Analysis Report of 3 question.

#1.

a) The total number of objects downloaded for each website were found to be as under:-

(i). nytimes.com : 197
(ii). Vox.com : 79

#3.

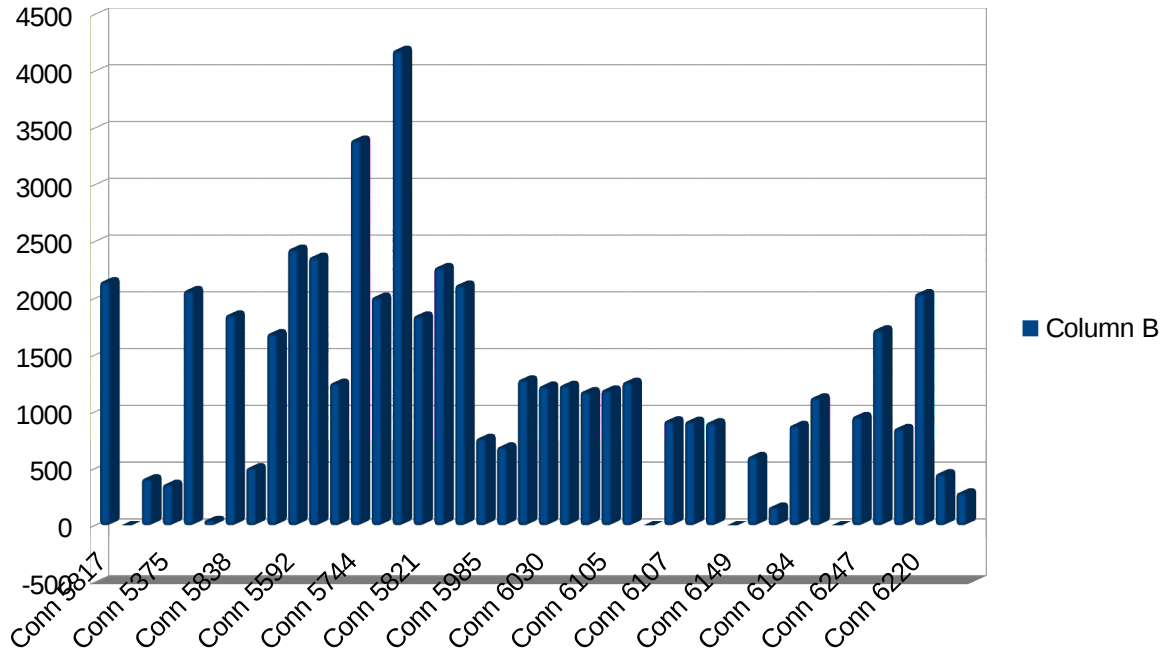
a. Find the page load time, ie. time from sending the first DNS request to the end time of loading the last object
Ans: PageLoadTime for www.nytimes.com is 58527.145 milliseconds.
PageLoadTime for www.vox.com is 267297.631 milliseconds

b)For each domain, finding the time spent in the DNS query when opening the first TCP connection
Ans: As we have cleared the DNS cache before generating the HAR file the DNS time for initial request would be some initial value but for some requests it is very less and for some it is zero. The reason that DNS value is zero is that hostname is present in DNS Cache of the network we used for downloading the HAR file.

c) Timing Analysis for Each TCP Connection:

i) Connection Establishment Time :
The Connection Establishment Time for each TCP Connection should be non zero for the first request being sent on the connection as a TCP HandShake has to occur as it is the case for almost all the Connections. However subsequent requests of the objects on the same connection do not need the Handshake to occur again. Therefore the subsequent requests on same connection have -1 as DNS Value. But some connections on their establishment itself have -1 as value which is clearly a bug.

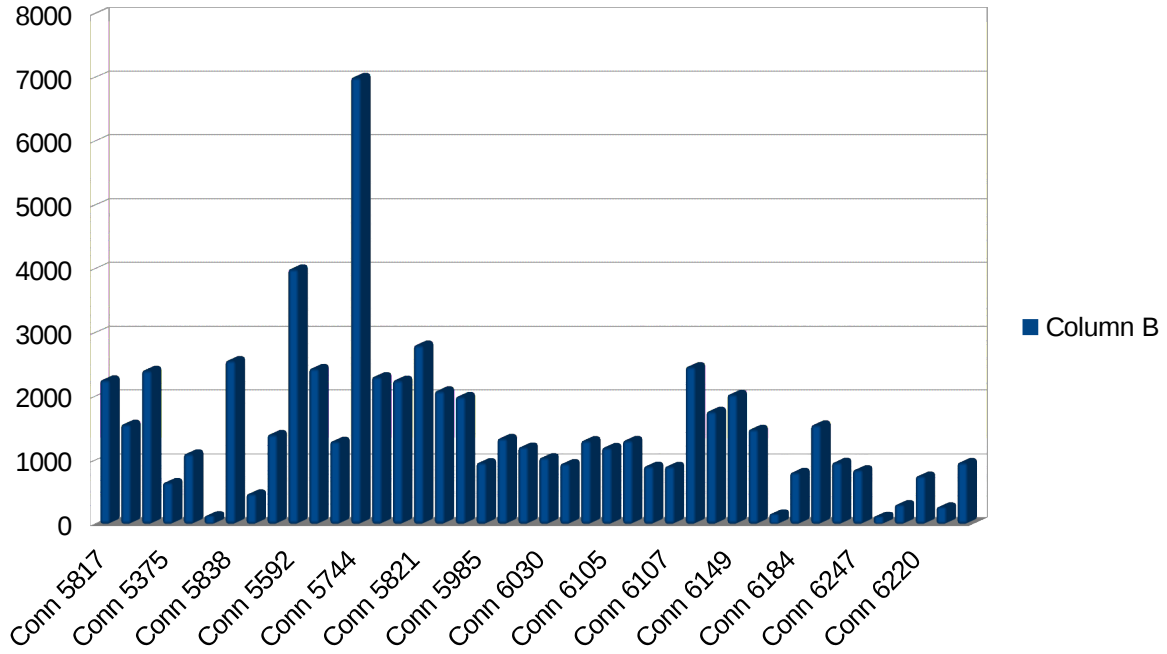
TCP_Connection vs Establishment_Time
(for nytimes.com)



ii) Wait Time:

Total time spent in waiting for the server to respond. Entries.timings.wait accessed for calculations of this.

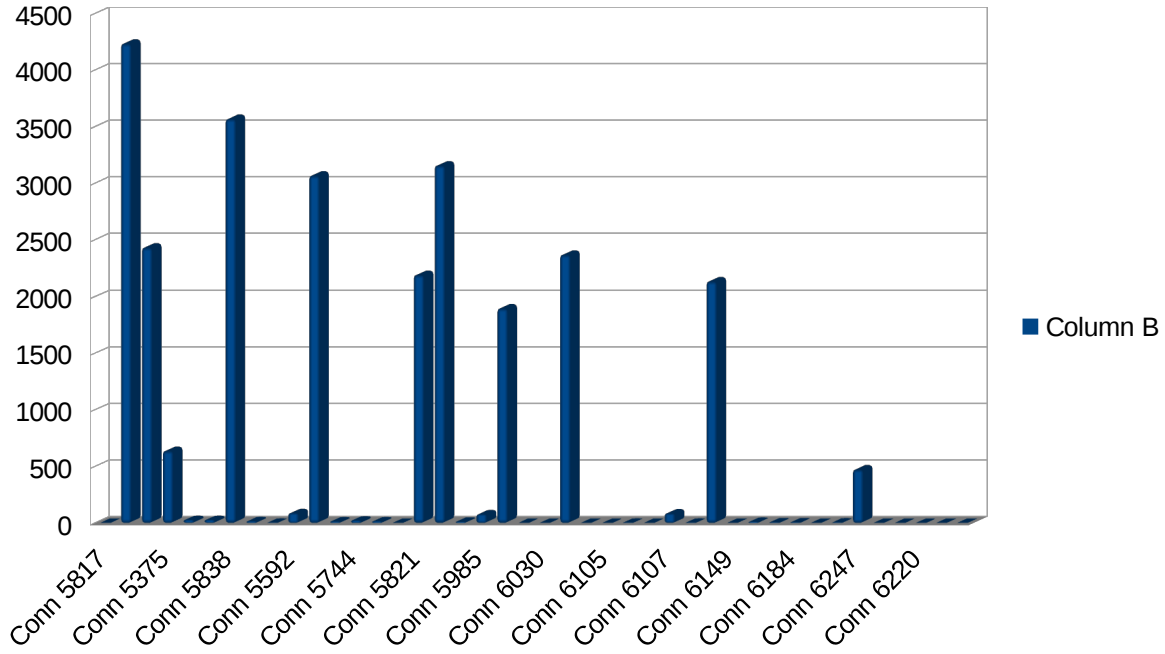
TCP_Connection vs Wait_Time



iii) Receive Time:

Total time spent in receiving data. Entries.timings.receive is accessed for calculating it.

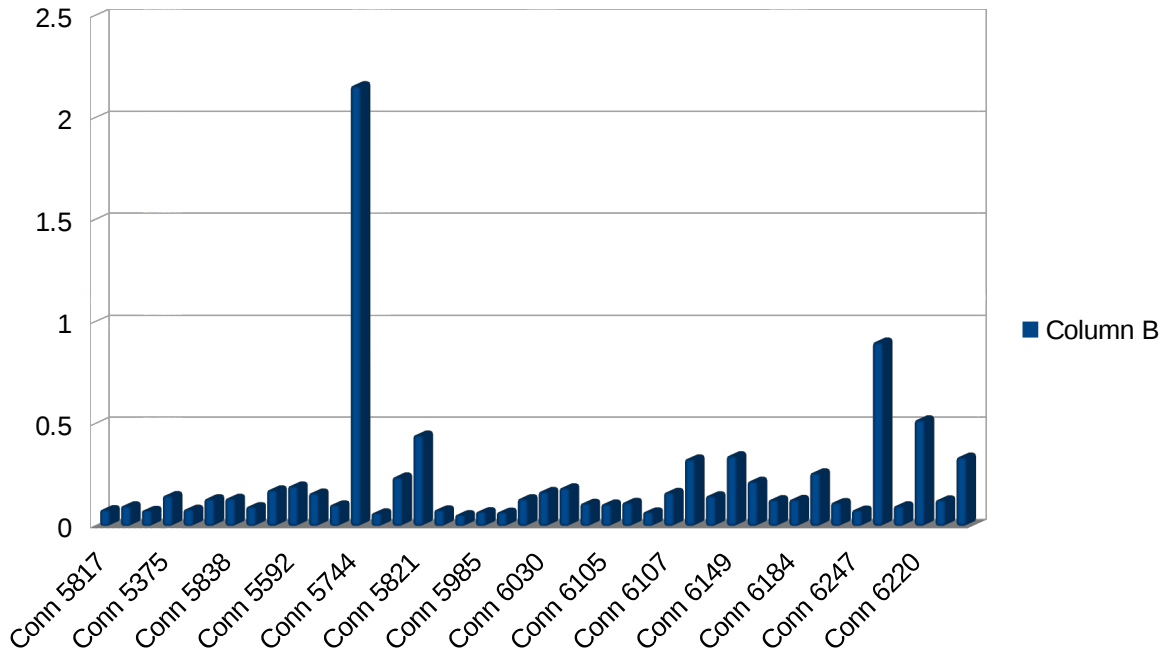
TCP_Connection vs Receive_Time



iv) Send Time:

Ideally server should send data (provide services to client) as soon as it receives the request. But due to various reasons server is not able to respond so, that delay is stored in HAR File across every object in every TCP Connection. This kind of situation arise when, Number of request to the Server is high in comparision to the number of requests it can service. Or the server is not avaliable with enough resources to service a particular request so, It waits until the point it is available with its resources and then it responds back.

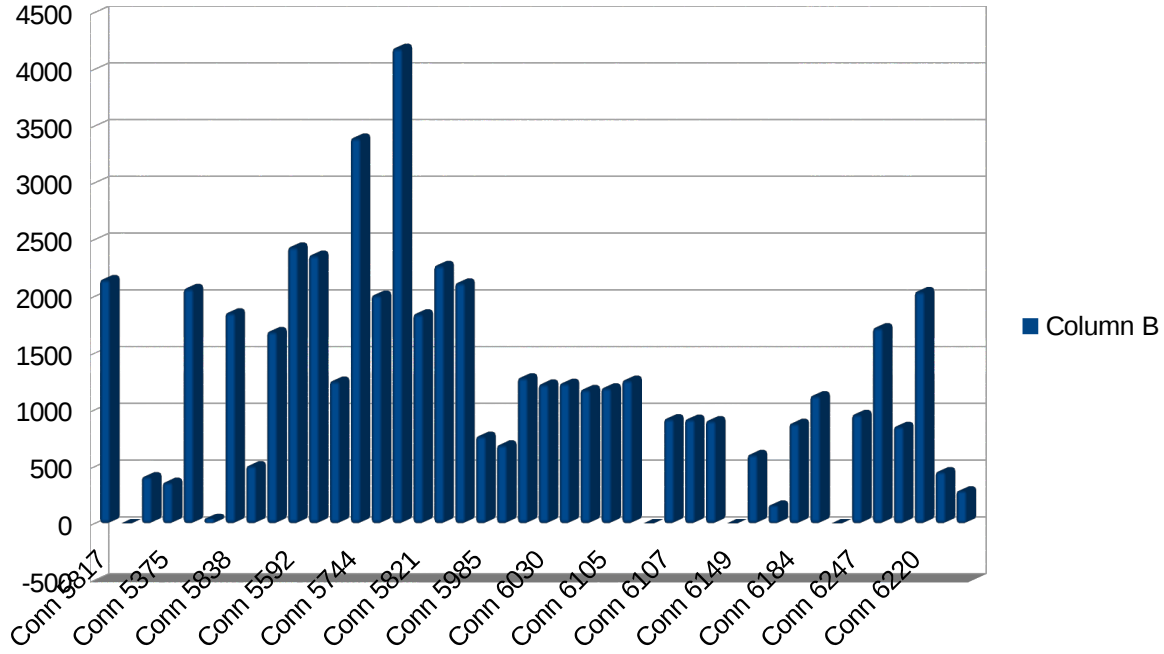
TCP_Connection vs Send_Time



v) Connection Active Time:

The start time of the first object being downloaded on the connection is seen. The blocking time for this object is not considered(as we have

TCP_Connection vs Active_Time



to calculate from the time when the first request is being sent).Next we see the last object being downloaded on that connection. We also add the receive time of this object to its start time. From this the start time of the first object is subtracted to get the total active time of the connection.

vi) Active Percentage:

Calculated from Formula given

Active percentage = [sending + waiting + receiving] / [total active time] * 100

Idle Percentage = (1 – Active Percentage) * 100

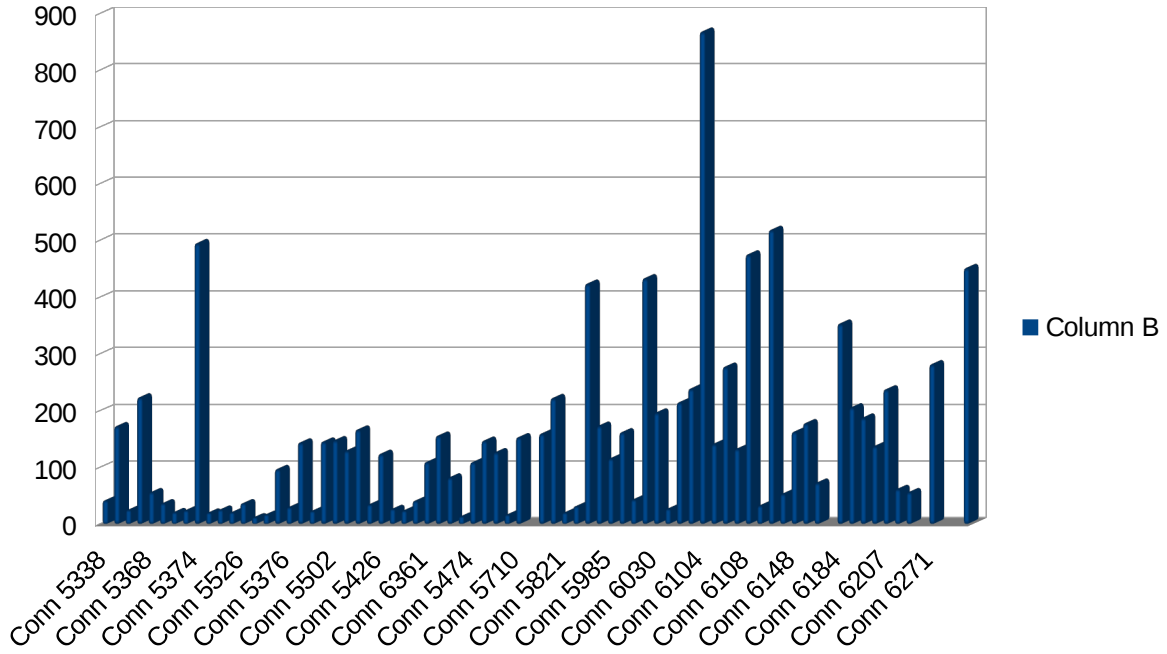
viii) Average GoodPut of Connection:

It is calculated by

Average goodput = [total data received] / [total time spent in receiving]

Since the Connection contains many object downloads the avergae good put is calculated by dividing the total size of objects received in that particular connection to total time spent in receiving the objects.

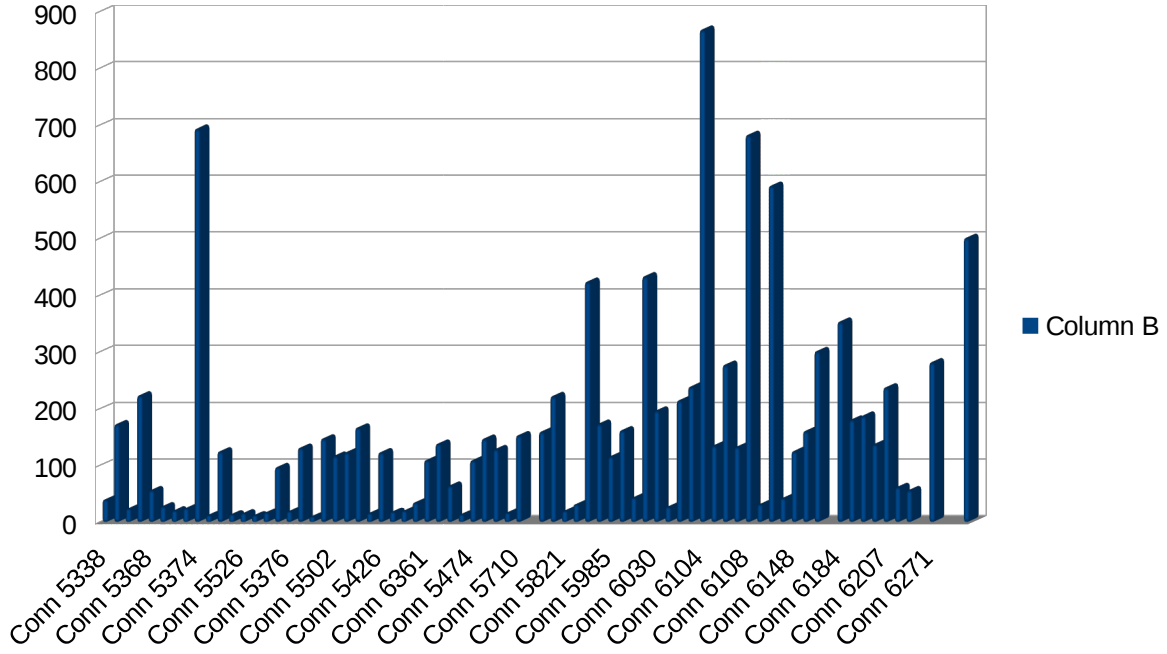
TCP_Connection vs Avg Good Put



ix)Maximum GoodPut:

Max GoodPut = [Largest Object Received]/[Time spent in receiving that object]
The largest object downloaded on the connection was extracted and then it was divided by its receive time. The results are stored along with the script. The largest objects for each connection were then dowloaded directly from the browser and their timings were observed. The goodput was observed to be more than what has been calculated above because in the case of a web page there are other objects also which are being downloaded on other connections.

TCP_Connection vs Max GoodPut



x)Average Goodput Across the Network:

Average GoodPut for the network is 97.6053748772 B/ms for vox.com and 168.850950024 B/ms for nytimes.com respectively. It is calculated by average Goodput from all the domains.

xi) The maximum of the maximum achieved goodput of all the connections for each website has been found out to be for vox.com 1575.22123895 and for nytimes.com 867.768595081 respectively. Comparing this with the average goodput of the network, it is observed that the average goodput is less than these values and the network utilisation has been not satisfactory.

#4 : Browser Scheduling Policies:

(a) The Maximum Number of TCP Connections opened per Domain is 7 and nytimes.com respectively

(b) The Maximum number of TCP Connections opened per domain Simultaneously 6 for nytimes.com

(c) Hence there is cap by the browser on No.of Simultaneous Connection i.e 6

5. What if Scenarios .

(a)
With the help of the object tree, we can collapse all GET requests of the connection and achieve pipelining. In this case the waiting time of subsequent objects would reduce and we would achieve lesser latency. Also the server will have more data to send and will thus lead to better utilisation and throughput. However the objects would be returned strictly in the order in which they were requested. For the page load time we will do the following calculations:-

Analysis:

(i)
The start time of the first request sent on the connection is seen. Then receivetime of the last object received on the connection is seen. This is the total time for this connection(the blocking time is also considered).
(ii)
Similar time is calculated for each connection. The page load time would be given by Last of finishing times of connections – Earliest of start times of connections .

(b)
Assuming that the entire data is downloaded on a single connection, the page load time would be calculated as under :-

(i)
Size of content downloaded on each domain is already available to us. This would be divided by the maximum achieved throughput of that domain to get the time taken for that domain.Also the time to download the base page or the first object would be in addition to this time as all domains can only be connected to after the first object has been downloaded.
(ii)
The above exercise would be repeated for all domains. Assuming that all domains would be connected to in parallel, the page load time would be calculated by the maximum of domain times.

(c)
The result in the first case would be better as more number of parallel connections would be available in that case. We are assuming that all connections would be in parallel, impying we can have more than one connection to a domain operating in parallel,with pipelining in place. Also the result would be better, provided we are not limited by the network threshold.