HIGH PERFORMANCE COMPUTING

I Semester: CSE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
BCSC06	Elective	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Total Tutorials: Nil	Total Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

High performance computing is to give solid foundations for developing, analyzing, and implementing parallel and locality-efficient algorithms. This course focuses on theoretical underpinnings. To give a practical feeling for how algorithms map to and behave on real systems, we will supplement algorithmic theory with hands-on exercises on modern HPC systems, such as Cilk Plus or OpenMP on shared memory nodes, CUDA for graphics co-processors (GPUs), and MPI and PGAS models for distributed memory systems.

II. COURSE OBJECTIVES:

The students will try to learn:

1. Understand the fundamental principles in design and programming of parallel algorithms.
2. Study the approaches to achieve high performance models in real time applications.
3. Explore on parallel computing development tools and technologies.
4. Illustrate on add on tools to address the performance issues, analysis, data transformation and visualization.

III. COURSE OUTCOMES:

After successful completion of the course, students should be able to :

CO 1	**Make use of** GPU programming for running Highly Parallel general purpose competitions in 2-Dimensional and 3-Dimensional thread mapping.	Create
CO 2	**Identify** different types of memories used in GPUs for performance evaluation for a specific application.	Apply
CO 3	**Develop** a GPU program for usage of concurrent data structures applied in different types of functions.	Evaluate
CO 4	**Make use of** stream processing techniques used in GPUs in applications of weather modeling & medical applications.	Apply
CO 5	**Develop** a GPU program in application of image processing, graph algorithms & deep learning.	Knowledge

IV. SYLLABUS:

MODULE-I: DESIGN OF PARALLEL ALGORITHMS (10)

Parallel computers and computation, a parallel machine model, a parallel programming model, parallel algorithm examples, partitioning, communication, agglomeration, mapping, load balancing algorithms, task, scheduling algorithms, case studies, random numbers generation, hypercube algorithms, vector reduction, matrix transposition, merge sort.

MODULE-II: APPROACHES TO PERFORMANCE MODELING (10)

A quantitative basis for design, defining performance, approaches to performance modeling, developing models, performance parameters, time, scalability, overheads, bandwidth, efficiency, speed, interconnection networks, input/output; Case study: Shortest path algorithms, Floyd's algorithm, dijkstra's algorithm , modular design review, modularity and parallel computing performance analysis; Case study: Convolution, tuple space and matrix multiplication.

MODULE-III: PARALLEL COMPUTING DEVELOPMENT TOOLS (08)

C++ review, C, C++ introduction, concurrency, locality, processor objects, global pointers thread placement, communication, remote operations. Synchronization, mutual exclusion, data transfer functions, asynchronous communication, determinism, mapping, modularity performance issues.

MODULE-IV: PARALLEL COMPUTING DEVELOPMENT TOOLS (09)

Fortran M, concurrency, communication, unstructured communication, asynchronous communication, determinism, argument passing, mapping, modularity, high performance Fortran, data parallelism, concurrency, data distribution, dummy arguments and modularity other HPF features, performance issues.

MODULE-V: ADD ON TOOLS FOR DEVELOPMENT (09)

Message passing libraries: The MPI programming model, MPI basics, C and Fortran language bindings with MPI functions, global operations, asynchronous communication, modularity, other MPI features, performance issues, performance tools, performance analysis, data collection, data transformation and visualization tools, paragraph, upshot—pablo, gauge, paraide, IBM's parallel environment, AIMS, custom tools.

V. TEXT BOOKS:

1. Ion Foster, “Designing and Building Parallel Programs”, Addison Wesley, 1st Edition, 2003.

VI. REFERENCE BOOKS:

1. Arjen Markus, “Modern Fortran in Practice”, Cambridge University Press, 1st Edition, 2012.
2. Charles H. Koelbe, “High Performance Fortran Handbook”, MIT Press, 1st Edition, 1993.
3. Michael J. Quinn, “Parallel Programming in C with MPI and Open MPI”, Tata McGraw-Hill Publishing Company Ltd, 1st Edition, 2003.

VII. WEB REFERENCES:

1. http://www.drdobbs.com/parallel/designing-parallel-algorithms-part-1/223100878.
2. http://searchcloudapplications.techtarget.com/tip/How-to-use-application-performance-modelingtechniques.
3. https://computing.llnl.gov/tutorials/parallel_comp/.

VIII. E-TEXT BOOKS:

1. https://www.free-ebooks.net/ebook/High-Performance-Computing.
2. https://archive.org/details/HighPerformanceComputing.