ME 766 : HW1 Submission

Name: Harsh Chandak
Roll No. 150040016
Mechanical Department, IITB

- Integration of f(x) = sin(x) in the interval [0, π] using various methods
- The value of the integral obtained from numerical method (regular integration) = 2
- π value used = 3.14159

1.(a) Trapezoidal Rule - Serial Integration code in the file hw1_serial_trapezoidal.c

OpenMP parallel code in the file hw1_trapezoidal.c

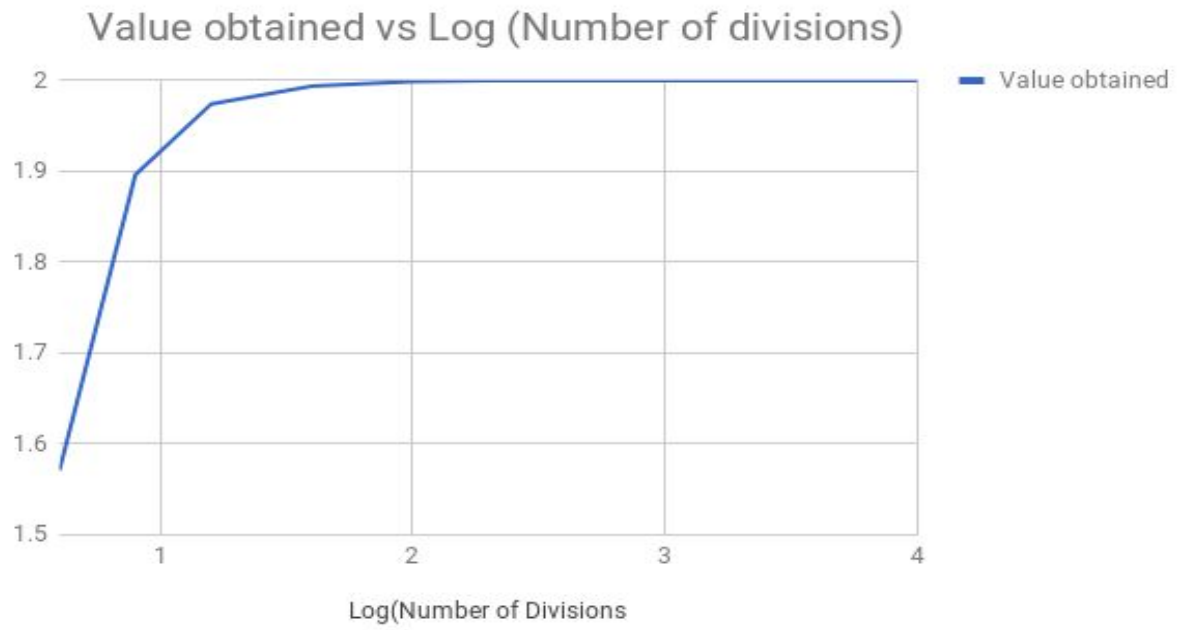
1.(b) Montecarlo Method - Integration code in the file hw1_serial_montecarlo.c

OpenMP parallel code in the file hw1_omp_montecarlo.c

2.(a) Convergence study of Trapezoidal method using different numbers of divisions

Number of Divisions	Log (no. of divisions)	Value obtained	Actual Value	Percentage error
2	0.3010299957	1.570797	2	21.46015
4	0.6020599913	1.896119	2	5.19405
8	0.903089987	1.974232	2	1.2884
16	1.204119983	1.99357	2	0.3215
40	1.602059991	1.998972	2	0.0514
100	2	1.999836	2	0.0082
200	2.301029996	1.999836	2	0.0082
500	2.698970004	1.999993	2	0.00035

1000	3	2.000000	2	0
5000	3.698970004	2.000000	2	0
10000	4	2.000000	2	0

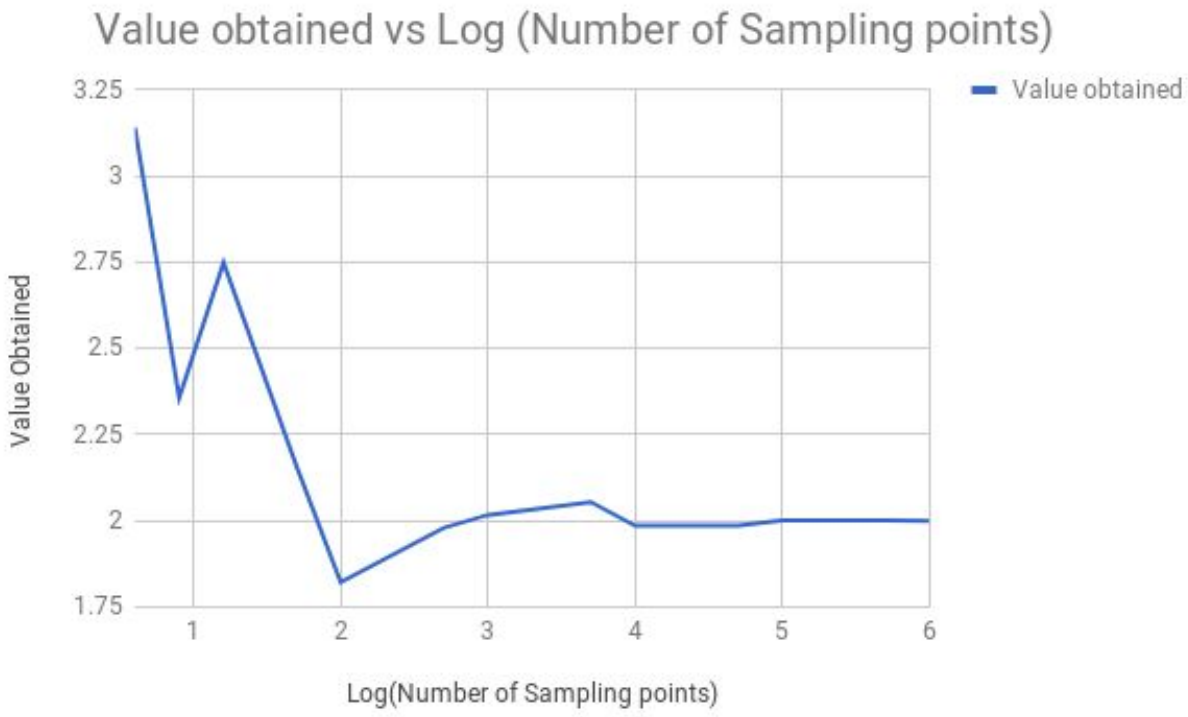


The values obtained in case of Trapezoidal serial code tend to 2 as we increase the number of divisions. All the values obtained are less than or equal to 2.

2.(b) Convergence study of Montecarlo method using different numbers of sampling points

Number of Sampling Points	Log (Number of Sampling Points)	Value obtained	Actual Value	Percentage error
2	0.3010299957	3.14159	2	-57.0795
4	0.6020599913	2.356192	2	-17.8096
8	0.903089987	2.748891	2	-37.44455
16	1.204119983	2.159843	2	-7.99215

50	1.698970004	1.822122	2	8.8939
100	2	1.979202	2	1.0399
500	2.698970004	2.016901	2	-0.84505
1000	3	2.0546	2	-2.73
5000	3.698970004	1.985485	2	0.72575
10000	4	1.985799	2	0.71005
50000	4.698970004	2.001067	2	-0.05335
100000	5	2.000439		-0.02195
500000	5.698970004	1.999327		0.03365
1000000	6	2.001001		-0.05005

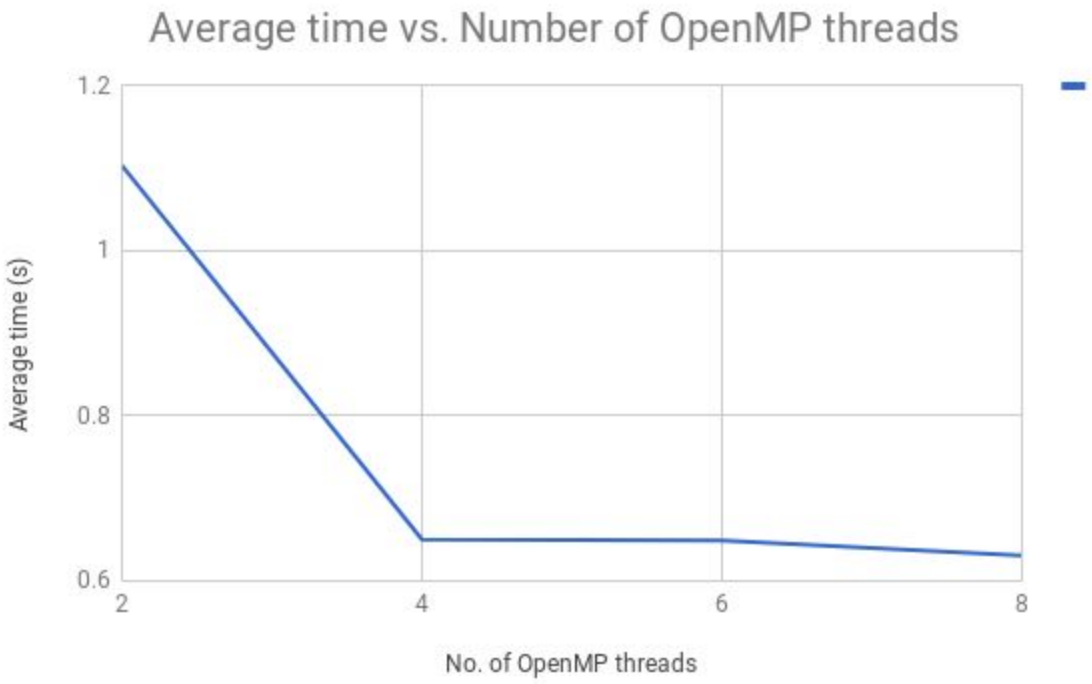


The value obtained varies around 2 and finally converges to 2 for large number of sampling points. There is a large fluctuation in the value obtained for small

number of sampling points but for large values it approximates the value obtained very well.

3.(a) Timing study using 2,4,6 and 8 OpenMP threads for Trapezoidal Method

No. of divisons	OpenMP threads	T1 (s)	T2 (s)	T3 (s)	T4 (s)	T5 (s)	Average time (s)
10000000	2	1.102539	1.102539	1.101562	1.106445	1.103516	1.1033202
10000000	4	0.645508	0.651367	0.649414	0.651367	0.65332	0.6501952
10000000	6	0.648438	0.674805	0.654297	0.638672	0.628906	0.6490236
10000000	8	0.635742	0.639648	0.62793	0.623047	0.626953	0.630664

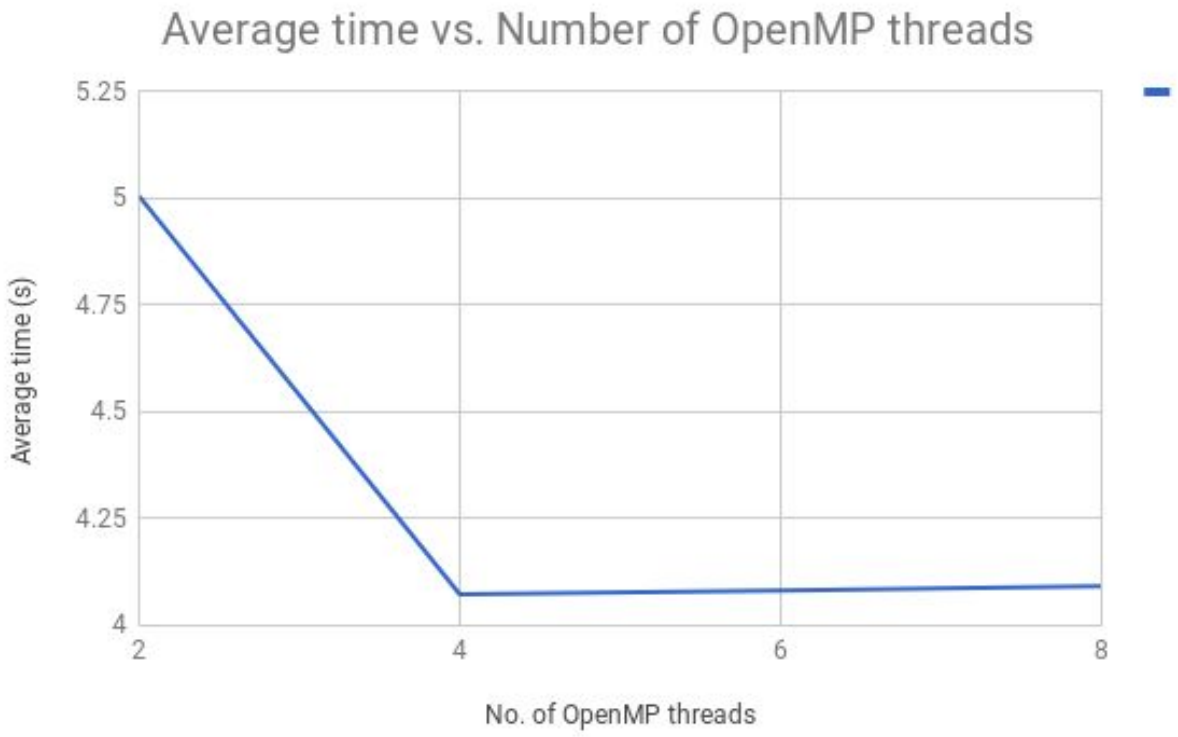


Time taken for 10000000 number of divisions in Trapezoidal serial code is **2.002821 seconds** which is larger than the time taken for trapezoidal parallel code (for all 2,4,6 & 8 OpenMP nodes)

Here, using OpenMP or parallelising reduces the time required (which should happen in an ideal case if we use parallel code instead of serial code).

Here, we get minimum average time for 8 OpenMP threads (for 10000000 no. of divisions) as compared to 2,4 & 6 OpenMP threads.

No. of sampling points	OpenMP threads	T1 (s)	T2 (s)	T3 (s)	T4 (s)	T5 (s)	Average time (s)
10000000	2	6.169922	5.623047	4.246094	3.298828	5.6875	5.0050782
10000000	4	4.078125	4.097656	4.048828	4.068359	4.072266	4.0730468
10000000	6	4.082031	4.029297	4.101562	4.091797	4.103516	4.0816406
10000000	8	4.080078	4.09375	4.105469	4.087891	4.091797	4.091797



Time taken by for 10000000 number of divisions in Montecarlo serial code is **2.000192 seconds.** In this case serial Montecarlo code is taking less time as compared to parallel montecarlo code. This is different than the general scenario in which parallel code takes less time than the serial code.

4 OpenMP threads takes the minimum time in parallel Montecarlo code for 10000000 sampling points as compared to 2,6 & 8 OpenMP threads.