NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL
DEPARTMENT OF INFORMATION TECHNOLOGY

IT 301 Parallel Computing LAB 5

31st August 2021

Faculty: Dr. Geetha V

Execute following programs and put screen shots of the output. Write analysis of the result before uploading in IRIS as a single pdf file. For programming exercises, write the code and also put screenshot of the results.

Total Marks =2+8=10 marks

1. How to compare sequential and parallel program execution times. ?

Include following header files in the program.

```
#include <sys/time.h>

#include <stdlib.h>
```

//Declare following variables.

```
struct timeval TimeValue_Start;

struct timezone TimeZone_Start;

struct timeval TimeValue_Final;

struct timezone TimeZone_Final;

long time_start, time_end;

double time_overhead;
```

Just before starting parallel region code , note down the time(start time)

```
gettimeofday(&TimeValue_Start, &TimeZone_Start);
```

After finishing parallel region, get end time.

```
gettimeofday(&TimeValue_Final, &TimeZone_Final);
```

Calculate the overhead time as follows:

```
time_start = TimeValue_Start.tv_sec * 1000000 + TimeValue_Start.tv_usec;

time_end = TimeValue_Final.tv_sec * 1000000 + TimeValue_Final.tv_usec;

time_overhead = (time_end - time_start)/1000000.0;

printf("\n\n\t\t Time in Seconds (T) : %lf",time_overhead);
```

Example: Execute the program and compare sequential and parallel executions. [sequential result : 1Mark +Parallel Execution 1 Mark=2 Marks]

```
#include <stdio.h>
#include <sys/time.h>
#include <omp.h>
#include <stdlib.h>
int main(void){
struct timeval TimeValue_Start;
struct timezone TimeZone_Start;
struct timeval TimeValue_Final;
struct timezone TimeZone_Final;
long time_start, time_end;
double time_overhead;double pi,x;
int i,N;
pi=0.0;
N=1000;
gettimeofday(&TimeValue_Start, &TimeZone_Start);
#pragma omp parallel for private(x) reduction(+:pi)
for(i=0;i<=N;i++){
x=(double)i/N;
pi+=4/(1+x*x);
}
gettimeofday(&TimeValue_Final, &TimeZone_Final);
time_start = TimeValue_Start.tv_sec * 1000000 + TimeValue_Start.tv_usec;
time_end = TimeValue_Final.tv_sec * 1000000 + TimeValue_Final.tv_usec;
time_overhead = (time_end - time_start)/1000000.0;
printf("\n\n\tTime in Seconds (T) : %lf\n",time_overhead);
pi=pi/N;
printf("\n \tPi is %f\n\n",pi);
}
```

---

2. Write a sequential program to add elements of two arrays (c[i]=a[i]*b[i]. Convert the same program for parallel execution. Initialize array with random numbers. Consider an array size as 10k, 50k and 100k. Analyse the result for maximum number of threads and various schedule()

function. Based on observation, perform analysis of the total execution time and explain the result by plotting the graph. [increase array size until parallel execution time is less than sequential execution.]

Schedule()	Total Execution time for number of iterations 5K	Total execution for number of iterations 10K	Total execution for number of iterations 50K	Total execution for number of iterations 100K
Sequential execution				
static				
Static, chunksize				
Dynamic, chunksize				
Guided				
runtime				

Plot the graph using any software and write your observation.

Results in each Row in the above table=1 mark=> 6 * 1= 6 Marks

Plotting graph=1 Mark

Observation =1 Marks

Total = 8 Marks