University of Gujrat
Department of Information Technology

<table>
<tr><td>Title</td><td>Parallel and Distributed Computing</td></tr>
<tr><td>Books</td><td>1. Distributed Systems: Principles and Paradigms, A. S. Tanenbaum and M. V. Steen, Prentice Hall, 2nd Edition, 2007 2. Distributed and Cloud Computing: Clusters, Grids, Clouds, and the Future Internet, K Hwang, J Dongarra and GC. C. Fox, Elsevier, 1st Ed.</td></tr>
<tr><td>Course Intro</td><td>At the end of the course the students will be able to: 1. Learn about parallel and distributed computers. 2. Write portable programs for parallel or distributed architectures using Message-Passing Interface (MPI) library 3. Analytical modelling and performance of parallel programs. 4. Analyze complex problems with shared memory programming with open MP. * BT= Bloom’s Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective domain</td></tr>
<tr><td>Aims and Objectives</td><td><table>
<tr><td>CLO-1</td><td>Learn about parallel and distributed computers.</td></tr>
<tr><td>CLO-2</td><td>Write portable programs for parallel or distributed architectures using Message-Passing Interface (MPI) library</td></tr>
<tr><td>CLO-3</td><td>Analyze complex problems with shared memory programming with openMP.</td></tr>
</table></td></tr>
<tr><td>Course Description</td><td>Asynchronous/synchronous computation/communication, , fault tolerance, GPU architecture and programming, heterogeneity, interconnection topologies, load balancing, memory consistency model, memory hierarchies, Message passing interface (MPI), MIMD/SIMD, multithreaded programming, parallel algorithms & architectures, parallel I/O, performance analysis and tuning, power, programming models (data parallel, task parallel, process-centric, shared/distributed memory), scalability and performance studies, scheduling, storage systems, synchronization, and tools (Cuda, Swift, Globus, Condor, Amazon AWS, OpenStack, Cilk, gdb, threads, MPICH, OpenMP, Hadoop, FUSE).</td></tr>
<tr><td>Recommended Text Books</td><td>1. Distributed Systems: Principles and Paradigms, A. S. Tanenbaum and M. V.Steen, Prentice Hall, 2nd Edition,</td></tr>
</table>

	2007 Distributed and Cloud Computing: Clusters, Grids, Clouds, and the Future Internet,KHwang, J Dongarra and GC. C. Fox, Elsevier, 1st Ed.

Sixteen-week lecture plan

Week	Lecture	Topic
extra	1	load balancing,
extra	2	MIMD/SIMD,
extra	3	parallel I/O
extra	4	performance analysis and tuning
extra	5	power, programming models
extra	6	programming models(data parallel, task parallel, process-centric, shared/distributed memory),
extra	7	Cilk (lec#27), gdb (lec#28)
Mam-extra		Communication Cost in parallel Machine (lec #3(1))
Clg	1	**Parallel and Distributed Computing (lec#1), Parallelism**
Clg	2	**Multi-processor, Multi-Computer (lec#2), Flynn's Taxonomy(lec#7)**
Clg	3	**Asynchronous/synchronous computation/communication**(lec #3(1) pdf)
Clg	4	**Concurrency control, Fault tolerance**
Clg	5	**Heterogeneity(lec#6), Interconnection topologies(lec#8)**
Clg	6	**Parallel Algorithm Design Life Cycle(lec#9)**
Clg	7	**Decomposition Techniques (lec#11)**
Clg	8	**Memory Consistency Model , Memory Hierarchies , Mapping Schemes(lec#10)**
Clg	9	**Distributed Memory Programming Model (lec #15)**
Clg	10	**Multithreaded Programming, Parallel Algorithms & Architectures** (parallel algorithm models lec 5)
Clg	11	**Basic Communication Operations (lec# 14)** 11.1 All-to-All Broadcast 11.2 All-to-All Reduction
Clg	12	**Task Parallel, Shared/Distributed Memory, Synchronization**
Clg	13	**Scalability and Performance Studies, Scheduling, Storage Systems**
Clg	14	**GPU** architecture and programming, (lec#16(2))

Clg	15	**Tools** 15.1 Cuda (lec#16(1)) 15.2 Swift (lec#19) 15.3 Globus (lec#20) 15.4 Condor (lec#21) 15.5 Amazon AWS (lec#22) 15.6 OpenStack (lec#23) 15.7 Threads (lec#24) 15.8 MPI (lec#19) 15.9 OpenMP (lec#20) 15.10 Hadoop (lec#25) 15.11 FUSE (lec#26)

Main Coding Areas

You'll write code in these categories:

1. **Message Passing Interface (MPI)** —
   Writing programs where multiple computers (nodes) communicate using messages.
   → Example: matrix multiplication using MPI.
2. **OpenMP / Multithreading** —
   Shared memory programming using threads.
   → Example: parallel sorting using OpenMP.
3. **CUDA / GPU Programming** —
   Using NVIDIA GPUs for high-speed computation.
   → Example: image processing using CUDA kernels.
4. **Parallel Algorithms** —
   Implementing algorithms that divide work across processors.
   → Example: parallel search, merge sort, or prefix sum.
5. **Cloud and Distributed Tools (Optional or Project-Based)** —
   Using tools like **Hadoop, AWS, OpenStack, Globus**, etc., to handle distributed computation or storage.
   → Example: running MapReduce on Hadoop.

Name	What is it? (Language / Tool / API)	Main Use	Used in Parallel Computing?	Used in Distributed Computing?	Advantages	Simple Example (if applicable)
CUDA	Programming platform & API by NVIDIA	GPU programming	✅ Yes (massive parallelism on GPU)	❌ No	Very fast, thousands of threads, great for ML & simulations	`c\n__global__ void add(int *a){ a[0]+=1; }\n`
Swift	Programming Language (Apple)	App development	❌ No	❌ No	Fast, safe, modern syntax	`swift\nprint(\"Hello Swift\")\n`
Globus	Middleware tool	Secure file transfer	❌ No	✅ Yes	Reliable data transfer, security, used in grids	❌
Condor (HTCondor)	Job scheduling system	High-throughput computing	❌ No	✅ Yes	Automatic job management, fault tolerance	❌
Amazon AWS	Cloud platform	Cloud computing	⚠ Indirect (EC2 clusters)	✅ Yes	Scalable, pay-as-you-go, global infrastructure	❌
OpenStack	Cloud OS / Platform	Private cloud	⚠ Indirect	✅ Yes	Open-source, flexible, private cloud	❌
Threads	Programming concept	Multithreading	✅ Yes (shared memory)	❌ No	Fast communication, low overhead	`c\npthread_create(&t,NULL,func,NULL);\n`

MPI	Message Passing Interface (API)	Cluster computing	✕ No	✓ Yes	Scalable, portable, high performance	`c\nMPI_Send(&x,1,MPI_INT,1,0,MPI_COMM_WORLD);\n`
OpenMP	Parallel programming API	Shared-memory systems	✓ Yes	✕ No	Easy to use, compiler directives	`c\n#pragma omp parallel for\nfor(int i=0;i<n;i++){}\n`
Hadoop	Distributed framework	Big data processing	✕ No	✓ Yes	Fault tolerance, handles huge data	✕
FUSE	Filesystem interface	User-space filesystems	✕ No	⚠ Indirect	Easy FS development, flexible	✕