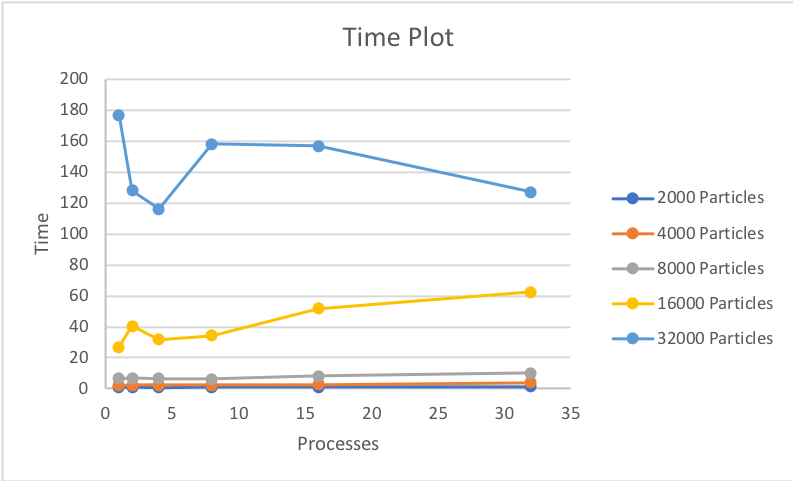
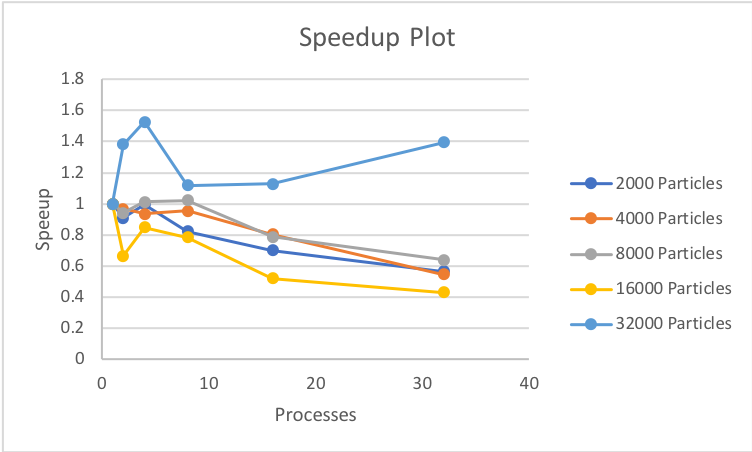
4F03 Final Project Report

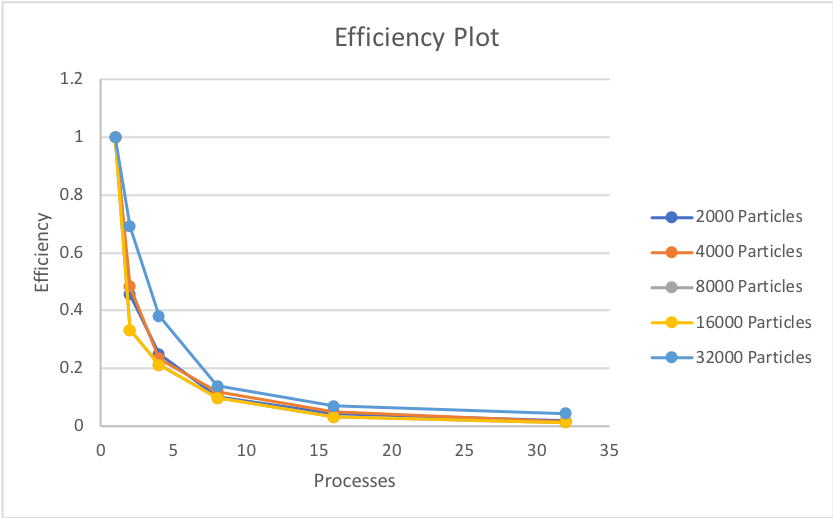
Prabhbir Pooni, Yanting Zhang, Kelvin Lin

400010290, 001207766, ?????

MPI Runtime

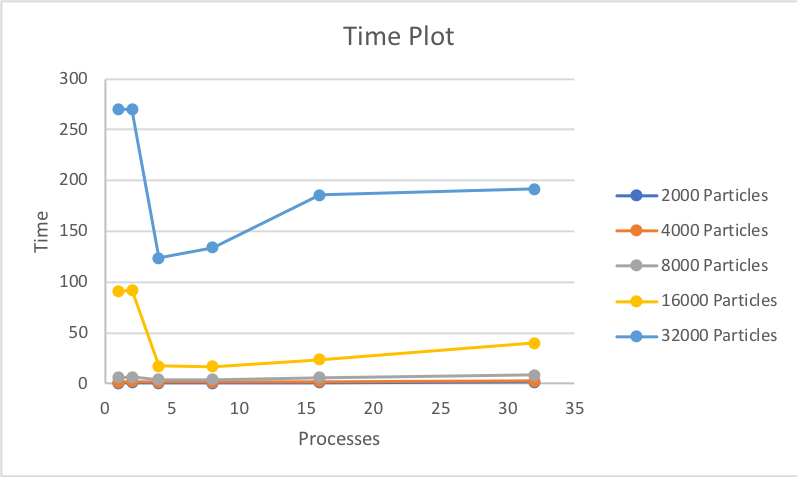




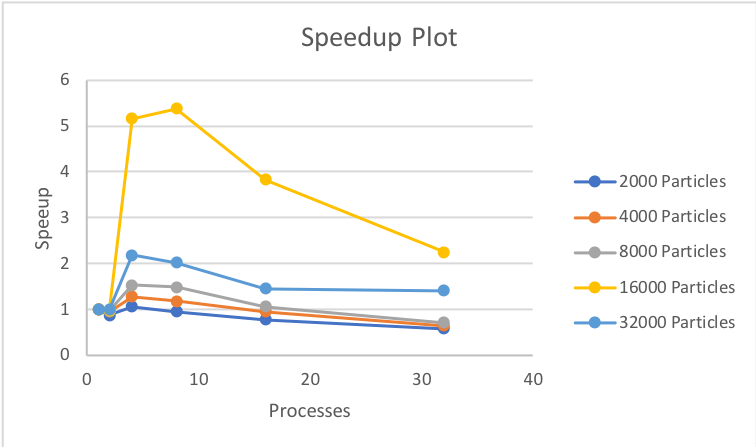


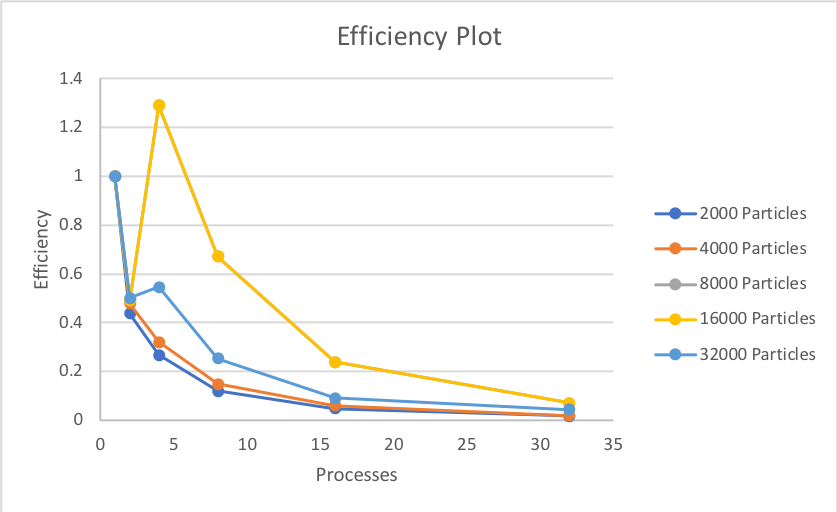
This algorithm is neither strongly scalable or weakly scalable over any given time period. The only anomalies in this data would be the increase in time to run the program as the number of processors increase. This might be due to the overhead required of using MPI or higher demand on the server during the runs for retrieving the time of the lower particle runs.

MPI with OpenMP Runtimes



This implementation of MPI with OpenMP does have better timing than the MPI implementation. This is probably do the distribution of the second for loop which calculates the force, velocity and position over multiple processers. This allows for no hang ups on a single processor which could be doing a larger computation than the other processors.





The MPI with OpenMP algorithm is neither strongly scalable or weakly scalable however it does show signs of being strongly scalable with increased efficiency for 16,000 particles. Perhaps do to the server demand during the 32,000 particles run it may have been unable to show appropriate scaling which could have proved strongly scale ability for larger amount of particles. However, with the current data overall trend is that it is not strongly scalable. The one anomaly in this plot would be at 4 processors where the efficiency exceeds 1. The reason for this might be due to better caching done by the program however that would be unlikely as it does not normally lead to a 30% increase. It also could be due to a larger amount of work being processed on the server during the serial time which led to a longer computational time for the program.