Arturo Villalobos

Tanzir Ahmed

CSCE 313-503

2-13-21

PA1

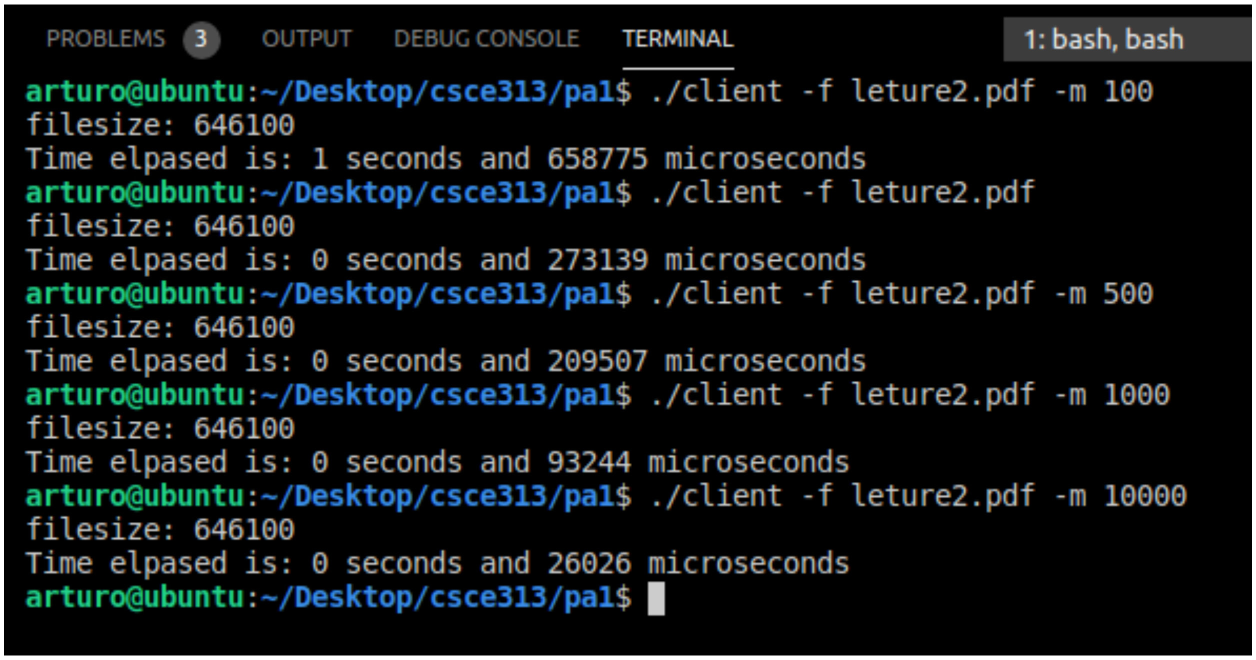
Video link: https://youtu.be/qf1BstY-ZkM

Design:

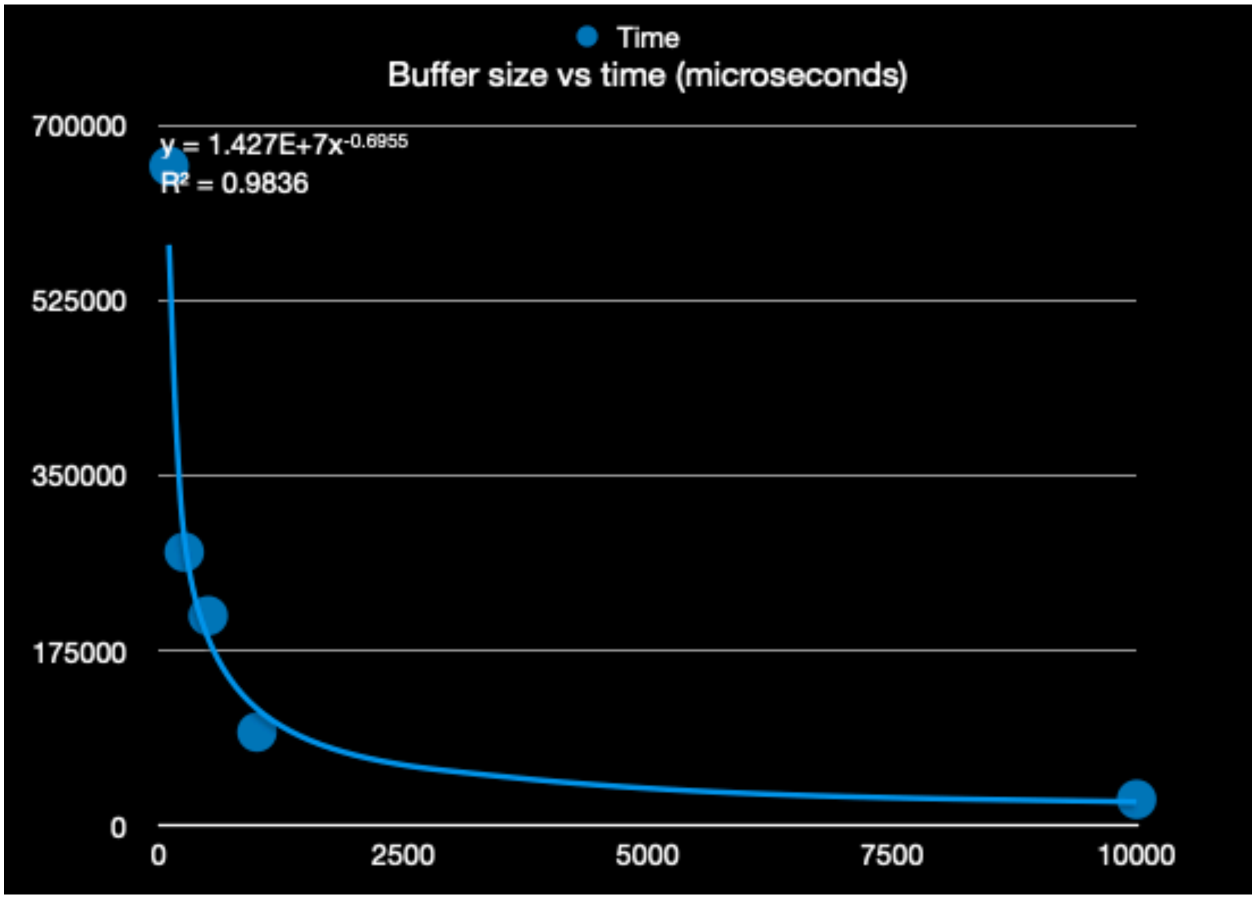
The main design choice I made was to not use helper functions, rather, my program depends on the initial conditions of the variables I have set at the outset. The program can be broken down into 4 main sections, the switch case , the 1000 data point file transfer, the file transfer, and the FIFO request/termination block. Unfortunately I was not able to get the server as a child process. After several attempts either process wouldn't terminate after running the command. I talk more in detail about the individual lines of code in the video.

Results:

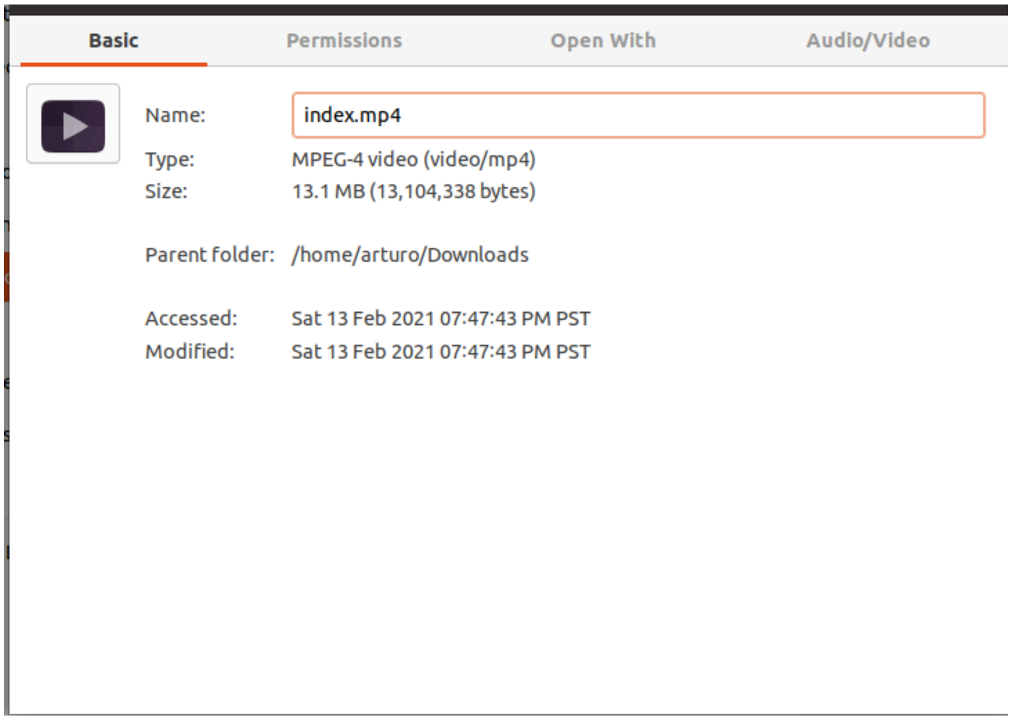
As expected, file transfer is faster when the -m flag is made higher. Below are some screen shots and a graph showing this as a fact. For the testing I used the lecture 2 pdf with varying sizes of m:



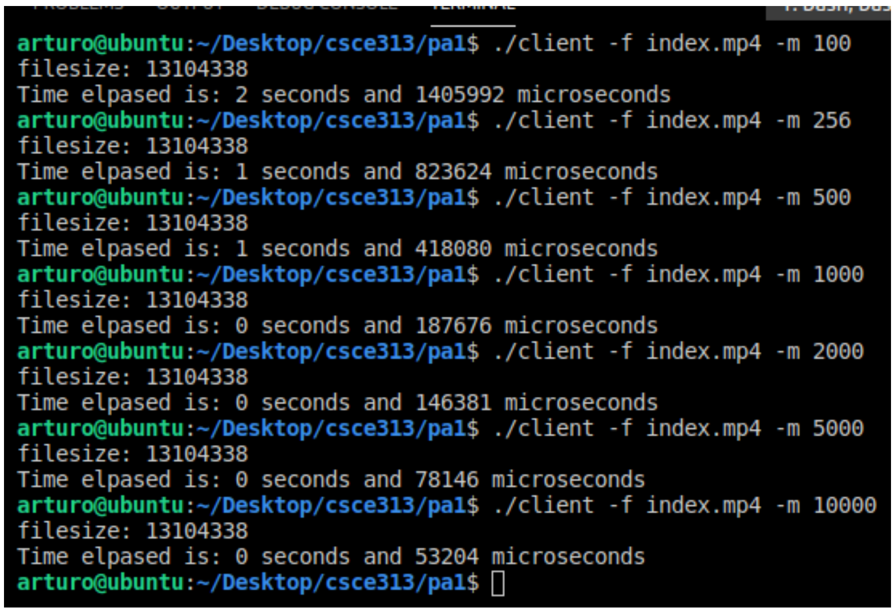
```
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f leture2.pdf -m 100
filesize: 646100
Time elpased is: 1 seconds and 658775 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f leture2.pdf
filesize: 646100
Time elpased is: 0 seconds and 273139 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f leture2.pdf -m 500
filesize: 646100
Time elpased is: 0 seconds and 209507 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f leture2.pdf -m 1000
filesize: 646100
Time elpased is: 0 seconds and 93244 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f leture2.pdf -m 10000
filesize: 646100
Time elpased is: 0 seconds and 26026 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ 
```



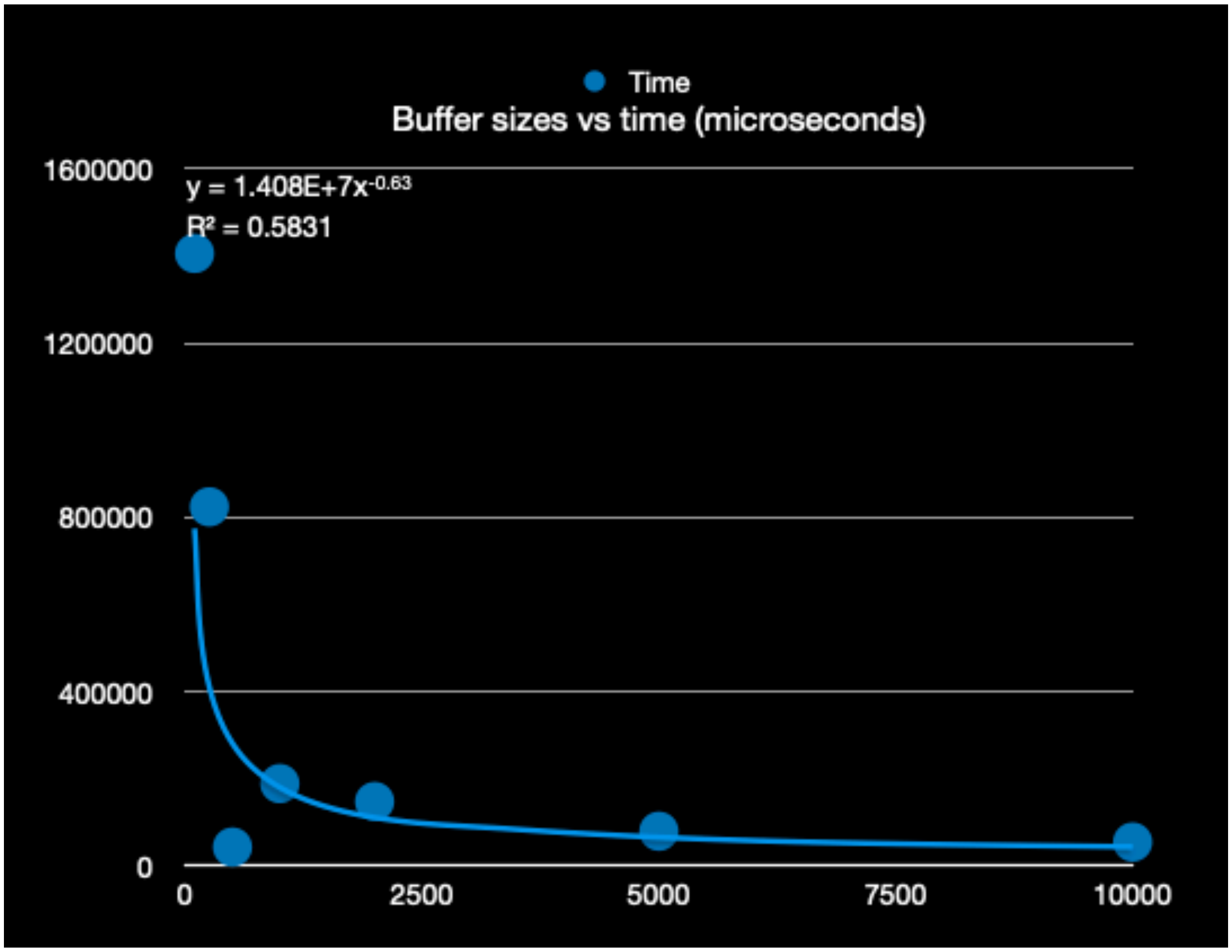
As can be seen via the graph the time it takes to transfer relatively big files approaches 0 as the buffer size (-m) flag increase even just to 10000. Other examples of file transfers are shown in the video but for a larger size file I downloaded a 10 min YouTube video at 144p with a size of 13.1 mb. Data shown below:



Name: index.mp4
Type: MPEG-4 video (video/mp4)
Size: 13.1 MB (13,104,338 bytes)
Parent folder: /home/arturo/Downloads
Accessed: Sat 13 Feb 2021 07:47:43 PM PST
Modified: Sat 13 Feb 2021 07:47:43 PM PST

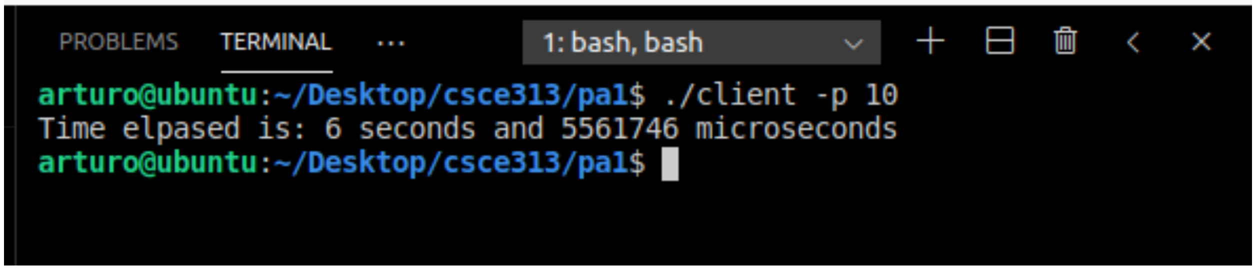


```
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 100
filesize: 13104338
Time elpased is: 2 seconds and 1405992 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 256
filesize: 13104338
Time elpased is: 1 seconds and 823624 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 500
filesize: 13104338
Time elpased is: 1 seconds and 418080 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 1000
filesize: 13104338
Time elpased is: 0 seconds and 187676 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 2000
filesize: 13104338
Time elpased is: 0 seconds and 146381 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 5000
filesize: 13104338
Time elpased is: 0 seconds and 78146 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -f index.mp4 -m 10000
filesize: 13104338
Time elpased is: 0 seconds and 53204 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ 
```



Again the same trend appears even in larger size files.

Lastly the time data for 1K data points is shown below with -p 10:



```
arturo@ubuntu:~/Desktop/csce313/pa1$ ./client -p 10
Time elpased is: 6 seconds and 5561746 microseconds
arturo@ubuntu:~/Desktop/csce313/pa1$ 
```

My program iterates through a loop requesting a single data point at a time, and since there are in reality 2k data points we are fetching the process is quite slow when compared to file transfer.