Mauro Lopez
Csce-313-504

A Client Speaking to a Server Process Report

Design:

to start this assignment I first needed to understand how the server and client communicate. The data server and client would communicate using channel structures. Meaning when the server wrote to one channel, the client would read from it. This meant that if I wanted to send a request, I first needed to write that message into a buffer. This buffer contained the message in an array of chars, but limited to 256 characters. This meant, that I had to cast my messages from their respected types to char. In the other end, the server was in a loop constantly reading the request from the client.

Requesting Data Points

Requesting data points for a patient required the client to give the patient number, what time, and either ecg1 or ecg2. This message was stored in a *datamsg* class. This variable was then converted into an array of chars. This was sent via buffer, along with the size. The server read it, and sent back the respected data in a double type

Requesting a Text File

To request a file we first needed to know how big the file was. So we first packed the name of the file we want, along with the message of type *filemsg* to let the server perform the right request. Once we got the size of the file, we then need to figure out how many messages is going to take to get the entire file. We then send a series of messages with an offset and a length until we gather the entire file. The same code was used for the binary file.

Requesting a New Channel:

To request a new channel, we also needed to send a message in the form of chars. We begin by declaring the new channel variable, we then cast it to be an array of chars. We then write that along with the size into our buffer. The server reads the buffer performs the task, creates a new channel name and sends back the name of the new channel to the client. The client can now use this new channel. To avoid leaving this channels open, we need to send out a quit signal that we erase this temporary files from our directory.

Timing Data

Patient	Data points request (sec)	Text File request (sec)
1	101.973775	0.000343
2	103.580820	0.000196
3	109.191899	0.000310
4	107.822037	0.000300
5	107.984043	0.000136
6	107.701770	0.000155
7	106.148745	0.000188
8	116.274350	0.000197
9	110.774843	0.000138

Mauro Lopez
Csce-313-504

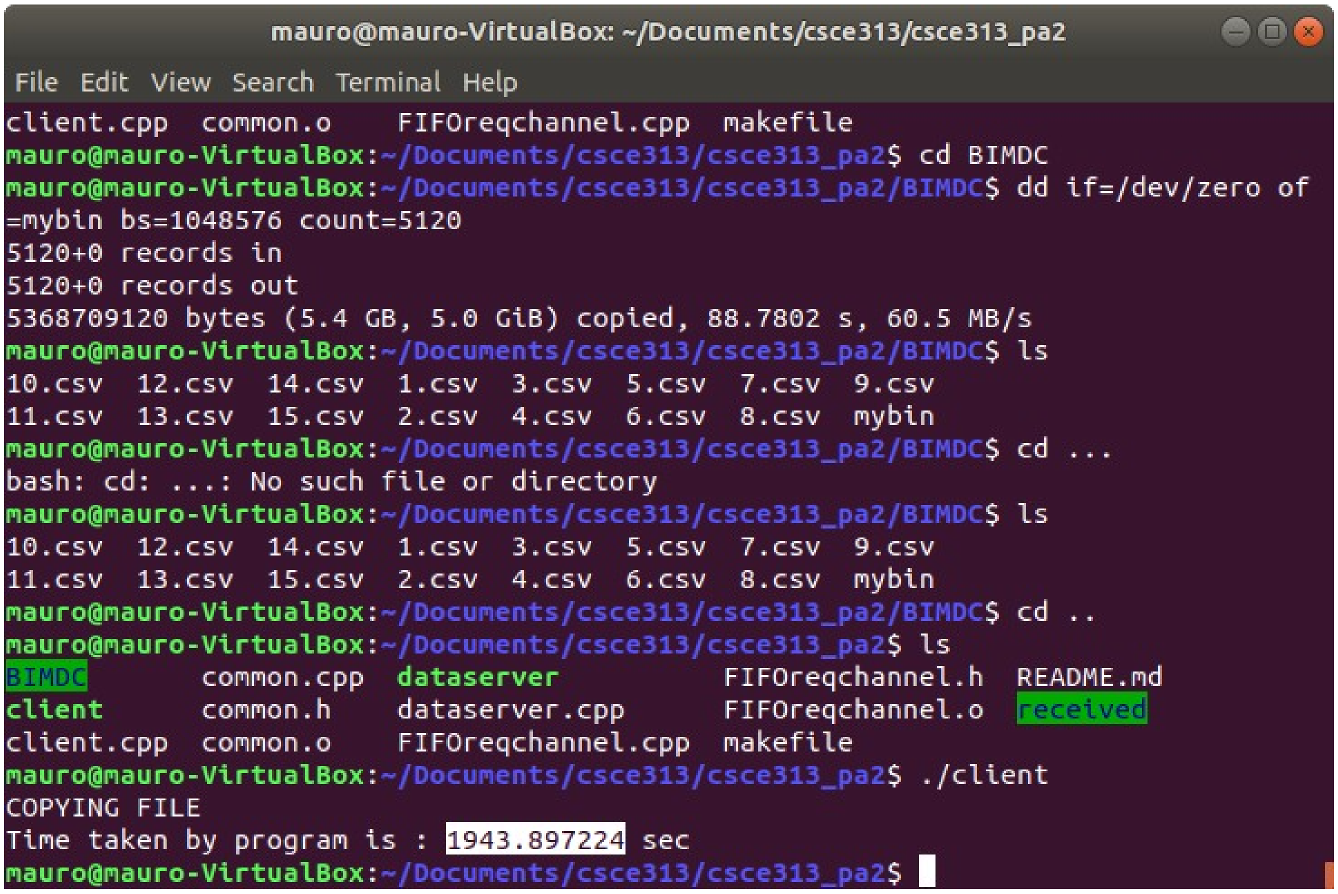
Binary File Size	Run Time
1G	391.979904
5G	1943.897224

Bottleneck:
When we look at the time of our data points request and text file request, we can see that our data points request is much slower. The main reason for this is that, unlike the file request the server has to look for this values in the 15k list of values. Also the fact that the client is asking for those 15k points. In the other hand, our file transfer actually maximizes its resources to perform the best. Instead of sending individual data points, it sends out 256 bytes worth of data every time. Meaning is going to be faster than data points request.

Another bottleneck also occurs in the files with bigger size such as the 5G and text files. Similar to our data points and text file we can see that the size of the buffer is affecting the run time. To improve this, I think we first need to make the size of the MAX_MESSAGE_SIZE bigger so that we can transfer more data faster. To also improve this we can also call a fork() function that can work together and split the file into separate smaller files and transfer that all in parallel.

Screenshot:

Mauro Lopez
Csce-313-504



mauro@mauro-VirtualBox: ~/Documents/csce313/csce313_pa2

```
client.cpp  common.o    FIFOreqchannel.cpp  makefile
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2$ cd BIMDC
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2/BIMDC$ dd if=/dev/zero of
=mybin bs=1048576 count=5120
5120+0 records in
5120+0 records out
5368709120 bytes (5.4 GB, 5.0 GiB) copied, 88.7802 s, 60.5 MB/s
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2/BIMDC$ ls
10.csv  12.csv  14.csv  1.csv  3.csv  5.csv  7.csv  9.csv
11.csv  13.csv  15.csv  2.csv  4.csv  6.csv  8.csv  mybin
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2/BIMDC$ cd ...
bash: cd: ...: No such file or directory
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2/BIMDC$ ls
10.csv  12.csv  14.csv  1.csv  3.csv  5.csv  7.csv  9.csv
11.csv  13.csv  15.csv  2.csv  4.csv  6.csv  8.csv  mybin
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2/BIMDC$ cd ..
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2$ ls
BIMDC       common.cpp  dataserver          FIFOreqchannel.h  README.md
client      common.h    dataserver.cpp      FIFOreqchannel.o  received
client.cpp  common.o    FIFOreqchannel.cpp  makefile
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2$ ./client
COPYING FILE
Time taken by program is : 1943.897224 sec
mauro@mauro-VirtualBox:~/Documents/csce313/csce313_pa2$
```