Name: Rajat Rajesh shetty
CWID: 10477484

Assignment 4

pg. (210) 7
9
22
26
28

(7) suppose within your browser you can click on a link to obtain a webpage. The IP address for the associated URL is not cached in your localhost, so a DNS lookup is necessary to obtain the IP address. suppose that n DNS servers are visited before your host recieves the IP address from DNS, the successive visits incur an RTT of $RTT_1, \ldots RTT_n$. Further suppose that the web page associated visit with link contains exactly one object, consisting of a small amount of HTML text. Let RTT_0 denote the RTT between the local host & the Server containing the object. Assuming zero transmission time of the object, how much time elapses from when the Client clicks on a link until the client recieves the object?

=) consider the IP address of total amount of time:

$RTT_1 + RTT_2 + \cdots RTT_n$

Time elapses from when the client clicks on the link until the client recieves the object:

$2RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$

(9) consider figure 2.12 for which there is an institutional network connected to Internet. suppose that the avg object size is 850,000 bits and that the average request rate from the institution's browser to the origin server is 16 requests per second. Also suppose that the amount of time it takes from router on the Internet side of the access link forwards an HTTP request until it recieves the response is 3 sec on average. use $\Delta/(1-\Delta\beta)$, where the average time required to send an object over the access link & β is the arrival rate of object to access link.

a) Find the total avg. response time.

b) Now suppose a cache installed in the institutional LAN, suppose the miss rate is 0.4. Find total response time.

⇒

a). The time to transmit an object of size L over a link or rate is L/R. The avg time is the average size of the Object divided by R.

$$= \frac{850{,}000 \text{ bits}}{(15{,}000{,}000 \text{ bits/sec}} \Rightarrow \underline{0.567 \text{ sec}}$$

(b) The traffic intensity on the link is given by $\beta\Delta$ =
(16 requests/sec) (.0567 sec /request)
$$= \underline{0.907}$$

Thus avg. access delay is $\frac{(0.567 \text{ sec})}{(1-0.907)}$
$$\simeq \underline{0.6 \text{ sec}}$$

∴ the total average response time is ∴ .6 sec
+3 sec
⇒ $\underline{3.6 \text{sec}}$

b) The traffic intensity on the access link is reduced by 60% since the 60% of the requests are satisfied within institutional network. Thus the avg. access day is $\frac{(.0567 \text{sec})}{(4)(.907)} = \underline{0.089 \text{ secs.}}$

The response time is approximately zero if the request is satisfied by the cache.
the average response time is 0.089 sec +3 sec
≈ 3.089 sec
for cache misses (which happens 40% of the time).
So, the average response time is 10.67(0 sec) +
(.4)(3.089 sec) $\underline{1.24 \text{ sec}}$

Thus the avg response of time is reduced from ~~3.8~~ 3.65 to $\underline{1.24 \text{sec}}$

22) Consider distributing a file of $F = 15$ Gbits to N peers. The server has an upload rate of $u_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps & an upload rate of u. For N = 10, 100, 1000 and u = 300 Kbps, 700 kbps & 2 Mbps, prepare a chart giving in the minimum distribution time for each of the combination of N and u for both client-server distribution and P2P distribution.

=) For calculating the minimum distribution time for client-server distribution, we use the following formula:

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

Similarly for calculating the minimum distribution time for P2P distribution, we use the formula:

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

Given

$F = 15$ Gbits $= 15360$ Mbits. $U_s = 30$ Mbps $d_i = 2$ Mbps $= d_{min}$

Note: 300 kbps = 300/1024 Mbps

Client-server

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

example to solve

$$D_{cs} = \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\}$$

$$= \max\left\{\frac{10 \times 15360}{30}, \frac{15360}{2}\right\}$$

$= \max\{5120, 7680\}$

$= 7680$ sec

Peer to Peer

u		10	100	1000
	300 Kbps	7680	25906	47569
	700 Kbps	7680	15616	21525
	2 Mbps	7680	7680	7680

solved one case.

$U_s = 30$ Mbps

$F = 15360$ Mbit

$u = 300$ kbps

$= \frac{300}{1024} = 0.2929$ Mbps

$\therefore \sum_{i=1}^{N} u_i = \sum_{i=1}^{100} 0.2929$

$= 100 \times 0.2929$

$= 29.29$

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

$$= \max\left\{\frac{15360}{30}, \frac{15360}{2}, \frac{100 \times 15360}{30 + 29.29}\right\}$$

$= \max\{512, 7680, 25906\} \Rightarrow 25906$ sec

26) Suppose Bob joins a bittorrent torrent, but he doesnot want to upload any data to anyother peers.

a) Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is Bob's claim possible? why or why not?

b) Bob futher claims that he can further make his "Free-riding" more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?

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

b) His second claim is also true. he can run a client on each host; let each client "Free-ride," and combine 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 file. This is actually a kind of sybil attack in P2P network.

---

28). Install & compile the python prog. TCPclient & UDPclient on the host & TCPserver & UDPserver on another Host.

a. suppose you run TCPclient before you run TCPserver. what happens? why?

b. suppose you run UDP client before you run UDP server, what happens? why?

c. what happens if you use different port numbers for the client & server sides?

⇒ a) If you run TCPclient first, then client will attempt to make a TCP connection with a non existent server process. A TCP connection will not be made.

b) UDPclient doesnot establish a TCP connection with the server. Thus everything should work fine if you first run UDPclient, then run UDPserver, and then type some input into the keyboard.

c) If you use a different port numbers, then the client will attempt to establish a TCP connectn with the wrong process or a non-existent process. Error will occur.

---

28) Continue

TCP client

```
from socket import *

def client():
 HosT = "10.211.55.9'
 PORT = 10521
 BUFSIZE = 1024.

Clientsocket = socket(AF-INET, SOCK-STREAM)
clientsocket.connect((HOST, PORT))
while True:
   data = input('I>')
   if not data:
     break
  clientsocket.send(data.encode())
 data= client socket.recv(BUFSIZE).decode()
if not data:
    break
 print (data)
Client()
```

TCP server

```
from socket import *
from time import ctime

def server():
   HoST = ''
   PORT = 10521
 BUFSIZE = 1024
 ADDR = (HOST, PORT)
```

```
server-socket = socket (AF-INET, SOCK-STREAM)
server-socket.bind (ADDR)
server-socket.listen (5)
while True:
   print ('waiting for connecting · · · ')
tcpclientsocket, addr = server_socket.accept()
print ('conneted from', addr)
 while True:
    data = tcpclientsocket.recv (BUFSIZE).decode().
 if not data:
     break
   print (data)
 data = input ('I>')
 tcp.client socket.send (('[%s]%s' %((time(), data)).encode())
tcpclientsocket.close()
server-socket.close()
server()
```