Course number : **ME522**
Course Name : **High-Performance Scientific Computing**
Credit : **3**
Distribution : **2-0-1-3**
Intended for : **BTech / MTech / MS / MSc / PhD**
Prerequisite : **Engineering Mathematics, Introduction to Programming or an equivalent course**
Mutual Exclusion: **CS508 (Introduction to Heterogeneous Computing)**

---

1. Preamble:

To introduce high-performance scientific computing (HPSC) to the postgraduate students in engineering and sciences starting their research in numerical sciences (e.g. computational mechanics, molecular dynamics, etc.). This course will cover efficient programming techniques (including Python and shell scripting), version control, parallel programming using OpenMP and MPI, graphics and visualisation, and cloud computing. These techniques will help the students carry out their numerical research more effectively. Fortran/C will be used as the base programming language for implementing OpenMP and MPI directives. *Each course lecture is supplement by a one-hour lab helping reinforce the concepts*. In addition, the course will make significant use of the *HPC cluster of IIT Mandi* for course assignments and projects, helping students get comfortable with the infrastructure for future use in research.

2. Course Modules with quantitative lecture hours:

Introduction to the course: Definition of HPC, history and latest developments, Moore's law, introduction to scientific computing, challenges with setting up HPC/data centres (storage, power supply and thermal management), topology of processors, demonstration of the use of HPC to solve a heat conduction problem, sparse matrices, binary storage, fixed-point and floating-point real numbers, IEEE standards, virtual machines, Unix shell and commands. **(2 Hours)**

Introduction to programming languages, version control and makefiles: Introduction to compiled programming languages (precision, compiler optimisation, timing codes, LAPACK and BLAS), Git (Git commands, Github, graphical Git tools, Git demo), build systems and dependency checking. **(6 Hours)**

Python: Introduction, interpreted versus compiled languages, object-oriented language, syntax, conditionals, loops, functions, modules, data structures (lists and arrays), mutable and immutable objects, NumPy, linear algebra in Python, Pylab, SciPy, IPython, IPython notebook (Jupyter), unit test, nosetests, graphics and visualisation (matplotlib, Mayavi, Visualisation Tool Kit, ParaView), debugging, just-in-time compilers for Python (such as PyPy, Numba, LLVM), ASCII and binary output, HDF and NetCDF binary formats, demonstrations. **(4 Hours)**

Parallel computing: Introduction to parallelization, computer architecture (memory hierarchy, CPU, registers, cache), latency and throughput, cache lines, spatial locality, array ordering, cache collisions, padding, parallelizing algorithms (strip mining and loop reordering), shared-memory and distributed-memory parallelism, threads, parallelization issues (contention, dependencies, synchronization, and cache coherence), scaling (Amdahl's law, speedup, strong and weak scaling), SPMD (single program, multiple data) and SIMD (single instruction, multiple data), fine grain and coarse grain parallelism.

OpenMP: introduction, fork and join, synchronizations, race conditions, compiler directives, heap and stack memory, barriers, overheads, reductions, data dependencies, thread-safe functions, pseudo-random number generators in parallel, example codes, and demonstrations.

MPI: introduction, message passing, domain decomposition, MPI communicators, MPI modules and functions (broadcast, reduce, allreduce, MPI send and receive), master–worker paradigm, example codes, and demonstrations.

Comparison of OpenMP and MPI – numerical integration using adaptive quadratures

Use of HPC clusters for computing **(10 Hours)**

Scientific computing: Solution to the steady-state diffusion problem using the finite difference method, Jacobi method with OpenMP and MPI, numerical integrals using Monte Carlo (MC) method, solution to Poisson problem using MC method **(3 Hours)**

Cloud computing: Cloud computing demonstration using machine images **(1 Hour)**

Laboratory/practical/tutorial Modules: One hour lab following each lecture

Details of labs:

Hours 1 – 2: Introductory heat conduction problem and linear algebra, and Unix shell

Hours 3 – 8: Roots of a polynomial using numerical methods, version control, makefiles

Hours 9 – 12: Python lists and modules using numerical integration, Jupyter notebook, Numpy arrays and timing codes

Hours 13 – 22: Vector normalisation, Approximation of pi, adaptive quadratures, norms of matrices, LU factorisation, iterative methds to solve linear systems, Finite difference method for solving diffusion equation. All problems using OpenMP and MPI.

Hours 23 – 25: Monte Carlo methods to solve a steady-state diffusion problem

Hour 26: Cloud computing exercise

3. Text books:

No text book for this course

4. References:

L. R. Scott, T. Clark, B. Bagheri, Scientific Parallel Computing, Princeton University Press, 2005.

C. Lin and L. Snyder, Principles of Parallel Programming, Pearson, 2008.

R. Chandra, L. Dagum, et. al., Parallel Programming in OpenMP, Academic Press, 2001.

M. Quinn, Parallel Programming in C with MPI and OpenMP, McGraw-Hill Education, 2003.

G. Karniadakis, R. Kirby II, Parallel Scientific Computing in C++ and MPI, Cambridge University Press, 2003.

W. Gropp, E. Lusk, A. Skjellum, Using MPI, The MIT Press, 2014.

LD Fosdick, ER Jessup, CJC Schauble, G Domik An Introduction to High-performance Scientific Computing, MIT Pres, 1996.

Indian Institute of Technology Mandi