ATME College of Engineering

Department of Computer Science Engineering – (Data Science)

COURSE MODULE FOR THE SESSION 2025-2026 (ODD SEMESTER)

Course Syllabi with CO's

Faculty Name: Dr. Vinod Kumar P	**Academic Year:** 2025 – 2026
Course Name: Parallel Programming	

Course Code	**BDS701**	CIE Marks	50
Teaching Hours/Week (L:T:P: S)	3:0:2:0	SEE Marks	50
Total Hours of Pedagogy	40 hours Theory + 8-10 Lab slots	Total Marks	100
Credits	04	Exam Hours	03
Examination nature (SEE)	Theory/Practical		

Course objectives:

This course will enable to,

- Explore the need for parallel programming
- Explain how to parallelize on MIMD systems
- To demonstrate how to apply MPI library and parallelize the suitable programs
- To demonstrate how to apply OpenMP pragma and directives to parallelize the suitable programs
- To demonstrate how to design CUDA program

MODULE-1

Introduction to parallel programming, Parallel hardware and parallel software – Classifications of parallel computers, SIMD systems, MIMD systems, Interconnection networks, Cache coherence, Shared-memory vs. distributed-memory, Coordinating the processes/threads, Shared-memory, Distributed-memory.

MODULE-2

GPU programming, Programming hybrid systems, MIMD systems, GPUs, Performance – Speedup and efficiency in MIMD systems, Amdahl's law, Scalability in MIMD systems, Taking timings of MIMD programs, GPU performance.

MODULE-3

Distributed memory programming with MPI – MPI functions, The trapezoidal rule in MPI, Dealing with I/O, Collective communication, MPI-derived datatypes, Performance evaluation of MPI programs, A parallel sorting algorithm.

MODULE-4

Shared-memory programming with OpenMP – openmp pragmas and directives, The trapezoidal rule, Scope of variables, The reduction clause, loop carried dependency, scheduling, producers and consumers, Caches, cache coherence and false sharing in openmp, tasking, tasking, thread safety.

MODULE-5

Department of Computer Science Engineering – (Data Science)

GPU programming with CUDA - GPUs and GPGPU, GPU architectures, Heterogeneous computing, Threads, blocks, and grids Nvidia compute capabilities and device architectures, Vector addition, Returning results from CUDA kernels, CUDA trapezoidal rule I, CUDA trapezoidal rule II: improving performance, CUDA trapezoidal rule III: blocks with more than one warp.

PRACTICAL COMPONENT OF IPCC

Sl.NO	Experiments
1	Write a OpenMP program to sort an array on n elements using both sequential and parallel mergesort (using Section). Record the difference in execution time.
2	Write an OpenMP program that divides the Iterations into chunks containing 2 iterations, respectively (OMP_SCHEDULE=static,2). Its input should be the number of iterations, and its output should be which iterations of a parallelized for loop are executed by which thread. For example, if there are two threads and four iterations, the output might be the following: a. Thread 0 : Iterations 0 –– 1 b. Thread 1 : Iterations 2 –– 3
3	Write a OpenMP program to calculate n Fibonacci numbers using tasks.
4	Write a OpenMP program to find the prime numbers from 1 to n employing parallel for directive. Record both serial and parallel execution times.
5	Write a MPI Program to demonstration of MPI_Send and MPI_Recv.
6	Write a MPI program to demonstration of deadlock using point to point communication and avoidance of deadlock by altering the call sequence
7	Write a MPI Program to demonstration of Broadcast operation.
8	Write a MPI Program demonstration of MPI_Scatter and MPI_Gather
9	Write a MPI Program to demonstration of MPI_Reduce and MPI_Allreduce (MPI_MAX, MPI_MIN, MPI_SUM, MPI_PROD)

Course outcomes (Course Skill Set):

At the end of the course, the student will be able to:

- Explain the need for parallel programming
- Demonstrate parallelism in MIMD system.
- Apply MPI library to parallelize the code to solve the given problem.
- Apply OpenMP pragma and directives to parallelize the code to solve the given problem
- Design a CUDA program for the given problem.

Suggested Learning Resources:

Textbook:

1. Peter S Pacheco, Matthew Malensek – An Introduction to Parallel Programming, second edition, Morgan Kauffman.
2. Michael J Quinn – Parallel Programming in C with MPI and OpenMp, McGrawHill.

Reference Books:

1. Calvin Lin, Lawrence Snyder – Principles of Parallel Programming, Pearson
2. Barbara Chapman – Using OpenMP: Portable Shared Memory Parallel Programming, Scientific and Engineering Computation
3. William Gropp, Ewing Lusk – Using MPI:Portable Parallel Programing, Third edition, Scientific and Engineering Computation.

Web links and Video Lectures (e-Resources):

Introduction to parallel programming: https://nptel.ac.in/courses/106102163

Activity Based Learning (Suggested Activities in Class)/ Practical Based learning

Programming Assignment at higher bloom level (10 Marks)

The Correlation of Course Outcomes (CO's) and Program Outcomes (PO's)

Course Code	BDS701	TITLE: Parallel Programming			Faculty Name		Dr. Vinod Kumar P				
List of Course Outcomes	Program Outcomes										
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO-1	3	-	-	-	-	-	-		-	-	-	-
CO-2	2	-	-	-	-	-	-	-	-	-	-	-
CO-3	3	3	3	-	-	-	-	-	-	-	-	-
CO-4	3	3	3	2	-	-	-	-	-	-	-	-
CO-5	3	3	3	2	-	-	-	-	-	-	-	2

Note: 3 = Strong Contribution, 2 = Average Contribution, 1 = Weak Contribution, - = No Contribution

The Correlation of Course Outcomes (CO's) and Program Specific Outcomes (PSO's)

Course Code	**BDS701**	**Title:** Parallel Programming	**Faculty name: Dr. Vinod Kumar P**
List of Course Outcomes	**Program Specific Outcomes**		
	PSO1	**PSO2**	**PSO3**
CO1	2	-	-
CO2	2	-	-
CO3	3	-	-
CO4	3	-	-
CO5	3	-	-

Note: 3 = Strong Contribution, 2 = Average Contribution, 1 = Weak Contribution, - = No Contribution