HPC-2 Solution

Q1.

1. Row wise addition profiling with gprof:

Gprof results show the amount of time spend by main funtion is about 13.93 for the entire program to run. Each sample hit covers 2 byte(s) for 0.07% of 13.93 seconds.

1. Column wise addition profiling with gprof:

Gprof results show the amount of time spend by main funtion is about 50.98 for the entire program to run. Each sample hit covers 2 byte(s) for 0.02% of 50.98 seconds.

Since gprof does not give more information about cache misses, we use PAPI to observe how the loop variation can change the performance of the code.

2. The following are the PAPI results for row wise addtion and column wise addition of the two matrix. We can clearly see that there many more cache misses in the coumn wise addition than row wise addition. Since C++ is row oriented, row wise strides of data are loaded in the cache.

Row Additon: Papi

Size of matrix	L1 cache miss	L2 cache miss
100	2053	1419
1000	233556	75282
10000	5790453	4708884
15000	10130890	9860745
20000	13630715	16112804

Column Additon: Papi

Size of matrix	L1 cache miss	L2 cache miss
100	120354331	4331
1000	40049003734519	3734519
10000	592330343	566175374
15000	1381128042	1442371242
20000	2505827165	2572285821

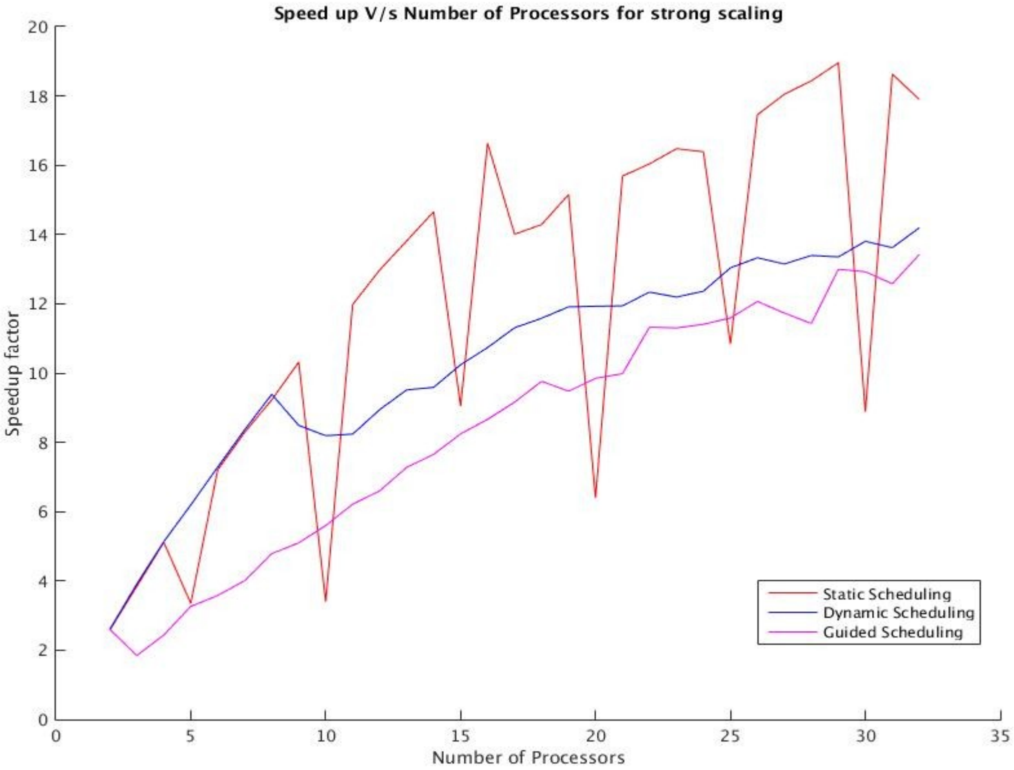
Q2.

1. The area of mandelbrot set is equal to 1.507.

2. Performance of the code: analysed by papi

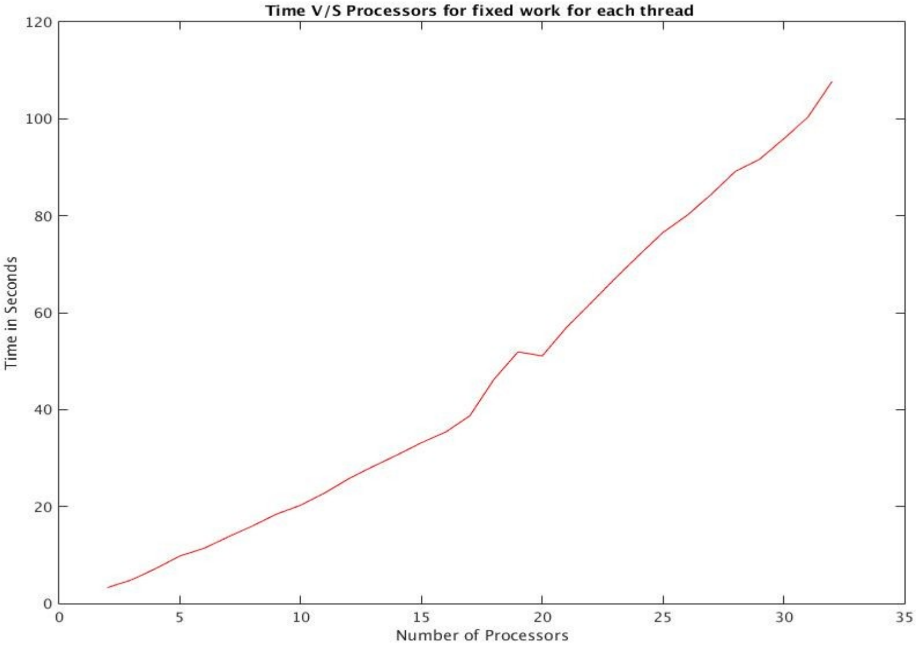
Size of matrix	L1 cache miss	L2 cache miss
100	328433	7782
500	9224905	732971
1000	33017192	1344884
1500	70018899	1798378

3.1 Study of strong scaling for different type scheduling. The speedup factor is calulcated as time required to calculate area of mandelbrot set for 500*500 grid using one processor divided by the time required to calculate area of mandelbrot set for 500*500 grid using n different number of threads.

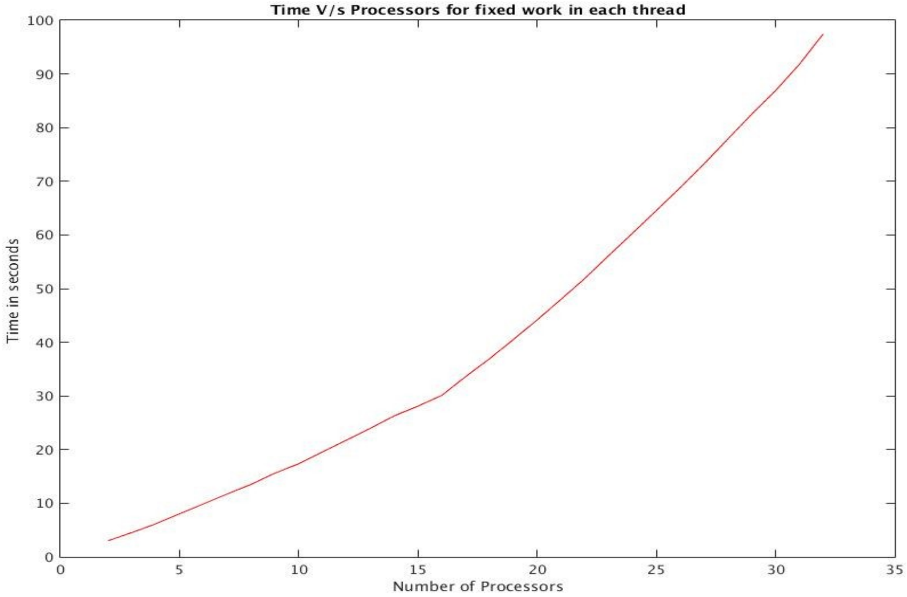


Q2. 3.2 Study of weak scaling for static scheduling:

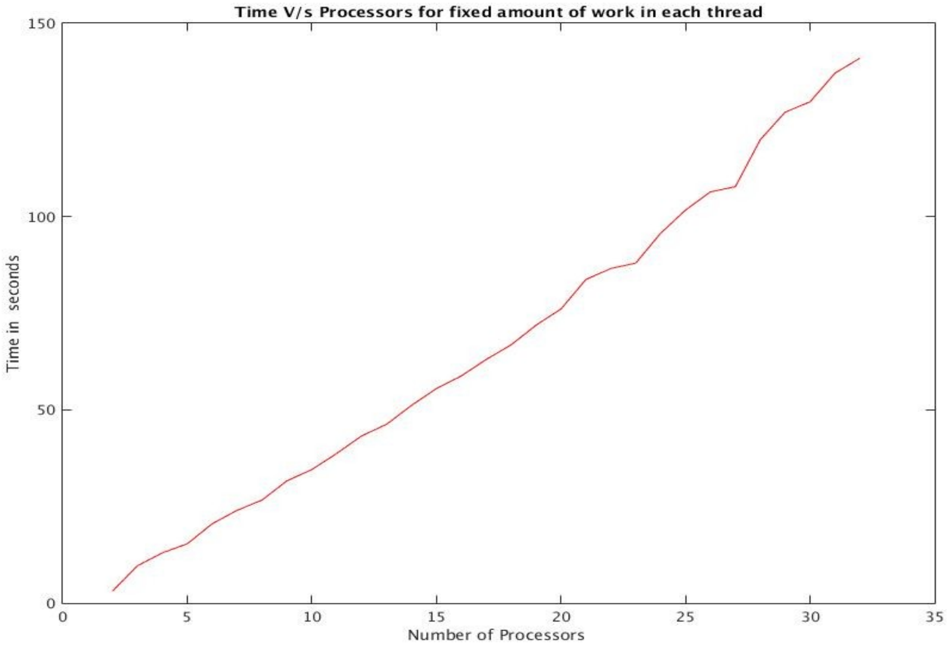
The weak scaling shows increase in time with number of processors. Since the problem is 2D, doubling the size of the nodes actually increases both points in x and y here real and imaginary axis. So the actual work per thread is four times the original. Hence we can see the general increase in time with increase in number of threads and size of the problem.



Q2. 3.2 Study of weak scaling for dynamic scheduling:

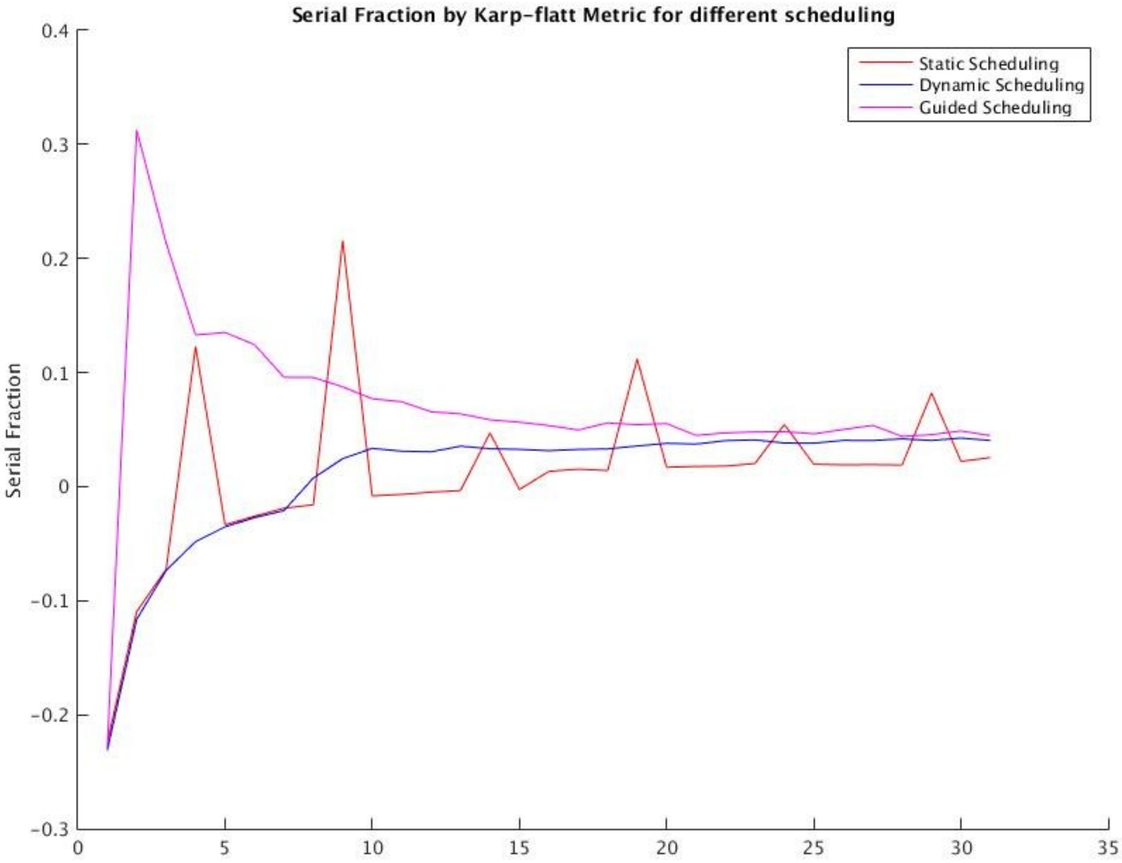


Q2. 3.2 Study of weak scaling for guided scheduling:



Q2. 4. Karp- Flatt Metric calculation

Karp- flatt metric is calculated by the following formula= (1/s – 1/p)/(1- 1/p). Where s is the speedup time and p is the number of processors. We have 3 different speedup for different scheduling scheme I.e static, dynamic and guided. Hence we have 3 different karp- flatt metric

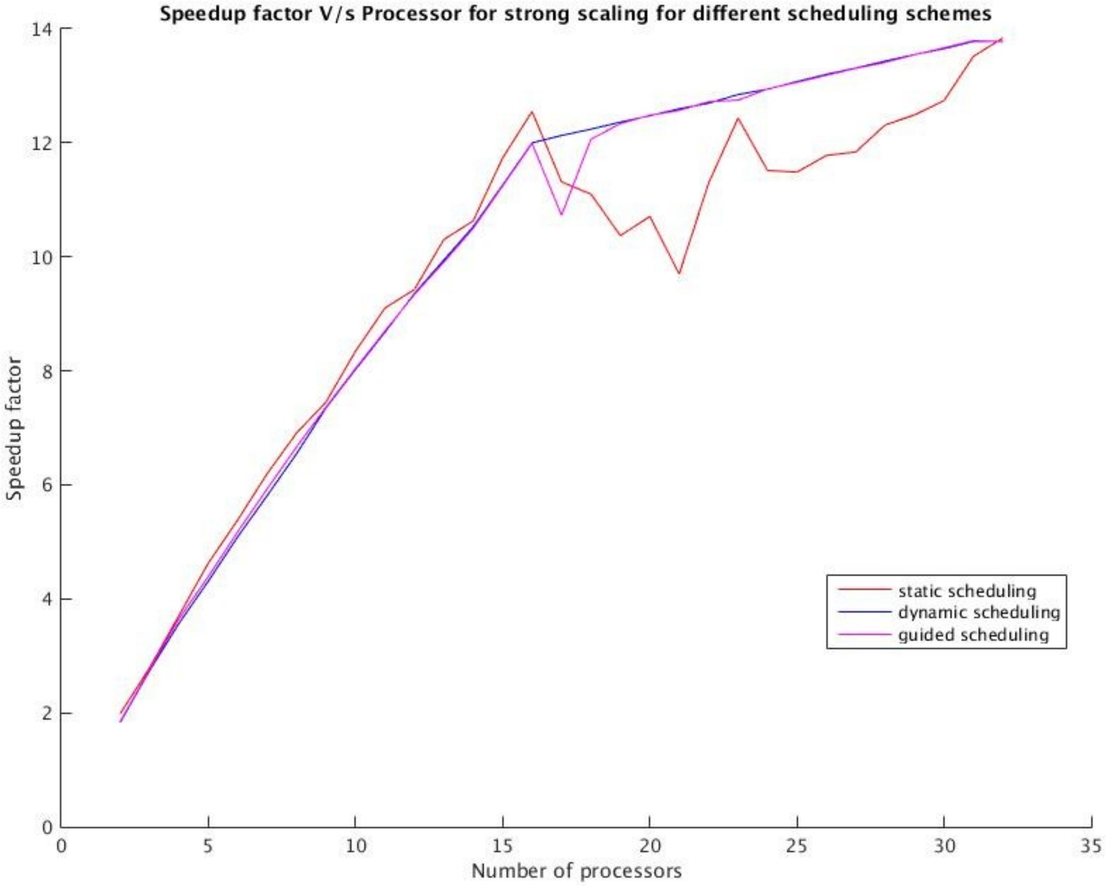


Q3.
1. The goldbach conjecture holds true. I have checked all the even numbers till 32000.

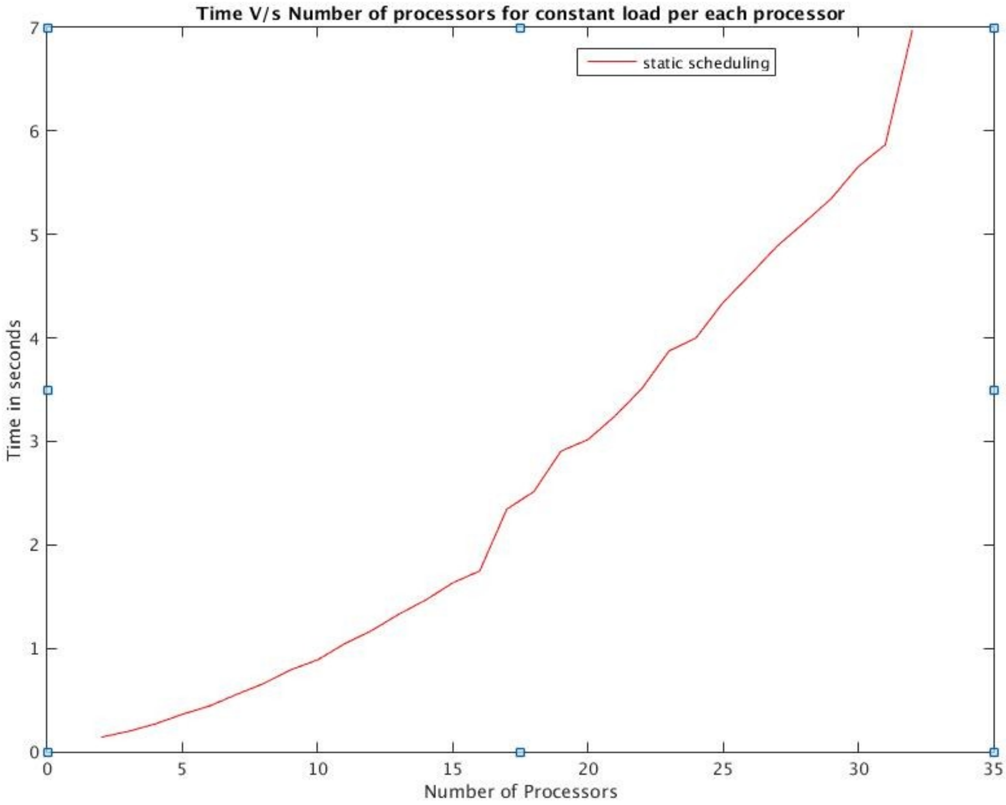
2. Performance of the code: analysed by papi

Number	L1 cache miss	L2 cache miss
100	256	79
10000	3190	1569
150000	5403	818
2000000	501985	50866

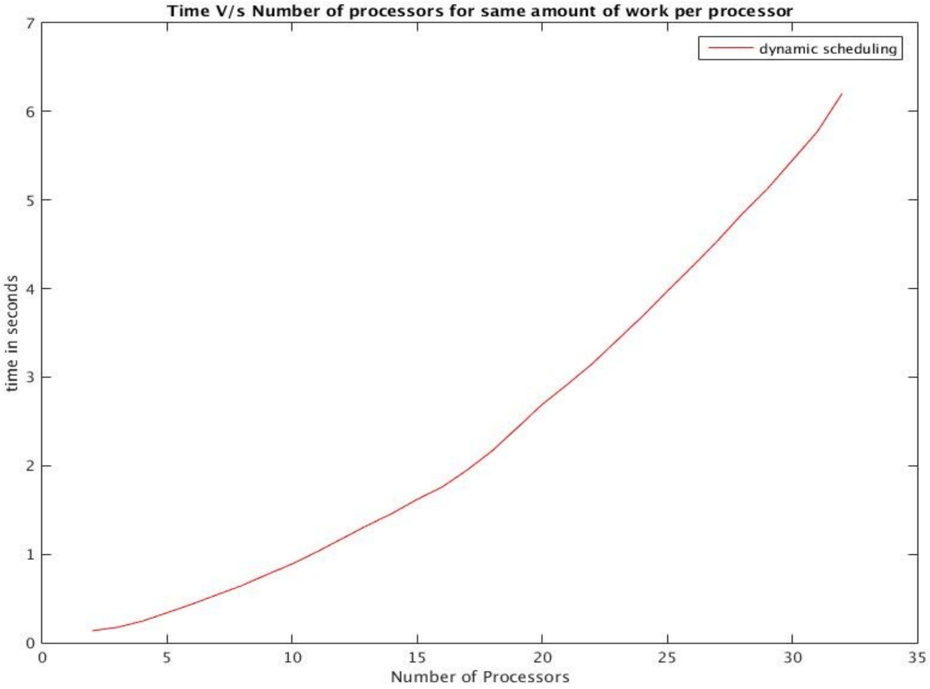
3.1 Study of strong scaling for different type scheduling. The speedup factor is calulcated as time required to check goldbach conjecture for 150000 using one processor divided by the time required by using n different number of threads.



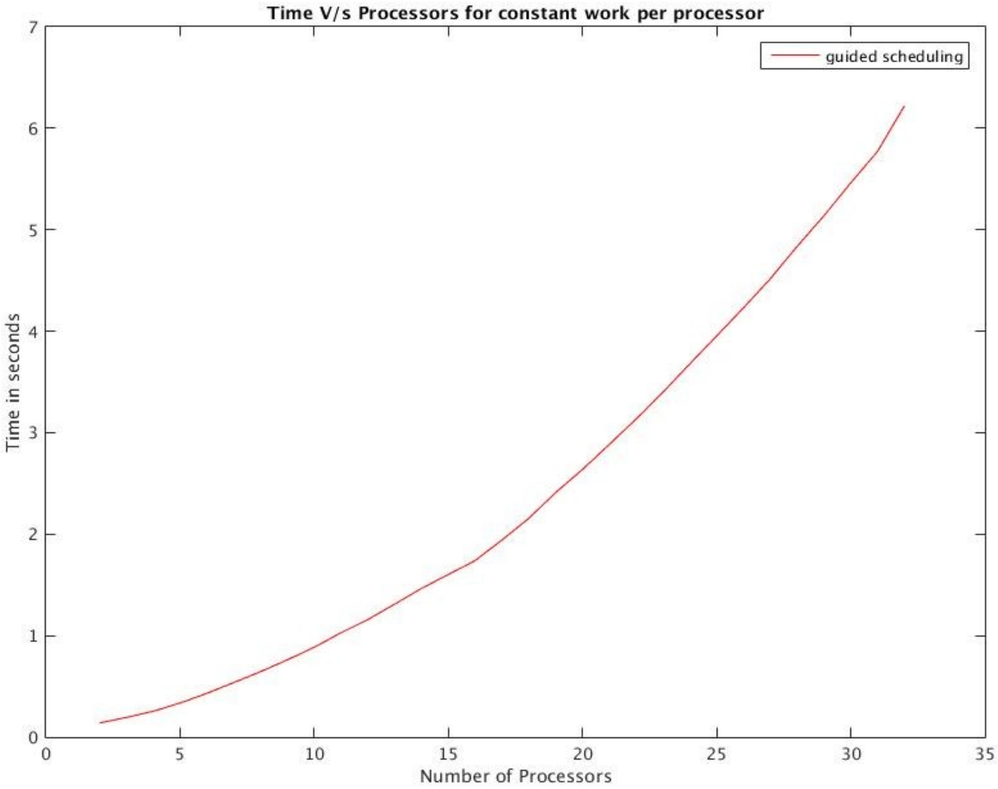
Q3. 3.2 Study of weak scaling for static scheduling:



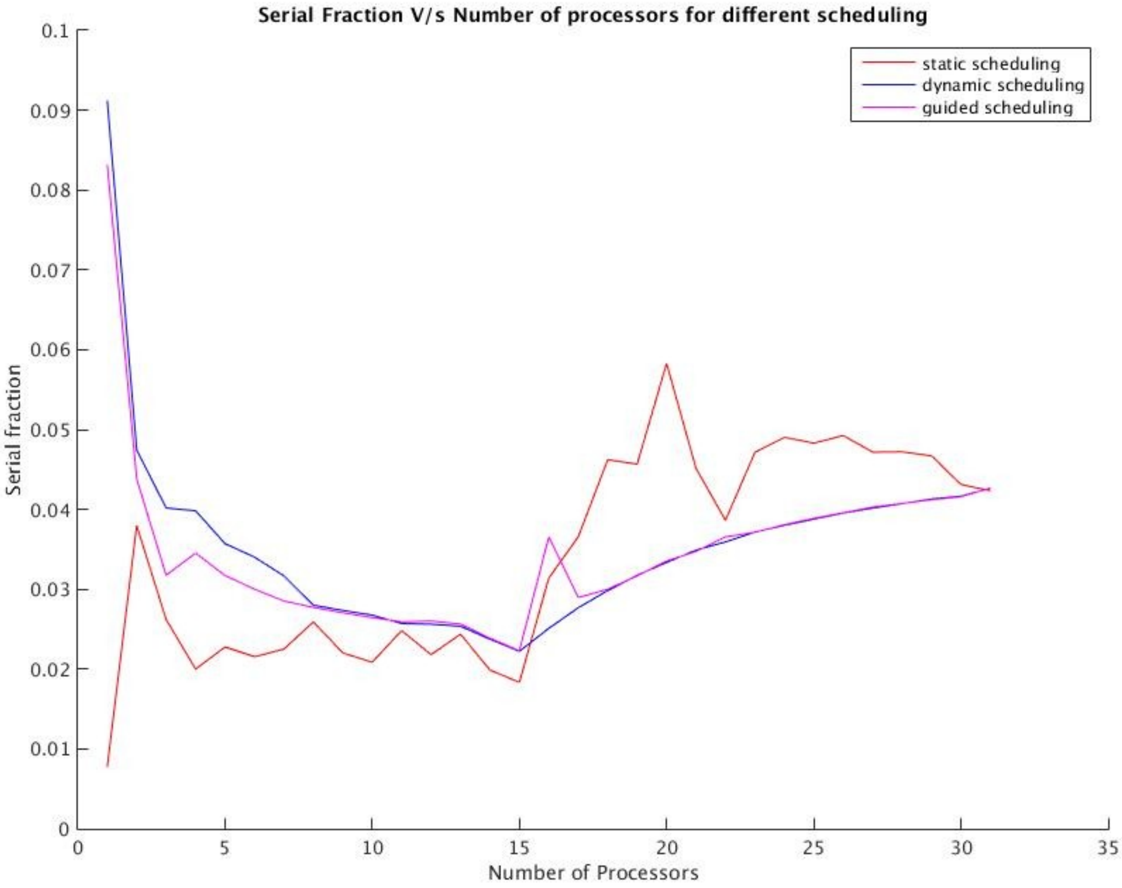
Q3. 3.2 Study of weak scaling for dynamic scheduling:



Q3. 3.2 Study of weak scaling for guided scheduling:



Q3. 4. Karp- Flatt Metric calculation



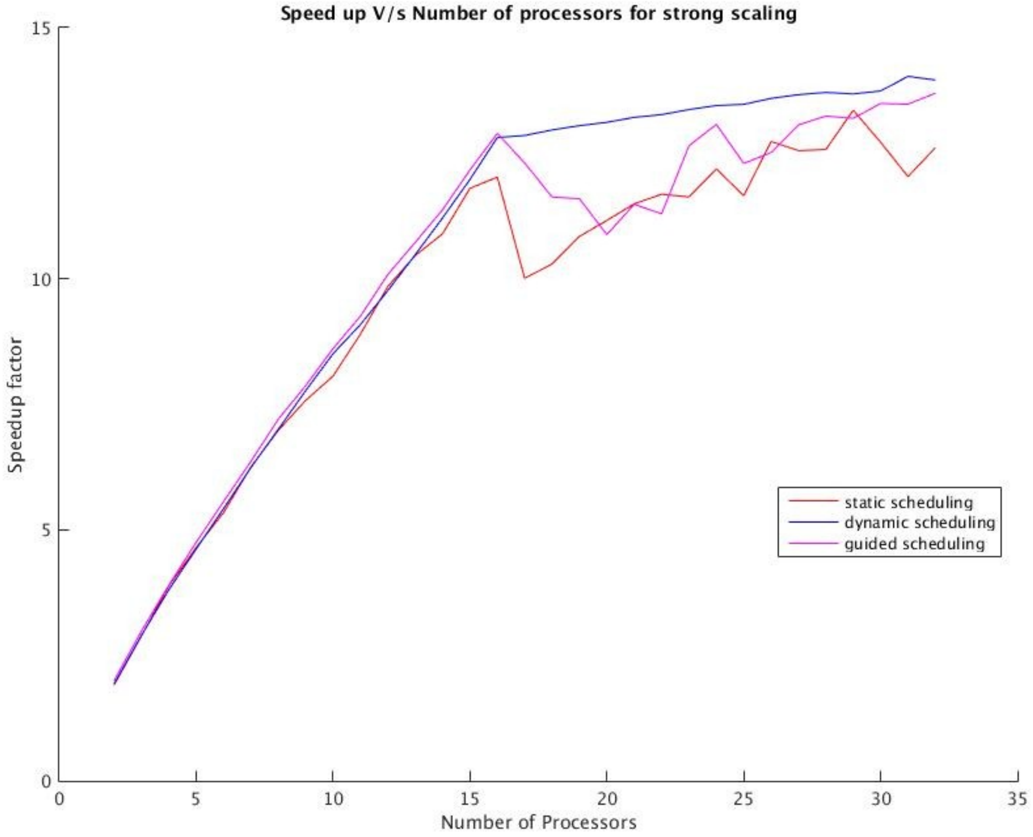
Q4

1. The estimated value of pi = 3.14159.

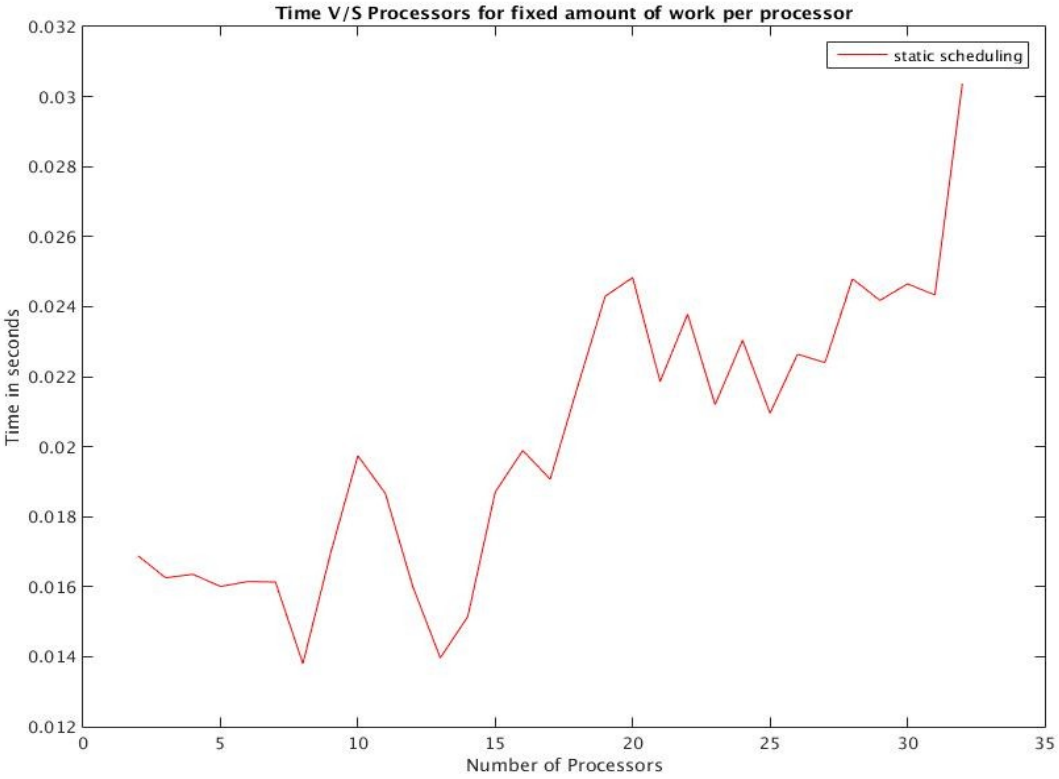
2. Performance of the code: analysed by papi

Number	L1 cache miss	L2 cache miss
1e9	115848	22607
2e9	119077	14452
5e9	193590	5608

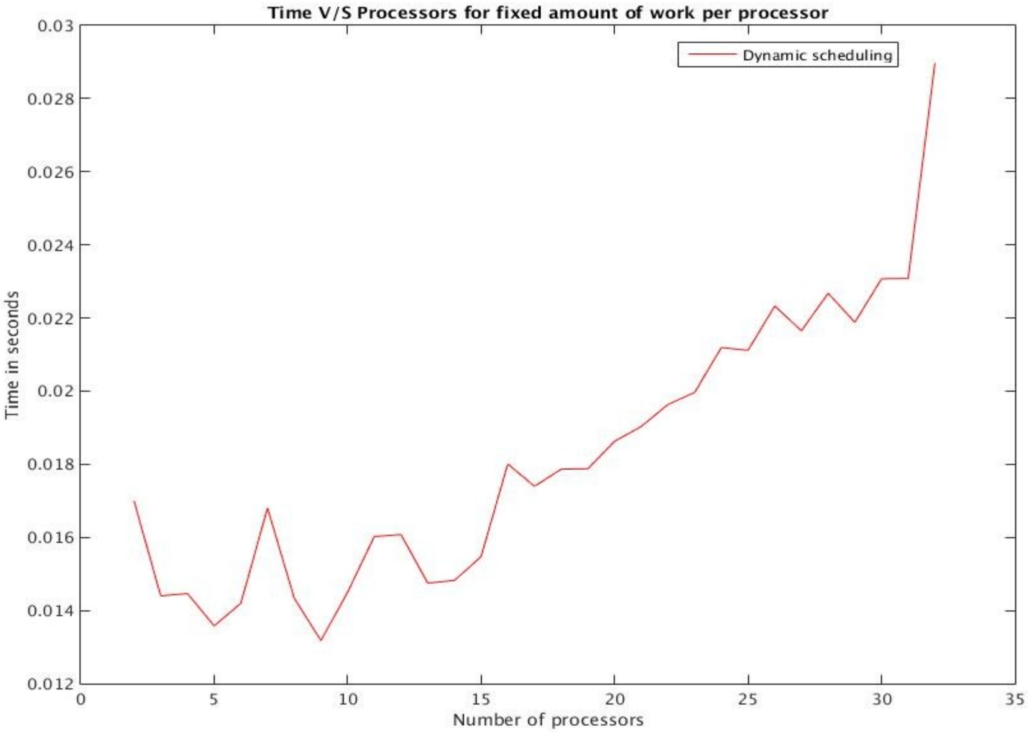
3. 3.1 Study of strong scaling for different type scheduling. The speedup factor is calulcated as time required to calculate value of pi for grid 5e9 using one processor divided by the time required by using n different number of threads.



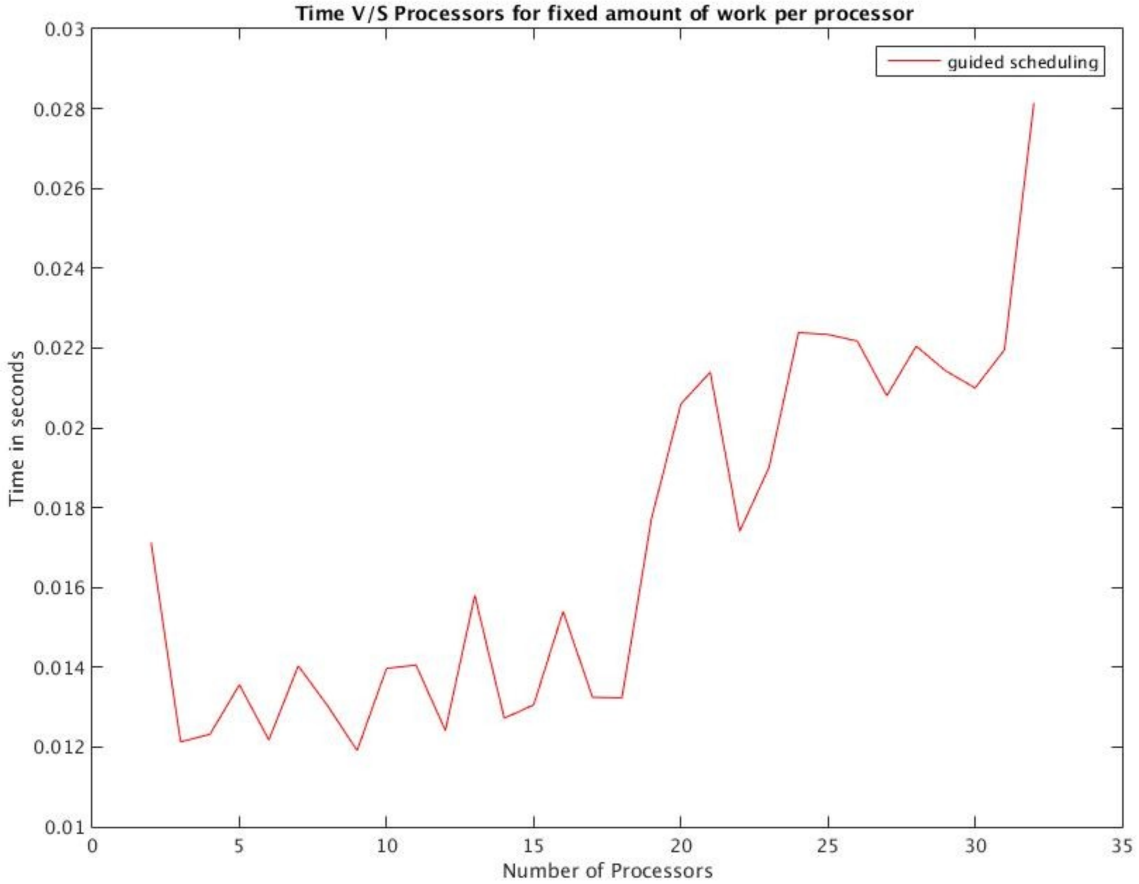
Q4. 3.2 Study of weak scaling for static scheduling:



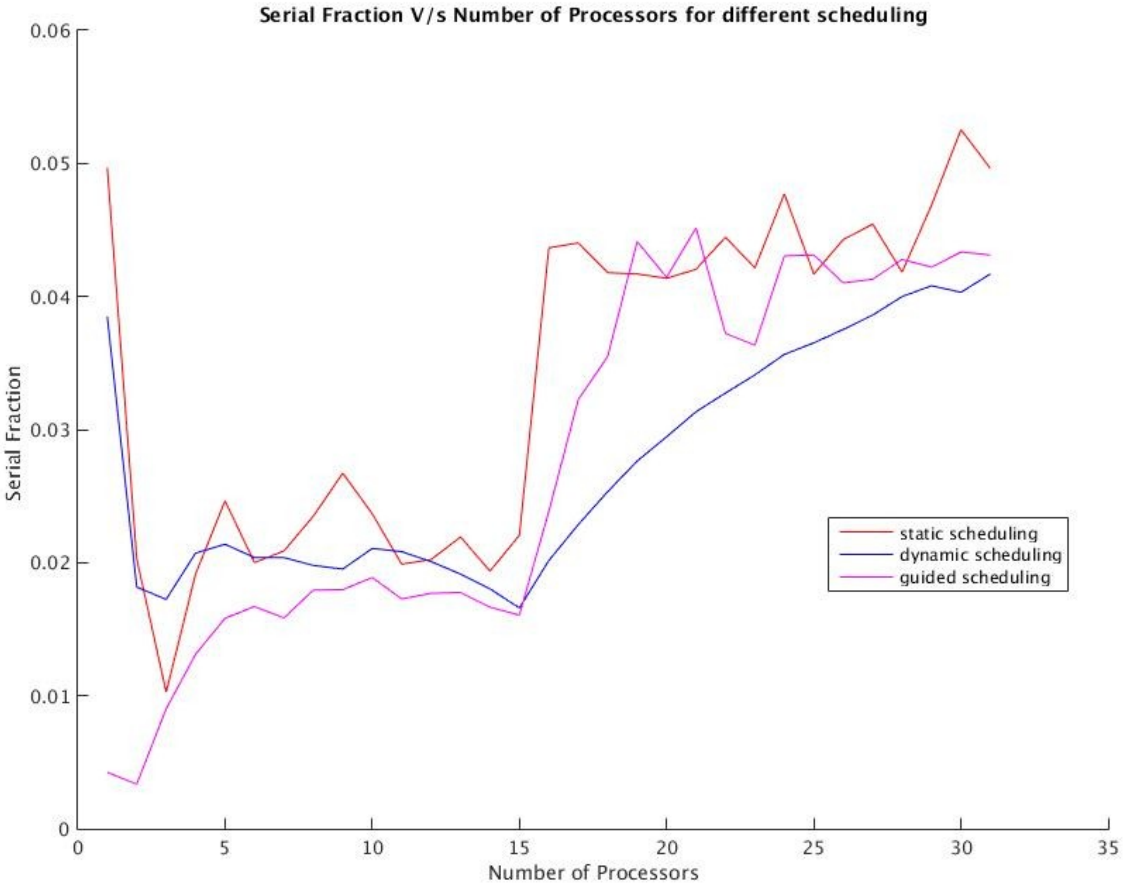
Q4. 3.2 Study of weak scaling for dynamic scheduling:



Q4. 3.2 Study of weak scaling for guided scheduling:



Q4. 4. Karp- Flatt Metric calculation


Serial Fraction V/s Number of Processors for different scheduling
Serial Fraction
Number of Processors
0
0.01
0.02
0.03
0.04
0.05
0.06
0
5
10
15
20
25
30
35
static scheduling
dynamic scheduling
guided scheduling