CSCE 313 Programming Assignment 3

Threading and Synchronization

Aldo Leon Marquez UIN: 326004699

The goal of the assignment is to introduce us to what threading is, how to do a basic implementation and the benefits and limits of it. To do so we build on top of the concepts, and code, from the previous assignment. We start with the server and client, and then proceed to split the workload into threads.

To First step to implement the assignment is to create a Bounded Buffer, a B.B allows us to use the concept of work stealing were the maximum fold-up is no longer dependent on the producer of data, rather, we have a pool of jobs that “workers” can take from, the B.B allows us break that foldup dependency and also prevent the pool from being to big, hence why its bounded.

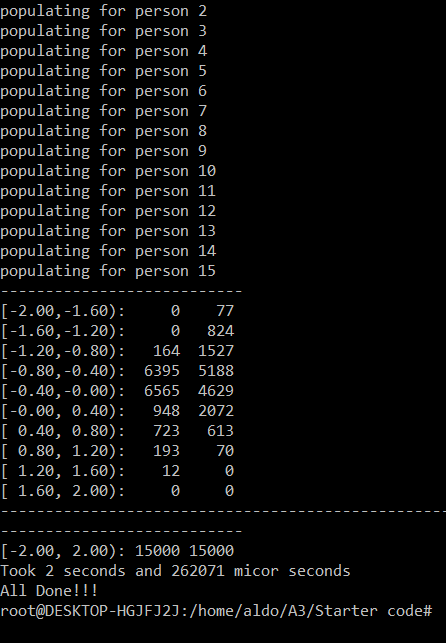
The base code for the B.B was provided, with the missing implementation of the push and pop int the queue , job pool. The important part of this to is to implement the regular push and pop features thread safely and also when they are supposed to be called, we use a mutex lock to prevent the critical section from being affected, and a wait for both push and pop to allow the thread to continue when available.

After that, we just start to thread the client, to do so we need Producer and Consumer threads, this threads are stored in a vector to ensure they do go out of scope. Each Consumer will have a histogram object to print at the end, and each Consumer will have a FIFO Channel. Once the vectors are correctly filled with the corresponding threads, its time to join and execute the functions of each.

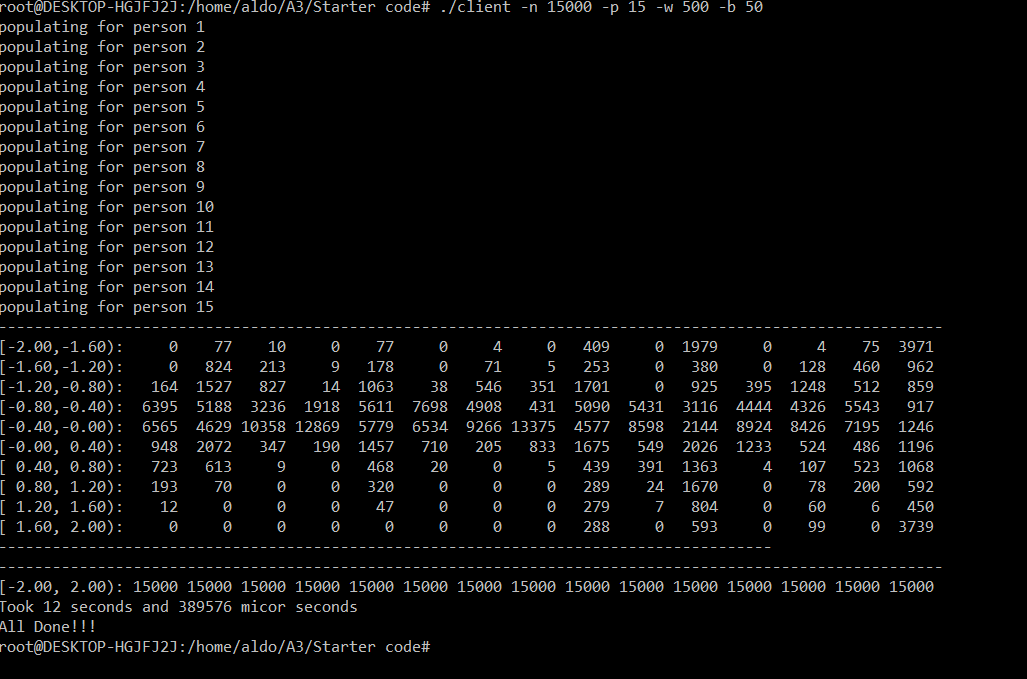
The Consumer will execute the patient\_funciton, a function that simply formats the server messages and pushes to the job pool, its also in charge of, when all of them are done, to send the QUIT\_MESSAGE to the B.B so that the workers can end.

The Workers will execute the worker\_function that creates the Channel for the worker, pops a job from the pool and repeats until the EXIT\_MESSAGE is popped(note: it is important to push it back so other thread can get it), The function is also in charge of updating the Histogram and to handle both Data and File Msg, to handle those, an int is passed indicating the mode 0 or 1, it will then work the correct logic accordingly

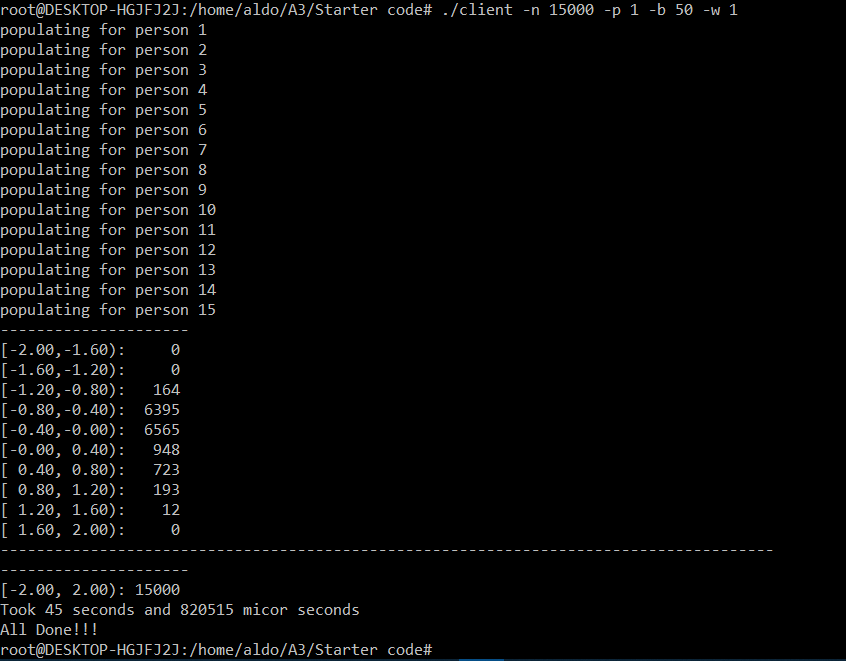
The final implementation does all of the expected, with a few minor details. The images below show some examples of it

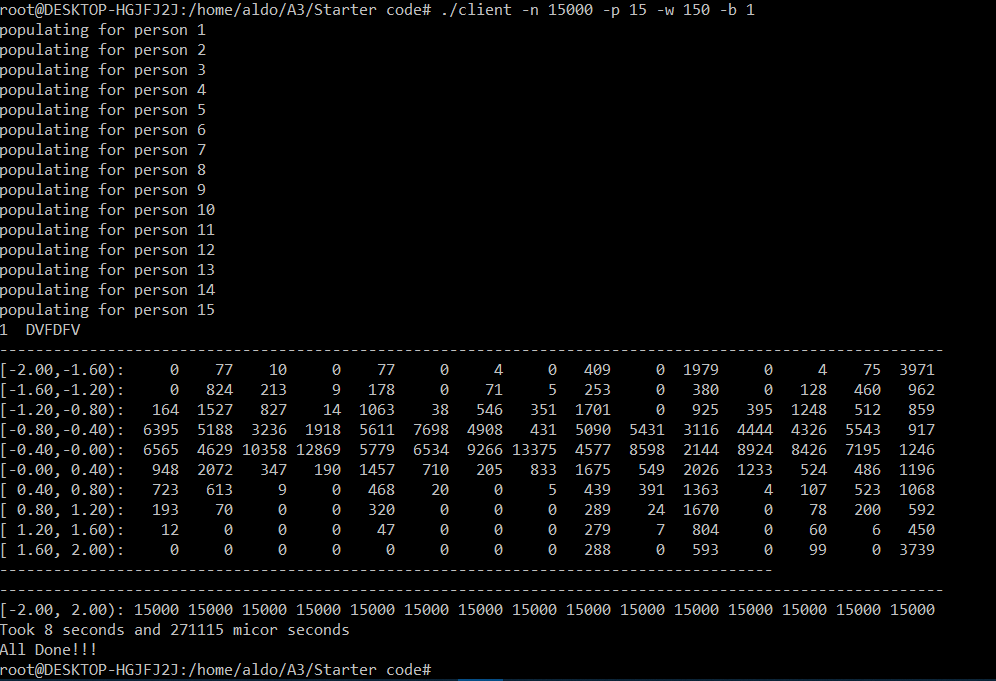


Output for the Pdf Example

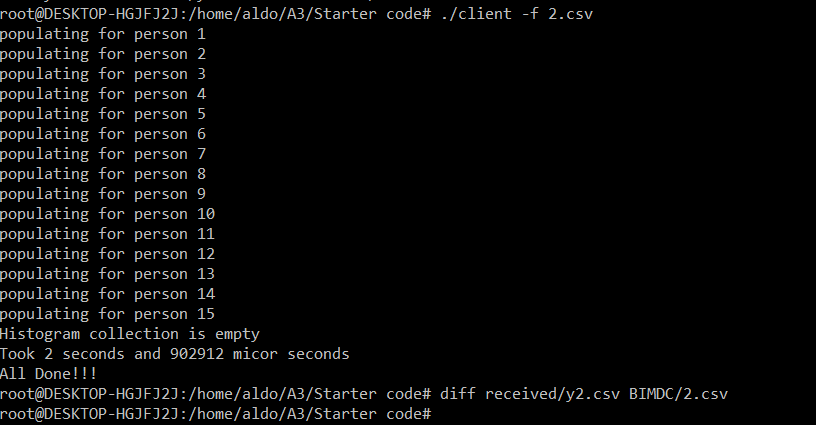


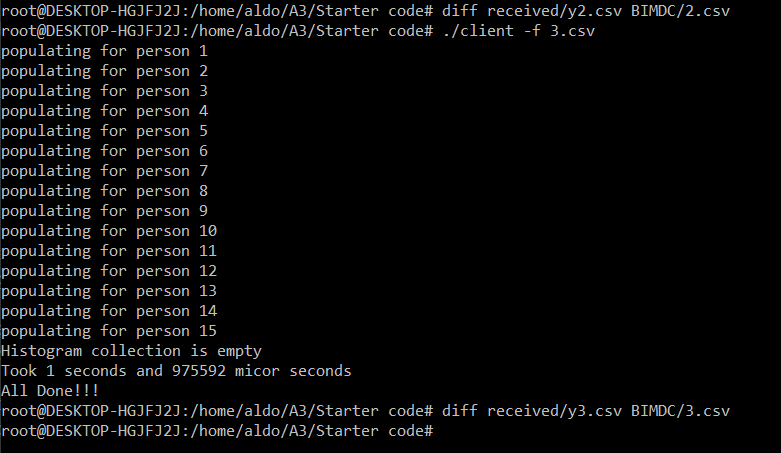
Full Histogram with default conditions



Proof for one worker only with only 2 patients

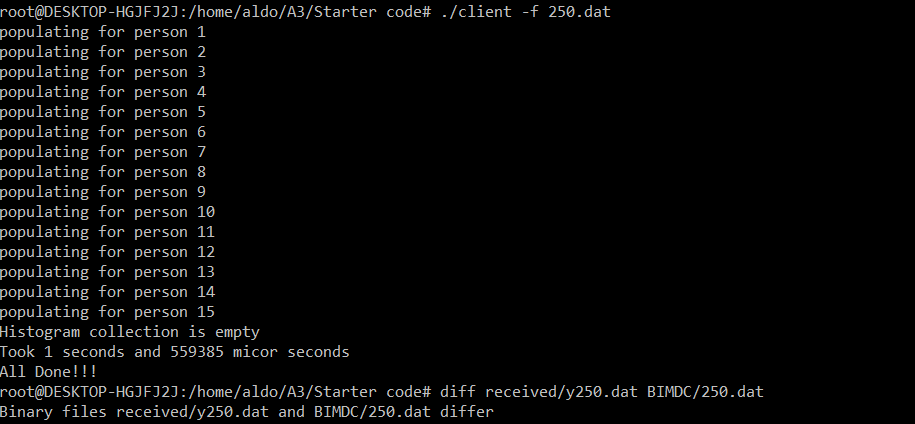
Full Histogram with buffer size 1

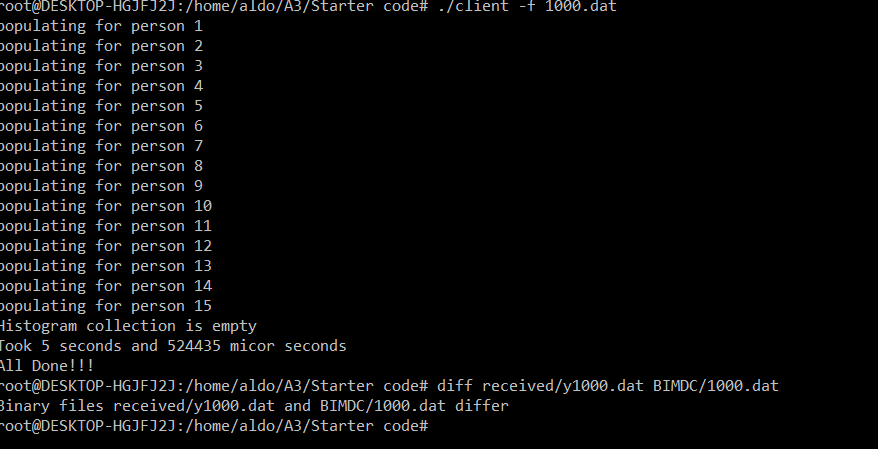


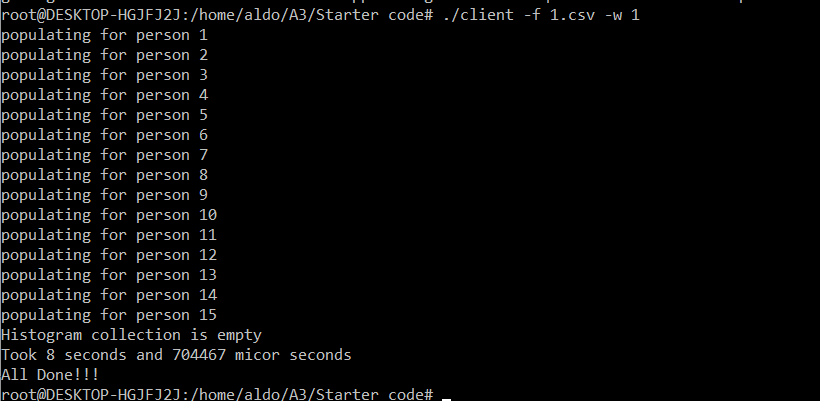


File Transfer examples, with diff check





.dat Files with diff Check, 100 perfect, 250 and 1000 differ, however they do show the correct size on the explorer and, are also empty as they should



A file Transfer with just one worker

**Performance and diminishing returns:**

For the data transfers we can see a pretty clear performance improvement once we move of the 2 one-digit worker threads, a solid performance for the first few hundred, a clear slowdown just before the thousands and, a clear declined after a the first one thousand threads. This can be explained by context switching and file transfer speeds, As the number of threads grows, the context switching penalty starts to become apparent, with so many context switching it becomes pointless to thread, there is a clear sweet spot for this program, for 150 -300, then for 300 to 600 is almost identical and above that there’s a clear point of diminishing returns. The graph bellows contains the data just discussed

As for the buffer size I found no problem with increasing it, as long as it was above 40 the performance was pretty solid.

Finally, for the data transfers, we get the same behavior as we increase the number of worker threads, the performance slows down and becomes pointless to thread, for the exact same reason. For my code the sweet spot is around 10 worker threads.

So in conclusion the assignment was almost perfectly implemented aside from that .dat diff error, that I couldn’t find a solution both the file showed empty and with the correct size.

