EE 450

Solutions to HW3

Dr. Walker

P2.1)

a) False: That is false, since the client first asks for the webpage and when receive the HTML response will understand the rest of 3 images and ask for each separately. There will be 4 request messages.
b) True: If they are on the same Host upon having the persistent connection with that host they can be asked during the length of the connection.
c) False: It is not possible.
d) False: Date field shows the time at which the HTTP response is created and sent out of the server.
e) False: It may have empty message body.

P2.3)

In the application layer first the application need to retrieve the host IP from the DNS. So DNS protocol need to be used and for fetching the data HTTP is needed.

In the transport layer to communicate with DNS, UDP will be used and to communicate with HTTP Server TCP will be the used protocol.

P2.4)

a) The Url is the name of the server concatenated with the address of the file on the server:
http://gaia.cs.umass.edu/cs453/index.html.
b) Indicated just before the first <cr><lf> pairs: The browser is running HTTP version 1.1.
c) Browser is asking for a persistent connection since it has indicated "keep alive" in the Connection filed of the message.
d) You cannot know the sender's IP from an HTTP message. IP information is enclosed in IP datagrams. It is carried the TCP segment which carried the HTTP GET request.

e) Mozilia 5.0. It is needed since the server need to know what version of the requested object need to be sent.

P2.7)

Transmission time is negligible. So there is only the propagation time which should be counted and it can be captured in RTTs.

To find the IP, we need to visit n different DNS which takes: $RTT_1 + \cdots + RTT_n$.

One the IP is known it takes RTT_0 to establish a connection to the server and another RTT_0 to get the webpage.

The total time needed is: Delay = $2RTT_0 + RTT_1 + \cdots + RTT_n$

P2.10)

Note that each downloaded object can be completely put into one data packet. Let Tp denote the one-way propagation delay between the client and the server.
First consider parallel downloads using non-persistent connections. Parallel downloads would allow 10 connections to share the 150 bits/sec bandwidth, giving each just 15 bits/sec. Thus, the total time needed to receive all objects is given by:

(200/150+Tp + 200/150 +Tp + 200/150+Tp + 100,000/150+ Tp)
+ (200/(150/10)+Tp + 200/(150/10) +Tp + 200/(150/10)+Tp + 100,000/(150/10)+ Tp)
= 7377 + 8*Tp (seconds)

Now consider a persistent HTTP connection. The total time needed is given by:

(200/150+Tp + 200/150 +Tp + 200/150+Tp + 100,000/150+ Tp)
+ 10*(200/150+Tp + 100,000/150+ Tp)
=7351 + 24*Tp (seconds)

Assuming the speed of light is 300*10^6 m/sec, then Tp=10/(300*10^6)=0.03 microsec. Tp is therefore negligible compared with transmission delay.

Thus, we see that persistent HTTP is not significantly faster (less than 1 percent) than the non-persistent case with parallel download.

P2.14)

SMTP uses a dot at the end of message to indicate EOM.
HTTP uses "Content-Length header field" to show the length of the message and indicate the end of message.

No, Http cannot use the SMTP way to indicate EOM. SMTP only uses ASCII characters, so it possible to parse a message and get each character out and search for a specific one. However, Http can use binary data for which the same method SMTP cannot be used.

P2.22)

Following formula can be used to calculate the minimum distribution time when we can client-server structure.

$$D_{cs} = max \{NF/u_s, F/d_{min}\}$$

In the case of P2P communication the minimum distribution time will be calculated according to this formulaL

$$D_{P2P} = max\{F/u_s, F/d_{min}, NF/(u_s + \sum_{i=1}^{N} u_i)\}$$

Where, F = 15 Gbits = 15 * 1024 Mbits
u_s = 30 Mbps
$d_{min} = d_i$ = 2 Mbps

Client Server

		N		
		10	**100**	**1000**
	300 Kbps	7680	51200	512000
u	**700 Kbps**	7680	51200	512000
	2 Mbps	7680	51200	512000

Peer to Peer

		N		
		10	**100**	**1000**
	300 Kbps	7680	25904	47559
u	**700 Kbps**	7680	15616	21525

2 Mbps	7680	7680	7680

P2.24)

a) lets u = u1 + u2 + ….. + uN.

According to the Problem:

$u_s <= (u_s + u)/N$ *

Divide the file into N parts, with the ith part having size $(u_i/u)F$. The server transmits the ith part to peer i at rate $r_i = (u_i/u)u_s$. Note that $r_1 + r_2 + \ldots.. + r_N = u_s$, so that the aggregate server rate does not exceed the link rate of the server. Also have each peer i forward the bits it receives to each of the $N\text{-}1$ peers at rate r_i. The aggregate forwarding rate by peer i is $(N\text{-}1)r_i$. We have

$$(N-1)r_i = (N-1)(u_s u_i)/u <= u_i,$$

where the last inequality follows from (*). Thus the aggregate forwarding rate of peer i is less than its link rate u_i.

In this distribution scheme, peer i receives bits at an aggregate rate of

$$r_i + \sum_{j<>i} r_j = u_s$$

Thus each peer receives the file in F/u_s.

b) In this case lets $u = u_1 + u_2 + \ldots.. + u_N$

$u_s >= (u_s + u)/N$ **

Let $r_i = u_i/(N\text{-}1)$ and
$r_{N+1} = (u_s – u/(N\text{-}1))/N$

In this distribution scheme, the file is broken into $N+1$ parts. The server sends bits from the ith part to the ith peer (i = $1, \ldots., N$) at rate r_i. Each peer i forwards the bits arriving at rate r_i to each of the other N-1 peers. Additionally, the server sends bits from the $(N+1)$st part at rate r_{N+1} to each of the N peers. The peers do not forward the bits from the $(N+1)$st part.

The aggregate send rate of the server is

$r_1 + \ldots + r_N + N\, r_{N+1} = u/(N-1) + u_s - u/(N-1) = u_s$

Thus, the server's send rate does not exceed its link rate. The aggregate send rate of peer i is

$(N-1) r_i = u_i$

Thus, each peer's send rate does not exceed its link rate.
In this distribution scheme, peer i receives bits at an aggregate rate of

$$r_i + r_{N+1} + \sum_{j<>i} r_j = u/(N-1) + (u_s - u/(N-1))/N = (u_s + u)/N$$

Thus each peer receives the file in $NF/(u_s+u)$.
(For simplicity, we neglected to specify the size of the file part for i = 1, …., N+1. We now provide that here. Let $\Delta = (u_s+u)/N$ be the distribution time. For i = 1, …, N, the ith file part is $F_i = r_i \Delta$ bits. The (N+1)st file part is $F_{N+1} = r_{N+1} \Delta$ bits. It is straightforward to show that $F_1 + \ldots + F_{N+1} = F$.)

c) The solution to this part is similar to that of 17 (c). We know from section 2.6 that

$$D_{P2P} >= max\{F/u_s, NF/(u_s + u)\}$$

using this with a) and b) gives the requested result.

P2.26)

Yes. His first claim is possible, as long as there are enough peers staying in the swarm for a long enough time. Bob can always receive data through optimistic unchoking by other peers.

His second claim is also true. He can run a client on each host, let each client "free-ride," and combine the collected chunks from the different hosts into a single file. He can even write a small scheduling program to make the different hosts ask for different chunks of the file. This is actually a kind of Sybil attack in P2P networks.

P2.27)

When Peer 3 learns that peer 5 has left the system, it will asks its first successor (Peer 4) for the identifier of its immediate successor (peer 8).

So it will replace peer 8 by peer 5.