CS2001D LOGIC DESIGN

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to :
CO1: Learn and understand various number systems and their applications in digital design.
CO2: Design and implement logic functions utilizing logic gates and programmable logic.
CO3: Learn and understand HDLs used to implement digital systems.
CO4: Design simple digital systems using HDLs.

Module 1: (13 Hours)

Number systems and codes, Boolean algebra: postulates and theorems, constants, variables and functions, switching algebra, Boolean functions and logical operations, Karnaugh map: prime cubes, minimum sum of products and product of sums, Introduction of HDLs and their syntax.

Module 2: (13 Hours)

Quine-McCluskey algorithm, prime implicant chart, cyclic prime implicant chart, Petrick's method, Combinational Logic: introduction, analysis and design of combinational logic circuits, parallel adders and look-ahead adders, comparators, decoders and encoders, code conversion, multiplexers and demultiplexers, parity generators and checkers

Module 3: (13 Hours)

Programmable Logic Devices, ROMs, PALs, PLAs, PLA folding, design for testability. Introduction to sequential circuits, memory elements, latches

Module 4: (13 Hours)

Flip-flops, analysis of sequential circuits, state tables, state diagrams, design of sequential circuits, excitation tables, Mealy and Moore models, registers, shift registers, counters

References:

1. T. L. Floyd and R. P. Jain, *Digital Fundamentals*, 8/e, Pearson Education, 2006.
2. C. H. Roth, Jr., and L. L. Kinney, *Fundamentals of Logic Design*, 6/e, Cengage Learning, 2009.
3. M. M. Mano and M. D. Ciletti, *Digital Design*, 4/e, Pearson Education, 2008.
4. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1/e, Springer, 2017.

CS2002D PROGRAM DESIGN

Prerequisites: ZZ1004D Computer Programming

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to

CO1:Design and analyse simple iterative and recursive algorithms.
CO2:Design algorithms for sorting and searching and analyze them using mathematical tools, like formulation and solving of recurrences, asymptotic analysis
CO3: Define and describe simple data structures: arrays, linked lists and trees
CO4: Analyze the correctness of algorithms

Module 1: (13 Hours)

Review of Programming Constructs- Conditional and Iterative constructs, Data types, Control Structures, Functions, Parameter passing- calling conventions, Recursion, Asymptotic notation for complexity analysis.

Module 2: (13 Hours)

Searching - Linear and Binary, Sorting- Insertion and Selection sorting, Divide and conquer, Quick sort, Merge Sort, Heap Sort, External Sorting.

Module 3: (13 Hours)

Pointers and dynamic memory allocation, Abstract Data Types, Lists, Stacks, Queues, Trees, Search Trees and traversal algorithms, Heaps and Priority queues.

Module 4: (13 Hours)

Memory Management, Garbage collection algorithms, Storage allocation for objects with mixed sizes, Buddy systems, Storage compaction.

References:

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3/e, MIT Press, 2009.
3. E. Horowitz, S. Sahni, and D. Mehta, *Fundamentals of Data Structures in C++*, 2/e, Universities Press, 2008.

CS2006D DISCRETE STRUCTURES

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to :

CO1: Solve elementary combinatorial problems on graphs using recursive formulations and simple probabilistic methods.
CO2: Prove simple properties of numbers and polynomials using algebraic (group and ring theoretic) properties.
CO3: Estimate the cardinality of a given infinite set using set theoretic arguments.
CO4: Simplify and construct boolean expressions, and prove some basic theorems from a given set of boolean axioms.

Module 1: (13 Hours)

Combinatorics: Asymptotic analysis of recurrence - solution to recurrences. Graph Theory: Elementary properties - planar graphs - Euler's theorem - Five colour theorem.

Module 2: (13 Hours)

Discrete Probability : Discrete probability spaces, events, random variables, probabilistic method for solving combinatorial problems. Conditional probability, Bayes Theorem. Independent events. Binomial distribution and Geometric distribution. Linearity of expectations, method of conditional expectation, applications to analysis of randomized algorithms. Variance of a random variable. Markov and Chebyshev bounds.

Module 3: (13 Hours)

Algebra: Groups, Lagrange's theorem, Subgroups, Cyclic subgroups, Group Homomorphisms, Homomorphism theorem, Kernel of a homomorphism, Normal subgroups. Rings and Fields, Ring Homomorphisms, Ideals. Division rings, integral domains. Structure of the ring Zn and the unit group Zn*, polynomials over Zp. Order Structures: Equivalence relations, posets, lattices and boolean lattices.

Module 4: (13 Hours)

Logic and Set Theory: Boolean logic, Resolution in propositional logic - introduction to first order logic - set theory - countable and uncountable sets - diagonalization.

References:

1. R. P. Grimaldi, *Discrete and Combinatorial Mathematics: An Applied Introduction*, Addison Wesley, 1998.
2. L. Lovasz, J. Pelikan, and K. Vesztergombi, *Discrete Mathematics*, Springer, 2003.
3. I. M. Copi, *Symbolic logic*, Prentice Hall, 1979

CS2091D LOGIC DESIGN LABORATORY

Prerequisites: NIL

L	T	P	C
0	0	3	2

Total Hours: 39

Course Outcomes:

Students will be able to :

CO1: Learn and understand the underlying components of logic gates and circuits.
CO2: Learn and understand the HDLs (Hardware description languages)
CO3: Design and implement the combinational and the sequential logic circuits using HDLs

Practical: (39 Hours)

Design and implementation of the following logic gates and circuits using HDLs

1. AND, OR, NOT, NAND, NOR and XOR
2. Latches and Flip-flops
3. Adders and subtractors
4. Multiplexers and demultiplexers
5. Parity generators, code converters and comparators
6. Counters and Registers

References:

1. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1/e, Springer, 2017.
2. S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with Verilog Design*, McGraw-Hill Higher Education, Har/Cdr edition, 2002.
3. M. M. Mano and M. D. Ciletti, *Digital Design*, 4/e, Pearson Education, 2008.
4. T. L. Floyd and R. P. Jain, *Digital Fundamentals,* 8/e, Pearson Education, 2006.

CS2092D PROGRAMMING LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Implement fundamental algorithms like sorting and searching
CO2: Implement simple data structures (arrays, linked lists and trees) and their operations.
CO3: Analyze the computing problems given and assess the suitability of different data structures and algorithms to solve the problems.

Theory (13 Hours)

Review of dynamic memory allocation - use of pointers - review of recursion. File organization.

Practical (39 Hours)

1. Iterative and recursive algorithms
2. Searching: Binary search implementation
3. Sorting: Heap sort, Quick sort and Merge sort implementation
4. Stack and Queue implementation using linked list
5. Arithmetic expression to postfix
6. Postfix to expression tree, tree traversal and evaluation
7. Binary search tree - insert, delete and search.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3/e, MIT Press, 2009.
2. E. Horowitz, S. Sahni S, and D. Mehta, *Fundamentals of Data Structures in C*++, 2/e, Universities Press, 2008.
3. M. A. Weiss, *Data structures and algorithm analysis*, Addison-Wesley 1992.

CS2004D COMPUTER ORGANIZATION

Prerequisites: CS2001D Logic Design

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to :

CO1: Describe the hardware components of a computer its architecture and performance evaluation
CO2: Specify the instruction set of MIPS architecture to build a basic processor
CO3: Design construct and analyse a basic processor using single cycle, multi cycle, pipelined techniques
CO4: Analyze and specify new memory interactions to improve the performance of a computing system.

Module 1: (13 Hours)

Computer abstraction and technology: basic principles, hardware components, Measuring performance: evaluating, comparing and summarizing performance.
Instructions: operations and operands of the computer hardware, representing instructions, making decision, supporting procedures, character manipulation, styles of addressing, starting a program.

Module 2: (13 Hours)

Computer arithmetic: signed and unsigned numbers, addition and subtraction, logical operations, constructing an ALU, multiplication and division, floating point representation and arithmetic, Parallelism and computer arithmetic.

Module 3: (13 Hours)

The processor: building a data path, simple and multicycle implementations, microprogramming, exceptions, Pipelining, pipeline data path and Control , hazards in pipelined processors

Module 4: (13 Hours)

Memory hierarchy: caches, cache performance, virtual memory, common framework for memory hierarchies

References:

1. D. A. Patterson and J. L. Hennessy, *Computer Organisation and Design: The Hardware/Software Interface*, 5/e, Morgan Kaufmann, 2014.
2. V. P. Heuring and H. F. Jordan, *Computer System Design and Architecture*, Prentice Hall, 2003.

CS2005D DATA STRUCTURES AND ALGORITHMS

Prerequisites: CS2002D Program Design, CS2006D Discrete Structures

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to :

CO1: Analyze algorithms and data structures applying methods for amortized analysis
CO2: Evaluate methods for performance improvement of dictionaries and mergeable heaps
CO3: Analyze and assess the applicability of fundamental graph algorithms to applications
CO4: Assess and evaluate data structures based on randomization
CO5: Describe and apply data structures for distributed computing applications

Module 1: (07 Hours)

Review: Time and space complexity analysis, proof of correctness of algorithms, simple data structures and applications, Dictionaries, Hashing. Probabilistic Analysis. Amortized Analysis. Methods and examples.

Module 2: (19 Hours)

Graphs,Trees and Positional trees. Review of Binary Trees and Binary Search Trees. Rotations. Red black Trees, AVL Trees, Splay trees. Mergeable heaps. Fibonacci Heaps. Data structures for disjoint sets - union by rank and path compression

Module 3: (13 hours)

Graph representation- DFS, BFS, minimum spanning tree problem - Kruskal's algorithm - analysis and implementation using disjoint set data structure – Prim’s algorithm - Shortest path problem - Dijkstra's algorithm - analysis and implementation of Prim's and Dijkstra's algorithms using priority queues. Bellman Ford, Floyd-Warshall algorithms.

Module 4: (13 Hours)

Randomized Data Structures. Treaps, Skip lists. Randomized primality testing. Distributed Hashing and searching. Persistent Data Structures

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3/e, MIT Press, 2003
2. A. V. Aho, J. D. Ullman, and J. E. Hopcroft, *Data Structures and Algorithms*, Addison Wesley, 1983.
3. D. Kozen, *The Design and Analysis of Algorithms,* Springer, 1991.
4. C. Okasaki, *Purely Functional Data Structures,* Cambridge University press, 1999

CS2093D HARDWARE LABORATORY

Prerequisites: NIL

L	T	P	C
2	0	2	3

Total Hours: 26T+26P

Course Outcomes:

Students will be able to :

CO1: Describe the x86 architecture, its instruction set and basics of NASM Assembler
CO2: Use software interrupts for implementing input output operations in NASM programs
CO3: Design programs using integer, strings and floating point operations.

Theory (26 Hours)

Introduction to 8086 Microprocessor; Architecture of 8086, Memory addressing, Assembly Language Programming using 8086, Instruction set of 8086, Data movement Instructions, Arithmetic and logic instructions, Program control instructions, String handling instructions, procedures, recursions, floating point instructions, Basics of SIMD programming.

Memory and I/O interfacing, interfacing with 8255- Programmable peripheral interface, 8279 – programmable keyboard interface, 8254 timer interface - 16550 UART interface - ADC/DAC interfaces. Interrupts, hardware interrupts, Programmable interrupt controller 8259, Interrupt examples. Direct Memory Access, Basic DMA operation, 8257 DMA controller, Bus interface, ISA, VESA, PCI, USB

Introduction to NASM assembler, Sections in NASM, variables declarations, Basic instruction set, Basic I/O operations in NASM, Linux system calls, Interrupts, Linux 0x80h interrupt, Subprograms in NSAM, Arrays and strings, Using C Library functions in NASM, executing NASM programs, Sample programs, Floating point operations, SIMD operations.

Practical (26 Hours)

1. 80X86 Assembly language programming:
   Integer operations,
   Operations on arrays,
   Recursive subroutines,
   String manipulation,
   Floating point operations
   SIMD operations
2. Familiarization of PC hardware and troubleshooting

References:

1. P. Abel, *IBM PC Assembly Language and Programming,* 5/e, Prentice Hall, 2001.
2. B. B. Brey, *Intel Microprocessors: Architecture and Programming,* Prentice Hall, 2008.

CS2094D DATA STRUCTURES LABORATORY

Prerequisites: CS2002D Program Design, CS2006D Discrete Structures

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Analyze the computing problems given and assess the suitability of different data structures and algorithms to solve the problems
CO2: Design and implement solutions to graph based problems
CO3: Design and implement algorithmic solutions within suitable time constraints to face real life practical situations in the computing industry by following the ethics of computing

Theory (13 Hours)

Review of dynamic memory allocation - use of pointers - review of recursion. File organization.

Practical (39 Hours)

1. Linear time DFS and BFS implementation with adjacency list representation. (3)
2. Kruskal's algorithm implementation in O((n+e)log n) complexity. (3)
3. Prim's algorithm implementation in O((n+e)log n) complexity. (3)
4. Dijkstra's algorithm implementation in O((n+e)log n) complexity. (3)
5. Implementation of BST, rotations, and red black trees. (8)
6. Implementation of splay trees. (6)
7. Implementation of skip lists. (6)
8. Implementation of Random Treaps. (7)

References:

1. T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, PHI, 1998
2. S. Sahni, *Data Structures, Algorithms, and Applications in C++*, McGraw Hill, 1998

CS3001D THEORY OF COMPUTATION

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total Hours: 52

Course Outcomes:

Students will be able to :

CO1: Classify a given language according to its level in the Chomsky hierarchy and design grammars / Machines for the language.
CO2 : Construct finite state machines, pushdown automata and turing machines for a given language.
CO3: Prove undecidability of a given problem using diagonal method or reduction.
CO4: Prove NP completeness of a given problem using polynomial time reductions, and prove NP completeness of SAT by Cook-Levin Theorem.

Module 1: (13 Hours)

Basic concepts of Languages, Automata and Grammar. Regular Languages - Regular expression - finite automata equivalence, Myhill Nerode theorem and DFA State Minimization, Pumping Lemma and proof for the existence of non-regular languages.

Module 2: (13 Hours)

Context Free languages, CFL-PDA equivalence, Pumping Lemma and proof for existence of non- Context Free languages, CYK Algorithm, Deterministic CFLs, Chomsky Schutzenberger Theorem.

Module 3: (13 Hours)

Turing Machines: recursive and recursively enumerable languages, Universality of Turing Machine, Church Thesis. Chomsky Hierarchy, Undecidability, Reducibility, Undecidability: Recursive and Recursively enumerable sets.

Module 4: (13 Hours)

Complexity: Time and space complexity classes, hierarchy theorems, reductions and completeness, NP Completeness and PSPACE completeness, examples.

References:

1. M. Sipser, *Introduction to the Theory of Computation*, Thomson, 2001.
2. D. C. Kozen, *Automata and Computability*, Addison Wesley, 1994.
3. J. C. Martin, *Introduction to Languages and the Theory of Computation,* McGraw Hill, 2002.

CS3002D DATABASE MANAGEMENT SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T + 26P

Course Outcomes:

Students will be able to :

CO1: Model, design and normalize databases for real life applications.
CO2: Code and deploy databases for applications using RDBMS like ORACLE
CO3: Query Database applications using Query Languages like SQL
CO4: Deploy efficient IT solutions using free and open software and help the society

Module 1: (10T+8P Hours)

Database System Concepts and architecture, Data Modeling using Entity Relationship (ER) model and Enhanced ER model, Specialization, Generalization, Data Storage and indexing, Single level and multi level indexing, Dynamic Multi level indexing using B Trees and B+ Trees.

Module 2: (10T+8P Hours)

The Relational Model, Relational Database design using ER to relational mapping, Relational algebra and relational calculus, Tuple Relational Calculus, Domain Relational Calculus, SQL.

Module 3: (10T+5P Hours)

Database design theory and methodology, Functional Dependencies and Normalization of relations, Normal Forms, Properties of relational decomposition, Algorithms for relational database schema design.

Module 4: (9T+5P Hours)

Transaction processing concepts, Schedules and serializability, Concurrency control, Two Phase Locking Techniques, Optimistic Concurrency Control, Database recovery concepts and techniques, Introduction to database security.

References:

1. R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems,* 6/e, Pearson Education, 2011.
2. R. Ramakrishnan and J. Gehrke, *Database Management Systems,* 3/e, McGraw Hill, 2003.
3. P. Rob and C. Coronel, *Database Systems- Design, Implementation and Management,* 7/e, Cengage Learning, 2007.

CS3003D OPERATING SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Describe the working of a computing system from hardware to application program.
CO2: Design a basic operating system
CO3: Define processes, threads, interprocess and thread communication mechanisms and synchronization techniques.
CO4: Describe the kernel functions - process management, memory management, device management and file management

Module 1: (9T+6P Hours)

Review of operating system strategies - resources - processes - threads - objects - operating system organization - design factors - functions and implementation considerations - devices - characteristics - controllers - drivers - device management - approaches - buffering - device drivers - typical scenarios such as serial communications - storage devices etc

Module 2: (10T+8P Hours)

Process management - system view - process address space - process and resource abstraction - process hierarchy - scheduling mechanisms - uniprocessor and multiprocessor scheduling-various strategies - synchronization - interacting & coordinating processes - semaphores - deadlock - prevention - avoidance - detection and recovery

Module 3: (10T+6P Hours)

Memory management - issues - memory allocation - dynamic relocation - various management strategies - virtual memory - paging - issues and algorithms - segmentation - typical implementations of paging & segmentation systems

Module 4: (10T+6P Hours)

File management - files - implementations - storage abstractions - memory mapped files - directories and their implementation - protection and security - policy and mechanism - authentication - authorization - case study of Unix and Linux kernel .Virtual machines – virtual machine monitors – issues in processor, memory and I/O virtualization, hardware support for virtualization.

References:

1. A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Principles*, 9/e, John Wiley,2013.
2. W. Stallings, *Operating Systems:Internals and design Principles*, 7/e, Pearson Education, 2012.
3. A. S. Tanenbaum, *Modern Operating Systems*, 4/e, Pearson Education, 2017.
4. G. J. Nutt, *Operating Systems - A Modern Perspective,* 3/e, Pearson Education, 2009.

CS3004D SOFTWARE ENGINEERING

Prerequisites: CS2002D Program Design, CS2006D Discrete Structures

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Apply the basic concepts, principles and theories in software engineering to build software systems from the scratch, considering both technical and managerial aspects.
CO2: Design and implement different phases in the life cycle of software development and identify appropriate process models.
CO3: Analyze real problems/requirements and design systems by developing specifications and abstractions to make development of complex systems easy.

Module 1: (10T+5P Hours)

Introduction to Software Engineering – Reasons for software project failure – Similarities and differences between software and other engineering products. Software Development Life Cycle (SDLC) – Overview of Phases. Detailed Study of Requirements Phase.

Module 2: (10T+14P Hours)

Principles of software Design - Problem partitioning (subdivision) - Power of Abstraction. Concept of functional decomposition – UML diagrams - emphasis on class, object, sequence, activity diagrams. ER diagrams. Introduction to architectural patterns including MVC.
Coding and Testing: Structured programming – Methods and tools for version control - Maintainability. Types of testing – Specification of test cases – Code review and inspection process.

Module 3: (10T+7P Hours)

Software Project Management: Introduction to metrics. Software Process Models. Costing, Scheduling and Tracking techniques.
Methods of software licensing including free and open source software licenses.

Module 4: (9T Hours)

Current trends in Software Engineering: Extreme Programming - Values, Principles, Practices. Agile approach and manifesto. Introduction to Service Oriented Architecture - Entities and Characteristics - Web Service as an example of SOA Implementation- Evolution of Web Services- Technologies behind Web Service - SOAP, WSDL,UDDI, BPEL -RESTful Web Service Architecture- Micro Services.

References:

1. R. S. Pressman, *Software Engineering: A Practitioner's Approach,* 6/e, McGraw Hill, 2008.
2. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering,* 1/e, Tata McGraw Hill,2004.
3. K. Beck, *Extreme Programming,* 2/e*,* Pearson Education, 2006.
4. C. Fowler, *The Passionate Programmer,* SPD Pvt. Ltd., 2009.

CS3005D COMPILER DESIGN

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain lexical, syntax, and semantic analysis of programs.
CO2: Translate source code to target code.
CO3: Analyze a program and perform code optimizations.

Module 1: (9T+10P Hours)

Introduction to Programming language translation. Lexical analysis: Specification and recognition of tokens.

Module 2: (10T+10P Hours)

Syntax analysis: Top-down parsing-Recursive descent and Predictive Parsers. Bottom-up Parsing- LR (0), SLR, and LR (1) Parsers.

Module 3: (10T+6P Hours)

Semantic analysis: Type expression, type systems, symbol tables and type checking.
Intermediate code generation: Intermediate languages. Intermediate representation - Three address code and quadruples. Syntax-directed translation of declarations, assignments statements, conditional constructs and looping constructs.

Module 4: (10 Hours)

Runtime Environments: Storage organization, activation records. Introduction to machine code generation and code optimizations.

References:

1. A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques and Tools*, Pearson Education, 2007.
2. A. W. Appel and J. Palsberg, *Modern Compiler Implementation in Java*, Cambridge University Press, 2002.
3. D. Grune, K. van Reeuwijk, H. E Bal, C. J. H. Jacobs, and K. Langendoen. *Modern Compiler Design*, 2/e, Springer 2012.

CS3006D COMPUTER NETWORKS

Prerequisites: CS2005D Data Structures and Algorithms

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Define and describe functionality and services offered at various layers of TCP/IP protocol stack.
CO2: Design and implement simple application based on socket programming
CO3: Adapt the concepts learned in computer networking to solve the real life scenarios

Module 1: (10T+7P Hours)

Computer Networks and Internet, The network edge, The network core, Network access, Delay and loss, Protocol layers and services, Application layer protocols, Web 2.0, Socket Programming,

Module 2: (10T+7P Hours)

Transport layer services, UDP, TCP, New transport layer Protocols, Congestion control, New versions of TCP, Network layer services, Routing, IP, routing in Internet, Router, IPV6, Multicast routing.

Module 3: (10T+7P Hours)

Link layer services, Error detection and correction, Multiple access protocols, ARP, Ethernet, Hubs, Bridges, Switches, MPLS, VLAN.

Module 4: (09T+5P Hours)

wireless links, Mobility, Multimedia networking, Streaming stored audio and video, Real-time protocols, Network management.

References:

1. J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring Internet*, 6/e, Pearson Education, 2012.
2. L. L. Peterson and B. S. Davie, *Computer Networks, A systems approach*, 5/e, Morgan Kaufmann, 2011.
3. A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5/e, Pearson, 2013.

CS4023D ARTIFICIAL INTELLIGENCE

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Apply State-space search strategies for problem solving and implement it using LISP.
CO2: Use heuristic functions in search strategies and Games.
CO3: Explain and apply various knowledge representation mechanisms.
CO4: Identify and choose the appropriate machine learning approach for solving various problems.

Module 1: (10T+4P Hours)

Artificial Intelligence: Introduction,History and Applications; Intelligent Agents;Solving problems by Searching: Structures and Strategies for state space search- Data driven and goal driven search,Uninformed Search strategies, Informed(Heuristic) Search Strategies, Heuristic Functions, Local Search Algorithms and Optimization Problems, Searching with Nondeterministic actions, Constraint satisfaction, Using heuristics in games- Minimax Search, Alpha Beta Procedure,Stochastic Games.

Module 2: (10T+6P Hours)

Knowledge representation: Knowledge based agents,Propositional calculus, First-Order Logic (Predicate Calculus),Inference in First-Order Logic, Forward and Backward chaining, Theorem proving by Resolution, Answer Extraction, AI Representational Schemes- Semantic Nets, Conceptual Dependency, Scripts, Frames, Planning,Planning and acting in the real world.

Module 3: (11T+8P Hours)

Learning: Learning From Examples, Knowledge in Learning, Learning probabilistic Models, Reinforcement Learning, The Genetic Algorithm- Genetic Programming, Overview of Expert System Technology, Introduction to Natural Language Processing.

Module 4: (8T+8P Hours)

Languages and Programming Techniques for AI- Introduction to PROLOG and LISP, Search strategies and Logic Programming in LISP, Production System examples in PROLOG.

References:

1. S. Russell and P. Norvig, *Artificial Intelligence:A Modern Approach,* 2/e, Pearson Education, 2002.
2. G. F. Luger, *Artificial Intelligence- Structures and Strategies for Complex Problem Solving,* 4/e, 2002, Pearson Education.
3. E. Rich and K. Knight, *Artificial Intelligence,* 2/e, Tata McGraw Hill.
4. P. H. Winston, *LISP*, 3/e, Addison Wesley, 1989.
5. I. Bratko, *Prolog Programming for Artificial Intelligence,* 3/e, Addison Wesley, 2000.

CS4098D PROJECT: Part I

Prerequisites: NIL

L	T	P	C
-	-	6	3

Course Outcomes:

Students will :

CO1: Conduct literature survey and document a problem specification.
CO2: Identify a problem to work on for part II of the project, and lay out a plan of action for the project towards solving the problem.
CO3: Document the literature survey and problem specification into a report.

Students shall form a team of at most four for the project and identify a faculty member as the project guide, with whom they associate for the project work for a period of two semesters.

In this part of the project, the team, in consultation with the guide shall identify an area of work and conduct a detailed literature survey of the relevant work in the area. The team shall then identify a problem and prepare a report of the problem they are going to work on for the second part of the project.

References:

1. G. J. Alred, C. T. Brusaw, and W. E. Oliu, *The Handbook of Technical Writing*, 11/e, Bedford/St. Martins, 2015.
2. G. R. Marczyk, D. DeMatteo, and D. Festinger, *Essentials of Research Design and Methodology,* John Wiley & Sons, Inc, 2005.

CS4099D PROJECT: Part II

Prerequisites: CS4098D Project: Part I

L	T	P	C
-	-	16	8

Course Outcomes:

Students will be able to :

Students will :

CO1: Design a solution for the problem identified in part I of the project.
CO2: Implement a solution and collect the observations and results of the work.
CO3: Document the design, and the results and observations from the work into a report.

The team identified in Part I of the project will design a solution to the problem identified in the first semester. The solution shall be implemented and the results, observations and conclusions tabulated. The design, results and conclusions will be documented to form a project report which shall be presented.

References:

1. G. J. Alred, C. T. Brusaw, and W. E. Oliu, *The Handbook of Technical Writing*, 11/e, Bedford/St. Martins, 2015.
2. G. R. Marczyk, D. DeMatteo, and D. Festinger, *Essentials of Research Design and Methodology,* John Wiley & Sons, Inc, 2005.

CS2007D ENVIRONMENTAL STUDIES

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Define and describe a system component or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety and sustainability.
CO2: State the fundamental environmental science and engineering principles necessary to solve equitable use of resources and sustainable lifestyles.
CO3: Analyze environmental pollution and methods of prevention like air, water, soil, marine, thermal, nuclear and noise pollution and waste management.
CO4: Analyze human population and human rights

Module 1: (10 Hours)

Definition, scope and importance - renewable and non-renewable resources - Natural resources - forest, water, mineral, food and energy and land resources - study of problems - Role of individual in conservation - equitable use of resources and sustainable lifestyles.

Module 2: (10 Hours)

Ecosystems - structure and function - producer, consumer and decomposer - energy flow - ecological succession- food chains- forest, grassland, desert and aquatic ecosystems - Biodiversity and conservation.

Module 3: (10 Hours)

Environmental pollution - air, water, soil, marine, thermal, nuclear and noise pollution- methods of prevention - waste management - disaster management - environmental ethics - sustainable development models - water conservation - climate change and global warming - ozone layer depletion - nuclear holocaust - case studies - consumerism and waste products.

Module 4: (9 Hours)

Human population and environment - family welfare - human health and environment - human rights.

References:

1. E. Bharucha, *Environmental Studies*, Universities Press, 2005.
2. UGC Syllabus on environmental studies available at http://www.ugc.ac.in/inside/syllabus.html accessed on 12-04-2018.

CS4021D NUMBER THEORY AND CRYPTOGRAPHY

Prerequisites : NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Solve number theoretic problems and understand their role in cryptosystems.
CO2: Apply cryptographic techniques for encryption, hashing, and signature.
CO3: Construct mathematical arguments about the security of the cryptosystem.

Module 1: (12T+5P Hours)

Divisibility theory in integers: Extended Euclid's algorithm. Modular Arithmetic – exponentiation and inversion. Fermat's Little Theorem, Euler's Theorem. Solution to congruences, Chinese Remainder Theorem.

Module 2: (15T+15P Hours)

Review of abstract algebra: Study of Ring Zn, multiplicative group Zn* and finite field Zp – Gauss Theorem (cyclicity of Zp*) - Quadratic Reciprocity.
Primality Testing – Fermat test, Carmichael numbers, Solovay Strassen Test, Miller Rabin Test - detailed analysis.

Module 3: (12T+6P Hours)

Notions of security: Introduction to one secret key cryptosystem (DES) and one cryptographic hash scheme (SHA). Public Key Cryptosystems – Diffie Hellman Key Agreement Protocol, Knapsack crypto systems, RSA. Elgamal encryption and signature scheme. Key Management Protocols

References:

1. H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications*, Springer-Verlag, 2002.
2. S. Vaudenay, *A Classical Introduction to Cryptography: Applications for Communications Security*, Springer, 2009.
3. B. A. Forouzan and D. Mukhopadhyay, *Cryptography and Network Security,* 2/e*,* Tata McGraw Hill, 2010.

CS4022D PRINCIPLES OF PROGRAMMING LANGUAGES

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain the fundamental concepts and constructs in a programming language.
CO2: Develop formal semantics for programming language constructs.
CO3: Model Programming Language features using Lambda Calculus.
CO4: Design type systems for language safety.

Module 1: (10T+7P Hours)

Programming Languages: Concepts and Constructs. Untyped Arithmetic Expressions – Introduction, Semantics, Evaluation.

Module 2: (10T+6P Hours)

Untyped Lambda Calculus – Basics, Semantics. Programming in Lambda Calculus.

Module 3: (10T+7P Hours)

Typed Arithmetic Expressions – Types and Typing relations, Type Safety.
Simply Typed Lambda Calculus – Function types, Typing relations, Properties of typing.

Module 4: (9T+6P Hours)

Extensions to Simply Typed Lambda Calculus – Unit type, Let bindings, Pairs, Records, Sums, Variants, References, Exceptions.

References:

1. B. C. Pierce, *Types and Programming Languages*, MIT Press, 2002.
2. D. A. Schmidt, *Programming Language Semantics. In Allen B. Tucker, Ed. Handbook of Computer Science and Engineering,* CRC Press, 1996.
3. L. Cardelli, *Type Systems. In Allen B. Tucker, Ed. Handbook of Computer Science and Engineering*, CRC Press, 1996.
4. M. L. Scott, *Programming Language Pragmatics*, 2/e, Elsevier, 2004.

CS4024D INFORMATION THEORY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Estimate the entropy of a given discrete source and design appropriate lossless compression schemes.
CO2: Estimate the capacity of various discrete channel models.
CO3: State and prove simple mathematical properties of secure encryption.

Module 1: (10 Hours)

Foundations: Review of probability theory, entropy and information, random sources, i.i.d and Markov sources, discrete finite state stationary Markov sources, Entropy rate of stationary sources, Computation of stationary distributions.

Module 2: (10 Hours)

Source Coding: Prefix and uniquely decodable codes - Kraft's and Macmillan's inequalities - Shannon's source coding theorem - Shannon Fano code, Huffman code - optimality.

Module 3: (10 Hours)

Channel Coding: BSC and BEC channel models - Channel capacity - Shannon's channel coding theorem - existence of capacity achieving codes for BEC, Fano-Elias Inequality.

Module 4: (9 Hours)

Cryptography: Information theoretic security - Perfect secrecy - Shannon's theorem - perfectly secret codes - Introduction to computational security and pseudo random sources.

References:

1. T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Addison Wesley, 1999.
2. D. J. Mackay, *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, 2002.
3. H. Delfs and H. Knebl, *Introduction to Cryptography,* 2/e*,* Springer, 2010.

CS4025D RANDOMIZED ALGORITHMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Apply probabilistic models to analyze the expected running time of algorithms.
CO2: Analyze the error probability and time complexity of randomized algorithms and classify problems into Randomized Complexity Classes.
CO3: Prove correctness of simple randomized algorithms for combinatorial and algebraic problems.
CO4: Design a deterministic algorithm from a given randomized algorithm, using the method of conditional expectations.

Module 1: Basics (10T Hours)

Sample Space, Events, Conditional probability, Independent events, Random Variables, Linearity of Expectation, Probabilistic method, Markov and Chebyshev inequalities, Moments of a Random variable, Chernoff bounds, Martingales and Azuma's inequality.

Module 2: (10T+10P Hours)

Randomized data structures and algorithms : Skip lists, Hashing, Randomized min-cut, Verifying matrix multiplication, Randomized quicksort, Randomized selection, Coupon Collector's algorithm, Randomized pattern matching. Number theoretic algorithms: Primality testing -- Miller Rabin test.

Module 3: (9T+7P Hours)

Markov chains and random walks, stationarity, Markov chain Monte Carlo (MCMC) methods volume estimation. Randomized Complexity Classes.

Module 4: (10T+9P Hours)

Probabilistic Method and Derandomization, The basic counting argument, The Expectation argument, Derandomization using conditional expectation, Sample and Modify, The Second Moment method, Lovasz Local Lemma, Explicit Constructions using the Local Lemma, Schwartz–Zippel lemma.

References:

1. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
2. M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and probabilistic techniques in algorithms and data analysis,* 2/e, Cambridge University Press, 2017.
3. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.

CS4026D COMBINATORIAL ALGORITHMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Formulate the primal and dual LP for a given optimization problem specification.
CO2: Analyze correctness and complexity of classic algorithms like Ford-Fulkerson algorithm and Edmond's Blossom algorithm.
CO3: Formulate matching and network flow problems in graph theory as integer linear programs and establish duality relations between them.
CO4: Design approximation algorithms using randomized rounding and primal-dual schema.

Module 1: (10T+7P Hours)

Network Flows: Review of graph theory – spanning trees, shortest paths. Connectivity, Network Flows - Max flow min cut theorem, algorithms of Ford-Fulkerson, Edmond Karp, preflow-push algorithms.

Module 2: (10T+5P Hours)

Primal Dual Theory: Linear programming, Primal dual theory, LP-duality based algorithm design. Applications to Network flow and other combinatorial problems, Applications to graph theory - Konig's theorem, Hall's theorem, Menger's theorem.

Module 3: (9T+7P Hours)

Matching Theory: Tutte's theorem, Primal dual algorithms – Hungarian algorithm, Hopcroft Karp algorithm, Edmonds' 'Blossom' maximum matching algorithm for general graphs. Application to marriage problems.

Module 4: (10T+7P Hours)

Approximation: Primal Dual approximation algorithms for set cover, Maximum satisfiability, Steiner tree, multicut, Steiner forest, sparsest cut and k-medians.

References:

1. D. West, *Graph Theory*, Prentice Hall, 2002.
2. D. Jungnickel, *Graphs Networks and Algorithms*, Springer 2005.
3. U. Vazirani, *Approximation Algorithms*, Springer 2003.
4. D. P. Williamson and D. B. Shmoys, *The Design of Approximation Algorithms*, Cambridge University Press, 2011.

CS4027D TOPICS IN ALGORITHMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Illustrate and analyze Linear Algebra based algorithms for LUP decomposition and matrix inversion.
CO2: Analyze the performance and compare advanced data structures like binary search trees, B tree, Splay trees, Skip lists, Red black tree, Fibonacci heaps and Splay trees.
CO3: Prove the correctness and running time of convex hull algorithms.
CO4: Discuss improved algorithms for classic problems like integer sorting, shortest paths, minimum spanning tree and network flows.

Module 1: (10T+7P Hours)

Computational Linear Algebra: LUP decomposition, matrix inversion, Strassen's algorithm, least squares approximation. Fourier transform: Discrete Fourier transform and the fast fourier transform algorithm.

Module 2: (10T+6P Hours)

Advanced data structures: Balanced binary search trees, B tree, Splay trees, Skip lists, Red black tree. Fibonacci heaps and self-adjusting search trees, Splay trees, linking and cutting trees. Universal hashing.

Module 3: (10T+6P Hours)

Geometric algorithms: Selection algorithms and application to convex hull, Ultimate convex hull algorithm, linear programming in two and three dimensions.

Module 4: (9T+7P Hours)

Integer sorting and improved algorithms for shortest paths and minimum spanning tree. Preflow-push algorithms and Scaling algorithms for network flow problems.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3/e, Prentice Hall India, 2010.
2. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
3. D. C. Kozen, *The Design and Analysis of Algorithms*, Springer Verlag N.Y, 1992.

CS4028D QUANTUM COMPUTATION

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: To design simple circuits with quantum gates.
CO2: Prove correctness of simple quantum circuits.
CO3: Mathematically analyze the functioning of standard quantum algorithms like Grover search and Shor's algorithm.
CO4**:** Mathematically analyze standard quantum error correcting codes.

Module 1: (10 Hours)

Review of Linear Algebra. The postulates of quantum mechanics. Review of Theory of Finite Dimensional Hilbert Spaces and Tensor Products, spectral theorem for Hermitian and Normal operators.

Module 2: (10 Hours)

Complexity classes. Models for Quantum Computation. Qubits. Single and multiple qubit gates. Quantum circuits. Bell states. Single qubit operations. Controlled operations and measurement. Universal quantum gates. Quantum Complexity classes and relationship with classical complexity classes

Module 3: (10 Hours)

Quantum Algorithms – Quantum search algorithm - geometric visualization and performance.
Quantum search as a quantum simulation. Speeding up the solution of NP Complete problems.
Quantum search as an unstructured database. Grover’s and Shor’s Algorithms.

Module 4: (09 Hours)

Introduction to Quantum Coding Theory. Quantum error correction. The Shor code. Discretization of errors, Independent error models, Degenerate Codes. The quantum Hamming bound. Constructing quantum codes – Classical linear codes, Shannon entropy and Von Neumann Entropy.

References:

1. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information,* Cambridge, UK, Cambridge University Press, 2002.
2. J. Gruska, *Quantum Computing*, McGraw Hill, 1999.
3. A. Peres, *Quantum Theory: Concepts and Methods,* Springer, 1993.

CS4029D TOPICS IN COMPLEXITY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Develop proofs of simple complexity theoretic properties.
CO2: Prove inclusion relationships between various complexity classes.
CO3: Analyze proofs of standard results in complexity theory.

Module 1: (13 Hours)

Hardness of approximation: Probabilistically checkable proofs, PCP theorem (no proof), Proofs for weaker versions of PCP theorem, Hardness of approximation, gap reductions, hardness results for 3SAT, Clique, vertex cover etc. reductions using expander graphs,

Module 2: (13 Hours)

Circuit lower bounds - Switching lemma, classes ACC0 and TC0. Circuit lower bound for PARITY and MAJORITY. Inequivalence of ACC0 and NEXP (no proof).

Module 3: (13 Hours)

Cryptography - one-way functions and pseudo random generators, Yao's theorem, Goldreich-Levin theorem, zero knowledge proofs.

References:

1. S. Arora and B. Barak, *Computational Complexity: A Modern Approach*, Cambridge University Press, 2009.
2. D. C. Kozen, *Theory of Computation,* Springer, 2007.
3. C. H. Papadimitriou, *Computational Complexity*, 1/e, Addison Wesley, 1993.

CS4030D COMPUTATIONAL COMPLEXITY

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

Students will be able to :

CO1: Describe the properties of elementary complexity classes.
CO2: Place a given computational problem to the appropriate complexity class and establish connections between problems using reductions
CO3: Prove simple properties connecting complexity classes applying known relationships between the classes.

Module 1: (14 Hours)

Review of Complexity Classes: L, NL, P, NP, PSPACE and EXP, log-space and polynomial-time reductions, completeness, Hierarchy theorems, Savitch, and Immerman theorems.

Module 2: (14 Hours)

Circuit complexity, P/Poly, NC and AC, P-completeness, polynomial hierarchy, Karp Lipton theorem, alternation, relationship between circuit depth and space complexity.

Module 3: (12 Hours)

Randomized Complexity classes: Adleman's theorem, Sipser Gacs theorem, counting class, #P, Valiant's Theorem, Toda's theorem (no proof).

Module 4: (12 Hours)

Arthur Merlin games, Graph Isomorphism problem, Goldwasser-Sipser theorem, Interactive Proofs, Shamir's theorem.

References:

1. S. Arora and B. Barak, *Computational Complexity: A Modern Approach*, Cambridge University Press, 2009.
2. C. H. Papadimitriou, *Computational Complexity*, 1/e, Addison Wesley, 1993.
3. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
4. V. Vazirani, *Approximation Algorithms*, 1/e, Springer, 2004.

CS4031D COMPUTATIONAL ALGEBRA

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Analyze algorithms for computational problems in number theory, algebra.
CO2: Find programming solutions for simple problems involving algebraic and computational nature.
CO3: Design efficient computational methods for simple algebraic and number theoretic problems

Module 1: (10T+6P Hours)

Number Theory: Review of groups and rings and vector spaces, Euclid's algorithm, Structure of the ring Zn Algorithms for computation in the ring Zn - modular inversion, exponentiation, Chinese remaindering.

Module 2: (10T+6P Hours)

Finite fields: Structure theory of finite fields - Factorization of polynomials over finite fields - Berlekamp's algorithm, Cantor Zassenhaus algorithm, Fourier Transform algorithm for finite fields.

Module 3: (9T+7P Hours)

Primality Testing: Solovay Strassen test, Miller Rabin test, Agrawal, Kayal Saxena algorithm.

Module 4: (10T+7P Hours)

Applications: Euclid's algorithm for rational polynomial approximation and decoding BCH and RS codes. Applications to public key cryptography.

References:

1. V. Shoup, *A computational Introduction to Number Theory and Algebra*, Cambridge University Press, 2005.
2. H. Delfs and H. Knebl, *Introduction to Cryptography,* Springer, 1998.
3. J. von zur Gathen, *Modern Computer Algebra*, Cambridge University Press, 2003.
4. W. C. Huffman and V. Pless, *Fundamentals of Error Correcting Codes*, Cambridge University press, 2003.

CS4032D COMPUTER ARCHITECTURE

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1**:** Analyze the performance of a processor using standard benchmark suites.
CO2: Define Instruction Level Parallelism(ILP), Thread Level Parallelism(TLP),Data level parallelism (DLP) and Request level Parallelism and design mechanisms on hardware and software to exploit ILP,TLP,DLP and RLP.
CO3: Design and optimize the performance of memory hierarchy for modern processors.

Module 1: (10T+7P Hours)

Fundamentals - Technology trend -performance measurement - Comparing and summarizing performance- quantitative principles of computer design - Amdahl's law - instruction set architectures - memory addressing - type and size of operands - encoding an instruction set - role of compilers - Review of pipelining - MIPS architecture Memory hierarchy design - reducing cache misses and miss penalty, reducing hit time Main memory organization - virtual memory and its protection.

Module 2: (10T+7P Hours)

Instruction level parallelism and its limits - static and dynamic scheduling - static and dynamic branch prediction - multiple issue processor - multiple issue with dynamic scheduling-hardware based speculation - limitation of ILP-Multithreading,

Module 3: (10T+7P Hours)

Multiprocessor and thread level parallelism- classification of parallel architecture-models of communication and memory architecture-Symmetric shared memory architecture-cache coherence protocols-distributed shared memory architecture-directory based cache coherence protocol - Memory consistency-relaxed consistency models.

Module 4: (9T +5P Hours)

Data Level Parallelism-Vector processors-SIMD extensions, GPU, GPU and CUDA, Overview of CUDA C; threads, blocks and grids, warps, different GPU memories, Kernel-Based Parallel Programming, Request level parallelism, Domain specific Architecture

References

1. J. L Hennessy and D. A. Patterson, A.*Computer Architecture: A Quantitative approach*, 6/e, Morgan Kaufmann, 2017.
2. D. A. Patterson and J. L. Hennessy, *Computer Organisation and Design: The Hardware/ Software Interface*, 5/e, Harcourt Asia Pte Ltd (Morgan Kaufman), 2014.

CS4033D DISTRIBUTED COMPUTING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Understand the principles and techniques behind the design of distributed systems
CO2: Gain practical experience in designing, implementing and debugging distributed systems.
CO3: Model, Design and develop concurrent and distributed algorithms
CO4: Apply the concepts of self stabilization to model autonomic computing systems

Module 1: (10T+7P Hours)

Characteristics of Distributed Systems, Distributed systems Versus Parallel systems, Models of distributed systems**,** Happened Before and Potential Causality Model, Models based on States, Logical clocks, Vector clocks, Verifying clock algorithms, Direct dependency clocks.

Module 2: (10T+7P Hours)

Mutual exclusion using Timestamps, Distributed Mutual Exclusion (DME) using timestamps, token and Quorums, Centralized and distributed algorithms, proofs of correctness and complexity analysis. Drinking philosophers problem, Dining philosophers problem under heavy and light load conditions. Implementation and performance evaluation of DME algorithms.

Module 3: (10T+7P Hours)

Leader election algorithms, Global state detection, Global predicates, Termination Detection, Control of distributed computation, disjunctive predicates. Performance evaluation of leader election algorithms on simulated environments.

Module 4: (9T+5P Hours)

Self stabilization, knowledge and common knowledge, Distributed consensus, Consensus under Asynchrony and Synchrony, Checkpointing for Recovery, R- Graphs

References:

1. V. K. Garg, *Elements of Distributed Computing*, Wiley & Sons, 2002.
2. S. Ghosh, *Distributed Systems An Algorithmic Approach*, Chapman & Hall, CRC Computer and Information Science Series, 2006.
3. A. S. Tanenbaum and M. V. Steen, *Distributed Systems: Principles and Paradigms*, 2/e, Pearson, 2007.
4. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, *Distributed Systems: Concepts and Design*, 5/e, Addison Wesley, 2011.
5. R. Chow and T. Johnson, *Distributed Operating Systems and Algorithms*, Addison Wesley, 2002.

CS4034D MIDDLEWARE TECHNOLOGIES

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Model, Design and develop middleware for distributed applications
CO2: Develop Distributed Event Based Systems for loosely coupled distributed systems
CO3:.Develop algorithms for adaptive route optimization with QoS Guarantees

Module 1: (10T+7P Hours)

Publish/Subscribe matching algorithm, event based systems, notification filtering mechanisms, Composite event processing, content based routing, content based models and matching, matching algorithms, distributed hash tables (DHT)

Module 2: (10T+7P Hours)

Distributed notification routing, content based routing algorithms, engineering event based systems, Accessing publish/subscribe functionality using APIs. Scoping, event based systems with scopes, notification mappings, transmission policies, implementation strategies for scoping.

Module 3: (9T+5P Hours)

Composite event detection, detection architectures, security, fault tolerance, congestion control, mobility, existing notification standards- JMS, DDS, HLA.

Module 4: (10T+7P Hours)

Topic based systems, Overlays, P2P systems, overlay routing, Case studies- REBECA, HERMES, Gryphon. Commercial systems- IBM Websphere MQ, TIBCO Rendezvous.

References:

1. G. Muhl, L. Fiege, and P. R. Pietzuch, *Distributed Event Based Systems,* Springer, 2006.
2. C. Britton and P. Bye, *IT Architectures and Middleware*, 2/e, Pearson Education, 2005.
3. Y. Diao and M. J. Franklin, *Query Processing for High-Volume XML Message Brokering*, VLDB, 2003.
4. C. Chan, M. Garofalakis, and R. Rastogi, *RE-Tree: An Efficient Index Structure for Regular Expressions*, VLDB, 2002.

CS4035D COMPUTER SECURITY

Prerequisites: CS4021D Number Theory and Cryptography

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Identify the confidentiality, integrity, availability requirements for a given information domain.
CO2: Analyze basic security loopholes and develop defenses against few threats.
CO3: Develop solutions for security issues in current domains

Module 1: (15T+0P Hours)

Review of Cryptography Fundamentals - Cryptographic Protocols for Authentication - Authentication Protocols: One way and Mutual Authentication, Challenge Response protocols, Lamport's scheme, Needham Schroeder protocol. Interactive proof systems, Zero Knowledge Proof systems – soundness and completeness – Fiat-Shamir identification scheme.
Access Control – MAC, DAC, RBAC. Formal models of security - BLP, Biba, Chinese Wall and Clark Wilson. Overview of SELinux.

Module 2: (15T+18P Hours)

Network Security - Security at different layers – IPSec / SSL-TLS / PGP.
Principles of Secure Design - Software vulnerabilities - Buffer and stack overflow, Phishing. Malware - Viruses, Worms and Trojans.
Security problems in network domain - DoS, DDoS, ARP spoofing and session hijacking. DNS attacks and DNSSEC. Cross-site scripting XSS, SQL injection attacks.
Defense Mechanisms - Intrusion Detection Systems (IDS) and Firewalls.

Module 3: (9T+8P Hours)

Cloud computing architecture - Introduction from a security perspective - guiding principles for designing and implementing appropriate safeguards in the cloud domain.
Security in current applications – Two out of the following four areas may be subjected to a detailed study from the security perspective. Online Banking, Digital Currency, E-Voting, Internet of Things.

References:

1. B. Menezes, *Network security and Cryptography*, Cengage Learning India, 2010.
2. B. A. Forouzan and D. Mukhopadhyay, *Cryptography and Network Security,* 2/e*,* Tata McGraw Hill, 2010.
3. D. Gollmann, *Computer Security*, John Wiley and Sons Ltd., 2006.
4. C. Kaufman, R. Perlman, and M. Speciner, *Network Security,* Pearson India, 2017.

CS4036D ADVANCED DATABASE MANAGEMENT SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Model, Design and develop concurrent, distributed and spatial database applications
CO2: Query spatial databases using spatial query languages.
CO3: Port existing database applications into the cloud database environment.
CO4: Deploy efficient database solutions using free and open software.

Module 1: (10T+7P Hours)

Distributed database concepts - overview of client - server architecture and its relationship to distributed databases, Concurrency control Heterogeneity issues, Persistent Programming Languages, Object Identity and its implementation, Clustering, Indexing, Client Server Object Bases, Cache Coherence.

Module 2: (10T+7P Hours)

Parallel Databases: Parallel Architectures, performance measures, shared nothing/shared disk/shared memory based architectures, Data partitioning, Intra-operator parallelism, Pipelining, Scheduling, Load balancing, Query processing- Index based, Query optimization: cost estimation, Query optimization: algorithms, Online query processing and optimization, XML, DTD, XPath, XML
indexing, Adaptive query processing

Module 3: (10T+7P Hours)

Advanced Transaction Models: Savepoints, Sagas, Nested Transactions, Multi Level Transactions. Recovery: Multi-level recovery, Shared disk systems, Distributed systems 2PC, 3PC, replication and hot spares, Data storage, security and privacy- Multidimensional K- Anonymity, Data stream management.

Module 4: (9T+5P Hours)

Models of Spatial Data: Conceptual Data Models for spatial databases (e.g. pictogram enhanced ERDs), Logical data models for spatial databases: raster model (map algebra), vector model, Spatial query languages, Need for spatial operators and relations, SQL3 and ADT. Spatial operators, OGIS queries

References:

1. A. Silberschatz, H. Korth, and S. Sudarshan. *Database System Concepts*, 5/e, McGraw Hill, 2005.
2. S. Shekhar and S. Chawla. *Spatial Databases: A Tour*, Prentice Hall, 2003.
3. R. H. Guting and M. Schneider, *Moving Objects Databases,* Morgan Kaufman, 2005.
4. R. Elmasri and S. Navathe, *Fundamentals of Database Systems,* 5/e, Benjamin- Cummings, 2007.

CS4037D CLOUD COMPUTING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Apply the concepts of virtualization technology on distributed systems to develop cloud based systems.
CO2: Propose appropriate solutions to cloud computing problems using technologies like Hadoop MapReduce
CO3: Develop optimized solutions for problems like resource allocation and utility billing.
CO4: Evolve methodologies for migrating parallel and distributed systems into the cloud model.

Module 1: (10T+7P Hours)

New Computing Paradigms & Services: Cloud computing , Edge computing , Grid computing , Utility computing , Cloud Computing Architectural Framework, Cloud Deployment Models, Virtualization in Cloud Computing, Parallelization in Cloud Computing, Security for Cloud Computing, Cloud Economics , Metering of services.

Module 2: (10T+7P Hours)

Cloud Service Models: Software as a Service (SaaS), Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Service Oriented Architecture (SoA), Elastic Computing, On Demand Computing, Cloud Architecture, Introduction to virtualization.

Module 3: (10T+6P Hours)

Types of Virtualization, Grid technology , Browser as a platform, Web 2.0, Autonomic Systems, Cloud Computing Operating System, Deployment of applications on the cloud, Case studies- Xen, VMware, Eucalyptus, Amazon EC2.

Module 4: (9T+6P Hours)

Introduction to Mapreduce, Information retrieval through Mapreduce, Hadoop File System, GFS, Page Ranking using Map Reduce, Security threats and solutions in clouds, mobile cloud computing, Case studies- Ajax, Hadoop.

References:

1. T. White, *Hadoop: The Definitive Guide*, O'Reilly Media, 2009
2. J. Venner, *Pro Hadoop*, Apress, 2009
3. T. Chou, *Introduction to cloud computing & Business*, Active Book Press, 2010

CS4038D DATA MINING

Prerequisites: CS3002D Database Management Systems

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Apply data preprocessing and visualization techniques on a dataset
CO2: Evaluate various data mining models to select the best model for pattern identification
CO3: Apply data mining model to solve real world problems

Module 1: (10T+7P Hours)

Introduction to data mining-challenges and tasks Data preprocessing data analysis, measures of similarity and dissimilarity, Data visualization –concepts and techniques

Module 2: (10T+7P Hours)

Classification- decision tree-performance evaluation of the classifier, comparison of different classifiers, Rule based classifier, Nearest-neighbor classifiers-Bayesian classifiers-support vector machines, Class imbalance problem

Module 3: (10T+6P Hours)

Association analysis –frequent item generation rule generation, evaluation of association patterns

Module 4: (9T+6P Hours)

Cluster analysis,-types of clusters, K means algorithm, cluster evaluation, application of data mining to web mining and Bioinformatics

References:

1. P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Pearson Education 2006.
2. J. Han and M. Kamber, *Data Mining: Concepts and Techniques,* 2/e, Morgan Kaufmann, 2005.
3. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning - Data Mining, Inference, and Prediction,* 2/e, Springer, California, 2008.

CS4039D MULTI AGENT SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Define an agent and model agent-based solutions for distributed problem solving.
CO2: Use software tools for modeling multi-agent based solutions.
CO3: Design of protocols for agent-agent communications.
CO4: Design of different negotiation models: game theoretical, heuristic approach and argumentation based approach.
CO5: Formulate solutions for distributed decision-making and design criteria for the evaluation of decisions.

Module 1: (10T+7P Hours)

Introduction to agent and multi-agent systems, different types of agents, abstract architecture, distributed problem solving, application areas, Software tools for modeling Multi-Agent Systems

Module 2: (10T+7P Hours)

Agent communication, communication languages KQML and FIPA ACL Communication policies and protocols, Protocol Modeling

Module 3: (10T+6P Hours)

Negotiation in multi-agent- agent environment, game theoretical model , heuristic approach, argumentation based approach

Module 4: (9T+6P Hours)

Distributed decision making – Evaluation criteria - Social welfare, Pareto Efficiency, Individual Rationality, Stability, Application of multiagent systems in complex distributed problem solving, Modeling distributed multi-agent systems.

References:

1. M. Wooldridge, *An Introduction to multiagent systems*, Wiley, 2009.
2. R. Norvig, *Artificial Intelligence: A modern approach*, Prentice Hall, 2010.

CS4040D BIOINFORMATICS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Define and describe the importance of biomolecules like DNA, RNA and Protein
CO2: Abstract and formalize Bioinformatics problems
CO3: Apply various algorithm design and analysis techniques to solve Bioinformatics problems
CO4: Implement various Bioinformatics algorithms
CO5: Apply different Bioinformatics tools and databases

Module 1: (10T+7P Hours)

Molecular Biology primer, gene structure and information content, Bioinformatics tools and databases, genomic information content, Sequence Alignment, Algorithms for global and local alignments, Scoring matrices, Dynamic Programming algorithms.

Module 2: (10T+7P Hours)

Introduction to Bio-programming Languages, Motif finding, Gene Prediction, Molecular Phylogenetics, Phylogenetic trees, Algorithms for Phylogenetic Tree construction.

Module 3: (9T+6P Hours)

Combinatorial Pattern Matching, Repeat finding, Keyword Trees, Suffix Trees, Heuristic similarity search algorithms, Approximate Pattern Matching.

Module 4: (10T+6P Hours)

Microarrays, Gene expression, Algorithms for Analyzing Gene Expression Data, Protein and RNA structure prediction, Algorithms for Structure Prediction, Emerging Trends in Bioinformatics Algorithms and Databases.

References:

1. N. C. Jones and P. A. Pevzner, *An Introduction to Bioinformatics Algorithms*, MIT Press, 2004.
2. D. E. Krane and M. L. Raymer, *Fundamental Concepts of Bioinformatics*, Pearson Education, 2003.
3. T. K. Attwood and D. J. Parry-Smith, *Introduction to Bioinformatics,* Pearson Education, 2003.
4. Current Literature.

CS4041D NATURAL LANGUAGE PROCESSING

Prerequisites: CS2005D Data Structures and Algorithms

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Define and describe the various challenges associated with natural language understanding such as syntactic ambiguity, semantic ambiguity, discourse analysis and pragmatics.
CO2: Relate fundamental mathematical principles of probability theory, statistics and linear algebra and basic computer science principles like dynamic programming and parsing.
CO3: Explain and apply various important tasks in natural language processing such as language modeling, information extraction, named entity recognition, information retrieval, text classification, word sense disambiguation, automatic question answering and text summarization.
CO4: Use natural language processing (NLP) tools and libraries (such as python, nltk) and develop softwares for various NLP tasks such as tagging, parsing and text classification.
CO5: Recall fundamental grammatical structure of natural languages, with a special emphasis on English grammar.

Module 1: (10T+7P Hours)

Introduction to Natural Language Processing, Different Levels of language analysis, Representation and understanding, Linguistic background. Grammars and parsing, Top down and Bottom up parsers.

Module 2: (9T+6P Hours)

Transition Network Grammars, Feature systems and augmented grammars, Morphological analysis and the lexicon, Parsing with features, Augmented Transition Networks.

Module 3: (10T+7P Hours)

Grammars for natural language, Movement phenomenon in language, Handling questions in context free grammars, Hold mechanisms in ATNs, Gap threading, Human preferences in parsing, Shift reduce parsers, Deterministic parsers, Statistical methods for Ambiguity resolution

Module 4: (10T+6P Hours)

Semantic Interpretation, word senses and ambiguity, Basic logical form language, Encoding ambiguity in logical from, Thematic roles, Linking syntax and semantics, Information Retrieval, Recent trends in NLP.

References:

1. J. Allen, *Natural Language Understanding,* 2/e, Pearson Education, 2003.
2. T. Siddiqui and U. S. Tiwary, *Natural Language Processing and Information Retrieval*, Oxford University Press, 2008.
3. D. Jurafsky and J. H. Martin, *Speech and Language Processing,* Pearson Education, 2000.

CS4042D WEB PROGRAMMING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Analyze the complexity of a problem and propose a web based solution. .
CO2: Design and develop of web applications in languages like PHP and RoR
CO3: Develop Web Services

Module 1: (10T+7P Hours)

Internet and its architecture, Client Server Architecture- Creating an Internet Client, Application Protocols and http, Presentation aspects html, CSS and Javascript, Creating a web server, Serving Dynamic Content- CGI – overview of technologies like PHP, AJAX.

Module 2: (10T+7P Hours)

Web development frameworks – Detailed study of one open source web framework - Ruby Scripting, Ruby on rails – Design, Implementation and Maintenance aspects.

Module 3: (10T+6P Hours)

Designing Web Services- Web Technologies- SOAP, WSDL, UDDI, BPEL- Implementing a simple webservice, RESTful web service.

Module 4: (9T+6P Hours)

Shared memory synchronization, Performance measurement and workload models. Comparison using existing benchmarks. Designing web solutions defensively - evaluation of threats and mitigation strategy

References:

1. D. Thomas, C. Fowler, and A. Hunt, *Programming Ruby: The Pragmatic Programmer's Guide,* 3/e, Pragmatic Programmers, May 2008.
2. L. Richardson and S. Ruby, *Restful Web Services*, 1/e, O Reilly, 2007.

CS4043D IMAGE PROCESSING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain and Apply the fundamental concepts of image processing.
CO2: Design solutions for simple and complex image processing problems using the various spatial domain and transform domain techniques.
CO3: Apply various filtering techniques and segmentation methods for image enhancement and image segmentation.

Module 1: (10T+8P Hours)

Fundamentals of Image processing: Digital image representation, Elements of Digital image processing systems, Image model, Sampling and Quantization, Basic relations between pixels.
Image transforms: One dimensional Fourier transform, Two dimensional Fourier transform, Properties of two dimensional Fourier transform. Walsh transform, Hadamard transform, Discrete cosine transform, Haar transform, Slant transform.

Module 2: (10T+6P Hours)

Image enhancement techniques: Spatial domain methods, Frequency domain methods, Intensity transform, Histogram processing, Image subtraction, Image averaging, Smoothing filters, Sharpening filters, Spatial masks from frequency domain.

Module 3: (10T+6P Hours)

Image Segmentation: Thresholding: Different types of thresholding methods, Point detection, Edge detection: Different types of edge operators, Line detection, Edge linking and boundary detection, Region growing, Region splitting, Region Merging.

Module 4: (9T+6P Hours)

Image Data Compression: Fundamentals, Compression models, Error free compression, Lossy Compression, Image compression standards.
Applications of Image Processing: Medical imaging, Robot vision, Character recognition, Remote Sensing.

References:

1. R. C. Gonzalez and R. E. Woods*, Digital Image Processing*, Addison-Wesley Publishing Company, 2007.
2. A. K. Jain, *Fundamentals of Digital Images Processing*, Pearson Education India, 2015

CS4044D MACHINE LEARNING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Differentiate the kinds of Pattern Recognition tasks such as supervised, unsupervised learning, classification and regression.
CO2: Describe linear classifiers like logistic regression, least squares classifier and perceptron, and non-linear classifiers like SVM and artificial neural networks.
CO3: Use classification tools and libraries (such as weka, iPython notebook) for solving real world pattern recognition problems.

Module 1: (10T+7P Hours)

Introduction: Machine Perception , Pattern Recognition Systems, The Design Cycle, Learning and Adaptation.
Bayes Decision Theory: Bayes Decision Theory, Minimum Error rate Classification, Classifiers, Discriminant functions and Decision Surfaces, Normal Density, Discriminant functions for the Normal Density, Bayes Decision Theory for Discrete features

Module 2: (10T+7P Hours)

Parameter Estimation :Maximum Likelihood Estimation,Maximum A Posteriori Estimation and Bayesian Parameter Estimation. Gaussian Case and General Theory.Non Parametric Techniques: Density Estimation, Parzen Windows , K- Nearest Neighbor Estimation,NN rule, Metrics and NN Classification, Unsupervised Methods - Clustering Algorithms- K Means, Gaussian Mixture Models, Fuzzy Classification.

Module 3: (10T+7P Hours)

Linear Discriminant Functions : Linear Discriminant Functions and Decision Surfaces, Generalized Discriminant Functions, The two-category linearly separable case , Minimizing the perceptron criterion function, relaxation procedures, non- separable behavior, Minimum Squared- Error procedures.
Linear Methods : Linear regression, logistic regression, Principal Component Analysis,Fisher's Linear Discriminant Analysis.
Non-linear methods - Kernel Methods - Kernel version of PCA, LDA,SVMs

Module 4: (9T+5P Hours)

Multi Layer Neural Networks : Feedforward Operation, Classification, Back – propagation Algorithm, Error Surfaces, Back-propagation as Feature mapping, COnvolutional Neural Networks and Deep Learning.

References:

1. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification* , John-Wiley, 2004.
2. J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*, Wiley, 1974.
3. C. M. Bishop, *Pattern Recognition and Machine Learning,* Springer, 2006.

CS4045D MEDICAL IMAGE PROCESSING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain and Apply the fundamental concepts of image processing.
CO2: Design solutions for simple and complex medical image processing problems using the various spatial domain and transform domain techniques.
CO3: Apply various filtering techniques and segmentation methods for medical image enhancement, medical image segmentation and thus for efficient medical image analysis.

Module 1: (10T+7P Hours)

Introduction to digital image processing: images, image quality, basic operations.
Radiography: Introduction, X-rays, interaction with matter, detectors, dual energy imaging, quality clinical use, biologic effect and safety, Fourier Slice Theorem Basics.

Module 2: (10T+7P Hours)

X-ray Computed tomography: Introduction, X-ray detectors in CT, imaging, cardiac CT, image quality, clinical use, biologic effects and safety.
Magnetic resonance imaging: Introduction, physics of transmitted signal, interaction with tissue, signal detection and detector, imaging. Biologic effects and safety

Module 3: (10T+7P Hours)

Nuclear imaging, Introduction, radionuclides, interaction of Gamma-photons and particles with matter, data acquisition, imaging, image quality, equipment, clinical use, biologic effects and safety
Ultrasound imaging: Physics of acoustic waves, generation and detection of ultrasound, grayscale imaging, Doppler imaging, image quality, equipment, clinical use, biologic effects and safety.

Module 4: (9T+5P Hours)

Medical image analysis: Manual and automated analysis, computation strategies for automated medical image analysis, pixel classification.

References:

1. P. Suetens, *Fundamentals of medical imaging*, Cambridge University Press, 2009.
2. J. A. Bushberg et. al., *The Essential Physics of Medical Imaging,* 2/e, L. Williams and Wilkins, 2002.

CS4046D COMPUTER VISION

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain and apply the fundamentals of image formation and color representation.
CO2: Apply different techniques to find edges, corners, shape and texture of objects in images.
CO3: Learn to use various segmentation and object detection techniques.

Module 1: (9T+ 6P Hours)

Introduction and overview, pinhole cameras, radiometry terminology. Sources, shadows and shading: Local shading models- point, line and area sources; photometric stereo. Color: Physics of color; human color perception, Representing color; A model for image color; surface color from image color.

Module 2: (10T+ 8P Hours)

Linear filters: Linear filters and convolution; shift invariant linear systems- discrete convolution, continuous convolution, edge effects in discrete convolution; Spatial frequency and fourier transforms; Sampling and aliasing; filters as templates; Normalized correlations and finding patterns. Edge detection: Noise; estimating derivatives; detecting edges. Texture: Representing texture; Analysis using oriented pyramid; Applications; Shape from texture. The geometry and views: Two views.

Module 3: (10T+ 6P Hours)

Stereopsis: Reconstruction; human stereo; Binocular fusion; using color camera.Structure from Motion;Motion Segmentation by Parameter Estimation

Module 4: (10T+ 6P Hours)

Segmentation by clustering: Human vision, applications, segmentation by graph theoretic clustering. Segmentation by fitting a model, Hough transform; fitting lines, fitting curves;Object Detection and Classification.

References:

1. D. A. Forsyth and J. Ponce, *Computer Vision- A modern approach*, Pearson education series, 2015.
2. R. J. Schalkoff, *Digital Image Processing and Computer Vision*, John Wiley, 2004.
3. R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer-Verlag London Limited 2011.
4. E. R. Davies, *Machine Vision:Theory Algorithms and Practices,* Elsevier, 2004.

CS4047D COMPUTER GRAPHICS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Develop techniques for 3D modelling, image synthesis and rendering.
CO2 : Design lighting and shading techniques
CO3 : Devise techniques to produce 3D models using graphical software.

Module 1: (9T+5P Hours)

Graphics Pipeline - overview of vertex processing, primitive generation, transformations and projections, clipping, rasterization, fragment processing - Graphics Hardware - overview of GPU architecture, how GPUs SIMD architecture suits computer graphics.

Module 2: (10T+7P Hours)

Coordinate Systems - representations, homogeneous coordinates, object, camera, world, and screen coordinate system, changing coordinate systems. Transformations - affine transformations, translation, rotation, scaling in homogeneous coordinates, matrix representations, cumulation of transformations. Viewing and Projections - orthographic and perspective projection, camera positioning, Hidden Surface Removal - its importance in rendering, z buffer algorithm, clipping, culling, Data Structures for efficient implementation of the transformations and projections.

Module 3: (10T+7P Hours)

Lighting and Shading - light sources, normal computation, reflection models, flat and smooth shading , Introduction to Textures and Mapping - Rendering Techniques - slicing, volume rendering, iso-surface extraction, ray casting, multiresolution representations for large data rendering. Data Structures for efficient implementation.

Module 4: (10T+7P Hours)

Geometric Modelling - Data structures - tree representations, hierarchical models, scene graphs - particle systems and representations - introduction to modeling and solving dynamics based on physics, Introduction to Curves Surfaces (Bezier, splines) and Meshes - structured and unstructured.

References:

1. E. S. Angel, *Interactive Computer Graphics, A top-down approach with OpenGL*, 5/e, Pearson Education, 2009.
2. D. Hearn and M. P. Baker, *Computer Graphics with OpenGL*, 3/e, Prentice Hall, 2003.
3. S. Marschner and P. Shirley, *Fundamentals of Computer Graphics*, 4/e, CRC Press, 2015.

CS4048D MATHEMATICAL FOUNDATIONS OF MACHINE LEARNING

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total Hours: 39

Course Outcomes:

Students will be able to :

CO1: Apply the fundamental concepts of Probability and Linear Algebra
CO2: Explain and apply Multivariate analysis and Optimization Techniques
CO3: Use the mathematical foundations for learning Machine Learning Concepts

Module 1: (09 hours)

Review of Probability Theory: Discrete and Continuous Random Variables, Joint and Marginal Distributions, Markov, Chebyshev, Jensen and Hausdorff Inequalities, Law of Large Numbers, Central Limit Theorem (No proof).
Classification and Estimation: Bayes classifier, maximum likelihood and Bayesian estimation techniques.

Module 2: (10 hours)

Review of Linear Algebra: Vector spaces, Rank Nullity Theorem, Metric and Normed Linear Spaces, Inner product spaces, Gram Schmidt Orthogonalization, Projections and Orthogonal Projections, Introduction to Hilbert spaces.
Orthogonal Decomposition algorithms: Eigen Decomposition, Singular Value Decomposition, Principal component analysis, LU, QR, Cholesky Decompositions, Least Squares Approximation

Module 3: (10 hours)

Review of Multivariate Analysis: Sequences and Limits, Differentiability, Level Sets and Gradients, multivariate Taylor Series.
Unconstrained Optimization: Conditions for Local Minimizers of Continuously Differentiable Functions, Gradient Search, Analysis of Newton's method, Levenberg-Marquardt Modification, Quasi-Newton Methods, Rank One Correction Formula, DFP and BFGS Algorithms.

Module 4: (10 hours)

Constrained Optimization: Tangent and Normal Spaces, Lagrange Condition, Second-Order-Conditions, Karush-Kuhn-Tucker Condition. Convex Optimization: Lagrange and Fenchel Duality, Proximal Algorithms, ADMM Algorithm.

References:

1. R. S. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, Academic Press, 2014.
2. K. Hoffman and R. Kunze. *Linear Algebra*, 2/e, Prentice Hall of India, 1990.
3. E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, 2/e, John Wiley & Sons, 2004.
4. B. Stephen and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
5. J. Eckstein and W. Yao, *Augmented Lagrangian and Alternating Direction Methods for Convex Optimization: A Tutorial and some Illustrative Computational Results*, *RUTCOR Research Reports* 32, 2012.

CS4049D ADVANCED COMPUTER NETWORKS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Recall classic protocols in computer networks.
CO2: Demonstrate performance of existing protocols by mathematical or simulation tools
CO3: Solve challenges in various layers of TCP/IP protocol stacks.

Module 1: (10T+7P Hours)

Introduction- Internet design philosophy, Layering and end to end design principle. MAC protocols for high-speed LANS, MANs, Wireless LANs and mobile networks, VLAN. Fast access technologies.

Module 2: (10T+7P Hours)

IPv6: Why IPv6, Basic protocol, extensions and options, Support for QoS, Security, Neighbour discovery, Auto-configuration, Routing. Changes to other protocols. Application Programming Interface for IPv6, 6bone. IP Multicasting, Wide area multicasting, Reliable multicast. Routing layer issues, ISPs and peering, BGP, IGP, Traffic Engineering, Routing mechanisms: Queue management, Packet scheduling. MPLS, VPNs

Module 3: (10T+7P Hours)

TCP extensions for high-speed networks, Transaction-oriented applications. New options in TCP, TCP performance issues over wireless networks, SCTP, DCCP.

Module 4: (9T+5P Hours)

DNS issues, other naming mechanisms, Overlay networks, P2P networks, Web server systems, Web 2.0, Internet traffic modelling, Internet measurements. Security – Firewalls, Unified threat Management System, Network Access Control.

References:

1. A. Farrel, *The Internet and its protocols a comparative approach*, Elsevier, 2005
2. M. Gonsalves and K. Niles, *IPv6 Networks,* McGraw Hill, 1998.
3. W. R. Stevens, *TCP/IP Illustrated*, Volume 1: The protocols, Addison Wesley, 1994.
4. G. R. Wright, *TCP/IP Illustrated,* Volume 2: The Implementation, Addison Wesley, 1995.

CS4050D DESIGN AND ANALYSIS OF ALGORITHMS

Prerequisites: CS2005D Data Structures and Algorithms

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Analyze the amortized time complexity of a given algorithm and data structure operations
CO2: Decide the appropriate design methodology for a given problem from among the paradigms of Divide and Conquer, Dynamic Programming, Greedy, Branch and Bound.
CO3: Prove the NP completeness of a given problem by using the technique of many-one reductions.
CO4: Compute the expected running time of a given randomized algorithm using probabilistic models.

Module 1: (10T+7P Hours)

Analysis: RAM model, Asymptotic Analysis, Recurrence relations, Probabilistic analysis, Worst and Average Case Analysis of Sorting Algorithms, Binary search and Hashing algorithms, Lower bound proofs for the above problems, Amortized analysis - Analysis of Knuth-Morris-Pratt algorithm, Amortized weight balanced trees

Module 2: (10T+7P Hours)

Design:, Algorithm paradigms, Divide and conquer - Strassen's algorithm, O(n) median finding algorithm, Dynamic Programming - Matrix Chain Multiplication, Optimal Polygon Triangulation, Optimal Binary Search trees, Floyd-Warshall algorithm, CYK algorithm - greedy - Huffman coding, Kruskal and Prim algorithms, Knapsack - Backtracking - Branch and Bound - Traveling salesman problem.

Module 3: (9T+5P Hours)

Complexity: complexity classes - P, NP, Co-NP, NP-Hard and NP-Complete problems - Cook's theorem- NP-Completeness reductions for Clique - Vertex Cover - Subset Sum - Hamiltonian cycle - TSP - Integer Programming - Approximation algorithms - Vertex cover - TSP - Set Covering and Subset Sum

Module 4: (10T+7P Hours)

Probabilistic algorithms: Pseudo Random Number Generation Methods - Monte Carlo algorithms - Probabilistic counting - Verifying matrix multiplication - Las Vegas algorithms - Randomized Selection and Sorting - Randomized max cut algorithm.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3/e, Prentice Hall India, 2010
2. R. Motwani and P. Raghavan, *Randomized Algorithms,* Cambridge University Press, 2001
3. A. Levitin, *Introduction to the Design & Analysis of Algorithms,* Pearson Education. 2003

CS4051D CODING THEORY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total Hours: 39

Course Outcomes:

Students will be able to :

CO1: Describe standard error control schemes and prove their mathematical properties.
CO2: Analyze a given linear and cyclic codes to find its minimum distance, rate etc.
CO3: Design cyclic codes for given design parameters.

Module 1: (9 Hours)

Linear Codes: Review of linear algebra - Linear codes and syndrome decoding. Generator and parity check matrices, Singleton bound, Plotkin bound. Hamming code, Hadamard code.

Module 2: (10 Hours)

Cyclic codes: BCH codes, BCH–key equation and algorithms. Berlekamp's Iterative decoding Algorithm, decoding with Euclid's algorithm, Reed Solomon codes, decoding Reed Solomon codes.

Module 3: (10 Hours)

List decoding: Johnson's bound, Sudan-Guruswami algorithm. Convolutional codes: trellis decoding, Viterbi's algorithm.

Module 4: (10 Hours)

Codes on Graphs: Concept of girth and minimum distance in graph theoretic codes. Expander Graphs and Codes – linear time decoding. Basic expander based construction of list decodable codes. Sipser Spielman algorithm. Bounding results.

References:

1. W. C. Huffman and V. Pless, *Fundamentals of error correcting codes*, Cambridge University Press, 2003.
2. J. H. Van Lint, *An Introduction to Coding Theory,* 2/e, New York: Springer-Verlag, 1992.
3. R. J. McEliece, *The Theory of Information and Coding,* Addison Wesley, 1997.

CS4052D LOGIC FOR COMPUTER SCIENCE

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total Hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Model simple reactive systems with finite state machine models.
CO2: Describe safety specifications in LTL/CTL.
CO3: Use SPIN/SMV for verifying models against specifications.

Module 1: (10T+7P Hours)

Omega regular languages and Buchi automata - properties, complementation, deterministic Buchi-regular languages. Buchi's complementation theorem,

Module 2: (10T+7P Hours)

Linear time temporal logic - Vardi-Wolper construction, model checking using Buchi automata. Introduction to SPIN. Software verification using SPIN.

Module 3: (10T+7P Hours)

Monadic second order logic, equivalence of MSO and Buchi automata, Safra construction.
S2S. Rabin's tree automata theorem (no proof).

Module 4: (9T+5P Hours)

Computation Tree Logic, Binary decision diagrams, SMV model checker. Software verification using SMV.

References:

1. M. Huth and M. Ryan, *Logic in Computer Science: Modeling and Reasoning about Systems*, 2/e, Cambridge University Press, 2005.
2. C. Baler, *Principles of Model Checking*, MIT Press, 2008.
3. E. M. Clerke. Jr, O. Grumberg, and D. Peled, *Model Checking,* MIT Press, 2001.

CS4053D TOPICS IN LOGIC

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Describe basic theorems and proofs of elementary properties of various systems of logic.
CO2: Prove expressibility and inexpressibility of simple properties in various systems of logic.
CO3: Describe the inclusions between classes of logics and classes of computational complexity.

Module 1: (9 Hours)

Logic: Review of compactness and resolution of propositional logic. First order logic: Syntax and semantics, compactness theorem, completeness theorem, Herbrandization, resolution.

Module 2: (10 Hours)

Computability: Undecidability, recursion theorem, Godel's incompleteness theorem, decidability of Presburger arithmetic.

Module 3: (10 Hours)

Finite model theory: Ehrenfeucht Fraisse games, locality arguments, first order inexpressibility results.

Module 4: (10 Hours)

Complexity: Fagin's theorem, logical characterizations of complexity classes P, PH, and PSPACE.

References:

1. C. H. Papadimitriou, *Computational Complexity,* Addison Wesley, 1994
2. L. Libkin, *Elements of finite model theory*, Springer, 2004.
3. H. Ebbinghaus, *Finite model theory*, Springer, 2005.

CS4054D PARAMETERIZED ALGORITHMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Classify the problems into tractable and intractable problems
CO2 Describe the techniques for the design of fixed-parameter tractable algorithms
CO3: Analyze the running time of fixed-parameter tractable algorithms for some of the classical NP-Hard graph problems

Module 1: (10T+7P Hours)

Review of complexity classes - P, NP, Co-NP, NP-Hard and NP-complete problems - Cook's Theorem. Strategies for Coping with hard algorithmic problems; Exact exponential algorithms and the notion of fixed-parameter tractability. Parameterizations and Parameterized problems- Satisfiability problem, Vertex cover - An illustrative example, Art of problem Parameterization.

Module 2: (10T+6P Hours)

Algorithmic methods: Data reduction and problem kernels. Depth bounded search trees for Maximum satisfiability, Cluster editing, Vertex cover, 3-Hitting set, Dominating set in Planar graphs.

Module 3: (10T+7P Hours)

Dynamic programming – Set cover, Tree structured variants of set cover and Steiner Trees. Randomized methods in Parameterized algorithms – Simple randomized algorithm for Vertex cover, Feedback Vertex Set, Color coding algorithm for Longest path.

Module 4: (9T+6P Hours)

Path and Tree decomposition – Dynamic Programming on graphs of bounded treewidth – Treewidth and Monadic second-order logic, Graph searching, interval and chordal graphs. Computing treewidth – Balanced separators and separations – FPT approximation algorithm for treewidth.

References:

1. M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, June 2015.
2. R. Niedermeier, *Invitation to fixed-parameter algorithms*, Oxford university press, 2006.
3. R. G. Downey and M. R. Fellows, *Fundamentals of Parameterized Complexity*, Springer, 2013.

CS4055D PARAMETERIZED COMPLEXITY THEORY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Classify the parameterized problems into fixed-parameter tractable and intractable.
CO2: Describe the hierarchy of hardness of parameterized intractable problems
CO3: Analyze proofs of standard results in Parameterized complexity theory

Module 1: (10 Hours)

Fixed Parameter Tractability - Introduction, Parameterized Problems and Fixed-Parameter Tractability, Reductions and Parameterized Intractability - Fixed-parameter Tractable reductions - The class para-NP, and The class XP.

Module 2: (10 Hours)

Parameterized complexity theory – Complexity class W[1] - Concrete parameterized reductions – W[1]-hardness proofs – Further reductions and W[2]-hardness. Lower bounds and the complexity class M[1], lower bounds and linear FPT-reductions.

Module 3: (10 Hours)

Kernelization and Linear Programming Techniques, Tree Decomposition of Graphs, Computing Tree Decompositions, Algorithms on structures of Bounded Tree width. Applications of Courcelle's Theorem - Tree width reductions and Graph Minors

Module 4: (9 Hours)

Machine models, limited nondeterminism, and bounded FPT. Selected case studies: Graph modification problems and Capacitated Vertex cover, Constraint Bipartite vertex cover, Graph coloring, Crossing number, Power dominating set.

References:

1. J. Flum and M. Grohe, *Parameterized Complexity Theory,* Springer, 2006.
2. R. Niedermeier, *Invitation to fixed-parameter algorithms*, Oxford university press, 2006.
3. R. G. Downey and M. R. Fellows**,** *Fundamentals of Parameterized Complexity*, Springer, 2013.

CS4056D INTRODUCTION TO HIGH PERFORMANCE COMPUTING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

The students will be able to:
CO1: Describe parallel computing paradigms.
CO2: Design homogenous parallel programs using OpenMP and MPI
CO3: Design heterogeneous parallel programs using CUDA

Module 1: (10T+5P Hours)

Modern Processors : Stored Program Computer Architecture - Pipelining- Superscalar City - Multicore processors - Multithreaded processors- Vector Processors, Parallel Computers: Taxonomy of parallel computing paradigms- SIMD & MIMD classes of computers, Shared memory computers- Cache coherence- UMA - ccNUMA - Distributed-memory computers - Basics of parallelization - Data Parallelism - Function Parallelism- Parallel Scalability- Factors that limit parallel execution; Homogeneous and heterogeneous parallel computing systems, Supercomputer top500 and Green500 listing

Module 2: (10T+7P Hours)

Homogenous Parallel Computing system, Distributed memory parallel programming with MPI : message passing - introduction to MPI – example - messages and point-to-point communication - collective communication – nonblocking point-to-point communication, Efficient MPI programming : MPI performance tools- communication parameters- Synchronization, serialization, contention- Reducing communication overhead- optimal domain decomposition- Aggregating messages – Non Blocking Vs Asynchronous communication- Collective communication- Understanding intra-node point-to-point communication.

Module 3: (10T+7P Hours)

Shared memory parallel programming with OpenMp : introduction to OpenMp - parallel execution - data scoping- OpenMp work sharing for loops- synchronization - reductions - loop scheduling - tasking - case study: OpenMp- parallel jacobi algorithm- advanced OpenMp wavefront parallelization, Efficient OpenMP programming: Profiling OpenMP programs - Performance pitfalls- Case study: Parallel Sparse matrix-vector multiply.

Module 4: (9T+7P Hours)

Heterogeneous parallel computing system, Accelerators - GPUs, Xeon Phi, FPGAs, Parallel programming using GPU and CUDA, Overview of CUDA C; threads, blocks and grids, warps, different GPU memories, Kernel-Based Parallel Programming, Case studies: vector addition, vector-vector multiplication, matrix-matrix multiplication, Xeon phi and its architecture, parallel programming using Xeon phi, Sample programs

References:

1. G. Hager and G. Wellein, *Introduction to High Performance Computing for Scientists and Engineers*, Chapman & Hall / CRC Computational Science series, 2011.
2. D. Kirk and W. Hwu, *Programming Massively Parallel Processors - A Hands-on Approach*, 3/e, Morgan Kaufmann, 2017
3. M. J. Quinn, *Parallel Programming in C with MPI and OpenMP*, McGraw-Hill, 2003.

CS4057D EMBEDDED SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able :

CO1: Describe the basic structure and design of an Embedded System
CO2: Explain the basics of RTOS for Embedded systems.
CO3: Describe the recent trends in embedded systems.

Module 1: (10T+5P Hours)

Definition of Embedded System, Embedded Systems Vs General Computing Systems, History of Embedded Systems, Classification, applications of embedded systems.automotive embedded system, mobile phones, washing machine, artificial pacemaker, trends in embedded software development

Module 2: (10T+7P Hours)

Characteristics and Quality Attributes of Embedded Systems. Core of the Embedded System: General Purpose and Domain Specific Processors, Microcontrollers, DSPs, FPGAs, ASICs, PLDs, Commercial Off-The-Shelf Components (COTS), Memory: ROM, RAM, Memory according to the type of Interface, Memory Shadowing, Memory selection for Embedded Systems, Sensors and Actuators, Reset Circuit, Brownout Protection Circuit, Real Time Clock, Watchdog Timer

Module 3: (10T+7P Hours)

Software engineering practices in the embedded software development process, embedded software development environments, development tools for target processors, real-time embedded software. RTOS, GPOS Vs RTOS, Tasks, Process and Threads, Multiprocessing and Multitasking, Task Scheduling, Shared Memory, Message Passing, Remote Procedure Call and Sockets, Device Drivers, How to Choose an RTOS

Module 4: (9T+7P Hours)

Embedded Software Development Tools, Host and target machines – Linkers / Locators for Embedded Software – Debugging techniques – Instruction set simulators Laboratory tools – Practical example – Source code. Recent trends in Embedded Systems.

References:

1. S. Heath, *Embedded System Design*, 2/e, Elsevier, 2004.
2. D. E. Simon, *An Embedded Software Primer*, Pearson Education, 2010
3. DreamTech Software Team, *Programming of Embedded Systems*, Wiley DreamTech, 2002.

CS4058D COMPUTATIONAL GEOMETRY

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Design algorithms for geometric problems and analyze these algorithms.
CO2: Analyze the correctness of the geometric algorithms.
CO3: Formulate a geometric solution for an application and design an algorithm with efficient data structure.
CO4: Write programs using CGAL.

Module 1: (10T+10P Hours)

Introduction to Computational Geometry, its applications. Preliminaries of asymptotic analysis and the notations to represent asymptotic complexity, Example of a geometric problem and its complexity analysis. Art Gallery problem and its associated theorems, Triangulation of a polygon and its theory, Area of a polygon. Polygon partitioning, Monotone partitioning, Trapezoidalization, Plane sweep, Partitioning to monotone mountains, Linear time triangulation. Introduction to Computational Geometric Algorithms Library (CGAL) and openGL and coding of simple programs with visualization using QT.

Module 2: (10T+6P Hours)

Convex hull in two dimensions, Algorithms for convex hull with their complexity analysis: Extreme points, Extreme edges, Gift wrapping, Quickhull, Graham's algorithm, Incremental algorithm, Divide and conquer. Applications of convex hull.hard Implement Convex Hull algorithms and one application using CGAL & visualization using QT**.**

Module 3: (10T+5P Hours)

Voronoi diagram: Basic concepts, Definitions, Properties, Algorithm for construction of Voronoi diagram with its complexity analysis. Delaunay triangulation : Preliminaries and properties. Medial axis transform and its properties. Applications of Voronoi Diagram / Delaunay triangulation / Medial axis transform: Facility location, Reconstruction problem and its algorithms Implement one application of Voronoi diagram/ Delaunay triangulation using CGAL & visualization using QT.

Module 4: (9T+5P Hours)

Arrangements, Incremental algorithm, Voronoi diagram, Delaunay triangulation and convex hull in three dimensions and their applications. Implement one application of 3D Voronoi diagram/ 3D Delaunay triangulation using CGAL & visualization using QGL Viewer.

References:

1. J. O'Rourke, *Computational Geometry in C*, 2/e, Cambridge University Press, 1998.
2. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, 2/e (revised), Springer-Verlag,2000.
3. F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

CS4059D TOPICS IN COMPUTATIONAL GEOMETRY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Design and develop algorithms and data structures for geometric problems.
CO2: Formulate a geometric solution using randomization as a tool.
CO3: Use CGAL to implement advanced geometric problems.

Module 1: (10 Hours)

Binary space partitions : Definition, basic concepts, construction using randomized algorithm, theorems, . CGAL implementation of Painter's algorithm

Module 2: (10 Hours)

Robot motion planning: Workspace and configuration space, Point robot, Minkowski sums, Translational motion planning, Quadtrees: Uniform and non-uniform meshes, Quadtrees for point sets, Quadtrees to meshes.

Module 3: (10 Hours)

Visibility Graphs: Shortest paths for a point robot, Visibility graphs, Shortest paths for a translating polygonal robot.

Module 4: (9 Hours)

Interval Trees, Priority Search Trees, Segment Trees, Partition trees, Multi-level partition trees. Simplex Range Searching.

References:

1. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, 2/e(revised), Springer-Verlag, 2000.
2. S. L. Devadoss and J. O'Rourke, *Discrete and Computational Geometry*, Princeton University Press, 2011.
3. K. Mulmuley, *Computational Geometry: An Introduction through Randomized Algorithms*, Prentice-Hall, 1994.

CS4060D INTRODUCTION TO DATA SCIENCE

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Describe what Data Science is and acquire the skill sets needed to be a Data Scientist.
CO2: Define problem,collect and analyze data to build model and implement solution.
CO3: Use R to carry out basic data analysis.
CO4: Explain the significance of Exploratory Data Analysis (EDA) in Data Science.
CO5: Apply basic machine learning algorithms for predictive data modeling and analysis.

Module 1: (10T+6P Hours)

Introduction to Data Science, Review of Basic Concepts in Statistics, Probability and Linear Algebra, Data Modelling: Supervised and Unsupervised Learning - Linear Regression, k-Nearest Neighbours (k-NN), k-Means Clustering.

Module 2: (10T+8P Hours)

Exploratory Data Analysis, The Data Science process, Introduction to R programming, Extracting Meaning From Data- Feature Generation, Feature Selection algorithms, Advanced supervised learning approaches - Linear Support Vector Machines, Decision Trees, Random Forests.

Module 3: (10T+6P Hours)

Recommendation Systems - Algorithmic Ingredients, Dimensionality Reduction, Singular Value Decomposition, Principal Component Analysis, Data Visualization, Ideas and Tools for Data Visualization.

Module 3: (9T+6P Hours)

Working with Big Data – Social Networks as Graphs, Clustering of graphs, Direct discovery of communities in graphs, Partitioning of graphs, Emerging Trends in Data Science.

References:

1. C. O'Neil and R. Schutt, *Doing Data Science: Straight Talk from the Frontline* , O'Reilly, 2014.
2. J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, V 2.1, Cambridge University Press, 2014.
3. M. J. Zaki and W. Meira Jr, *Data Mining and Analysis: Fundamental Concepts and Algorithms*, Cambridge University Press, 2014.
4. A. Blum, J. E. Hopcroft and R. Kannan, *Foundations of Data Science*, e-book, 2013
5. J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3/e, Elsevier, 2012.

CS4061D TOPICS IN DATA ANALYTICS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Identify and avoid common pitfalls found in big data analytics.
CO2: Identify the best analytic approach for a given problem.
CO3: Develop solutions to deal with the challenges in Massive Data Analytics
CO4: Apply Machine learning techniques to enable predictive analytics.

Module 1: (10T+7P Hours)

Introduction to Data Analytics- Revision of Mathematical Basics - Probability, Statistics, Linear Algebra and Calculus - Data Preparation and Cleansing - Descriptive Statistics- Probability Distributions- Inferential Statistics through hypothesis test- Regression and ANOVA

Module 2: (10T+7P Hours)

Machine Learning: Introduction and Concepts- Differentiating algorithmic and model based frameworks - Regression: Ordinary Least Squares, Ridge Regression, Lasso Regression, K Nearest Neighbours Regression & Classification- Supervised Learning with Regression and Classification techniques -Bias-Variance Dichotomy, Model Validation Approaches, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Regression and Classification Trees, Support Vector Machines

Module 3: (10T+6P Hours)

Ensemble Methods: Random Forest , Neural Networks, Deep learning, Unsupervised Learning and Challenges for Big Data Analytics – Clustering Associative Rule Mining, Challenges for big data analytics
Prescriptive analytics - Creating data for analytics through designed experiments, Creating data for analytics through Active learning, Creating data for analytics through reinforcement learning

Module 4: (9T+6P Hours)

Massive Data Analytics - Social Network Analytics - Security and Privacy Issues

References:

1. T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2/e, Springer, 2009.
2. D. C. Montgomery and G. C. Runger, *Applied statistics and probability for engineers,* John Wiley & Sons, 2010.
3. K. P. Murphy, *Machine Learning, a probabilistic perspective*, MIT Press Cambridge, Massachusetts, 2012
4. A. Smola and S. V. N. Vishwanathan, *Introduction to Machine Learning*, Cambridge University Press, 2008.
5. J. E. Hopcroft and R. Kannan, *Foundations of Data Science*, e-book, 2013

CS4062D INTRODUCTION TO INFORMATION SECURITY

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Analyze the basic concepts of information security to be kept in mind while designing Computer based systems.
CO2: Understand cryptographic principles and techniques for securing information in the digital world.
CO3: Apply security management controls based on ISO27001.

Module 1: (10 Hours)

Introduction to Information Security - CIA triad - Cyber Attacks - Defence Strategies and Techniques - Secure Software Design Principles, Access Control and Authentication, Biometric Authentication - Malwares Viruses and Worms - Firewalls and Intrusion Detection System

Module 2: (15 Hours)

Cryptography- Basics of Cryptography - Elementary Substitution and Transposition Ciphers - Hill Cipher, Vigenere Cipher - Symmetric Cipher - DES - Kerckhoff's principle - Asymmetric Cipher - RSA - Cryptographic Hashing - SHA1 - Digital Signature

Module 3: (14 Hours)

Introduction to security of information storage, processing, and transmission.
Information Security Management - The ISO Standards relating to Information Security - Other Information Security Management Frameworks - Security Policies - Security Controls - The Risk Management Process - Regulations and legal frameworks

References:

1. B. Menezes, *Network security and Cryptography*, Cengage Learning India, 2010.
2. B. A. Forouzan and D. Mukhopadhyay, *Cryptography and Network Security,* 2/e, Tata McGraw Hill, 2010.

CS4063D TOPICS IN CRYPTOGRAPHY

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Demonstrate knowledge of fundamental cryptographic concepts.
CO2: Analyze and construct private-key encryption schemes.
CO3: Analyze and construct public-key encryption schemes.

Module 1: (9T Hours)

Review of classical cryptography - Historical ciphers and their cryptanalysis. Principles of Modern Cryptography - Formal definitions, precise assumptions, proof of security. Provable security.
Perfectly secret encryption - The one time pad. Limitations of perfect secrecy. Shannon's Theorem.
Practical construction of stream ciphers.

Module 2: (16 T+16 P Hours)

Private-Key Encryption - Computational Security and Computationally Secure Encryption - Constructing Secure Encryption Schemes - Linear and Differential Cryptanalysis.
Message Authentication Codes - Constructing Secure MAC - Authenticated Encryption.
Hash functions - Generic attacks and proof of security.
Practical block ciphers and hash functions.

Module 3: (14 T+10 P Hours)

Public-Key Encryption - Review of number theory concepts - Primality Testing - Solovay Strassen, Miller Rabin tests. Algorithms for factoring. Algorithms for computing discrete logarithms.
Review of RSA - Attacks on RSA. Security of Public Key Encryption systems against chosen plaintext and chosen ciphertext attacks. Introduction to Elliptic Curve Cryptosystems, Homomorphic Encryption, Threshold Encryption.

References:

1. J. Katz and Y. Lindell, *Introduction to Modern Cryptography, 2/e, CRC Press, 2014*
2. H. Delfs and H. Knebl, *Introduction to Cryptography Principles and Applications*, Springer, 2002
3. D. R. Stinson, *Cryptography Theory and Practice, 3/e*, CRC Press, 2006

CS4064D PROGRAM ANALYSIS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain the different approaches to program analysis.
CO2: Apply program analysis to derive program properties.
CO3: Analyze the properties of data-flow analysis algorithms.
CO4: Formulate a program analysis to derive program properties.

Module 1: (9T+7P Hours)

Approaches to Program Analysis - Data-flow Analysis, Constraint based analysis, Abstract Interpretation, Type and effect systems

Module 2: (10T+6P Hours)

Data-flow Analysis: Program Representations - Control Flow Graph, Analysis for computing : available expressions, reaching definitions, live variables. Applications of analysis in code improving transformations

Module 3: (10T+7P Hours)

Data-flow Analysis Framework: Lattice theoretic modeling - Monotone frameworks, Distributive frameworks. Constant Propagation framework, Iterative algorithm for monotone frameworks - Maximal Fixed Point solution, Meet Over All Paths solution.

Module 4: (10T+6P Hours)

Type and effect Systems - control flow analysis, Introduction to inter procedural analysis and shape analysis.

References:

1. F. Nielsen, H. R. Nielson, and C. Hankin, *Principles of Program Analysis*, Springer 1999.
2. A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques, and Tools*, Pearson Education, 2007
3. S. Muchnick, *Advanced Compiler Design and Implementation,* Morgan Kaufmann, 1997

CS4065D FORMAL SEMANTICS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain the fundamental idea behind different semantic specification methods.
CO2: Formulate the operational semantics specification of language constructs.
CO3: Formulate the denotational semantic specification of language constructs.
CO4: Prove program correctness by applying axiomatic methods.

Module 1: (9T+7P Hours)

Preliminaries, Mathematical foundations. Methods for Semantics Specification - Introduction to Operational, Denotational, and Axiomatic semantics.

Module 2: (10T+6P Hours)

Operational Semantics - Natural semantics, Structural operational semantics. Semantics of a simple imperative language - operational semantics of expressions, assignments, conditional and looping constructs.

Module 3: (10T+7P Hours)

Denotational Semantics: Semantic functions, Semantics of expressions, assignments, conditional and looping constructs. Fixed points, existence of fixed points.

Module 4: (10T+6P Hours)

Axiomatic Program Verification - Partial Correctness assertions, Inference systems. Extensions of the axiomatic system – Total correctness assertions.

References:

1. F. Nielson and H. R. Nielson, *Semantics with Applications: A Formal Introduction*, John Wiley & Sons Inc., 1992.
2. G. Winskel, *The Formal Semantics of Programming Languages - An Introduction,* MIT Press, Cambridge, MA, 1994.
3. D. Schmidt, *Denotational Semantics: A Methodology for Language Development*., Kansas State University, 2011

CS4066D ALGORITHMIC DECISION MAKING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Design automated decision agents
CO2: Evaluate different machine learning approaches and choose the right one applicable in a context
CO3: Examine deep learning approaches for agent design

Module 1: (13T+10P Hours)

Probability Theory, Bayesian Inference, Bayesian Networks, Modeling and Reasoning with Bayesian Networks, Applications of Bayesian Networks.
Decision Trees (basics), Influence Diagrams, Modeling and solving Influence Diagrams.
Bayesian Networks and Influence diagrams in dynamic decision making.

Module 2: (13T+10P Hours)

Parameter Learning - Learning a Single Parameter, Beta Density Function, Learning Parameters in a Bayesian Network.
Structure Learning- Schema and Procedure for learning Structure, Model Averaging, Probabilistic Model selection, Hidden Variable DAG models.

Module 3: (13T+6P Hours)

Reinforcement Learning - Introduction, Markov Property, Markov Decision Processes,Value Functions.
Dynamic Programming in Policy Evaluation, Policy Improvement, Policy Iteration and Value Iteration Methods

References:

1. R. E. Neapolitan, *Learning Bayesian Networks,* Pearson, 2003.
2. A. Darwiche, *Modeling and Reasoning with Bayesian Networks,* Cambridge University Press, 2009.
3. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.
4. M. J. Kochenderfer, *Decision Making Under Uncertainty, Theory and Application*, MIT Press, 2015.

CS4067D FOUNDATIONS OF PROGRAMMING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Explain concepts and constructs of programming languages
CO2: Explain and apply procedural abstraction, data abstraction and modular design.
CO3: Prepare formal problem specification.
CO4: Design solutions and establish correctness of the solutions.

Module 1: (10T+7P Hours)

Programming methodology. Specification, Design, Coding. Separation of concerns such as correctness, efficiency, and maintainability. Fundamental concepts and constructs in programming languages.

Module 2: (10T+7P Hours)

Procedural Abstraction: Expressions - Naming and Environment - Combinators - Evaluation - Procedures - Substitution model - Conditional expression and predicates. Linear Recursion and Iteration - Tree recursion. Abstractions with Higher Order Procedures - Procedures as arguments - Constructing procedures – examples.

Module 3: (10T+6P Hours)

Data Abstraction: Hierarchical Data and Closure property - Symbolic Data - Data Directed Programming - Generic Operators - Combining data of different types

Module 4: (9T+6P Hours)

Modular design: Modularity, Objects, and State: Local state - assignment, environment model for evaluation - frames, Modeling with mutable data. Encapsulation, inheritance, and polymorphism.

References:

1. H. Abelson and G. J. Sussman, *Structure and Interpretation of Computer Programs,* 2/e, Universities Press, 2005.
2. Companion Site to the Textbook. Available at http://mitpress.mit.edu/sicp/ Accessed on December 1, 2010.
3. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering,* 1/e, Tata McGraw Hill, 2004.

CS4068D DNA COMPUTING MODELS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Learn and analyse the non-traditional and non-silicon DNA (Deoxyribonucleic Acid) computing models.
CO2: Adopt an efficient DNA computing model to achieve parallelism.
CO3:Solve massive problems using adopted and an efficient DNA Computing models.

Module 1: (10 Hours)

Introduction, Theoretical Computer Science: Graphs, Finite State Automata, Computability, Formal Grammars, Combinatorial Logic, Computational Complexity, Molecular Biology: DNA, Gene, Gene expression.

Module 2: (10 Hours)

Word Design for DNA Computing: Constraints, DNA Languages, DNA Code Constructions and Bounds, In Vitro Random Selection.

Module 3: (10 Hours)

Non-Autonomous DNA Models: Seminal Work, Filtering Models, Sticker Systems, Splicing Systems

Module 4: (9 Hours)

Autonomous DNA Models: Algorithmic Self-Assembly, Finite State Automaton Models, DNA Hairpin Model, Computational Models

References:

1. Z. Ignatova, I. Martínez-Pérez, and K. H. Zimmermann, *DNA Computing Models,* Springer, 2008.
2. G. Paun, G. Rozenberg, and A. Salomaa, *DNA Computing: New Computing Paradigms,* Springer Science & Business Media, 2005.

CS4069D HASHING TECHNIQUES FOR BIG DATA

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Outline the various challenges and opportunities in big data indexing and hashing
CO2: Survey and analyze the various supervised and unsupervised hashing techniques.
CO3: Solve real life big data hashing problems using efficient hashing techniques.

Module 1: (10 Hours)

Low dimensional index structures; Hashing - Static Hashing, Dynamic Hashing, Locality Sensitive Hashing, Multidimensional Hashing, Space Filling Curves.
Memory based index structures - Binary Search Tree, Quad Tree, K-D Tree, Range Tree, Voronoi Diagram, Tries, Suffix Trees.
Disk based index structures - B Tree, B+ Tree, K-D B Tree, General Framework, R Tree, R* Tree, R+ Tree, Hilbert R Tree, SS Tree, SR Tree, P Tree, Bulk Loading; Distances:

Module 2: (10 Hours)

Distance functions - Metric Spaces, L_p Norm, Quadratic Form Distance, Cosine Similarity, Statistical Distance measures, Distance between set of objects, Earth Mover's Distance, Edit Distance.
Distance based Structures - Triangular Inequality, VP Tree, GH tree, GNAT, M Tree, SA Tree, AESA, Linear AESA, AESA for vector spaces.

Module 3: (10 Hours)

High-Dimensional Spaces- Analysis of Search for High-Dimensional Data, Expected Nearest Neighbour Distance, Expected Number of Page Accesses, Curse of Dimensionality. High-Dimensionality Structures - X Tree, Pyramid Technique, MinMAx, VA File, A Tree, IQ Tree
Dimensionality Reduction Techniques: Properties Useful for Similarity search, Quality Measures, Embedding, Singular Value Decomposition, Principal Component Analysis, Multi-Dimensional Scaling, IsoMap, FastMap, Embedding Methods, Bounds of Distortion.

Module 4: (9 Hours)

Data Representation Techniques - Discrete Fourier Transform, Discrete Cosine Transform, Discrete Wavelet Transform, V-Optimal Histogram
Data-independent Hashing, Data-Dependent Hashing - Unsupervised Hashing, Supervised Hashing, Ranking-Based Hashing, Multimodal Hashing, Deep Hashing, Online Hashing, Quantization for Hashing, Distributed Hashing.

References:

1. A. Bhattacharya, *Fundamentals of Database Indexing and Searching*, 1/e, Chapman and Hall/CRC, 2016.
2. H. Samet, *The Design and Analysis of Spatial Data Structures*, 1/e, Addison-Wesley, 1989
3. T. Teofili, *Deep Learning for Search*, 1/e, Manning Publications, 2018.

CS4070D TOPICS IN COMPUTER NETWORKS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Recall the classic protocol in TCP/IP stack.
CO2: Analyze the challenges in the modern computer networking world.
CO3: Propose solutions to the research problems in computer networking.

Module 1: (10 Hours)

Networking basics, Gigabit Ethernet, Spanning Tree Algorithm and Redundancy, VLAN, Software-Defined Networking, Virtual Private Networks, WiFi, WiMax, LTE, 5G, ATM, Optical Networking, Backbone Network Design

Module 2: (10 Hours)

IP version 6, Implications of IPv6, MPLS, Large Scale routing- BGP. Routing in Mobile adhoc networks, SCTP, DCCP, Newer TCP Implementations - Highspeed TCP; TCP Vegas; FAST TCP; TCP Westwood; TCP Illinois; Compound TCP; TCP Veno; TCP Hybla; DCTCP; H-TCP; TCP CUBIC; TCP BBR

Module 3: (10 Hours)

Queuing and scheduling, Queuing and real time traffic, Traffic management, Queuing Disciplines - Priority Queuing; Fair Queuing; Hierarchical Queuing; Hierarchical Weighted Fair Queuing; Token Bucket Filters

Module 4: (9 Hours)

Traffic engineering, Quality of Service - Net Neutrality; Real-time Traffic; Integrated Services / RSVP; Global IP Multicast; RSVP; Differentiated Services; RED with In and Out; NSIS; Real-time Transport Protocol (RTP); Multi-Protocol Label Switching (MPLS)
SNMP and multimedia over Internet, Enterprise Network Security

References:

1. N. F. Mir, *Computer and Communication Networks*, 1/e, Prentice Hall, 2006.
2. W. Stallings, *Data and Computer Communications*, 10/e, Pearson Education India, 2013..
3. P. Loshin, *TCP/IP Clearly Explained*, 4/e, Morgan Kaufmann, 2002.

CS4071D NETWORK ANALYSIS IN BIOINFORMATICS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Describe network based approach to system biology
CO2: Analyse biological interactions through network theory
CO3: Describe various approaches in network biology
CO4: Apply various tools and packages for biological network analysis

Module 1: (10 Hours)

Introduction to molecular biology- DNA,RNA,Protein, Gene, Cell Metabolism, Systems biology- Biological networks- Network representation- Network types: PPI, Genetic interaction networks, Metabolic networks, Gene/transcriptional regulatory networks, Cell signaling networks, Pathways, Phylogenetic trees

Module 2: (10 Hours)

Introduction to network science- Network characteristics- Shortest paths- Network flows- Centrality measures- Network communities and modules- Network motifs- Network clustering- Spectral clustering-Subgraphs-Network Diffusion- Random walk models

Module 3: (10 Hours)

Heterogeneous Information Networks- Definition- Properties- Meta paths- HIN algorithms- Clustering & classification in HIN- Similarity search- Relationship mining
Multilayer networks- Data collection and preprocessing- Visualization- Community detection- Information diffusion- Edge pattern

Module 4: (9 Hours)

Biological network analysis in R- Manipulation and visualization of network data- igraph package- Implementation of centrality measures- Basics of graph modeling in R- Bioconductor- BioNet package- Other network analysis tools (eg. Cytoscape)

References:

1. B. Alberts et al., *Molecular Biology of the Cell,* Garland Science Publishers, 2014
2. B. Junker and F. Schreiber, *Analysis of Biological Networks,* Wiley Publishers, 2007
3. A. L. Barabasi, *Network Science*, Cambridge University Press, 2016
4. Y. Sun and J. Han, *Mining Heterogeneous Information Networks- Principles & Methodologies,* Morgan & Claypool Publishers, 2012
5. M. E. Dickison et al, *Multilayer Social Networks*, Cambridge University Press, 2016
6. E. D. Kolaczyk and G. Csardi, *Statistical Analysis of Network Data with R*, Springer-Verlag, New York, 2014

CS4089D TERM PAPER

Prerequisites: NIL

L	T	P	C
0	0	8	3

Course Outcomes:

At the end of the course the student will:

CO1: Prepare a literature survey on a specific area of scientific/engineering work relating to computing.
CO2: Prepare a summary of major work in the area.
CO3: Prepare a report of outstanding open problems in the area of work.
CO4: Compile the data into a report.

The student is required to conduct a detailed literature survey in an area of research in the field of computing and compile a report of her/his study in the area. The student may include any observations from her/his own experimentation in the area in the report.

References:

1. G. J. Alred, C. T. Brusaw, and W. E. Oliu, *The Handbook of Technical Writing*, 11/e, Bedford/St. Martins, 2015.
2. P. A. Laplante, *Technical Writing: A practical guide for engineers and scientists,* CRC Press, 2011.

CS3007D OBJECT ORIENTED SYSTEMS

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T+26P

Course Outcomes:

Students will be able to :

CO1: Design systems using object oriented principles
CO2: Model software solutions to real problems using object oriented principles keeping quality in mind.

Module 1: (10T+7P Hours)

An Overview of Object Oriented Systems Development- Object Basics- Object Oriented System Development Life Cycle - Object Oriented Methodologies- Unified Modelling Language

Module 2: (10T+7P Hours)

Use Case based Object Oriented Analysis - Classification - Identification of Object Relationships and Methods. Object Oriented Design - Principles - Classes - Patterns - Designing Data Store , User Interface

Module 3: (10T+6P Hours)

Object Oriented Architecture - OOAD Testing and Quality Strategies - Object Oriented Metrics

Module 4: (9T+6P Hours)

OOAD Implementation Strategies - Pragmatics - Management Planning , Release Management, Handling Changes, Reuse

References:

1. A. Bahrami, *Object Oriented Systems Development*, 1/e, McGraw-Hill, 2000
2. G. Booch, R. A. Maksimchuk, M. W. Engle, B. J. Young, and J. Conallen, *Object-Oriented Analysis and Design with Applications*, 3/e, The Addison Wesley Object Technology Series.
3. B. D. McLaughlin, G. Pollice, and D. West, *Head First Object-Oriented Analysis and Design*, 1/e, O'Reilly .

CS4072D ADVANCED PROGRAMMING AND DATA STRUCTURES FOR ENGINEERS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Describe basic data structures and their applications.
CO2: Describe the complexity of simple programs, sorting algorithms and elementary data structures.
CO3: Describe simple algorithms that uses standard algorithm design paradigms.

The course is designed for students from non CS departments to gain a primary understanding of data structures and algorithm design.

Module 1: (13 Hours)

Introduction to time complexity analysis, asymptotic notation and solution to simple recurrences, analysis of binary search, quicksort, merge sort and heap sort algorithms.

Module 2: (13 Hours)

Pointers and linked lists - linked list, stack, queue, binary search tree, worst and average case analysis, preorder, inorder and postorder traversals, evaluation of arithmetic expressions, complexity analysis.

Module 3: (13 Hours)

Algorithm design paradigms, divide and conquer - dynamic programming and greedy algorithms, Introduction to NP completeness.

References:

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3/e, MIT Press, 2009.
3. E. Horowitz, S. Sahni, and D. Mehta, *Fundamentals of Data Structures in C++*, 2/e, Universities Press, 2008.

CS4073D COMPUTING SYSTEMS FOR ENGINEERS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to :

CO1: Design simple sequential and combinational circuits.
CO2: Analyze the processor design of the Hack processor.
CO3: Design simple programs for the Hack processor.
CO4: Describe the functions of system programs like OS and compilers.

The course is designed for students from non CS departments to gain a primary understanding of computer hardware design and systems software.

Module 1: (13 Hours)

Arithmetic Logic Unit: Gates and flip flop, number systems - two's complement addition, comparator circuits, design of the Hack processor's ALU.

Memory design: multiplexers and demultiplexers, D-flip flops, design of the Hack processor's memory

Module 2: (13 Hours)

Control unit - system architecture, Hack keyboard and memory interface, design of the Hack
Hack machine instruction set. Hack assembly language, simple programming exercises on the
Hack platform.

Module 3: (13 Hours)

Introduction to systems software - compilers, and operating systems. Simple examples of high level language translation. OS functions and interfaces.

References:

1. N. Nisan and S. Schocken, *The elements of computing systems: Building a modern computer from first principles*, MIT Press, 2008.
2. C. Petzold, *Code: The hidden language of computer hardware and software*, Microsoft Press, 2000.

CS3091D COMPILER LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Translate a given specification for a compiler to a functional implementation.
CO2: Lex/Flex and Yacc/Bison for lexical analysis and syntax analysis.
CO3: Modularize and large programs to synthesise large systems.

Theory (13 Hours)

Introduction to the use of the Flex and Bison. Syntax directed translation scheme in bison. Code generation from abstract syntax tree, run time activation records, dynamic memory allocation.

Practical (39 Hours)

1. Generation of lexical analyzer using Flex.
2. Parsing using Bison.
3. Symbol table implementation.
4. Type checking using syntax directed translation scheme of Bison.
5. Intermediate code generation - abstract syntax tree representation of programs.
6. Run time stack implementation for subroutine invocations.
7. Register allocation and code generation.

References:

1. W. Appel, *Modern Compiler Implementation in C*, Cambridge University Press, 1998.
2. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers- Principles, Techniques & Tools*, 2/e, Pearson Education, 2007.
3. D. Grune, K. V. Reeuwijk, H. E. Bal, C. J. H. Jacobs, and K. Langendoen, *Modern Compiler Design*, 2/e, Pearson Education, 2007.

CS3092D OPERATING SYSTEMS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Implement a primitive file system for a given design.
CO2: Implement multiprogramming support for a given specification.
CO3: Design and Implement shell and system utilities for a user level requirement specification.
CO4: Implement a given system call interface specification.

Theory (13 Hours)

System programming fundamentals and system call interface. System calls for file system, process management and process synchronization.

Practical (39 Hours)

1. Loading executable programs into memory, implement file system calls Read(), Write(), Open (), Seek() and Close().
2. Multiprogramming-Memory management- Implementation of Fork(), Wait(), Exec() and Exit() System calls.
3. IPC system call implementation - Signal-Wait, Semaphores.
4. Demand paging and swapping -implementation.
5. Shell and System utilities - implementation.
6. Implementation Low level routines - disk and terminal driver, timer interrupt handler and scheduler.

References:

1. G. J. Nutt, *Operating Systems*, 3/e, Pearson Education, 2004.
2. D. P. Bovet and M. Cesati , *Understanding the Linux Kernel,* 3/e, O'Reilly Media, 2005
3. C. Crowley, *Operating Systems: A design oriented approach*, 1/e, Mc. Graw Hill, 2006.

CS3093D NETWORKS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Design and implement simple client server programs using STREAM and.DGRAM sockets.
CO2: Design and implement application in which server can accept multiple connection and do concurrent servicing of clients using either, fork, select/poll or threads.
CO3: Simulate network topology and analyze its performance using simulation/emulation tools like NS3, Mininet.

Theory (13 Hours)

Introduction, Overview of Unix Programming Environment, Unix Programing Tools, Introduction to Computer Networking and TCP/IP, Introduction to Socket Programming, TCP Sockets and Concurrent Servers, Threads, I/O Multiplexing and Socket Options, UDP Sockets and Name and Address Conversions, Daemon Processes and Inetd Superserver, Advanced I/O and Timeouts, Non-blocking Sockets, Unix Domain Sockets, Broadcasting, Multicasting, Advanced UDP Sockets, Ioctl Operations.
Introduction to open source firewall packages. Introduction to network emulators and simulators.

Practical (39 Hours)

1. Implementation of basic Client Server program using TCP Socket (Eg. Day time server and clent).
2. Implementation of basic Client Server program using UDP Socket.
3. Implementing a program with TCP Server and UDP Client.
4. Implementation of TCP Client Server program with concurrent connection from clients.
5. Implementing fully concurrent application with a TCP server acting as a directory server and client programs allowing concurrent connection and message transfer (Eg. Chat system).
6. Fully decentralized application like a Peer to Peer system. This program is to implement without a designated Server as in the case of experiment 5.
7. Experiments with open source firewall/proxy packages like iptables, ufw, squid etc.
8. Experiments with Emulator like Mininet / Netkit / Emulab
9. Experiments with Simulator like NS2,/NCTUns .

References:

1. W. R. Stevens, *Unix Network Programming – Networking APIs: Sockets and XTI,* 2/e, Volume 1, Pearson Education, 2004.
2. W. W. Gay, *Linux Socket Programming by Example,* 1/e, Que Press, 2000.

CS3094D SYSTEMS PROGRAMMING LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Write simple Unix shell programs.
CO2: Implement dynamic memory allocation routines
CO3: Solve problems using synchronization primitives of the OS.

Theory (13 Hours)

Basics of shell programming. Unix system call interface, semaphores, shared memory, Posix threads, thread and process synchronization.

Practical (39 Hours)

1. Unix/Linux shell programming exercises.
2. Inter-process communication and process synchronization, semaphores and shared memory - solution to dining philosophers problem, reader-writer problem.
3. Signal handling, wait and kill operations.
4. Threads for multiprogramming - posix threads, reader- writer problem using threads.
5. Software synchronization - Peterson's algorithm.
6. Dynamic memory allocation - variable cell allocation, Buddy system allocation.

References:

1. B. W. Kernighan and R. Pike, *The Unix Programming Environment*, Prentice Hall, 1983.
2. W. R. Stevens, *Advanced Programming in the Unix Environment*, 3/e, Addison Wesley, 2013.

CS3095D DATABASE MANAGEMENT SYSTEMS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Design of Entity-Relationship model for real world problems and development using workbenches
CO2: Design and normalization of database based on ER model and functional dependencies
CO3: Create and query a database
CO4: Develop web based database applications for real world problems
CO5: Implement Relational algebra queries

Theory (13 Hours)

Study of Postgresql, PL/SQL programming and JDBC. Concepts of views, scripts, triggers and transactions, SQL DBA, PHP, Eclipse. Servlets.

Practical (39 Hours)

Laboratory exercises for the following:

1. Defining schemas for applications
2. Creation of databases
3. Writing SQL and PL/SQL queries, to retrieve information from the databases
4. Use of host languages
5. Interface with embedded SQL
6. Use of forms & report writing packages available with the chosen RDBMS product preferably Postgres
7. SQL Programming exercises on using scripting languages like PHP
8. Giving web interfaces for back end database applications.
9. Programming in Java for connecting Postgresql databases using JDBC.
10. Creating web page interfaces for database applications using servlets.

References:

1. A. Silberschatz, H. Korth, and S. Sudarshan, *Database System Concepts*, 5/e, McGraw Hill, 2005.
2. R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 5/e, Addison Wesley, 2007.
3. J. D. Ullman and J. Widom, *A First Course in Database Systems*, 3/e, India: Pearson, 2016.

CS4090D COMPUTER SECURITY LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Simulate well-known security attacks
CO2: Build cryptographic solutions using existing libraries in programming languages.
CO3: Assess the security of networks using state of the art tools.

Theory (13 Hours)

Review of Computer Networks - Associated vulnerabilities and prevention mechanisms.
Review of Cryptographic Algorithms - Secret and Public Key Cryptography, Key Management, Hashing, Signature
Review of Software Vulnerabilities - Buffer Overflow, Format String, Race Conditions
Review of Internet Vulnerabilities - DoS, SQL injection, XSS

Practical (39 Hours)

Simulating attacks based on vulnerabilities in Network, Web and Software (includes attacks like buffer overflow, format string, SQL injection, XSS)
Implementation of cryptographic primitives and solutions using cryptographic libraries in Python / Java
Deploying and using tools like Wireshark, Webgoat, Nmap, Metasploit, Ettercap

References:

1. Michael Gregg, *Build Your Own Security Lab,* Wiley India, 2008
2. B. Menezes, *Network security and Cryptography*, Cengage Learning India, 2010.

CS4091D DATA ANALYTICS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Implement various data analysis techniques
CO2: Apply dimensionality reduction techniques.
CO3: Implement data retrieval techniques using various hashing methods.

Theory (13 Hours)
Supervised and Unsupervised learning methods, Dimensionality reduction methods, Data independent and data dependent hashing methods, Unsupervised Hashing, Supervised Hashing, Ranking-Based Hashing, Multi-Modal Hashing, Deep Hashing, Online Hashing, Quantization for Hashing, Distributed Hashing, Hadoop and MapReduce.

Practical (39 Hours)
1. Generative model linear classification
2. Discriminative model linear classification
3. Clustering
4. Dimensionality reduction
5. Approximate nearest neighbour search
6. Supervised hashing
7. Unsupervised hashing
8. Deep hashing
9. Distributed Hashing

References:

1. W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*, 2/e, Shroff/ O'Reilly Reprints, 2017.
2. J. VanderPlas, *Python Data Science Handbook: Essential Tools for Working with Data*, 1/e, O'Reilly Media, 2016
3. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, 1/e, MIT Press, 2017.

CS4092D MACHINE LEARNING LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Apply various classification and parameter estimation techniques.
CO2: Apply Dimensionality reduction to the feature set.
CO3: Build classifiers using ANN and CNN.

Theory (13 Hours)

Fundamentals of Machine Learning, Bayes classifier, Parameter Estimation, Discriminant Functions, SVM, ANN.

Practical (39 Hours)

1. Bayes Classifier.
2. Parameter Estimation(MLE and Non-parametric Estimation).
3. Linear Data Analysis Methods(LDA,PCA,Regression).
4. Support Vector Machines.
5. KNN Classifier.
6. Decision Trees and Random Forest.
7. Artificial Neural Network and Deep Learning.

References:

1. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification* , John-Wiley, 2004.
2. C. M. Bishop, *Pattern Recognition and Machine Learning,* Springer, 2006.
3. Yuxi(Hayden) Liu, *Python Machine Learning by Example*, Packt, 2017.

CS4093D IMAGE PROCESSING LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Apply the fundamental concepts, called sampling and quantisation , of image processing to resample images.
CO2: Develop algorithms for simple and complex image processing problems using the various spatial domain and transform domain techniques.
CO3: Apply various filtering techniques and segmentation methods for image enhancement and image segmentation.

Theory (13 Hours)

An introduction to digital images- sampling, quantization. Basic image processing, arithmetic processing. Image enhancement and point operation. Image enhancement and spatial operation. Color images and models models. Frequency domain operations. Image Analysis

Practical (39 Hours)

1. Image resampling
2. Basic image processing, arithmetic processing
3. Image enhancement and point operation- Linear point operation, clipping, thresholding, negation, non-linear mapping, intensity slicing, image histogram, histogram equalization.
4. Image enhancement and spatial operation- Convolution, correlation, linear filtering, edge detection.
5. Color image processing - color models, color enhancement, color thresholding.
6. Frequency domain operations- fourier transform, freq domain filtering
7. Image analysis after basic image processing algorithms.

References:

1. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, 2007.
2. A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, 2002.

CS4094D ADVANCED COMPUTER NETWORKS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Analyze and compare performance of the existing protocols in the TCP/IP stack.
CO2: Improve the performance of the existing protocols/solutions
CO3: Experiment with networking protocols directly in the protocol stack.

Theory (13 Hours)

Review of socket programming basics, Introduction to ns-3, Introduction to mininet, Introduction to netkit, Software defined networking, Introduction to Linux IP networking

Practical (39 Hours)

1. A Single TCP Sender
2. Two TCP Senders Competing
3. TCP Loss Events and Synchronized Losses
4. TCP Reno versus TCP Vegas
5. TCP Competition: Reno vs Vegas
6. TCP Competition: Reno vs BBR
7. Wireless Simulation
8. Multiple Switches in a Line
9. IP Routers in a Line
10. IP Routers With Simple Distance-Vector Implementation
11. Linux Traffic Control (tc)
12. OpenFlow and the POX Controller
13. Implementation and Modification of the Linux Protocol Stack

References:

1. T. Issariyakul, *Introduction to Network Simulator NS2*, 2/e, Springer, 2012.
2. J. Liebeherr and M. E. Zarki, *Mastering Networks: An Internet Lab Manual*, 1/e, USA, Addison-Wesley Longman Publishing Co., 2003.
3. C. Benvenuti, *Understanding The Linux Network Internals*, 1/e, Om Books, 2006.

CS4096D SOFTWARE ENGINEERING LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Design and Develop a high quality software
CO2: Prioritize engineering tasks in software development
CO3: Work in a team for software development and interact with clients
CO4: Solve customer problems through software

Theory (13 Hours)

Introductory Lectures on the use of appropriate tools is to be given.
Peer review discussions of deliverables will also be done in theory sessions.

Practical (39 Hours)

Objective is to develop a significant software product using sound software engineering principles by small student groups. Choice of appropriate methodology and standard tools are also expected. The lab will have deliverables at each milestone of development.

1. Problem Statement / Product Specification
2. Project Plan – Project Management Tool to be identified and Estimation and Costing to be done.
3. Requirements Document – Specification Tool choice to be justified - In class Review
4. Design Document – Choice of Methodology to be justified - In class Review
5. Code and Test Report – Peer review documents of standards adherence to be provided
6. Demo – Integrated Product or Solution to the problem
7. Review of the process and analysis of variation from initial plan and estimation.

These steps may be replaced by Agile Methodologies and associated tool set like Trello, Git, Docker, CircleCi, Heroku etc. depending on the kind of project and availability of committed customers.

References:

1. R. S. Pressman, *Software Engineering: A Practitioner's Approach,* 6/e, McGraw Hill, 2008.
2. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering,* 1/e, Tata McGraw Hill, 2004.
3. K. Beck, *Extreme Programming,* 2/e, Pearson Education, 2006.
4. J Knapp, *SPRINT*, Bantam Press, 2016.

CS4097D OBJECT ORIENTED SYSTEMS LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Design and model solutions using Object Oriented Principles using UML
CO2: Implement reusable solution keeping quality in mind.

Theory (13 Hours)

Procedural vs. Object oriented approaches – Concept of Abstraction - Design and analysis using OO methodologies - Introduction to UML.

Practical (39 Hours)

The implementation has to be done using languages like C++/Java/C#.

1. Functions – Control structures – String handling – File handling
2. Error and Exception handling
3. Class – Object Instantiation
4. Principles of Inheritance, Encapsulation, Polymorphism – Overloading, Virtual functions.
5. OO Design with stress on interface specification.
6. Automated code generation and component reuse - UML

References:

1. B. Stroustrup, *The C++ Programming Language,* 3/e, Addison Wesley, 1997.
2. S. Oualline, *Practical C++ Programming,* 2/e, O'Reilly & Associates, 2002.
3. J. Nino and F. A. Hosch, *An introduction to programming and object oriented design using Java,* Wiley India, 2010

CS4088D ADVANCED HARDWARE LABORATORY

Prerequisites: NIL

L	T	P	C
1	0	3	3

Total Hours: 13T+39P

Course Outcomes:

Students will be able to :

CO1: Design and implement finite state machines and memory blocks
CO2: Design and implement simple and an enhanced processors
CO3: Implement algorithms in hardware using HDLs
CO4: Implement algorithms in SoC hardware using CUDA

Theory (13 Hours)

Review of logic circuits with HDLs, Introduction to FPGAs, Programming with FPGAs, Embedded Computer Architecture Fundamentals, Processor Design Flow, General-Purpose Embedded Processor Cores, Customizable Processors and Processor Customization, Designing Soft-Core Processors for FPGAs

Introduction to GPUs, Programming with GPus, CUDA C, Tegra SoC as GPU mobile processor. Jetson TX2 GPU onboard supercomputer. Jetson based compute intensive project works for computer vision, robotics, Audio/Video processing and medicine

Practical (39 Hours)

Design and implement the following using FPGA

1. Finite State Machines.
2. Memory blocks.
3. A simple processor.
4. An enhanced processor.
5. Algorithms in hardware.

Design and implement the following using Jetson TX2 GPU board

6. Speech processing application
7. video processing
8. AI based projects (using deep learning)
9. Path planning for computer vision
10. Medicine projects

References:

1. J. Nurmi, *Processor Design: System-On-Chip Computing for ASICs and FPGAs,* 1/e, Springer, 2007
2. S. Palnitkar, *Verilog HDL,* 2/e, Pearson Education, 2003
3. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1/e, Springer, 2017.
4. NVIDIA, Whitepaper NVIDIA Tegra X1 NVIDIA'S New Mobile Superchip, https://international.download.nvidia.com/pdf/tegra/Tegra-X1-whitepaper-v1.0.pdf

CS2003D INTRODUCTION TO PROGRAMMING

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: 39T + 26P

Course Outcomes:

Students will be able to

CO1: Design and implement programs for simple computational problems.
CO2: Describe elementary data structures and algorithms for standard computing problems.
CO3: Analyze the time complexity of simple algorithms.

Module 1: (13T+8P Hours)

Review of programming: Data types, Identifiers and keywords, variables, constants, arrays, expressions and statements, iterative and conditional constructs, break and continue, input and output statements, programming examples, bubble sort, insertion sort, sequential and binary search.

Module 2: (13T+10P Hours)

Introduction to complexity analysis: asymptotic notation and simple examples of analysis of iterative algorithms like bubble sort, insertion sort and binary search.
Subroutines: functions and parameter passing, call semantics, recursion, programming examples with recursive programs, recursive programs for binary search, quicksort, merge sort and heapsort. Complexity analysis of recursive programs using recurrences.

Module 3: (13T+8P Hours)

Structures and unions, pointers and dynamic memory allocation, programming examples: linked list, stack and queue implementation using linked lists, binary search trees, recursive inorder, preorder and postorder traversals, evaluation of arithmetic expressions.

References:

1. W. Kernighan, *The Practice of Programming*, Addison-Wesley, 1999.
2. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.
3. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3/e, MIT Press, 2009.

MA6010D DISCRETE MATHEMATICS

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to

CO1: have sufficient knowledge about Propositional calculus and predicate calculus
CO2: have knowledge about relations, functions partial order and Boolean algebra
CO3: be able to understand the concepts of semigroups, monoids, groups and sub groups
CO4: have knowledge about Rings, Integral domains, fields and field extensions

Module 1: (10 hours)

Propositional Calculus: Propositions, Truth tables , tautologies and contradictions, logical equivalence, logical arguments, normal forms, consistency completeness and independence, formal proofs , natural deduction. Predicate Calculus: predicates, quantifiers, arguments, theory of inference, resolution algorithm.

Module 2: (10 hours)

Relations and functions, pigeonhole principle, cardinals, countable and uncountable sets, diagonalization, equivalence relations and partitions, partial order, lattices, Boolean Algebra.

Module 3: (10 hours)

Semi groups, monoids, groups and subgroups, homomorphism, cosets, normal sub groups, products and quotients, Lagrange's theorem, permutation groups, Cayley's theorem.

Module 4: (9 hours)

Rings, Integral domains, fields, ideals and quotient rings, Euclidean domain, polynomial rings, division algorithm, field factorization, unique factorization, field extensions.

References

1. P. Grimaldi, *Discrete and Combinatorial Mathematics*, Addison Wesley, 1994.
2. J. P. Tremblay and R. Manohar, *Discrete Mathematical Structures with applications to Computer Science,* Tata McGraw Hill, New Delhi, 2003.
3. B. Kolman and R. C. Busby, *Discrete Mathematical Structures for Computer Science*, PHI, 1994.
4. C. L. Liu, *Elements of Discrete Mathematics,* 2/e, Mcgraw Hill, 1985.
5. J. L. Mott, A.Kandel, and T.P Baker, *Discrete Mathematics for Computer Scientists and Mathematicians*, 2/e, PHI, 1986.
6. J. K. Truss, Discrete Mathematics for Computer Scientists, Addison Wesley, 1999.
7. I. N. Herstein, *Topics in Algebra*, Wiley Eastern, 1975.

MA6020D STATISTICAL METHODS

Prerequisites: NIL

L	T	P	C
3	1	0	3

Total hours: 39

Course Outcomes:

Students will be able to

CO1: Handle problems involving random variables and functions of random variables.
CO2: Identify statistical problems and make use of statistical inference for decision making or choosing actions.
CO3: Collect and analyze data for extracting useful information.
CO4: Apply regression and correlation analysis for studying relationship between variables.
CO5: Use probabilistic and statistical analysis in various applications of engineering.

Module 1: (15 hours)

Probability distributions:- Introduction to Probability, Definitions and basic results, conditional probability and independence, Random variables, Mean and variance, Discrete probability distributions-Binomial distribution, Hyper- geometric distribution, Poisson distribution, Geometric distribution. Continuous probability distributions- Normal Distribution, Uniform distribution, Gamma distribution, Chi-Square distribution, Joint distribution of random variables.

Module 2: (12 hours)

Statistical inference I:- Population and samples, The sampling distribution of the mean (δ^2 known and δ^2 unknown), Sampling distribution of the variance, Point estimation and interval estimation, Tests of hypothesis, Hypothesis concerning one mean, Inference concerning two means. Estimation of variances, Hypothesis concerning one variance, Hypothesis concerning two variances.

Module 3: (12 hours)

Statistical Inference II – Inference concerning one proportion and two proportions, Analysis of r x c tables, Chi – square test for goodness of fit. Curve fitting by the method of least squares, Fitting straight lines, parabolas and other non-linear curves, Simple linear regression models, Inference concerning Correlation coefficient.

References:

1. R. A. Johnson and C. B. Gupta, *Miller & Freund's Probability and Statistics for Engineers*, 7/e, Pearson Education Inc, 2005.
2. W. H. Hines, Montgomery, et. al., *Probability and Statistics for Engineering*, John Wiley & Sons, Inc., 2003.
3. J. S. Milton and J. C. Arnold, *Introduction to Probability and Statistic*, 4/e, Tata McGraw-Hill, 2003.
4. Sheldon M. Ross, *A First Course in Probability*, 9/e, Pearson Education Inc, 2014.

MA6224D GRAPH THEORY AND COMBINATORICS

Prerequisites: NIL

L	T	P	C
3	1	0	3

Total hours: 39

Course Outcomes:

Students will be able to

CO1: have sufficient knowledge about Graphs, subgraphs, trees and matrix representation of graphs
CO2: have knowledge about connectivity, traversability and planarity of graphs
CO3: be able to find minimum spanning tree and shortest path in a graph.
CO4: be able to use techniques of generating functions and recurrence relations.

Module 1: (9 hours)

Graphs: subgraphs, paths and cycles, isomorphism, cut vertex, bridge, block, bipartite graph, complement of a graph, line graph, Degree sequence, Trees, metric in graph, eccentricity , centre, median, centroid, Matrix representation of graph.

Module 2: (10 hours)

Connectivity: Vertex and Edge connectivity, Whitney's theorem, n - connected graphs Menger's' theorem.
Traversability: Hamiltonian graphs: Ore's theorem, Posa's theorem, Other sufficient conditions for hamiltonicity, Euler graphs, Planar graphs, Euler formula, platonic bodies. Non planar graphs.

Module 3: (10 hours)

Graph Coloring, chromatic polynomials, The four color problem, The five color theorem.
Digraphs: Connectedness - Acyclic Digraph, Strong digraphs, Tournaments, Directed trees, binary trees, weighted trees and prefix codes, BFS, DFS, Kruskal's, Prim's, Dijkstra's & Floyd's algorithms.

Module 4: (10 hours)

Generating functions, Partitions of integers, The exponential generating function, The summation operator, recurrence relations, first order and second order nonhomogeneous recurrence relations, method of generating functions.

References:

1. G. Chartrand and P. Zhang, *Introduction to Graph Theory*, McGraw Hill International Edition, 2005.
2. C. Vasudev, *Graph Theory with Applications*, New Age international publishers, 2006.
3. R. P. Grimaldi, *Discrete and Combinatorial Mathematics: An Applied Introduction,* Addison Wesley, 1994.
4. C. R. Foulds, *Graph Theory Applications*, Narosa Publishing House, 1994.
5. F. Harary, *Graph Theory*, Addison Wesley, 1972.
6. B. Bollobas, *Modern Graph Theory*, Springer Verlag, 2005.

MA6005D OPTIMIZATION TECHNIQUES I

Prerequisites: NIL

L	T	P	C
3	0	0	3

Total hours: 39

Course Outcomes:

Students will be able to

CO1: Use mathematical modeling techniques such as linear programming for effective and efficient utilization of resources in decision making problems.
CO2: Apply Game theory techniques for decision making in competitive situations.
CO3: Apply Dynamic programming technique to sequential decision problems.
CO4: Represent decision making problems through networks and apply suitable techniques for determining shortest path in the network and maximal flow through network.

Module 1: (10 hours)

Elementary Linear Programming: Systems of linear equations & inequalities – Convex sets – Convex functions- Formulation of linear programming problems- Theory of Simplex method- Simplex Algorithm-Charnes M-Method- Two phase method-Duality in linear programming-Dual Simplex method

Module 2: (10 hours)

Advanced Linear Programming: Sensitivity analysis – Parametric programming- Bounded Variables problem- Transportation problem – Integrality property –MODI method- Degeneracy- Unbalanced problem – Assignment Problem – Development of Hungarian method – Routing problem.

Module 3: (10 hours)

Dynamic Programming and Game Theory: Nature of Dynamic Programming problem – Bellman's optimality principle. Cargo loading problem –Replacement problem- Multistage production planning and allocation problem-Rectangular Games- Two person –Zero sum games-Pure and mixed strategies-2xn and mx 2 games. Relation between theory of games and linear programming.

Module 4: (9 hours)

Network Path Models: Tree Networks – Minimal Spanning Tree - Kruskal's Algorithm, Prim's Algorithm – Shortest path problems- solution methods – Dijkstra's Method- Floyd's Algorithm –Network flow Algorithms-Maximal flow algorithm-The method of Ford and Fulkerson

References:

1. M. S. Bazarra, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network flows*, 2/e, John Wiley, 1990.
2. M.S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming Theory and Algorithms*, 2/e, John Wiley,1993.
3. G. Hadley, *Linear Programming,* Narosa Publishing House,1990.
4. F. S. Hillier and G. T. Lieberman, *Introduction to OR*, 7/e, Mc.Grand Hill,2010
5. H. A. Taha, *Operations Research-An introduction,* 6/e, Prentice Hall,India, 1999.
6. L. R. Foulds, *Graph Theory and Applications*, Springer, Delhi,1992

CS3099D PROJECT

Prerequisites: NIL

L	T	P	C
-	-	-	15

Course Outcomes:

Students will be able to

CO1: Conduct literature survey and document a problem specification, identify a problem to work on, and lay out a plan of action for the project towards solving the problem.
CO2: Design a solution for the problem.
CO3: Implement a solution and collect the observations and results of the work.
CO4: Document the literature survey, problem specification, design and the results and observations from the work into a report.

Each student shall identify a faculty member as the project guide, with whom they associate for the project work for a period of one semester.

Student, in consultation with the guide, shall (i) Identify an area of work and conduct a detailed literature survey of the relevant work in the area, (ii) Identify a problem and prepare a report of the problem she is going to work on, (iii) Design a solution to the problem identified. This could be done either as an internal project (at NIT Calicut) or as an internship (at a Company, under guidance from one guide within the company/R&D organization/Collaborating institution in addition to the internal guide at NIT Calicut). The solution shall be implemented and the results, observations and conclusions tabulated. The design, results and conclusions shall be collected to form a project report which shall be presented before a committee of faculty members designated to evaluate the project work.

References:

1. G. J. Alred, C. T. Brusaw, and W. E. Oliu, *The Handbook of Technical Writing*, 11/e, Bedford/St. Martins, 2015.
2. G. R. Marczyk, D. DeMatteo, and D. Festinger, *Essentials of Research Design and Methodology,* Wiley, 2005.

MA2003D MATHEMATICS IV

Prerequisites: MA1001D Mathematics I, MA1002D Mathematics II

L	T	P	C
3	1	0	3

Total Hours: 39

Course Outcomes:

Students will be able to :

CO1: have sufficient knowledge about vector spaces, linear transformation and be able to use these to diagonalise symmetric matrices.
CO2: have knowledge about different types of linear operators on inner product spaces and their spectral decomposition.
CO3: be able to identify analytic functions and find harmonic conjugates.
CO4: be able to find images of regions under complex transformations.
CO5: be able to evaluate line integrals in the complex plane
CO6: be able to use techniques of complex analysis to evaluate integrals of real valued functions.

Module 1: (9 hours)

Review of basic linear algebra topics, Direct sum of vector spaces, Rank-nullity theorem and its proof, Matrix representation of linear transformation, change of basis, Invariant subspaces, Polynomials applied to operators, Upper triangular representation for complex operators, Diagonalisation, Invariant subspaces on real vector spaces.

Module 2: (11 hours)

Inner product spaces, Orthogonal basis, Orthogonal projection, Best approximation, Linear functional, Adjoint of a linear transformation, Self-adjoint and normal operators, Spectral theorem for normal operators on complex inner product spaces, Spectral theorem for self adjoint operators, Normal operators on real inner product spaces, Positive operators, Isometries.

Module 3: (9 hours)

Complex functions, Derivative, Analytic function, Cauchy-Riemann equations, Laplace's equation. Geometry of analytic functions: Conformal mapping, Linear fractional Transformations, Schwarz - Christoffel transformation, Transformation by other functions.

Module 4: (10 hours)

Line integral in the Complex plane, Cauchy's Integral Theorem, Cauchy's Integral formula, Derivatives of analytic functions. Power series, Functions given by power series, Taylor series and Maclaurin series. Laurent's series, Singularities and Zeros, Residue integration method, Evaluation of real integrals.

References:

1. S. Axler, *Linear algebra done right*, 2/e, Springer, 2015.
2. S. Lipschutz and M. Lipson, *Schaum's outline of linear algebra*, 6/e, McGraw-Hill, 2017.
3. E. Kreyszig, H. Kreyszig, and E. J. Norminton, *Advanced engineering mathematics: international student version*, 10/e, New Delhi: Wiley, 2015.
4. C. R. Wylie and L. C. Barrett, *Advanced engineering mathematics*, 6/e, McGraw-Hill, 1995.