Val Robichaux

CSCE 313 - 503

Dr. Tanzir Ahmed

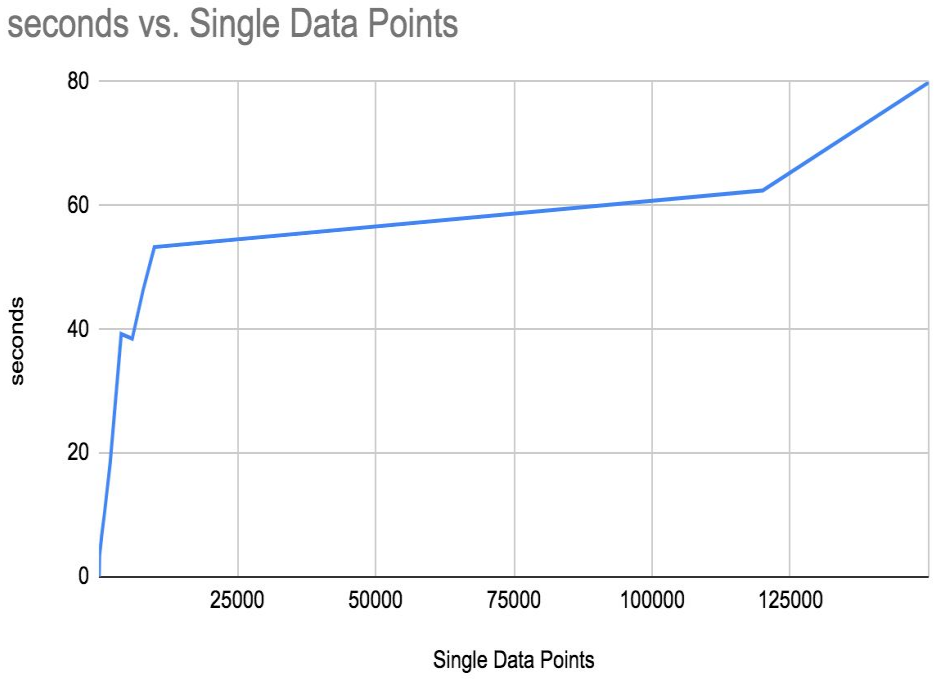
21 February 2020

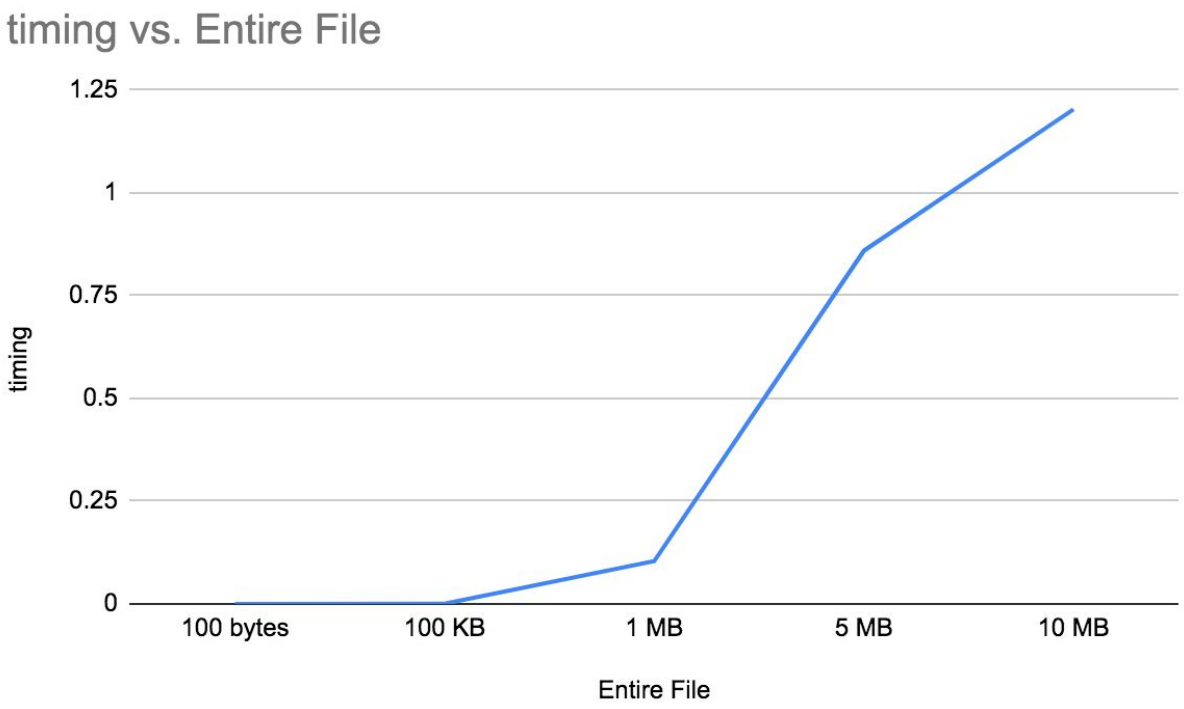
Client and Server Processes

The client and server communication process was focused around creating a program that could adapt to take multiple users and communicate to a server in order to obtain multiple data points. This is applicable to today's world because it is used everyday in just about every processing company that may use a username and password, server are needed. The design I used to approach this project was one that was very sequential and made sure I could get single points and files correctly before moving to creating new channels and speaking to the server. First, I would go about just retrieving a single data point by moving the data from the server to a buffer and then taking that data and writing it to my own file. I did this exact same thing for the multiple points just recursively calling each point over and over until I reached the end of the file. Next for the file retrieval, I would take chunks of the file at a time and return the number of requests as well as the size of the file to show that we retrieved the correct data points and the correct file. Finally I created a new channel by assigning a new message type of c and created a new channel buffer so that I could pull data points from this new channel to ensure that is was working correctly. The timing data that I used for the data collection for the entire files usually took around 80 seconds for an entire file full of data points that was only in binary. For acquiring the text files using truncate, the file retrieval was very quick and often was under a second for just the file retrieval, even with the largest files that were as 10MB. This doesn't take as long because we are not writing a lot of points to a given file that we want and storing it in a directory. We are simply just asking to retrieve the file and place its same name under our received directory. The bottleneck was very clear when it came to

making our max buffer size different as this would cause the client and server to have a ton of requests for the given files. Doing this over and over again caused my program to run very slowly. Below are graphs and examples of the timings. Here we can see the correlation between just retrieving the files and mapping the single data points. I think we can deduce that it takes a much larger amount of time the more data points we need, rather than just retrieving a file of a truncated size.

Single Data Points	seconds
1	0.003
20	0.02
100	3.453
500	6.793
1000	10.3592
2000	18.3942
4000	39.23942
6000	38.49283
8000	46.3923
10000	53.2943
120000	62.4392
150000	79.889





Entire File	timing
100 bytes	0.00003
100 KB	0.00032
1 MB	0.10394
5 MB	0.85932
10 MB	1.2034