**Session 2025-2026**

|  |  |
| --- | --- |
| **Vision:** To help businesses uncover crucial  insights | **Mission:** To be a good data scientist |

**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):** 1. Understand and Apply Parallel Programming Concepts

2. Analyse and Improve Program Performance.

3. Demonstrate Practical Skills in HPC Tools and Environments.

**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.

**Name and Signature of Student and Date**

Shreyas Chaurey – 24/10/2025

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Session** | **2025-26 (ODD)** | | **Course Name** | **HPC Lab** | |
| **Semester** | **7** | | **Course Code** | 22ADS706 | |
| **Roll No** | 61 | | **Name of Student** | Shreyas Chaurey | |
|  |  | |  |  |  |
| Practical Number | | 6 | | | |
| Course Outcome | | 1. Understand and Apply Parallel Programming Concepts 2. Analyse and Improve Program Performance | | | |
| Aim | | Parallel Pi Calculation using MPI | | | |
| Problem Definition | | Parallel Pi Calculation using MPI | | | |
| Theory  (100 words) | | The value of π can be approximated using the Monte Carlo method or numerical integration.  One common numerical method is based on the integration of the area under a curve:    This integral can be approximated by dividing the interval [0,1] into N subintervals and summing the area of rectangles:    Using MPI, the work of summing these rectangles can be distributed among multiple  processes. Each process computes a partial sum, and the master process (rank 0) collects the  results to compute the final value of π.  Software/Hardware Requirements:  Hardware: Multi-core CPU or cluster with multiple nodes  Software:  o Linux/Unix OS  o MPICH or OpenMPI  o GCC Compiler  **Algorithm:**  1. Initialize MPI environment using MPI\_Init.  2. Get the rank (ID) of each process and total number of processes using  MPI\_Comm\_rank and MPI\_Comm\_size.  3. Divide the range [0,1] among processes. Each process computes a partial sum of π  for its assigned range.  4. Use MPI\_Reduce to collect and sum all partial results at the root process.  5. The root process prints the final value of π.  6. Finalize MPI using MPI\_Finalize. | | | |
| Code: | | #include &lt;stdio.h&gt;  #include &lt;mpi.h&gt;  int main(int argc, char\* argv[]) {  int rank, size, n = 1000000, i;  double h, sum = 0.0, x, local\_sum = 0.0, pi;  MPI\_Init(&amp;argc, &amp;argv);  MPI\_Comm\_rank(MPI\_COMM\_WORLD, &amp;rank);  MPI\_Comm\_size(MPI\_COMM\_WORLD, &amp;size);  h = 1.0 / (double) n;  // Each process computes its portion  for (i = rank; i &lt; n; i += size) {  x = h \* (i + 0.5);  local\_sum += 4.0 / (1.0 + x \* x);  }  local\_sum \*= h;  // Reduce all local sums to get the final result  MPI\_Reduce(&amp;local\_sum, &amp;pi, 1, MPI\_DOUBLE, MPI\_SUM, 0,  MPI\_COMM\_WORLD);  if (rank == 0) {  printf(&quot;Calculated value of Pi = %.16f\n&quot;, pi);  }  MPI\_Finalize();  return 0;  } | | | |
| Output | | A computer screen shot of a program  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect. | | | |
| Output Analysis | | OpenMPI executes the program successfully and gives us the calculated value of Pi. | | | |
| Link of student Github profile where lab assignment has been uploaded | |  | | | |
| Conclusion | | The Parallel Pi Calculation using MPI experiment successfully demonstrated that utilizing the Message Passing Interface significantly reduces the computation time compared to a sequential approach by distributing the numerical integration workload among multiple processes. | | | |
| Plag Report (Similarity index < 12%) | |  | | | |
| Date | | 24/10/2025 | | | |