HPC Lab Week 2 Report 1

Name :	Prateek Agrawal
Roll Number :	CED18I040
Programming Environment :	OpenMP
Problem Statement :	Matrix Addition
Date :	19th August 2021

Systems Specifications :

CPU Name :	Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
Number of Sockets: :	1
Cores per Socket :	6
Threads per core :	2
L1 Cache size	192 Kb
L2 Cache size	1.5 MB
L3 Cache size(Shared):	9 MB
RAM	32 GB

Serial Code: (Run Time : 20.152976)

```
/*
* @Author: prateek
* @Date:   2021-08-19 17:20:58
* @Last Modified by:   prateek
* @Last Modified time: 2021-08-19 17:33:27
*/

#include <stdio.h>
#include <time.h>
#include <omp.h>

#define size 100

int main ()
{
	int i, j, p;

	double a[size][size] = {999.956};
	double b[size][size] = {4516.524};
	double c[size][size];
	double start, end;


		start = omp_get_wtime (); //addition starts here

			for (i = 0; i < size; i++)
			{
				for (j = 0; j < size; j++)
				{

					for(int k = 0;k<100000;k++)
					c[i][j] = a[i][j] + b[i][j];
				}
			}

		end = omp_get_wtime (); //addition ends here

		printf ("%f \n", end - start);

		return 0;
	}
```

Parallel Code :

```
/*
* @Author: prateek
* @Date:   2021-08-19 17:20:58
* @Last Modified by:   prateek
* @Last Modified time: 2021-08-19 17:29:16
*/

#include <stdio.h>
#include <time.h>
#include <omp.h>

#define size 100

int main ()
{
	int i, j, p;

	double a[size][size] = {999.956};
	double b[size][size] = {4516.524};
	double c[size][size];
	double start, end;

	int threads[] = {1,2,4,6,8,10,12,16,32,64,128,150};
	int th_index = sizeof(threads) / sizeof(threads[0]);

	for (p = 0; p < th_index; p++)
	{
		//omp_set_num_threads (threads [p]);
		start = omp_get_wtime (); //addition starts here

		#pragma omp parallel for num_threads(threads[p]) shared (a, b) private (i, j)
			for (i = 0; i < size; i++)
			{
				for (j = 0; j < size; j++)
				{

					for(int k = 0;k<100000;k++)
					c[i][j] = a[i][j] + b[i][j];
				}
			}

		end = omp_get_wtime (); //addition ends here
```

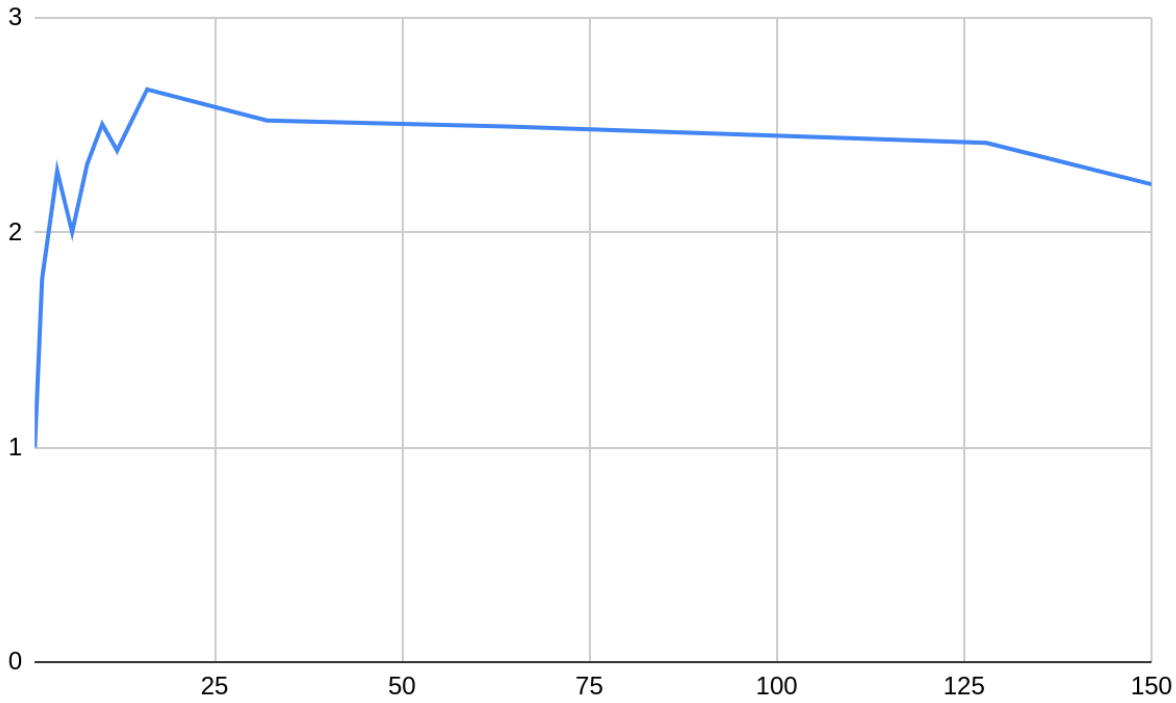
```
                printf ("%d\t%lf\n", threads[p], end - start);
        }

        return 0;
}
```

Compilation and Execution:

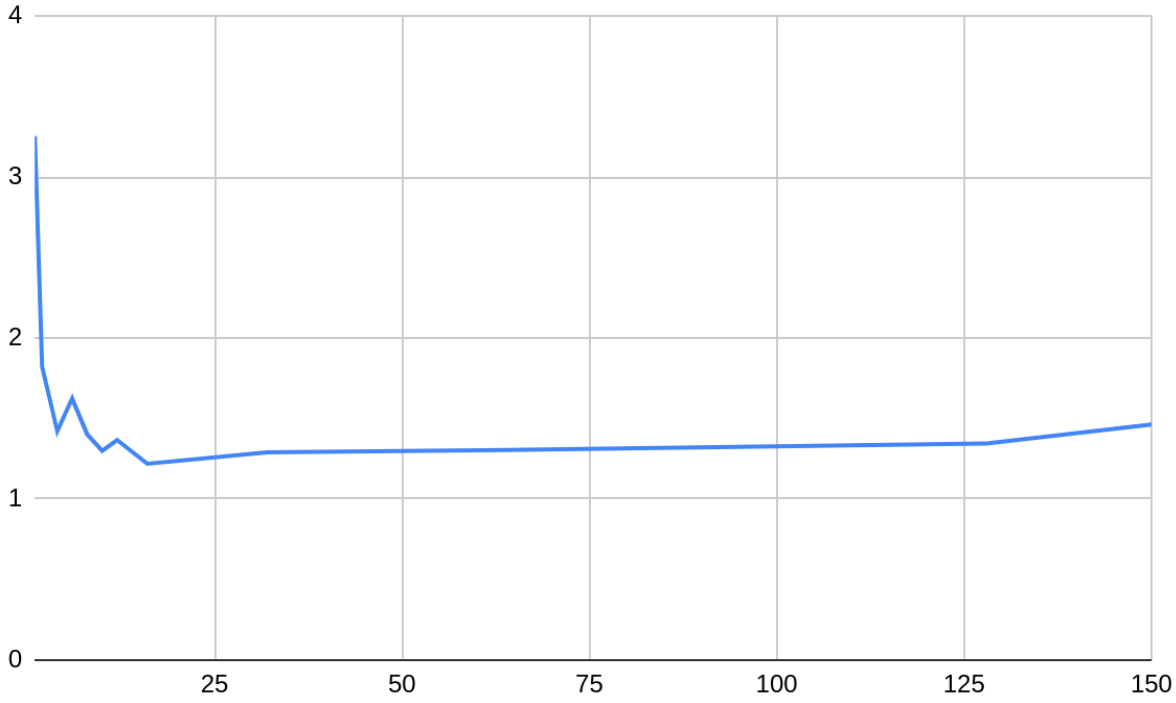
For enabling OpenMP environment use -fopenmp flag while compiling using gcc.

```
gcc -fopenmp vectoradd.c -o vectoradd
```

NUM Threads	Execution Time	Speed-Up	Parallelization Fraction
1	3.253682	1	0
2	1.821909	1.785864168	88.00939981
4	1.421637	2.288686915	75.07576545
6	1.624522	2.002854994	60.08552772
8	1.402618	2.319720694	65.01869925
10	1.299224	2.504327198	66.74346172
12	1.365607	2.382590306	63.30422524
16	1.219748	2.667503452	66.6792145
32	1.289826	2.522574363	62.30498843
64	1.304175	2.494820097	60.86801216
128	1.345546	2.418112796	59.10721124
150	1.462061	2.225407832	55.43397609



NUMBER OF THREADS vs SPEED-UP



NUMBER OF THREADS vs Execution Time

Inference: (Note: Execution time, graph and inference will be based on hardware configuration)

• At thread count 16 maximum speedup is observed.
• If thread count is more than 16 then the execution time increases/decreases slightly and tapers out.