

CS 433

Computer Networks

Assignment 4

Chris Francis

18110041

November 23, 2020

I am using a fixed buffer size of 32768 Bytes (32 KB) for all these experiments unless stated otherwise, since it gave me the best throughput(in both TCP and UDP) in the last assignment.

Persistent or non-persistent connections

1 a)

TCP

To make a single client download the 5 files over a persistent connection, run the following:

In Terminal 1:

```
cd TCP/server
python3 TCPserver_persistent.py 32768
```

In Terminal 2:

```
cd TCP
python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt
War_and_Peace.txt Brothers_Karamazov.txt
```

Output Screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPserver_persistent.py 32768
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 55934) is connected.
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
^

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt War_and_Peace.txt Brothers_Karamazov.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected(took 7.867813110351562e-05 seconds).
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received in 0.004480838775634766 seconds. Saved as BibleTCP7123.txt
File found.
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received in 0.002510547637939453 seconds. Saved as RamayanaTCP7123.txt
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received in 0.002065420150756836 seconds. Saved as Anna_KareninaTCP7123.txt
File found.
Receiving War_and_Peace.txt...
War_and_Peace.txt(3359584 Bytes) received in 0.0033321380615234375 seconds. Saved as War_and_PeaceTCP7123.txt
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received in 0.002071380615234375 seconds. Saved as Brothers_KaramazovTCP7123.txt
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$
```

One-time connection setup time = 7.87×10^{-5} seconds

Total download time(including connection setup time) = 0.01453900337 seconds

Aggregate throughput = Total data downloaded/Total download time
= 939.6499893 MBps

Download time and achieved throughput for each file:

File Name	Data transferred(Bytes)	Time (s)	Throughput(MBps)
Bible.txt	4456586	0.004480838776	948.512504
Ramayana.txt	2396753	0.002510547638	910.4474834
Anna_Karenina.txt	2068079	0.002065420151	954.901997
War_and_Peace.txt	3359584	0.003332138062	961.5294791
Brothers_Karamazov.txt	2044197	0.002071380615	941.1588398

UDP

To make a single client download the 5 files over a persistent connection, run the following:

In Terminal 1:

```
cd UDP/server
python3 UDPserver_persistent.py 32768
```

In Terminal 2:

```
cd UDP
python3 UDPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt
War_and_Peace.txt Brothers_Karamazov.txt
```

Output Screenshots:

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver_persistent.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Client exited.
[*] Ready to receive as 127.0.0.1 : 12345
█

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt War_and_Peace.txt Brothers_Karamazov.txt
File found.
Receiving Bible.txt...
Bible.txt(1900682 Bytes) received in 0.0025167465209960938 seconds. Saved as BibleUDP8786.txt
File found.
Receiving Ramayana.txt...
Ramayana.txt(1348177 Bytes) received in 0.0020246505737304688 seconds. Saved as RamayanaUDP8786.txt
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(1412719 Bytes) received in 0.0017719268798828125 seconds. Saved as Anna_KareninaUDP8786.txt
File found.
Receiving War_and_Peace.txt...
War_and_Peace.txt(2343776 Bytes) received in 0.002676248550415039 seconds. Saved as War_and_PeaceUDP8786.txt
File found.
Receiving Brothers_Karamazov.txt...
Time out: Closing socket
Brothers_Karamazov.txt(1323301 Bytes) received in 0.001676321029663086 seconds. Saved as Brothers_KaramazovUDP8786.txt
```

One-time connection setup time is not present as there is no connection.

Total download time = 0.01066589355 seconds

Aggregate throughput = Total data downloaded/Total download time
= 744.6937589 MBps

Download time and achieved throughput for each file:

File Name	Data transferred(Bytes)	Time (s)	Throughput(MBps)
Bible.txt	1900682	0.002516746521	720.2281167
Ramayana.txt	1348177	0.002024650574	635.0339143
Anna_Karenina.txt	1412719	0.00177192688	760.3439182
War_and_Peace.txt	2343776	0.00267624855	835.1985746
Brothers_Karamazov.txt	1323301	0.00167632103	752.8380031

1 b)

I repeated the required measurements for non-persistent instead of using the results from the last assignment, to ensure that the measurements for persistent and non-persistent are taken in almost similar conditions(state of the computer, network etc).

TCP

Output screenshots for non-persistent:

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPserver.py 32768 n n
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 56028) is connected.
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPserver.py 32768 n n
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 56064) is connected.
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPclient.py 32768 n n Bible.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP7591.txt
Elapsed time: 0.0051441192626953125 seconds
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPclient.py 32768 n n Ramayana.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP7686.txt
Elapsed time: 0.0032134056091308594 seconds
```

(continued next page)

```

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPServer.py 32768 n n
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 56102) is connected.
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPServer.py 32768 n n
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 56118) is connected.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPServer.py 32768 n n
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 56146) is connected.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$

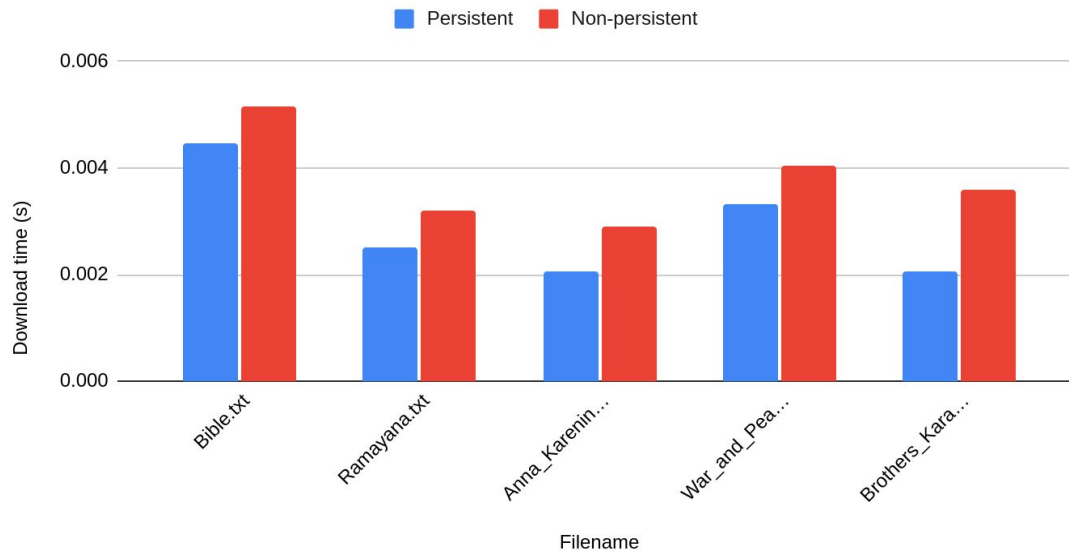
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPClient.py 32768 n n Anna_Karenina.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP7884.txt
Elapsed time: 0.0029163360595703125 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPClient.py 32768 n n War_and_Peace.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
Receiving War_and_Peace.txt...
War_and_Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP7954.txt
Elapsed time: 0.0040361881256103516 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPClient.py 32768 n n Brothers_Karamazov.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP8061.txt
Elapsed time: 0.0035986900329589844 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$

```

Comparison of persistent vs non-persistent in terms of individual file download time(s):

File name	Persistent	Non-persistent
Bible.txt	0.004480838776	0.005144119263
Ramayana.txt	0.002510547638	0.003213405609
Anna_Karenina.txt	0.002065420151	0.00291633606
War_and_Peace.txt	0.003332138062	0.004036188126
Brothers_Karamazov.txt	0.002071380615	0.003598690033

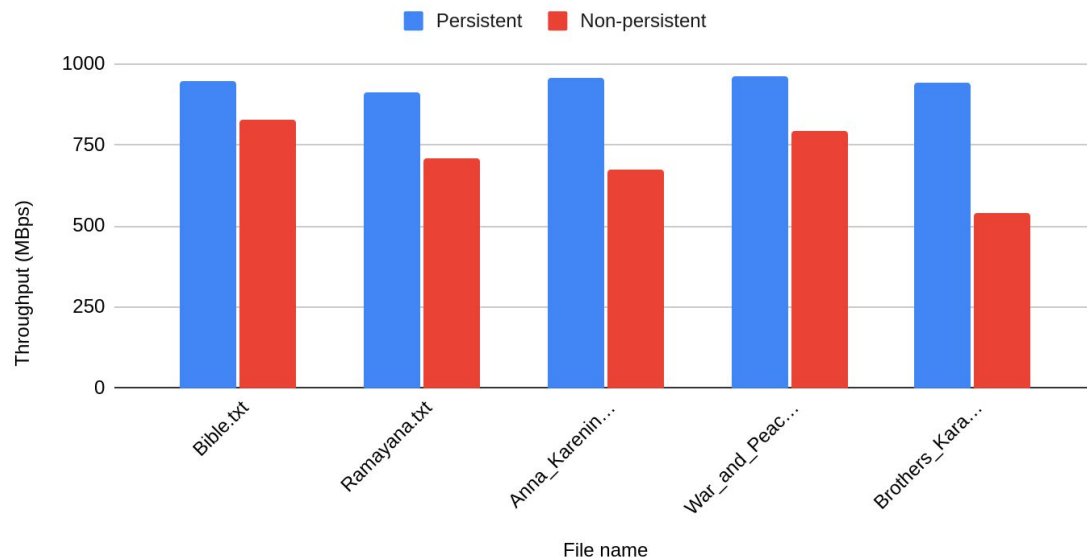
Comparison of persistent and non-persistent in terms of individual file download time - [TCP]



Comparison of non-persistent vs persistent in terms of individual file throughput(MBps):

File name	Persistent	Non-persistent
Bible.txt	948.512504	826.2117167
Ramayana.txt	910.4474834	711.308206
Anna_Karenina.txt	954.901997	676.2848267
War_and_Peace.txt	961.5294791	793.8056589
Brothers_Karamazov.txt	941.1588398	541.7243938

Comparison of persistent and non-persistent in terms of individual file throughput - [TCP]



Inference: We observe that the persistent connection has lower download times and higher throughput, which agrees with what we have learnt. Thus, the persistent connection is useful.

(continued next page)

UDP

Output screenshots for non-persistent:

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient.py 32768 Bible.txt
File found.
Receiving Bible.txt...
Time out: Closing socket

Bible.txt(2359434 Bytes) received. Saved as BibleUDP9099.txt
Elapsed time: 0.002737283706665039
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient.py 32768 Ramayana.txt
File found.
Receiving Ramayana.txt...
Time out: Closing socket

Ramayana.txt(856657 Bytes) received. Saved as RamayanaUDP9131.txt
Elapsed time: 0.0016658306121826172
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient.py 32768 Anna_Karenina.txt
File found.
Receiving Anna_Karenina.txt...
Time out: Closing socket

Anna_Karenina.txt(819200 Bytes) received. Saved as Anna_KareninaUDP9160.txt
Elapsed time: 0.001659393310546875
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$ python3 UDPserver.py 32768
[*] Ready to receive as 127.0.0.1 : 12345
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP/server$

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient.py 32768 War_and_Peace.txt
File found.
Receiving War_and_Peace.txt...
Time out: Closing socket

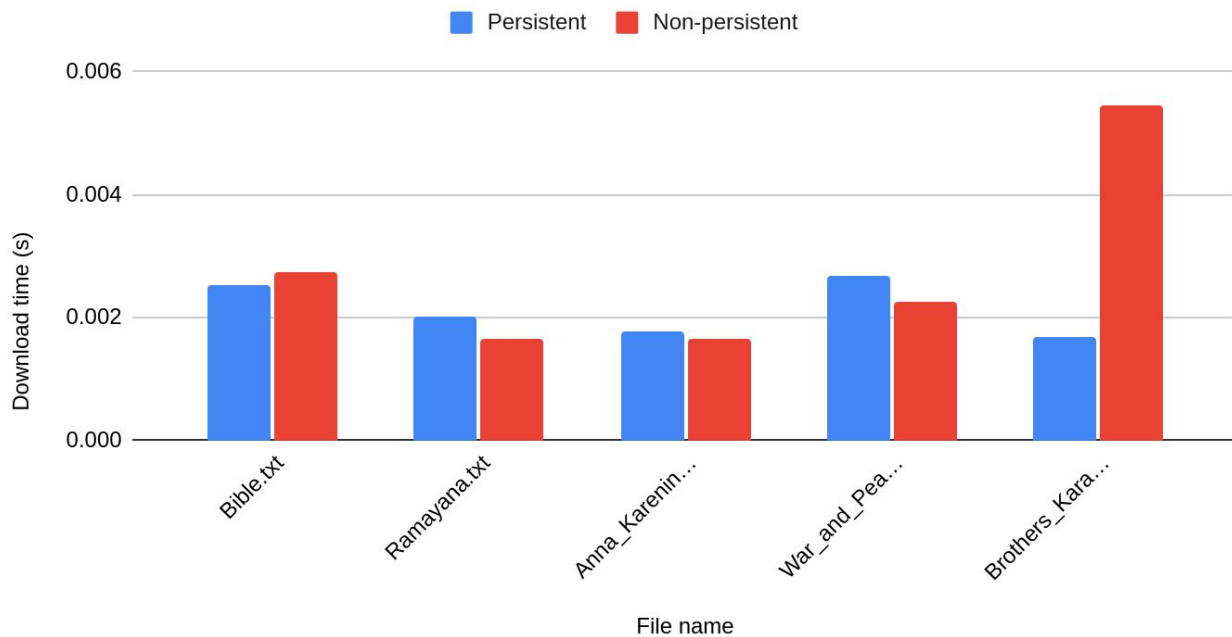
War_and_Peace.txt(1655648 Bytes) received. Saved as War_and_PeaceUDP9311.txt
Elapsed time: 0.0022635459899902344
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$ python3 UDPclient.py 32768 Brothers_Karamazov.txt
File found.
Receiving Brothers_Karamazov.txt...
Time out: Closing socket

Brothers_Karamazov.txt(950272 Bytes) received. Saved as Brothers_KaramazovUDP9358.txt
Elapsed time: 0.005456209182739258
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/UDP$
```

Comparison of persistent vs non-persistent in terms of individual file download time(s):

File name	Persistent	Non-persistent
Bible.txt	0.002516746521	0.002737283707
Ramayana.txt	0.002024650574	0.001665830612
Anna_Karenina.txt	0.00177192688	0.001659393311
War_and_Peace.txt	0.00267624855	0.00226354599
Brothers_Karamazov.txt	0.00167632103	0.005456209183

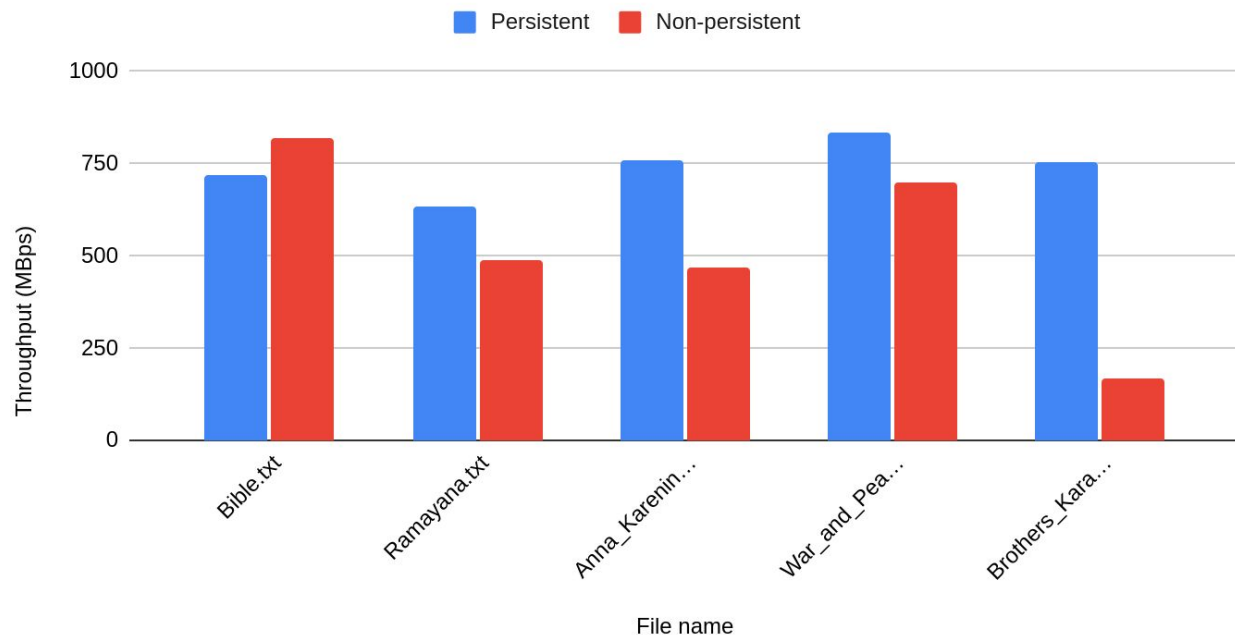
Comparison of persistent and non-persistent in terms of individual file download time - [UDP]



Comparison of non-persistent vs persistent in terms of individual file throughput(MBps):

File name	Persistent	Non-persistent
Bible.txt	720.2281167	822.0308336
Ramayana.txt	635.0339143	490.4290826
Anna_Karenina.txt	760.3439182	470.8045977
War_and_Peace.txt	835.1985746	697.5555087
Brothers_Karamazov.txt	752.8380031	166.0951715

Comparison of persistent and non-persistent in terms of individual file throughput - [UDP]



Inference: We observe that the persistent connection has higher throughput, which agrees with what we have learnt. Thus, the persistent connection is useful. The download time is not a reliable metric here since it is UDP and the entire file may not get transferred.

Concurrent servers: forks vs threads

2

Code for the fork model and thread model are included in the submission as TCP/server/TCPserver_fork.py and TCP/server/TCPserver_thread.py

c)

Fork model

Run the fork model server:

```
cd TCP/server
```

```
python3 TCPserver_fork.py 32768
```

Run the persistent client with 5 filenames as arguments:

cd TCP

python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt
War_and_Peace.txt Brothers_Karamazov.txt

Output screenshots:

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPserver_fork.py 32768
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 57788) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Closing The connection.
[*] Listening as 127.0.0.1 : 12345
█

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt War_and_Peace.txt Brothers_Karamazov.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected(took 8.654594421386719e-05 seconds).
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received in 0.005774259567260742 seconds. Saved as BibleTCP24769.txt
File found.
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received in 0.002615213394165039 seconds. Saved as RamayanaTCP24769.txt
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received in 0.002122640609741211 seconds. Saved as Anna_KareninaTCP24769.txt
File found.
Receiving War_and_Peace.txt...
War_and_Peace.txt(3359584 Bytes) received in 0.004543304443359375 seconds. Saved as War_and_PeaceTCP24769.txt
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received in 0.002964019775390625 seconds. Saved as Brothers_KaramazovTCP24769.txt
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ █
```

Thread model

Run the thread model server:

cd TCP/server

python3 TCPserver_thread.py 32768

Run the persistent client with 5 filenames as arguments:

cd TCP

python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt
War_and_Peace.txt Brothers_Karamazov.txt

(continued next page)

Output screenshots:

```

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP/server$ python3 TCPserver_thread.py 32768
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 57804) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
File is available.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
File is available.
Sending Anna_Karenina.txt(2068079 Bytes)...
Anna_Karenina.txt(2068079 Bytes) sent.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Closing the connection.
^CTraceback (most recent call last):
  File "TCPserver_thread.py", line 76, in <module>
    client socket, address = server socket.accept()

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ python3 TCPclient_persistent.py 32768 Bible.txt Ramayana.txt Anna_Karenina.txt War_and_Peace.txt Brothers_Karamazov.txt
[+] Connecting to 127.0.0.1 : 12345
[+] Connected(took 8.988380432128906e-05 seconds).
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received in 0.005226850509643555 seconds. Saved as BibleTCP24987.txt
File found.
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received in 0.002611875534057617 seconds. Saved as RamayanaTCP24987.txt
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received in 0.0023233890533447266 seconds. Saved as Anna_KareninaTCP24987.txt
File found.
Receiving War_and_Peace.txt...
War_and_Peace.txt(3359584 Bytes) received in 0.003579378128051758 seconds. Saved as War_and_PeaceTCP24987.txt
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received in 0.002386331558227539 seconds. Saved as Brothers_KaramazovTCP24987.txt

```

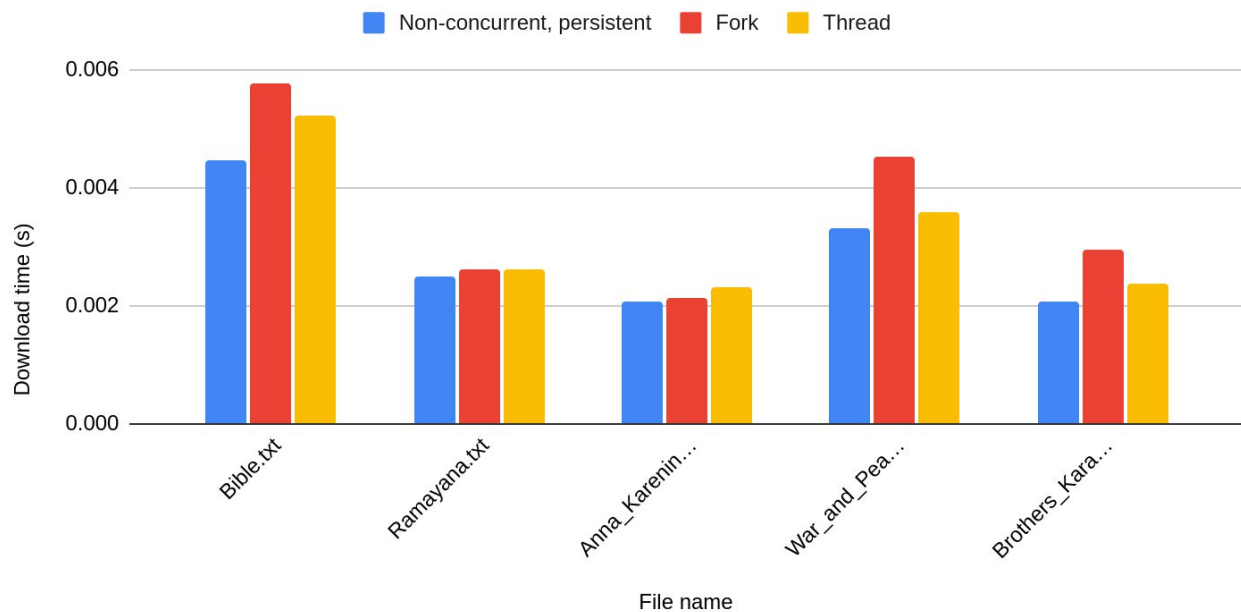
Comparison of total time(including connection setup time) and aggregate throughput for fork, thread models and the non-concurrent, persistent model(used in q1):

	Non-concurrent, persistent	Fork model	Thread model
Total time (s)	0.01453900337	0.01810598373	0.01621770859
Aggregate throughput (MBps)	939.6499893	754.5336704	842.386228

Comparison of fork, thread models and the non-concurrent, persistent model in terms of individual file download time(**s**):

Filename	Non-concurrent, persistent	Fork model	Thread model
Bible.txt	0.004480838776	0.005774259567	0.00522685051
Ramayana.txt	0.002510547638	0.002615213394	0.002611875534
Anna_Karenina.txt	0.002065420151	0.00212264061	0.002323389053
War_and_Peace.txt	0.003332138062	0.004543304443	0.003579378128
Brothers_Karamazov.txt	0.002071380615	0.002964019775	0.002386331558

Comparison between persistent, fork and thread models in terms of individual file download time

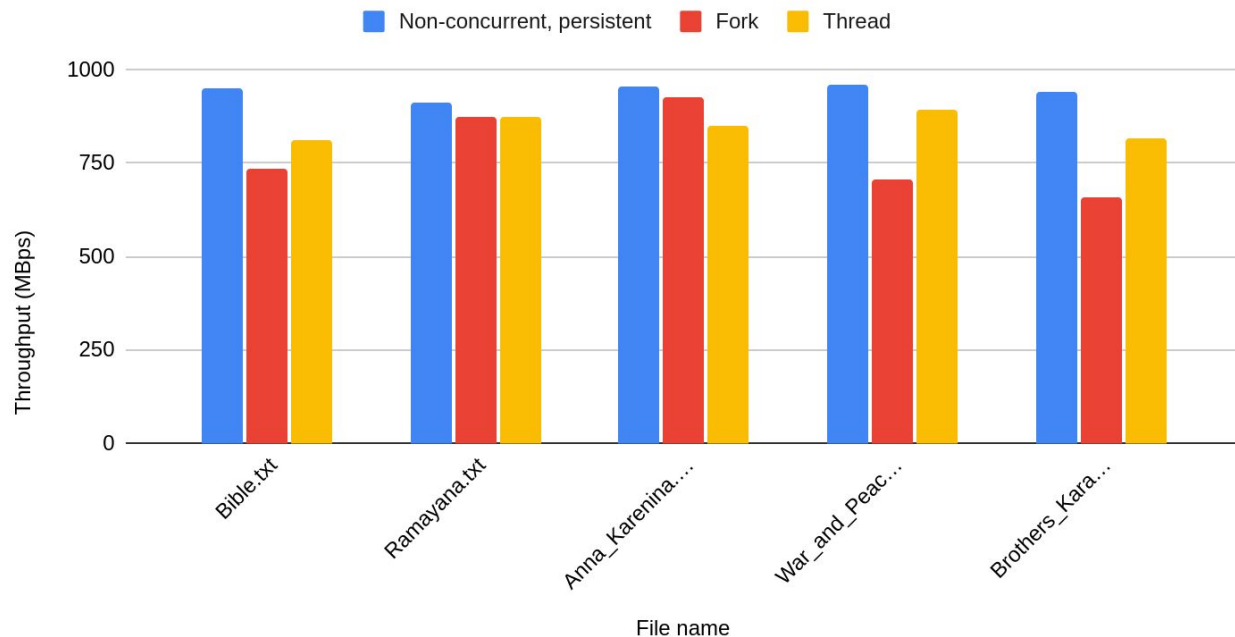


Comparison of fork, thread models and the non-concurrent, persistent model in terms of individual file throughput **(MBps)**:

Filename	Non-concurrent, persistent	Fork model	Thread model
Bible.txt	948.512504	736.0478963	813.1343338
Ramayana.txt	910.4474834	874.0096636	875.1266089
Anna_Karenina.txt	954.901997	929.1605077	848.8779887
War_and_Peace.txt	961.5294791	705.202351	895.1133018
Brothers_Karamazov.txt	941.1588398	657.7210425	816.9435508

(continued next page)

Comparison between persistent, fork and thread models in terms of individual file throughput



Inference: We see that the non-concurrent, persistent model used in Q1 has the best performance followed by the thread model and lastly the fork model. This is because in this question, we did not make use of the support for concurrency that is available in the thread and fork models, since we had only a single client downloading files. Thus the overheads of supporting concurrency caused them to perform worse than the simple model. However, if we had more clients at the same time, we can expect the concurrent models to perform better.

d)

Fork model

Run the fork model server:

```
cd TCP/server
```

```
python3 TCPserver_fork.py 32768
```

Run the run_multiple_clients.sh file to start 5 non-persistent TCP clients simultaneously:

```
cd TCP
```

```
./run_multiple_clients.sh
```


Fork model output screenshots:

```
[+] ('127.0.0.1', 58774) is connected.
File is available.
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 58776) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
File is available.
Sending Anna Karenina.txt(2068079 Bytes)...
Anna Karenina.txt(2068079 Bytes) sent.
Sending Ramayana.txt(2396753 Bytes)...
Ramayana.txt(2396753 Bytes) sent.
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 58778) is connected.
[+] ('127.0.0.1', 58780) is connected.
[*] Listening as 127.0.0.1 : 12345
[*] Listening as 127.0.0.1 : 12345
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
File is available.
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
Sending War_and_Peace.txt(3359584 Bytes)...
War_and_Peace.txt(3359584 Bytes) sent.
Closing the connection.
[*] Listening as 127.0.0.1 : 12345
[]

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ ./run_multiple_clients.sh
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ [+] Connecting to 127.0.0.1 : 12345
[+] Connected.
[+] Connected.
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
File found.
File found.
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP29444.txt. Elapsed time : 0.0048220157623291016 seconds
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP29443.txt. Elapsed time: 0.008872509002685547 seconds
[+] Connecting to 127.0.0.1 : 12345
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
[+] Connected.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP29442.txt. Elapsed time: 0.010697126388549805 seconds
File found.
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP29446.txt. Elapsed time: 0.005471706390380859 seconds
Receiving War_and_Peace.txt...
War_and_Peace.txt(3342336 Bytes) received. Saved as War_and_PeaceTCP29445.txt. Elapsed time : 0.009490489959716797 seconds
```

Thread model

Run the thread model server:

cd TCP/server

python3 TCPserver_thread.py 32768

Run the run_multiple_clients.sh file to start 5 non-persistent TCP clients simultaneously:

cd TCP

./run_multiple_clients.sh

Thread model output screenshots:

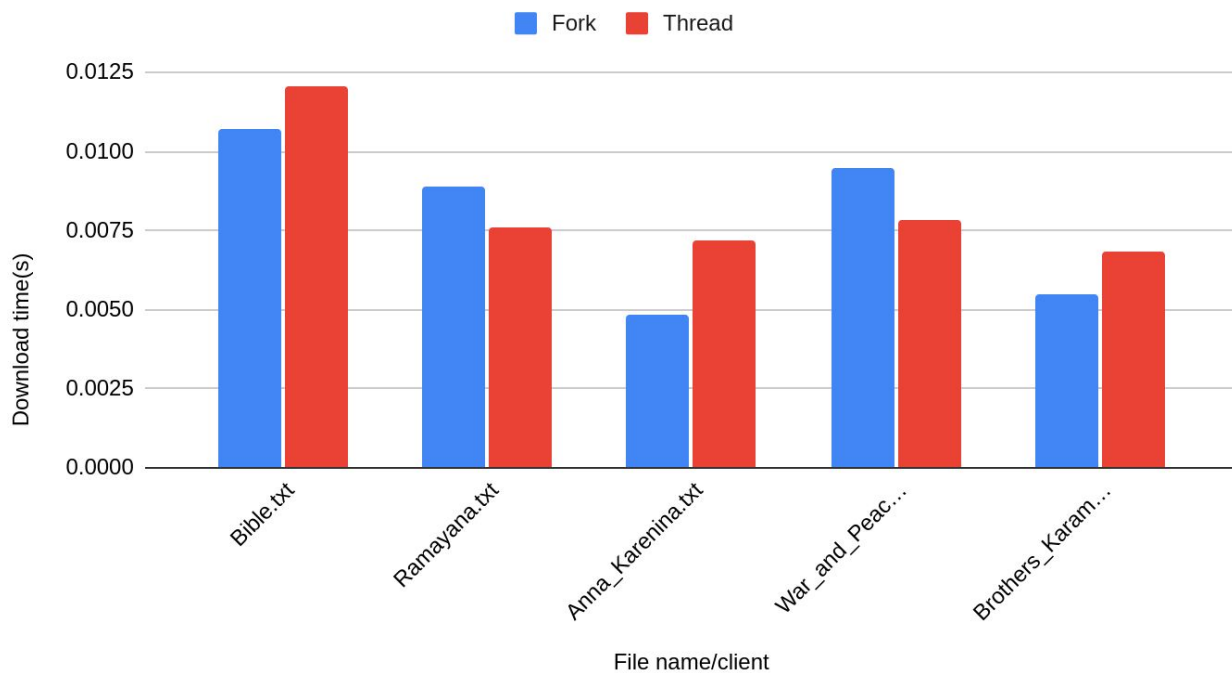
```
[+] ('127.0.0.1', 58810) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
Sending Brothers_Karamazov.txt(2044197 Bytes)...[+] ('127.0.0.1', 58812) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
[+] ('127.0.0.1', 58814) is connected.
[*] Listening as 127.0.0.1 : 12345
Sending Anna_Karenina.txt(2068079 Bytes)...File is available.
Sending Bible.txt(4456586 Bytes)...
Brothers_Karamazov.txt(2044197 Bytes) sent.
Closing the connection.
Bible.txt(4456586 Bytes) sent.
Closing the connection.
Anna_Karenina.txt(2068079 Bytes) sent.
Closing the connection.
[+] ('127.0.0.1', 58816) is connected.
[*] Listening as 127.0.0.1 : 12345
[+] ('127.0.0.1', 58818) is connected.
[*] Listening as 127.0.0.1 : 12345
File is available.
File is available.
Sending War_and_Peace.txt(3359584 Bytes)...Sending Ramayana.txt(2396753 Bytes)...Closing the connection.
Ramayana.txt(2396753 Bytes) sent.
War_and_Peace.txt(3359584 Bytes) sent.
Closing the connection.
Closing the connection.
[]

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ ./run_multiple_clients.sh
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/TCP$ [+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
File found.
File found.
Receiving Brothers_Karamazov.txt...
Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP29707.txt. Elapsed time: 0.006810188293457031 seconds
Receiving Anna_Karenina.txt...
Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP29705.txt. Elapsed time : 0.0072138309478759766 seconds
[+] Connecting to 127.0.0.1 : 12345
[+] Connecting to 127.0.0.1 : 12345
[+] Connected.
[+] Connected.
File found.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP29703.txt. Elapsed time: 0.012053728103637695 seconds
Receiving Ramayana.txt...
Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP29704.txt. Elapsed time: 0.007582426071166992 seconds
Receiving War_and_Peace.txt...
War_and_Peace.txt(3342336 Bytes) received. Saved as War_and_PeaceTCP29706.txt. Elapsed time : 0.007812023162841797 seconds
```

Comparison of download time (**s**) for fork and thread models:

File requested by the client	Fork model	Thread model
Bible.txt	0.01069712639	0.0120537281
Ramayana.txt	0.008872509003	0.007582426071
Anna_Karenina.txt	0.004822015762	0.007213830948
War_and_Peace.txt	0.00949048996	0.007812023163
Brothers_Karamazov.txt	0.00547170639	0.006810188293

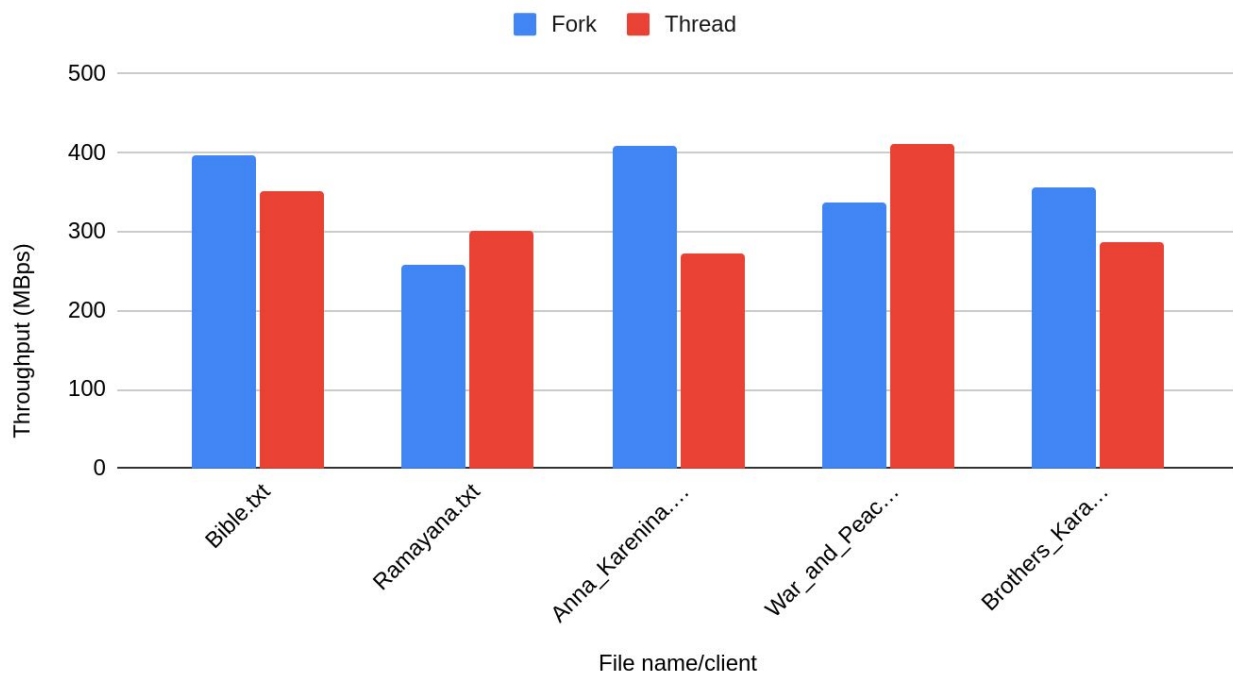
Comparison of fork and thread models based on download time



Comparison of throughput(**MBps**) for fork and thread models:

File requested by the client	Fork model	Thread model
Bible.txt	397.3152651	352.5989279
Ramayana.txt	257.6184232	301.4499261
Anna_Karenina.txt	409.0143881	273.4017252
War_and_Peace.txt	337.5957393	410.1305011
Brothers_Karamazov.txt	356.2870588	286.2620081

Comparison of fork and thread models based on throughput



Comparison of completion time(**s**) and aggregate throughput(**MBps**) for fork and thread models:

	Fork model	Thread model
Completion Time(s)	0.01069712639	0.0120537281
Aggregate Throughput (MBps)	1277.125638	1133.389956

Assuming the completion time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the completion time.

Observations: The fork model works better for me overall. However, the difference is very small and in the case of individual files, sometimes the thread model does better and sometimes the fork model does better. Thus, they are both roughly similar in terms of performance. However, since forking requires more resources, thread model is generally preferred over fork model.

Migrating to Mininet

3

Changes required for the client and server to operate with Mininet:

1. Modify client and server code such that the IP address of the server is given as argument while running the script, instead of being hard-coded.

The server I am using for Mininet experiments is the thread-model server, and the client being used is a non-persistent client(since each client will only download 1 file, and also because we need to compare 3e with 2d). The server and client that can be run with Mininet is Mininet/server/TCPserver_thread_mn.py and Mininet/TCPclient_mn.py

e)

I did this question using the Mininet Python API instead of using the CLI since we have to start the 5 clients simultaneously, which is not possible using the Mininet CLI.

The python script Mininet/3e_and_3f.py can be used for questions 3e and 3f. It uses the Mininet Python API and creates the required topology(single, with 1 switch and 6 hosts) and runs the server on host 1. It then starts the clients on hosts 2-6. This is done using popen(). Since popen() is not a blocking function and since I measured the time taken for a popen() call to be around 1ms, we can use 5 popen() calls to start the clients simultaneously. The fact that it is similar to having threads can be seen by the fact that host 4 or host 5 may start the client faster than host 2 even though popen() was called on host 2 first. So, the 5 clients can be considered to be starting simultaneously.

The names of the files to be downloaded by the clients should be provided as arguments when running 3e_and_3f.py. Make sure to use 'sudo' while running 3e_and_3f.py since Mininet needs to be run as root.

Run 3e_and_3f.py with proper arguments:

```
cd Mininet
```

```
sudo python3 3e_and_3f.py Bible.txt Ramayana.txt Anna_Karenina.txt War_and_Peace.txt  
Brothers_Karamazov.txt
```

(continued next page)

Output screenshot:

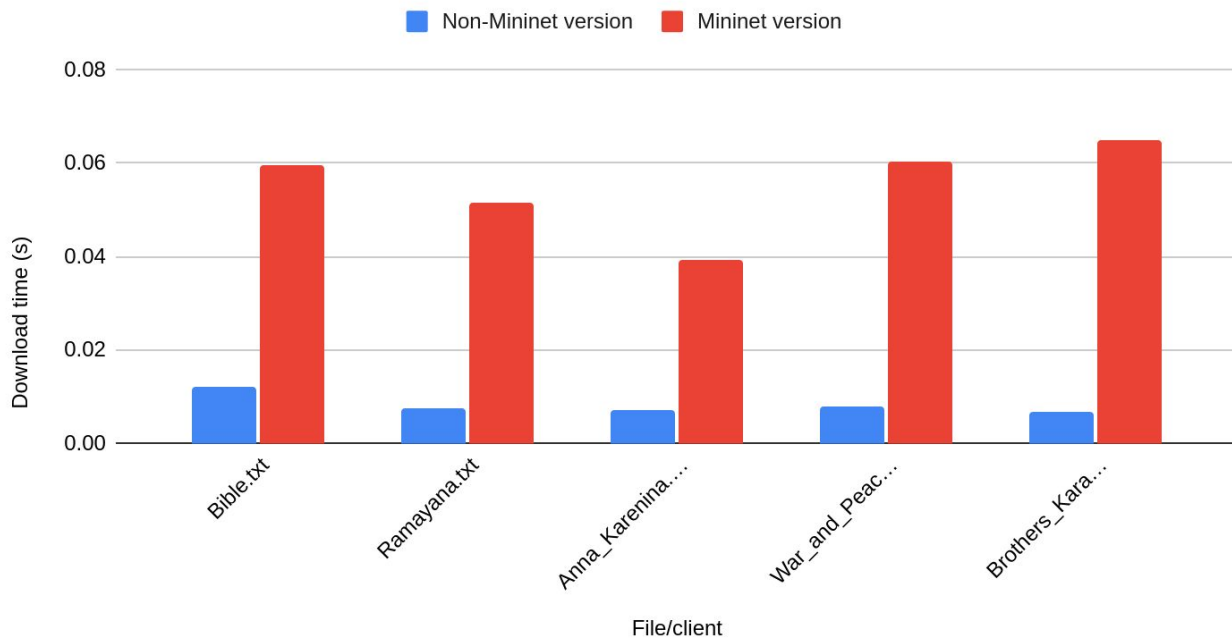
```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3e_and_3f.py Bible.txt Ramayana.
txt Anna_Karenina.txt War_and_Peace.txt Brothers_Karamazov.txt
[sudo] password for chris:
h4: [+] Connecting to 10.0.0.1 : 12345
h4: [+] Connected.
h4: File found.
h4: Receiving Anna_Karenina.txt...
h4: Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP114097.txt. Elapsed time: 0.03924751281738281 seconds
h3: [+] Connecting to 10.0.0.1 : 12345
h3: [+] Connected.
h3: File found.
h3: Receiving Ramayana.txt...
h3: Ramayana.txt(2254288 Bytes) received. Saved as RamayanaTCP114096.txt. Elapsed time: 0.051526784896850586 seconds
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP114095.txt. Elapsed time: 0.05952048301696777 seconds
h5: [+] Connecting to 10.0.0.1 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving War and Peace.txt...
h5: War_and_Peace.txt(3342336 Bytes) received. Saved as War_and_PeaceTCP114098.txt. Elapsed time: 0.060507774353027344 seconds
h6: [+] Connecting to 10.0.0.1 : 12345
h6: [+] Connected.
h6: File found.
h6: Receiving Brothers_Karamazov.txt...
h6: Brothers_Karamazov.txt(1966080 Bytes) received. Saved as Brothers_KaramazovTCP114099.txt. Elapsed time: 0.06491565704345703 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

Comparison of non-mininet version and mininet version based on download time (s) for each client:

File requested by the client	Non-mininet (download time,s)	Mininet(download time,s)
Bible.txt	0.0120537281	0.05952048302
Ramayana.txt	0.007582426071	0.0515267849
Anna_Karenina.txt	0.007213830948	0.03924751282
War_and_Peace.txt	0.007812023163	0.06050777435
Brothers_Karamazov.txt	0.006810188293	0.06491565704

(continued next page)

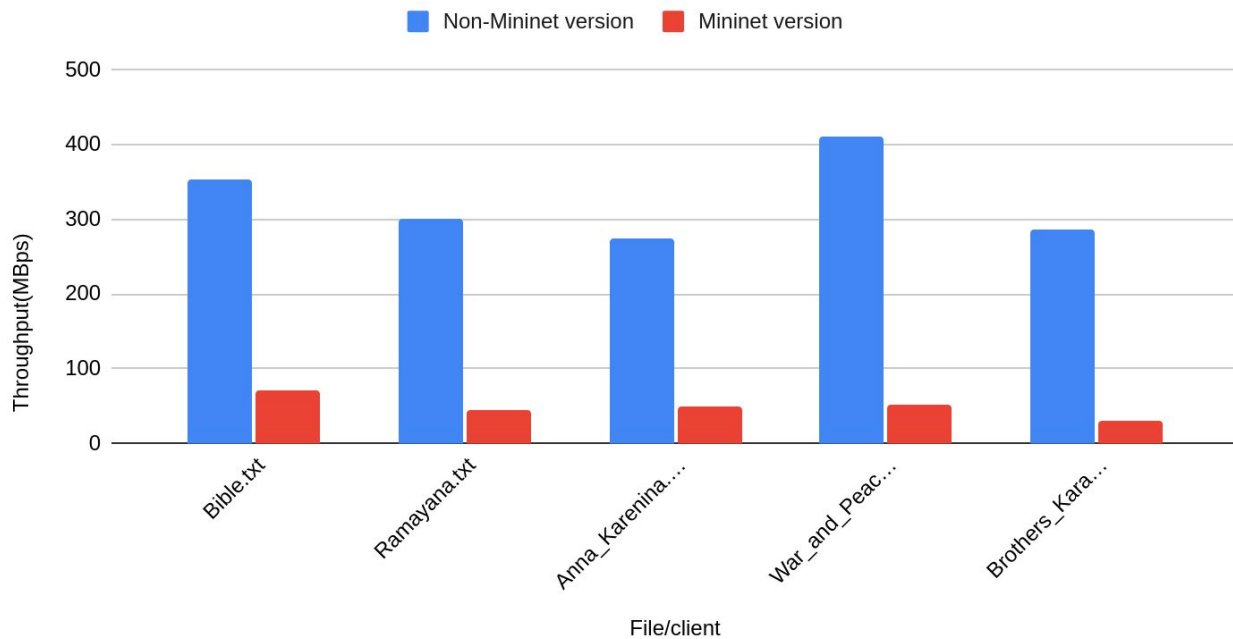
Comparison of non-mininet version and mininet version based on download time (s) for each client



Comparison of non-mininet version and mininet version based on throughput(MBps) for each client:

File requested by the client	Non-mininet(throughput, MBps)	Mininet(throughput, MBps)
Bible.txt	352.5989279	71.40620156
Ramayana.txt	301.4499261	44.35987581
Anna_Karenina.txt	273.4017252	50.25219906
War_and_Peace.txt	410.1305011	52.95102999
Brothers_Karamazov.txt	286.2620081	30.0312477

Comparison of non-mininet version and mininet version based on throughput(MBps) for each client



	Non-mininet	Mininet
Total Time (s)	0.0120537281	0.06491565704
Aggregate Throughput (MBps)	1133.389956	210.4511452

Assuming the completion time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the completion time.

a) Observations: We see that the non-mininet version is around 5 times better in terms of both throughput and download time(both individual files and overall). This is due to Mininet's constraints which means that the resources available are lesser. Also, in the non-Mininet version, we are running the server and client on the same machine, while in Mininet, we model the server to be on a separate host while clients are on other hosts.

f)

I did this question using the Mininet Python API instead of using the CLI since we have to start the 5 clients simultaneously, which is not possible using the Mininet CLI.

We can reuse the python script 3e_and_3f.py by providing the argument Bible.txt(largest file) 5 times. This will start 5 clients simultaneously and each of them will download Bible.txt

Run 3e_and_3f.py with proper arguments:

cd Mininet

sudo python3 3e_and_3f.py Bible.txt Bible.txt Bible.txt Bible.txt Bible.txt

Output screenshot:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3e_and_3f.py Bible.txt Bible.txt
Bible.txt Bible.txt Bible.txt
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4423680 Bytes) received. Saved as BibleTCP114671.txt. Elapsed time: 0.02060866355895996 seconds
h3: [+] Connecting to 10.0.0.1 : 12345
h3: [+] Connected.
h3: File found.
h3: Receiving Bible.txt...
h3: Bible.txt(4456586 Bytes) received. Saved as BibleTCP114673.txt. Elapsed time: 0.05035996437072754 seconds
h5: [+] Connecting to 10.0.0.1 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving Bible.txt...
h5: Bible.txt(4226160 Bytes) received. Saved as BibleTCP114675.txt. Elapsed time: 0.05570697784423828 seconds
h6: [+] Connecting to 10.0.0.1 : 12345
h6: [+] Connected.
h6: File found.
h6: Receiving Bible.txt...
h6: Bible.txt(4227072 Bytes) received. Saved as BibleTCP114676.txt. Elapsed time: 0.056529998779296875 seconds
h4: [+] Connecting to 10.0.0.1 : 12345
h4: [+] Connected.
h4: File found.
h4: Receiving Bible.txt...
h4: Bible.txt(4423680 Bytes) received. Saved as BibleTCP114674.txt. Elapsed time: 0.07497739791870117 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

Download time and throughput for each of the 5 clients:

File requested by the client	Download Time(s)	Throughput(MBps)
Bible.txt	0.02060866356	206.2303358
Bible.txt	0.05035996437	84.39504793
Bible.txt	0.07497739792	56.68550423
Bible.txt	0.05570697784	76.29442076
Bible.txt	0.05652999878	75.18364937

Total Time(s)	0.07497739792
Aggregate Throughput(MBps)	283.4275212

Assuming the total time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the total time.

a) Observations: We see that there are variations even though each client is downloading the same file and each client is equal on the topology.

g)

The command:

```
sudo mn --link tc,bw=10
```

will create a topology with 2 hosts and 1 switch with the hosts connected to the switch through 10 Mbps links, as required in the question.

Commands:

on terminal:

```
cd Mininet
```

```
sudo mn --link tc,bw=10 // 10 Mbps
```

on Mininet:

```
xterm h1
```

on xterm:

```
python3 server/TCPserver_thread_mn.py 8 10.0.0.1
```

on Mininet:

```
h2 python3 TCPclient_mn.py 8 h1 Bible.txt
```

(Similarly for other bandwidths)

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=10
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit) (10.00Mbit) (h1, s1) (10.00Mbit) (10.00Mbit) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (10.00Mbit) (10.00Mbit)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP118268.txt. Elapsed time: 3.7473304271698 seconds
mininet> 
```

```
"Node: h1"
root@chris-Aspire-A515-51G:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1 0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55062) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=100
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(100.00Mbit) (100.00Mbit) (h1, s1) (100.00Mbit) (100.00Mbit) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (100.00Mbit) (100.00Mbit)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP118960.txt. Elapsed time: 0.6886024475097656 seconds
mininet> 
```

```
"Node: h1"
root@chris-Aspire-A515-51G:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1 0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55100) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

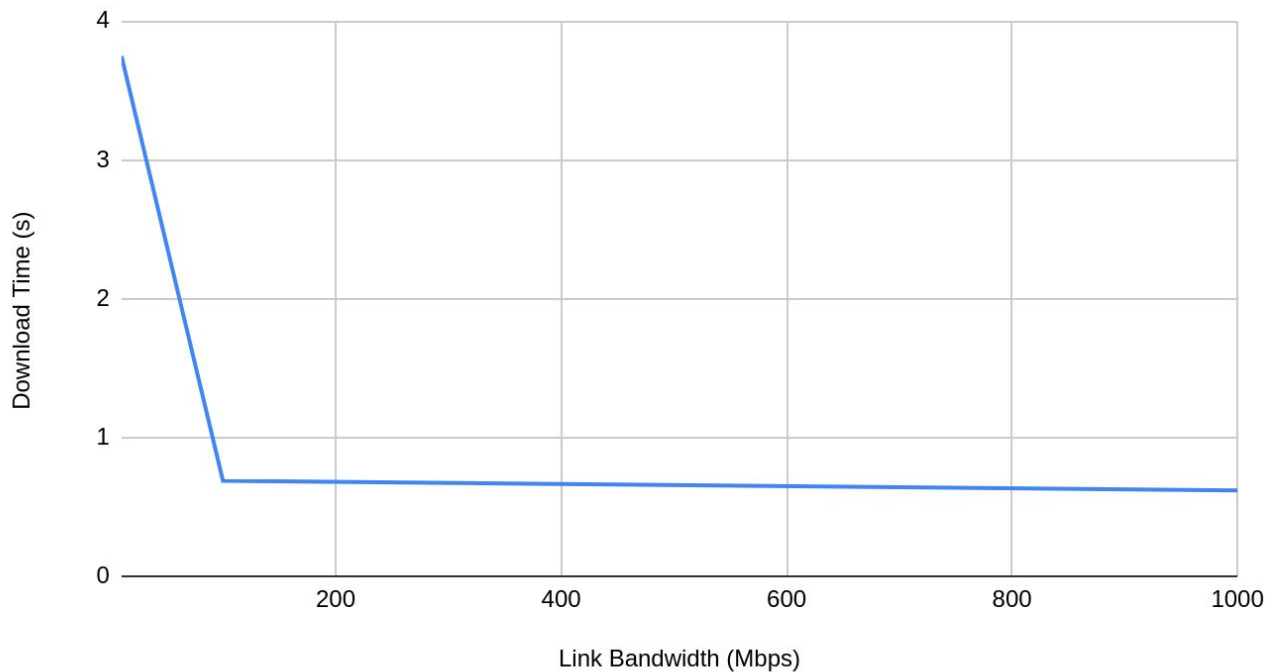
```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit) (1000.00Mbit) (h1, s1) (1000.00Mbit) (1000.00Mbit) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit) (1000.00Mbit)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP120342.txt. Elapsed time: 0.6191756725311279 seconds
mininet> 
```

```
"Node: h1"
root@chris-Aspire-A515-51G:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1 0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55272) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

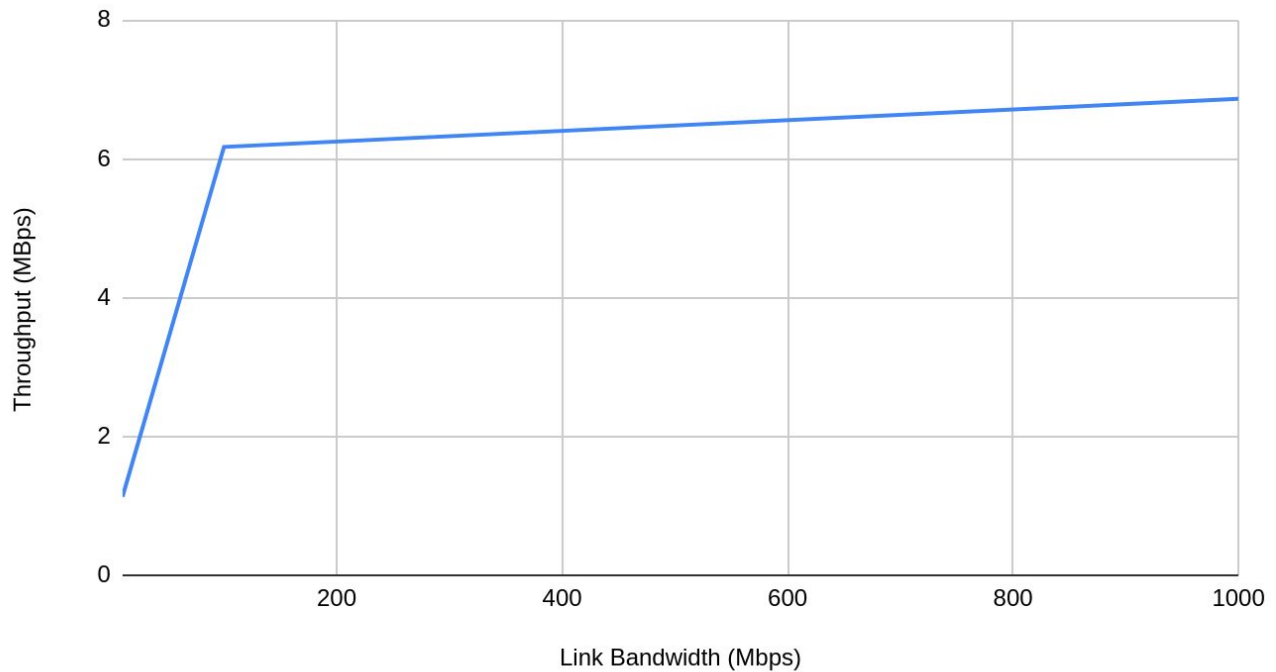

Link Bandwidth	Download Time(s)	Throughput(MBps)
10 Mbps	3.747330427	1.134175833
100 Mbps	0.6886024475	6.172112258
1 Gbps	0.6191756725	6.864177318

Observations: With increase in the link bandwidth, the download time decreases and the throughput increases as can be seen from the above table and following plots. The best throughput was for a bandwidth of 1 Gbps.

Download time vs Link Bandwidth



Throughput vs Link Bandwidth



h)

The best throughput was for a bandwidth of 1 Gbps.

The command:

```
sudo mn --link tc,bw=1000,delay=1ms
```

will create a topology with 2 hosts and 1 switch with the hosts connected to the switch through 1 Gbps links having 1ms delay, as required in the question.

Commands:

on terminal:

```
cd Mininet
```

```
sudo mn --link tc,bw=1000,delay=1ms // 1ms delay
```

on Mininet:

```
xterm h1
```

on xterm:

```
python3 server/TCPserver_thread_mn.py 8 10.0.0.1
```

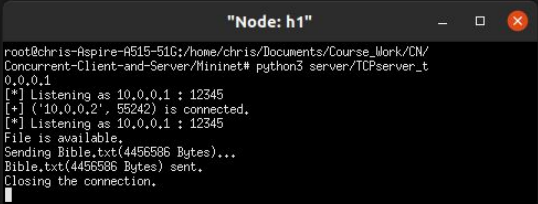
on Mininet:

h2 python3 TCPclient_mn.py 8 h1 Bible.txt

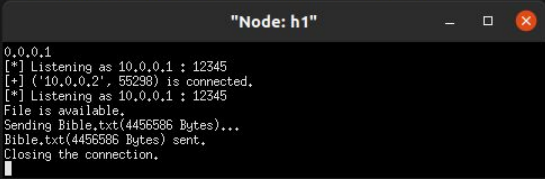
(Similarly for other values of delay)

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,delay=1ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 1ms delay) (1000.00Mbit 1ms delay) (h1, s1) (1000.00Mbit 1ms delay) (1000.00Mbit 1ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 1ms delay) (1000.00Mbit 1ms delay)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP119946.txt. Elapsed time: 0.6781575679779053 seconds
mininet> []
```



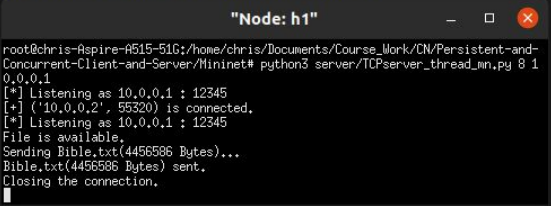
```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,delay=2ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 2ms delay) (1000.00Mbit 2ms delay) (h1, s1) (1000.00Mbit 2ms delay) (1000.00Mbit 2ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 2ms delay) (1000.00Mbit 2ms delay)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP120707.txt. Elapsed time: 0.7050926685333252 seconds
mininet> []
```



```

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,delay=5ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 5ms delay) (1000.00Mbit 5ms delay) (h1, s1) (1000.00Mbit 5ms delay) (1000.00Mbit 5ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 5ms delay) (1000.00Mbit 5ms delay)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP121072.txt. Elapsed time: 0.8294103145599365 seconds
mininet>

```



```

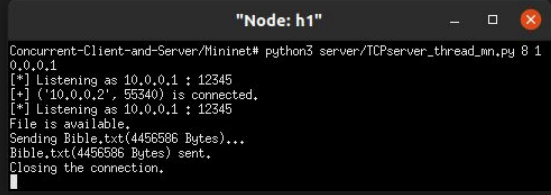
"Node: h1"
root@chris-Aspire-A515-516:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1 0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[*] ('10.0.0.2', 55320) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.

```

```

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,delay=10ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 10ms delay) (1000.00Mbit 10ms delay) (h1, s1) (1000.00Mbit 10ms delay) (1000.00Mbit 10ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 10ms delay) (1000.00Mbit 10ms delay)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP121429.txt. Elapsed time: 1.0212225914001465 seconds
mininet>

```



```

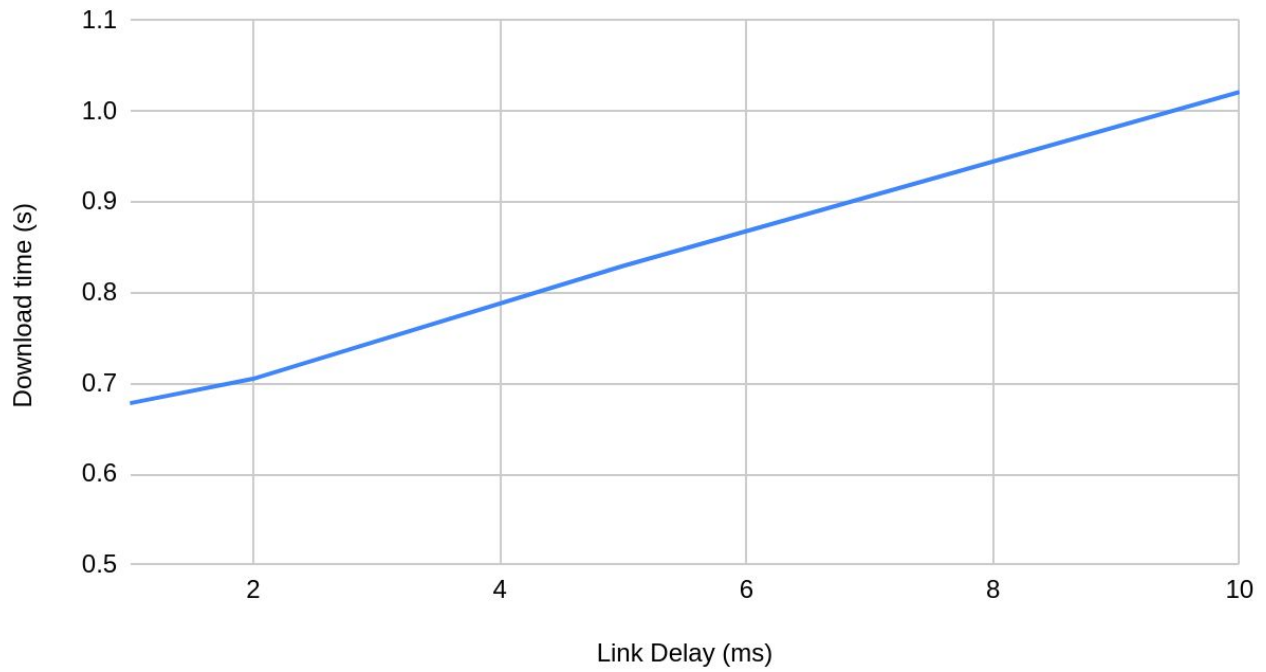
"Node: h1"
Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1 0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[*] ('10.0.0.2', 55340) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.

```

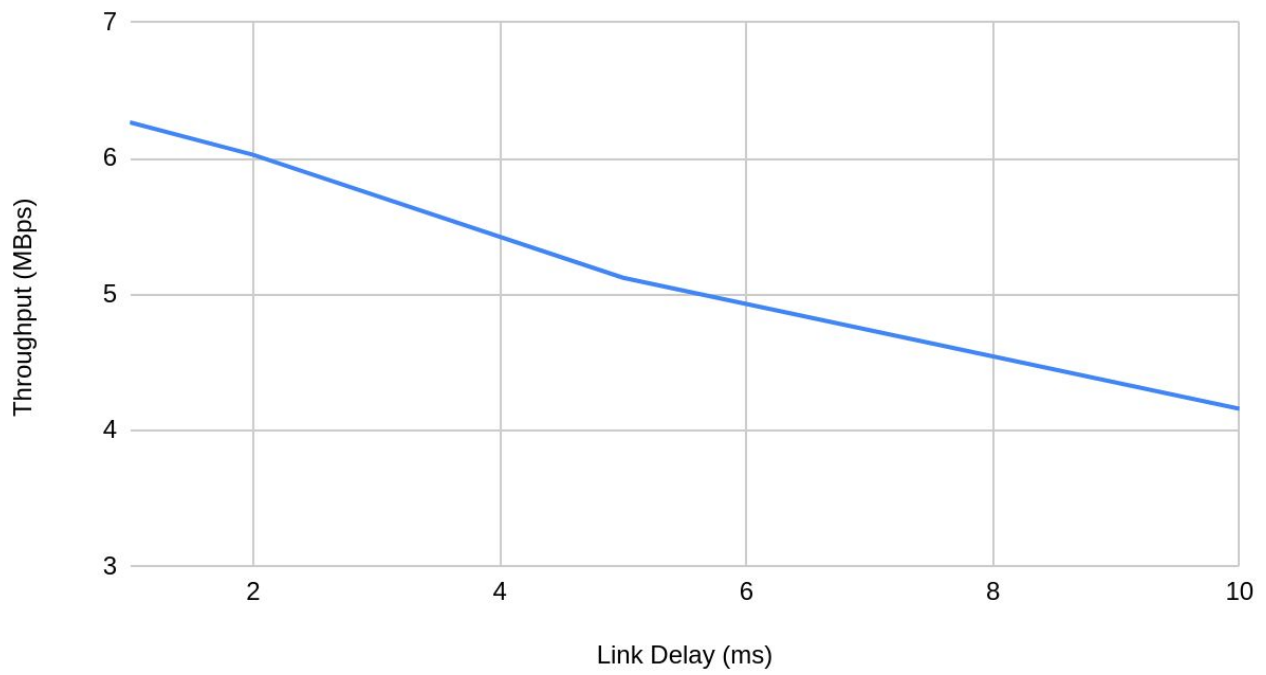
Delay (ms)	Time(s)	Throughput(MBps)
1	0.678157568	6.26717419
2	0.7050926685	6.027763153
5	0.8294103146	5.12428111
10	1.021222591	4.161807272

Observations: With increase in the link delay, the download time increases and the throughput decreases as can be seen from the above table and following plots.

Download time vs Link Delay



Throughput vs Link Delay



i) The best throughput was for a bandwidth of 1 Gbps.

The command:

```
sudo mn --link tc,bw=1000,loss=1
```

will create a topology with 2 hosts and 1 switch with the hosts connected to the switch through 1 Gbps links having 1% loss, as required in the question.

Commands:

on terminal:

```
cd Mininet
```

```
sudo mn --link tc,bw=1000,loss=1 // 1% loss
```

on Mininet:

```
xterm h1
```

on xterm:

```
python3 server/TCPserver_thread_mn.py 8 10.0.0.1
```

on Mininet:

```
h2 python3 TCPclient_mn.py 8 h1 Bible.txt
```

(Similarly for other values of loss)

Output screenshots:

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,loss=1
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 1.00000% loss) (1000.00Mbit 1.00000% loss) (h1, s1) (1000.00Mbit 1.00000% loss) (1000.00Mbit 1.00000% loss) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 1.00000% loss) (1000.00Mbit 1.00000% loss)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP121832.txt. Elapsed time: 1.6699624061584473 seconds
mininet> 
```

"Node: h1"

```
Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1
0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55404) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,loss=2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 2.00000% loss) (1000.00Mbit 2.00000% loss) (h1, s1) (1000.00Mbit 2.00000% loss) (1000.00Mbit 2.00000% loss) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 2.00000% loss) (1000.00Mbit 2.00000% loss)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP12228.txt. Elapsed time: 4.338545799255371 seconds
mininet> 
```

"Node: h1"

```
root@chris-Aspire-A515-516:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1
0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55434) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

```
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo mn --link tc,bw=1000,loss=5
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(1000.00Mbit 5.00000% loss) (1000.00Mbit 5.00000% loss) (h1, s1) (1000.00Mbit 5.00000% loss) (1000.00Mbit 5.00000% loss) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (1000.00Mbit 5.00000% loss) (1000.00Mbit 5.00000% loss)
*** Starting CLI:
mininet> xterm h1
mininet> h2 python3 TCPclient_mn.py 8 h1 Bible.txt
[+] Connecting to 10.0.0.1 : 12345
[+] Connected.
File found.
Receiving Bible.txt...
Bible.txt(4456586 Bytes) received. Saved as BibleTCP122649.txt. Elapsed time: 15.010062217712402 seconds
mininet> 
```

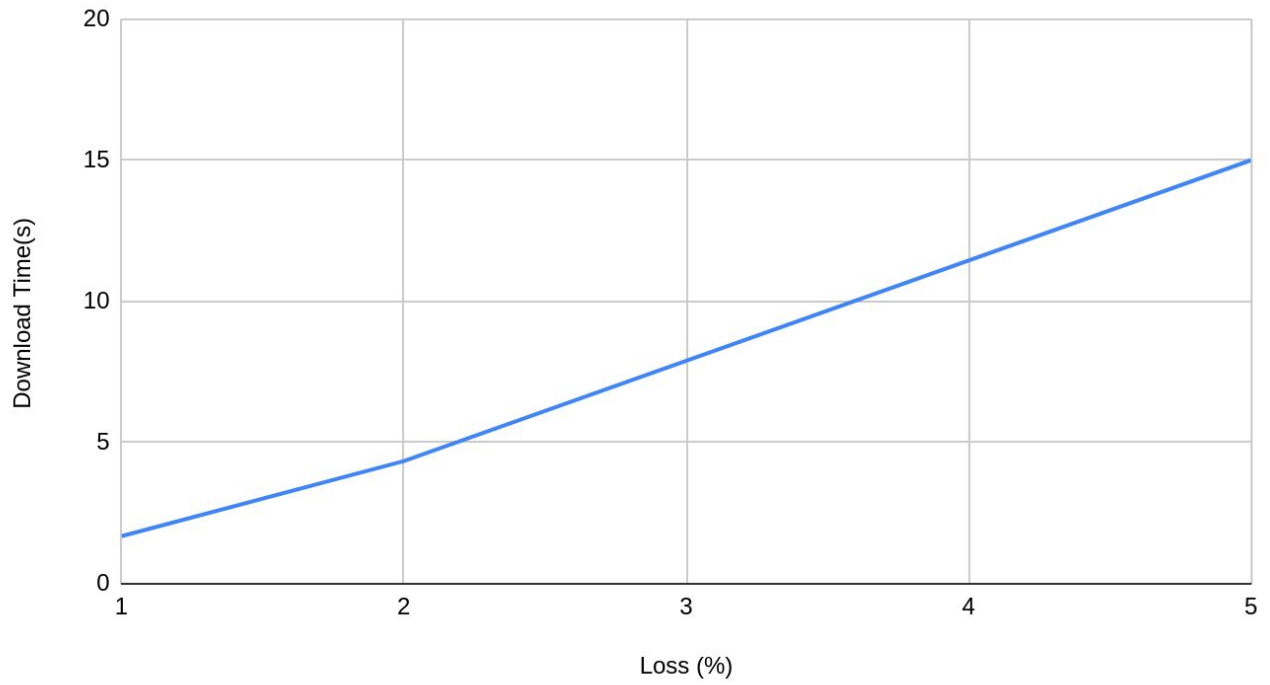
"Node: h1"

```
root@chris-Aspire-A515-516:/home/chris/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet# python3 server/TCPserver_thread_mn.py 8 1
0.0.0.1
[*] Listening as 10.0.0.1 : 12345
[+] ('10.0.0.2', 55454) is connected.
[*] Listening as 10.0.0.1 : 12345
File is available.
Sending Bible.txt(4456586 Bytes)...
Bible.txt(4456586 Bytes) sent.
Closing the connection.
```

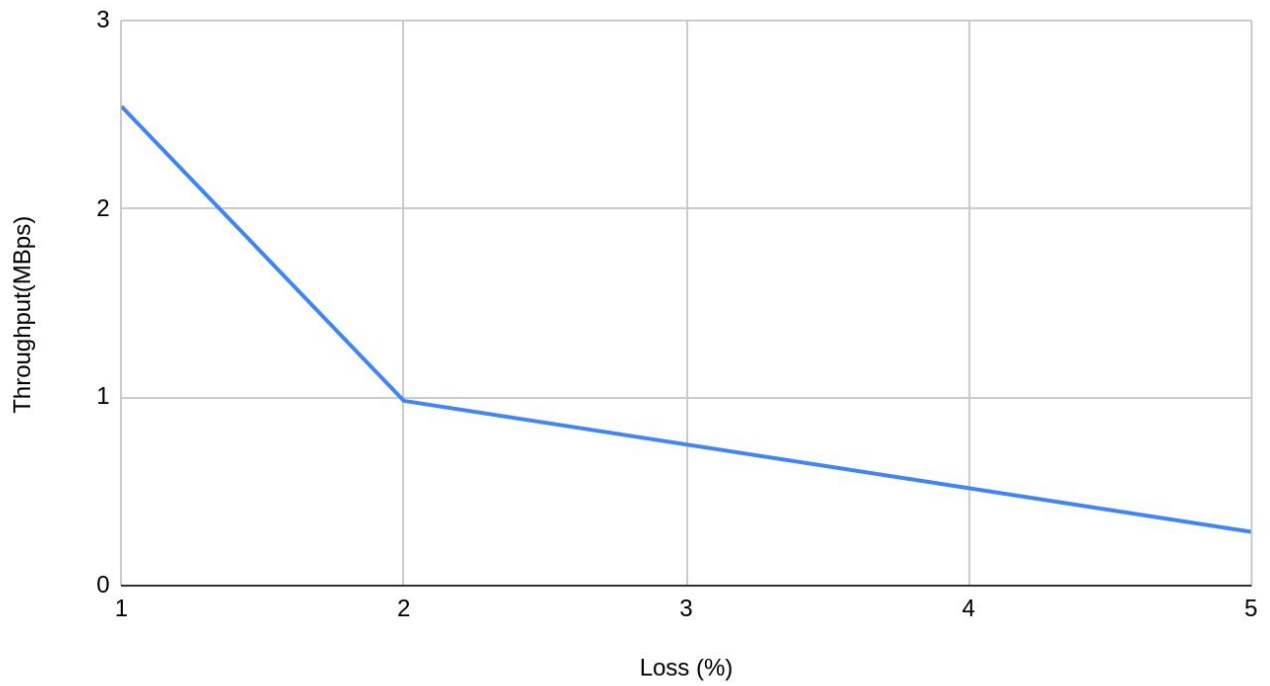
Loss (%)	Download Time(s)	Throughput(MBps)
1	1.669962406	2.545046279
2	4.338545799	0.979621238
5	15.01006222	0.2831521646

Observations: With increase in the loss, the download time increases and the throughput decreases as can be seen from the above table and following plots.

Download Time vs Loss %



Throughput vs Loss %



j) I did this question using the Mininet Python API instead of using the CLI. The python script Mininet/3j.py uses the Mininet Python API and creates the topology with the required number of switches and runs the server on host 1 and the client on host 2. The number of switches should be provided as an argument while running 3j.py. Make sure to use 'sudo' while running 3j.py since Mininet needs to be run as root.

Run 3j.py with proper arguments:

cd Mininet

sudo python3 3j.py 2 // 2 switches

sudo python3 3j.py 4 // 4 switches

sudo python3 3j.py 6 // 6 switches

sudo python3 3j.py 8 // 8 switches

sudo python3 3j.py 10 // 10 switches

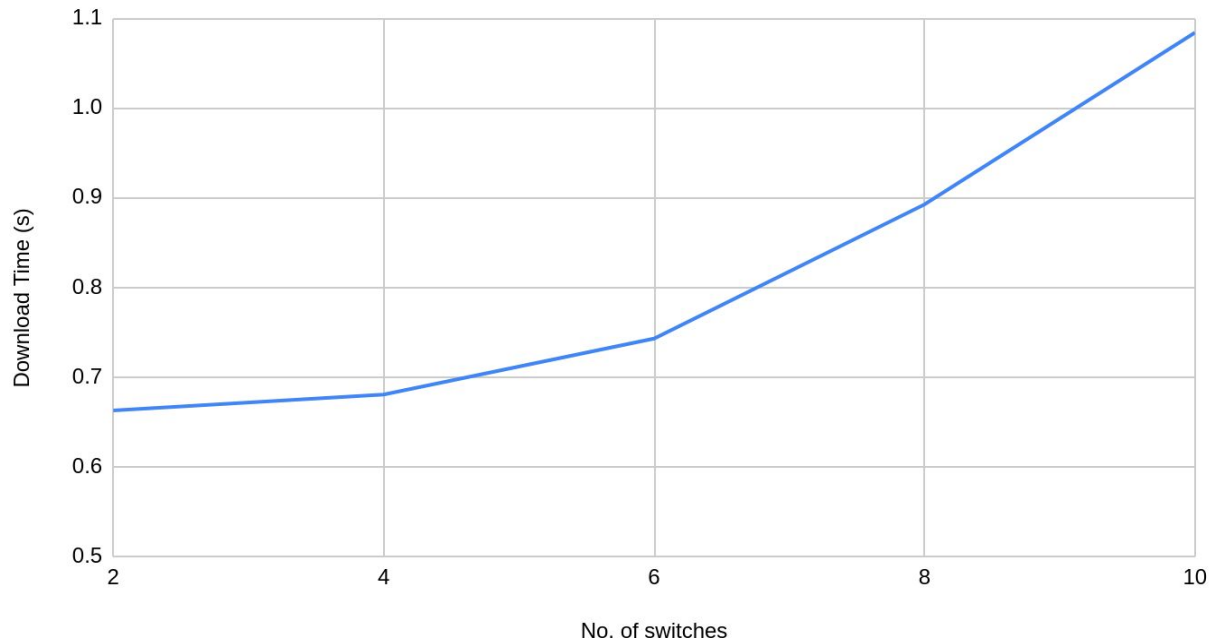
Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3j.py 2
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP132900.txt. Elapsed time: 0.6624937057495117 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3j.py 4
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP133247.txt. Elapsed time: 0.6802332401275635 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3j.py 6
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP134050.txt. Elapsed time: 0.7427501678466797 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3j.py 8
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP135513.txt. Elapsed time: 0.892254114151001 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 3j.py 10
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Bible.txt...
h2: Bible.txt(4456586 Bytes) received. Saved as BibleTCP137845.txt. Elapsed time: 1.083860158920288 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

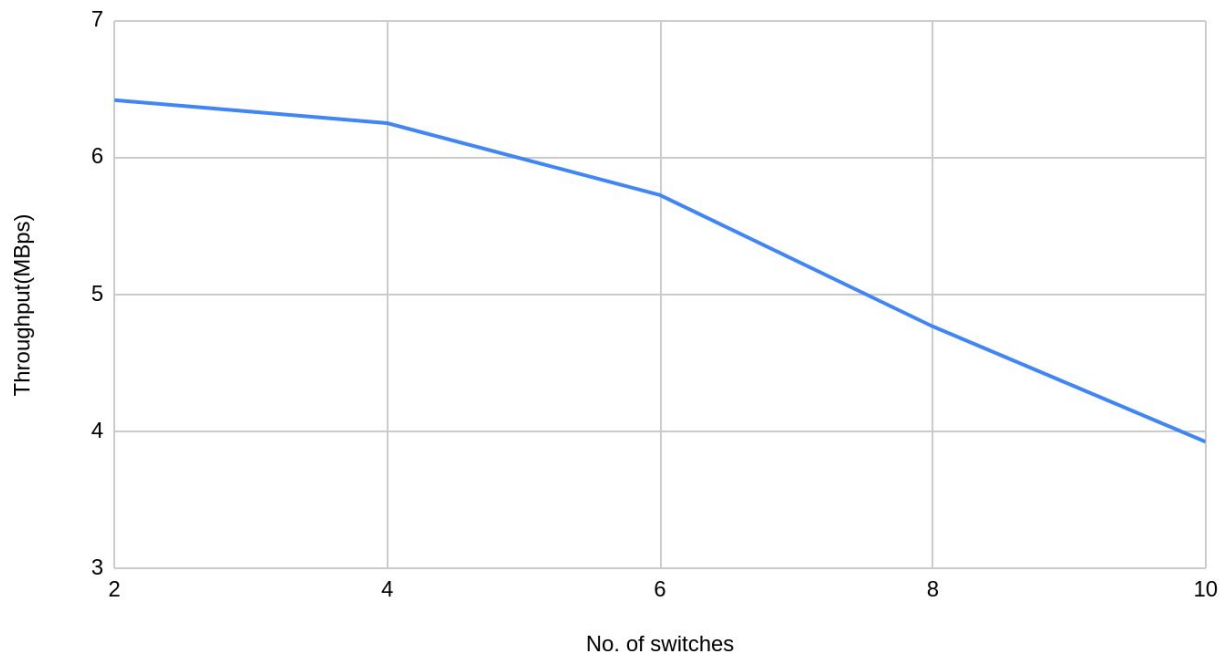
No. of switches	Download Time(s)	Throughput(MBps)
2	0.6624937057	6.415353942
4	0.6802332401	6.248050457
6	0.7427501678	5.722155027
8	0.8922541142	4.763364539
10	1.083860159	3.92129148

Observations: With increase in the no. of switches, the download time increases and the throughput decreases as can be seen from the above table and following plots.

Download time vs no. of switches



Throughput vs no. of switches



Grace Question-Mininet(Custom Topology)

4 I did this question using the Mininet Python API.

The topology for 4k, 4l and 4m are all implemented in the script Mininet/4.py and can be run as required by passing the sub-question number as argument while running it.

The script should be run as follows:

```
sudo python3 4.py (subquestion number) (largest bandwidth)
```

Example:

```
sudo python3 4.py ka 1000
```

where (subquestion number) can be ka, kb, la, lb, ma or mb as required. (largest bandwidth) is the bandwidth in Mbps that will be used for the link AS. The other link bandwidths will be determined accordingly as fractions of (largest bandwidth).

4 files chosen:

Ramayana.txt, Anna_Karenina.txt, War_and_Peace.txt, Brothers_Karamazov.txt

Bandwidth for AS = 1000 Mbps = 1 Gbps

k)

a)

Run commands:

```
cd Mininet
```

```
sudo python3 4.py ka 1000
```

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py ka 1000
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Ramayana.txt...
h2: Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP189879.txt. Elapsed time: 5.612987041473389 seconds
h3: [+] Connecting to 10.0.0.1 : 12345
h3: [+] Connected.
h3: File found.
h3: Receiving Anna Karenina.txt...
h3: Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP189880.txt. Elapsed time: 5.937425851821899 seconds
h4: [+] Connecting to 10.0.0.1 : 12345
h4: [+] Connected.
h4: File found.
h4: Receiving War and Peace.txt...
h4: War_and_Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP189882.txt. Elapsed time: 6.209612607955933 seconds
h5: [+] Connecting to 10.0.0.1 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving Brothers Karamazov.txt...
h5: Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP189884.txt. Elapsed time: 6.0512471199035645 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```


Client	Filename	File Size (B)	Download Time(s)	Throughput(MBps)
H	Ramayana.txt	2396753	5.612987041	0.407220213
I	Anna_Karenina.txt	2068079	5.937425852	0.3321765822
J	War_and_Peace.txt	3359584	6.209612608	0.5159659993
K	Brothers_Karamazov.txt	2044197	6.05124712	0.322164694

Total time(s)	6.209612608
Aggregate throughput(MBps)	1.515624782

Assuming the total time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the total time.

b)

Run commands:

cd Mininet

sudo python3 4.py kb 1000

Output screenshots:

```

chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py kb 1000
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Ramayana.txt...
h2: Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP186229.txt. Elapsed time: 6.408299684524536 seconds
h5: [+] Connecting to 10.0.0.1 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving Anna_Karenina.txt...
h5: Anna_Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP186231.txt. Elapsed time: 6.366269588470459 seconds
h7: [+] Connecting to 10.0.0.1 : 12345
h7: [+] Connected.
h7: File found.
h7: Receiving War_and_Peace.txt...
h7: War_and_Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP186232.txt. Elapsed time: 6.329824924468994 seconds
h8: [+] Connecting to 10.0.0.1 : 12345
h8: [+] Connected.
h8: File found.
h8: Receiving Brothers_Karamazov.txt...
h8: Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP186235.txt. Elapsed time: 6.3601014614105225 seconds
chris@chris-Aspire-A515-516:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$

```

Client	Filename	File Size (B)	Download Time(s)	Throughput(MBps)
H	Ramayana.txt	2396753	6.408299685	0.3566814742
K	Anna_Karenina.txt	2068079	6.366269588	0.309800551
M	War_and_Peace.txt	3359584	6.329824924	0.5061670762
N	Brothers_Karamazov.txt	2044197	6.360101461	0.3065199806

Total time(s)	6.408299685
Aggregate throughput(MBps)	1.468633369

Assuming the total time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the total time.

Observations: The total time and aggregate throughput are almost similar and very close for a) and b), though slightly better for a).

I)

a)

Run commands:

cd Mininet

sudo python3 4.py la 1000

Output screenshots:

```

chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py la 1000
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Ramayana.txt...
h2: Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP204882.txt. Elapsed time: 1.502579927444458 seconds
h3: [+] Connecting to 10.0.0.1 : 12345
h3: [+] Connected.
h3: File found.
h3: Receiving Anna Karenina.txt...
h3: Anna Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP204885.txt. Elapsed time: 1.6669206619262695 seconds
h4: [+] Connecting to 10.0.0.10 : 12345
h4: [+] Connected.
h4: File found.
h4: Receiving War and Peace.txt...
h4: War and Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP204886.txt. Elapsed time: 1.6970479488372803 seconds
h5: [+] Connecting to 10.0.0.11 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving Brothers_Karamazov.txt...
h5: Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP204889.txt. Elapsed time: 1.112757682800293 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ █

```

Client	Filename	File Size(B)	Download Time(s)	Throughput(MBps)
H	Ramayana.txt	2396753	1.502579927	1.521198132
I	Anna_Karenina.txt	2068079	1.666920662	1.183183982
J	War_and_Peace.txt	3359584	1.697047949	1.887954301
K	Brothers_Karamazov.txt	2044197	1.112757683	1.751952116

Total time(s)	1.697047949
Aggregate throughput(MBps)	5.545773037

Assuming the total time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the total time.

b)

Run commands:

cd Mininet

sudo python3 4.py lb 1000

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py lb 1000
h2: [+] Connecting to 10.0.0.1 : 12345
h2: [+] Connected.
h2: File found.
h2: Receiving Ramayana.txt...
h2: Ramayana.txt(2396753 Bytes) received. Saved as RamayanaTCP212284.txt. Elapsed time: 1.0741407871246338 seconds
h5: [+] Connecting to 10.0.0.1 : 12345
h5: [+] Connected.
h5: File found.
h5: Receiving Anna Karenina.txt...
h5: Anna Karenina.txt(2068079 Bytes) received. Saved as Anna_KareninaTCP212286.txt. Elapsed time: 1.3708453178405762 seconds
h7: [+] Connecting to 10.0.0.10 : 12345
h7: [+] Connected.
h7: File found.
h7: Receiving War and Peace.txt...
h7: War and Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP212287.txt. Elapsed time: 0.8411281108856201 seconds
h8: [+] Connecting to 10.0.0.11 : 12345
h8: [+] Connected.
h8: File found.
h8: Receiving Brothers_Karamazov.txt...
h8: Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP212289.txt. Elapsed time: 0.5749897956848145 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

Client	Filename	File Size(B)	Download Time(s)	Throughput(MBps)
H	Ramayana.txt	2396753	1.074140787	2.127953622
K	Anna_Karenina.txt	2068079	1.370845318	1.438728207
M	War_and_Peace.txt	3359584	0.8411281109	3.809109377
N	Brothers_Karamazov.txt	2044197	0.5749897957	3.390491781

Total time(s)	1.370845318
Aggregate throughput(MBps)	6.865430136

Assuming the total time to be the maximum value among the individual file download times since they are in parallel. Aggregate throughput is the sum of sizes of all files divided by the total time.

Observations(Differences between a and b): The total time and aggregate throughput are better for b) than a) since 2 of the 4 clients become closer to servers. This can also be seen clearly when checking the individual time and throughput for M and N hosts. This is due to the decrease in the distance between client and server. Also, the time and throughput are better for k) than l) since there are more servers running.

m)

a)

Run commands:

cd Mininet

sudo python3 4.py ma 1000

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py ma 1000
h2: [+] Connecting to 10.0.0.1 : 12345
h3: [+] Connecting to 10.0.0.1 : 12345
h4: [+] Connecting to 10.0.0.10 : 12345
h5: [+] Connecting to 10.0.0.11 : 12345
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

None of the clients connected to servers.

b)

Run commands:

cd Mininet

sudo python3 4.py mb 1000

Output screenshots:

```
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$ sudo python3 4.py mb 1000
h5: [+] Connecting to 10.0.0.1 : 12345
h7: [+] Connecting to 10.0.0.10 : 12345
h7: [+] Connected.
h7: File found.
h7: Receiving War_and_Peace.txt...
h7: War_and_Peace.txt(3359584 Bytes) received. Saved as War_and_PeaceTCP235373.txt. Elapsed time: 0.9962539672851562 seconds
h8: [+] Connecting to 10.0.0.11 : 12345
h8: [+] Connected.
h8: File found.
h8: Receiving Brothers_Karamazov.txt...
h8: Brothers_Karamazov.txt(2044197 Bytes) received. Saved as Brothers_KaramazovTCP235374.txt. Elapsed time: 1.097947597503662 seconds
chris@chris-Aspire-A515-51G:~/Documents/Course_Work/CN/Persistent-and-Concurrent-Client-and-Server/Mininet$
```

Two clients didn't connect to the servers.

Client	Filename	File Size(B)	Download Time(S)	Throughput(MBps)
H	Ramayana.txt	2396753	-	-
K	Anna_Karenina.txt	2068079	-	-
M	War_and_Peace.txt	3359584	0.9962539673	3.215996202
N	Brothers_Karamazov.txt	2044197	1.097947598	1.775583991

Observations: We see that in a) none of the clients are able to connect to the servers and in b) 2 of the clients are not able to connect to the servers. It is interesting to note that the clients that are unable to connect belong to the subtree rooted at B.

Analysis: Since we have introduced a loop and from the above observations, we can assume that the presence of the loop is causing the clients to be unable to connect to the servers. This prevents clients H, I, J, K which are in the subtree rooted at B, from connecting to any server on the other side of the loop. In other words, they are not able to cross the loop and are getting stuck in the loop.