GUJARAT TECHNOLOGICAL UNIVERSITY

Parallel Programming Tools and Model
SUBJECT CODE:3710220

Type of course: Elective

Prerequisite: Data Structures, Design and Analysis of Algorithms, Computer Architecture

Rationale: Parallel computing has become mainstream and very affordable today with the growing number of cores on a chip. Programming them efficiently has become an indispensable knowledge for the future. Parallel Programming Tools and Techniques is a hands-on course involving significant parallel programming on compute-clusters, multi-core CPUs and massive-core GPUs.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks				Total Marks
L	T	P	C	Theory Marks		Practical Marks		
				ESE(E)	PA (M)	PA (V)	PA (I)	
3	0	2	4	70	30	30	20	150

Content:

Sr. No.	Topics	Teaching Hours	Module Weightage
1	**Introduction to Parallel Programming Paradigms and performance analysis:** Necessary background to follow an parallel programming course. Issues when programming multicore architectures. General introduction of the main techniques and basic features of current performance analysis tools.	3	6%
2	**Parallel Architecture:** Introduction to parallel hardware: Multi-cores and multiprocessors, Parallel Architecture Components, Flynn's Taxonomy, Amdahl's Law, Network Topology, Multiprocessor organization and Cache Coherence, Cache Coherence Protocols, Memory Consistency models.	8	17%
3	**Shared and Distributed Memory Programming : OpenMP, MPI and PVM:** Synchronization Locks and barriers, Hardware primitives for efficient lock implementation, Lock algorithms, Relaxed consistency models, High-level language memory models (such Java and/or C++), Memory fences, Summary of basic features in OpenMP, MPI, PVM and POSIX thread API. Advanced features in OpenMP, MPI, PVM and hybrid programming.	11	23%
4	**Data acquisition and performance analytics:** Tracing of sequential and parallel applications, Trace processing and performance analytics, Profiling, Profile Tools.	4	8%
5	**Models and performance prediction**: Trace-based modeling of parallel performance. Architectural parameters: CPU, memory, interconnect.	3	6%
6	**Dataflow programming and novel paradigms for accelerator-based architectures**: Dataflow paradigms (OmpSs). Runtime exploitation of parallelism and architecture hiding. Advanced parallel programming using accelerators: CUDA, OpenCL, OpenACC and others if any.	9	19%

7	**Principles of Parallel Algorithm Design:** Embarrassingly Parallel Computations, Partitioning and Divide-and-Conquer Strategies, Pipelined Computations, Synchronous Computations, Load Balancing and Termination Detection, Sorting Algorithms, Numeric Algorithms, Image Processing Algorithms	7	15%
8	**Analysis and optimization of real applications**: Analysis of large applications (sequential and/or parallel) and optimization using hybrid programming paradigms (dataflow, shared- and distributed-memory and accelerators).	3	6%

Reference Books:

1) Parallel Programming in OpenMP, Rohit Chandra, Leo Dagum, Dave Kohr, Dror Maydan, Jeff McDonald, Ramesh Menon, Academic Press Morgan Kaufmann Publishers, San Diego, CA, 2001.
2) Programming Massively Parallel Processors: A Hands-on Approach; David Kirk, Wen-mei Hwu; Morgan Kaufman; 2010 (ISBN: 978-0123814722)
3) CUDA Programming: A Developer's Guide to Parallel Computing with GPUs; Shane Cook; Morgan Kaufman; 2012 (ISBN: 978-0124159334)
4) Peter S Pacheco, An Introduction to Parallel Programming, Morgan Kaufmann,2011.
5) M Herlihy and N Shavit, The Art of Multiprocessor Programming Morgan Kaufmann, 2008.
6) J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach,4th Ed., Morgan Kaufmann/Els India, 2006.
7) M. SasiKumar, Dinesh Shikhare P. Raviprakash, Introduction to Parallel Processing, PHI Publication
8) V. Rajaraman And C. Siva Ram Murthy, Parallel Computers – Architecture And Programming, Second Edition, PHI Publication
9) M. J. Quinn, Parallel Computing: Theory and Practice, McGraw Hill, Second Edition
10) Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar "Introduction to Parallel Computing", Second Edition, Addison Wesley, 2003. ISBN: 0-201-64865
11) Wilkinson, M.Allen,"Parallel Programming Techniques and Applications using networked workstations and parallel computers", Prentice Hall, 1999.

Course Outcomes:

- Classify parallel architectures parameters that are essential for the classification of modern parallel processing systems.
- Design efficient parallel solutions to scientific problems
- Implement parallel programs in prominent programming models and evaluate their performance using various metrics

Practical List:

Use Valgrind, Vtune Amplifier, Nvidia Visual Profiler and Nvidia Nsight to identify hotspots and other parameters for detailed analysis of following the practicals.

1) Calculate standard deviation using Pthread, OpenMP and MPI.
2) Write parallel code for Matrix Matrix Multiplication using MPI cluster of 4 nodes, OpenMP, PVM cluster of 4 nodes, OpenACC and CUDA and compare and plot the performance in terms of execution time for Matrix size of 1000 x 1000, 5000 x 5000, 10,000 x 10,000, 20,000 x 20,000.
3) Write the programs in MPD or in C with the Pthreads library for the following:

a) Sequential Jacobi iteration program
b) Parallel Jacobi iteration program
c) Sequential multigrid program
d) Parallel multigrid program

4) Perform Monte Carlo simulation using NVIDIA's CURAND library for random number generation.

Write your own small program to compute the average value of

$$az^2 + bz + c$$

where z is a standard Normal random variable (i.e. zero mean and unit variance, which is what the random number generator produces) and a, b, c are constants which you should store in constant memory.

It is suggested to use each thread to average over 100 values, and then write this to a device array which gets copied back to the host for the averaging over the contributions from each of the threads.

(Note: the average value should be close to a + c.)

5) Implement 3D Laplace Finite Solver using CUDA and OpenACC.