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

Class: *Final Year (Computer Science and Engineering)*

Year: *2024-25* **Semester:** *1*

Course: *High Performance Computing Lab*

Practical No. 2

Exam Seat No:

Title of practical: Study and implementation of basic OpenMP clauses

Implement following Programs using OpenMP with C:

1. *Vector Scalar Addition*
2. *Calculation of value of Pi*
   Analyse the performance of your programs for different number of threads and Data size.

Problem Statement 1:
Screenshots:

Code :

```
#include <iostream>
#include <omp.h>
#include <vector>
#include <cstdlib>
#include <ctime>

void vectorScalarAdditionSequential(const std::vector<double>& vec, double scalar,
std::vector<double>& result) {
for (size_t i = 0; i < vec.size(); ++i) {
result[i] = vec[i] + scalar;
}
}

void vectorScalarAdditionParallel(const std::vector<double>& vec, double scalar,
std::vector<double>& result) {
#pragma omp parallel for
for (size_t i = 0; i < vec.size(); ++i) {
result[i] = vec[i] + scalar;
}
}
```

```
int main() {
size_t n = 1000000; // Size of vector
std::vector<double> vec(n, 1.0); // Vector with all elements initialized to 1.0
double scalar = 2.0;
std::vector<double> result(n);

// Sequential execution
double start_time = clock();
vectorScalarAdditionSequential(vec, scalar, result);
double end_time = clock();
std::cout << "Sequential Time taken for vector scalar addition: " << (end_time - start_time) /
CLOCKS_PER_SEC << " seconds" << std::endl;

// Parallel execution
start_time = omp_get_wtime();
vectorScalarAdditionParallel(vec, scalar, result);
end_time = omp_get_wtime();
std::cout << "Parallel Time taken for vector scalar addition: " << (end_time - start_time) << "
seconds" << std::endl;

return 0;
}
```

Output:

```
ubuntu@ubuntu-VirtualBox:~/Documents/Assignment02$ g++ -fopenmp -o a 02_01_a.cpp
ubuntu@ubuntu-VirtualBox:~/Documents/Assignment02$ ./a
Sequential Time taken for vector scalar addition: 0.006754 seconds
Parallel Time taken for vector scalar addition: 0.00537863 seconds
```

Information:
Analysis:

Parallel Time < Sequential Time
Size of vector increase Parallel time decreases compare to Sequential time

Problem Statement 2:
Screenshots:

Code:

```
#include <iostream>
#include <omp.h>
#include <cmath>
#include <ctime>

double calculatePiSequential(int num_steps) {
double step = 1.0 / num_steps;
double pi = 0.0;
for (int i = 0; i < num_steps; ++i) {
double x = (i + 0.5) * step;
pi += 4.0 / (1.0 + x * x);
}
return step * pi;
}

double calculatePiParallel(int num_steps) {
double step = 1.0 / num_steps;
double pi = 0.0;

#pragma omp parallel
{
double local_pi = 0.0;
#pragma omp for
for (int i = 0; i < num_steps; ++i) {
double x = (i + 0.5) * step;
local_pi += 4.0 / (1.0 + x * x);
}

#pragma omp atomic
pi += local_pi;
}

return step * pi;
}

int main() {
int num_steps = 100000000; // Number of steps for Pi calculation

// Sequential execution
double start_time = clock();
double pi_seq = calculatePiSequential(num_steps);
double end_time = clock();
std::cout << "Sequential calculated value of Pi: " << pi_seq << std::endl;
```

```
std::cout << "Sequential Time taken for Pi calculation: " << (end_time - start_time) /
CLOCKS_PER_SEC << " seconds" << std::endl;

// Parallel execution
start_time = omp_get_wtime();
double pi_par = calculatePiParallel(num_steps);
end_time = omp_get_wtime();
std::cout << "Parallel calculated value of Pi: " << pi_par << std::endl;
std::cout << "Parallel Time taken for Pi calculation: " << (end_time - start_time) << " seconds" <<
std::endl;

return 0;
}
```

Output:

```
ubuntu@ubuntu-VirtualBox:~/Documents/Assignment02$ g++ -fopenmp -o a 02_02_a.cpp
ubuntu@ubuntu-VirtualBox:~/Documents/Assignment02$ ./a
Sequential calculated value of Pi: 3.14159
Sequential Time taken for Pi calculation: 0.323127 seconds
Parallel calculated value of Pi: 3.14159
Parallel Time taken for Pi calculation: 0.25386 seconds
ubuntu@ubuntu-VirtualBox:~/Documents/Assignment02$
```

Information:
Analysis:

Parallel Time < Sequential Time
Number of Steps increase Parallel time decreases compare to Sequential time

Github Link:

https://github.com/omkarauti11/HPC_LAB