Walchand College of Engineering, Sangli Department of Computer Science and Engineering

Class: Final Year (Computer Science and Engineering)

Year: 2021-22 **Semester:** 1

Course: High Performance Computing Lab

Practical No. 3

Exam Seat No:

1. Exam Seat Number - 2018BTECS00100

Problem Statement 1:

Screenshot #:

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS1$ gcc -fopenmp ps1.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS1$ ./a.out
Enter the number of iterations used to estimate pi: 50
# of trials= 50 , estimate of pi is 2.64
```

Information #:

```
#include <math.h>
#include <string.h>
#include<omp.h>
#define SEED 35791246

int main()
{

int niter=0;
double x,y;
int i,count=0; /* # of points in the 1st quadrant of unit circle */
double z;
double pi;
printf("Enter the number of iterations used to estimate pi: ");
scanf("%d",&niter);

srand(SEED);
count=0;

#pragma omp parallel for   shared(niter)  num_threads(4)
for (i=0; i<niter; i++) {
x = (double)rand()/RAND_MAX;
y = (double)rand()/RAND_MAX;
z = x*x+y*y;

if (z<=1) count++;
}
pi=(double)count/niter*4;
printf("# of trials= %d , estimate of pi is %g \n",niter,pi);
```

Problem Statement 2:

Screenshot #:

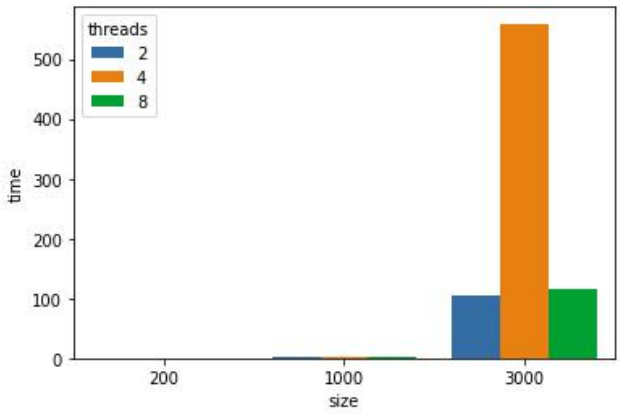
```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
200

Executed when size = 200 and threads =2
Done in 0.045931 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fopenmp ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
200

Executed when size = 200 and threads =4
Done in 0.035858 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fopenmp ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
200

Executed when size = 200 and threads =8
Done in 0.034517 seconds
```

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fopenm
p ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
1000

Executed when size = 1000 and threads =2
Done in 2.324062 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fopenm
p ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
1000

Executed when size = 1000 and threads =4
Done in 3.504843 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fopenm
p ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
1000

Executed when size = 1000 and threads =8
Done in 3.076934 seconds
```

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fope
nmp ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
3000

Executed when size = 3000 and threads =2
Done in 104.905248 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fope
nmp ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
3000

Executed when size = 3000 and threads =4
Done in 559.008342 seconds
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ g++ -fope
nmp ps2.cpp
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS2$ ./a.out
Size of matrix
3000

Executed when size = 3000 and threads =8
Done in 115.919575 seconds
```

Information #:

```
<AxesSubplot:xlabel='size', ylabel='time'>
```



Problem Statement 3:
1) With static scheduling:

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp static.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=100 and execution time=0.010081
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp static.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and execution time=0.001204
```

Significant improvment in time as the chunk size in increased.

2)With dynamic scheduling:

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp dynamic.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and execution time=0.000722
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp dynamic.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=100 and execution time=0.002822
```

3)With nowait

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp nowait.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and 0.001259 execution time
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp nowait.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=100 and 0.002083 execution time
```

Compairing all three:

```
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp static.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and execution time=0.006866
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp dynamic.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and execution time=0.009883
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ gcc -fopenmp nowait.c
prax@prakx-ideapad:~/Desktop/HPC/Practical3/PS3$ ./a.out

 Using 4 no of threads with chunk size=500 and 0.005672 execution time
```

Conlusion: Significant improvment in time with nowait.

Github Link: