Course Title: Parallel Computing **Credit: 3**
Course No: CSIT.421 **Number of period per week: 3+3**
Nature of the Course: Theory + Lab **Total hours: 45+45**
Year: Fourth, Semester: Eighth
Level: B. Sc. CSIT

1. Course Introduction

In a parallel computation, multiple processors work together to solve a given problem. While parallel machines provide enormous raw computational power, it is often not easy to make effective use of all this power. This course will describe different techniques used to solve the problems, in order to develop efficient parallel algorithms for a variety of problems. We will also pay much attention to practical aspects of implementing parallel code that actually yields good performance on real parallel machines.

2. Objectives

At the end of this course, you should be able to accomplish the objectives given below.

- Describe different parallel architectures; inter-connect networks, programming models, and algorithms for common operations such as matrix-vector multiplication.
- Given a problem, develop an efficient parallel algorithm to solve it and analyze its time complexity as a function of the problem size and number of processors.
- Given a parallel algorithm, implement it using MPI, OpenMP, pthreads, or a combination of MPI and OpenMP.
- Given a parallel code, analyze its performance, determine computational bottlenecks, and optimize the performance of the code.

3. Specific Objectives and Contents

Specific Objectives	**Contents**
• To understand basics of parallel programming. • To explain Flynn's classification and parallel algorithm cdesign model • To design elementary parallel algorithms.	**Unit I: Parallel Programming (5)** 1.1. Introduction to parallel programming, data parallelism, functional parallelism, pipelining 1.2. Flynn's taxonomy, parallel algorithm design – task/channel model , Foster's design methodology 1.3. case studies: boundary value problem, finding the, maximum – Speedup and efficiency,

	Amdahl's law, Gustafson Barsis's Law, Karp-Flatt Metric, Isoefficiency metric
• To explain message passing programming model. • To understand MPI interface and use common methods provided by it • To handle timing issues in MPI programs. • To write simple programs using MPI.	**Unit II: Message Passing Programming (10)** 2.1. The message-passing model, the message-passing interface, MPI standard, basic concepts of MPI: MPI_Init, MPI_Comm_size, MPI_Comm_rank, MPI_Send, MPI_Recv, MPI_Finalize, 2.2. Timing the MPI programs: MPI_Wtime, MPI_Wtick, collective, communication: MPI_Reduce, MPI_Barrier, MPI_Bcast, MPI_Gather, MPI_Scatter 2.3. case studies: the sieve of Eratosthenes, Floyd's algorithm, Matrix-vector multiplication
• To understand shared memory model of parallel programming and OpenMP standard. • To explain loops, critical section, function, etc in parallel programming • To write simple programs by using shared memory paradigm.	**Unit III: Shared Memory Programming (10)** 3.1. Shared-memory model, OpenMP standard, parallel for loops, parallel for pragma, private variables, critical sections 3.2. Reductions, parallel loop optimizations, general, data parallelism, functional parallelism 3.3. Case studies: the sieve of Eratosthenes, Floyd's algorithm, matrix-vector multiplication, distributed shared-memory programming, DSM primitives
• To understand basic principles of parallel algorithms • To understand principles of Monte Carlo method in algorithm design • To design parallel algorithms in specified topics.	**Unit IV: Parallel Algorithms I (10)** 4.1. Monte Carlo methods, parallel random number generators, random number distributions 4.2. Case studies: Matrix multiplication, row-wise block-stripped algorithm, Cannon's algorithm, solving linear systems, back substitution, Gaussian elimination, iterative methods, conjugate gradient method
• To design parallel algorithm for sorting data • To design searching and FFT parallel algorithms	**Unit V: Parallel Algorithms II (10)** 5.1. Sorting algorithms: quicksort, parallel quicksort, hyper quicksort, sorting by regular sampling 5.2. Fast fourier transform, combinatorial search, divide and conquer, parallel backtrack search, parallel branch and bound, parallel alpha-beta search.

Evaluation System

Undergraduate Programs							
External Evaluation	**Marks**	**Internal Evaluation**	**Weightage**	**Marks**	**Practical**	**Weightage**	**Mark**
End semester examination	60	Assignments	20%	20	Practical Report copy	25%	20
(Details are given in the separate table at the end)		Quizzes	10%		Viva	25%	
		Attendance	20%		Practical Exam	50%	
		Internal Exams	50%				
Total External	60	Total Internal	100%	20		100%	20
Full Marks 60+20+20 = 100							

External evaluation

1. **End semester examination:**
   It is a written examination at the end of the semester. The questions will be asked covering all the units of the course. The question model, full marks, time and others will be as per the following grid.

2. **External Practical Evaluation:**
   After completing the end semester theoretical examination, practical examination will be held. External examiner will conduct the practical examination according to the above mentioned evaluation. There will be an internal examiner to assist the external examiner. Three hours time will be given for the practical examination. In this examination Students must demonstrate the knowledge of the subject matter.

Full Marks: 100, Pass Marks: 45, Time: 3 Hrs

Nature of question	Total questions to be asked	Total questions to be answered	Total marks	Weightage
Group A: multiple choice*	20	20	20×1 = 20	60%
Group B: Short answer type questions	7	6	6×8 = 48	60%
Group C: Long answer type questions	3	2	2×16 =32	60%
			100	100%

Each student must secure at least 50% marks in internal evaluation in order to appear in the end semester examination. Failed student will not be eligible to appear in the end semester examinations.

Internal evaluation

Assignment: Each student must submit the assignment individually. The stipulated time for submission of the assignment will be seriously taken.

Quizzes: Unannounced and announced quizzes/tests will be taken by the respective subject teachers. Such quizzes/tests will be conducted twice per semester. The students will be evaluated accordingly.

Attendance in class: Students should regularly attend and participate in class discussion. Eighty percent class attendance is mandatory for the students to enable them to appear in the end semester examination. Below 80% attendance in the class will signify NOT QUALIFIED (NQ) to attend the end semester examination.

Presentation: Students will be divided into groups and each group will be provided with a topic for presentation. It will be evaluated individually as well as group-wise. Individual students have to make presentations on the given topics.

Mid-term examination: It is a written examination and the questions will be asked covering all the topics in the session of the course.

Discussion and participation: Students will be evaluated on the basis of their active participation in the classroom discussions.

Instructional Techniques: All topics are discussed with emphasis on real-world application. List of instructional techniques is as follows:

- Lecture and Discussion
- Group work and Individual work
- Assignments
- Presentation by Students
- Quizzes
- Guest Lecture

Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during that period. If a student fails to attend a formal exam/quiz/test, there won't be any provision for re-exam. Unless and until the student clears one semester he/she will not be allowed to study in the following semesters.

Laboratory Work

Students should practice small scale parallel programs, message passing programs, and shared memory programs. Besides this student should implement parallel algorithms discussed in the course.

Prescribed Text

- Michael J. Quinn, "Parallel Programming in C with MPI and OpenMP", Tata McGraw-Hill Publishing Company Ltd., 2003.

References

- B. Wilkinson and M. Allen, "Parallel Programming – Techniques and applications using networked workstations and parallel computers", Second Edition, Pearson Education, 2005.
- 2. M. J. Quinn, "Parallel Computing – Theory and Practice", Second Edition, Tata McGraw-Hill Publishing Company Ltd., 2002.