CS 6101D Mathematical Foundations of Computer Science

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Analyze and solve practical computing problems involving foundations of basic number theoretic and algebraic concepts.

CO2: Analyze and solve computing problems using fundamental principles of linear spaces and analyze algorithms that use these concepts.

CO3: Derive algorithmic solutions for algebraic computing problems.

Module 1: (13 T Hours)

Divisibility, gcd, prime numbers, fundamental theorem of arithmetic, Congruences, Fermat's theorem, Euler function, primality testing, solution of congruences, Chinese remainder theorem, Wilson's theorem.

Module 2: (13 T Hours)

Groups and subgroups, homomorphism theorems, cosets and normal subgroups, Lagrange's theorem, rings, finite fields, polynomial arithmetic, quadratic residues, reciprocity, discrete logarithms.

Module 3: (13 T Hours)

Vector spaces, basis, dimension, linear maps, rank nullity theorem, duality theorem, Eigenvalues and Eigenvectors.

Module 4: (13 T Hours)

Inner product spaces, orthogonality, orthogonal projections, Hermitian and unitary operators, spectral theorem for Hermitian and unitary operators, singular value decomposition, Cholesky decomposition.

References

1. K. Ireland and R. A. Rosen, *A Classical Introduction to Modern Number Theory,* 2/e, Springer, 1998.
2. Niven, H.S. Zuckerman, and Montgomery, *An Introduction to the Theory of Numbers*, 3/e, John Wiley and Sons, New York, 1992.
3. S. Axler, *Linear Algebra Done Right*, 2/e, Springer, 1997.

CS 6111D Algorithms and Complexity

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Apply methods for amortized analysis and probabilistic analysis to analyze algorithms.

CO2: Assess the hardness of problems using reductions, applying knowledge about basic complexity classes.

CO3: Analyze different problems for approximability and amenability to randomized approaches.

Module 1: (13 T Hours)

Analysis: RAM model – Notations, Recurrence analysis - Master's theorem and its proof - Amortized analysis - Advanced Data Structures: B-Trees, Binomial Heaps, Fibonacci Heaps, Disjoint Sets, Union by Rank and Path Compression

Module 2: (13 T Hours)

Graph Algorithms and Complexity: Matroid Theory, All-Pairs Shortest Paths, Maximum Flow and Bipartite Matching.

Module 3: (13 T Hours)

Randomized Algorithms: Fingerprinting, Pattern Matching, Graph Problems, Algebraic Methods, Probabilistic Primality Testing, De-Randomization

Module 4: (13 T Hours)

Complexity classes - NP-Hard and NP-complete Problems - Cook's theorem NP completeness reductions. Approximation algorithms – Polynomial Time and Fully Polynomial time Approximation Schemes.
Probabilistic Complexity Classes, Probabilistic Proof Theory and Certificates

References

1. D. C. Kozen, *The Design and Analysis of Algorithms*, Springer, 1992.
2. T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, PHI, 1990.
3. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
4. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.

CS 6103D Software Systems Laboratory

Prerequisites: NIL

L	T	P	C
1	0	6	4

Total hours: (13 T + 78 P)

Course Outcomes:

CO1: Apply scripting tools and programming languages for software development.

CO2: Use documentation tools for preparing documents, articles and presentations.

CO3: Design and build web based solutions using software engineering concepts.

Theory (13 Hours)

General purpose programming tools(e.g. Java, C++, use of GUI tools), Web programming tools (e.g. HTML, Java with applets/servlets/JSP/J2EE, CGI, Perl), Development Frameworks (Ruby on Rails, Django).

Tools for good software development process. Make/gmake, source code control systems (e.g. git), debuggers (e.g. gdb) and memory allocation debuggers, Introduction to Integrated Development Environments (e.g. eclipse).

Scripting languages (e.g. Python, Perl). Tools for text processing (e.g. AWK, Python, Lex, Yacc).

Exposure to document creation tools (e.g. Latex), plotting tools (e.g. gnuplot, Excel, pstricks).

The contents may be adapted to software practices and trends at the time of offering the course. Hence the contents in parenthesis are simply examples and not strict requirements.

Laboratory (78 Hours)

Programming and Data Structures - Review (12 Hours)
Web Development - Using web programming tools. Using framework. (24 Hours)
Software development tools in the Linux Environment (12 Hours)
Scripting languages - Text Processing (18 Hours)
Documentation and Plotting (12 Hours)

References

1. L. Wall, T. Christiansen and R. L. Schwartz, *Programming Perl*, 3/e, O'Reilly *Media,*2000.
2. S. Guelich, S. Gundavaram and G. Birznieks, *CGI Programming with Perl,* O'Reilly Media, Third Edition, June 2000.
3. M. Summerfield, *Programming in Python 3*, 2/e, Addison Wesley Professional, November 2009.
4. M. Summerfield, *Rapid GUI Programming with Python and Qt*, Prentice Hall, 2009.
5. M. Hart. *Ruby on Rails Tutorial,* Available online at https://www.railstutorial.org/book last accessed 12/4/2018.
6. Wikibook Contributors LaTeX, Wikibooks, available at https://upload.wikimedia.org/wikipedia/commons/2/2d/LaTeX.pdf last accessed 12/4/2018
7. J. Levine, *flex & bison*, O'Reilly Media, 1/e, 2009.
8. B. Eckel, *Thinking in Java*, 3/e, Prentice Hall, 2002, Available online at www.bruceeckel.com last accessed 28/3/2010.
9. B. Eckel, *Thinking in C++,* 2/e, Vol. 1 and Vol. 2, Prentice Hall. 2003, Available online at www.bruceeckel.com last accessed 28/3/2010.

CS 7198D Project Part 1

Prerequisites: NIL

L	T	P	C
0	0	20	14

Total hours: 260

Course Outcomes:

CO1: Demonstrate sound fundamentals in a chosen area of computing.

CO2: Identify and formulate a problem of research interest in the chosen area of computing.

CO3: Analyze the computing problem and propose solutions.

CO4: Effectively communicate the work at all stages of the project.

The student is expected to carry out supervised research in this course. An intensive literature in the chosen area, should result in sound knowledge in the area and result in the identification of a suitable research problem, and its formulation and analysis. Study of relevant supplementary literature, such mastering useful programming languages and tools for the problem, are also expected at this stage of the project. The student is expected to present three reports at different evaluation points during the semester, with clearly defined achievements and plans for further steps.

References

1. Relevant literature for the computing problem.

CS 7199D Project Part 2

Prerequisites: CS 7198D Project Part 1

L	T	P	C
0	0	28	16

Total hours: 364

Course Outcomes:

CO1: Reflectively analyze proposed solutions to the identified computing problem.

CO2: Design and develop solutions to the problem and analyze results.

CO3: Prepare a thesis report and defend the thesis on the work done.

CO4: Augment the knowledge base in the chosen area of computing, adhering to ethical practices at every stage.

The student is expected to demonstrate the core competency aimed by this course, i.e., the development of enhancements to the knowledge base in the area of interest in computing. The secondary competencies include the management of time bound projects involving research, analysis of problem complexities, design and development of effective solutions and communication of the project's progress, adhering to ethical practices at every stage. This stage of the project evaluates the state of maturity of these competencies. The student is expected to present two reports at intermediate stages, as well as prepare and defend a thesis on his research work.

References

1. Relevant literature for the computing problem.

CS 6102D Compiler Design

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Formulate new data-flow analysis to capture the information required for an optimization and build an algorithm for the optimization.

CO2: Translate intermediate code to machine code.

CO3: Evaluate the efficiency and optimality of a given optimization and suggest improvements.

Module 1: (10 T + 6 P Hours)

Review of compiler phases – Symbol Table Structure – Intermediate Representations. Control Flow Analysis: Basic Blocks and CFG, Dominators and Loops.

Module 2: (10 T + 7 P Hours)

Data Flow Analysis: Reaching Definitions, Available Expressions, and Live Variable Analysis. Optimizations: Redundancy Elimination – Loop Optimizations – Value Numbering.

Module 3: (10 T + 7 P Hours)

Static Single Assignment Form (SSA): SSA Construction – Optimizations on SSA Form. Register Allocation – Graph Colouring Algorithm.

Module 4: (9 T + 6 P Hours)

Machine Code Generation: Instruction Selection - Maximal munch and Dynamic programming Algorithm. Code Generation – Target Machine – Code Generation for Run- time Stage Management. Code Generation Algorithms.

References

1. A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques, and Tool*s. Pearson Education, 2007.
2. S. Muchnick., *Advanced Compiler Implementation*, Morgan Kaufmann Publishers, 1997.
3. A. W. Appel and J. Palsberg, *Modern Compiler Implementation in Java*, Cambridge University Press, 2002.
4. Y. N. Srikant and P. Shankar, *The Compiler Design Handbook: Optimization and Machine Code Generation*, CRC Press, 2003.

CS 6112D Operating System Design

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Describe the major components of modern Operating Systems beginning with a mobile OS to Cloud OS.

CO2: Analyze distributed scheduling and file systems.

CO3: Propose operating systems architecture for various hardware platforms.

Module 1: (10 T + 6 P Hours)

OS Structure: Monolithic, Microkernel, ExoKernel, Multi Kernel. Linux kernel - Process; Fork, Exec, Threads, Process Scheduling, O(1) Scheduler, Completely Fair Scheduler (CFS), Earliest Deadline First, Dominant Resource Fairness (DRF), Two-Level Scheduling, Lithe Scheduling.

Module 2: (10 T + 7 P Hours)

Process Synchronization, Semaphores, Monitors, IPC, Virtual Memory, TLBs, Paging, SLAB allocator, File Systems, Journaling, VFS, RAID, Device Driver Structure, Device Drivers and Synchronization.

Module 3: (10 T + 7 P Hours)

Mobile Operating System, History of Mobile OS, Design Considerations for Mobile OS, Process Management in Mobile OS, Android – An Unexpected Rival of iPhone, Memory Management in Mobile OS, Storage in Mobile OS, Android OS Stack, Linux Kernel Vs. Android Kernel, IPC in Android, Real time Operating Systems Mars Pathfinder: Priority Inversion Problem, Multi-core OS, Scheduling, Load Balancing

Module 4: (9 T + 6 P Hours)

Parallel Operating systems, Shared and Parallel File Systems, PVFS, GPFS, Distributed Operating Systems: Architectures, Synchronization, Communication, Resource Management: Distributed File Systems, GFS, Distributed Shared Memory, code migration and Distributed Scheduling; Failure Recovery, Virtualization, Hypervisors, VMs, Para Virtualization and Full Virtualization, Cloud Operating System, Recent Trends in Cloud Computing

References

1. A. S. Tanenbaum, *Distributed Operating Systems,* 4/e, Pearson, 2017.
2. C. Crowley, *Operating systems - A Design Oriented Approach*, Tata McGraw-Hill, 1998.
3. Current literature.

CS 6121D Computability Theory

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Analyze the models of computation of various computing capacities and prove separation results.

CO2: Analyze metrics and structures of computational complexity and prove inclusion relationships between Structures.

CO3: Demonstrate understanding of the computational complexity of concrete problems and place given problems into the appropriate levels of complexity.

Module 1: (13 T Hours)

Automata Theory**:** Review of Induction and Diagonalization - Finite Automata – Myhill-Nerode Theorem, Pumping Lemma. Turing Machines – Turing Acceptable, Decidable and Enumerable languages**.**

Module 2: (13 T Hours)

Computability: Closure Properties of RE and R Sets - Undecidability – Reductions – RE Completeness – Non - RE languages - Rice Theorems.

Module 3: (13 T Hours)

Introduction to Complexity: Time and Space Complexity Classes – Relations between Deterministic and Non- Deterministic Time and Space Complexity Classes – Hierarchy Theorems - Savitch's Theorem - Immerman Szelepcsenyi Theorem.

Module 4: (13 T Hours)

Model Independent Complexity Classes, NP – Completeness – Cook's Theorem – Reductions – PSPACE Completeness, NL Completeness.

References

1. M. Sipser, *Introduction to the Theory of Computation*, PWS Publishing Company, 1997.
2. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
3. J. E. Hopcroft and J. D. Ullman, J. D, *Introduction to Automata Theory, Languages and Computation*, 1/e, Addison Wesley, 1979.

CS 6122D Computer Architecture

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Analyze the performance of a machine using standard benchmark suites.

CO2: Describe computer architecture concepts and mechanisms related to the design of modern processors, memories, and explain how these concepts and mechanisms interact.

CO3: Define Thread Level Parallelism (TLP) and design techniques to maximize performance using TLP in uni-processors and multiprocessors.

CO4: Define Data level and Request level parallelism, and design hardware to exploit DLP and RLP.

Module 1: (11 T + 7 P Hours)

Performance Evaluation, Processor Architecture, Pipelining, Pipeline Hazards, Issues in Pipelined processor implementation. Instruction Level Parallelism, Hardware and Compiler Support for Branch Prediction, Out-of-order Execution, Speculative Execution and other techniques for high-performance computing.

Module 2: (11 T + 7 P Hours)

Limits to ILP, Thread Level Parallelism, Simultaneous Multithreading, Multiple Processor Systems, Shared Memory System, Memory Models, Cache Coherence. Memory Consistency, Shared Memory Consistency models, Relaxed Consistency Models, Multicore Interconnect, Network - on - chip.

Module 3: (8 T + 7 P Hours)

Instruction and Data Cache Organizations, Multi Level Caches, Parallel Memory Systems, Support for Virtual Memory.

Module 4: (9 T + 5 P Hours)

Data Level Parallelism, Vector Processors, SIMD extensions, GPU architecture, GPU programming, Request Level Parallelism, Architecture of Data Centers, Virtual Machine Monitor, Architectural Support for VMM, Domain Specific Architecture

References

1. J. L. Hennessy, D. Patterson, *Computer Architecture – A quantitative Approac*h, 6/e, Morgan Koffman, 2017.
2. J. P. Shen, M. Lipasti. *Modern Processor Design – Fundamentals of Superscalar Processors*, McGraw Hill International Edition, 2005.
3. Current Literature.

CS 6123D Database Design

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Translate from Entity Relationship Model to Relational Model in real life applications.

CO2: Design and normalize a database application using relational model.

CO3: Demonstrate the knowledge of Object Oriented, Object relational and extended relational database systems and analyze the real time uses of these databases.

CO4: Build data mining algorithms on emerging database technologies.

Module 1: (10 T + 6 P Hours)

Database System concepts and applications, Data modeling using Entity- Relationship model, Record Storage and File organization**.**

Module 2: (9 T + 7 P Hours)

The Relational Data Model, Relational constraints and the Relational Algebra, SQL, ER to Relational mapping, Examples of RDBMS.

Module 3: (10 T + 7 P Hours)

Database Design Theory and Methodology- Functional Dependencies and Normalization for Relational Databases, Relational Database design algorithms, Practical Database Design and Tuning**.**

Module 4: (10 T + 6 P Hours)

Object Oriented Database concepts, Object Relational and Extended Relational Database Systems, Data Warehousing and Data Mining, Emerging Database Technologies and Applications.

References

1. R. Elmasri, S.B Navathe. *Fundamentals of Database Systems*, 7/e, Pearson Education, 2000
2. T. Connolly, C. Begg, *Database Systems*, 3/e, Pearson Education, 2003.
3. A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, Tata McGraw Hill, 2003.
4. J. D. Ullman, *Principles of Database Systems*, Galgotia Publications, 1996.

CS 6124D Topics in Programming Languages

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Develop formal semantics for programming language constructs.

CO2: Model Programming Language features using Lambda Calculus.

CO3: Design type systems for language safety.

CO4: Design a programming language (formal semantics and type system) with required constructs.

Module 1: (8 T + 6 P Hours)

Introduction to Programming Languages. Untyped Arithmetic Expressions: Syntax and Semantics - Properties of the language of Untyped Arithmetic Expressions.

Module 2: (11 T + 6 P Hours)

Untyped Lambda Calculus: Syntax, Operational Semantics, Evaluation strategies – Programming in Lambda Calculus. Typed arithmetic Expressions: Typing relation – Type safety.

Module 3: (10 T + 7 P Hours)

Simply Typed lambda Calculus: Typing relation – Properties of the Language – Type safety. Extensions: Basic Types, Derived Forms, Let Bindings.

Module 4: (10 T + 7 P Hours)

Extensions to Lambda Calculus: Pairs, Tuples, Records, Sums, Variants, References, Exceptions. Subtyping, Recursive Types, Polymorphism.

References

1. B. C Pierce, *Types and Programming Languages*, MIT Press, 2002.
2. A. B. Tucker, *Handbook of Computer Science Engineering,* CRC Press, 1996.
3. M. L. Scott, *Programming Languages Pragmatics,* Elsevier, 2004.

CS 6125D Computer Networking

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Analyze and compare new protocols in computer networks.

CO2: Evaluate performance of protocol enhancements using modern tools.

CO3: Propose new solutions to recent problems of interest in literature and compare different possible solutions.

Module 1: (9 T + 7 P Hours)

Overview of computer networks, TCP/IP protocol, Application layer protocols, Software defined networking, content distribution, Web 2.0, overlay networks, P2P networks.

Module 2: (9 T + 7 P Hours)

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

Module 3: (11 T + 7 P 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 4: (10 T + 5 P Hours)

MAC protocols for high-speed LANS, MANs, Wireless LANs and mobile networks, VLAN. Fast access technologies. Gigabit Ethernet. Multimedia networking, Network management.

References

1. W. R. Stevens, *TCP/IP Illustrated*, Vol. 1: The Protocols, Addison Wesley, 1994.
2. G. R. Wright, *TCP/IP Illustrated,* Vol. 2: The Implementation, Addison Wesley, 1995.
3. P. Loshin, *IPv6: Theory, Protocol, and Practice,* 2/e, Morgan Kaufmann, 2003.

CS 6131D Logic and Computation

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Apply Linear Temporal Logic (LTL), Computation Tree Logic (CTL), Real Time Computation Tree Logic (RTCTL) for modelling of systems.

CO2: Analyze and choose an appropriate logic for a system and model the system using the logic, to improve problem solving skills.

CO3: Apply the theory learned by the usage of SPIN, SMV, Uppaal and Kronos.

Module 1: (10 T + 7 P Hours)

Temporal Logic based verification, Buchi automata, Linear Temporal Logic (LTL), Relational product , Modelling systems

Module 2: (9 T + 7 P Hours)

Computation Tree Logic (CTL), Concept of fairness, Symbolic model checking Tools: Spin**,** SMV

Module 3: (10 T + 6 P Hours)

Verification of infinite-state systems, Verification of real-time systems, Timed automata, Modelling real-time systems, RTCTL. Complexity Consideration, Tools: Uppaal, Kronos

Module 4: (10 T + 6 P Hours)

Verification of pushdown systems, Verification of security protocols, Formal Methods, Abstract Interpretation Model

References

1. E. M. Clarke, O. Grumberg, and D. Peled, *Model Checkin*g. MIT Press, 1999.
2. B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, P. Schnoebelen, and P. McKenzie, *Systems and Software Verification: Model-Checking Techniques and Tools*, Springer Verlag, 2001.
3. J. A. Gallier, *Logic for Computer Science Foundations of Automatic Theorem Proving,* Wiley, 1986.

CS 6132D Topics in Algorithms

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Illustrate and analyze Linear Algebra based algorithms for LUP decomposition and matrix inversion.

CO2: Design and analyze randomized algorithms and design a deterministic algorithm from a given randomized algorithm, using the method of conditional expectations.

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 : (10 T + 7 P 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: (10 T + 7 P Hours)

Randomized Algorithms : Maxcut, Vertex cover, Pattern Matching, Graph Problems, Algebraic Methods, Probabilistic Primality Testing, De-Randomization using conditional expectations.

Module 3: (10 T + 6 P Hours)

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

Module 4: (9 T + 6 P 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. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
2. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
3. T. H. Cormen, C. E. Leiserson, R. L. Rivest. and C. Stein, *Introduction to Algorithms*, 3/e, Prentice Hall India, 2010.
4. D. C. Kozen, *The Design and Analysis of Algorithms*, Springer Verlag N.Y, 1992.

CS 6133D Game Theory

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Demonstrate an understanding of Non Cooperative game theory and Nash equilibrium.

CO2: Evaluate various strategies for non-cooperative game theory.

CO3: Design different mechanism including optimal mechanisms.

CO4: Demonstrate an understanding of co-operative game theory and evaluate various solution concepts.

Module 1: (7 T + 6 P Hours)

Introduction to Non Cooperative Game Theory: Extensive Form Games, Strategic Form Games, Pure Strategy Nash Equilibrium

Module 2: (9 T + 7 P Hours)

Noncooperative Game Theory (in detail), Mixed Strategies, Existence of Nash Equilibrium, Computation of Nash Equilibrium, Two Player Zero-Sum Games, Bayesian Games

Module 3: (11 T + 6 P Hours)

Mechanism Design : An Introduction, Dominant Strategy Implementation of Mechanisms, Vickrey - Clarke - Groves Mechanisms, Bayesian Implementation of Mechanisms, Revenue Equivalence Theorem, Design of Optimal Mechanisms

Module 4: (12 T + 7 P Hours)

Cooperative Game Theory, Correlated Strategies, Correlated Equilibria, The Two Person Bargaining Problem, Games in Coalitional Form, The Core Shapley Value, Other Solution Concepts for Co-operative Games.

References

1. R. B. Myerson, *Game Theory: Analysis of Conflict*, Harvard University Press, September 1997.
2. A. Mas-Colell, M. D. Whinston, and J. R. Green, *Microeconomic Theory*. Oxford University Press,
   New York, 1995.
3. M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, The MIT Press, August 1994.
4. P. D. Straffin, *Game Theory and Strategy*, The Mathematical Association of America, Jan1993.
5. K. Binmore, *Fun and Games : A Text On Game Theory*, D. C. Heath & Company, 1992.
6. P. Klemperer, *Auctions: Theory and Practice*, The Toulouse Lectures in Economics, Princeton University Press, 2004.

CS 6134D Quantum Computation

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Analyze the quantum computational model and compare with the classical models of computation with respect to computational power and algorithmic efficiency.

CO2: Judge the applicability of translating classical algorithmic solutions to quantum models to achieve Higher efficiency and perform the translation when applicable.

CO3: Illustrate and summarize well known quantum algorithms for problems like integer factorization and search.

CO4: Illustrate and summarize well known methods for quantum error correction.

Module 1: (10 T Hours)

Foundations:Finite Dimensional Hilbert Spaces – Tensor Products and Operators on Hilbert Space – Hermitian and Trace Operators - Basic Quantum Mechanics necessary for the course.

Module 2: (10 T + 13 P Hours)

Model of Computation:Quantum Gates and operators and Measurement – Quantum Computational Model – Quantum Complexity – Schemes for Physical realization (Only peripheral treatment expected).

Module 3: (9 T + 13 P Hours)

Algorithms and Complexity Shor's Algorithm – Application to Integer Factorization – Grover's Algorithm – Quantum Complexity Classes and their relationship with classical complexity classes**.**

Module 4: (10 T Hours)

Coding Theory Quantum Noise – Introduction to the theory of Quantum Error Correction –Quantum Hamming Bound – Coding Schemes – Calderbank-Shor-Steane codes – Stabilizer Codes.

References

1. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2002.
2. J. Gruska, *Quantum Computing*, McGraw Hill, 1999.
3. P. R. Halmos, *Finite Dimensional Vector Spaces*, Van Nostrand, 1958.
4. J. Brown, *Minds, Machines and the Multiverse*: *The Quest for the Quantum Computer,* Simon and Schuster, 2000.

CS 6135D Logic for Computer Science

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Apply propositional and predicate logic to model real life problems, and prove theorems associated with these logic and proof systems.

CO2: Demonstrate an improvement in logical thinking skills.

CO3: Demonstrate the concept completeness and compactness.

CO4: Explain Incompleteness, examine logical characterization of Complexity Classes, discuss open problems in the area and assess how to address the problems and propose new solutions (if possible) to them.

Module 1: (6 T + 2 P Hours)

Propositional Logic**:** Review of undecidability and Complexity classes, Formal Systems, Syntax and Semantics of Propositional Calculus, Completeness Theorem, Compactness**.**

Module 2: (11 T + 8 P Hours)

Predicate Logic**:** Syntax and semantics, Herbrand's Expansion Theorem, Proof Systems, Completeness and Compactness, Undecidability of Satisfiability.

Module 3: (11 T + 8 P Hours)

Incompleteness: First order Axiomatization of number theory, Godel's Incompleteness Theorem, Limitations of first order logic.

Module 4: (11 T + 8 P Hours)

Logic and Computation: Bucchi's Theorem, Logical Characterization for Complexity classes, Fagin's Theorem.

References

1. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
2. J. A. Gallier, *Logic for Computer Science Foundations of Automatic Theorem Proving*, Wiley, 1986.
3. E. M. Clarke, O. Grumberg, and D. Peled, *Model Checking.* MIT Press, 1999.

CS 6136D Topics in Combinatorial Algorithms

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Formulate a given combinatorial optimization problem into a linear programming/integer programming problem when applicable, and judge whether a given combinatorial optimization problem provides a inear/integer programming formulation that can be efficiently solved or approximated.

CO2: Formulate matching and network flow problems in graph theory as integer linear programs and establish duality relations between them, Analyze correctness and complexity of classic algorithms like Edmond's Blossom algorithm.

CO3: Design exact/approximation algorithms for integer linear programming problems using randomized rounding and / or primal-dual schema.

CO4: Analyze the algorithms designed for integer/linear programming formulations using methods of probabilistic analysis when applicable, judge when a problem is non-approximable.

Module 1: (10 T + 8 P Hours)

Primal dual theory of linear programming, and application to max flow, Min cuts, Dijkstra's algorithm and Floyd Warshall algorithms. Applications to graph theory - Konig's theorem, Hall's theorem, Menger's theorem.

Module 2: (10 T + 8 P Hours)

Application of primal dual theory to matching, bipartite matching, non bipartite matching, spanning trees and matroids and other related combinatorial optimization problems**.** Hungarian algorithm, Hopcroft Karp algorithm, Edmonds' Blossom maximum matching algorithm for general graphs.

Module 3: (11 T + 8 P Hours)

Approximation: Primal Dual approximation algorithms for set cover, Maximum satisfiability, Steiner tree, multicut, Steiner forest, sparsest cut and k-medians. Randomized rounding algorithm for MAX-SAT, The probabilistic method, Derandomization using conditional expectations.

Module 4: (8 T + 2 P Hours)

Hardness of Approximation: PCP theorem (no proof), gap reductions, examples of non approximable problems, gap amplification, self reducibility, gap amplification using expanders.

References

1. U. Vazirani, *Approximation Algorithms*, Springer 2003.
2. D. P. Williamson and D. B. Shmoys, *The Design of Approximation Algorithms*, Cambridge University Press, 2011.
3. C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover, 1998.
4. G. Ausiello et.al., *Complexity and Approximation: Combinatorial Algorithms and their Approximability Properties*, Springer, 2002.

CS 6139D Computational Geometry

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Design algorithms for geometric problems and analyze these algorithms.

CO2: Analyze the correctness and the theoretical guarantee of the geometric algorithms.

CO3: Formulate a geometric solution for an application and design an efficient algorithm with the most suitable data structure.

CO4: Write advanced programs using CGAL.

Module 1: (10 T + 10 P Hours)

Introduction to Computational Geometry, its applications. 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, Convex partitioning. Introduction to Computational Geometric Algorithms Library (CGAL) and openGL and coding of simple programs with visualization using QT.

Module 2: (10 T+ 6 P 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, Introduction to polyhedra and convex hull in three dimensions. Applications of convex hull.
Implement Convex Hull algorithms and one application using CGAL & visualization using QT.

Module 3: (10 T + 5 P 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 with theoretical guarantee. Implement one application of Voronoi diagram/ Delaunay triangulation using CGAL & visualization using QT.

Module 4: (9 T + 5 P Hours)

Arrangements, Incremental algorithm, Voronoi diagram, Delaunay triangulation and Medial axis transform in three dimensions and their applications such as decomposition.
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*, Revised Second Edition, Springer-Verlag,2000.
3. F. P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

CS 6140D Topics in Computational Geometry

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

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 T + 7 P Hours)

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

Module 2: (10 T + 7 P Hours)

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

Module 3: (10 T + 7 P Hours)

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

Module 4: (9 T + 5 P 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.

CS 6141D Distributed Computing

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Design solutions for software development problems involving asynchronous clocks.

CO2: Develop algorithms for distributed problems involving distributed mutual exclusion.

CO3: Solve problems of distributed nature where leader election issues are present.

CO4: Implement self stabilizing algorithms for failure handling in distributed applications.

Module 1 : (10 T + 7 P Hours)

Characterization of Distributed Systems, System Models, Networking and Internetworking, Interprocess communication

Module 2 : (10 T + 7 P Hours)

Logical clocks, verifying clock algorithms, Mutual Exclusion, Mutual exclusion using timestamps, tokens and Quorums.

Module 3 : (9 T + 6 P Hours)

Name Services and Domain Name System, Directory and Discovery Systems, Drinking philosophers problem, leader elections, Global state, Termination Detection

Module 4: (10 T + 6 P Hours)

Transactions and Concurrency Control, Distributed Transactions, Distributed Deadlocks, Transaction Recovery, Fault-tolerant Services, Distributed Shared Memory, Distributed consensus.

References

1. V. K. Garg, *Elements of Distributed Computing*, Wiley Interscience, 2002.
2. N. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers Inc., 1996.
3. G. Coulouris, J. Dollimore, and T. Kindberg, *Distributed Systems Concepts And Design*, 3/e, Addison Wesley 2004.
4. A. S. Tanenbaum and V. S. Maarten, *Distributed Systems Principles and Paradigms*, Pearson Education 2004.
5. R. Chow and T. Johnson, *Distributed Operating Systems and Algorithms*, Addison Wesley 2003.
6. A. S. Tanenbaum, *Distributed Operating Systems*, Pearson Education 2005.
7. A. D. Kshemkalyani and M. Singhal, *Distributed Principles, Algorithms, and Systems,* Cambridge University Press, 2008.

CS 6142D Topics in Computer Architecture

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Read, understand and analyze articles containing topics of current interest in computer architecture.

CO2: Use different modern tools for performance analysis of design enhancements and analyze the system level implications of design enhancements.

CO3: Critically analyze different architecture level design options, and evolve thought processes leading to different solutions.

CO4: Propose new solutions to issues of current interest, and do comparative analysis of different possibilities.

Module 1: (10 T + 7 P Hours)

Advanced ILP Exploitation Techniques**:** Hardware and software techniques for ILP extraction, speculative execution, studies on ILP.

Module 2: (9 T + 6 P Hours)

Multithreaded processors and Multicore processors Concept, methodologies and analysis. Speculative multithreading.Multicore processor design and compilation issues, scheduling. CMPs and Polymorphic processors Concept, Studies and Analysis

Module 3: (10 T + 7 P Hours)

Simulators in Computer Architecture Introduction – methods, ADLs, traces, dynamic compilation. Multicore simulators. Functional and performance Simulators.

Module 4: (10 T + 6 P Hours)

Processor Based Security. Virtualization, Virtual Machines. Hypervisors. Security Issues.

References

1. J. L. Hennessy and D. Patterson, *Computer Architecture: A Quantitative Approach*, 6/e, Morgan Kaufmann, 2017.
2. ACM SIGARCH Computer Architecture News.
3. The WWW Computer Architecture page. http://www.cs.wisc.edu/arch/www/ last accessed 23/3/2016.

CS 6143D Trends in Middleware Systems

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Define Middleware framework for event based systems.

CO2: Apply distributed composite event detection architectures.

CO3: Implement query processing for high-volume XML message brokering with the help of tools like IBM
Websphere, MQ, TIBCO and Rendezvous.

CO4: Analyze topic based systems, peer-to-peer systems and overlay routing algorithms using case studies.

Module 1: (11 T + 7 P 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: (10 T + 6 P 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: (9 T + 6 P Hours)

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

Module 4: (9 T + 7 P 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. Franklin, *Query Processing for High-Volume XML Message Brokering*,VLD2003.
4. C. Chan, M. Garofalakis, and R. Rastogi, *RE-Tree: An Efficient Index Structure for Regular Expressions*, VLDB 2002.
5. R. P. Pietzuch, B. Shand, and J. Bacon, *A Framework for Event Composition in Distributed Systems*, Proc. of the 4th Int. Conf. on Middleware (MW'03).
6. A. Carzaniga and A.L. Wolf, *Forwarding in a Content-Based Network*, Proceedings of ACM SIGCOMM 2003. p. 163-174. Karlsruhe, Germany, August, 2003.

CS 6144D Multicore Systems

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Describe Multicore architectures and develop programs using optimization techniques for multicore Systems.

CO2: Describe pipeline design issues, memory hierarchy and memory system issues in multicore systems.

CO3: Analyse the performance of multicore processors using benchmark suites.

CO4: Solve problems using message passing model, programming with MPI and openMP in multicore Systems.

Module 1: (12 T + 6 P Hours)

Introduction to multicore architectures, issues in writing code for multi-core architectures, Challenges in developing programs for multi-core architectures, Introduction - Program optimizations techniques and building these techniques in compilers. Pipeline Design issues, memory hierarchy and memory system issues in multicore system. Memory consistency models. Synchronization primitives; Performance implications in shared memory programs;

Module 2: (10 T + 6 P Hours)

Chip Multiprocessors: Why CMP(Moore's law, wire delay); shared L2 vs. tiled CMP; core complexity;power/ performance; Multicore, multiprocessor, super speculative & VLIW architectures – a comparative study. Chip multiprocessor case studies.

Module 3: (8 T + 7 P Hours)

Optimization: Analysis & Optimization for multicore architectures, Dataflow analysis, Pointer analysis, alias analysis; Data dependence analysis, solving data dependence equations (integer linear programming problem) Loop optimizations; Memory hierarchy issues in code optimization.

Module 4: (9 T + 7 P Hours)

Multicore programming methodologies and measuring performance, Case studies from Applications such as Digital Signal Processing, Image processing, Speech processing.

References

1. M. J. Quinn, *Parallel Programming in C with MPI and OpenMP*, McGraw Hill, 2003.
2. D. E. Culler, J. P. Singh, and A. Gupta, *Parallel Computer Architecture*: A Hardware/Software Approach, 1998.
3. S. Akhter and J. Roberts, Multi-core Programming, Increasing Performance through Software Multi-threading, Inter Press, 2006.

CS 6151D Software Engineering

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1 : Create a Formal Software Requirements Specification Document through a Requirements Engineering process for the System.

CO2: Create a Software Design Document by developing UML Diagrams to capture various design aspects of the Solution.

CO3: Apply principles of Software Management for Managing Resources to build a High Quality and Reliable Software Product.

CO4: Apply improved design methods and programming styles for producing quality software.

Module 1: (9 T + 7 P Hours)

The Software Process: Life cycle models, from specification to maintenance. Critical systems, Requirements Engineering: Functional and non-functional requirements, preparing and validating feasibility studies

Module 2: (10 T + 6 P Hours)

Software Design: Architectural, Object Oriented, Real-time, and Reusable Designs. Object Oriented design with UML

Module 3: (9 T + 7 P Hours)

Software Project Management: Managing Process, People, Product and Quality, Reliability

Module 4: (11 T + 6 P Hours)

Formal specification and verification of programs and software. Project Delays - The symptoms, reasons, and solutions. Performance concerns - Improving design methods and programming styles for producing better software

References

1. I. Sommerville, *Software Engineering*, 6/e, Pearson Education Asia, 2001.
2. B. Oestereich, *Developing Software with UML*, Addison Wesley, 2002.
3. S. McConnell, *Code Complete*, Microsoft Press, 1993.
4. F. P. Brooks, Jr., *Mythical Man-Month*, 2/e,Addison Wesley, 1995.
5. J. L. Bentley, *Programming Pearls*, 2/e, Addison Wesley, 1999.

CS 6152D Object Oriented Modeling and Design

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:
CO1: Create requirements model using UML class notations and use-cases based on statements of user requirements, and to analyze requirements models for correctness and quality.

CO2: Design of an object oriented system from the requirements model and specify it in terms of a high-level architecture description, and low-level models of structural organization and dynamic behaviour using UML class, object, and sequence diagrams.

CO3: Comprehend the nature of design patterns and apply these patterns in creating object oriented design. CO4: Evaluate a given design.

Module 1 : (9 T + 7 P Hours)

Structural Modeling: Object Oriented Fundamentals, Basic structural Modeling, UML Model, Class Diagrams, Object Diagrams, Packages and Interfaces, Case Studies**.**

Module 2 : (11 T + 6 P Hours)

Behavioral and Architectural Modeling: Use Case Diagrams, Interaction Diagrams, State Chart Diagrams, Collaborations, Design Patterns, Component Diagrams, Deployment Diagrams, Case Studies

Module 3 : (9 T + 7 P Hours)

Object oriented Testing Methodologies: Implications of Inheritance on Testing, State Based Testing, Adequacy and Coverage, Scenario Based Testing, Testing Workflow, Case Studies , Object Oriented Metrics

Module 4: (10 T + 6 P Hours)

Components: Abuses of inheritance, danger of polymorphism, mix-in classes, rings of operations, class cohesion and support of states and behavior, components and objects, design of a component, lightweight and heavyweight components, advantages and disadvantages of using components.

References

1. J. M. Page, *Fundamentals of Object Oriented Design in UML*, Pearson Education, 2002.
2. G. Booch, J. Rumbaugh and I. Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley, 2002.
3. A. Bahrami, *Object Oriented System Development*, McGraw Hill, 2003.
4. J. Baugh, I. Jacobson, and G. Booch, *The Unified Modeling Language Reference Manua*l, Addison Wesley, 1999.
5. C. Larman, *Applying UML & Patterns: An Introduction to Object-Oriented Analysis and Design*, Addison Wesley, 2002.
6. R. Ooley and P. Stevens, *Using UML: Software Engineering with Objects & Components*, Addison Wesley, 2000.

CS 6154D Topics in Database Design

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Study and analyze articles containing topics of current interest in database design.

CO2: Critically analyze Parallel, Distributed and Object oriented databases, and solve advanced problems in nternet databases with the help of Data Mining Algorithms.

CO3: Experiment with partial and temporal databases like MongoDB, Hadoop GIS, and discuss the concepts of mobile and multimedia databases.

CO4: Design, develop a database project and deploy efficient IT solutions using free and open software to help society.

Module 1: (10 T + 7 P Hours)

Parallel and Distributed Databases: Architecture of Parallel Databases, Parallel Query Optimization, Distributed DBMS Architectures, Distributed Query Processing, Distributed Concurrency Control, Distributed recovery.

Module 2: (10 T + 7 P Hours)

Internet Databases and Data Mining : XML –QL, Ranked Keyword searches on the Web, Data Mining, Clustering, Similarity Search over Sequences**.**

Module 3: (9 T + 7 P Hours)

Object Oriented Database Systems: User Defined ADTs, Objects, Object Identity and Reference types, Database Design for ORDBMS, OODBMS, Comparison of RDBMS with OODBMS and ORDBMS**.**

Module 4: (10 T + 5 P Hours)

Spatial and Deductive Databases: Spatial and Temporal Databases, Temporal Logic, Spatial Indexes, Introduction to Recursive Queries, Introduction to Mobile Databases, Main Memory and Multimedia Databases

References

1. R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 3/e, Addison Wesley.
2. P. O'Neil and E. O'Neil , *Database Principles, Programming, and Performance*, 2/e, Harcourt Asia, Morgan Kaufman.
3. A. Silberschatz , H. F. Korth, and S. Sudarshan , *Database System Concepts*, Tata McGraw Hill, 2003.
4. J. D. Ullman, *Principles of Database Systems*, Galgotia Publications,1996.
5. C. J. Date, *An Introduction to Database Systems*, Addison Wesley, 2000.
6. R. Ramakrishnan and J. Gehrke, *Database Management Systems*, 3/e, McGraw Hill, 2004.

CS 6161D Embedded Systems and Applications

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Understand the basic structure and design of an Embedded System.

CO2: Understand the basics of RTOS for Embedded systems.

CO3: Develop the programming concepts of Embedded Systems.

CO4: Analyse the recent applications trends in embedded systems.

Module 1: (10 T + 5 P Hours)

Embedded Systems, applications of embedded systems - automotive system, mobile phones, Consumer Electronics, Satellite systems, artificial pacemaker, trends in embedded software development

Module 2: (12 T + 7 P Hours)

Core of the Embedded System: General Purpose and Domain Specific Processors, Microcontrollers, DSPs, FPGAs, ASICs, PLDs, Memory: ROM, RAM,. Reset Circuit, Brownout Protection Circuit, Oscillator Unit, Real Time Clock, Watchdog Timer, Interrupts. Embedded OS; RTOS, Tasks, Process and Threads, Multiprocessing and Multitasking, Task Scheduling, Shared Memory, Message Passing, Remote Procedure Call and Sockets, Task Synchronization: Task Communication/Synchronization Issues, Device Drivers, How to Choose an RTOS

Module 3: (7 T + 7 P Hours)

Software engineering practices in the embedded software development process, embedded software development environments.. Recent trends in Embedded Systems Circuit, Oscillator Unit

Module 4: (10 T + 7 P Hours)

Embedded Software Development Tools, Host and target machines – Linkers / Locators for Embedded Software – Debugging techniques, Case studies: Top 10 Single Board Computers, Raspberry Pi 3, Arduino Uno R3, NVIDIA's Jetson TX1 - Credit-Card Sized Supercomputers, VP889 high-performance FPGA embedded computing board for electronic warfare (EW); radar and satellite communications

References

1. S. Heath, *Embedded System Design*, Elsevier, 2/e, 2004.
2. D. E. Simon, *An Embedded Software Primer* - Pearson Education.
3. J. G. Ganssle, *The Art of Designing Embedded Systems*, Butterworth-Heinemann, 1999.
4. DreamTech Software Team, *Programming of Embedded Systems*, Wiley DreamTech, 2002.

CS 6171D Natural Language Processing

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Distinguish the applications of language modeling, information extraction, named entity recognition, information retrieval, text classification, word sense disambiguation, automatic question answering and text summarization in real world problems.

CO2: Appraise the complexity of natural language grammatical structures with an emphasis on the English language by demonstrating various parsing techniques.

CO3: Apply 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.

CO4: Propose new solutions to issues of current interest from recently published literature, and do a comparative analysis of different possibilities.

Module 1: (9 T + 5 P Hours)

Natural Language understanding: The study of language, Applications of NLP, Evaluating language understanding systems, Levels of language analysis, Representations and Understanding, Organization of Natural Language Understanding systems, Linguistic background: An outline of English syntax.

Module 2: (10 T + 7 P Hours)

Grammars and parsing: grammars and sentence structure, top-down and bottom-up parsers, transition network grammars, top-down chart parsing. feature systems and augmented grammars: basic feature system for English, morphological analysis and the lexicon, parsing with features, Augmented Transition Networks.

Module 3: (10 T + 7 P Hours)

Grammars for Natural language: Auxiliary Verbs and Verb Phrases, Movement phenomenon in language, Handling questions in context-free grammars, Human preferences in parsing, Encoding uncertainty, Deterministic parser, Ambiguity resolution: Statistical methods, Estimating probabilities.

Module 4: (10 T + 7 P Hours)

Part-of-Speech tagging, Probabilistic Context-free Grammars, Semantics and Logical form, Word senses and Ambiguity, Information Extraction, Named Entity Recognition, Machine Translation, Summarization, Question Answering, Recent trends in NLP.

References

1. J. Allen, *Natural Language Understanding*, 2/e, Pearson Education, 2003.
2. D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 2/e, Pearson Education, 2009.
3. C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*, The MIT Press, Cambridge, Massachusetts.1999.
4. Current Literature.

CS 6172D Artificial Intelligence

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Explain and illustrate various strategies and algorithms for state space search, so as to apply them to solve problems in the domain.

CO2: Critically analyze the various methods for knowledge representation and apply them for problem solving.

CO3: Construct novel algorithms and techniques for problems involving Machine Learning and Natural Language Processing.

Module 1: (10 T + 4 P 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: (10 T + 6 P 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,, Planning,Planning and acting in the real world,Decision Making.

Module 3: (11 T + 8 P Hours)

Learning - Learning From Examples, Knowledge in Learning, Learning probabilistic Models, Reinforcement Learning,Artificial Neural Networks and Deep learning, The Genetic Algorithm-Genetic Programming, Overview of Expert System Technology, Introduction to Natural Language Processing.

Module 4: (8 T + 8 P Hours)

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

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. N. J Nilsson, *Artificial Intelligence a New Synthesis*, Elsevier,1998.
5. P. Winston and B. Horn, *LISP*, 3/e, Addison Wesley, 1989.
6. I. Bratko, *Prolog Programming for Artificial Intelligence, 3/e*, Addison Wesley, 2000.

CS 6173D Image Processing

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Use different modern software/tools for implementing image processing applications.

CO2: Design and build image processing algorithms/applications that can be used in domains like Medical imaging, Satellite Imaging and Video Surveillance.

CO3: Analyzing the performance and complexity of various techniques being used to design domain specific algorithms for improving the performance.

CO4: Propose new solutions to issues of current interest from recently published literature, do a comparative analysis of different possibilities and write research articles based on inferences made from the literature review and analysis.

Module 1: (19 T Hours)

Introduction - Digital image representation - Fundamental steps in image processing - Elements of digital image processing systems - Digital image fundamentals - Elements of visual perception - A simple image model - Sampling and quantization - Basic relationship between pixels - Image geometry - Image transforms - Introduction to Fourier transform – Discrete Fourier transform - Some properties of 2d-fourier transform (DFT)- Other separable image transforms - Hotelling transform

Module 2: (12 T Hours)

Image enhancement - Point processing - Spatial filtering - Frequency domain - Image restoration - Degradation model - Diagonalization of circulant and block circulant matrices - Inverse filtering - Least mean square filter.

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 3: (11 T Hours)

Image compression - Image compression models - Elements of information theory - Error-free compression - Lossy compression - Image compression standards

Module 4: (10 T Hours)

Image reconstruction from projections - Basics of projection - Parallel beam and fan beam projection - Method of generating projections - Fourier slice theorem - Filtered back projection algorithms - Testing back projection algorithms

References

1. R. C., Gonzalez and Woods R.E, *Digital Image Processing*, Addison Wesley, 1999.
2. A. Rosenfeld A and A. C. Kak, *Digital Picture Processing*, Academic Press, 1998.
3. A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, 2002.
4. W. K. Pratt, *Digital Image Processing*, John Wiley, 2002.

CS 6174D Pattern Recognition

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Formulate pattern recognition tasks in relation to fundamental mathematical principles of probability theory, linear algebra and optimization.

CO2: Apply linear classifiers like logistic regression, least squares classifier and perceptron, and non-linear classifiers like Support Vector Machines and Artificial Neural Networks.

CO4: Design and rate new pattern recognition models for problems of current interest, analyse their performances and compare with existing approaches.

Module 1: (15 T Hours)

Introduction- Introduction to statistical, syntactic and descriptive approaches, Features and feature extraction, Learning.Bayes Decision theory- Introduction, continuous case, 2-category classification, Minimum error rate classification, Classifiers, discriminant functions, and decision surfaces. Error probabilities and integrals, normal density**,** Discriminant functions for normal density, Bayes Decision theory Discrete case.

Module 2: (13 T Hours)

Parameter estimation and supervised learning- Maximum likelihood estimation, the Bayes classifier, Learning the mean of a normal density, General bayesian learning.Nonparametric technique- Density estimation, Parzen windows, K-nearest Neighbor estimation, Estimation of posterior probabilities, K-nearest neighbor rule, Nearest- neighbor rule, K-nearest neighbor rule.

Module 3: (12 T Hours)

Multiplayer neural networks- Feed forward operation and classification, Backpropagation algorithm, Error surfaces, Back propagation as feature mapping, Practical techniques for improving backpropagation, Convolutional Neural Networks and Deep Learning**.**

Module 4: (12 T Hours)

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
Unsupervised learning and clustering- Mixture densities and identifiably, Maximum likelihood estimates, Application to normal mixtures, Unsupervised Bayesian learning, Data description and clustering, Hierarchical clustering, Low dimensional representation of multidimensional map

References

1. C. M Bishop, *Pattern Recognition and Machine Learning, Springer,* 2006.
2. K. S. Fu, *Syntactic Pattern Recognition and Applications*, Prentice Hall, 1982.
3. R. O. Duda, P. E. Hart and D.G Stork, *Pattern Classification*, 2/e, John Wiley, 2001.

CS 6181D Bioinformatics

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Define and analyze the structure of biomolecules such as DNA, RNA and Protein.

CO2: Analyze Biological information using tools and databases.

CO3: Apply various data structures to represent Biological data for efficiently solving Biological problems.

CO4: Critically analyze different Bioinformatics algorithms, and develop novel and efficient methods for Biological data analysis.

CO5: Propose novel solutions to potential problems of interest from current literature, and do a comparative analysis of different possibilities.

Module 1: (10 T + 7 P Hours)

Introduction to Molecular biology, Gene structure and Information content, Molecular Biology tools, Algorithms for Sequence Alignment, Sequence Databases and Tools.

Module 2: (10 T + 7 P Hours)

Molecular Phylogenetics, Phylogenetic trees, Algorithms for Phylogenetic Tree construction, Introduction to Perl programming for Bioinformatics.

Module 3: (10 T + 6 P Hours)

Introduction to Protein Structure, Algorithms for Protein Structure Prediction, Gene Expression Analysis, Microarray, Pathway Analysis.

Module 4: (9 T + 6 P Hours)

Pattern Matching Algorithms, Bio-data analysis, Data Mining in Bioinformatics, Algorithms and Data Structures for Efficient Analysis of Biological Data, Emerging Trends in Bioinformatics.

References

1. D. E. Krane and M. L. Raymer, *Fundamental Concepts of Bioinformatics*, Pearson Education, 2003.
2. T. K. Attwood and D. J. Parry-Smith, *Introduction to Bioinformatics*, Pearson Education, 2003.
3. A. M Lesk, *Introduction to Bioinformatics*, Oxford University Press, 2002.
4. J. M. Claverie and C. Notredame, *Bioinformatics – A Beginner's Guide*, Wiley., 2003.
5. N. C Jones and P. A. Pevzner, *An Introduction to Bioinformatics Algorithms*, MIT Press, 2004.
6. Current Literature.

CS 6155D Topics in Data Analytics

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Identify and avoid common pitfalls in big data analytics.

CO2: Choose best analytic approach for a given problem.

CO3: Challenges in Massive Data Analytics.

Module 1: (10 T + 7 P 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: (10 T +7 P 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, 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: (10 T + 6 P 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: (9 T + 6 P Hours)

Formal models for social network data, Ego Networks, Metrics for social relations,, Graph representation of social relations Cliques and subgroups, , 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. John Hopcroft and Ravi Kannan, Foundations of Data Science, e-book, 2013.
4. Kevin P. Murphey, Machine Learning, a Probabilistic Perspective, The MIT Press,Cambridge, Massachusetts, 2012.
5. Hanneman, Robert A., and Mark Riddle. Introduction to Social Network Methods. E-book (web:http://faculty.ucr.edu/~hanneman/), 2005.

CS 6145D Heterogeneous Parallel Programming

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Describe the concepts of modern processors.

CO2: Analyze and compare parallel computing paradigms.

CO3: Design parallel programs using OpenMP and MPI.

CO4: Design heterogeneous parallel programs using CUDA.

Module 1: (10 T + 5 P Hours)

Introduction to Parallel Programming; Needs for parallel computations. Challenges of parallel programming, Computer system classification, Overview of some parallel systems, multiprocessors, multicomputers and Clusters, Modeling and Analysis of Parallel Computations; Efficiency characteristics of parallel computation: speedup, efficiency, estimating the maximum possible parallelization, computational load balancing. The Amdahl's and Gustavson-Barsis's laws.

Module 2: (10 T + 7 P Hours)

Homogenous parallel computing; programming models, Programming with Shared Memory; Overview of the OpenMP standard. Parallel regions. Computational load distributing among the threads, Programming with Message Passing; Overview of the MPI standard. Point-to-point communication operations. Synchronous and asynchronous modes of data transmission. Collective operations. Case studies: matrix computations, solving partial differential equations.

Module 3: (10 T + 7 P Hours)

Heterogeneous parallel computing; Accelerators, GPUs, architecture of GPU GPU Vs CPU, parallel programming model for GPU- 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 and parallel scan. Performance bottlenecks and program optimization techniques

Module 4: (9 T + 7 P Hours)

Accelerator based computing & its advantages, Parallel programming using Xeon phi. Xeon Phi – processor architecture, performance bottlenecks and solution, case studies with Knights landing – a many core Xeon phi with some sample programs. FPGA, basics, parallel programming with FPGA. Problem solving on clusters using Hadoop and MapReduce

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,* 2/e, Morgan Kaufmann, 2010.
3. A. Grama, A. Gupta, G. Karypis and V. Kumar, *Introduction to Parallel Computing*, 2/e, pearson, 2011.
4. M. J. Quinn, *Parallel Programming in C with MPI and OpenMP*, McGraw-Hill, 2010.

CS 6137D Parameterized Algorithms

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Classify problems into tractable and intractable problems.

CO2 Illustrate 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: (10 T + 7 P 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.

Module 2: (10 T + 6 P Hours)

Parameterizations and Parameterized problems- Kernelization - Formal definitions - Some simple kernels, Crown decomposition, Expansion lemma, Bounded Search trees - Vertex Cover, Feedback Vertex Set, Vertex Cover above LP.

Module 3: (10 T + 7 P Hours)

Iterative Compression - Illustration of the technique - Feedback vertex set - Odd Cycle Transversal - Dynamic programming over subsets – Set cover, Tree structured variants of set cover and Steiner Trees. Randomized methods in Parameterized algorithms – Simple randomized algorithm for Feedback Vertex Set, Color coding algorithm for Longest path.

Module 4: (9 T + 6 P Hours)

Trees - narrow grids and dynamic programming - 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.

CS 6138D Parameterized Complexity Theory

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Classify the parameterized problems into fixed-parameter tractable and fixed-parameter intractable.

CO2: Illustrate the hierarchy of hardness of parameterized intractable problems.

CO3: Analyze proofs of standard results in Parameterized complexity theory.

Module 1: (13 T Hours)

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

Module 2: (13 T Hours)

Parameterized complexity theory – The W-hierarchy - Problems complete for W[1] and W[2] - Parameterized reductions and examples

Module 3: (13 T Hours)

Kernelization and Linear Programming Techniques, Tree Decomposition of Graphs - Courcelle's Theorem - Tree width reductions and Graph Minors - Algebraic techniques: Sieves, convolutions and polynomials Matroids - Classes of matroids - Algorithms for matroid problems

Module 4: (13 T Hours)

Lower bounds based on the Exponential-Time Hypothesis - Motivation and basic results - ETH and classical complexity - ETH and fixed-parameter tractable problems - ETH and W[1]-hard problems - Lower bounds for kernelization - Compositionality - Examples

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.
4. M. Cygan et.al., *Parameterized Algorithms*, Springer, June 2015.

CS 6130D Topics in Computational Complexity

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

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 T Hours)

Classical Complexity: Hierarchy theorems, Savitch and Immerman-Szelepcsenyi theorems, NP completeness, PSPACE completeness.

Module 2: (13 T Hours)

Polynomial hierarchy, Toda's theorem, Karp-Lipton theorem, Randomized classes, Sipser Gacs theorem, Adleman's theorem, Interactive proofs, Shamir's theorem.

Module 3: (13 T Hours)

Circuit complexity, NC, P completeness, AC, ACC and TC circuit families and complexity classes, Circuit lower bounds - Switching lemma, classes ACC0 and TC0. Circuit lower bound for PARITY and MAJORITY. Inequivalence of ACC0 and NEXP (no proof).

Module 4: (13 T Hours)

Hardness of approximation: Probabilistically checkable proofs, PCP theorem (no proof), Proofs for weaker versions of PCP theorem, gap reductions.

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.

CS 6191D Mathematical Foundations of Machine Learning

Prerequisites: NIL

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

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: (13 T 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: (13 T 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: (13 T 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: (13 T 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. S. M. 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. S. Boyd 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.

CS 6192D Machine Learning

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

CO1: Explain the fundamental concepts of Statistical Learning Theory required for applying Machine Learning
Techniques.

CO2: Apply different estimation techniques and graphical models.

CO3: Explain the concepts of optimal marginal hyperplanes and SVM and to build classifiers using SVM.

Module 1: (10 T Hours)

Statistical Learning Theory: Learning and Generalization, PAC learning framework, Empirical Risk Minimization, Consistency of Empirical Risk Minimization. Complexity of learning problems - VC Dimension, No Free Lunch Theorem; Model selection and model estimation; Bias-variance trade-off, Regularization - Assessing Learned classifiers - Cross Validation.

Module 2: (9 T + 8 P Hours)

Estimation Techniques: Mixture Densities, EM Algorithm, Approximate Inference, Variational Inference, Variational Mixture of Gaussians, Variational Linear Regression, Sampling Methods, Markov Chain Monte Carlo Sampling, Gibbs Sampling.

Module 3: (10 T + 8 P Hours)

Probabilistic Graphical Models: Directed Graphical Models: Bayesian Networks, Conditional Independence, D-separation – Markov and Hidden Markov Models – Applications of Directed Graphical Models.
Undirected Graphical Models: Markov Random Fields, Conditional independence properties, Factorization Properties, Hammersley Clifford Theorem, Inference and Learning, Application to Image de-noising, Ising Model, Gaussian MRF, Conditional Random Fields : Inference and Learning, Structural SVM.

Module 4: (10 T + 10 P Hours)

Kernel Functions: Positive definite kernels, reproducing kernel Hilbert space, Representer Theorem, Mercer Theorem. Support Vector Machines: Optimal margin hyperplanes, SVM Formulation with Slack Variables, Nonlinear SVM Classifiers, Support Vector Regression, Overview Of SMO and other Algorithms.

References

1. K. P. Murphy, *Machine learning: a Probabilistic Perspective*, MIT press, 2012.
2. C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
3. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, John Wiley & Sons, 2012.

CS 6193D Machine Learning Laboratory

Prerequisites: NIL

L	T	P	C
1	0	6	4

Total hours: (13 T + 78 P)

Course Outcomes:

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 (78 Hours)

1. Bayes Classifier.(9)
2. Parameter Estimation(MLE and Non-parametric Estimation).(6)
3. Linear Data Analysis Methods(LDA,PCA,Regression).(12)
4. Support Vector Machines.(12)
5. KNN Classifier.(6)
6. Decision Trees and Random Forest.(9)
7. Artificial Neural Network and Deep Learning.(24)

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. Y.Liu, *Python Machine Learning by Example*, Packt Publishing Limited, 2017.

CS 6201D Cryptography

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Demonstrate knowledge of historical and modern day ciphers and principles of security behind them.

CO2: Evaluate the security of private key encryption schemes and public key encryption schemes.

CO3: Choose the appropriate encryption scheme for a given context.

Module 1: (18 T + 10 Hours)

Historical Ciphers - Modern day Ciphers. Provable Security and other varying security notions. Perfect Secrecy. One Time Pad. Practical Implementation and limitations.

Private-Key Encryption - Constructing Secure Encryption Schemes - Linear and Differential Cryptanalysis. Practical implementation of block ciphers.

Module 2: (9 T + 6 P Hours)

Cryptographic Techniques for Integrity - Message Authentication Codes - Hash Functions - Attacks. Practical implementation of hash functions.

Module 3: (12 T + 10 P Hours)

Number theory concepts - Primality testing algorithms - Factoring algorithms - Algorithms for computing discrete logarithm. Review of RSA - Attacks on RSA.
Introduction to advanced cryptosystems like 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.
4. A. J. Menezes, P. C. Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press,
1996.

CS 6211D Formal Methods in Secure Computing

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Demonstrate and understanding of different models for representing security protocols' security.

CO2: Analyze the security of protocols designed for providing various security properties.

CO3: Apply various mathematical and logic based conceptual tools for protocol verification and establishment of correctness.

CO4: Analyze the use of Strand Spaces as a formal model and design verifiers based on strand spaces.

Module 1: (10 T + 6 P Hours)

Decidability of security, Access control, take grant model, SPM, Expressive power of models, typed access control models

Module 2: (8 T + 6 P Hours)

Authentication and key establishment, Freshness, general design principles, common attacks, forward secrecy, multi party authentication, Anonymity

Module 3: (10 T +7 P Hours)

Protocol Verification and Correctness, Logic based Models, BAN Logic, Spi calculus

Module 4: (11 T + 7 P Hours)

Strand space based analysis, Applicability to group protocols

References

1. W. H. Ware, C. P. Pfleeger, and S. L. Pfleeger, *Security in Computing*, 3/e, Prentice Hall, 2003.
2. T. Dimitrakos and F. Martinelli, *Formal Aspects In Security And Trust*: Ifip TN Wg1.7 Workshop on
Formal Aspects in Security, Springer, 2005.
3. S. Bosworth and M. E. Kabay, *Computer Security Handbook*, 4/e, 2002.
4. M. Bishop and S. S, Venkatramanayya, *Introduction to Computer Security,* Pearson Education Asia,
2005.
5. M. Abadi and A. D. Gordon, *A Calculus for Cryptographic Protocols — The Spi Calculus*, Research
report SRC 149, 1998.

CS 6212D Network Security

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Differentiate the different security protocols and their application area.

CO2: Infer possible attacks and find the possible countermeasures.

CO3: Examine current work in the area and apply it to solve the problem encounters.

Module 1: (10 T + 7 P Hours)

Review of wired/wireless network protocols, Intrusion detection systems, Malicious software.

Module 2: (10 T + 7 P Hours)

Review of cryptographic algorithms and protocols, Cryptanalysis, Authentication and signature protocols.

Module 3: (10 T + 7 P Hours)

Kerberos, PKI, Real-time communication security, IPSec: AH, ESP, IKE.

Module 4: (9 T + 5 P Hours)

SSL/TLS, E-mail security, PEM and S/MIMIE, PGP, Web security, Network management security, Wireless security.

References

1. C. Kaufman, R. Perlman, and M. Speciner, *Network Security*: *Private Communication in a Public World*, 2/e, Prentice Hall, 2002.
2. J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet* , 3/e, Pearson Education Asia, 2005.
3. W. Stallings, *Cryptography and Network Security Principles and Practice*, 3/e, Pearson Education Asia, 2003.

CS 6213D Foundations of Information Security

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Classify and analyze different cryptographic techniques.

CO2: Identify and categorize authentication protocols.

CO3: Select appropriate network protocols for securing data in motion.

Module 1: (16 T + 12 P Hours)

Introductory Topics: Overview of number theory concepts for cryptography. Historical Ciphers - Monoalphabetic and polyalphabetic ciphers.
Cryptography Topics: Secret vs. Public, Secret Key - DES, Public Key - Knapsack and RSA, Cryptographic hash - SHA1, Discrete Log - Diffie Hellman.

Module 2: (13 T Hours)

Key Management Topics: Digital certificates and PKI
Authentication topics: One way and two way authentication, Centralised Authentication, Needham-Schroeder protocol.

Module 3: (10 T + 14 P Hours)

Network security topics: Network layer security - IPSec, SSL-TLS, PGP. Overview of network security tools.

References

1. B. Menezes, *Network Security and Cryptography*, Cengage Learning India, 2010.
2. H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications*, Springer - Verlag, 2002
3. Whitman and Mattord, *Principles of Information Security*, Cengage Learning, 2006.

CS 6214D Topics in Information Security

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Evaluate the suitability of cryptographic solutions and models in an application context.

CO2: Demonstrate security attacks by exploiting vulnerabilities.

CO3: Defend networks by applying the knowledge on attacks and by deploying detection and prevention
systems.

Module 1: (14 T + 6 P Hours)

Cryptography topics - Symmetric Key - AES, Public Key - ECC, Linear Cryptanalysis in SPN. Attacks on RSA.

System security topics : Access Control - MAC. DAC, RBAC. Security Models as basis for OS security - BLP, Biba, Chinese Wall and Clark Wilson - Reference Monitor.

Module 2: (14 T + 12 P Hours)

Software vulnerabilities - Buffer and stack overflow, Phishing. Malware - Viruses, Worms and Trojans. Topological worms. Propagation models.

Entity Authentication - Password, Challenge Response, Zero Knowledge Protocols. Kerberos. Biometrics for authentication - methods and error types.

Module 3: (11 T + 8 P Hours)

Network Security Topics - DoS, DDoS, ARP spoofing, session hijacking, pharming. Firewalls - placement and configuration. Intrusion Detection Systems - DDoS detection, Malware defense.

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.

CS 6231D Theoretical Aspects of Cryptographic Algorithms

Prerequisites: NIL.

L	T	P	C
4	0	0	4

Total hours: 52

Course Outcomes:

CO1: Discuss the theoretical basis of cryptographic algorithms.

CO2: Analyze the hardness assumptions forming the basis of cryptographic algorithms.

CO3: Explore algebraic aspects of cryptographic foundations for possible improvements and adaptations.

Module 1: (14 T Hours)

Euclid's algorithm for integers and polynomials – two squares theorem, applications to modular inversion, Rational polynomial approximation

Module 2: (14 T Hours)

Quadratic reciprocity and application to primality testing, primality testing algorithms

Module 3: (14 T Hours)

Polynomial and integer factorization, hardness, algorithms, deterministic algorithms.

Module 4: (10 T Hours)

Rings, Finite Fields and Applications to cryptography and coding theory.

References

1. V. Shoup, *A Computational Introduction to Number Theory and Algebra*, Cambridge University Press,
   Second Print edition, 2008.
2. J. Von Zur Gathen, J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2003.
3. Current Literature.

CS 6232D Crypto Complexity

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Demonstrate knowledge of fundamental complexity theory, cryptology, randomized complexity classes and probabilistic analysis.

CO2: Apply knowledge of cryptosystems to security issues.

CO3: Analyze the complexity of algorithms based on cryptographic concepts.

Module 1: (9 T Hours)

Review of Relevant Mathematics, Complexity Theory, Foundations of Cryptology, Hierarchies based on NP.

Module 2: (10 T Hours)

Randomized algorithms and Complexity classes, probabilistic Polynomial time classes, Quantifiers, Graph Isomorphism and lowness.

Module 3: (10 T + 13 P Hours)

RSA Cryptosystem, primality and factoring, Primality Tests, Factoring Methods, Security of RSA.

Module 4: (10 T + 13 P Hours)

Diffie Hellman, ElGamal and other protocols, Arthur Merlin Games and Zero knowledge.

References

1. J. Roth, *Complexity Theory and Cryptology – An introduction to Crypto Complexity*, Springer, 2005.
2. H. Anton, *Elementary Linear Algebra*, John Wiley and Sons, New York, 8/e, 2000.
3. G. Brassard. *A Note on the Complexity of Cryptography*, IEEE Transactions on Information Theory, 25(2):232-233, 1979.

CS 6233D Information Theory and Coding

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Demonstrate understanding of standard source and channel models, their capacities and coding methods.

CO2: Estimate the theoretical capacity of a source/channel coding scheme.

CO3: Analytically compute theoretical efficiency of a given source/channel coding scheme.

Module 1: (9 T + 6 P Hours)

Introduction to probability, information, noiseless coding, noisy coding, cyclic redundancy checks

Module 2: (10 T + 7 P Hours)

Permutation of sets, finite fields, linear codes, bounds for codes Hamming (Sphere-Packing) Bound. Gilbert-Varshamov Bound. Singleton Bound.

Module 3: (10 T + 6 P Hours)

Primitive polynomials, Testing for Primitivity. Periods of LFSR's. Vandermonde Determinants. Check Matrices for Cyclic Codes. RS Codes. Hamming Codes (Again). BCH Codes. Decoding BCH Codes.

Module 4: (10 T + 7 P Hours)

Concatenated codes, curves and codes. Plane Curves. Curves in Higher Dimensions. Geometric Goppa Codes.

References

1. P. Garrett, *The Mathematics of Coding Theory: Information, Compression, Error Correction and Finite Fields*, Pearson Education, 2004.
2. S. Lin and D. J. Costello, *Error Control Coding - Fundamentals and Applications*, Prentice Hall Inc.Englewood Cliffs, 1983.
3. S. Ling, *Coding Theory – A First Course*, Cambridge Press, 2004.

CS 6271D Data Compression

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Define basic techniques of data compression and basics of statistical methods of data compression .

CO2: Explain and apply dictionary methods for string compression, approaches to image, audio and video compression.

CO3: Implement and analyse the performance of various data compression algorithms.

Module 1: (9 T + 6 P Hours)

Introduction to data compression-Basic Techniques- Runlength encoding, RLe Text compression, RLE image compression, Move-to-front coding, Scalar quantization.Statistical Methods- Information theory concepts, variable length codes, prefix codes, Shannon- Fano coding, Huffman coding, Adaptive Huffman coding, Arithmetic coding.

Module 2: (10 T + 7 P Hours)

Dictionary methods- string compression, LZ77 sliding window, MZW, Gif images. Image Compression-Approaches to image compression, intuitive methods, image transform, test images, JPEG, Progressive image compression, Vector quantization,

Module 3: (10 T + 7 P Hours)

Wavelet Methods- Fourier transform, frequency domain, Fourier image compression, CWT and inverse CWT, Haar transform, filter bank, DWT, JPEG 2000. Video compression- analog video, Composite and component video, digital video, video compression, MPEG.

Module 4: (10 T + 6 P Hours)

Audio Compression- Sound, digital audio, human auditory system, MPEG-1 audio layer. Fractal based compression- IFS. Comparison of compression algorithms. Implementation of compression algorithms.

References

1. D. Salomon, *Data compression: The Complete Reference*, 2/e, Springer-Verlag, New York. 2000.
2. S. Welstead, *Fractal and Wavelet Image Compression Techniques*, PHI, New Delhi-1, 1999.
3. K. Sayood, *Introduction to Data compression*, Morgan Kaufmann Publishers, 2003 (reprint).

CS 6283D Computer Laws and Ethics

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Outline the domain of intellectual property in the digital context.

CO2: Differentiate between different kinds of legal contracts in the digital domain.

CO3: Identify those digital interactions which are termed as computer crime and analyze their aspects.

CO4: Survey the different laws and acts in India in the IT domain and review the same in the global context.

Module 1: (10 T Hours)

Intellectual property rights, computer software copyrights, copyright in databases and electronic publishing, law of confidence, patent laws, trademarks, product designs, international law .

Module 2: (10 T Hours)

Computer contracts, liability for defective hardware and software, software contracts, web and hardware contracts, electronic contracts and torts, liabilities.

Module 3 : (10 T + 13 P Hours)

Computer crime, computer fraud, hacking, unauthorized modification of information, piracy, computer pornography and harassment.

Module 4: (9 T + 13 P Hours)

Cyber laws in India, IT Act 2000, Data Subjects' Rights, ethical issues in computer security, case studies.
Data Privacy and Protection Bill 2017

References

1. D. Bainbridge, *Introduction to Computer Law*, 5/e, Pearson Education, 2004.
2. P. Duggal, *Cyber Law: The Indian Perspective*, 2005.
3. C. P. Fleeger and S. L. Fleeger, *Security In Computing*, 3/e, Pearson Education, 2003.

CS 6285D Information Security Management

Prerequisites: NIL

L	T	P	C
3	0	2	4

Total hours: (39 T + 26 P)

Course Outcomes:

CO1: Review information systems in organizations and threats to these information systems.

CO2: Model security measures for organizations and examine key aspects of network security.

CO3: Differentiate between notions of privacy and security and Model mechanisms for attacks and defense.

CO4: Choose the relevant building blocks of information systems for an organization with prime emphasis on security.

Module 1: (9 T + 6 P Hours)

Information as an Asset – creation and maintenance. Information systems in organizations of a global setting – Building blocks and review of current status. Threats to information systems. Information Security Management (ISM) in organizations. Information Asset Management and Risk analysis.

Module 2: (10 T + 7 P Hours)

Managing Physical and Environmental security. Perimeter security. Use of biometrics in this context. Access control models and Role based approaches for organizational hierarchy. Managing Network Security. Firewalls, VPNs and IDS. Digital certificates and CAs. Managing wireless network security.

Module 3: (10 T + 6 P Hours)

Application Security. Business Applications – choice of security architecture for third party software and turnkey software projects. Choosing the building blocks of information systems of the firm with security considerations – OS and databases, email and web servers.

Module 4: (10 T + 7 P Hours)

Disaster recovery approaches business continuity. Security models and frameworks. Security standards and compliance needs in different business domains. IT Act and other legal requirements – relevance to organizations operating in the country.

References

1. N. Godbole, *Information Systems Security: Security Management, Metrics, Frameworks and Best Practices*, John Wiley and Sons Ltd., 2009.
2. H. F. Tipton and M. K. Nozaki, *Information Security Management Handbook*, 4/e, Auerbach, 2000.
3. S. M. Furnell, S. Katsikas, J. Lopez, and A. Patel, *Securing Information and Communication Systems: Principles, Technologies and Applications*, Artech House Inc., 2008.
4. W. Michael and H. Mattord, *Management of Information Security*, Cengage Learning, 2007.

CS 6203D Information Security Laboratory

Prerequisites: NIL

L	T	P	C
1	0	6	4

Total Hours: (13 T + 78 P)

Course Outcomes:

CO1: Apply security procedures for enhancing security.

CO2: Build cryptographic solutions using existing libraries in programming languages.

CO3: Use current-day tools for assessing information security and improving security levels.

Theory (13 Hours)

Computer Networks Review
Number Theory Review
Review of Software Vulnerabilities - Both Applications and Systems
Review of Network Vulnerabilities

Practical (78 Hours)

Number theoretic algorithms implementation and comparison with existing libraries for correctness and performance (18 Hours)
Creation of a network testbed for carrying out experiments (9)
Deploy the necessary tools so as to carry out observations (9)
Plant vulnerabilities and simulate attacks at OS and Applications levels (18)
Network attacks - simulation, analysis using tools, prevention using systems like IDS (24)

References

1. H. Delfs and H. Knebl, *Introduction to Cryptography Principles and Applications*, Springer, 2002.
2. M. Gregg, *Build Your Own Security Lab,* Wiley India, 2008.
3. B. Menezes, *Network Security and Cryptography*, Cengage Learning India, 2010.

CS 6104D Term Paper

Prerequisites: NIL

L	T	P	C
1	0	6	4

Total hours: (13 T + 78 P)

Course Outcomes:

CO1: Survey literature in a chosen area.

CO2: Identify issues and problems pertinent to the and assess their relevance.

CO3: Evaluate the scope and depth of required solutions for identified problems.

CO4: Effectively communicate the results of the study and analysis.

Theory (13 Hours)

Topics not covered within the scope of offered courses, but necessary for the chosen area of research, that require classroom teaching/learning

Practical (78 Hours)

The student is expected to do an extensive literature survey and analysis in an area related to computer science, chosen by her/him, under the supervision of a faculty member from the department. The study should preferably result in design ideas, designs, algorithms, theoretical contributions in the form of theorems and proofs, new methods of proof, new techniques or heuristics with analytical studies, implementations and analysis of results. The student should give two seminars on his work, one in the middle of the semester and the other at the end of the semester, and submit a technical report.

References

Articles from ACM/IEEE Journals/Conference Proceedings and equivalent documents, Standard Textbooks and Web Based Material, approved by the supervisor.