CMP - 202 Presentation

By Joseph Roper

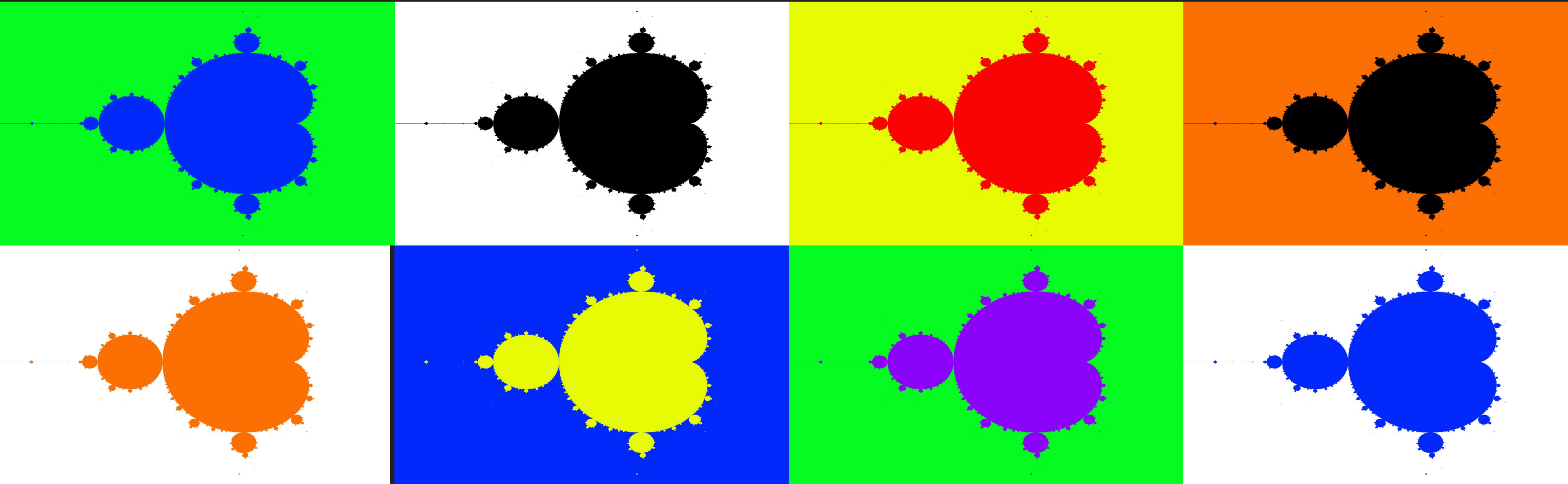
Mandelbrot Program

- This produces an image by plotting complex numbers calculated using a formula across an imaginary plane.

- I am trying to make this computation faster by parralising the program.

Mandelbrot Colours

In my project I allow the user to choose the foreground and background colour of the mandelbrot image. The colours available are; white black, red, green, blue, orange, yellow and purple.



Parallelized Code

The Mandelbrot calculation is split up in the project.cpp using two for loops lasting as long as the user defined variables for how many slices they want in the x and y direction. Each slice is then calculated separately using the farm class.

I made a farm class which builds a queue of tasks to run, it then makes as many worker threads as the user has defined, the maximum number is the amount of CPU’s in the users computer. It then computes these threads at the same time resulting in the mandelbrot computation being faster. This was accomplished by using the fork-join pattern.

Threads

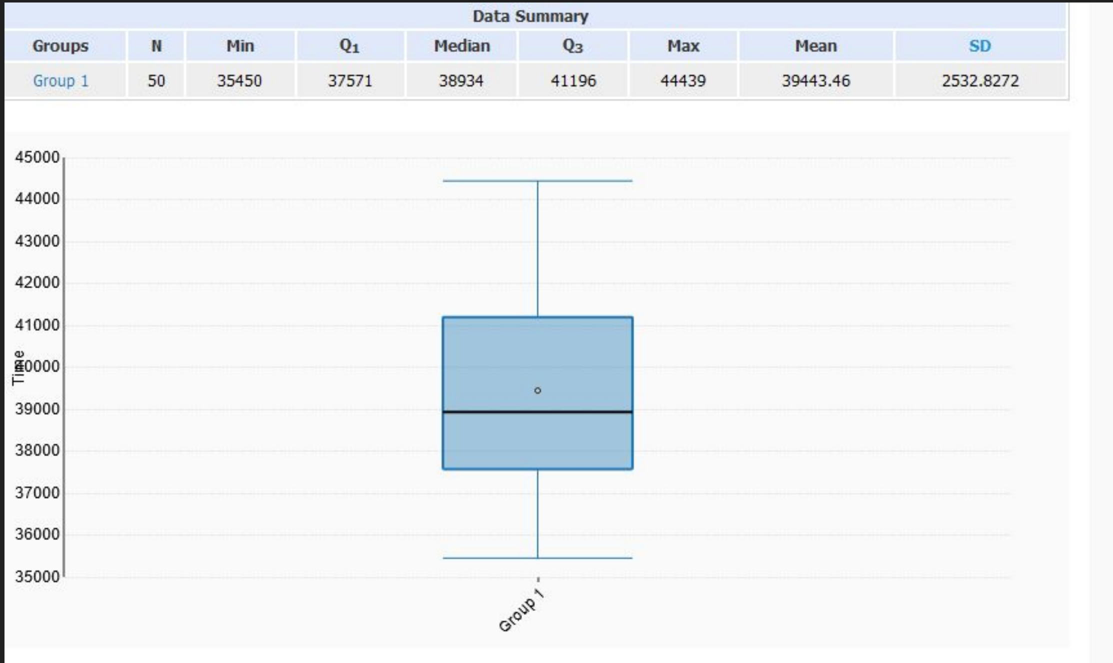
I have 2 thread functions within my program, firstly the farm class which creates as many threads as the user defines based on the CPU amount of their hardware, it then uses these threads to calculate the split up mandelbrot image all at once then joining them together. Every operation using threads is protected by a mutex in my program so that I dont cause a race condition or deadlock issue.

My second thread function is tasked with waiting for the farm class to finish its job before writing the image computed to a file. I do this by creating a condition variable in the farm class which signals the thread function when the image is computed which then unlocks a mutex stopping the write file function to run.

Performance Evaluation

To test my mandelbrot program I ran it 50 times each with 16, 8 and 4 CPU cores being used as well as then running the application with 50, 100 and 200 slices in the x and y of the image.

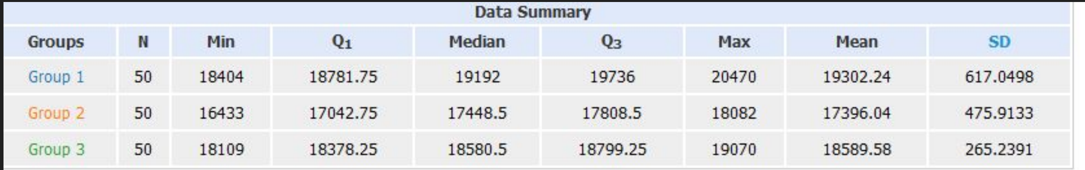
Original Mandelbrot:


Data Summary

Groups	N	Min	Q_1	Median	Q_3	Max	Mean	SD
Group 1	50	35450	37571	38934	41196	44439	39443.46	2532.8272

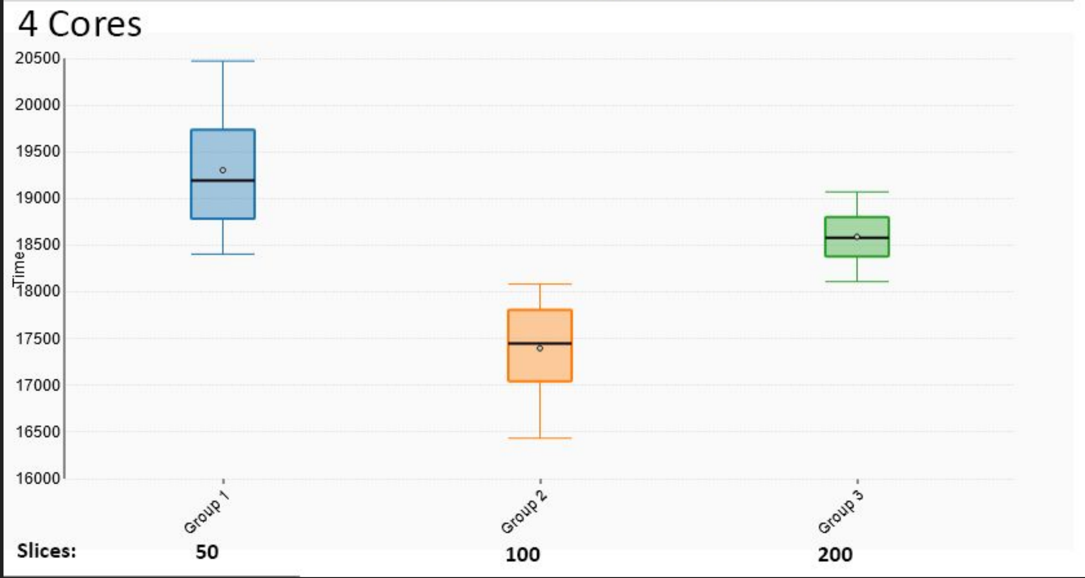
CPU Cores: 4

From these results we can see that the only noticeable difference in times is when 100 slices in the x and y are made. I believe this is because when 200 slices are made it splits up the workload too much for the 4 CPU cores to compute at the same time.



Data Summary

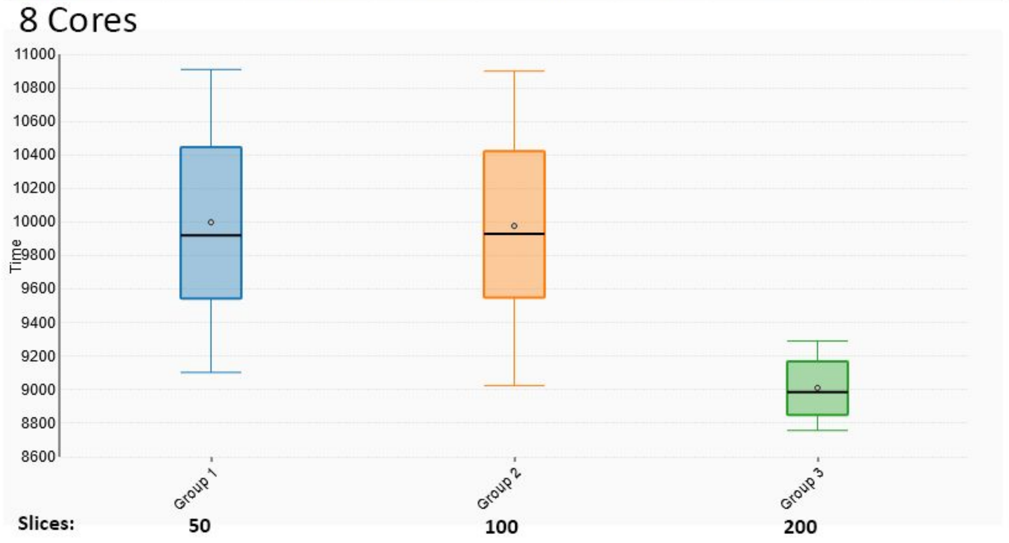
Groups	N	Min	Q1	Median	Q3	Max	Mean	SD
Group 1	50	18404	18781.75	19192	19736	20470	19302.24	617.0498
Group 2	50	16433	17042.75	17448.5	17808.5	18082	17396.04	475.9133
Group 3	50	18109	18378.25	18580.5	18799.25	19070	18589.58	265.2391



CPU Cores: 8

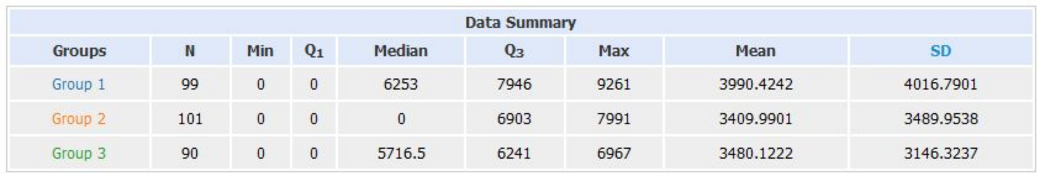
From these results we can see the only noticeable time difference is when 200 slices in the x and y are made. This shows that the more slices the faster the program is.

Data Summary								
Groups	N	Min	Q1	Median	Q3	Max	Mean	SD
Group 1	90	0	0	9258.5	10103.25	10909	5554.5556	5012.2155
Group 2	50	9024	9549.25	9928.5	10422	10899	9976.18	553.5245
Group 3	50	8757	8849.25	8986.5	9169	9290	9010.72	171.0571

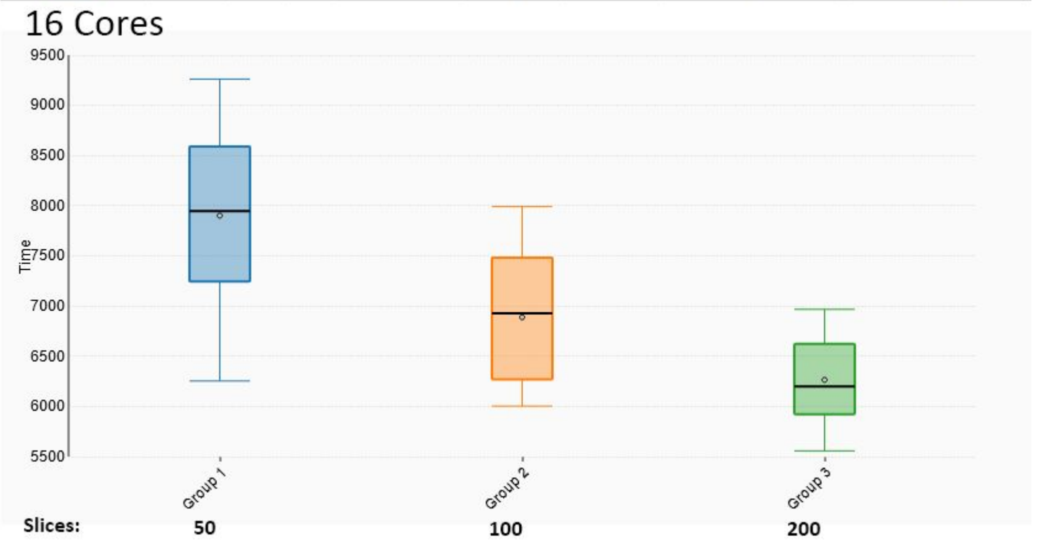


CPU Cores: 16

From these results we can see that when the amount of slices in the x and y direction is higher the program is faster.



Data Summary								
Groups	N	Min	Q_1	Median	Q_3	Max	Mean	SD
Group 1	99	0	0	6253	7946	9261	3990.4242	4016.7901
Group 2	101	0	0	0	6903	7991	3409.9901	3489.9538
Group 3	90	0	0	5716.5	6241	6967	3480.1222	3146.3237



Overall Performance

From testing the mandelbrot extensively i have come to the conclusion that when the mandelbrot is sliced hundreds of times the program will be faster. However this will only take effect if there's a valid amount of CPU cores able to handle calculating multiple parts of the image at once.

Evaluation

Overall I have made the computation time of the mandelbrot faster completing the purpose of the project.

However I do believe the program can be made faster by taking out the write file thread function and just having it run after the mandelbrot code rather than signalling between the farm class as it slows down the program a little bit.