High Performance Computing

Assignment-1 Report

Team No: 9

Team Members:

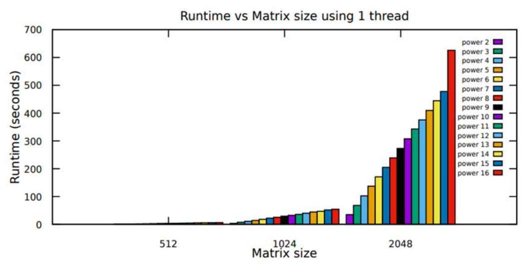
Anirudh Chimpidi – SE20UCSE019

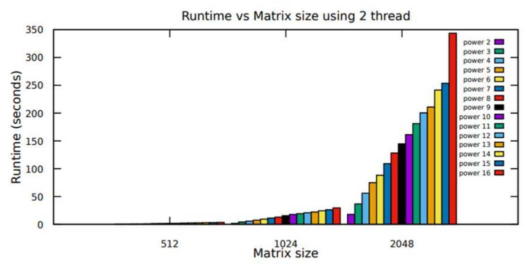
Naga Tharun Makkena – SE20UCSE105

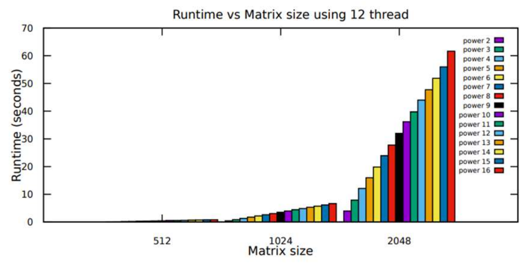
Sri Harsha Vandanapu - SE20UCSE184

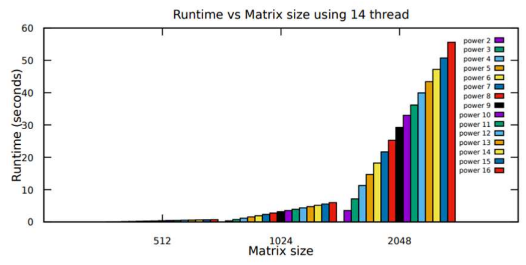
Rohan Potta - SE20UCSE145

Dhanush Bommavaram - SE20UCSE039

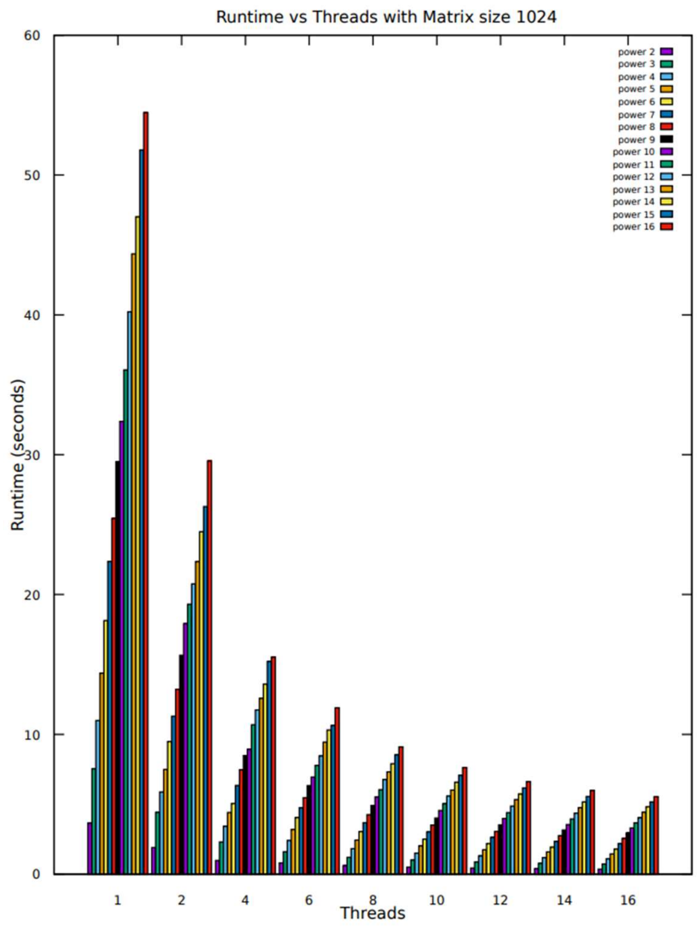




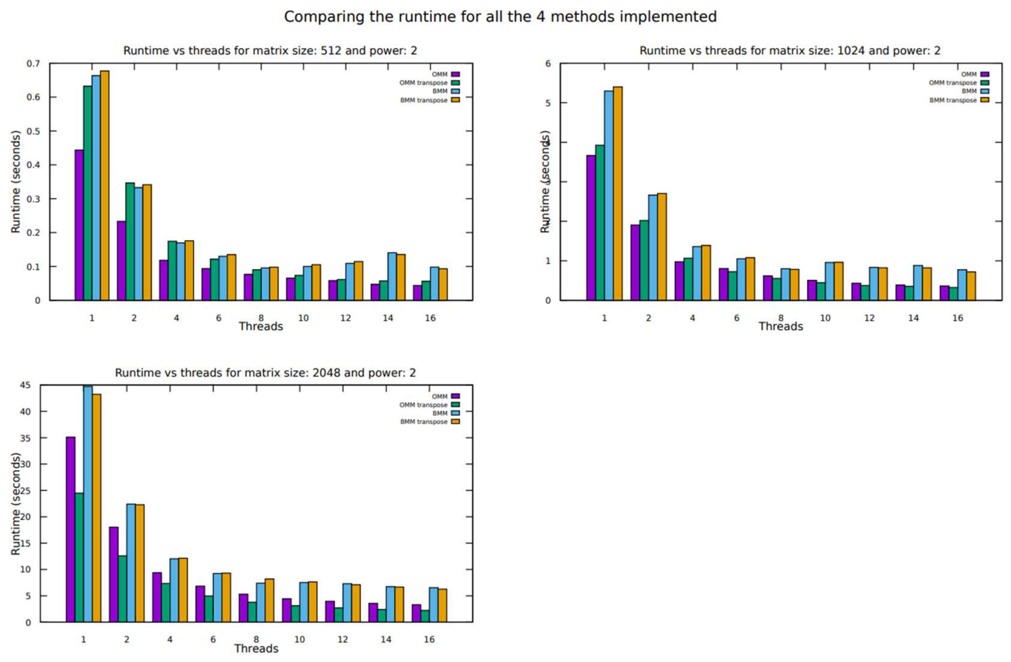


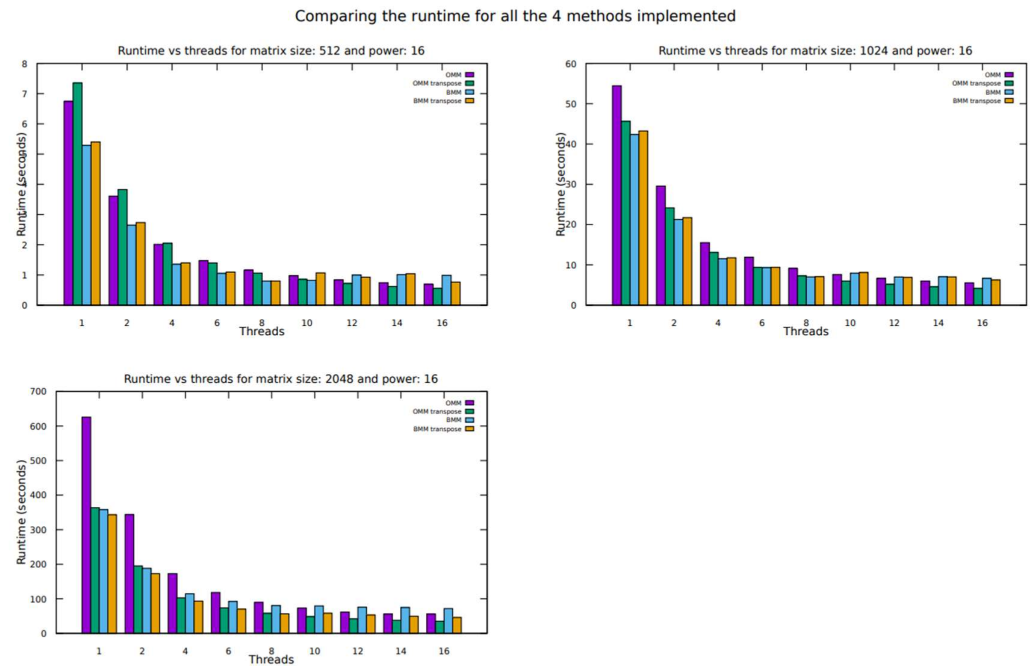


Here, we can observe that by increasing the size of the matrix the run time increases as well. And from the 4 graphs, we can conclude that by using more threads for the process we can reduce the computation cost (run time) by nearly 10 times.



Here, we can observe the difference in run time by varying the number of threads available for the process to run. As the number of threads keep on increasing we can observe an exponential decrease in the run time on a matrix size of 1024. We can also observe the time taken for computation of various powers from 2 to 16 also varies in a linear way.





We can observe that the computation cost of Ordinary Matrix Multiplication (OMM), Ordinary Matrix Multiplication using transpose of A (OMM transpose), Block Matrix Multiplication (BMM), Block Matrix Multiplication using transpose of A (BMM transpose).

From the above set of plots, we can conclude that for smaller sized matrices, low powers OMM takes significantly less time than the other methods. And for larger sized matrices and large powers the BMM and BMM transpose takes less time. The transpose methods take less time as the memory access is Row Major which increases the memory access time for columns in OMM and BMM. By using transpose of matrix, we can access rows instead of columns thus reducing the access time by a significant amount for larger powers and matrix sizes.