Project Report

Althof Pandyansa Fajar (9770937)

Introduction:

Abstract
In 2020, a climate analyst developed a new method to analyze the content of O3 in the atmosphere throughout Europe and surrounding areas. There are difficulties with this method, one of which is the processing requirements and time for massive data like the method to analyze O3, which uses NC files. To reduce the processing time, it requires big data analysis to determine processing ways to load the data using the least amount of processing to load the NC file of the O3 data.

Main
This project is an experiment and big data analysis on data provided by the client. The objective of this analysis is to investigate processing methods that minimize the amount of time that is needed to process the data, determine the methods that could minimize processing time. The analysis will be using the techniques being taught throughout the 5011CEM module.

This report will contain a detailed description of the code, the comparison between sequential and parallel processing, the analysis results, the recommendations to be made based on the outcome of the analysis, the conclusion to the whole project and module and the appendix that contains additional, non-essential information for this report.

Code Description:

There are 2 codes that I did before merging them, there are sequential and parallel processing. The files for the code and other details of the project are on the GitHub repository.

Sequential Processing

In sequential processing, the code grouped into several parts. The first part of the code consists of the commands to load the required file and loads latitude and longitude. The second part consists of the parameters, which has variables that contain the cluster radius.

The third part encompasses more variables for the starting point and the number of latitudes and longitudes, followed by a for-loop that loops through every hour from 1 to a given number. Once inside the loop, it will set the data layer to one and a for-loop that will load all the data, with layers increased by 1 after each iteration. After all the data a function provided by the customer will prepare for the sequential analysis.

On the sequential analysis, 2 variables will set a timer and it will take place on a for loop that loops from 1 to a given number of data. Within the loop, a function provided by the client will ensemble value and an if statement that will track the time for every 50 locations and prints the result afterwards. 2 end statements will follow to end the if-statement and the for-loop.

After completing the sequential processing it will record the processing time for each hour and end the loop. It will be followed by another end statement and records the overall time for the sequential processing.

Parallel Processing

Parallel Processing contains more parts than sequential one. The first part is started by loading the file similar to that of and detailed parameters are provided. A pre-processing will later take place, it is done by reducing each latitude and longitude by 4 and multiplying each other.

Once the memory has been pre-allocation, it will cycle through hours, it is started by setting the time counter to 0 and will loop through hours. Inside the loop, in the beginning, it is similar to sequential, it loops to load data and preprocessing will occur.

After the data has processed, it will perform the parallel analysis. Instead of one large loop, like sequential analysis, sectioned into multiple variables and loops. It started by creating a parallel pool. a variable containing

the pool size and acts as an input into the parallel pool, determining its size. Several variables are later set to pre-process before starting the actual processing.

In the actual processing, it will ensemble the a value into every location that is set. It will take place in a for-loop, that will cycle through each number of data. After each iteration, it will ensemble the value, which consists of the parameters provided, and records the time to process. After the number of data is exceeded, it will end the loop and record the total processing time, followed by ending the hour loop.

Automated Testing Code modifications

For the automated testing version of both codes, few modifications were made to the code. For Parallel Processing, the first modification is 3 new empty arrays that will be able to collect the recorded time from 3 different numbers of data. 2 newly filled arrays are also added, which contains the data sizes and the number of workers requested by the customer. The hour loop is now nestled below the pool size loop, which itself is also nested below the data loop. After the actual processing, an if-statement is added to store the recorded time and pool size, allocating it up to 3 different arrays. After which the elements are plotted into a graph. In the end, break tests are also added to prevent the hours to be a NaN (Not a Number).

For Sequential processing, several modifications were also made to automate it. Like parallel processing, their arrays are empty arrays to record processing time and a plotting code at the end to allow the client to plot the result automatically. Another modification made into it is a for loop, where it iterates through each different data size and the hour loop is nested within the Data loop, where it selects different data sizes from the array and an hour loop is nested bellowed it, it was modified to 3 since the objective for sequential processing is to observe the processing time using 3 different hours.

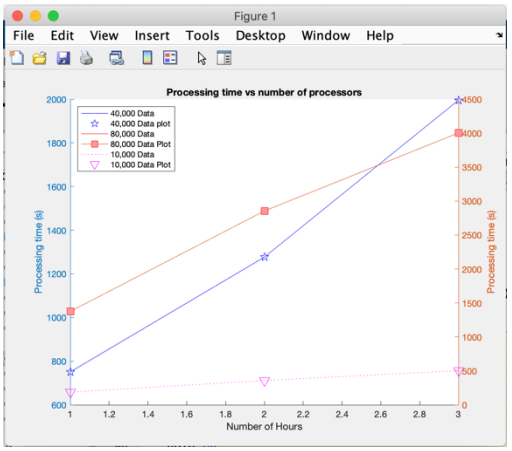
Sequential and Parallel Processing:

There are two methods used for this analysis, which are sequential and parallel processing. Sequential processing is where the processing took place with one processor, while parallel processing uses multiple numbers of processors.

The number of hours requested to load this data is somewhere under 2.5 hours. At first, the test was done manually by running it one by one and plotting the results, it was later automated due to efficiency and allowing the client to modify and see the error without the user interference.

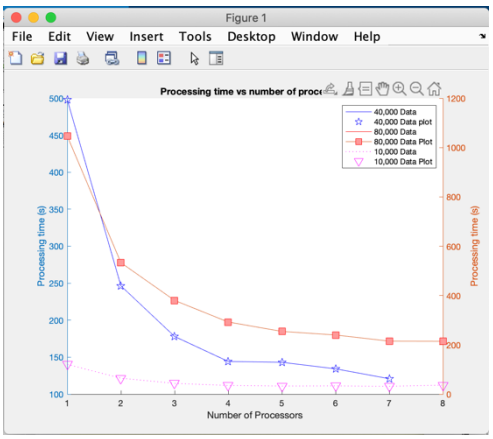
For the testing of both methods, the amount of data that is used for both processing methods is 80000, 40000 and 10000, to experiment with different sizes to show its consistency.

For sequential processing, each data size is tested with 3 different hours, which is 1,2 and 2.5 hours prospectively. The processing speed is plotted using the script that is also provided (highly modified), Below is the plotted outcome:

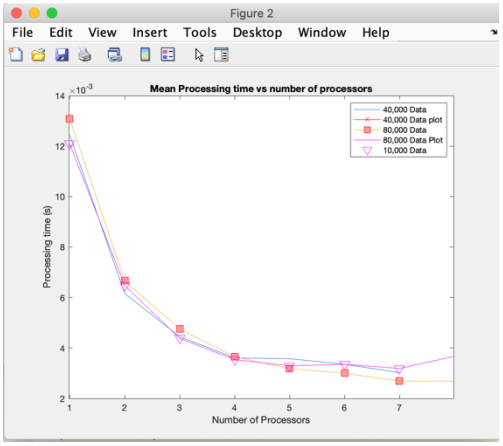


With parallel processing, it is tested with different numbers of processors ranging from 1 to 8. Like sequential processing, the result is tested using automated testing.

With different numbers of processors, the time required to load the data will be varied significantly, it is hypothesized that the quickness will peak at some point due to the overhead. The results show that the number of seconds will continuously decrease with the increase of the number of processors, as is plotted below:

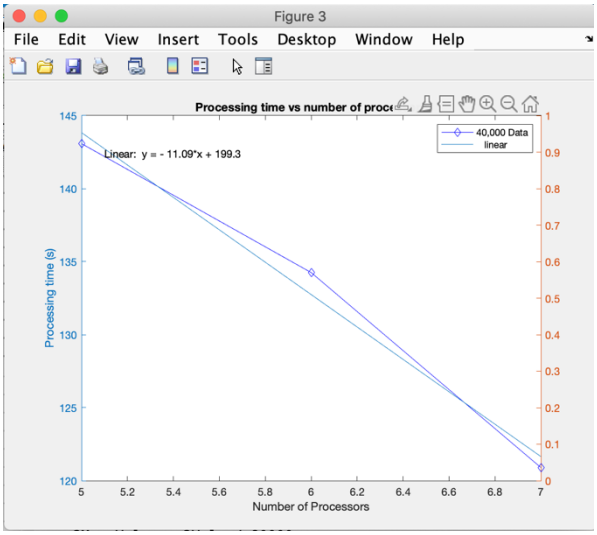


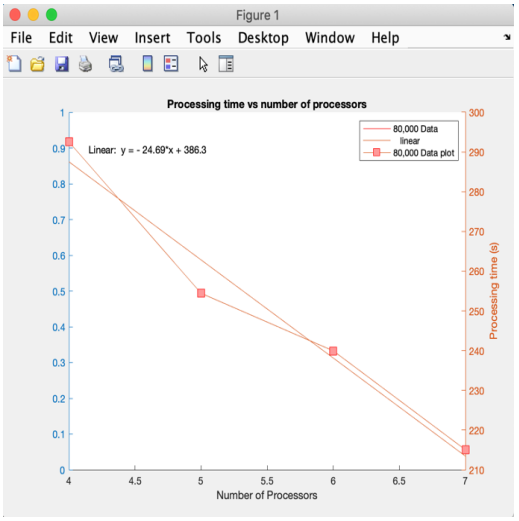
Once processing time is recorded, the mean processing time can be calculated. It is done by dividing time by the number of data or expressed as Mean = Processing Time/Number of Data. For example, the time it takes to process 80000 data using 1 processor is 1046, which means we did it by dividing 1046 with 80000. It will form a range that varies differently between processors, with a trend of decreasing.



From this plot, we can see that when processing a bigger number of data, the processing time decreases, although the rate at which the time gets smaller decreases as well. With smaller data like 10000, the overall trend is also decreasing, however, there tends to be a point where it fluctuates and increases a bit, such as the difference between 7 and 8 processors.

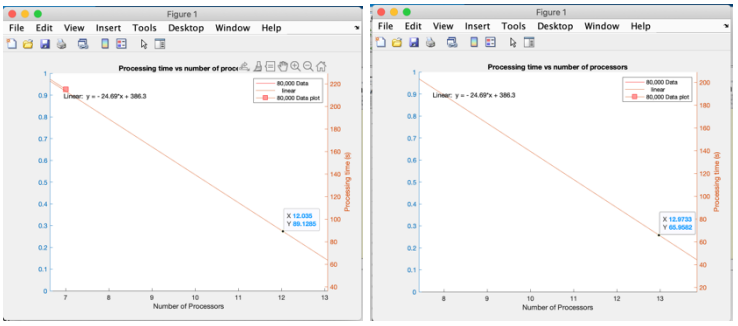
Later, the plots are fitted with the line created from the linear extrapolation from the processing time graph and make it as straight as possible to fit the linear. It is done by reducing the number of individual plots within the graph to make the shape of the graph is close to linear. for 80000, the plots that are removed are the plots from 1, 2, 3, and 8 processors, being nearly fitted within. For 40000, since the graph is less linear than 80000, more of the plots must be removed to fit in the linear.





Once the graph has been plotted, we can make predictions by using basic fitting to make predictions for the required processors needed for this project. For this purpose, only 80000 data will be used as a sample for the predicted requirements. The first step is using the linear equation y = mx + c, where x is the number of processors and y is the time required. Based on the linear equation we can calculate the mean time per data by using the formula (mean time = Number of Data x Hours / Total Time) and Time for all data can be calculated by using formula (Total Time = Mean Time Per Data * Total Number of Data * Hours). The overall area of and areas 277000 data, the amount of data that the file has. The total time is 2 hours, hence it would be equal to 7200 seconds. The average is 1/961 data per second. Since the sample used is 80000, it will be multiplied by the sample and meantime is 84.24 seconds.

Once the mean time per data is detected, the linear extrapolation will be used to estimate the number of processors required for the data. The mean time per data is inserted in one of the y-axes and the predicted requirements are x when it goes below or equal to the mean time per data. The results are the following:



Although 12 is close to the average, it is still greater in value compared to it. Hence the predicted requirement for the processing is at least 13 processors.

Once the analysis is done and the results are recorded, a conclusion can be made between each outcome. A table is also being made as a comparison between Sequential and Parallel using different numbers of processor using 1 hour each, for this table only, it is done using manual processing:

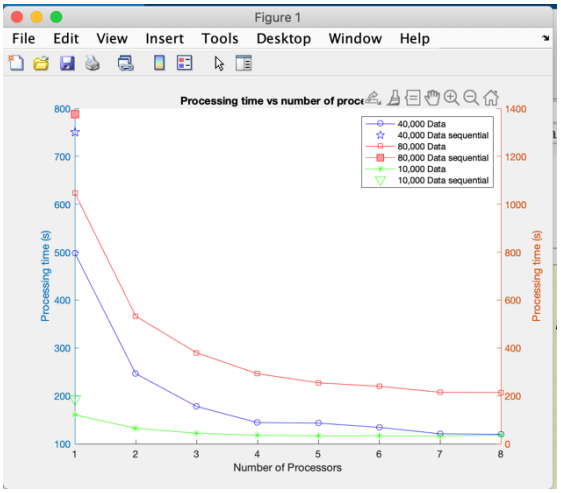
10000 Data	
No. of processor	Time
Sequential	187.22
1	120.92
2	64.74
3	43.83
4	35.39
5	32.91

6	33.54
7	31.87

40000 Data	
No. of processor	Time
Sequential	750.91
1	497.96
2	246.35
3	177.99
4	144.21
5	143.08
6	134.25
7	120.90
8	119.72

80000 Data	
No. of processor	Time
Sequential	1375.48
1	1046.65
2	532.97
3	380.14
4	292.53
5	254.43
6	239.95
7	215.07
8	214.36

The chart result in the following graph:



Conclusion and Recommendations:

Based on the experiment, parallel processing is the recommended processing method, due to its capability to process the same amount of data in a shorter time, when a parallel processing method is used using a single processor. With Parallel processing, the recommended number of processors to process the data is at least 13 processors to full fill the required average time.

Since the results have shown that increasing the number of processors continuously reduces the amount of processing time, especially in larger numbers like 227000. Different machines can provide different results, due to different processor specifications and conditions. Additional research with more devices allows more accurate data, due to more results to compare. Since the decrease in time gets slower as the number of processors got bigger, it shows how the trends might defer if a larger number of processors are used such as 60.

References:

Draw.io. (n.d.). *Flowchart Maker & Online Diagram Software*. App.diagrams.net. https://app.diagrams.net/
ECMWF *Datasets*. (2018, April 6). ECMWF. https://www.ecmwf.int/en/forecasts/datasets

Fajar, A., (2021). *5011CEM2021*. [online] Github Enterprise.
https://github.coventry.ac.uk/althofpana/5011CEM2021

FITZPATRICK, J. M., & LÉDECZI, Á. (2015). *COMPUTER PROGRAMMING WITH MATLAB* (pp. 99–101). https://www.cea-wismar.de/pawel/Study/z_Links_and_Sources/2015--ComputerProgrammingWithMATLAB--Textbook-PDF-3rdRevisedEdition-June-2015.pdf

GanttProject. (2019). *GanttProject*. Ganttproject.biz; https://www.ganttproject.biz/
MATLAB R2020b - MathWorks. (2021). Mathworks.com. https://www.mathworks.com/products/matlab.html
MATLAB Online (n.d) MathWorks https://matlab.mathworks.com/

MATLAB: Programming Fundamentals (R2021a, Vols. 2–26). (2021). MathWorks. (Original work published 2004) https://www.mathworks.com/help/pdf_doc/matlab/matlab_prog.pdf

Appendix:

Combined Automated Testing Code Description:

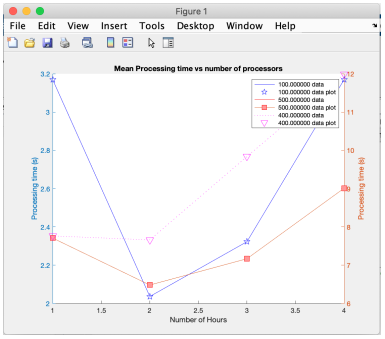
There is also a file where both the parallel and sequential processing is combined that allows both processing methods to be compared using the same plot. The hour arrays are removed from the data and several variables are merged, one of which is the array that records the processing speeds. The number of hours is limited to 3 hours due to the client setting the limit for time to be below 2.5.

More into sequential Processing:

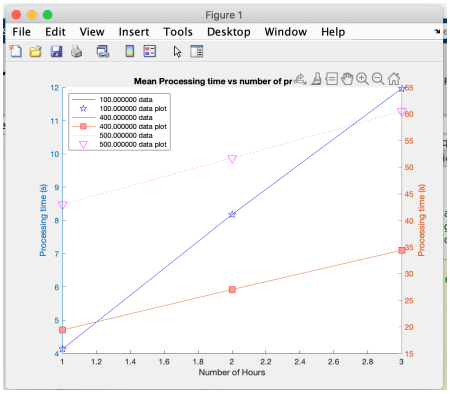
Once all steps have been completed, it will end the if statement, followed by the data loop. Once the loop is closed, a variable will record and calculate the total processing time for the hour of the loop. After the results are printed, the hour loop is closed using the end statement, and the total processing time for all hours will be calculated and printed.

Being Tested with a Smaller Data:

When being tested with smaller data, such as 100, 400 and 500 numbers of data with this case, the trends differ significantly in contrast with using larger data, namely above 40000. Instead of continuously declining, the time required to process fluctuates, having its lows at 2 processors for parallel processing.



With sequential processing, when being tested by smaller data, the results form a linear plot, with a near consistent rate when increasing, which the increase is less linear with bigger data sizes and changes rate slightly.



Overall, both plots have shown that smaller data produces a different result than big data, even though the processing is the same. Since this is a big data project, the results of the big data are considered more accurate and relevant.

More into Parallel Processing:

The Parameters provided in parallel processing is more complex than the one that is provided in sequential, it contains the starting point and the numbers latitude and longitude in addition to the 3 parameters that is also provided in sequential.

In Parallel analysis, there are several variables and loops to pre-process before the actual processing. It started with a variable that contains the pool size, which will determine the number of workers that the analysis will use, followed by an if statement that will set the number of workers if the parallel pool is empty or turned off. Once the parallel pool is set, it will be followed by a function that will attach the ensembled value, where it allows all of the processors to access the file. Attaching is done to allow the file to be accessible to individual processors without parsing it each time, followed by a variable to monitor the processing. And then there is a variable that contains the function to create variables within the parallel pool. Later the waiting bar is set to prevent the user from forcefully stopping the processing.

SMART Targets:

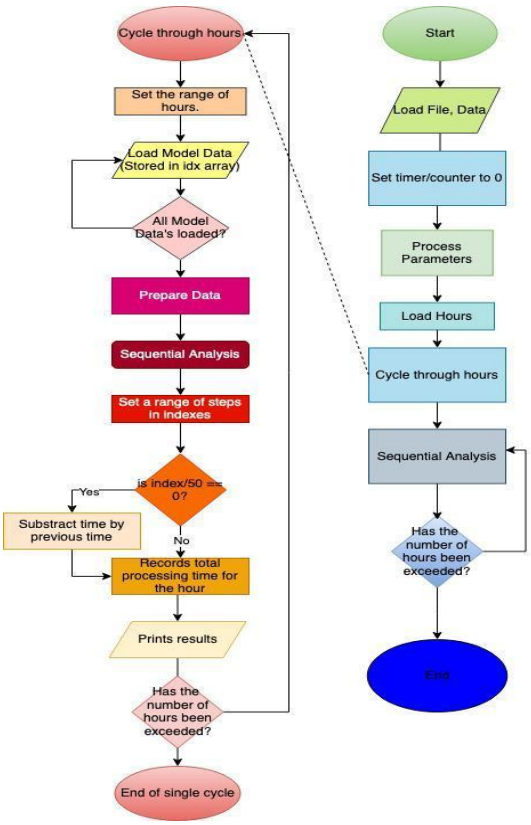
SMART target or criteria stands for Smart, Measurable, Achievable, Realistic, and Time-Bound. For this project, five SMART targets are created. The following are the SMART targets, with 3 relating to the code and 2 relating to the project itself.

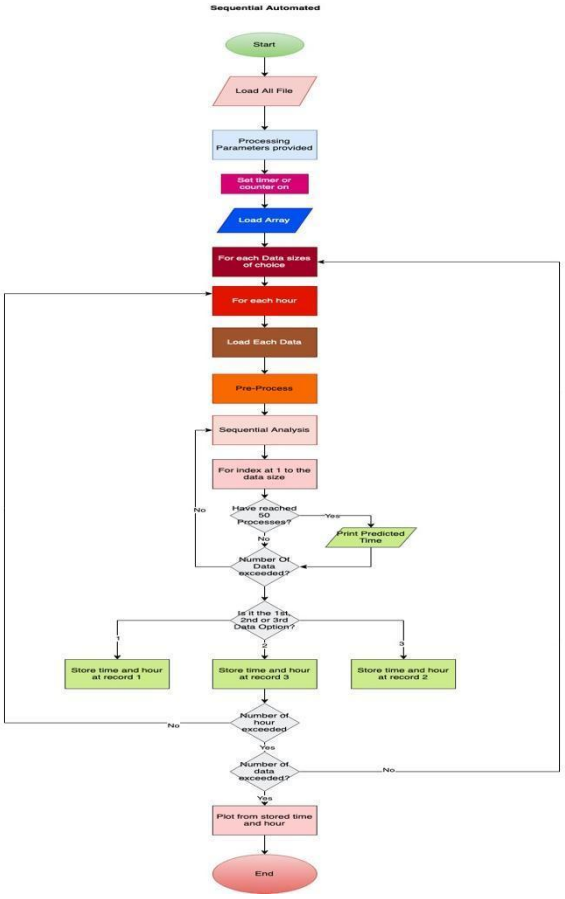
1. The number of processors that are being tested for parallel processing by the user is around 1,2,3,4,5,6,7, and 8 workers. From the 8 Hours of processors. The client requested 8 processors, so the processors that the customer requested is within that range. After calculating, the number of processors estimated to be required is 13.
2. The number of hours that the customer demands are anywhere below 2.5 hours or around 24 hours in under 2 hours. For Parallel Processing, tests are done within 1 hour and for Sequential Processing, the test is done from 1 to 3 processors.
3. In parallel, the number of Locations is (latitude-4) multiplied by (longitude-4), both of which are set by the customer. The latitude and longitude are 400 and is set to start at 1. So overall, there are 156816 Locations.
4. The hardware or machine that is used for this project is a 2018 MacBook Air and the Operating System is MacOS Catalina version 10.15.6. It has a processor of 1.6 GHz Dual-Core Intel Core i5. It has random overall and is 8 GB and a storage capacity of 121 GB (excluding cloud). The name of the software used for this project is MATLAB, which is developed by MathWorks, and the script used for this analysis is called the MATLAB script. An online version of MATLAB will also be used as a secondary device and provides more accurate results due to being in an ideal condition.
5. The module officially takes place between the 27th of January 2021 from the first lecture, up to the 16th of April 2021, where the Report is originally due. Each week there is a total of one session or meeting, since there are 13 weeks, there are 13 sessions in total. Each session also has 2 hours, which means that there are 36 hours of the overall session. The original target of the module is to spend 150 hours on the subject throughout the semester, however, I spent exactly 272 hours on the module after recorded it on my log book.

Flowchart:

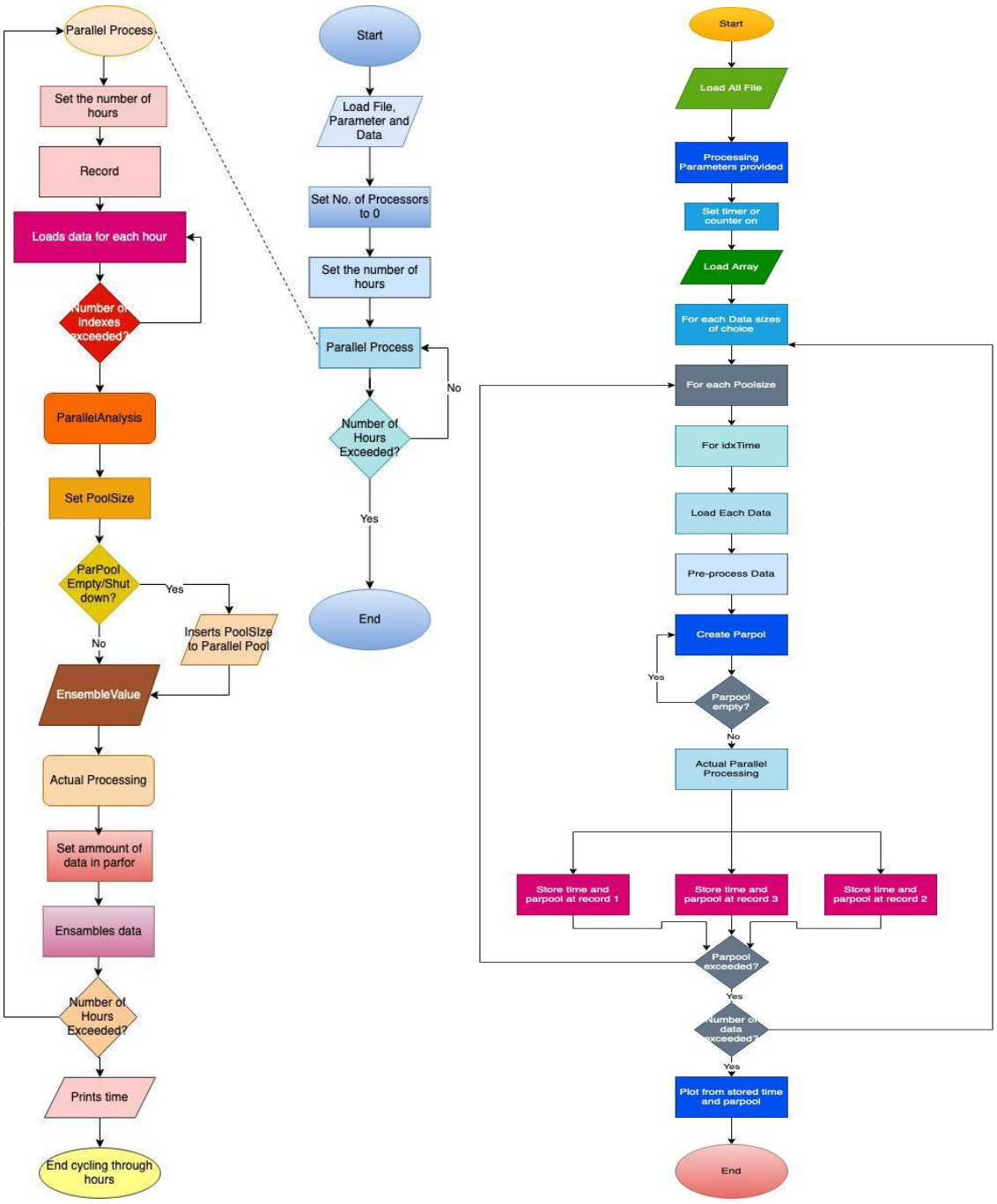
A flowchart is one of the ways that a process (namely codes) or workflow can be planned and represented using graphics and symbols. It represented by different shapes that could mean different processes and the flows are directed by the arrow. For example, the oval represents the beginning and the end, while the parallelogram represents inputs and outputs. The following contains the flowchart for parallel and sequential processing, both containing 2 versions each, the automated and the manual testing.

The following contains the flowchart for the sequential processing, both containing the Manual Testing (left) and the Automated Testing (right).


Cycle through hours
Set the range of hours.
Load Model Data (Stored in idx array)
All Model Data's loaded?
Prepare Data
Sequential Analysis
Set a range of steps in indexes
is index/50 == 0?
Yes
No
Substract time by previous time
Records total processing time for the hour
Prints results
Has the number of hours been exceeded?
End of single cycle
Start
Load File, Data
Set timer/counter to 0
Process Parameters
Load Hours
Cycle through hours
Sequential Analysis
Has the number of hours been exceeded?
End


Sequential Automated
Start
Load All File
Processing Parameters provided
Set timer or counter on
Load Array
For each Data sizes of choice
For each hour
Load Each Data
Pre-Process
Sequential Analysis
For index at 1 to the data size
Have reached 50 Processes?
Yes
No
Print Predicted Time
Number Of Data exceeded?
Is it the 1st, 2nd or 3rd Data Option?
1
2
3
Store time and hour at record 1
Store time and hour at record 3
Store time and hour at record 2
Number of hour exceeded
No
Yes
Number of data exceeded?
No
Yes
Plot from stored time and hour
End

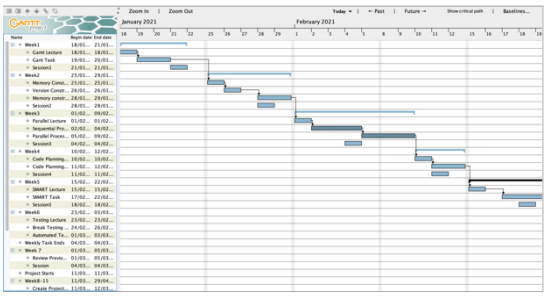
And below is the flowchart for Parallel Processing for both manual testing (left) and automated testing (right)

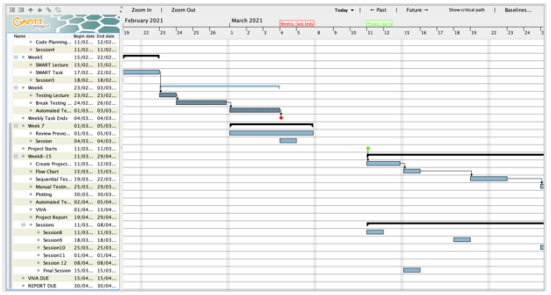


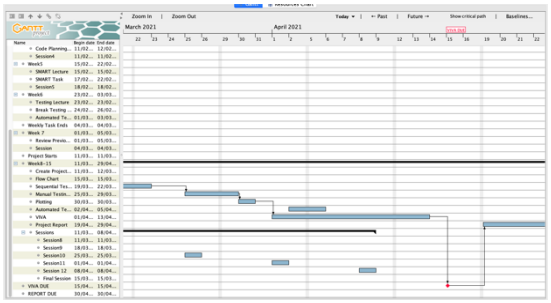
For Automated testing, the flowchart is simpler and focused more on the added features from the manual testing.

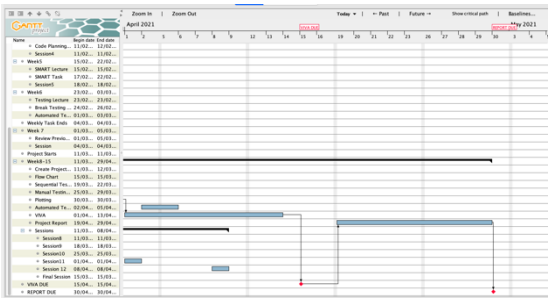
Gantt Chart:

A Gantt Chart is one of the charts that allow the user to plan on which tasks it will do. It represents the schedule of a project using graphical representations. For this project, the planning has been done by using Gantt chars, created using the Gantt Project software. The following is the Gantt chart that is created to plan the project









LogBook:

In contrast to Gantt Chart, a Logbook will record each activity that is done. It will track the amount of time that the user spends on the task and the progression of the user. Logbooks are created in a table and mostly created with table software such as Microsoft Excel or Google Sheets.

Date and Time	Hou	Activities	Progress	Achievement
21/01/2021	2	Session1	Introduction to the module	
21/01/2021	4	Gantt Lecture	First lecture of the module	
24/01/2021	4	Gantt Tasks	Finished all the gannt task	Week 1 Finished
26/01/2021	2	Memory Saving Lectures		
26/01/2021	2	Version Control Lectures		
26/01/2021	4	Memory Saving Tasks	First coding tasks involving MATLAB. Finished the exploring and analyzing the first file.	
28/01/2021	2	Session2		
28/01/2021	3	Memory Saving Tasks	Second part of the task. Finished analyzing the memory saving file.	Week 2 Finished
30/01/2021	3	Parallel Processing Lecture		
30/01/2021	4	Sequential Processing Task	This task is an introduction on doing a sequential analysis and processing on an nc file that is provided by the client.	
04/02/2021	2	Session3		
04/02/2021	4	Parallel Processing Task	Learned on how to set up data memory, modifying the ammount of core, and extract the results to create a plot	
08/02/2021	2	Parallel Processing Task	Successfully estimated the requirments for the ammount of processors.	Week 3 Finished
08/02/2021	1	Code Planning Lecture Part1		
10/02/2021	1	Code Planning Lecture Part2		
11/02/2021	2	Code Planning Tasks	Introduced to the Flow Chart. Tfinished Task 1 and Task 2	
11/02/2021	2	Session4		
12/02/2021	1	Code Planning Tasks	Finished Task 3, unable to progress to Task 4 due to the required to automated testing	
16/02/2021	2	SMART Lecture		
16/01/1900	2	SMART Tasks	Finished Task 1 and Task 2	
18/02/2021	2	Session5	Discussion about emails, followed by a support and Q&A sessions	
18/02/2021	2	SMART Tasks	Finished Task 1 and Task 2	
23/01/2021	2	Testing Lecture		
23/01/2021	6	Break Testing Task	The first part of the week 6 Tasks. Finished Task 1, 2, and 3.	
25/02/2021	2	Session6		
25/01/1900	4	Break Testing Task	Finished task 3,4 and 5.	
	2	Automated Testing Task	The second part of the Week 6 Tasks and the Final Weekly Tasks of the module before starting with the project. Successfully automate.	
	1	Code Planning tasks Part 3	Finished the final task of part 3 after completing the automated testing task	All weekly tasks completed
04/03/2021	2	Session7	Briefing on the project and what to expect. Followed by support session	
04/03/2021	6	Review All Task	Review the previous tasks that has already been learned so far	
10/03/2021	1	Setting up for the project	Creates folders and download all the files required	Project Started
10/03/2021	2	Code	Started making parallel and sequential processes for the project	
11/03/2021	2	Session8		
11/03/2021	3	Manual Testing	Seperated the manual parallel and sequential processing file into several files.	
	4	Flowchart	Started and finished the first attempt of the flowchart for both parallel and sequential processing	
18/03/2021	2	Session9		
19/03/2021	2	Code	Started manual plotting of parallel processing with 10000 Data	
25/03/2021	2	Session10		
01/04/2021	2	Session11		
07/04/2021	5	VIVA Presentation	Started Making the presentation for VIVA	ViVA Started
08/04/2021	2	Session12		
11/04/2021	6	Manual Testing	Manual Testing with 80000 Datas with 8 number of processors and plot it.	
12/04/2021	7	Manual Testing	Manual Testing with 40000 Datas with 8 numbers of processors	
13/04/2021	3	Predicted Requirement	Calculates the predicted requirements	
15/04/2021	2	Session13	Final session with Richard, Q&A about VIVA	
15/04/2021	3	Code	Plot the results that will be required for the VIVA presentation	
15/04/2021	4	VIVA Presentation	Finished with the Results and Predicted Requirements	
16/04/2021	14	VIVA Presentation	Final VIVA recording and submission	VIVA Submitted
19/04/2021	14	Report	Creates the abstract and the introduction for the essay	Report Started
20/04/2021	12	Report	Starting	
21/04/2021	10	Report	Started Making the Abstract and Introduction of the essay	
22/04/2021	12	Automated Testing	Re-do the automated testing	
23/04/2021	10	Automated Testing	Successfully run different number of processors on parallel processing and creates automated plots	
24/04/2021	10	Automated Testing		
25/04/2021	12	Automated Testing	Finished with the automated running and plotting	
26/04/2021	10	Appendix	Finalizes flowchart and SMART Targets	Appendix Finished
27/04/2021	8	Essay		
28/04/2021	4	Essay	Finalizes essay and	
28/04/2021	4	Automated Testing	Attempted to implement break testing	
29/04/2021	8	Essay	Finalizes references and logbook	
30/04/2021	14	Final Report Day	Finalizes report before Submission	Report Submitted
Total Hours	272			End of the Module