PA 4 (draft)

CS 615

By Samson Haile

April 13, 2017

Assignment Description

The objective of this assignment is to determine differences in computation time when executing matrix multiplication. The computation involved multiplies two square matrices of same dimensions and times how long it takes to calculate the resultant matrix. The execution time is evaluated on different sizes of data, utilizing subsets of a single set of matrices represent the different sizes of data that are timed. In the case of a parallelized algorithm, the computation time is expected to decrease with an increase in the number of cores utilized. With respect to parallel computation, subtasks of the computation are determined by dividing the number of elements to be computed equally over the number of cores utilized. Additional changes might be needed to the parallel program to ensure memory constraints don't interfere with the program's execution. The sequential algorithm will be plotted in a two dimensional matrix dimension size versus time graph. The parallel algorithms will be plotted in a three dimensional matrix dimension size versus cores used versus time graph. An assessment will then be made to explain trends and differences between the graphs.

Assignment Methodology

The assignment was implemented in the form of two c files and two batch files, as well as a makefile. Each c file and batch file match to designate one of the two implementation types of the matrix multiplication. This includes sequential and parallel computation of the matrix multiplication algorithm. The sequential code simply executes the computation code on a single core, iterating across the different matrix sizes. The parallel programs use the general heuristic the sequential does, but involves extra components for properly sending and receiving work between the master and slave nodes. The number of cores the parallel programs run on is modified through the -n and -N parameters of the batch scripts used to run the programs. This is to ensure that the programs request only as much resources as it uses.

The sequential code operates by first reading the data into a two dimensional vector data structure. This operation has no impact on the execution time since it is done before any sorting is performed. After all data is read into their container objects, timing starts. The sequential algorithm then uses three nested for loops in order to calculate the resultant matrix, computing each element at a time. Timing is then stopped upon having calculated every element of the resultant matrix.

The c code is compiled through the usage of makefile which is capable of producing an executable to run the code, as well as the ability to remove the produced executable from the directory. Once the executable is produced, one of two batch files can be run in the form of the

command sbatch <file_name>. The batch file seqMat.sh measures the time it takes to multiply two matrices by each other in a sequential manner.The batch file parMat.sh measures the time it takes to multiply two matrices by each other in a parallel manner.

Data and Analysis

***numerical data is contained in .dat files in repository**

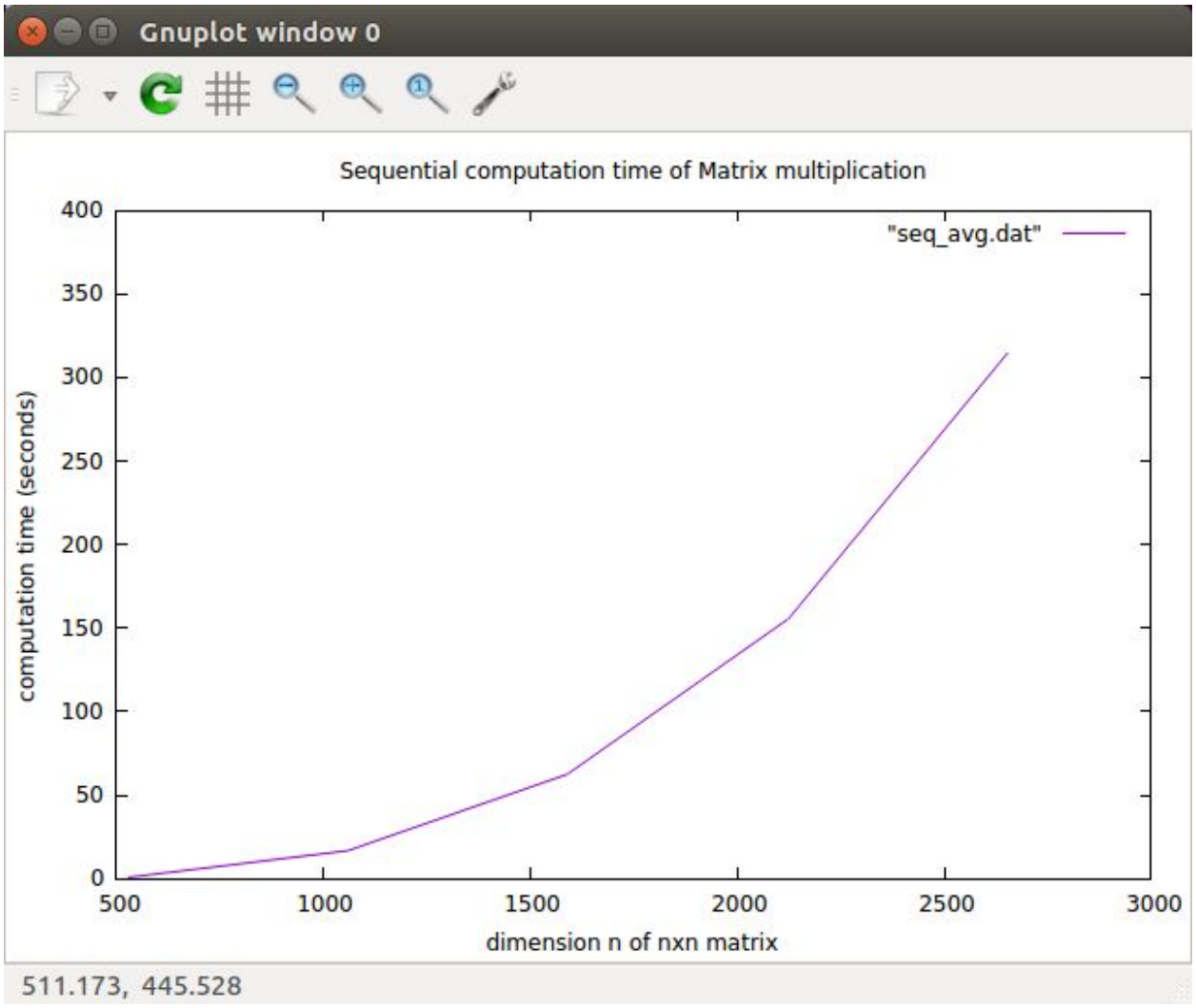

Figure 1:

The graph shows the execution time of sequential matrix multiplication over varying sizes of matrices. The graph displays the polynomial relation between the number of dimensions of the matrices multiplied and computation time. The legend denotes the file representing se the average of 5 sets of sequential runtime data.