**Class: Third Year B.Tech (Computer Science and Engineering)**

**Year: 2025-26 | Semester: Odd**

**Course: Cutting Edge Technologies Lab**

**Course Code: 7CS352**

**Practical No.: 6**

**Exam Seat No.: 23510030**

**Title of Practical: Study and Implementation of Reduction Operations and Nested Loop Parallelism Using collapse**

**Problem Statement 1: Matrix-Matrix Multiplication**

**Objective:**  
Implement matrix multiplication in parallel using OpenMP and analyze performance with reduction and nested loop parallelism.

**Code Example (Parallel using collapse):**

#include <stdio.h>

#include <omp.h>

#define N 500

int main() {

int A[N][N], B[N][N], C[N][N];

for(int i=0;i<N;i++)

for(int j=0;j<N;j++) {

A[i][j] = i+j;

B[i][j] = i-j;

C[i][j] = 0;

}

double start = omp\_get\_wtime();

#pragma omp parallel for collapse(2)

for(int i=0;i<N;i++)

for(int j=0;j<N;j++)

for(int k=0;k<N;k++)

C[i][j] += A[i][k] \* B[k][j];

double end = omp\_get\_wtime();

printf("Time for parallel matrix multiplication: %f seconds\n", end-start);

return 0;

}

**Screenshots:**

* Output (optional small matrix example)
* Execution time

**Information and Analysis:**

* Compare serial vs parallel execution times.
* Discuss effect of collapse(2) on nested loops.
* Calculate speedup:

Speedup=TserialTparallel\text{Speedup} = \frac{T\_{\text{serial}}}{T\_{\text{parallel}}}Speedup=Tparallel​Tserial​​

**Problem Statement 2: Vector Dot Product**

**Objective:**  
Compute the dot product of two vectors in parallel using reduction operations.

**Code Example (Using reduction):**

#include <stdio.h>

#include <omp.h>

#define N 1000000

int main() {

double A[N], B[N], dot = 0;

for(int i=0;i<N;i++) {

A[i] = i\*0.5;

B[i] = i\*0.3;

}

double start = omp\_get\_wtime();

#pragma omp parallel for reduction(+:dot)

for(int i=0;i<N;i++)

dot += A[i]\*B[i];

double end = omp\_get\_wtime();

printf("Dot product: %f\n", dot);

printf("Time for parallel dot product: %f seconds\n", end-start);

return 0;

}

**Screenshots:**

* Output and execution time

**Information and Analysis:**

* Reduction avoids race conditions when summing partial products.
* Compare serial vs parallel execution time.
* Speedup calculation.

**Problem Statement 3: Nested Loop Parallelism Example (Optional)**

**Objective:**  
Demonstrate parallelization of nested loops with collapse.

**Code Example:**

#include <stdio.h>

#include <omp.h>

#define N 500

int main() {

int matrix[N][N];

for(int i=0;i<N;i++)

for(int j=0;j<N;j++)

matrix[i][j] = i+j;

double start = omp\_get\_wtime();

#pragma omp parallel for collapse(2)

for(int i=0;i<N;i++)

for(int j=0;j<N;j++)

matrix[i][j] \*= 2; // some operation

double end = omp\_get\_wtime();

printf("Time for nested loop operation: %f seconds\n", end-start);

return 0;

}

**Screenshots:**

* Output (optional small array)
* Execution time

**Information and Analysis:**

* Discuss effect of collapse on load balancing among threads.
* Speedup analysis with different numbers of threads.