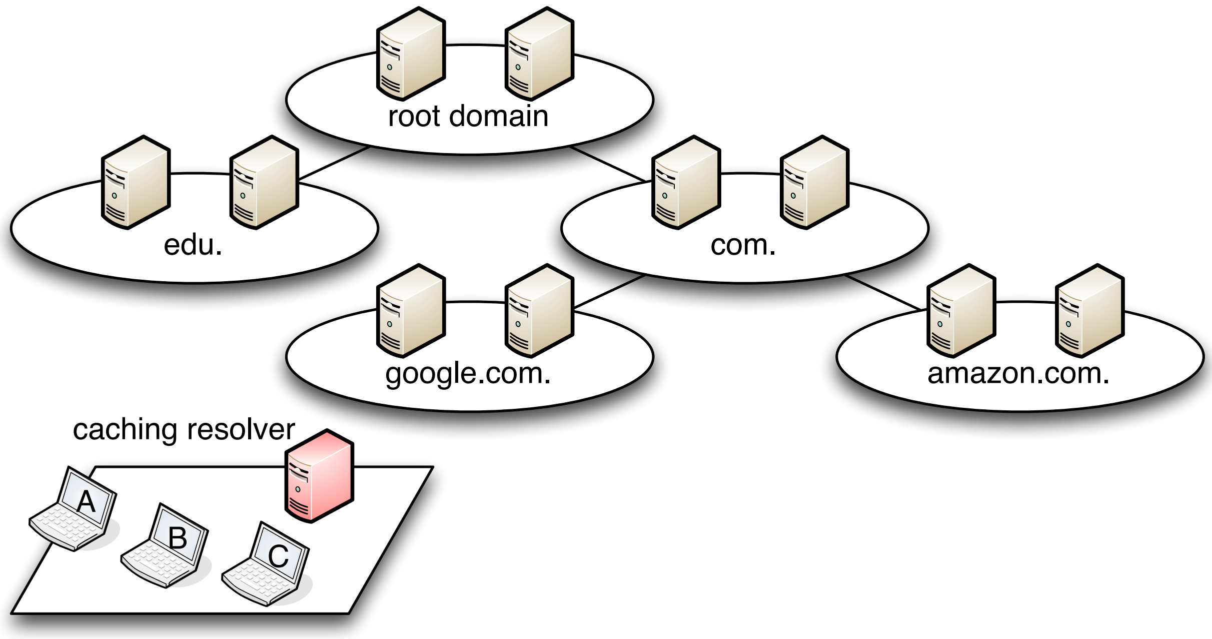
4. Since the bottleneck throughput for both clients is 50 Mbps, and the Server is capable of dealing with this load since $100Mbps <= (50Mbps + 50Mbps)$, the answer is once again 667 Mbps.

Problem 4

Consider the following environment with a local DNS caching resolver and a set of authoritative DNS name servers.

Assume that initially,

- the caching resolver cache is empty,
- TTL values for all records is 1 hour,
- RTT between stub resolvers (hosts A, B, and C) and the caching resolver is 20 ms,
- RTT between the caching resolver and any of the authoritative name servers is 150 ms,
- There are no packet losses,
- All processing delays are 0 ms



1. At T=0 min, Host-A sends a query for "A record for amazon.com", and after receiving the answer sends a query for "A record for www.amazon.com". How long did it take to receive all the answers?
2. At T=40 min, Host-B sends a query for "MX record for google.com" that returns

```
google.com.          3600   IN     MX     10 primary.google.com.
google.com.          3600   IN     MX     30 backup.google.com.
primary.google.com.  3600   IN     A      74.125.28.27
backup.google.com.   3600   IN     A      173.194.211.27
```

   (Similar to NS records, the DNS server may return "glue" A/AAAA records in addition to the requested MX records.) How long did it take to get the answer?

3. At T=70 min, Host-C sends a query for "AAAA (IPv6) record for mail.google.com", following at T=75 mins with a query for "AAAA (IPv6) record for hangout.google.com". How long did it take for Host-C to receive each of the answers (i.e., relative to T=70min for the first, and relative to T=75 mins for the second)?
4. List DNS records that the caching resolver has at T=90 minutes

Problem 4 continued on next page. . .

1. $D_{first} = D_{cache} + D_{root} + D_{.com} + D_{amazon.com}$

   $D_{first} = 20ms + 150ms + 150ms + 150ms = 470ms$

   $D_{second} = D_{cache} + D_{amazon.com}$

   $D_{second} = 20ms + 150ms = 170ms$

   $D_{total} = D_{first} + D_{second} = 470ms + 170ms = 640ms$

2. T=40 min, so .com info is still in cache. Therefore:

   $D = D_{cache} + D_{.com} + D_{google.com}$

   $D = 20ms + 150ms + 150ms = 320ms$

3. Since T=70 min, google.com info is still in cache. Therefore:

   $D_{first} = D_{cache} + D_{google.com}$

   $D_{first} = 20ms + 150ms = 170ms$

   $D_{second} = D_{cache} + D_{google.com}$

   $D_{second} = 20ms + 150ms = 170ms$

4. The content is as follows:

   google.com/NS, google.com/MX, primary.google.com/A, backup.google.com/A (Expires at T=100 min)

   mail.google.com/AAAA (Expires at T=130 min)

   hangout.google.com/AAAA (Expires at T=135 min)

Problem 5

(Food-for-Thought question) We will include a similar question in each homework. The question itself is simple, but serve as some food for new thought.

1. This simple exercise is to illustrate that, although statistical multiplexing of packet switching results in queueing delay at routers, this delay diminishes via technology advances: If we assume that an outgoing link of a router has 100 packets in the queue on average, each packet is 1250 bytes in size, if the link bandwidth is 10 Mbps (10^7 bits per second), what is the average queueing delay? How will this number change when the link bandwidth increases to 10 Gbps (10^{10} bits per second)?

2. As we learn from week-2 lectures, although HTTP and DNS serve very different purposes, they both make use of data caching: when different end users fetch the same HTTP objects (send DNS queries), the responses for earlier users can be cached to serve later requests. What are the benefits from utilizing caches?

1. Assuming that each packet waits on average for 99 packets before it:

   $D_{avg} = 100 * P_{packet}$

   $D_{avg} = 100 * \frac{1250*8bits}{1*10^7 bits/sec} = 100ms$

   With a 10 Gbps bandwidth:

   $D_{avg} = 99 * \frac{1250*8bits}{1*10^10bits/sec} = 100\mu s$

2. By utilizing caches we can return data to clients faster. Although speed is the primary benefit, other benefits include increased reliability, reduced server traffic, and distributed load. If a client's request corresponds to a cache hit, we can avoid a larger number of link hops which reduces the opportunity for packets to go missing. This also reduces the number of clients that the server has to interact with, thereby decreasing queuing delay and reducing request latency for other clients attempting to reach the server.