β-test- ParProc-TC v1.0

Parallel Processing Training

Parallel Processing Training Courseware *v*-1.0

Designed for Testing, Benchmarking & Performance Activities

(Draft Version: To be Reviewed on September 01, 2006)

Document Title	Parallel Processing Training Courseware (**β-test-ParProc-TC-*v*1.0**)
Users	Betatesting Group Members of NPSF, C-DAC,Pune
Source	Parallel Computing Workshops conducted by Betatesting group, C-DAC, Pune & Author's Experience on Parallel Processing Projects
Date:	August 01, 2006
No. of Pages	43
Author	VCV.Rao
Contact	betatest@cdac.in

Beta (β) Testing Group NPSF

Beta(β)- testing Group, National PARAM Supercomputing Facility, Centre for Development of Advanced Computing (C-DAC), Pune

Non-Disclosure

This material is developed and provided by the Centre for Development of Advanced Computing (C-DAC), Pune, India. Licensor grants to Licensee, an Indian Government agency or a non-profit educational institution ("Licensee"), a royalty free, non-transferable license to use the copy of the product as per terms and conditions of this License Agreement. Licensee is permitted to use the product solely for Licensee's own internal research purposes, and alter, modify, adapt the product for Licensee's own internal research purposes.

Contents:

1. Objective

The complete courseware forms a single concentrated course on parallel computing, which is a continuously evolving High Performance Computing resource at NPSF. The courseware can be easily tailored to the developer, testing group, user community to extract performance of large scale applications and gives a strong foundation on programming models for Benchmarking computing systems in the range of *teraflop* to *petaflop* computing systems. The philosophy is to introduce new functionality and concepts to solve a design, implementation or analysis of problem in this courseware. The aim is to design Parallel Processing training courseware to build expertise for *Testing and Benchmark large Parallel Computing Platforms.* The complete training program gives a strong foundation for Testing, Benchmarking large message passing clusters in the range of *teraflop* to *petaflop* computing systems.

The courseware is useful for beginners, middle level and advanced level parallel computing users. The courseware is designed as a standard training courseware for classroom adoption in Computer Science or Computer Engineering or High Performance Application groups at Research & Development HPC organizations or M.Tech/M.S courses at graduate level programme.

The foundation is focused on quickly adapt to developments in High Performance Computing discipline which include Parallel Computers Architecture, Parallel Processing, Distributed Computing, Tuning and Performance of Parallel Programs using different programming paradigms.

The course has been classified into three-tier level, focusing on various aspects of HPC and each tier module gives an overview of topics, which benefit the on-going project activities. The duration of each tier may suffice depends upon on-going project activities. The course curriculum is designed for Beta-testing group for *Testing and Benchmark large Parallel Computing Systems project activities.*

Each module may focus on theory and laboratory session as per requirements of project activities. However, some groups in C-DAC may require detailed contents of modules, pertaining to the on-going research projects. Suggested below are three ***short-term*** courses or preliminary/exhaustive training programme of HPC module offerings. Various parts of the courseware and details of modules are summarized below.

For Parallel Programming, all the users should start with material covered in Part –I. Logically the courseware work should flow from top to bottom as per the numbers indicated in shown in figure 1 and figure 2. For specific project work, it is necessary for the course participant to refer advanced books on Parallel Processing or visit important web sites.

2. An Overview of β-test-ParProc-TC -v1.0 Courseware

The **βtest-ParProc-TC** *v*1.0 courseware can be grouped into *ten* parts and each part has sufficient number of modules. Each module contents can be covered in classroom lectures and the Hands-on Session can be done on PARAM series of clusters or your own Parallel Computing systems.

Figure 1 illustrates the various parts of the courseware and Figure 2 illustrates the various modules of each part and its relations with other modules. A brief summary of various parts of the courseware and flow of several modules in each part are described in Figure 1 and Figure 2.

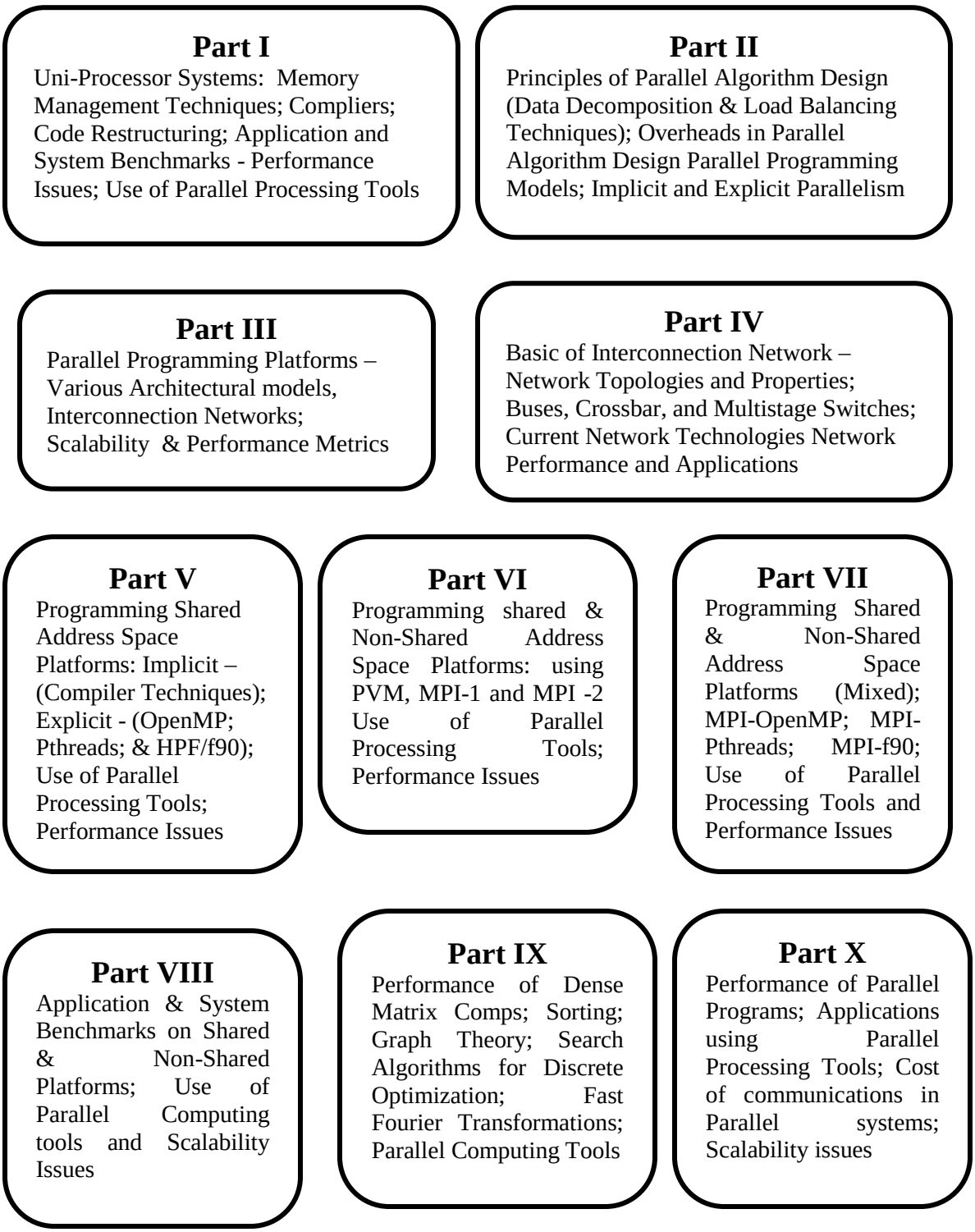

Figure 1. Representation of Courseware Contents

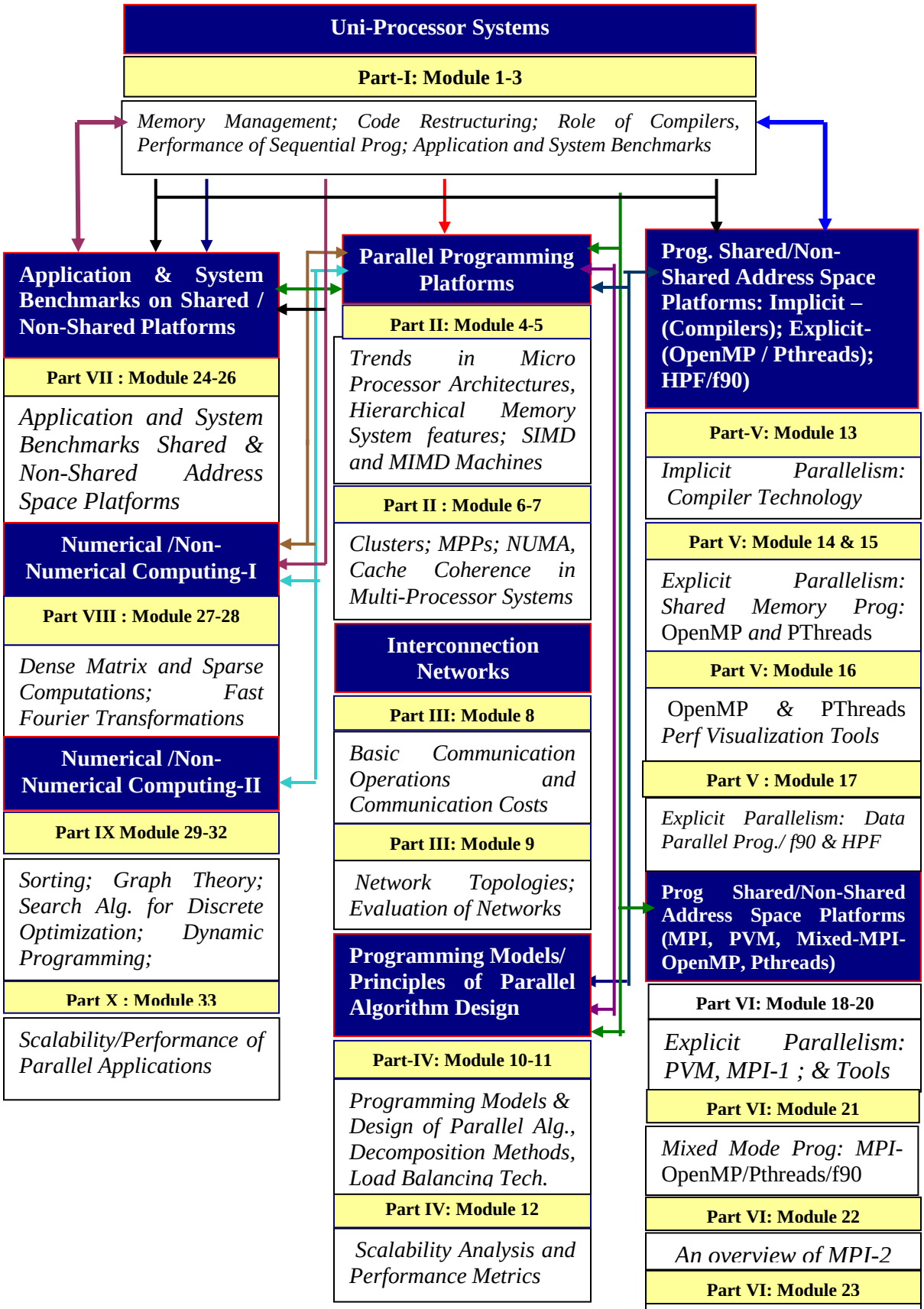

Figure 2. Flow of various modules

The courseware can be easily tailored to the developer, testing group, user community to extract performance of large scale applications and gives a strong foundation on programming models for Benchmarking computing systems in the range of *teraflop* to *petaflop* computing systems. The philosophy is to introduce new functionality and concepts to solve a design, implementation or analysis of problem in this courseware. To understand about the sorts of problems, beginners, developers encounter, when they begin thinking in parallel and writing parallel programs using different programming paradigms, the course contents may help.

To understand about the sorts of problems, beginners, developers encounter, when they begin thinking in parallel and writing parallel programs using different programming paradigms, the course contents may help and it is designed for short, mid and long-term duration of time schedule.

Most importantly, the courseware has sufficient number of programming assignments, which should play a central role to make strong foundations on Parallel Processing. Several modules defined in the courseware can be grouped together as per requirements of members who opt for short, mid, and long term time duration.

Figure 1 illustrates the various parts of the courseware and Figure 2 illustrates the various modules of each part and its relations with other modules. The solid arrow from Module A to Module B indicates Module B depends heavily upon material presented in Module A. For mid term and long-term courseware schedule, Module A and Module B have mutual relations, while performance of parallel programs are considered on target architecture.

All course contents given in each part is covered in numerical order, you will satisfy all requirements for Testing, Benchmarking, and Performance of large Parallel Computing Systems. However, you would like your members to start programming in C or Fortran with MPI or OpenMP as quickly as possible, you may wish to skip Part III or cover only two or three modules of the courseware. Definitely there is weak dependence across several modules of different parts and judicious choice should be taken up, which merely depends upon the duration of courseware i.e. Short, Mid or Long term.

If you wish to focus on performance of applications on *teraflop* to *petaflop* computing platforms, you may wish to skip Part III or cover only two or three modules, with more emphasis on Part IX Modules, focusing on parallel algorithms.

If you would like to start by having your members programming numerical computation algorithms with focus on performance, you can jump to Part X (Scalability and Performance) after covering important modules in remaining parts.

To get exposure to new functionality ‘just in time’ and performance of applications on *petaflop* computing systems, one can jump to various modules in Part VI & Part VII with strong foundation on Part III modules. A pre-requisite for this is to go through all modules as defined in short term course.

Most importantly, if you wish to get expertise on Benchmarks, programming various algorithms using MPI /OpenMP, you can work on Part VIII immediately after completion of necessary modules in other Parts.

3. Time Duration

The **short-term** course is focused on identifying suitable modules from Part-I to Part-X and quickly learns to write parallel programs with focus on algorithm design and performance. The emphasis is to design methodology to develop MPI/OpenMP programs that solve a series of progressively more difficult programming problems. Also, the focus for analyzing and predicting the performance of parallel systems is most important and it is addressed. The modules on architectural models, and important algorithms on numerical and non-numerical computations can be taken up as per requirements of user.

The **mid-term** course has more than enough material to write complex parallel programming, focusing on MPI, Pthreads, OpenMP and mixed MPI/OpenMP on different models of Parallel Computers. Special emphasis can be given to Application and System Benchmarks. Most importantly, use of Parallel Processing tools can be taken up which is required to know about the profiling a program, debugging the programs, visualization of programming behaviour.

The **long-term** course provides complete one semester course in parallel programming, focusing on turnkey projects. Even though the parallel programming is more demanding and it is hard to write programs, several algorithms for numerical and non-numerical computations can be taken up.

Suggested below is time duration for courseware program and equal weightage is given for the theory and hand-on on Parallel Computing systems.

- Tier - 1: Short term course and the time duration is 30 Calendar days
- Tier - 2: Mid term course and the time duration is 45 Calendar days.
- Tier - 3: Long term course and the time duration are 60 Calendar days.

Most importantly, the courseware has sufficient number of programming assignments, which should play a central role to make strong foundations on Parallel Processing.

.

4. Description of course contents in Modules

Part-I: Programming on Uni-Processsor Systems

Module Names : Module 1(Prg-UProc-1); Module 2(Prg-UProc-2); Module 3(Prg-UProc-3)
Project Type : Short term
Time Duration : 10 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand theoretical concepts; Programming exercises; Deliver lectures; Examination on theory; Demonstrate the results

Module 1 (Prg-UProc-1) — Calendar Days :4

Tuning & Performance of Sequential Programs: Performance Analysis, Profile the program and use of Parallel Processing tools, Tuning of Sequential Programs with the Compiler Options; Understand the behaviors of program performance without /with Compiler options

Module 2 (Prg-UProc-2) — Calendar Days :4

Tuning & Performance of Sequential Programs: Improving Effective Memory Latency using Caches; Impact of Memory Bandwidth, Multi-threading and Prefetchting Understand Memory Management Concepts, Understand Hierarchical Memory Feature - Data Locality; Code Restructuring Techniques: Performance Analysis &Tuning of Sequential Programs on Uni-Processor Systems, Use of System Provided Libraries,

Module 3 (Prg-UProc-3) — Calendar Days :4

Application and System Benchmarks: Performance of Application and System Benchmarks on Uni - Processor systems (STREAM, LLBENCH, LMBENCH, SPEC, LINPACK, NAS and other Benchmarks); Conclusions on Peak and Sustained Performance of Computing Systems for Benchmarks

Part-II: An Overview of SIMD/MIMD Parallel Programming Platforms

Module Details : Module-4(PPModel-1); Module-5(PPModel-2); Module-6(PPModel-3); Module-7(PPModel-4)
Project type : Short or Mid or Long Term
Time Duration : 10 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand theoretical concepts; Deliver lectures; Examination on theory; Group Discussion

Module 4 (PPModel-1) — Calendar Days: 2

An Overview of SIMD/ MIMD Parallel Programming Platforms: Trends in Micro Processor Architecture; Hierarchical memory features; An overview of Single-Instruction Multiple-data (SIMD); Multiple -Instruction Multiple-data (MIMD); Parallel Vector Processor (PVP) Systems

Module 5 (PPModel-2) Calendar Days: 2

An Overview of SIMD/MIMD Parallel Programming Platforms: Overview of Massively Parallel Processor (MPP); Machines; Symmetric Multiprocessor (SMP) and the Distributed Shared memory (DSM) Machines; Interconnect Network s and Performance Issues.

Module 6 (PPModel-3) Calendar Days: 2

An Overview of UMA, NUMA & CC-NUMA Architectural Models: Cache Coherence in Multi-Processor Systems; Scalability and Performance Issues.

Module 7 (PPModel-4) Calendar Days : 4

An Overview of Clusters: Cluster of SMP Servers, Challenges in Clusters, Classification, Cluster Architecture, Cluster Design Issues; Support of Single System Image, Job Management in Clusters; Cluster Communication Issues. Comparative Study of various MIMD Machines

Part-III: System Interconnection Networks (Network Topologies and Performance Issues)

Module Details : Module 8 (PPComm-1); Module 9 (PPINet-1)
Project type : Short or Mid or Long term
Time Duration : 4 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand theoretical concepts; Deliver lectures; Examination on theory; Group Discussion

Module 8 (PPComm-1) Calendar Days: 2

Basic Communication Operations Performance Issues & Algorithms: One-to-All Broadcast and All-to-One Reduction, All-to-All Broadcast and Reduction, All-to-All Personalized Communications, Performance Issues and Estimation of Communication Overheads.

Module 9 (PPINet-1) Calendar Days: 2

System Interconnections and Gigabit Networks: Basics of Network Topologies and Properties, Buses, Crossbar and Multistage Networks, Gigabit Network Technologies, Scalable Coherence Interface, Fast and Efficient communication algorithms, Key Problems in Network Communication & Performance

Part-IV: Parallel Programming Paradigms/Models Principles of Parallel Algorithm Design

Module Details : Module 10 (PrgPardm-1); Module 11(ParAlgr-1); Module 12 (PerfMetrics-1)
Project type : Short or Mid or Long term
Time Duration : 5 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand theoretical concepts; Deliver lectures; Examination on theory; Group Discussion

Module 10 (PrgPardm-1) Calendar Days: 3

Parallel Programming Paradigms and Models: Algorithmic Paradigms or Models (The Data-parallel Model, the Task Graph Model, the Work Pool Method; The Master-Slave Model; The Pipeline Model, Hybrid Models, Programmability Issues, Parallel Programming Models (Implicit, Explicit and other Parallel Programming Models)

Module 11(ParAlgr-1) Calendar Days :2

Principles of Parallel Algorithm Design: Decomposition, Tasks, Granularity, Concurrency, Decomposition Techniques (Recursive, Data, Exploratory, Speculative, Hybrid), Characteristics of Tasks and Interaction, Mapping techniques for Load Balancing; Interaction Overheads Estimation: Static and Dynamic Load Balancing, Interaction Overheads for different Algorithms)

Module 12 (PerfMetrics-1) Calendar Days: 2

Performance Metrics and Analytical Modeling of Parallel Programs: Basic Performance Metrics, Performance versus Cost, Speedup, Efficiency, Performance and Scalability of Parallel Computers, Performance of Parallel Programs, Scalability and Speedup Analysis, Sources of Overheads in Parallel Programs; The Effect of Granularity on Performance

Part-V: Programming Shared Address Space Platforms Implicit (Compilers) / Explicit (OpenMP;Pthreads; f90/HPF)

Module Details : Module 13(Prg-ImpComp-1); Module 14(Prg-ExThrd-1); Module 15(Prg-Ex-Opmp-1); Module 16(Prg-Ex-OpMP-2); Module 17(Prg-Datapar-1);
Project type : Short or Mid or Long term
Time Duration : 10 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Programming concepts; Examination on theory; Deliver lectures; Programming Exercises; Performance of Parallel Programs Demonstrate the results

Module 13 (Prg-ImpComp-1) Calendar Days :2

Implicit Parallelism: Automatic compiler Technology: Parallelizing Compilers; Effectiveness of Parallelizing compilers; Improving Effective Memory Latency using Caches; User Direction, Run-time Parallelization, Performance issues of Benchmarks

Module 14 (Prg-ExPThr-1) Calendar Days: 4

Explicit Parallelism: Shared Memory Programming: Pthreads: Parallel Programming-using The POSIX Threads (Pthreads) Model; Threads Synchronization; Tuning; Performance of Programs using Pthreads

Module 15 (Prg-ExOpmp-1) Calendar Days: 4

Explicit Parallelism: Shared Memory Programming: The Shared Memory Model, OpenMP: Parallel Programming-using OpenMP; Fork/Join Parallelism; Parallel `for` Loops, Parallel Programs using OpenMP Parallel directives

Module 16 (Prg-ExOpmp-2) Calendar Days: 3

Explicit Parallelism: Shared Memory Programming: Tuning/Performance improvements; Data Parallelism; Benchmarks; Profile the Program and Tools for Pthreads / OpenMP: Programs; functional Parallelism (Parallel Section); Critical Sections; Increment Parallelization; Comparison with Pthreads

Module 17(Prg-Datapar-1) Calendar Days :3

Explicit Parallelism: Data Parallel Programming: f90/HPF: Sequential /Parallel Programming using f90 & HPF; Profile the program- Opportunities for Parallelization; Tuning & Performance Issues

Part-VI:
Programming Shared /Non-Shared Address Space Platforms /Explicit (PVM, MPI-1, MPI-2, Mixed – MPI/OpenMP/Pthreads/f90)

Modules : Module 18 (Prg-ExMPIPVM-1); Module 19 (Prg-ExMPI-Tools1); Module 20 (Prg-ExMixedMPI-1); Module 21 (Prg-ExMPI2.0 –1); Module 22 (Prg-ExMPIBench-1)

Project type : Short or Mid or Long Term

Time Duration : 20 Calendar Days (Overlap time schedule with other modules)

Nature of Work : Understand Programming concepts; Examination on theory; Deliver lectures; Programming Exercises; Performance of Parallel Programs Demonstrate the results

Module 18(Prg-ExMPIPVM-1) Calendar Days: 4

Explicit Parallelism: Message Passing Programming PVM – Parallel Programming-using PVM; Write parallel programs for dense Matrix Computations

Module 19(Prg-ExMPI1.0 -1) Calendar Days: 4

Explicit Parallelism: Message Passing Programming – MPI 1.0: Parallel Programming using MPI 1.0 (Point-to-Point; Collective Communication, Collective communication &Computation; Advanced Point-Point Library calls) the measurement of Communication overheads; Performance Issues

Module 20 (Prg-ExMPITools-1) Calendar Days: 4

Explicit Parallelism: Message Passing Programming Tools – MPI 1.0; Parallel Programming using MPI-1 and Use of Performance Visualization tools; Use of Parallel Debuggers

Module 21 (Prg-ExMixedMPI-1) Calendar Days: 4

Explicit Parallelism: Mixed Mode Prog - MPI-OpenMP; MPI-Pthreads; MPI-f90: Parallel Programming-using MPI-Pthreads; MPI-OpenMP & MPI-f90 / MPI-HPF; Execution of Benchmarks; Profile the Program; Write programs for Numerical Computational problems; Execute on Clusters; Deliver Lectures

Module 22(Prg-ExMPI2.0 –1) Calendar Days: 4

Explicit Parallelism: Message Passing Programming – MPI 2.0: Parallel Programming using MPI 2.0 (Dynamic Process; One-Sided Communication; Non-blocking Collective communications); MPI-2 I/O features; understand the measurement of communication overheads: Performance Issues

Module 23(Prg-ExMPIBench-1) Calendar Days: 4

Explicit Parallelism: Message Passing Programming – MPI Benchmarks: Measure the Communication Overhead Measurement Time; Quantification of Overheads; Execute Benchmarks (MPPBENCH, P-COMS, PG-COMS; PALLAS, LLCBench, HLRS, MPIFORUM)

Part-VII: Application and System Benchmarks on Shared & Non-Shared Address space Platforms

Module Details : Module 24 (PP-Microbench-1), Module 25 (PP-Macrobench-1); Module-26 (PP-HPCCBench-1)
Project type : Short or Mid or Long Term
Time Duration : 10 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Programming concepts; Examination on theory; Deliver lectures; Programming Exercises; Performance of Parallel Programs; Demonstrate the results

Module 24 (PP-MicroBench-1) Calendar Days: 4

Micro Benchmarks: Application Kernels, Synthetic programs, I/O Speed; Memory Speed, Operating System Performance, networking (BLAS, BLACS, LINPACK, LMBENCH, STREAM, TOP-500; (P-COMS, PALLAS, MPPENCH, LLCBENCH, MPIFORUM; HLRS Benchmarks)

Module 25 (PP-MacroBench-1) Calendar Days: 4

Macro Benchmarks: Full-fledged applications or Kernels, (ParkBench; NAS; TPC Benchmarks, Third Party Applications)

Module 26 (PP-HPCCBench-1) Calendar Days: 4

HPC Challenge Benchmarks: Performance of TOP-500 and other HPCC Benchmarks Execution on commodity Clusters and PARAM Series

Part-VIII: Parallel Algorithms and Implementation: Numerical and Non-Numerical Computing –I: (Dense Matrix and Sparse Matrix computations Fast Fourier Transformations)

Module Details : Module27 (PNum-Matrix-1); Module28 (PNum-FFF-1);
Project type : Short or Mid or Long Term
Time Duration :10 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Performance of Program/ Programming Exercises/Deliver Lectures and Demonstrate Performance of Algorithms on Clusters

Module 27: (PNum-Matrix-1) Calendar Days: 6

Vector- Vector Multiplication; Matrix-Vector Multiplication; Matrix –Matrix Multiplication; Solving System of Linear Equations (Direct and Iterative Methods) – row wise partitioning; column-wise partitioning; Checker Board and Block cyclic partitioning, One- & Two dimensional Partitioning, Uniform and Non-Uniform Partitioning – MPI based Freely available BLAS libraries Execution; ScaLAPACK, LINPACK and other matrix libraries using different programming paradigms.

Module 28 (PNum-FFT-1) Calendar Days :4

An overview of serial Algorithm; The Binary-Exchange Algorithm (A Full Bandwidth Network; Limited Bandwidth Network); Computations in FFT; The Transpose Algorithm (The two dimensional and Generalized Transpose Algorithm); Understand & Execution of Freely downloadable Benchmarks on FFTs

Part-VIII: Parallel Algorithms and Implementation: Numerical and Non-Numerical Computing –II: Sorting; Graph Theory; Search Algorithms Discrete Optimization; Dynamic Programming)

Module Details : Module 29 (PNonNum-Sort-1); Module 30 (PNonNum-Graph-1); Module 31 (PPNonNum-Dsearch-1); Module 32 –PNonNum-DynProg-1)
Project type : Short or Mid or Long Term
Time Duration : 15 Calendar Days (Overlap time schedule with other modules)
NNature of Work : Performance of Programs/ Programming Exercises/Deliver Lectures and Demonstrate Performance of Algorithms on Clusters

Module 29 (PNonNum-sort-1) Calendar Days: 3

Issues in Sorting on Parallel Computers; Sorting Networks; Bitonic Sort; Bubble Sort; Quick Sort; Bucket and Sample Sort Algorithms & Implementation Performance Issues; *Understand & Execution of Freely downloadable Benchmarks on Sorting Algorithms*

Module 30 (PNonNum-Graph-1) Calendar Days :4

Graph Theory- Minimum Spanning Tree Algorithm; Single Source Shortest Paths: Dijkstra's Algorithm; All-Paris Shortest Paths; Search Based Algorithms; Algorithms for Sparse Graphs

Module 31(PNonNum-Search-1) Calendar Days: 4

Sequential search Algorithms (Depth-First Search & Best-First Search Algorithms); Search Overhead Factors; Parallel Depth First Search and Best-First Search Algorithms; Speedup Anomalies in Parallel Search Algorithm; Performance issues on shared and non-shared memory systems

Module 32 (PNonNum-DynProg-1) Calendar Days: 4

An overview of Dynamic programming; The Shortest path Problem); Parallel DP formulation; Source of Parallelism; Performance issues

Part-X:
Performance and Scalability of Applications

Module Details : Module 33 (PerfScalApp-1)
Project type : Short or Mid or Long Term
Time Duration : 4 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Performance of Programs/ Programming Exercises/Deliver Lectures and Demonstrate Performance of Algorithms on Clusters

Module 33 (PerfScalApp-1) Calendar Days :4

Integrate the application with important third party tools, which gives sustained performance of applications; Performance issues of I/O features; Direct and Iterative Solvers; Algorithm Scalability; Estimation of Communication overheads with respect to application. Understand & Execution of freely downloadable NAS Benchmarks)

Part I:
Module 1(Prg-UProc01): Tuning and Performance on Uni-Processor Systems: Optimizing with the Compilers (POWER4 & POWER5)

Module-1: Tuning and Performance of Sequential Programs on Uni-Processor Systems using Compiler Optimization Techniques

Duration-Part I :10 days **Module-1: 4** Calendar Days (Overlap time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Programming exercises; Deliver lectures; Examination; Demonstrate the results; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I	
Activities **1.** Understand target machine (POWER4/POWER5) specific Compiler options; Special Options for POWER4/POWER5 XL Fortran Intrinsic vector; **2.** Get the Object Code listing (The pseudo Assembler Generation IBM POWER4/POWER5)	**Deliverables** – Performance examples using various Compiler options – Object Code: Study & understand impact of compiler options used; identify potential problems with the compiler – Deliver Lecture on Phase-I
PHASE-II	
3. XL Fortran (IBM POWER4 & POWER5) compiler directives for tuning (Prefetch Directives; Loop-related directives, Cache and other directives) • Option Flags: –O,-O2, -O3, -O4, -O5 • Study performance – increase levels of optmisaiton • Options for –O4: -.qhot, -qipa, -O3, -qarch, -qtune=auto, - qcache=auto; **4.** For given numerical intensive application with different size of the problem: study which complier option gives better performance and analyze the performance results	– Analyze the performance of Fortran or C Compute intensive programs with different compiler options. – Execute programs with various compiler options on different machines and observe the performance. Deliver lecture on Phase-II – Group Discussion on Selective topics of Module -1
PHASE-III	
5. XL Fortran compiler directives for tuning (Prefetch Directives; Loop Oriented Directives; Cache and Other directives; (IBM POWER4 & POWER5) **6.** Basic Coding practices for performance; Language-independent tips; Fortran tips; C &C++ tips; Inlining procedure references; Structuring code for optimal grouping **7.** Tuning 64-Bit integer Performance **8.** Complier options for numerically intensive applications IBM POWER4 & POWER5 – Using MASS and ESSL – DGEMM libraries; Performance of various fundamental loops (BLAS-1,II, III) using compiler options for various problem size **9.** Comparing C and Fortran Complier Code Generation, for Numeric and Non-Numeric intensive codes – Understand flow of the code – in terms of memory access; Data Re-use; Cache & Memory latency measurement; examples on Structured/Unstructured Data	– Write programs that improve performance of codes using compiler directives for tuning. – Develop Concrete examples for Performance using Libraries (DGEMM, BLAS-I, II, III) – Deliver lectures on Selective topics of Module 1.

Part I:
Module 2(Prg-UProc-2): Tuning and Performance on Uni-Processor Systems: Code Restructuring Techniques (POWER4 & POWER5)

Module 2 : Tuning and Performance on Uni-Processor Systems –Code Restructuring Techniques

Duration-Part I :10 days **Module-2:** **4** Calendar Days (Overlap time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:**Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Programming exercises; Deliver lectures; Examination; Demonstrate the results; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	**Deliverables**
PHASE-I	
1. *Understand target machine (POWER4/POWER5) specific features:* The Level 1, 2,and 3 caches & TLB; Understand cache lines & set associativity –Prefetch data streaming; Summary on Floating Point Units; Understand Stride Minimization & Write programs; Programs on data prefetch streaming; Programs on Structuring for L1 Set- Associativity; and Data Cache blocking	– Performance examples using various Compiler options – Object Code: Study & understand impact of compiler options used; identify potential problems with the compiler
PHASE-II	
2. *Loop Unrolling*: Unrolling inner loops; Unrolling outer loops to increase the ratio of computation to load and store instruction, Performance computation and data movement; Basic Loop unrolling; Qualifying Candidates for Unrolling (Loops with low trip counts; Fat Loops; Loops Containing Procedure Calls; Loops with Branches in them; Recursive Loops); Loop Interchange **3.** Tuning Loops: The size of do OR for loop manageable; Access data Sequentially with unit stride; Minimize loop invariant IF statements in loops; Avoid subroutine or function calls in loops; Use Simple array subscripts; Use INTEGER loop variables;	– Analyze the performance of Compute intensive programs with /without Loop Unrolling techniques. – Performance of programs with various compiler options – Summary on Classical Optimizations – What an Optimizing compiler does – Example programs on Loop optimization
PHASE-III	
4. Memory Reference Optimizations: Memory access Patterns; Loop interchange to Ease Memory Access Patterns; Blocking to Ease Memory Access Patterns; Tuning for L2 Cache access **5.** Tuning for I/O: Understand I/O subsystem performance Characteristics; Advantage of Asynchronous I/O; Direct I/O (Synchronous I/O); Paging I/O; buffered I/O **6.** Use Profiling tools –CPU time used; Locating hot spots for performance - Use IBM AIX POWER4/POWER5: - *prof, gprof, tprof & Xprofiler*	– Example programs on Memory reference Optimization – Example programs - how the CPU time used by a program during execution is distributed over the code; Use Profiling tools: *prof, gprof, tprof & Xprofiler*
PHASE-IV	
7. Use System tuned Performance libraries: Use `netlib` provided BLAS-I, II &III; Use `ESSL` and `Parallel ESSL` routines for matrix–matrix multiplication with varying problem sizes; Performance examples using `ESSL`; **8.** Programs using The Mathematical acceleration subsystem (`MASS`) library and IBM James Watson Sparse Matrix Package (`WSMP`) & `SPARSE KIT`	– Example programs using system tuned libraries

Part I:
Module 3(Prg-Uproc-3): Application & System Benchmarks on Uni-Proc. Systems

Module 3 : Application and System Benchmarks (Micro & Macro) on Uni-Processor Systems

Duration-Part I:10 days **Module-3: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:**Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Programming exercises; Deliver lectures; Examination; Demonstrate the results; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I

Activities	Deliverables
1. **Set-UP Testing Environment**: Understand Testing-programming environment of a Cluster – Compiler Options; Set-UP production runtime environment of Cluster 2. **Benchmark Run Rules**: Define the test run-rules (Default baseline optimizations) for Cluster configuration with/without network specific optimizations & the specific rules governing the test conditions and procedures – including testing environment.	– Write down summary of testing environment and Run-Rules for Benchmarks

PHASE-II

3. **Study & Understand Micro Benchmarks:** Micro Benchmarks measure a specific aspect of a computer system, such as CPU speed, memory speed, I/O speed, Operating System Performance, networking etc. Micro Benchmarks tend to be synthetic kernels or applications or just kernels. 4. **Micro Benchmarks -I** • STREAM: Memory Bandwidth; • LMBENCH: System Calls and Data Movement operations in Unix/Linux Systems; • LLCBench : Low level Cache Benchmarks • Execution of BLAS-I, II and III libraries • DGEMM: Numerical computing (Linear algebra) – Dense Matrix into Matrix Multiplication • Execution of PARKBENCH (Matrix Computaiton) suites • Execution of LINPACK (TOP-500 Supercomputers): Numerical computing (Linear Algebra) Capabilities	– Analyze the performance of Fortran or C Compute intensive programs with different compiler options. – Execute programs with various compiler options on different machines and observe the performance. – For given numerical intensive application with different size of the problem: Obtain results, in which complier option gives better performance and analyze the performance results

PHASE-III

6. **Macro Benchmarks**: • Execution of NAS: The NPB Suite – The Numerical Aerodynamic Simulation (NAS) program at NASA Ames Research Centre developed the The NAS Parallel Benchmarks (NPB). • Execution of SPEC – Measure CPU performance, but ha extended to Client/Server Computing, commercial applications; I/O sub-systems; GPC –Graphics Performance Characterization-	– Performance (Gflop/s) for selective NAS Benchmarks

Part II : Module 4 (PPModel-1) SIMD /MIMD Parallel Programming Platforms	
Module 4 : An overview of SIMD/MIMD Parallel Programming Platforms & Architectural Models	
Duration-Part II :10 days **Module-4: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Systems; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** Trends in Micro Processor Architectures (Pipeling and SuperScalar Execution; VLIW Processors); **2.** Limitations of Memory System Performance (Improving Effective Memory Latency Using Caches; Hiding Memory Latency; Features of the memory sub-system hierarchy; Managing Memory overheads; Multithreading & Prefetching **3.** An overview of Microprocessor Architecture Families **4.** Dichotomy of Parallel Computing Platforms; Understand abstract machine models of Parallel Computing Systems; **5.** Classification of Physical Machine Models: (Single-Instruction Multiple-data (SIMD); Multiple Instruction Multiple-data (MIMD) **6.** Advances in Hardware and Software- Parallel Computing	**Deliverables** – Understand Parallel Computing Architecture and deliver lectures – Discussion and Deliver lectures on various topics with other members – Study and deliver lectures on Cluster Computers; – Study and deliver lectures on Cluster Interconnection Networks

Part II: Module 5 (PPModel-2)- SIMD /MIMD Parallel Programming Platforms	
Module 5 : An overview of SIMD/MIMD Parallel Programming Platforms & Architectural Models – Shared Memory Computing Systems and CC-NUMA Computing Systems	
Duration-Part II:10 days **Module-5: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Systems; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** An overview of Shared Memory Systems (SMPs): Single Address Space; Single System Image; Symmetry Features **2.** (SMPS): Features of Cache; Memory communication; Effective Memory Bandwidth; Latency; I/O Bandwidth; **3.** (SMPs): Scalability and Performance issues **4.** Cache Coherent: Non-Uniform Memory systems (CC-NUMA): An overview of distributed shared memory architecture; Architectural advantages; **5.** CC-NUMA: Advantages of CC-NUMA over SMPs; Performance Issues; Memory Bus Features; **6.** CC-NUMA: Scalability - Memory Capacity and I/O Capabilities	**Deliverables** – Understand Parallel Computing Architecture and deliver lectures – Group Discussion on an overview of SMPs – Deliver lectures on SMPS/CC-NUMA – Scalability Issues of SMPs /CC-NUMA and Performance Issues – Concrete examples on Performance on SMPs/CC-NUMA machines

Part II:
Module 6 (PPModel-3)- SIMD /MIMD Parallel Programming Platforms

Module 6 : An overview of SIMD/MIMD Parallel Programming Platforms & Architectural Models – MPPs ; DSM and PVP Computing Systems

Duration-Part II :10 days **Module-6: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:**Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Systems; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. Massively Parallel Processors (MPPS): An Overview of Micro processors used in MPPs **2.** MPPs : Distributed Memory over Processing Nodes; Interconnect: Low Latency and High Bandwidth **3.** MPPs: Current Trends: High Speed Memory bus; Local Memory bus; Network interface Circuitry (NIC); Fast Communication – tightly coupled nodes **4.** Distributed Shared Memory (DSM) Machines: An Overview; Single Address Space; Hardware and Software support of Single Address Space; **5.** Parallel Vector Processors (PVPs): An overview of powerful vector processors (VPs); concepts of vector registers; Performance issues of vector processors; PVP's –Interconnection networks; PVPs; Shared Memory	– Presentation on PVPs for one hour – Group Discussion on MPPs – Concrete Case Study on DSM – Deliver lectures on various models of Parallel Computing Systems – Concrete examples on Dense Matrix Computations & Performance on PVPs; MPPs – Scalability Issues of DSM and Performance Issues

Part II :
Module 7 (PPModel-4) SIMD /MIMD Parallel Programming Platforms

Module 7 : An overview of SIMD/MIMD Parallel Programming Platforms & Architectural Models – Cluster Computing Systems

Duration-Part II :10 days **Module-7: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:**Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Systems; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. Classification of Clusters; Cluster Architecture; Cluster Design Issues; Cluster Interconnect; An overview of Commodity off-the-shelf (COTS) Clusters; 2. Availability of Support for Clustering; Support for Single System Image; Global File System; Job Management in Clusters 3. Cluster: Fast Communication; Cost of Communication; Communication Performance (VIA); High-speed interconnects; I/O Issues; TCP/IP; UDP/IP traditional protocols; NIC Architectural support for Clusters **4.** Cluster Scalability and Cluster I/O Issues; Cluster Performance; An Overview of Cluster of SMP Servers	– Presentation on Cluster Computing for one hour – Group Discussion on Cluster computing – Concrete Case Study on Commercial Off-the Shelf Clusters – Deliver Talks and lectures on Cluster Interconnection Networks – Concrete examples on Dense Matrix Computations & Performance – Scalability Issues of Clusters

Part III: Module 8 (PPComm-1): Parallel Prog. Platforms: Basic Communication Operations	
Module 8: An Overview of Basic Communication Operations on Parallel Computing Systems	
Duration-Part III :4 days **Module-8: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Module; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** An Overview of topologies: Ring or Linear Array; Mesh, Hypercube; Balanced binary Tree; Detailed Algorithms; Cost analysis **2.** One-to-all Broadcast and All-to-One Reduction **3.** All-to-All Broadcast and Reduction; All-Reduce and Prefix-Sum Operations; All-to-All Personalized Communication; **4.** Improving the Speed of Communication Operations (Splitting and Routing Messages in Parts; All-Port Communication); Circular Shift Operations	**Deliverables** – Deliver lecture on Basic Communication Group Discussion on Speed of Communication Operations – Concrete Case Study on All-to-All Personalized Communication Operations on Mesh topology - Parallel Computing Systems – Performance of All-to-All communication operations on Mesh and Hypercube topology -Parallel computing systems

Part III: Module 9 (PPINet-1) - Parallel Computing Platforms: Interconnection networks	
Module 9 : An Overview of Interconnection networks on Parallel Computing Systems	
Duration-Part III :4 days **Module-9: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Module; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** Interconnection networks for Parallel Computers; Basics of networks; Network Topologies **2.** Evaluating Static Interconnection networks **3.** Evaluating Dynamic Interconnection networks **4.** Cache Coherence in Multiprocessor Systems, **5.** Routing Mechanisms for Interconnection Networks, **6.** Communication Costs in Parallel Machines**;** The `LogP` Communication Model **7.** An Overview of Active Messages and Fast Messages **8.** NIC Architectural Support of Operations - Algorithms	**Deliverables** – Deliver lecture on Evaluating networks for Parallel computers – Group Discussion on `LogP` Communication Model - Clusters – CCase Study on Cache Coherence in Multiprocessor Systems – Deliver lectures on NIC Architectural support of operations – Deliver lectures on an overview of Active Messages

Part IV: Module 10 (PrgPardm-1) -Parallel Programming Models & Parallel Paradigms	
Module 10: An Overview of Parallel Programming Models (Implicit & Explicit); Programmability Issues; Parallel Paradigms	
Duration-Part III :5 days **Module-10:3** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:**Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Module; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** An overview of Algorithmic Paradigms; **2.** Programmability Issues; Examples of Parallel Programs **3.** Parallel Programming Models: Implicit Parallelism; Parallel Compilers; Effectiveness of Parallel Compilers **4.** Explicit Parallel Models: Data Parallel; Message Passing; and Shared Variable **5.** An overview of Shared Memory Programming; An overview of different Programming Models (Functional; Logic, Object –oriented Programming; **6.** Comparison of various Parallel Programming Models	**Deliverables** – Understand Parallel Computing Architecture and deliver lectures – Discussion and Deliver lectures on various topics with other members – Develop Concrete examples and counter examples of different programming models – Indicate main limitations of Implicit Parallelism; develop concrete examples on limitations of parallel compilers

Part IV:
Module 11 (ParAlgr-1) - Principles of Parallel Algorithm Design

Module 11: An overview of Parallel Algorithm Design – Decomposition techniques; Load Balancing techniques; Algorithm Overheads estimation

Duration-Part IV :5 days **Module-11: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Module; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. Preliminaries: Decomposition Tasks and Decomposition, Tasks; Granularity; Concurrency and Task Interaction; Seeking Concurrency (Data Dependence Graphs; Data; functional and Pipeline Parallelism; Size considerations); Process and Mapping; Processes and Processors; **2.** Decomposition Techniques (Recursive Decomposition; Data Decomposition; Exploratory Decomposition; Speculative Decomposition; Hybrid Decomposition) **3.** Characteristics of Tasks and Interactions; Mapping techniques for Load Balancing (Schemes for Static and dynamic Load Balancing); Agglomeration Process -Tasks **4.** Methods for Containing Interaction Overheads (Maximizing Data Locality; Minimizing Contention and Hot Spots; Using Optimized collective interaction operations; Overlapping Comps. with Interactions; Quality of an Agglomeration **5.** Parallel Algorithmic Models (The data Parallel; The Task Graph; The Work Pool Model; The Master-Slave Model; The Pipeline Model; Hybrid Models)	– Understand Parallel Computing Architecture and deliver lectures – Discussion and Deliver lectures on various topics with other members – Develop Concrete examples and counter examples for different programming models – Indicate main limitations of Implicit Parallelism – Develop concrete examples on limitations of Implicit Parallelism – Examples on Dense Matrix computations to find degree of Concurrency – Develop concrete examples to find out algorithm overheads

Part V:
Module 12 (PerfMetrics-1)- Performance Metrics and Scalability Analysis

Module 12: Performance Metrics, Theoretical Speed-Up and Scalability Analysis

Duration-Part V:10 days **Module-12: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Discussions on an overview of Module; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. Performance versus Cost (Execution Time and Throughput; Utilization and Cost Effectiveness); Basic Performance Metrics (workload and Speed Metrics); 2. Performance of Parallel Computers (Communication Characteristics; Parallelism and Interaction Overheads; Overhead Quantification) 3. Performance Metrics: Execution time; Total Parallel Overhead, Speedup, Efficiency, and Cost 4. Source of Overhead in Parallel Programs; Scalability and Speed-Up Analysis (Amdahl's Law: Fixed Problem Size; Super Linear Speed-Up; Gustafson's Law Fixed Time; Sun and Ni's Law: Memory Bounding) 5. Scalability of Parallel Systems (The degree of concurrency, Isoefficency, Isoperformance Models)	– Deliver lectures on Performance metrics and Overheads on Parallel Computers – Concrete examples on Speed-Up Analysis (Amdahl' law) – Concrete programming examples on evaluation of Speed-up, Efficiency with respect to Amdahl' law, Gustafson's Law, Sun-Ni's Law – Deliver lectures on Speed-Up Metrics, focus on Measurement tools which can give performance in terms of Mflop/s

Part V:
Module 13 (Prg-ImpComp-1)-Implicit Parallelism: Parallel (Auto) Compilers

Module 13: Parallelization of sequential program using compiler technology on Shared memory Systems and Performance Analysis

Duration-Part V :10 days **Module-13: 2** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. Understand target machine (POWER4/POWER5-SMP machine) specific Auto Compiler options 2. Profile the selected sequential programs using Profile tools and identify parallelism 3. Identify optimizing techniques for eliminating dependency and use compiler directives for parallelization 4. Use Auto-compiler techniques on SMP machine for Dense Matrix Computation Codes 5. Improve the performance of codes using compiler directives; and code re-structuring techniques	– Performance examples using various Compiler options – Analyze the performance of Compute intensive programs – Demonstrate programs with various compiler options and show the performance – Write concrete example programs with `for` Loops and use compiler directives

Part V: Module 14 (Prg-ExPThr-1) Explicit Parallelism: Shared Memory Prog – Pthreads	
Module 14: An Overview of Shared Memory Programming –Pthreads Model; Performance of parallel programs using Pthreads	
Duration-Part V :10 days **Module-14: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** An overview of Shared Memory Programming **2.** The POSIX Threads (Pthreads) Model **3.** PThread Interaction; shared /private attributes **4.** PThread Management and Ptheads Synchronization & PThread Critical Sections **5.** Pthreads : Cancellation; composite Synchronization Constructs- Read-write Locks and Barrier **6.** Profile the selected sequential programs using Profile tools and identify parallelism	**Deliverables** – Deliver lecture on Shared Memory Programming - Pthreads – Demonstrate a parallel program on dense numerical computations using Pthreads library calls. – Concrete Case Study on performance of selective parallel programs using Pthreads library calls

Part V : Module 15 – (PrgExOpmp-1) Explicit Parallelism: Shared Memory Programming: OpenMP	
Module 15: An Overview of Shared Memory Programming –OpenMP Model; Performance of parallel programs using OpenMP Prgma's	
Duration-Part V:10 days **Module-15: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score: 100** **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities **1.** An Overview of OpenMP Progamming Model: Concepts & Parallel Directives **2.** OpenMP Library functions; Compilation, Inter Operability **3.** OpenMP Parallel Constructs and Regions Parallel `for` Loops **4.** OpenMP Parallel Do directives; Data Handling in OpenMP **5.** Specification of OpenMP library functions and OpenMP – functional Parallelism **6.** Performance of Sequential programs using OpenMP Directives (Shared, Private variables)	**Deliverables** – Presentation on Shared Memory Programming: OpenMP – Demonstrate performance of parallel programs on dense numerical computations using OpenMP pragmas – Summary of Performance improvements using OpenMP pragmas – Example program to write set of all primes less than equal to integer N (N > 1) (Use sieve of Eratosthenes method and Nest Procedure)

Part V:
Module 16 – (PrgExOpmp-2) Explicit Parallelism: Shared Memory Programming: OpenMP/Pthreads – Tools and Performance

Module 16 : An Overview of Shared memory programming: OpenMP and Pthreads – Tools and Performance of examples on dense numerical computations

Duration-Part V:10 days **Module-16: 3** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. OpenMP - Tasks Specification; Synchronization Constructs; Critical Section **2.** OpenMP : Profile the selected sequential programs using Profile tools and identify parallelism **3.** OpenMP – Data Parallelism **7.** OpenMP : Profile the selected sequential programs using tools and identify parallelism **8.** OpenMP : Benchmarks **4.** Pthreads : Profile the selected sequential programs using tools and identify parallelism **5.** Comparison of OpenMP and Pthead Programming model (performance Issues)	– Concrete Case Study on performance of selective parallel programs – Explicit Threads versus OpenMP Based Programming – Write programs using OpenMP pragmas for matrix into matrix computations – Profile the sequential program on dense matrix computations and get performance using OpenMP prgmas

Part V:
Module 17 – (Prg-ExDatapar-1)
Data Parallel Programming Model: f90 & HPF

Module 17: An Overview of Data Parallel Programming Model: OpenMP and Pthreads – Tools and Performance of examples on dense numerical computations

Duration-Part V :10 days **Module-17: 3** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. An Overview of Data Parallel Model (Features – Single threading, Parallel operations; synchronization, Interaction; Data allocation) **2.** The Fortran 90 Approach (Parallel Array Operational, Array Handling and Array Operations) **3.** The Fortan 90 Approach (Intrinsic functions in Fortran 90);- Matrix Operations; Reduction Operations **4.** High Performance Fortran (HPF) (Support for Data Parallelism); **5.** HPF: Data Mapping in HPF; Summary of Fortran 90 and HPF; Other Data Parallel Approaches	– Example programs: Fortran 90 for selective algorithms in Numerical computations – Example programs: HPF for selective algorithms in Numerical computations – Example f90/HPF program to write set of all primes less than equal to integer N (N > 1) (Use sieve of Eratosthenes method and Nest Procedure)

Part- VI: Module 18 (Prg-ExMpiPvm-1) An Overview of Message Passing Libraries; MPI-1and PVM	
Module 18: Explicit Parallelism -An Introduction to Message Passing model (MPI & PVM)	
Duration-Part VI :20 days **Module-18:4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
PHASE-I PVM	
Activities: Parallel Virtual Machine (PVM) **1.** History of Parallel virtual Machine (PVM) **2.** Understand concepts of PVM: PVM Daemon and PVM user-callable library **3.** PVM functions; Process Management, message passing, and virtual machine management **4.** PVM Console; Dynamic Configuration PVM group functions; **5.** PVM communication Functions and Communications protocols used in generic PVM; **6.** Understand communication paradigms (UDP/TCP) between two PVM daemons	**Deliverables** – Completion of PVM Programming Assignments; – Demonstration of PVM programs – Summary on cost estimation for PVM Library Calls with different message sizes on Clusters – Example programs using PVM library calls
PHASE-II: MPI	
Message Passing Libraries:MPI-1 **7.** History of Message Passing Interface (MPI) Messages; **8.** Understand Basics of (MPI-1) and Concepts of MPI Program, Understand SPMD & MPMD Programming paradigms **9.** Compilation & Execution of MPI program; Message Envelope in MPI (Data + Envelope); Basic Sending Messages & Receiving Messages; Understand SPMD & MPMD Programming paradigms **10.** Use of MPI Point-Point Blocking Send and Blocking Receive Library Calls – Understand exact syntax of MPI Send & MPI Recv Calls and Parameters used in Syntax **11.** Visit to MPI Forum & Know about Updates – and Discuss with team members	– Completion of MPI Programming Assignments; Discussion & Demonstration – Summary on Cost estimation for Point-to-Point Message Passing Library Calls with different message sizes on Clusters – Completion of Programming Assignments as given in Betatesting Web-Page, focusing on Dense Marix Computations

Part-VI:
Module 19 (Prg-ExMPI-1)
An Overview of Message Passing Library: MPI-1

Module 19 : An Overview of Message Passing Interface (MPI-1) Library and Performance of parallel programs using MPI-1

Duration-Part VI:20 days **Module-19: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I

Activities	Deliverables
1. Use of MPI Point-Point Blocking Send and Blocking Receive Library Calls; Message Envelope in MPI (Data + Envelope); MPI Environment Management – (Implementation Information, Timer functions; Startup; Error handling); MPI Profiling **2.** Use MPI Collective Communication Library Calls & Collective Communications and Comp Library Calls – Understand exact syntax of Parameters used, Tags, Safety, buffering, Synchronization **3.** Use MPI Advanced Collective Operations (Data Movement; Collective Computation Communications; Intra Communicator collective communications (MPI_Scatter(v), MPI_Gatherv, MPI_Alltoall(v), MPI_Reduce_scatter (v), MPI_Scan, MPI_Reduce)	– Completion of MPI Programming Assignments; Discussion and explanation of selective programs – Summary of Advanced Point-t-Point Communication library Calls Summary on Cost estimation for Point-to-Point Message Passing Library Calls with different message sizes

PHASE-II

4. Understand Grouping Data for Communication (MPI Derived Data types & MPI_Type_Struct); Other Derived Data Communications; Type Matching, Grouping Data Using MPI Pack /Unpack library calls **5.** Understand communicators and Topologies (Topologies, working with Groups, Contexts, and Communicators, MPI Cartesian topologies MPI_Comm_Split; Cartesian topology Management, Graph Topology Management) **6.** Use MPI Advanced Point-to-Point Communication Library Calls Persistent Communication Requests; Communication modes & Understand exact syntax of Parameters used in Syntax, Tags, Safety, buffering, synchronization **7.** MPI Intercommunicator routines (Local and Remote groups) **8.** MPI- Dealing with I/O on Clusters using MPI-1; File I/O; Array I/O; Understand Parallel I/O on Parallel Systems; MPI I/O MPI I/O library calls syntax,	– Summary on MPI Grouping Data for Communications; Communicators & Topologies – Completion of Programming Assignments as given in Betatesting Web-Page, focusing on Numerical Computations and given in courseware – Summary on Cost estimation of Collective Communication Library Calls – Summary on Cost estimation of Collective-Communication and Computation Message Passing library calls with different message sizes on Clusters

PHASE-III

9. MPI -1: Use of MPI parallel Libraries – PetSc, BLAS, BLAC, ScaLAPACK, LINPACK & Performance Issues **10.** MPI -1: Parallel Algorithms implementation MPI-1 for Allgather using Hypercube topology and Allgather using Ring topology with different message sizes.	– Demonstrate the Performance of compute Intensive programs using MPI parallel libraries

Part-VI: Module 20 (Prg-ExMPITools) An Overview of Message Passing Library- MPI-1.0 and MPI tools	
Module 20 : An Overview of Performance Visualization tools using Message Passing Library: MPI-1	
Duration-Part VI :20 days **Module-20: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
PHASE-I: MPI 1.0	
Activities **1.** Performance of MPI programs in Gflop/s using PAPI tool **2.** MPI performance Visualization tools; Use Upshot/JumpShot tools and for performance of MPI programs **3.** Using PRISM – Performance Visualization tools on PARAM 10000 /PARAM Padma **4.** Using Parallel Debuggers: To Debug Parallel Programs Using Commercial tools -VAMPIR, PRISM **5.** Using C-DAC HPC tools on PARAM 10000 & PARAM Padma - MPI performance Visualization tools	**Deliverables** – Completion of MPI-1 Programming Assignments; Discussion & Demonstration – Demonstrate the results of MP Program using Upshot /JumpShot/PAPI tools – Summary on Performance Visualization of MPI programs using tools

Part-VI: Module 21 (Prg-ExMPIMixed-1) Mixed Mode Programming (MPI /OpenMP & MPI/Pthreads)	
Module 21: Performance of parallel algorithms using mixed mode (MPI/OpenMP and MPI-Pthreads) Programming model	
Duration-Part VI :20 days **Module-21: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Understand theoretical concepts; Deliver lectures; Theoretical Examination; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities 1. MPI-OpenMP programs – and MPI-Pthreads Performance Issues 2. Parallelization of solution of matrix system of linear equations AX=b by JACOBI method using MPI-OpenMP and MPI & Pthreads mixed programming mode on Commodity Cluster (4 node cluster with quad CPU) 3. Performance issues of mixed mode of programming model MPI - OpenMP & MPI - Pthreads 4. Performance issues of benchmark dense matrix computations algorithm using MPI-OpenMP and MPI-Pthreads on a Commodity Cluster (4 node cluster with quad CPU)	**Deliverables** – Analysis of memory bandwidth used by OpenMP /Ptheads; Cache hit rate for example programs – Analysis on Performance of OpenMP functions used in MPI-OpenMP programs. – Benchmark MPI and MPI-OpenMP program on Commodity Clusters. – Performance of Benchmarks and Quantification of Overheads

Part-VI:
Module 22 (Prg-ExMPI2-1)
An Overview of Message Passing Library- MPI2

Module 22: An Overview of Message Passing Library- MPI-2 which is an extension of MPI-1

Duration-Part VI :20 days **Module-22: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Theoretical Examination; Programming Exercises; Performance of Programs; Assignments; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I: MPI 1.0

Activities	Deliverables
1. **MPI -2:** History of MPI-2 2. **MPI-2:** Parallel I/O-Non-contiguous Accesses and Collective I/O; Non-blocking I/O and Split Collective I/O; File Inter operability, Achieve High Performance with MPI; Examples 3. **MPI -2**: Understanding Synchronization in MPI 4. **MPI-2:** One Sided Communication – Remote Memory ones-sided communication (Memory windows, Moving Data, Completing Data Transfers; Examples of RMA Operations; Pitfalls in Accessing Memory; Performance tuning for RMA operations) 5. **MPI-2** Environment tools – Profiling, monitoring and debugging tools; Enhance functionality or reduce memory requirements; Accessibility of MPI objects 6. **MPI-2** Remote Memory Access features: Advanced Remote Memory Access Allocating Memory for MPI windows; Using Global Arrays; Managing distributed data Structures, MPI implementation of Dynamic distributed List; Complier Optimization; Scalable Synchronization 7. **MPI-2:** Dynamic Process Management features: Creating new MPI process; collecting MP Processes; Design of the MPI Dynamic Process Routines; Using MPI with Threads 8. **MPI-2:** Advanced Features: Defining New file Data Representations; Interface Functions; Mixed –Language Programming; Attribute caching; Error Handling 9. **MPI-2**: New Class of Parallel Programs; MPI-2 Implementation Status 10. **MPI-2**: Fortran 90; C++ and JAVA 11. **MPI-2:** MPI-2 Where Does MPI-1 go from Here? (Remote Memory Operations; More on Threads; More Language Bindings; More on Threads; Interoperability of MPI Implementations	– Summary on Cost estimation of MPI-2 Library Calls – Completion of MPI-2 Programming Assignments – Deliver lectures on MPI-2 – Demonstrate example programs results – Summary on MPI-2 - One sided Communications – Summary of MPI-2 Dynamic Process Management – Summary on Cost estimation of MPI-2 Library Calls – Summary of MPI-2 Performance Issues

Part- VI:
Module 23 (Prg-ExMPIBench-1)
MPI Communication Performance (MPI Benchmarks)

Module 23 : MPI Communication Performance (MPI Benchmarks)

Duration-Part VI:20 days **Module-23: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:**Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I

Activities	Deliverables
1. **Testing Environment**: Understand Testing environment of a Cluster and document the configuration of cluster. 2. Testing Rules and Software: Define the run-rules (Default baseline optimizations) for Cluster configuration and no network specific optimizations are applied in the production runtime environment	1. A brief presentation on Testing environment for the Hardware, and System Software

PHASE-II

3. **Sensitivity of networks:** The latency-sensitive nature of SANS, characteristics of low latency & High Bandwidth networks should be identified with benchmarks which run MPI over TCP; Study-the small and larger message size and their impact on performance 4. To understand, port & execute the Benchmarks for TCP Performance (`NTTCP suites`) on Cluster Configuration - a network-measurement tool 5. To understand, & execute the Benchmark `Netpipe` on Cluster environment - a network-measurement tool	2. Performance of Benchmarks and Quantification of Overheads in Parallel Computing Systems 3. Performance of Benchmarks and Quantification of Overheads in Parallel Computing Systems

PHASE-III

MPI Benchmarks	
6. To measure MPI performance `MPBench`; Evaluates several MPI performance metrics (A part of `LLCbench`: Low Level Characterization benchmarks) 7. Execution of Free downloadable MPI performance benchmarks used in `mpich` installation (`mpptest`) suites 8. Execution of `PALLAS`: Free downloadable MPI performance benchmarks; `SKaMPI`: Special Karlsruhe MPI Benchmarks; measure the performance of MPI. 9. Execution of `Sphinx`:An integrated Parallel Micro Benchmarks 10. Execution of `HLRS`: Aggregate Bandwidth, Bisection Bandwidth for Teraflop Clusters **11.** Execution of `P-COMS`: Free downloadable PARAM - Communication Overhead Suites; `PG-COMS`: Free downloadable PARAM–Generalized Communication Overhead Measurement Suites	4. Comparison of performance results of various MPI Benchmarks

Part-VII :
Module 24 (PP-MicroBench-1):
Application and System Benchmarks- Micro Benchmarks

Module 24 : Execution of micro benchmarks (Application and System Benchmarks) on Shared and Non-Shared Memory Computing systems

Duration-Part VII :20 days **Module-24: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:**100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I

Activities: Micro Benchmarks	Deliverables
1. LINPACK: Numerical Computing (Linear Algebra) **2.** Top-500: Top 500 Supercomputing Benchmarks **3.** ScaLAPACK**:** Numerical Computing (Linear Algebra) **4.** LMBENCH: System Calls and data movement operations in Unix; to measure the Operating system overheads and the capability of data transfer between processor, cache, memory and network, disk on various Unix platforms. **5.** STREAM: Memory Bandwidth (MB/s) and the corresponding computation rate **6.** Execution of MPI Communication Benchmarks **7.** Execution of selective HPC Challenge Benchmarks	– Performance of Benchmarks – Summary on Cost estimation for Point-to-Point Message Passing library Calls with different message size using MPI communication Benchmarks

Part-VII:
Module 25 (PP-MacroBench-1):
Application and System Benchmarks - MacroBenchmarks

Module 25: Execution of *Macro* Benchmarks (Application and System Benchmarks) on Shared and Non-Shared Memory Computing System)

Duration-Part VII :20 days **Module-25: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities: Macro Benchmarks	Deliverables
1. NAS : Parallel Computing Benchmarks **2.** PARKBENCH**:** Parallel Computing Benchmarks **3.** SPEC **:** Real Applications benchmarks – actual workload; CPU performance, Client /Server Computing, commercial applications, I/O subsystems etc…. **4.** Splash **:** Parallel Computing Benchmarks **5.** STAP:Signal Processing Benchmarks **6.** TPC **:** Commercial application Benchmarks (Transaction Processing Performance Council) **7.** Execution of selective HPC Challenge Benchmarks	– Performance of Benchmarks

Part-VII:
Module 26 (PP-HPCCBench): Execution of High Performance Computing Challenge Benchmarks (HPCC Benchmarks)

Module 26: High Performance Computing Challenge Benchmarks

Duration-Part VII :20 days **Module-26: 4** Calendar Days (Overlap with time schedule with other modules)
Starting Date: **Completion Date:**

Classification: Short or Mid or Long Term
Project Review: for every 48 Hours
Score: 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities: HPC Challenge Benchmarks (Seven)

1. **HPL** (The Linpack benchmark which measures the floating point rate of execution for solving linear system of equations -LU Factorization – Performance: Tflop/s; MPI on whole system is required)
2. **DGEMM** - measures the floating-point rate of execution of double precision real matrix-matrix multiplication.
3. **STREAM** - synthetic benchmark - measures sustainable memory bandwidth (in GB/s); Machine Balance – relative cost of memory accesses versus arithmetic; Equivalent MFLOPS rating; the corresponding computation rate for simple vector kernel. (Stress CPU, Memory System, Interconnect; Allow Optimizations; Effort needed for tuning – Single CPU); (Stream – Embarrassingly parallel whole system)- Used on Shared Memory Systems
4. **PTRANS** (Parallel Matrix Transpose) – exercises the communications where pairs of processors communicate with each other simultaneously. MPI on whole system is required) (It is PARKBENCH MATRIX KERNEL BENCHMARKS-Dense matrix multiply; Transpose; Dense LU factorization with partial pivoting; QR Decomposition; Matrix Tridiagonalization)
5. **RandomAccess** – Single CPU; (RandomAccess – embarrassingly parallel; Random access – read; update; & write; MPI on whole system is required) RandomAccess - measures the rate of integer random updates of memory (GUPS).
6. **FFTE -** measures the floating-point rate of execution of double precision complex one-dimensional Discrete Fast Fourier Transform (DFT). A software package to compute Discrete Fourier Transforms of 1-, 2- and 3-dimensional sequences of appropriate length
7. **b_eff** (effective bandwidth benchmark) – a set of tests to measure latency and bandwidth of a number of simultaneous communication patterns

Remark: Out of SEVEN, Two New Benchmarks (DGEMM; FFTE) are included recently. For details visit the web-site http://icl.cs.utk.edu/hpcc. (Rules for running the HPC Challenge benchmark should be followed)

Deliverables

- Performance and Analysis of Benchmarks
- Analysis of Sustained Performance versus Peak Performance
- Comparison of Benchmarks on different Parallel Computing systems
- Current Status of Top-500 Benchmarks

Part-VIII:
Parallel Algorithms and Implementation: Numerical Computing -I:
Module 27 (PNum-Matrix-1): Dense Matrix and Sparse Matrix computations

Module 27: Parallel Implementation of Dense Matrix Computations algorithms and Sparse Matrix into vector Multiplication Algorithms

Duration-Part VIII:10 days **Module-27:**6 Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I

Activities	Deliverables
1. Implementation of Vector - Vector Multiplication Algorithms (Uniform Partitioning, Non-Uniform Partitioning, and Cyclic Partitioning) using MPI; OpenMP;MPI-OpenMP Prog. Models; Implementation of Matrix-Vector Multiplication Algorithms (Rowwise one – dimensional Partitioning; Columnwise one –and Two dimensional Partitioning algorithms) using MPI; OpenMP; MPI-OpenMP Programming.Models **2.** Implementation of square Matrix Transpose Algorithm using different partitioning of square Matrix **3.** Implementation of Matrix-Matrix Multiplication Algorithms (Rowwise & Columnwise; one & Two dimensional partitioning; block Partitioning; Checker-Board Partitioning, Block Cyclic Partitioning) and Master Slave Algorithm; Cannon's algorithm, Fox Algorithm and The DNS Algorithm - using MPI; OpenMP;MPI-OpenMP Programming Models	– Deliver Lectures on Matrix Partitioning techniques and Performance issues on Parallel Computers – Performance of various dense Matrix Computational algorithms on Parallel Computers.

PHASE-II

4. Solving System of Linear Equations (Direct Methods – A simple Gaussian Elimination Algorithm, Gaussian Elimination with Partial Pivoting; Triangular System of Equations – back Substitution) using MPI; OpenMP;MPI-OpenMP Prog. Models **5.** Solving System of Linear Equations (Iterative Methods – A simple Gauss Jacobi and Siedel Method; Conjugate Gradient Method); Performance of these algorithms on Parallel Computers using MPI; OpenMP;MPI-OpenMP Prog. Models **6.** Numerical Considerations in Solving System of Linear equations on various Parallel Computing Systems	– Performance of direct and Iterative Methods for solution of system of linear equations on Parallel Computers using different programming paradigms

PHASE-III

7. Execution of Parallel BLAS-I, II and III libraries and Parallel BLACS libraries on Commodity Cluster **8.** Parallel DGEMM: Dense Matrix into Matrix Multiplication algorithm on Commodity Cluster **9.** Execution of PARKBENCH Matrix Computation Suites **10.** Performance of ScaLAPACK Algorithm on Clusters **11.** Performance of LINPACK and Top-500 Supercomputing Benchmark algorithm on Commodity Cluster **12.** Implementation of Sparse Matrix Computations	– Performance of MPI based Parallel libraries on Clusters – Performance of Sparse Matrix Computational libraries – Deliver lectures on Efficient implementation of Sparse Matrix Computations

Part-VIII:
Parallel Algorithms and Implementation: Numerical Computing -I:
Module 28 (PNum-FFT-1): Parallel Fast Fourier Transformations (FFTs)

Module 28 : Implementation of Parallel Algorithms for Fast Fourier Transformations (FFTs)

Duration-Part VIII:10 days **Module-28:4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored
Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

Activities	Deliverables
1. An Overview of Discrete Fourier Transform (Introduction; Fourier Analysis; Inverse discrete Fourier Transform; Sample Application; Polynomial Multiplication **2.** An Overview of Serial algorithm (The Recursive one-dimensional, radix –2 FFT Algorithm implementation) **3.** Implementation of *Cooley-Tukey* Algorithm for one-dimensional, un-ordered radix-2 FFT **4.** Recursive Sequential implementation of the FFT Algorithm and iterative sequential implementation of algorithm for FFT	– Performance of FFTs – Deliver lectures on efficient implementation of FFTs – Estimation of Communication overheads for implementation of FFTs on Parallel Computer
5. Parallel program Design – Partitioning and Communication; Agglomeration and Mapping; Speed-UP and Isoefficiency Analysis **6.** Parallel implementation based on sequential iterative algorithm, **7.** The Binary-Exchange Algorithm (A full Bandwidth Network; One task for process; Multiple Tasks for Process) **8.** Scalability Analysis of Binary-Exchange Algorithm **9.** The Binary-Exchange Algorithm (Limited Bandwidth Network; Implementation) **10.** Extra Computations in Parallel FFT and Performance Issues	– Performance of the Binary Exchange Algorithms on Parallel Computer
11. Implementation of The Transpose Algorithm (The Two dimensional Transpose algorithm; Comparison with the Binary-Exchange Algorithm) **12.** Implementation of The Transpose Algorithm (The Generalized Transpose Algorithm) **13.** Execution of FFT Benchmark– a part of NAS Benchmarks and HPCChallenge Benchmarks **14.** Performance of The Transpose Algorthm on Commodity Clusters using Programming Paradigms MPI, OpenMP, and MPI-OpenMP **15.** Total Communication Overheads for FFT for mesh and hyercube topology architecture **16.** Summary of aggregate Computations and Communication for a radix-4 FFT and radix-2 FFT on a hypercube topology architecture	– Deliver lectures on Communication overheads for FFT on Mesh and Hypercube topology architecture – Performance of FFT suite in HPC Challenge Benchmarks on Cluster.

Part IX: Parallel Algorithms and Implementation: Numerical Computing -II: Module 29 -(PNonNum-Sort-1) Parallel Sorting Algorithms	
Module 29 : Implementation of various Sorting algorithms and performance issues on Shared and Non-Shared programming platforms	
Duration-Part IX :15 days **Module-29 :**3 Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities : Sorting Algorithms **1.** Issues in Sorting on Parallel computers **2.** Implementation of Sorting Networks (Mapping a Bitonic Sort to a Hypercube and a Mesh topology) **3.** Buble sort algorithmes; **4.** Quick Sort (Parallel Quick sort; Pivot Selection); **5.** Implementation of Sample Sort and Bucket Sort and Other Sorting algorithms- Performance Issues **6.** Execution of Benchmark on Sorting Algorithms – a part of NAS Benchmarks	**Deliverables** – Parallel implementation of selective Sorting Algorithms – Performance of selective Benchmarks on Sorting Algorithms – Group Discussion on Sorting Algorithms – Deliver Talks on lectures on Performance issues of Sorting Algorithms

Part IX: Parallel Algorithms and Implementation: Numerical Computing -II: Module 30 -(PNonNum-Graph-1) : Parallel Graph Theory Computations	
Module 30: Implementation of Graph Theory Computational Algorithm on Shared and Non-Shared programming platforms and performance of algorithms	
Duration-Part IX :15 days **Module-30:** **4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities: Graph Algorithms: **1.** Implementation of Minimum Spanning Tree: Prim's algorithms and Single-source Shortest Paths: Dijkstra's Algorithm **2.** Implementation of All-Pairs of Shortest Paths (Dijkstra's Algorithm; Floyd Algorithm; Performance comparisons) **3.** Connected components: A Dept –First Search Based Algorithm and Best-First Search Algorithm **4.** Implementation of Algorithm for Sparse Graphs **5.** Implementation: Finding a Maximal Independent Set and Coloring a graph	**Deliverables** – Parallel implementation of selective Search Algorithms – Deliver Talks on lectures on Performance issues of Search Algorithms – Parallel implementation of All-Pairs Shortest Paths – Parallel implementation of Sparse Graphs

Part IX: Parallel Algorithms and Implementation: Numerical Computing -II: Module 31 (PNonNum-Search-1): Search Algorithms for Discrete Optimization	
Module 31 : Implementation of Search Algorithms for Discrete Optimization on Shared and Non-Shared programming platforms and performance of algorithms	
Duration-Part IX :15 days **Module-31: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities: Search Algorithms **1** An overview of Sequential search algorithms (Depth-First Search Algorithm; Best-First Search Algorithm) **2** Search Overhead Factor **3** Implementation of Parallel Depth First Search (DFS); **4** (A General framework for Analysis of Parallel DFS; Analysis of load balancing schemes; Experimental Results) 5 Implementation of Parallel Best First Search 6 Speedup Anomalies in Parallel Search Algorithms (Average speedup in Parallel DFS	**Deliverables** **1.** Performance of Selective Search algorithms on Commodity Clusters **2.** Deliver Talks on lectures on Performance issues of Search Algorithms

Part IX: Parallel Algorithms and Implementation - Numerical Computing -II: Module 32 (PNonNum-DynProg-1): Dynamic Programming	
Module 32 : Implementation of Dynamic Programming Algorithms on Shared and Non-Shared programming platforms and performance of algorithms	
Duration-Part IX :15 days **Module-32: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score:** 100 **Reference:** Refer Appendix-I
Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Assignments; Lectures on Algorithms; Evaluation is based on Performance & Marks scored **Resources:** PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles	
Activities: Dynamic Programming 1 An Overview of Dynamic Programming 2 Implementation aspects of The Shortest Path Problem; Nonserial Monadic Formulations 3 Serial Polyadic DP Formulations (Floyd's All-Paris Shortest Paths Algorithm); Non-Serial Polyadic DP Formulations; The optimal Matrix-Parenthesization Problem	**Deliverables** – Performance of Selective DP formulation on Commodity Clusters – Deliver Talks on lectures on Performance issues of Search Algorithms and Dynamic Programming

Part X:
Module 33 (PerfScalApp-1) : Scalability and Performance of Applications

Module 33 : Implementation issues for Performance and Scalability of Applications using different Programming paradigms

Duration-Part X :4 days **Module-33: 4** Calendar Days (Overlap with time schedule with other modules) **Starting Date:** **Completion Date:**	**Classification:** Short or Mid or Long Term **Project Review:** for every 48 Hours **Score**: 100 **Reference:** Refer Appendix-I

Nature of Work & Evaluation Criteria: Deliver lectures; Demonstration of Benchmarks; Programming Exercises; Evaluation is based on Performance of Projects & Marks scored

Resources: PARAM Padma, PARAM 10000, and Commodity Clusters at C-DAC or Parallel Computing Systems at C-DAC; Betatest (NPSF) Web-site; Internet Web-sites; Reference Books/Articles

PHASE-I	
Activities 1. Memory reference Optimization & Data Movement Operations 2. Understand I/O subsystem Performance 3. Use System tuned Libraries and measure performance 4. Estimate Performance using Mflop/s 5. Analyze Performance with respect to increase in problem size	**Deliverables** Deliver Lecture(s) on Scalability of Parallel Algorithms; Performance of Benchmarks
PHASE-II	
6. Choice of Right Programming Model (MPI, OpenMP, Pthreads, Mixed mode) for Scalability Analysis 7. Summary on Cost estimation for MPI Library Calls with different message sizes, used in Applications 8. Strategy for Overlap of Computations and Communications 9. Understand issues in Partitioning algorithms for Dense and Sparse Matrices with respect to Application **10.** Scalability of Algorithms for Direct and Iterative Matrix Solvers, used in Applications **11.** Performance issues of Selective Micro and Macro Benchmarks **12.** Asymptotic Analysis of Parallel Algorithms used in Application and the Degree of concurrency for given application problem size	
PHASE-III	
13. Understand Graph Partitioning Algorithms for Sparse Matrix Computations 14. Understand Dynamic Load Balancing Algorithms for Application 15. Measure performance in Gflop/s for Application using PAPI tool 16. Understand Adaptive re-partitioning algorithms for load Balancing of computations for Unstructured Computations	

5. Appendix-I: References

1. Anotonie Pettitet, Susan Blackford, Jack Dongarra, Brett Ellis, Graham Flagg, Kenneth Roche, and Sathish Vadhiyar, *Numerical Libraries And The Grid: The GrADS Experimnets with ScaLAPACK*, UT-CS-01, 460 report, University of Tennesse, Knoxville TN 37996, April **(2001)**
2. Gropp, W. et al., *A high Performance, Portable Implementation of the MPI message Passing Interface Standard Parallel Computing*, vol.22 (6); 00789 –828, (**1996)**
3. J. Dongarra, Top-500 web-site :*www.top500.org*
4. Whaley R., A Petitet, and J. Dongarra, *Automated Empirical Optimization of Software and the ATLAS*, Parallel Computing, Vol, 27 (1-2), page 03-25, (**2001)**
5. Bailey, D.H., T. Harris, W. Saphir, R. Van Der Wijingart, A. Woo, and M. Yarrow, The NAS Parallel Benchmarks 2.0, *NASA Ames Research Center Report NAS-95-020* (**1995**).
6. *PALLAS, Version 2.2.1,* http://www.pallas.com/, *(***2002)**
7. http://hpcf.nersc.gov/vendor_docs/ibm/essl/essl02.html,*ESSL – Engg. And Scientific Subroutine Library (***2004)**
8. http://hpcf.nersc.gov/vendor_docs/ibm/pessl/pessl02.html, *PESSL – Parallel Engg. And Scientific Subroutine Library, (***2004)**
9. http://www.nag.co.uk/numeric/numerical_libraries.asp*; NAG– The Numerical Algorithms Group Ltd.,* **(2004***)*
10. http://www-fp.mcs.anl.gov/petsc/, *PETSc – Portable, Extensible Toolkit for Scientific Computation, (***2004)**
11. *Karypis George and Vipin Kumar:* METIS and ParMETIS, Serial and Parallel Software Packages for Partitioning Unstructured Graphs for Computing Fill-reducing Orderings of Sparse Matrices. Army High Performance Computing Research Center, University of Minnesota, http://www.cs.umn.edu/~karypis/metis/metis.html **(1995)**
12. *Johnson A.A and T. E. Tezduyar*, Parallel Computation of Incompressible Flows with Complex Geometries. International Journal for Numerical Methods in Fluids, Vol. 24: pp. 1321-1340 (**1997)**
13. P.Henon and Yousef Saad, *A parallel multilevel {ILU} factorization based on a hierarchical graph decomposition*, Report umsi-2004-xx1, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, (**2004)**
14. Yousef Saad and Masha Sosonkina, *pARMS : A package for the parallel iterative solution of general large sparse linear systems user's guide, Report UMSI2004-8, Minnesota Supercomputer Institute, Univ of Minnesota, Minneapolis, MN* (**2004)**.
15. Foster, I., C. Kesselman, and S. Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations,* International Journal of High Perf. Computing Applications, Vol. 15, No. 3, (**2001)**
16. Indrajit S. Dhillon, Narendra K. Karmarkar, K.G. Ramakrishnan, " An Overview Of The Compilation process For A New Parallel Architecture", *Mathematical Sciences Research Center, AT&T Bell Laboratories, Indian Institute of Technology-Bombay, April,* (**1991)**
17. Narendra K. Karmarkar "A New Parallel Architecture For Sparse Matrix Computations", *Proc.Of The Work Shop On Parallel Computing, BARC, BOMBAY, 1-18, February (***1990)**
18. Narendra K. Karmarkar, " A New Parallel Architecture For Scientific Computation Based On Finite Projective Geometries", *Mathematical Programming: State Of The Art* (**1994***)*
19. Thomas Sterling, Paul Messina and Paul H. Smith," Enabling Technologies For *Petaflops*

Computing", *Scientific And Engineering Computation Series, MIT Press, Cambridge Massachusetts, London, England,* (**1995)**
20. Thomas Sterling," Interdependence of architecture and software for effective terascale computing", PRIMEUR, *The monthly news service for the European HPCN Community*, Issue 27, June **(2003)**
21. Tom R Halfhill, "IBM Makes Designer Genes", Micro Processor, MicroProcessor Report, @IN-STAT-MDR, URL:http://researchweb.watson.ibm.com/bluegene/; October 11, (**2004)**
22. *Exploring the benefits of FPGA-processor technology for genome analysis at Acconovis*; ISC 2003 Conference; Heidelberg, URL : http://www.acconovis.com; PRIMEUR, *The monthly news service for the European HPCN Community*, Issue 27, June **(2003)**
23. *High Performance Computing –FPGA technology*; URL:http://www.cdac.in;
24. Ernst L. Leiss, *Parallel and Vector Computing A practical Introduction*, McGraw-Hill Series on Computer Engineering, Newyork (**1995)**.
25. Peter S.Pacheco, Parallel Programming with MPI, Morgan Kaufmann Publishers,Inc San Francisco,California (**1997)**
26. Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, *Introduction to Parallel Computing*, Second Edition, Harlow, England, Addison Wesley (**2003)**
27. William Gropp, Rusty Lusk, *Tuning MPI Applications for Peak Performance*, Pittsburgh (**1996)**
28. Ian T. Foster, *Designing and Building Parallel Programs, Concepts and tools for Parallel Software Engineering*, Addison-Wesley Publishing Company (**1995)**
29. Kai Hwang, Zhiwei Xu, *Scalable Parallel Computing (Technology Architecture, Programming)*, McGraw Hill International Editions, (**1998)**
30. Kai Hwang and Fave A Briggs, *Computer Architecture and Parallel Processing***,** McGRAW HILL International Editions (**1984)**
31. Culler David E, Jaswinder Pal Singh with Anoop Gupta, *Parallel Computer Architecture, A Hardware/Software Approach*, Morgan Kaufmann Publishers, Inc, Reprinted (**2004)**
32. Quinn Michael J., *Parallel Programming in C with MPI and OpenMP*, Tata-McGRaw-Hill Edition, (**2003)**
33. Rajeev Matwani and Prabhakar Raghavan , *Randomised Algorithms*, Combridge University Press, (**2004)**
34. Nancy A Lynch, *Distributed Algorithms*, Morgan KAYFMANN, (**2003)**
35. Sartaj Sahani, *Data Structures Algorithms and Applications in JAVA*, (**2005)**
36. William Gropp, Rajeev Thakur: *Using MPI-2 Advanced Features of the Message-Passing Interface, The MIT Press*, **(2000)**
37. The Message Passing Interface (MPI), Standard; http://www.unix.mcs.anl.gov/mpi/
38. NTTCP : New TTCP program, http://www.leo.org/~elmar/nttcp/
39. NetPIPE ; New TTCP Program, http://www.scl.ameslab.gov/netpipe/
40. LLCBench Home Page. http://icl.cs.utk.edu/projects/llcbench/
41. MPI Benchmarks: P-COMS http://www.cdac.in/
42. The High Performance Computing Center (Höchstleistungsrechenzentrum) Stuttgart (HLRS) of the University of Stuttgart; http://www.hlrs.de/organization/par/