Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session 2025-2026

Vision: Dream of where you want.	**Mission:** Means to achieve Vision

Program Educational Objectives of the program (PEO): (broad statements that describe the professional and career accomplishments)

PEO1	**Preparation**	**P: Preparation**	**Pep-CL abbreviation pronounce as Pep-si-lL easy to recall**
PEO2	**Core Competence**	**E: Environment (Learning Environment)**	
PEO3	**Breadth**	**P: Professionalism**	
PEO4	**Professionalism**	**C: Core Competence**	
PEO5	**Learning Environment**	**L: Breadth (Learning in diverse areas)**	

Program Outcomes (PO): (statements that describe what a student should be able to do and know by the end of a program)

Keywords of POs:
Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

PSO Keywords: Cutting edge technologies, Research

"I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life." *to contribute to the development of cutting-edge technologies and Research*.

Integrity: I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

Sanskruti. Paunikar 24/10/2025
Name and Signature of Student and Date
(Signature and Date in Handwritten)

Session	2025-26 (ODD)	**Course Name**	High Performance Computing Lab
Semester	7 AIDS	**Course Code**	22ADS702
Roll No	21	**Name of Student**	Sanskruti. Paunikar

Practical Number	7
Course Outcome	**CO1:-**Understand and Apply Parallel Programming Concepts **CO1:-**Analyze and Improve Program Performance. **CO3:-**Demonstrate Practical Skills in HPC Tools and Environments.
Aim	Hybrid Programming with MPI + OpenMP Practical
Theory (100 words)	**Requirements** **1. Software:** • Linux OS (Ubuntu/RedHat recommended) • MPI library (OpenMPI / MPICH) • GCC compiler with OpenMP support **2. Hardware:** • Multi-core processor • Optional: Cluster with multiple nodes for full MPI execution 3. **MPI (Message Passing Interface)** • Used for communication between processes in a distributed memory system. • Each process has its own address space. **4. OpenMP (Open Multi-Processing)** • Used for parallelism within a shared memory node. • Allows multi-threading using #pragma omp parallel. **5. Hybrid Programming** • Combines MPI across nodes and OpenMP within nodes. • Reduces communication overhead and improves parallel efficiency.
Procedure and Execution (100 Words)	Steps of Implementation:- 1. Initialize MPI and get rank and size. 2. Distribute rows of the matrix A among MPI processes. 3. Each process computes its local result using OpenMP threads. 4. MPI_Reduce is used to gather results to the master process. 5. Master process prints the final result. **Steps for execution** **Step 1** - Compile: mpicc -fopenmp hybrid_mpi_openmp.c -o hybrid_mpi_openmp

Step 2 - Execute (using 2 MPI processes, adjust threads with OMP_NUM_THREADS): export OMP_NUM_THREADS=4 # Set number of OpenMP threads per process
mpirun -np 2 ./hybrid_mpi_openmp

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <mpi.h>
#include <omp.h>

#define N 8  // Size of matrix and vector

int main(int argc, char* argv[]) {
    int rank, size;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    int rows_per_proc = N / size;
    double A[rows_per_proc][N];
    double x[N];
    double y_local[rows_per_proc];
    double y[N];

    // Initialize vector x and matrix A
    if(rank == 0) {
        for(int i = 0; i < N; i++)
            x[i] = i + 1;  // Example vector: 1,2,3...
    }

    MPI_Bcast(x, N, MPI_DOUBLE, 0, MPI_COMM_WORLD); //
Broadcast vector to all processes

    // Initialize local part of matrix A
    for(int i = 0; i < rows_per_proc; i++) {
        for(int j = 0; j < N; j++) {
A[i][j] = (rank * rows_per_proc + i + 1) * (j + 1);
        }
    }
```

	`` // Parallel computation using OpenMP #pragma omp parallel for for(int i = 0; i < rows_per_proc; i++) { y_local[i] = 0.0; for(int j = 0; j < N; j++) { y_local[i] += A[i][j] * x[j]; } } // Gather results to root process MPI_Gather(y_local, rows_per_proc, MPI_DOUBLE, y, rows_per_proc, MPI_DOUBLE, 0, MPI_COMM_WORLD); // Print result in master process if(rank == 0) { printf("Result vector y:\n"); for(int i = 0; i < N; i++) { printf("%lf ", y[i]); } printf("\n"); } MPI_Finalize(); return 0; } ```

Output:

```c
#include <stdio.h>
#include <stdlib.h>
#include <mpi.h>
#include <omp.h>

#define N 8  // Size of matrix and vector

int main(int argc, char* argv[]) {
    int rank, size;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    int rows_per_proc = N / size;
    double A[rows_per_proc][N];
    double x[N];
    double y_local[rows_per_proc];
    double y[N];

    // Initialize vector x and matrix A
    if(rank == 0) {
        for(int i = 0; i < N; i++)
            x[i] = i + 1;  // Example vector: 1,2,3...
    }

    MPI_Bcast(x, N, MPI_DOUBLE, 0, MPI_COMM_WORLD); // Broadcast vector to all processes

    // Initialize local part of matrix A
    for(int i = 0; i < rows_per_proc; i++) {
        for(int j = 0; j < N; j++) {
            A[i][j] = (rank * rows_per_proc + i + 1) * (j + 1);
        }
    }
```

Do you want to install the recom
Pack' extension from Microsoft f

```
shreyyoo@localhost:~/Downloads/hpc_7$ ls
hybrid_mpi_openmp.c
shreyyoo@localhost:~/Downloads/hpc_7$ mpicc -fopenmp hybrid_mpi_open
mp.c -o hybrid_mpi_openmp
shreyyoo@localhost:~/Downloads/hpc_7$ mpirun -np 2 ./hybrid_mpi_open
mp
Result vector y:
204.000000 408.000000 612.000000 816.000000 1020.000000 1224.000000
1428.000000 1632.000000
shreyyoo@localhost:~/Downloads/hpc_7$
```

Output Analysis	The program executes successfully and gives us the resultant vector y as an output.
Github link	https://github.com/sanskruti-1234/HPC.git
Conclusion	The experiment successfully compiled and ran a hybrid MPI/OpenMP parallel program using 2 MPI processes. The program executed correctly, as confirmed by the output of the 8-element result vector y demonstrating that the system and the parallel code are properly configured for hybrid high-performance computing.

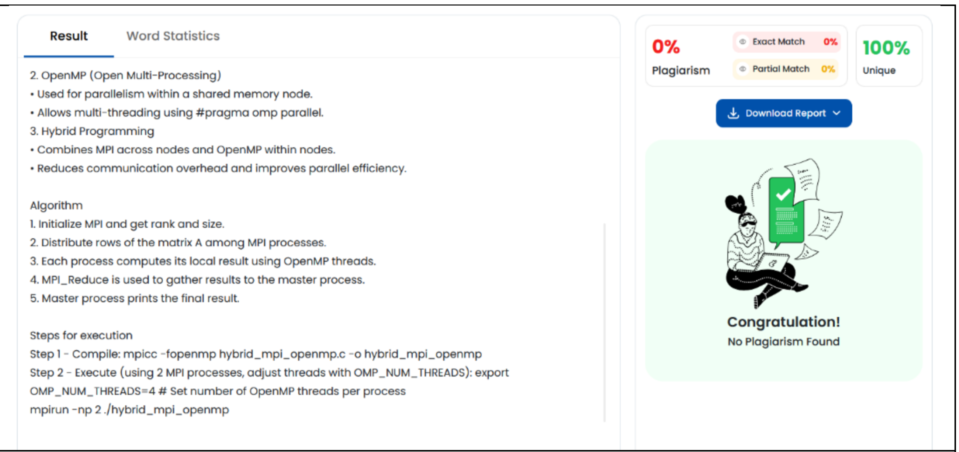
Nagar Yuwak Shikshan Sanstha’s
Yeshwantrao Chavan College of Engineering
(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)
Hingna Road, Wanadongri, Nagpur - 441 110
NAAC A++
Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu
Department of Artificial Intelligence & Data Science

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

<table>
<tr><td>Plag Report (Similarity index < 12%)</td><td>
</td></tr>
<tr><td>Date</td><td>24/10/2025</td></tr>
</table>