/

Problem 1

Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that n DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT_1, RTT_2, ..., RTT_n. Further suppose that the Web page associated with the link has a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the HTML file. Assume zero transmission time. Suppose the HTML file references 11 very small objects on the same server. Neglect transmission times, how much time elapses from when the client clicks on the link until the client receives all objects with:

(a) Non-persistent HTTP with no parallel TCP connections?

(b) Non-persistent HTTP with the browser configured for 5 parallel connections?

(c) Persistent HTTP with no parallel TCP connections?

(d) Persistent HTTP with the browser configured for arbitrarily many parallel connections?

Write your solution to Problem 1 in this box

a. time to get IP address: $RTT_1 + RTT_2 + \dots + RTT_N$

need to set up TCP & obtain index: $2RTT_0$

to obtain 11 objects: $11 \cdot 2 \cdot RTT_0$

total: $11(2RTT_0) + 2RTT_0 + RTT_1 + \dots + RTT_n$

$= \boxed{RTT_1 + \dots + RTT_n + 24RTT_0}$

b. 5 parallel connections ⇒ need to do this 3 times for 11 files

total: $3(2RTT_0) + 2RTT_0 + RTT_1 + \dots + RTT_n$

$= \boxed{RTT_1 + \dots + RTT_n + 8RTT_0}$

c. need to get IP address: $RTT_1 + \dots + RTT_n$

need to set up TCP: RTT_0

obtain index: RTT_0

obtain 11 objects: $11RTT_0$

total: $\boxed{RTT_1 + \dots + RTT_n + 13RTT_0}$

d. $RTT_1 + \dots + RTT_n + RTT_0 + RTT_0 + RTT_0$

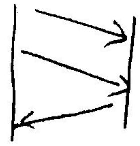
$= \boxed{RTT_1 + \dots + RTT_n + 3RTT_0}$

Problem 2

How does the web server (e.g., eBay) identify users when you do the Internet shopping? Briefly explain how it works.

Write your solution to Problem 2 in this box

A web server like eBay identifies users using cookies. The cookie is used by the web server to remember the user and to take a specific action for that user. The web server does this by creating a unique cookie entry for the client upon receipt of the first HTTP request message. The cookie is saved in the backend database of the server and also in the client's own local memory in its cookie file. When the client visits that site again, when sending its usual http request message, the cookie is also sent and the server uses this information to perform a specific action, depending on the user.



Problem 3

A Web browser running on the client host is requesting a webpage from the server. We make the following assumptions:

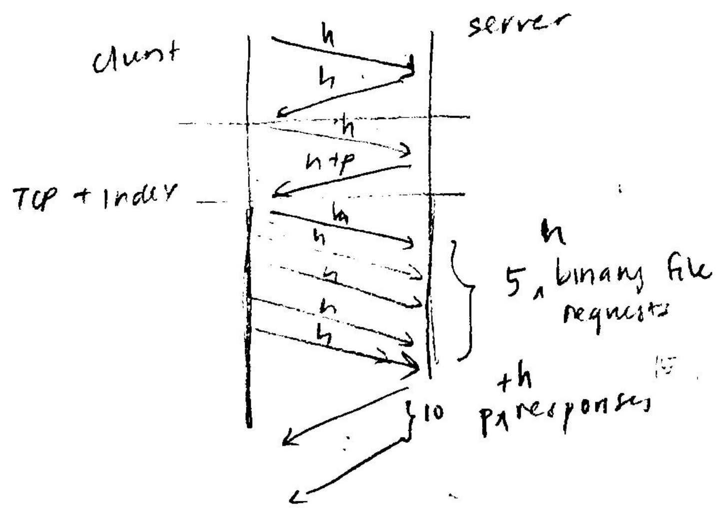
- TCP window is large once the TCP handshake is complete (i.e. ignore flow control). TCP header size is h bits, and the maximum payload size is p bits.
- The bandwidth is b bps, and the propagation delay is d seconds.
- Ignore DNS related delays, and ignore the payload in three-way handshake packets, ACK packets, and HTTP request packets. In other words, those packets consist of header only. h
- The client requests a webpage consisting of an HTML file that indexes 5 binary files on the same server. Each of the file is $2p$ bits long. In other words, each of the file can be sent in exactly 2 TCP packets. Piggybacking is used whenever possible.
- Each HTTP request is sent in one TCP packet.

Please answer the following questions:

(a) Suppose pipelining of HTTP requests is allowed and no parallel TCP connections are used. Calculate the minimal time it takes the browser to receive all the files.

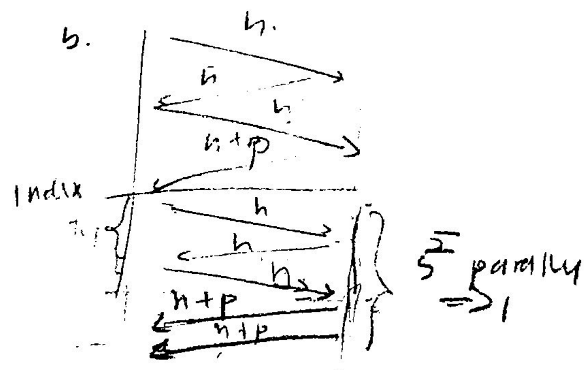
(b) Suppose the non-persistent, non-pipelining mode with parallel TCP connections is used, repeat the calculation.

(c) Which mode gives the smaller latency? Briefly justify your answer.



a. $$\frac{19h + 11p}{b} + 19d$$

b.

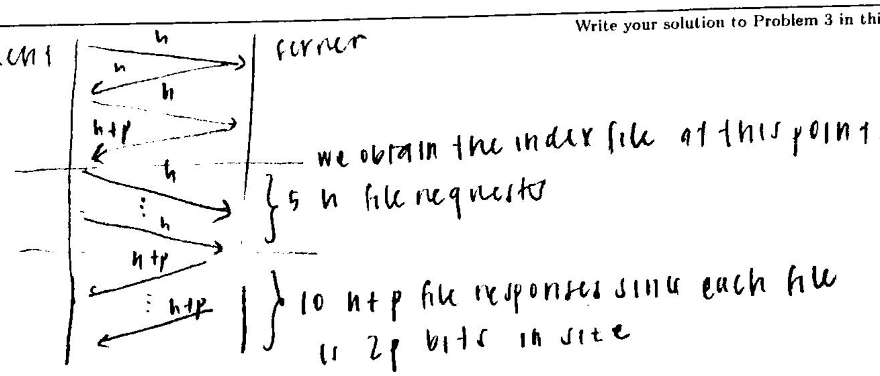


$$\frac{9h + 3p}{b} + 9d$$

Problem 3 continued on next page...

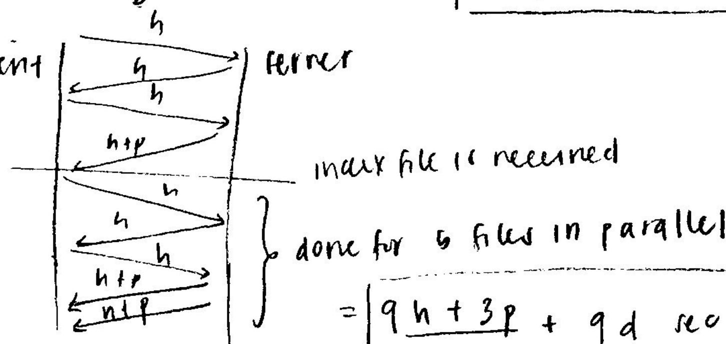
Write your solution to Problem 3 in this box

a. client



$$= \frac{10(h+p)+5h+4h+p}{b} + 19d = \boxed{\frac{19h+11p}{b} + 19d \text{ sec}}$$

b. client



$$= \boxed{\frac{9h+3p}{b} + 9d \text{ sec}}$$

c. Mode B gives smaller latency because the 5 requests and 10 responses for the 5 files are done in parallel on different sockets rather than on a single socket.

Problem 4

How does SMTP marks the end of a message body? How about HTTP? Can HTTP use the same method as SMTP to mark the end of the message body?

Write your solution to Problem 4 in this box

SMTP uses a CRLF.CRLF to mark the end of a message.
~~HTTP uses a single CRLF to indicate the end of a message body~~.
HTTP marks the end of the message body by indicating the content length in the "content-length:" field in the header. HTTP cannot use the same method as SMTP since HTTP can contain data of any type while SMTP must contain data in ASCII format.

Problem 5

Suppose your department has a local DNS server for all computers in the department.

(a) Suppose you are an ordinary user (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

(b) Now suppose you are a system administrator and can access the caches in the local DNS servers of your department. Can you propose a way to roughly determine the Web servers (outside your department) that are most popular among the users in your department? Explain.

Write your solution to Problem 5 in this box

a. You can query the site ~~and~~ using the dig function. The dig function performs the DNS lookup. ~~and sees~~ The resulting query time, if it is small, indicates whether it was likely accessed recently.

b. To ~~co~~ roughly determine ~~p~~ which web servers are most popular, as an administrator, you can record the entries that have been cached. ~~How~~ By looking at the most cached entries, you can determine the ~~rat~~ sites that are most popular.