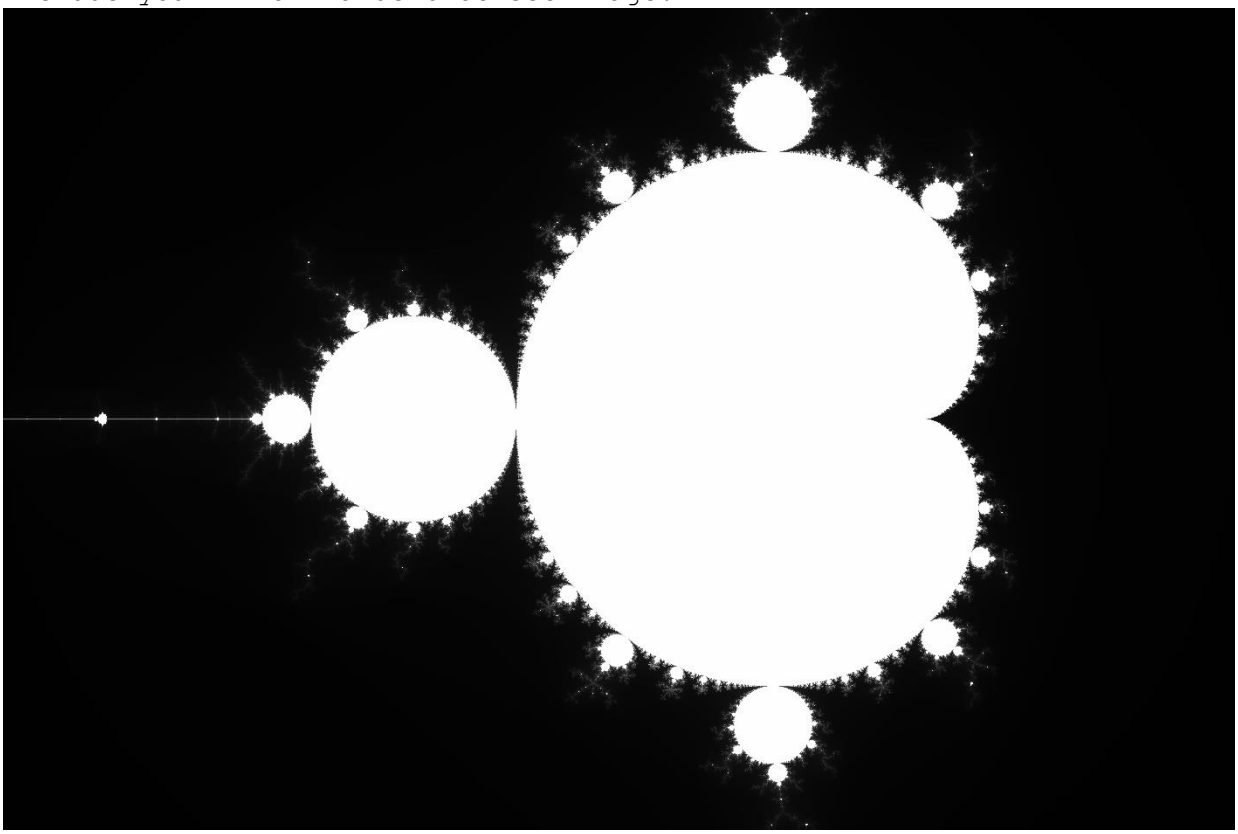
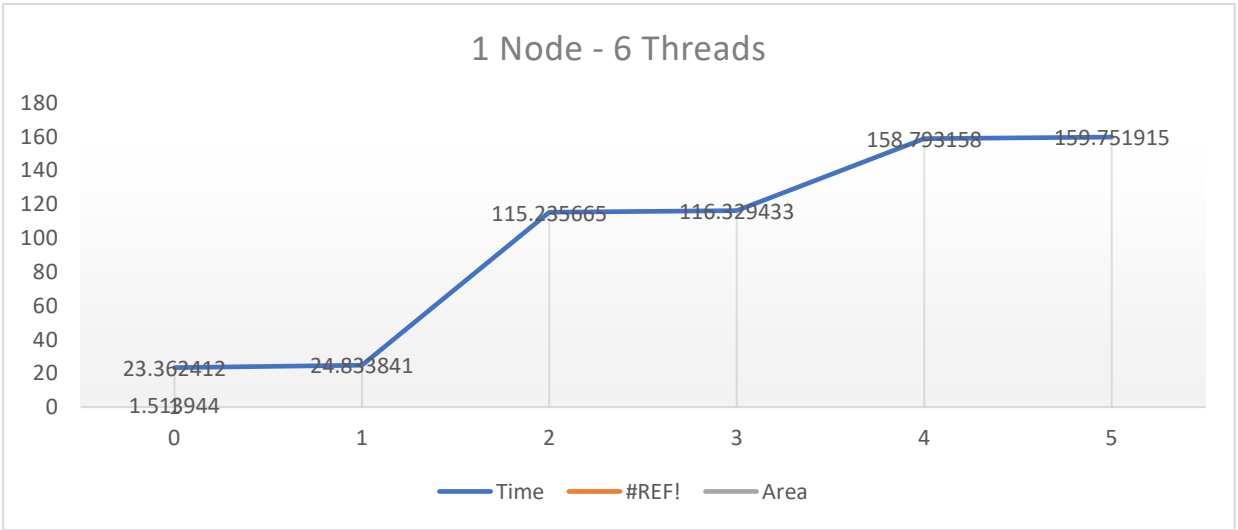
Describe the algorithm that you used to solve the problem.

I used the OpenMP approach for this problem. However, I timed everything using MPI.

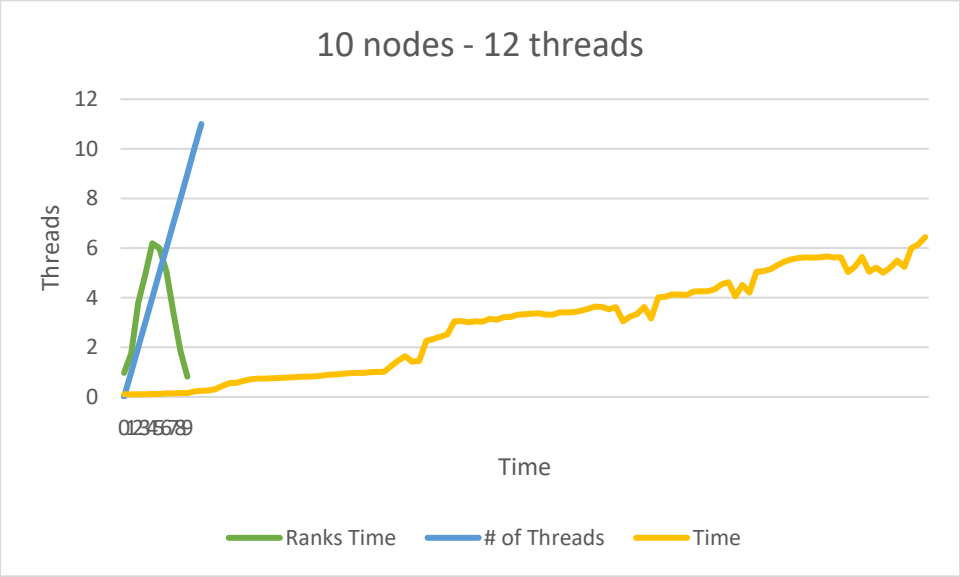
Include your final Mandelbrot set image.

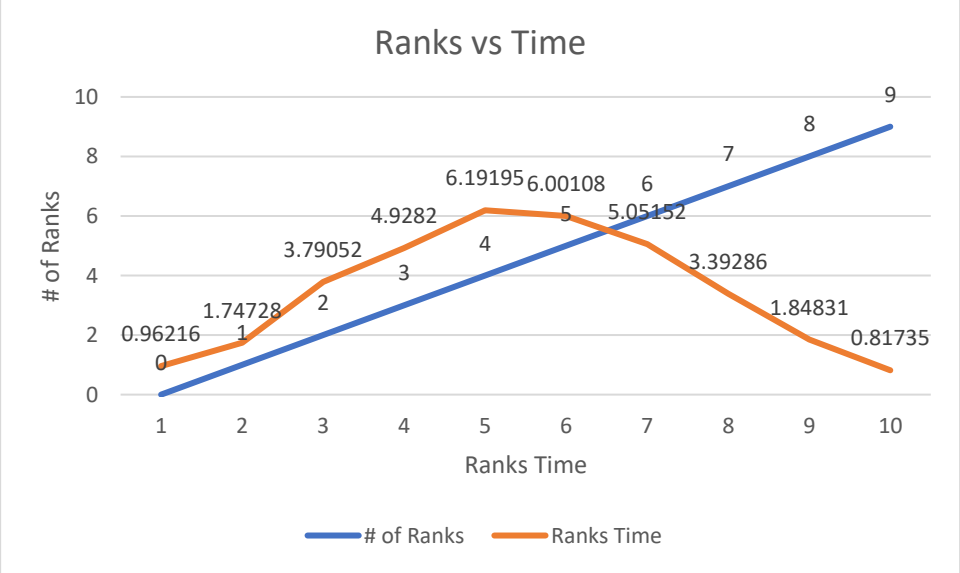


Time your application (excluding the saving of your final JPG image) and create a strong scaling plot.

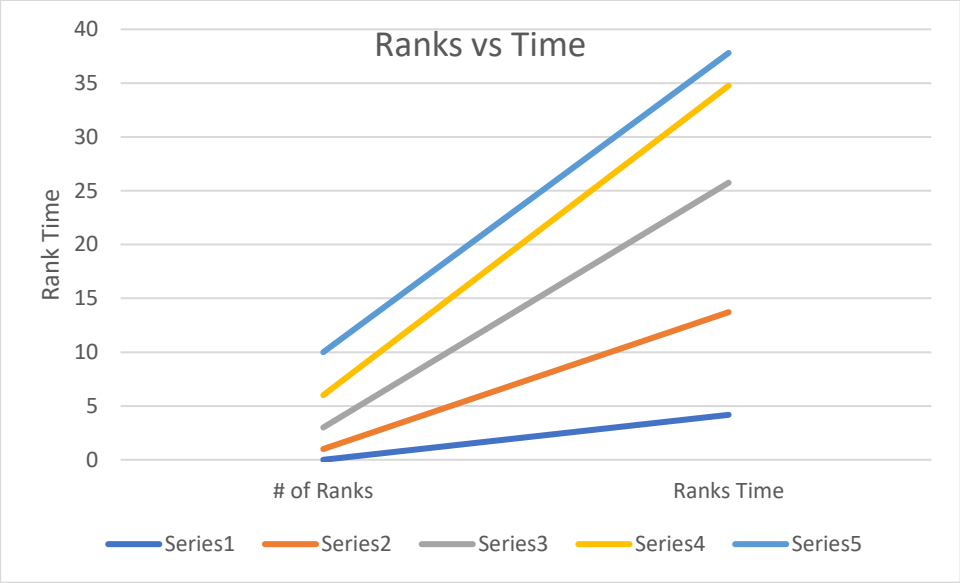


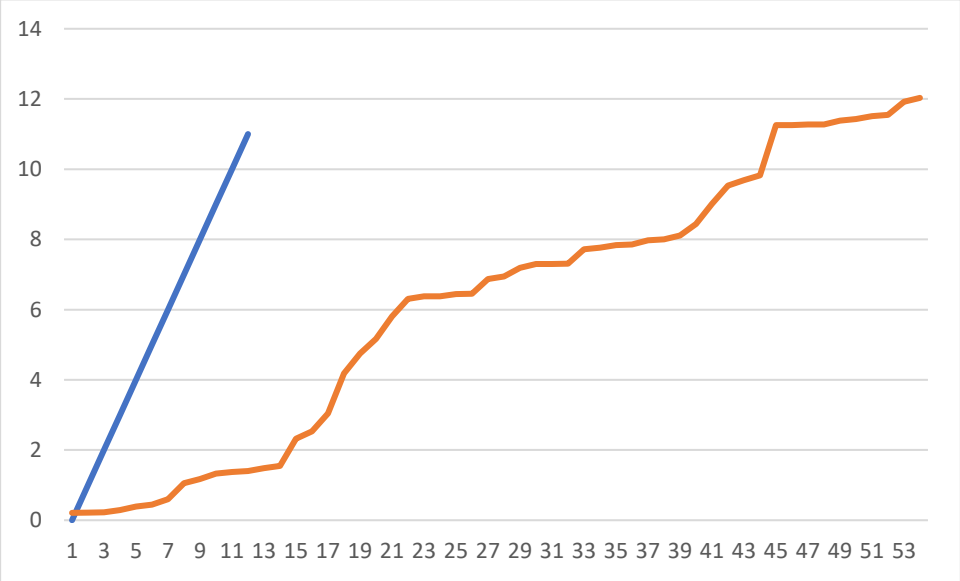
It seems like there is not a significant difference between consecutive nodes. For example, from 0-1 node, there is a very small difference. But if we add one more thread, we see a huge difference. The process repeats; 2-3 nodes, no big of a difference. But 3-4 there's a huge jump.





As we can see here, we have ten nodes and twelve threads – the best performance we can get. As time progresses, we can see that the performance of the threads keeps increasing (yellow line). The green and blue lines show the # of nodes and threads. I pretty much combined the second graph with the first one to see if the results will make more sense.





These final graphs have five nodes and twelve threads. The first graph is about ranks and ranks time. The second one is about threads and threads time. The first one shows that the more ranks we have, the fastest it will be. The second one shows the number of threads available. And, how effective it is while handling the calculations.