Parallel Computing Laboratory
IT-300
Fall-2019

By
Dr. B. Neelima
National Institute of Technology Karnataka
(NITK), Surathkal

Week 01: 7^{th} August-2019

This week the students will exercise the basic constructs and OpenMP variables studies. The students are free to choose their programming environment. But the preferred language is C, based on which the following assignments' guidance is given:

The file is saved as regular 'C' file name: filename.c

Compilation using GCC: **gcc -fopenmp filename.c**

You are free to use your own compiler and get the instruction for the compilation. It is also shared in OpenMP lecture notes.

All the required OpenMP syntaxes are available in OpenMP-API-Specification-5.0.pdf. This file is shared with you already and is openly available for reference.

Exercise 1: Introduction to OpenMP and Creating Threads

1. Verify working of your environment through a program that prints "Hello World"
2. Verify working of your OpenMP environment through a multithreaded program that prints "Hello World"
3. Write a multithreaded program where each thread prints "Hello World". (Hint: omp_get_thread_num())
4. Test the usage of clause num_threads(n), omp_set_num_threads(n)

Hint: Refer Lecture Notes

Exercise 2: Numerical Integration

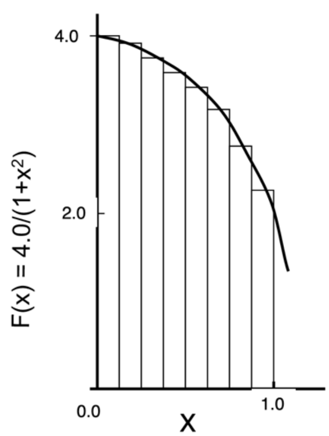
1. Write a serial PI program.
2. Create a parallel version of the pi program using a parallel construct
   # pragma omp parallel
   - Pay close attention to shared versus private variables
   - In addition to a parallel construct, you will need the runtime library routines–

- Int omp_get_num_threads();
- Int omp_get_thread_num();
- double omp_get_wtime();
- omp_set_num_threads();

3. Use synchronization constructs critical and atomic to avoid any conflicts in computation
4. Write PI program computation using a loop and reduction in OpenMP
5. Measure serial, parallel, critical, atomic and reduction execution time and report your observations along with values.

Hint:

Numerical integration



Mathematically, we know that:

$$\int_0^1 \frac{4.0}{(1+x^2)}\, dx = \pi$$

We can approximate the integral as a sum of rectangles:

$$\sum_{i=0}^{N} F(x_i)\Delta x \approx \pi$$

Where each rectangle has width Δx and height $F(x_i)$ at the middle of interval i.

Serial PI-program:

```
static long num_steps = 100000;
double step;
int main ()
{           int i;      double x, pi, sum = 0.0;

            step = 1.0/(double) num_steps;

            for (i=0;i< num_steps; i++){
                        x = (i+0.5)*step;
                        sum = sum + 4.0/(1.0+x*x);
            }
            pi = step * sum;
}
```