Mualla Argin
UIN : 728003004
CSCE 313 - 910

Programming Assignment 3

Instructions: Write a report describing your unique design choices, algorithms (e.g., implemented file input/output redirection, piping, etc.), and any implemented bonus features along with its implementation technique. The report must be a minimum of a page. We are not looking for generic items that are common to the entire class as part of this PA instructions, but we are looking for your insights and unique contributions.

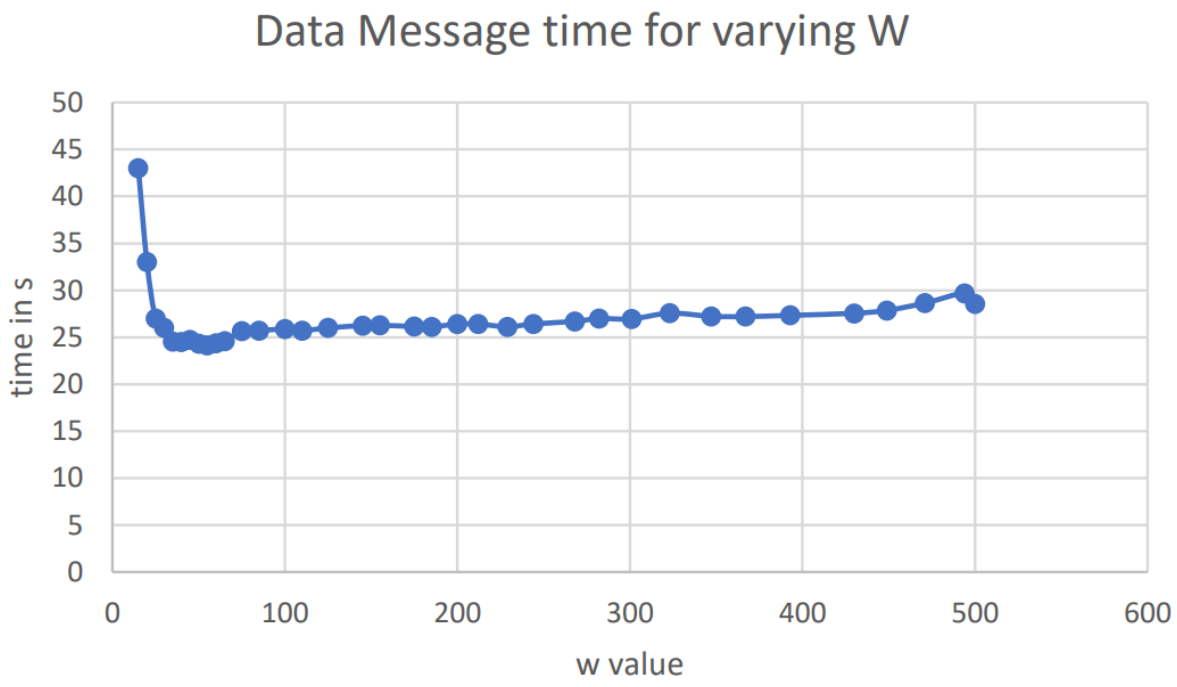
Design of Code

In my code I create a histogram object for each patient. This results in a p number of histograms. All the histograms created through this process are then added to the Histogram Collection. The Histogram Collection is tasked with listing and printing the histogram values to the console.In the case of data requests, the client prints the collection of histograms for each person. The client I created accepts a multitude of parameters like f,m,b,w,p,n and gives the user an option to decide whether they would like to request a file or request data. After this, patient and worker threads are initiated and then later joined.

When two or more threads access the same variable concurrently, there are bound to be issues. Hence, I sychronized threads with the use of mutexes. I used mutexes because they are designed to prevent race conditions by providing mutual exclusion. I also made sure the Bounded Buffer class is protected against overflow and underflow conditions through my two condition variables : `data_available` & `slot available`.

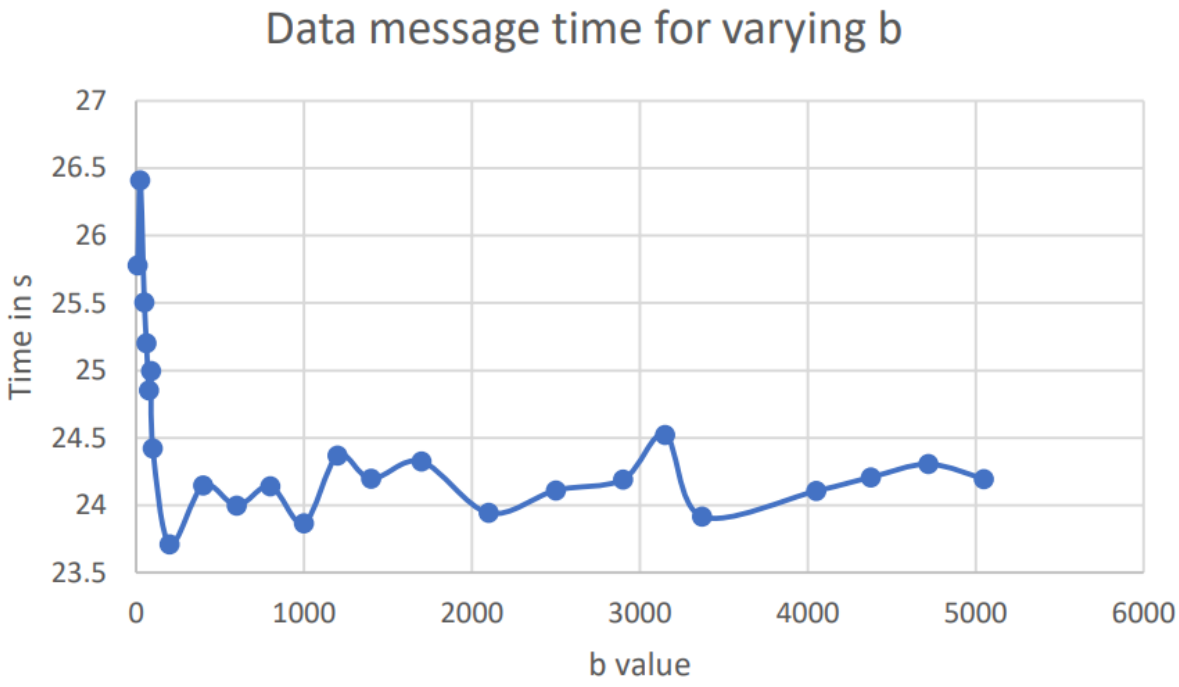
Mualla Argin
UIN : 728003004
CSCE 313 - 910

Data Requests

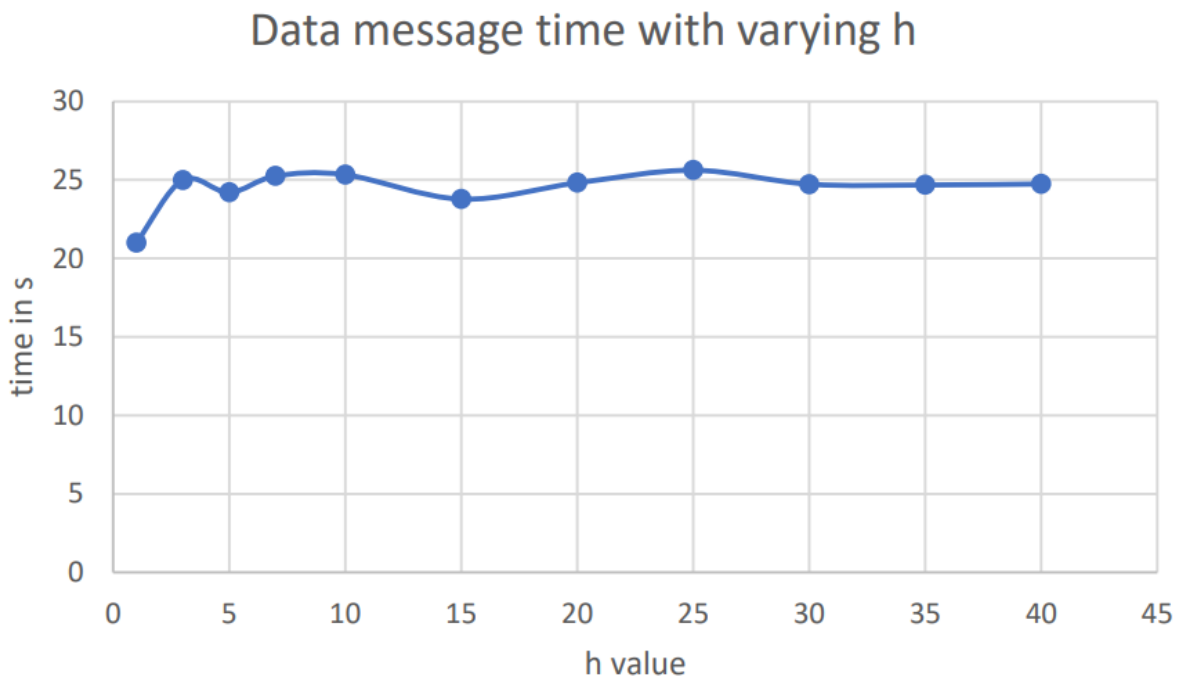


From the above graph, we can see how at 10 workers, tasks take about 33 seconds.Then, at 50 workers, tasks take 24 seconds. We can also see that at 80 workers we start experiencing a point of diminishing returns. No matter how many workers we employ, the rate at which tasks complete remains the same & stagnant (about 26 seconds). Before the point of diminishing returns however the increase in efficiency for workers was exponential.The addition of threads allowed for optimized utilization of CPU activity/cores. However, once the maximum number of cores were being utilized (we hit the point of diminishing returns) then there was overhead that decreased the speed of the program.

Mualla Argin
UIN : 728003004
CSCE 313 - 910



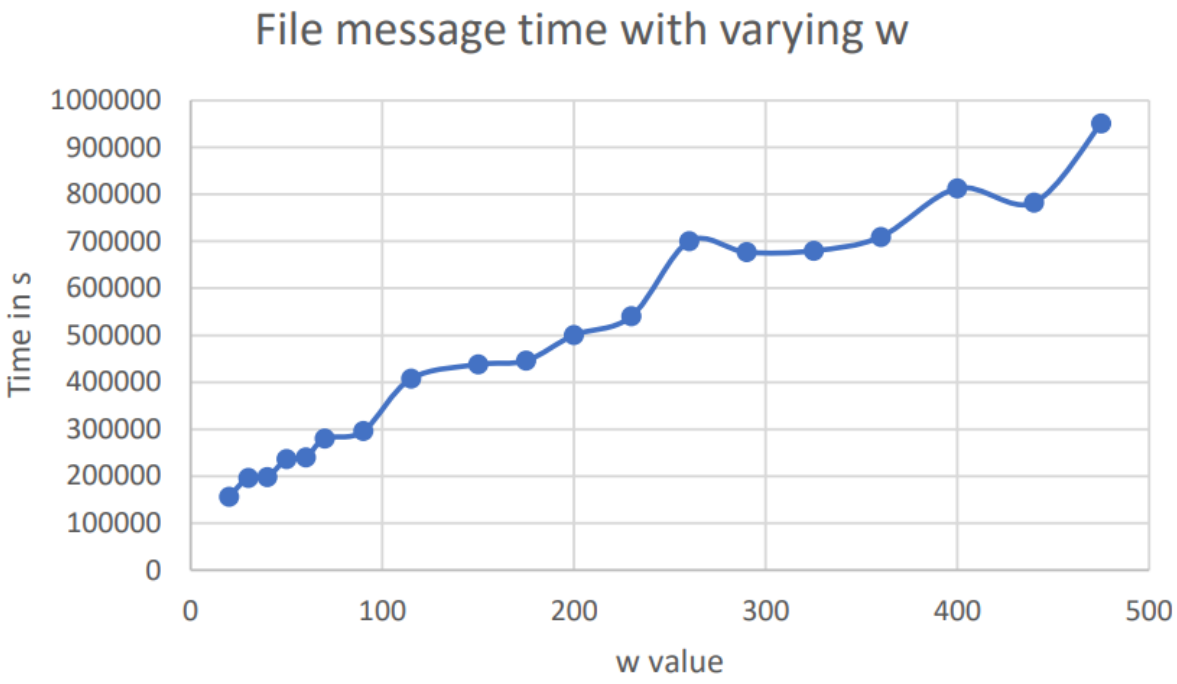
As shown in the above graph, changes in buffer size only impacted runtime when buffer sizes were significantly smaller. Otherwise, data message time remained constant with increasing buffer size.



Mualla Argin
UIN : 728003004
CSCE 313 - 910

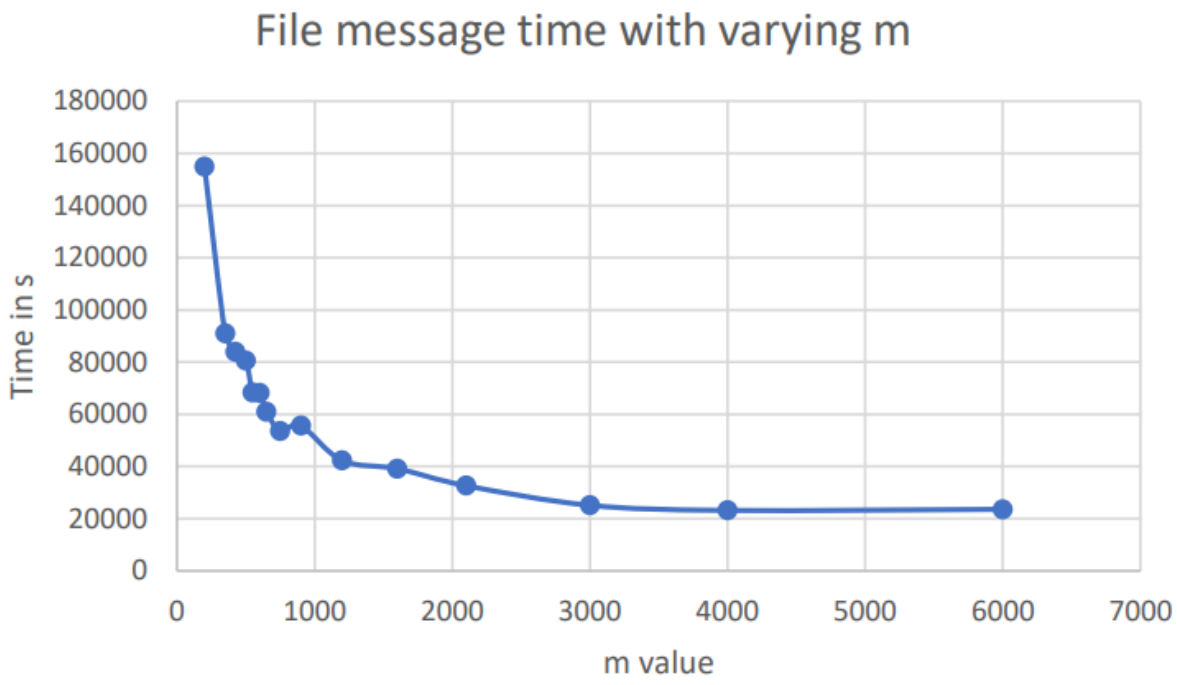
For the most part, execution time was not affected by the size of the h value. The execution time remained fairly constant for any h value given (approximately 25 second).

File Requests



As shown in the above graph, an increase in the worker thread count resulted in an increase in file message time. There is no scaling occurring for this graph. This may be due to context switching overhead.

Mualla Argin
UIN : 728003004
CSCE 313 - 910



As shown in the above graph, an increase in the buffer size results in a decrease in file message time. Logistically this makes sense because the more buffer capacity we have the more items we can insert into a single buffer. When we can insert more items into a buffer the program becomes more efficient. Hence, over time there is scaling.