# CS301: HIGH PERFORMANCE COMPUTING
# ASSIGNMENT ONE


## Submitted By:
## Shaleen Kumar Gupta (201301429)
## Visharad Bansal (201301438)


## PROBLEM STATEMENT

Calculate the value of pi using the trapezoid rule.

### METHOD

$f(x)= 4/(1+x^2)$

The value of pi is calculated by integrating f(x) in the interval [0,1].
Integration is done using the Trapezoid Rule, where the area is divided into n small trapezoids and then summed.

### COMPLEXITY

The serial code runs in O(n) time, where n is the number of integration steps.

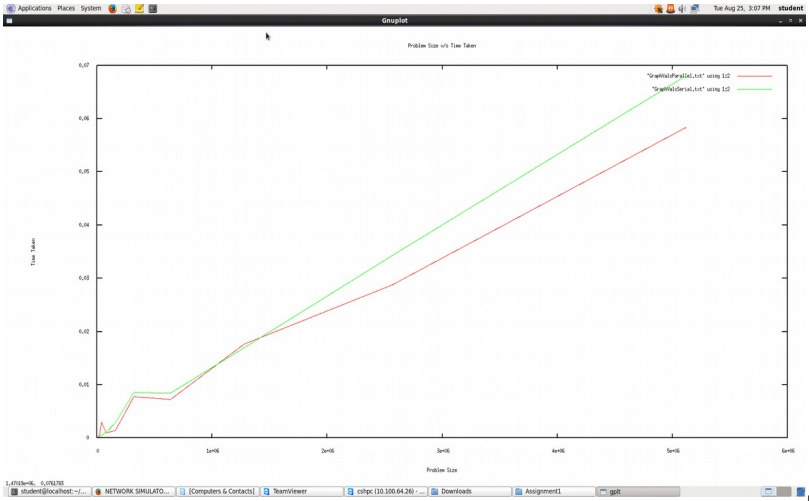
OpenMP is used to divide the serial code among 4 threads, giving a theoretical speedup of 4x

### OPTIMIZATION STRATEGY

The numerical addition has no loop dependency or data dependency, and it can be conveniently parallelized using the rank of each thread. For example, thread with rank 0 runs iterations 0,4,8...n.
Similarly, thread number 1 runs iterations 1,5,9..n etc.

### OBSERVATIONS AND PROBLEMS FACED

Using parallelization introduces many race conditions. In lieu of that, separate variables need to be used of local sums in each thread, and later a global variable is used to calculate the final sum.
Also, time is taken for data access, causing delays. Thus, as can be seen from the curve below, the parallel curve takes more time initially compared to the serial time, but as input size increases, the parallel code gives better efficiency.




## HARDWARE DETAILS

No. of Processors : 3
vendor_id : Genuine Intel
CPU family : 6
model : 60
model name : Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz
stepping : 3
CPU MHz : 800.000
cache size : 6144 KB
siblings : 4
CPU cores : 4

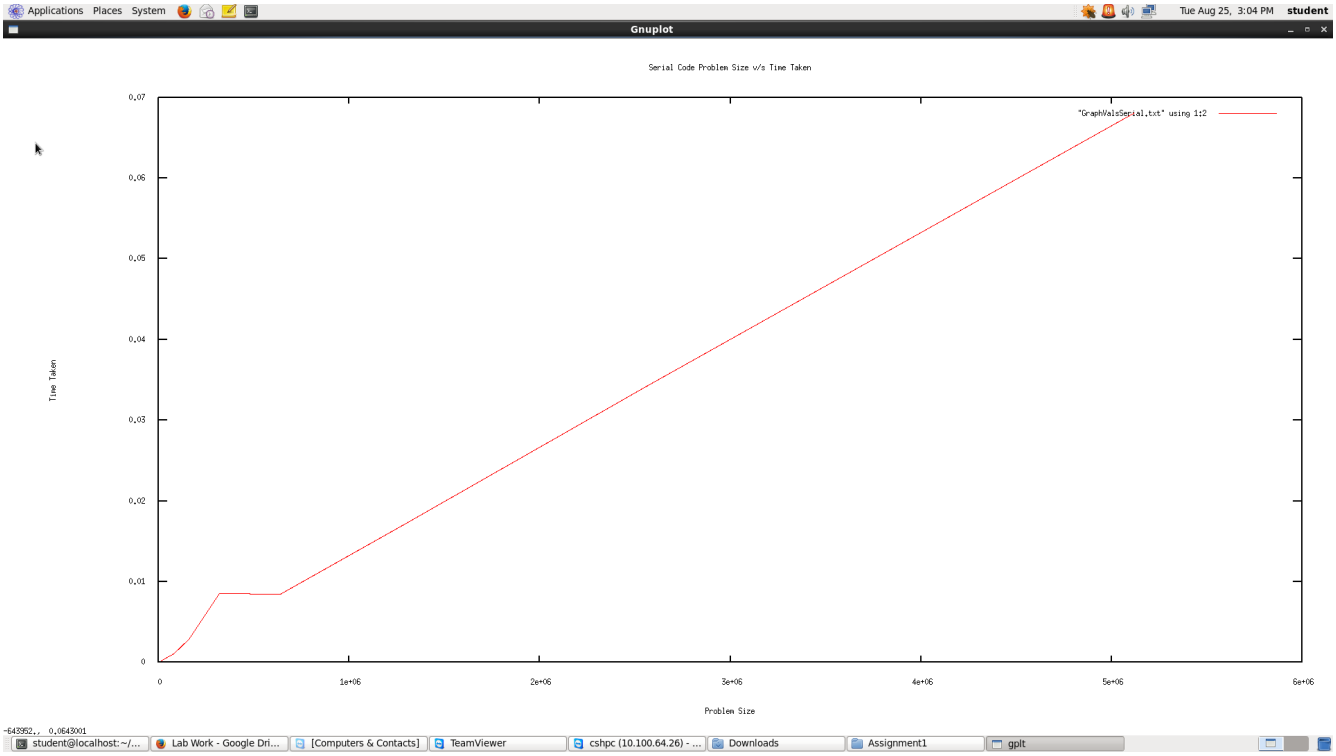
## INPUT PARAMETERS

Number of steps (n).
Range of n has been taken from 1000 to 10000000.

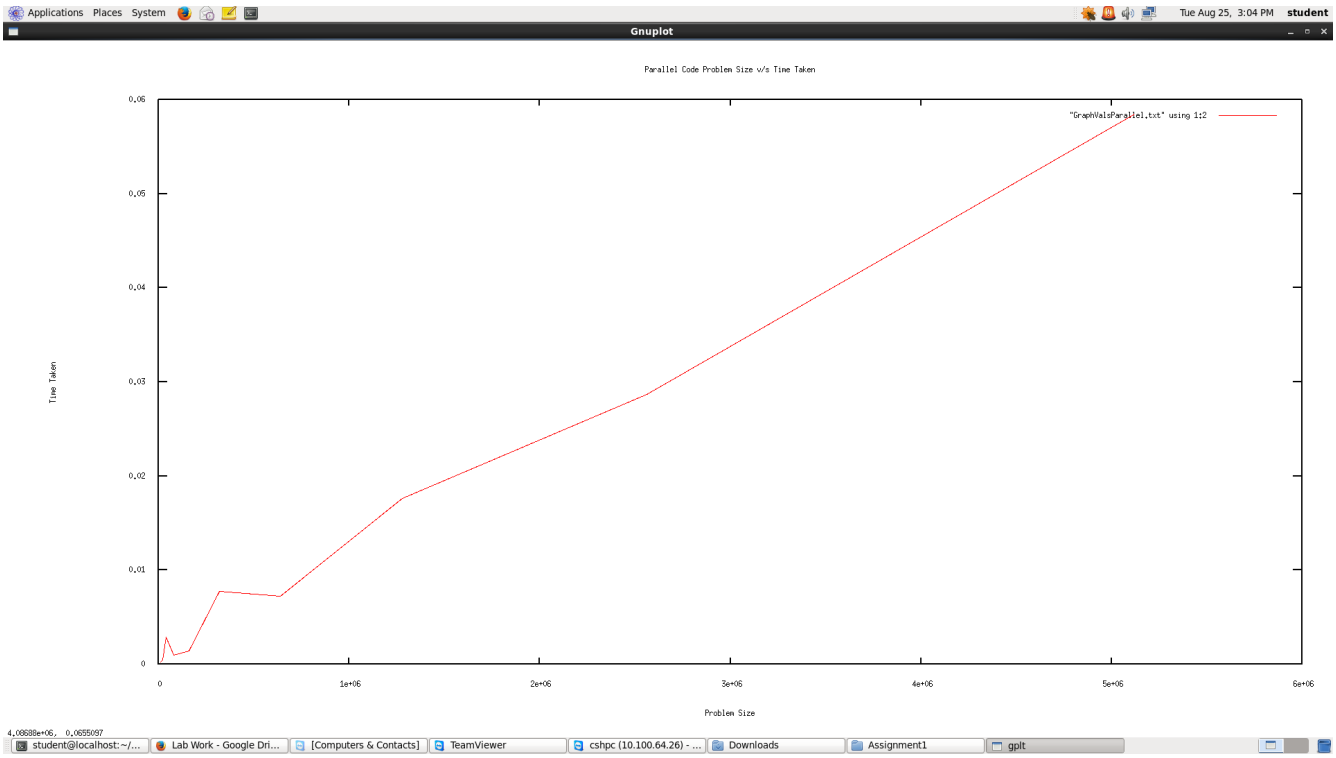
## OBSERVATIONS

Both serial and parallel codes gave almost linear speedup with respect to the problem size, with a few kinks, specially for small values of n.

Time was calculated using omp_get_wtime() library function include in the “omp.h” header file.




Serial Scaling

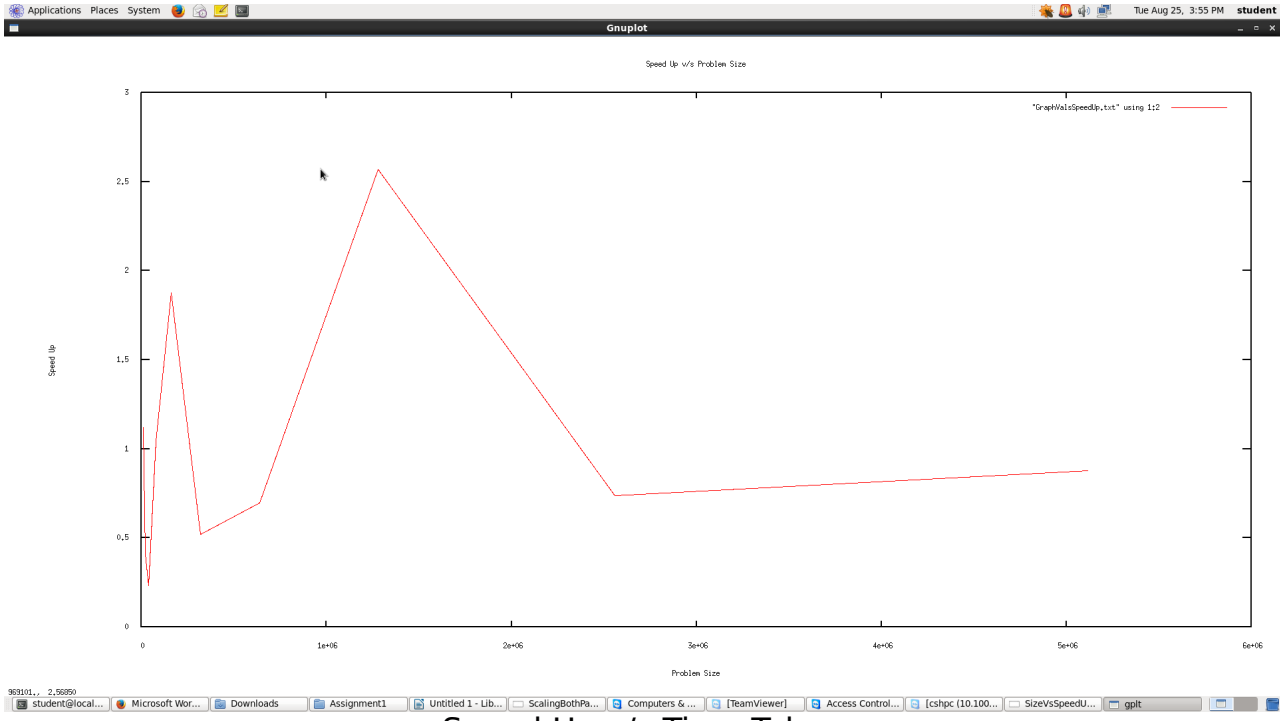



Parallel Scaling

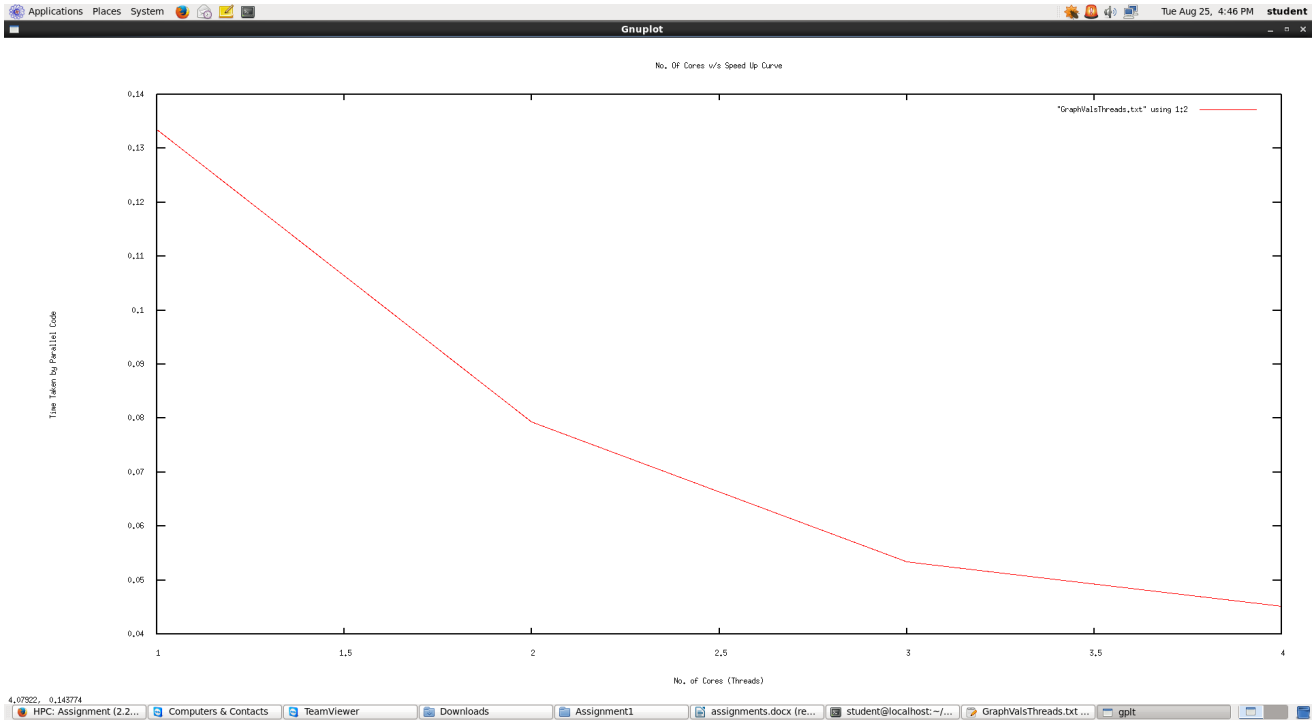
## SPEED UP

The maximum speed up achieved was ~2.5
Speed Up was calculated as:

Speed Up= (Time taken by serial code) / (Time taken by parallel code)




Speed Up v/s Time Taken




Number of Cores v/s Parallel Time Taken