INDIAN STATISTICAL INSTITUTE

Students' Brochure

MASTER OF TECHNOLOGY (M. TECH.)
IN COMPUTER SCIENCE

(Effective from the Academic Year 2020-21)

203 BARRACKPORE TRUNK ROAD
KOLKATA 700108

Eligibility

A candidate would be eligible to appear for the admission test if (s)he satisfies one of the following:

1. Have a four year B-Tech/B.E (or equivalent) degree in any stream.
2. Have a master's degree in any subject and have passed Mathematics at the 10+2 level.

Admission Test

The admission test would have two major components: a written test and an interview.

The written test would consist of two tests which would be conducted on the same day. The first test would be a multiple choice type test on Mathematics at the B.Sc. (pass) level, as it is now. The second test would be a subjective test consisting of two parts: (A) Mathematics at the undergraduate (B.Sc.) level [1] (B) Computer Science at B.E./B. Tech. level, a candidate has to answer either (A) or (B), not both.

The candidates who pass the written test(s) would be called for the interview.

The written test would be waived in the following case:

1. If a student has a valid GATE score above a threshold, the threshold would be decided by the admission committee each year.

A student who applies for admission and satisfies any of the above two would be directly called for the interview.

The final selection would be based on the performance of the candidates in the written tests and interview. The candidates who obtained a waiver in the written tests would be assigned a score in the written test based on his/her GATE score.

Streams

Every student who gets admission in the programme would be classified as a CS-stream student or a non-CS-stream student based on the following criteria:

1. A student who has written the Computer Science part in the written test would be classified as a CS-stream student.
2. A student who has got waiver from the written test but has a valid GATE score (above a threshold as decided by the admission committee) in Computer Science and Information Technology would be classified as a CS-stream student.
3. All other students would be classified as non-CS-stream student.

[1] a sketch of the syllabus for the test is in Appendix-I.

4. The M. Tech.(CS) admission committee should mention the streams of the students in the final selection list.

The curriculum for the two streams would be different. Each student would be required to follow the curriculum specified for his/her stream.

Structure for CS Stream

1. **Compulsory half semester non-credit:** Introduction to Programming.
    - This requirement can be waived if a student passes a programming test which would be designed by the mentor committee [2]. The programming test would be conducted in the first week of the first semester.
2. **Two compulsory courses** from the list of compulsory courses for the CS stream:
    - Design and Analysis of Algorithms.
    - Discrete Mathematics.
3. **Five Formative Courses** from the list of formative courses for the CS stream:

    Pool A
    - Probability and Stochastic Processes
    - Statistics
    - Linear Algebra
    - Algebraic Structures

    Pool B
    - Automata Theory, Languages and Computation
    - Operating Systems
    - Database Management Systems
    - Compiler Construction
    - Computer Networks
    - Principles of Programming Languages
    - Computing Laboratory
    - Computer Architecture

    At least two courses must be from Pool A.
4. **Eight elective courses** from the list of elective courses (the list appears later in this document).
5. **Dissertation** (equivalent to three courses).
6. **Minor Project**:
    - The minor project would be of one semester duration and can be opted for either in the 3rd or the 4th semester.
    - The minor project would have credit equivalent to two courses.
    - A student would be eligible to do a minor project only if (s)he satisfies the following:
        - Has successfully completed at least nine courses in the first two semesters.
        - The aggregate score of the best nine courses taken in the first two semesters is at least 75%.

[2] The structure of the Mentor committee is discussed later in the document

- The topic/problem chosen for a minor project should be significantly different from the topic/problem of the dissertation as judged by the dissertation committee.
- A student who opts for the minor project would decide a topic for the project within three weeks from the start of a semester. The topic has to be approved by the dissertation committee. The dissertation committee will also be responsible for the evaluation of the projects.
- An eligible student may choose not to do the minor project, in that case (s)he has to do two additional courses from the list of formative or elective courses.
- A student who is not eligible for the minor project has to do two additional courses from the list of formative or elective courses.

Structure for non-CS Stream

1. **Compulsory half semester non-credit:** Introduction to Programming.
   - This requirement can be waived if a student passes a programming test which would be designed by the mentor committee. The programming test would be conducted in the first week of the first semester.
2. **Five compulsory courses** from the list of compulsory courses for the non-CS stream:
   - Data Structures.
   - Design and Analysis of Algorithms.
   - Discrete Mathematics.
   - Computing Lab.
   - Operating Systems.
3. **Four Formative Courses** from the list of formative courses for the non-CS stream:

   Pool A
   - Probability and Stochastic Processes
   - Statistics
   - Linear Algebra
   - Algebraic Structures

   Pool B
   - Computer Organization
   - Automata Theory, Languages and Computation
   - Database Management Systems
   - Compiler Construction
   - Computer Networks
   - Principles of Programming Languages
   - Computer Architecture

   The following restrictions would be applicable for choosing the formative courses:
   - **For students with a masters degree in Mathematics/Statistics:** At least three courses must be from Pool B.
   - **For students without a masters degree in Mathematics/Statistics:** At least two courses from Pool A and at least one from Pool B.
4. **Eight elective courses** from the list of elective courses.
5. **Dissertation** (equivalent to three courses).

Electives

Electives are classified into four tracks: Theory, Systems, Cryptology and Security, Data Science. One elective can belong to multiple tracks. Completing a certain number of electives in a specific track will enable a student to obtain a specialisation (details regarding specialisation are noted later). The specialisation will be mentioned in the degree certificate.

The tentative list of electives along with tracks:

1. Advanced Operating Systems
   Track: Systems
2. Advanced Logic and Automata Theory
   Track: Theory
3. Algorithms for Big Data
   Tracks: Theory, Data Science
4. Algorithms for Electronic Design Automation
   Tracks: Systems
5. Coding Theory
   Tracks: Theory, Crypto & Security
6. Computational Algebra and Number Theory
   Tracks: Theory, Crypto & Security
7. Computational Complexity
   Tracks: Theory, Crypto & Security
8. Computational Finance
   Tracks: Theory, Data Science
9. Computational Game Theory
   Tracks: Theory, Data Science
10. Computational Geometry
    Track: Theory
11. Computational Molecular Biology and Bioinformatics
    Tracks: Data Science
12. Computational Topology
    Track: Theory, Data Science
13. Computer Graphics
    Track: Systems, Data Science
14. Computer Vision
    Tracks: Data Science, Systems
15. Computing Systems Security I
    Tracks: Systems, Crypto & Security
16. Computing Systems Security II
    Tracks: Systems, Crypto & Security

17. Cryptology I
    Tracks:Theory, Crypto & Security
18. Cryptology II
    Tracks: Theory, Crypto & Security
19. Cyber-Physical Systems
    Tracks: Systems
20. Digital Signal Processing
    Track: Data Science, Systems
21. Discrete and Combinatorial Geometry
    Track: Theory
22. Distributed Computing
    Track: Theory, Systems
23. Fault Tolerance and Testing
    Track: Systems
24. Graph Algorithms
    Track: Theory
25. Image Processing I
    Tracks: Data Science, Systems
26. Image Processing II
    Tracks: Data Science, Systems
27. Information Retrieval
    Track: Data Science
28. Information Theory
    Tracks: Theory, Data Science, Crypto & Security
29. Learning Theory
    Tracks: Theory, Data Science, Crypto & Security
30. Logic for Computer Science
    Track: Theory
31. Machine Learning I
    Track: Theory, Data Science
32. Machine Learnig II
    Track: Theory, Data Science
33. Mobile Computing
    Track: Systems
34. Natural Language Processing
    Tracks: Data Science
35. Neural Networks
    Tracks: Data Science, Systems

36. Optimization Techniques
Tracks: Theory, Data Science, Systems

37. Quantum Information Processing and Quantum Computation
Tracks: Theory, Crypto & Security

38. Randomized and Approximation Algorithms
Tracks: Theory, Crypto & Security

39. Specification and Verification of Programs
Tracks: Systems

40. Topics in Privacy
Tracks: Crypto & Security

41. Statistical Computing
Tracks: Data Science

Other Rules and Regulations

1. **Duration of the course**. Expected time for completion of the course is two years. A student may take up to three years for completion. However, after completion of the second year a student will not be eligible for stipends, contingency grants and hostel facilities. Special circumstances in this regard would be examined by the Dean in consultation with the teacher's committee/mentor committee in a case to case basis.

2. **Waiver for class attendance.** For a formative course, a CS-stream student can bypass regular classes and claim credit by completing the assignments and passing the examination(s) directly. This may give the student the flexibility to sit for an elective during that time and thus complete the course requirements earlier. This option is not available for a student in the non-CS stream.

   - If a student opts for this, (s)he would require to seek a permission from the mentor committee.
   - The concerned teacher of the course and Dean's office needs to be informed before the mid-semester week.
   - The usual attendance requirement for the student in such cases would be completely waived for the specific course.
   - Under this option the student has to obtain at least 60% to pass.
   - There would be no attendance waiver for laboratory courses.

3. **Registering for a Course:** Within four weeks of the start of a semester a student will have to inform the Dean's office the credit and non-credit courses that (s)he is taking.

4. **Dissertation.** With permission from the mentor committee, a student may carry out his/her dissertation outside the institute. However, the primary supervisor needs to be an ISI faculty. In this case, a joint supervision may be permitted and the student may be allowed to spend considerable time outside the institute, provided his/her course requirements are fulfilled.

5. **Internship/Industrial Training:** There would be a mandatory 12 weeks gap between the first and the second year in the academic calender. The students can pursue internship/industrial training outside the institute during this period. However, internship is not mandatory.

6. **Specialisation**

   - Among the eight electives, if a student successfully completes at least five electives from a specific track and does his/her dissertation in a topic which falls under that track, (s)he graduates with a specialisation. The classification of a dissertation into a track would be done by the dissertation committee during or before the mid-term evaluation.
   - A student would be eligible to obtain a **double specialisation** if (s)he fulfils the following:
     - Successfully completes at least 10 electives with at least five in the two separate tracks in which (s)he wishes to obtain the specialisations. One elective course cannot be counted for two different specialisations.
     - Successfully completes a minor project.
     - The minor project and the dissertation are in two different tracks for which (s)he wishes to obtain the specialisation.

7. **Final Grade**. Upon successful completion of all requirements for the degree, a student would get a final percentage which will be computed following the rules below:

   - For a student who does not opt for a specialisation, the total of the scores in the best seventeen courses and the dissertation would be the final grade. If the student has done a minor project then (s)he can use the grade in the minor project in lieu of two courses. The total marks for a student in this case is 2000.
   - For a student who opts for a single specialisation, the final grade would be the total of the seventeen courses as chosen by the student and the dissertation. If the student has done a minor project, then (s)he can use the the grade in the minor project in lieu of two courses. The seventeen courses should include at least five electives from the track in which the student desires to obtain the specialisation. The total marks for a student in this case is 2000.
   - For a student who opts for a double specialisation, the final grade would be the total of the seventeen courses as chosen by the student, the minor project and the dissertation. The seventeen courses should include at least ten electives with at least five in each track in which the student opts for the specialisations. The total marks for a student in this case is 2200.
   - In all cases, the grades of all the courses successfully completed by a student would be reflected in the final mark sheet.

8. **Stipend.** On admission, each student would receive the institute specified stipend in the first semester. In subsequent semesters the following rules would be applicable:

   - To get a full stipend in the second semester, a student must have successfully completed at least four courses in the first semester with an average score of 60% in the best four courses.
   - To get a full stipend in the third semester, a student must have successfully completed at least nine courses in the first two semesters with an average score of 60% in the best nine courses.

- To get a full stipend in the fourth semester, a student must have successfully completed at least fourteen courses (or twelve courses and minor project) in the three semesters with an average score of 60% in the best fourteen courses. Additionally, (s)he must pass the mid-term evaluation for the dissertation.

A student may be eligible to get a half stipend if (s)he completes the courses as per the schedule above but gets an average score of less than 60% (but more than 45%). A student will not be eligible to get stipend if s(he) gets an average score of less than 45%. The current attendance rule as applicable for stipend will be applied.

9. **Structure and Function of the Committees.** A group of students entering the M. Tech. (CS) programme in a particular year would be treated as a batch. The **Teachers Committee** for a batch in a particular semester would consist of all the teachers of the courses (including regular courses, Lab courses, Projects, and Dissertations) opted for by the students of that batch in that semester.

A **Mentor Committee** (suggested size 3-4) should be formed for every batch which will last till all the students leave the programme. The committee should be announced at the time of admission of the students. Students should approach the Mentor Committee if they have any problem. The committee will perform the following tasks:

- Advise the students with their choice of subjects and specialisations.
- Advise them with any other problem.

A **Project and Dissertation Committee** may be formed for a batch after the completion of the first semester. The committee will perform the following tasks:

- Pool a list of Projects and Dissertations from the faculty members and circulate the list among the students. However, the students will be free to choose a topic outside the list, provided some faculty member agrees to supervise such a project.
- Help the students in finding a Project/Dissertation, if they require such help.
- Ascertain for each student that the problems of Minor project and Dissertation are different. An approval of the committee would be mandatory before a student is assigned a Minor Project.
- Ascertain that the topic of Dissertation of a student is in the area of specialisation (s)he is opting for.
- Ascertain that the topic of Minor Project of a student is in the area of second specialisation (s)he is opting for.
- Conduct evaluation of the projects and dissertation at appropriate times.

Appendix I: Sketch of syllabus for the entrance examination of non-CS stream

A sksetch of the syllabus for the subjective test in mathematics at the undergraduate level for students seeking admission to the non-CS stream of the M.Tech.(CS) course is suggested below:

Analytical Reasoning

Algebra Arithmetic, geometric and harmonic progression. Continued fractions. Elementary combinatorics: Permutations and combinations, Binomial theorem. Theory of equations. Polynomials of a single variable. Inequalities. Complex numbers and De Moivre's theorem. Elementary set theory. Functions and relations. Elementary number theory: Divisibility, Congruences, Primality. Algebra of matrices. Determinant, rank and inverse of a matrix. Solutions of linear equations. Eigenvalues and eigenvectors of matrices. Groups and their properties. Subgroups, Normal subgroups, abelian groups. Boolean algebra.

Coordinate geometry Straight lines, circles, parabolas, ellipses and hyperbolas.

Calculus Sequences and series: Power series, Taylor and Maclaurin series. Limits and continuity of functions of one variable. Differentiation and integration of functions of one variable with applications. Definite integrals. Maxima and minima. Functions of several variables: limits, continuity, differentiability. Double integrals and their applications. Ordinary linear differential equations. Vector calculus.

Elementary discrete probability theory Combinatorial probability, Conditional probability, Bayes theorem. Binomial and Poisson distributions.

Graph Theory Graphs, different representations, subgraphs, connectivity, Trees and their properties.

Appendix II: Typical Semester-wise layout of the compulsory and formative courses

For the smooth running of the proposed M. Tech.(CS) syllabus, all compulsory and formative courses must be offered at least once in every academic year. The distribution of those subjects according to odd and even semesters is as follows:

Odd semester

The following subjects need to be offered in the odd semester (the first semester at the beginning of an academic year):

Subject	Course Type	
	CS-stream	non-CS stream
Introduction to Programming (half semester)	Compulsory	Compulsory
Data Structures	–	Formative
Discrete Mathematics	Compulsory	Compulsory
Computing Laboratory	Formative	Compulsory
Probability and Stochastic Processes	Formative	Formative
Algebraic Structures	Formative	Formative
Linear Algebra	Formative	Formative
Computer Organization	–	Formative
Design and Analysis of Algorithms	Compulsory	Compulsory

Even semester

The following subjects need to be offered in the even semester (the semester at the end of an academic year):

Subject	Course Type	
	CS-stream	non-CS stream
Design and Analysis of Algorithms	Compulsory	Compulsory
Statistics	Formative	Formative
Automata Theory, Languages and Computation	Formative	Formative
Operating Systems	Formative	Compulsory
Database Management Systems	Formative	Formative
Principles of Programming Languages	Formative	Formative
Compiler Construction	Formative	Formative
Computer Architecture	Formative	Formative
Computer Networks	Formative	Formative

Appendix III: Role of the Dean's office

The programme will require some responsibilities to be carried out by the Dean's office within a time frame.

Stream: The Dean's office needs to ensure that the final selection list of the M. Tech.(CS) students mentions the stream of each student.

Mentor committee: The Dean's office needs to form a mentor committee, comprising 3-4 faculty members, to oversee all the students taking admission to the M. Tech.(CS) programmeme in a particular year. The mentor committee of a particular year will be in existence until all students taking admission in that particular year is in the institute's roll. So, the duration of the mentor committee can be more than two years.

Programming test: The newly designed M. Tech.(CS) syllabus has a compulsory half-semester non-credit programming course. A student can seek a waiver from this course by passing a programming examination, to be designed by the mentor committee. This test has to be conducted in the first week of the first semester of the course and its result has to be published within a week.

Subject choice: The Dean's office will maintain the choices of the students regarding the formative and elective courses. The Dean's office can seek the help of the mentor committee.

Dissertation committee: A dissertation committee for a particular batch is to be formed at the end of the first semester. The responsibilities of the dissertation committee regarding dissertation and minor project are mentioned in the report.

Detailed Syllabus

The syllabus of the courses are given below.

- Against each course, five items are mentioned – (a) topics, (b) prerequisites, (c) lecture hours, (d) marks distribution and (e) references.
- The *prerequisites* are meant to serve as a guide for the students to choose a course. If necessary, a student should consult the *mentor committee* or the concerned teacher. However, the responsibility of the choice lies with the student.
- The courses are classified into two heads —
  - Compulsory and Formative Courses
  - Elective Courses

  The *elective courses* are classified into *four tracks*:
  - Theory
  - Data Science
  - Systems
  - Cryptology and Security.
- One elective can belong to multiple tracks. Completing a certain number of electives in a specific track will enable a student to obtain a *specialisation*.
- The list of electives along with tracks:
  1. Advanced Operating Systems
     Track: Systems
  2. Advanced Logic and Automata Theory
     Track: Theory
  3. Algorithms for Big Data
     Tracks: Theory, Data Science
  4. Algorithms for Electronic Design Automation
     Tracks: Systems
  5. Coding Theory
     Tracks: Theory, Cryptology & Security
  6. Computational Algebra and Number Theory
     Tracks: Theory, Cryptology & Security
  7. Computational Complexity
     Tracks: Theory, Cryptology & Security

8. Computational Finance
   Tracks: Theory, Data Science
9. Computational Game Theory
   Tracks: Theory, Data Science
10. Computational Geometry
    Track: Theory
11. Computational Molecular Biology and Bioinformatics
    Tracks: Data Science
12. Computational Topology
    Track: Theory, Data Science
13. Computer Graphics
    Track: Systems, Data Science
14. Computer Vision
    Tracks: Data Science, Systems
15. Computing Systems Security I
    Tracks: Systems, Cryptology & Security
16. Computing Systems Security II
    Tracks: Systems, Cryptology & Security
17. Cryptology I
    Tracks:Theory, Cryptology & Security
18. Cryptology II
    Tracks: Theory, Cryptology & Security
19. Cyber-Physical Systems
    Tracks: Systems
20. Digital Signal Processing
    Track: Data Science, Systems
21. Discrete and Combinatorial Geometry
    Track: Theory
22. Distributed Computing
    Track: Theory, Systems
23. Fault Tolerance and Testing
    Track: Systems
24. Graph Algorithms
    Track: Theory
25. Image Processing I
    Tracks: Data Science, Systems
26. Image Processing II
    Tracks: Data Science, Systems
27. Information Retrieval
    Track: Data Science
28. Information Theory
    Tracks: Theory, Data Science, Cryptology & Security
29. Learning Theory
    Tracks: Theory, Data Science, Cryptology & Security

30. Logic for Computer Science
    Track: Theory
31. Machine Learning I
    Track: Theory, Data Science
32. Machine Learnig II
    Track: Theory, Data Science
33. Mobile Computing
    Track: Systems
34. Natural Language Processing
    Tracks: Data Science
35. Neural Networks
    Tracks: Data Science, Systems
36. Optimization Techniques
    Tracks: Theory, Data Science, Systems
37. Quantum Information Processing and Quantum Computation
    Tracks: Theory, Cryptology & Security
38. Randomized and Approximation Algorithms
    Tracks: Theory, Cryptology & Security
39. Specification and Verification of Programs
    Tracks: Systems
40. Topics in Privacy
    Tracks: Cryptology & Security
41. Statistical Computing
    Tracks: Data Science

- Some of the elective courses have lists of topics for the teacher to choose from. It is expected that a significant portion from such a list would be covered

Compulsory and Formative Courses

Automata Theory, Languages and Computation

(a) **Topics:** *Automata and Languages:* Finite automata, regular languages, regular expressions, deterministic and non-deterministic finite automata, minimization of finite automata, closure properties, Kleene's Theorem, pumping lemma and its application, Myhill-Nerode theorem and its uses; Context-free grammars, context-free languages, Chomsky normal form, closure properties, pumping lemma for context-free languages, push down automata.

Computability: Turing machines and variants; Computable functions, primitive and recursive functions, universality, halting problem, recursive and recursively enumerable sets, parameter theorem, diagonalisation, reducibility, Rices Theorem and its applications. Equivalence of different models of computation and Church-Turing thesis.

Introduction to Complexity: Discussions on time and space complexities; P and NP, NP-completeness, Cook's Theorem, other NP-Complete problems; PSPACE; polynomial hierarchy.

(b) **Prerequisites:** Discrete Mathematics

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. N. J. Cutland, Computability: An Introduction to Recursive Function Theory, Cambridge University Press, London, 1980.
2. M. D. Davis, R. Sigal and E. J. Weyuker, Complexity, Computability and Languages, Academic Press, New York, 1994.
3. J. E. Hopcroft and J. D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, California, 1979.
4. J. E. Hopcroft, J. D. Ullman and R. Motwani, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, California, 2001.
5. H. R. Lewis and C. H. Papadimitriou, Elements of The Theory of Computation, Prentice Hall, Englewood Cliffs, 1981.
6. M. Sipser, Introduction to The Theory of Computation, PWS Pub. Co., New York, 1999.
7. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to The Theory of NP- Completeness, Freeman, New York, 1979.

Compiler Construction

(a) **Topics:** *Introduction:* compilers vs. interpreters, phases and passes, bootstrapping.

Lexical analysis: regular expressions and their application to lexical analysis, implementation of lexical analysers, lexical-analyser generators, use of a lexical analyser generator tool (e.g., lex, flex, or similar), symbol tables.

Parsing: formal grammars and their application to syntax analysis, ambiguity, recursive descent and predictive parsers, LR parsers, error detection and recovery.

Syntax directed translation: synthesised and inherited attributes, S-attributed and L-attributed definitions, augmented LL(1) and LR parsers, use of a parser generator tool (e.g., yacc, bison, or similar).

Type checking: represention of types, type checking.

Intermediate code generation: 3-address code, intermediate code generation for standard constructs (assignment statements, single and multi-dimensional array variables, flow control, function calls, variable declarations).

Memory management and runtime support: activation stack, stack frames, calling and return sequences, access to non-local storage.

Code generation: Register assignment and allocation problems, instruction selection, simple code-generation from intermediate code.

Code optimisation: peephole optimisation, syntax-driven and iterative data flow analysis, common sub-expression elimination, constant folding, copy propagation, dead code elimination, loop optimisation (code motion, induction variables).

Misc. topics (depending on time available): basic concepts of compiling object-oriented and functional languages; just in time compiling; interpreting byte code; garbage collection.

(b) **Prerequisites:** Computer Organisation; Automata Theory, Languages and Computation.

(c) **Hours:** Three lectures and one lab-session per week.

(d) **Marks Distribution:** Theory 60%, Assignments 10%, Project 30%.

(e) **References:**

1. A.V. Aho, R. Sethi and J. Ullman: Compilers: Principles, Techniques and Tools, Addison-Wesley, 1986.
2. A.V. Aho, M.S. Lam, R. Sethi and J. Ullman: Compilers: Principles, Techniques and Tools, 2nd ed., Pearson, 2007.
3. A.W. Appel, M. Ginsburg: Modern Compiler Implementation in C, Cambridge University Press, 1998.
4. A.I. Holub: Compiler Design in C, Prentice Hall, 1990. `https://holub.com/compiler/`

Computer Architecture

(a) **Topics:** Introduction and Basics, Design for Performance, Fundamental Concepts and ISA, ISA Trade-offs, Case Study.

Introduction to Microarchitecture Design, Single Cycle Microarchitecture, Microprogrammed Microarchitecture, Case Study.

Pipelining, Data and Control Dependence Handling, Data and Control Dependence Handling, Branch Prediction, Branch Handling and Branch Prediction II, Precise Exceptions, State Maintenance and State Recovery, Case Study.

Out-of-order execution, Out-of-order execution and Data Flow.

SIMD Processing (Vector and Array Processors), GPUs, VLIW, DAE, Case Study: Nvdia GPUs, Cray-I.

Memory Hierarchy and Caches, Advanced Caches, Virtual Memory, DRAM, Memory Controllers, Memory Management, Memory Latency Tolerance, Prefetching and Runahead execution, Emerging Memory Technologies, Case Study.

Multiprocessors, Memory Consistency and Cache Coherence.

Interconnection Networks.

(b) **Prerequisites:** Computer Organization

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 50%, Lab + Projects 50% [Should have a mandatory laboratory component]

(e) **References:**

1. John L. Hennessy and David A. Patterson: Computer Architecture: A Quantitative Approach, Morgan Kaufmann, 6th Edition
2. John P. Shen and Mikko H. Lipasti: Modern Processor Design: Fundamentals of Superscalar Processors, Waveland Pr Inc, 2013
3. David Culler and Anoop Gupta: Parallel Computer Architecture: a Hardware/Software Approach, Morgan Kaufmann

Computer Networks

(a) **Topics:** *Introduction:* Use of computer networks, Network hardware and software, Classifications of computer networks, Layered network structures, Reference models and their comparison.

Data transmission fundamentals: Analog and digital transmissions, Channel characteristics, Various transmission media, Different transmission impairments, Different modulation techniques.

Communication networks: Introduction to LANs, MANs, and WANs; Switching techniques: Circuit-switching and Packet-switching; Topological design of a network, LAN topologies, Ethernet, Performance of Ethernet, Repeaters and bridges, Asynchronous Transfer Mode.

Data link layer: Services and design issues, Framing techniques, Error detection and correction, Flow control: Stop-and-wait and Sliding window; MAC Protocols: ALOHA, CSMA, CSMA/CD, Collision free protocols, Limited contention protocol; Wireless LAN protocols: MACA, CSMA/CA;

Network Layer: Design issues, Organization of the subnet, Routing, Congestion control, IP protocol, IP addressing.

Transport Layer: Design issues, Transport service, elements of transport protocol, Connection establishment and release, TCP, UDP, TCP congestion control, QoS.

Application Layer: Email, DNS, WWW.

Labs: Interprocess communications and socket programming: Implementation and realization of simple echo client-server over TCP and UDP, proxy web server, FTP, TELNET, Chat programs, DNS and HTTP. Implementation of client-server applications using remote procedure call. Create sockets for handling multiple connections and concurrent server. Simulating PING and TRACEROUTE commands

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 70%, Labs 30% [Should have mandatory lab component]

(e) **References:**

1. Larry L. Peterson and Bruce S. Davie: Computer Networks: A Systems Approach, Morgan Kaufmann Publishers.
2. Andrew S. Tanenbaum: Computer Networks, Prentice Hall.
3. William Stallings: Data and Computer Communcations, Prentice Hall.
4. Bertsekas and Gallagher: Data Networks, Prentice Hall.
5. Behrouz A. Forouza: Data Communications and Networking, Mc Graw Hill Higher Education

Computer Organization

(a) **Topics:** *Binary Systems:* Information representation, number systems binary, octal and hexadecimal numbers; number base conversion; complements, binary codes.

Boolean algebra: Postulates and fundamental theorems, Representation of Boolean functions using Karnaughs map, truth tables, duality and complementation, canonical forms, fundamental Boolean operations - AND, OR, NAND, NOR, XOR, Universal Gates.

Minimization of Boolean functions: Using fundamental theorems, Karnaugh Maps, Mcclusky method.

Combinational Logic: Adders, Subtractors, code conversion, comparator, decoder, multiplexer, ROM, PLA.

Sequential Logic: Finite state models for sequential machines, pulse, level and clocked operations; flip-flops, registers, shift register, ripple counters, synchronous counters; state diagrams, characteristics and excitation tables of various memory elements, state minimization for synchronous and asynchronous sequential circuits.

ALU Design: Addition of numbers – carry look-ahead and pre-carry vector approaches, carry propagation-free addition. Multiplication - using ripple carry adders, carry save adders, redundant number system arithmetic, Booths algorithm. Division - restoring and non-restoring techniques, using repeated multiplication. Floating-point arithmetic IEEE 754-1985 format, multiplication and addition algorithms. ALU design, instruction formats, addressing modes.

Processor Design: ISA and Microarchitecture design, hardware control unit design, hardware programming language, microprogramming, horizontal, vertical and encoded-control microprogramming, microprogrammed control unit design, pipelining.

Memory Organization: Random and serial access memories, static and dynamic RAMs, ROM, Associative memory.

I/O Organization: Different techniques of addressing I/O devices, data transfer techniques, programmed interrupt, DMA, I/O channels, channel programming, data transfer over synchronous and asynchronous buses, bus control.

(b) **Prerequisites:** Nil

(c) **Hours:** Three lectures and one laboratory per week

(d) **Marks Distribution:** Theory 50%, Lab 50% [Mandatory laboratory component using hardware description language(s)]

(e) **References:**

1. Z. Kohavi, Switching and Finite Automata Theory, 2nd ed., McGraw Hill, New York, 1978.
2. E. J. Mcclusky, Logic Design Principles, Prentice Hall International, New York, 1986.
3. N. N. Biswas, Logic Design Theory, Prentice-Hall of India, New Delhi, 1994.
4. A. D. Freedman and P. R. Menon, Theory and Design of Switching Circuits, Computer Science Press, California, 1975.
5. T. C. Bartee, Digital Computer Fundamentals, 6th ed., McGraw Hill, New York, 1985.
6. J. P. Hayes, Computer Architecture and Organization, 2nd ed., McGraw Hill, New York, 1988
7. P. Pal Choudhury, Computer Organization and Design, Prentice Hall of India, New Delhi, 1994.
8. M. M. Mano, Computer System Architecture, 3rd ed., Prentice Hall of India, New Delhi, 1993.
9. Y. Chu, Computer Organization and Micro-Programming, Prentice Hall, Englewood Cliffs, 1972.
10. W. Stallings, Computer Organization and Architecture: Principles of Structure and Function, 2nd ed., Macmillan, New York, 1990.

Computing Laboratory

(a) **Topics:** This laboratory course has to be run in coordination with the Data Structures course. The assignments are to be designed based on the coverage in the Data Structures course. The initial programming language can be `C` or can be decided by the instructor based on the background of the students. The laboratory sessions should include but need not be limited to:

Programming techniques: Problem solving techniques like divide-and-conquer, dynamic programming, recursion.

Data Structures:
Arrays: Implementation of array operations.
Stacks and Queues, Circular Queues: Adding, deleting elements.
Merging Problem: Evaluation of expressions, operations on multiple stacks and queues.
Linked lists: Implementation of linked lists, inserting, deleting, and inverting a linked list. Implementation of stacks and queues using linked lists. Polynomial addition and multiplication. Sparse Matrix multiplication and addition.
Trees: Recursive and non-recursive traversal of trees; implementation of balanced search trees, e.g. AVL tree, Red-Black tree, etc., Heap.
Hashing: Hash table implementation, searching, inserting and deleting
Searching and sorting techniques

Object oriented programming: Introduction to object oriented programming, classes and methods, polymorphism, inheritance.

Introduction to other programming languages: Python, R, etc.

In addition, the following concepts need to be covered during the course of the lab session: (i) testing the program, developing test-plan, developing tests, concept of regression; (ii) version management, concept of CVS/SVN; (iii) concept of debugging; (iv) concept of writing automation scripts, using bash/tcsh; (v) concept of makefiles;

(b) **Prerequisite:** Data Structures (can be taken concurrently)

(c) **Hours:** Six hours per week

(d) **Marks Distribution:** Marks to be distributed among assignments (50%) and two/three laboratory tests (50%);

(e) **References:**

1. T. A. Standish: Data Structures, Algorithms and Software Principles in C, Addison-Wesley, Reading, Mass., 1995.
2. L. Nyhoff, C++ An Introduction to Data Structures, Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tenenbaum, Y. Langsam and M. J. Augenstein: Data Structures Using C, Pearson, 1998.
4. D. E. Knuth: The Art of Computer Programming. Vol. 1, 3rd. ed. Narosa/Addison-Wesley, New Delhi/London, 1997.
5. T. A. Standish: Data Structure Techniques, Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni: Fundamentals of Data Structures, Galgotia Booksource, New Delhi, 1977.
7. R. L. Kruse: Data Structures and Program Design in C, Prentice Hall of India, New Delhi, 1996.
8. A. Aho, J. Hopcroft, and J. Ullman: Data Structures and Algorithms, Addison-Wesley, Reading, Mass., 1983.
9. B. Salzberg: File Structures: An Analytical Approach, Prentice Hall, New Jersey, 1988.
10. T. Harbron: File System Structure and Algorithms, Prentice Hall, New Jersey, 1987.
11. P. E. Livadas: File Structure: Theory and Practice, Prentice Hall, New Jersey, 1990.
12. T. Cormen, C. Leiserson, R. Rivest and C. Stein: Introduction to Algorithms, PHI Learning Pvt. Ltd., New Delhi, 2009.
13. S. Sahni: Data Structure, Algorithms and Applications in JAVA, Universities Press (India) Pvt. Ltd., New York, 2005.
14. D. Wood: Data Structure, Algorithms and Performance, Addison-Wesley, Reading, Mass., 1993.
15. M. T. Goodrich, R. Tamassia and David Mount: Data Structures and Algorithms in C++, 2nd ed., Wiley, 2011.
16. B. W. Kernighan and D. M. Ritchie: The C Programming Language, Prentice Hall of India, 1994.
17. B. Gottfried: Programming in C, Schaum Outline Series, New Delhi, 1996.
18. B. W. Kernighan and R. Pike: The Unix Programming Environment, Prentice Hall of India, 1996.
19. R. G. Dromey: How to Solve it by Computers, Pearson, 2008.
20. R. Sedgewick and K. Wayne: Algorithms, 4^{th} edition, Addison-Wesley.
21. M.A. Weiss: Data Structures and Algorithm Analysis in C++, 4^{th} edition, Pearson.

Data Structures

(a) **Topics:** *Introduction:* Asymptotic notations; Idea of data structure design (in terms of static and dynamic data), and the basic operations needed; Initial ideas of algorithms and its resource usage in terms of space and time complexity; ideas of worst case, average case and amortized case analysis; Initial ideas of memory model, RAM model, memory hierarchy.

Construction and manipulation of basic data structures: Idea of Abstract Data Types and its concrete implementation; Basic data structures List, Array, Stack, Queue, Dequeue, Linked lists; binary tree and traversal algorithms, threaded tree, m-ary tree, its construction and traversals; Priority Queue and heap.

Data Structures for searching: Binary search trees, Height-Balanced binary search trees; Weight-Balanced binary search tree; Red-Black Tree; Binomial Heap; Splay Tree; Skip list; Trie; Hashing separate chaining, linear probing, quadratic probing.

Advanced data structures: Suffix array and suffix tree; Union Find for set operations; Data structures used in geometric searching Kd tree, Range tree; Quadtree; Data structures used for graphs.

External memory data structures: B tree; B+ tree.

Programming practices: Apart from theoretical analysis of data structures, implementations should be done as assignments. The "Programming and Data Structures Laboratory" course can act as a supplement to this course. Enough care to be taken so that duplication in assignments can be avoided.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. T. A. Standish: Data Structures, Algorithms and Software Principles in C, Addison-Wesley, Reading, Mass., 1995.
2. L. Nyhoff, C++ An Introduction to Data Structures, Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tenenbaum, Y. Langsam and M. J. Augenstein: Data Structures Using C, Pearson, 1998.
4. D. E. Knuth: The Art of Computer Programming. Vol. 1, 3rd. ed. Narosa/Addison-Wesley, New Delhi/London, 1997.
5. T. A. Standish: Data Structure Techniques, Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni: Fundamentals of Data Structures, Galgotia Booksource, New Delhi, 1977.
7. R. L. Kruse: Data Structures and Program Design in C, Prentice Hall of India, New Delhi, 1996.
8. A. Aho, J. Hopcroft, and J. Ullman: Data Structures and Algorithms, Addison-Wesley, Reading, Mass., 1983.
9. B. Salzberg: File Structures: An Analytical Approach, Prentice Hall, New Jersey, 1988.
10. T. Harbron: File System Structure and Algorithms, Prentice Hall, New Jersey, 1987.
11. P. E. Livadas: File Structure: Theory and Practice, Prentice Hall, New Jersey, 1990.

12. T. Coreman, C. Leiserson, R. Rivest and C. Stein: Introduction to Algorithms, PHI Learning Pvt. Ltd., New Delhi, 2009.
13. S. Sahani: Data Structure, Algorithms and Applications in JAVA, Universities Press (India) Pvt. Ltd., New York, 2005.
14. D. Wood: Data Structure, Algorithms and Performance, Addison-Wesley, Reading, Mass., 1993.
15. M. T. Goodrich, R. Tamassia and David Mount: Data Structures and Algorithms in C++, 2nd ed., Wiley, 2011.
16. R. Sedgewick and K. Wayne: Algorithms, 4^{th} edition, Addison-Wesley.
17. M.A. Weiss: Data Structures and Algorithm Analysis in C++, 4^{th} edition, Pearson.

Data Base Management Systems

(a) **Topics:**

Introduction: Purpose of database systems, data abstraction and modelling, instances and schemes, database manager, database users and their interactions, data definition and manipulation language, data dictionary, overall system structure.

Relational model: Structure of a relational database, operation on relations, relational algebra, tuple and domain relational calculus, salient feature of a query language.

SQL: domain types, construction, alteration and deletion of tables, query structure and examples, natural joins and other set operations, aggregations, nested sub-queries, inserting, modifying and deleting data, advanced joins, views, transactions, integrity constraints, cascading actions, authorization and roles. Hands on and practical assignments.

Entity - relationship model: Entities and entity sets, relationships and relationship sets, mapping constraints, E - R diagram, primary keys, strong and weak entities, reducing E - R diagrams to tables.

Introduction to hierarchical and network model: Data description and tree structure diagram for hierarchical model, retrieval and update facilities, limitations; Database task group (DDBTG) model, record and set constructs retrieval and update facilities, limitations.

Databases in application development: cursors, database APIs, JDBC and ODBC, JDBC drivers, Connections, Statements, ResultSets, Exceptions and Warnings. Practical case studies.

Normalization: Anomalies in RDBMS, importance of normalization, functional, multi-valued and join dependencies, closures of functional dependencies and attribute sets, 1NF, 2NF, 3NF and BCNF; (Optionally) 4NF and 5NF; Discussion on tradeoff between performance and normalization. Database tuning: Index selection and clustering, tuning of conceptual schema, denormalization, tuning queries and views;

Query optimization: Importance of query processing, equivalence of queries, join ordering, cost estimation, cost estimation for complex queries and joins, optimizing nested subqueries, I/O cost models, external sort.

Crash recovery: Failure classification, transactions, log maintenance, check point implementation, shadow paging, example of an actual implementation. Concurrency Control in RDBMS: Testing for serializability, lock based and time - stamp based protocols; Deadlock detection and Recovery.

NoSQL: Introduction to noSQL databases, ACID vs BASE requirements, practical exercises with one noSQL system (for example MongoDB).

MapReduce and Hadoop: Basics of MapReduce, Basics of Hadoop, Matrix-vector multiplication using MapReduce, relational algebra using MapReduce, matrix multiplication using MapReduce, combiners, cost of MapReduce algorithms, basics of Spark, practical exercises using Spark.

(b) **Prerequisites:** Nil

(c) **Hours:** Three lectures and one laboratory per week

(d) **Marks Distribution:** Theory 50%, Labs 50%

(e) **References:**

1. Database System Concepts, Sixth Edition: by Avi Silberschatz, Henry F. Korth, S. Sudarshan. McGraw-Hill. `http://www.db-book.com`
2. Database Management Systems, Third Edition: by Raghu Ramakrishnan and Johannes Gehrke. McGraw-Hill. `http://pages.cs.wisc.edu/~dbbook/`
3. Mining of Massive Datasets: by Jure Leskovec, Anand Rajaraman, Jeff Ullman. Cambridge University Press. `http://www.mmds.org`

Design and Analysis of Algorithms

(a) **Topics:** *Introduction and basic concepts:* Complexity measures, worst-case and average-case complexity functions, problem complexity, quick review of basic data structures and algorithm design principles. Paradigms of algorithm design – divide and conquer, induction, dynamic programming, etc.

Sorting and selection: Finding maximum and minimum, Selection of k-th largest element; Sorting by selection, tournament and heap sort methods, lower bound for sorting, other sorting algorithms – radix sort, quick sort, merge sort.

Searching and set manipulation: Searching in static table – binary search, path lengths in binary trees and applications, optimality of binary search in worst cast and average-case, binary search trees, construction of optimal weighted binary search trees; Searching in dynamic table – randomly grown binary search trees, AVL and (a, b) trees.

Hashing: Basic ingredients, analysis of hashing with chaining and with open addressing.

Union-Find problem: Tree representation of a set, weighted union and path compression-analysis and applications.

Graph problems: Graph searching – BFS, DFS, shortest first search, topological sort; connected and bi-connected components; shortest path and all-pair shortest path; minimum spanning trees, Kruskal's and Prim's algorithms, Johnsons implementation of Prim's algorithm using priority queue data structures. Introduction to Network flow and matching.

Algebraic problems: Evaluation of polynomials with or without preprocessing. Winograd's and Strassen's matrix multiplication algorithms and applications to related problems, FFT, simple lower bound results.

String processing: String searching and Pattern matching, Knuth-Morris-Pratt algorithm and its analysis.

Introduction to Randomized algorithms

NP-completeness: Concepts of deterministic and nondeterministic algorithms, polynomial reducibility, P and NP, NP-completeness, statement of Cooks theorem, some standard NP-complete problems, approximation algorithms.

(b) **Prerequisites:** Nil

(c) **Hours:** Three lectures and one two-hour tutorial per week

(d) **Marks Distribution:** Theory 70%, Assignments 30%
At least one assignment involving implementation of several algorithms of same asymptotic complexity for a problem and their empirical comparisons.

(e) **References:**

1. A. Aho, J. Hopcroft and J. Ullman: The Design and Analysis of Computer Algorithms, A. W. L, International Student Edition, Singapore, 1998
2. S. Baase: Computer Algorithms: Introduction to Design and Analysis, 2nd ed., Addison-Wesley, California, 1988.
3. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein : Introduction to Algorithms, third edition, MIT Press, 2009.
4. E. Horowitz and S. Sahni: Fundamental of Computer Algorithms, Galgotia Pub. /Pitman, New Delhi/London, 1987/1978.
5. K. Mehlhorn: Data Structures and Algorithms, Vol. 1 and Vol. 2, Springer-Verlag, Berlin, 1984.
6. A. Borodin and I. Munro: The Computational Complexity of Algebraic and Numeric Problems, American Elsevier, New York, 1975.
7. D. E. Knuth: The Art of Computer Programming, Vol. 1, Vol. 2 and Vol. 3. Vol. 1, 2nd ed., Narosa/Addison-Wesley, New Delhi/London, 1973; Vol. 2: 2nd ed., Addison-Wesley, London, 1981; Vol. 3: Addison-Wesley, London, 1973.
8. S. Winograd: The Arithmetic Complexity of Computation, SIAM, New York, 1980.
9. J. Kleinberg and Eva Tardos: Algorithm Design, Pearson Education.
10. S. Dasgupta, C. Papadimitriou and U. Vazirani: Algorithms, Tata McGraw-Hill.

Discrete Mathematics

(a) **Topics:** *Combinatorics*: Pigeonhole principle; Multinomial theorem, principle of inclusion exclusion; pigeonhole principle; Classification of recurrence relations, summation method, extension to asymptotic solutions from solutions for subsequences; Linear homogeneous relations, characteristic root method, general solution for distinct and repeated roots, non-homogeneous relations and examples, generating functions and their application to linear homogeneous recurrence relations, non-linear recurrence relations, exponential generating functions, brief introduction to Polya theory of counting.

Graph Theory: Graphs and digraphs, complement, isomorphism, connectedness and reachability, adjacency matrix, Eulerian paths and circuits in graphs and digraphs, Hamiltonian paths and circuits in graphs and tournaments, trees; Minimum spanning tree, rooted trees and binary trees, planar graphs, Euler's formula, statement of Kuratowski's theorem, dual of a planar graph, independence number and clique number, chromatic number, statement of Four-color theorem, dominating sets and covering sets.

Logic: Propositional calculus propositions and connectives, syntax; semantics truth assignments and truth tables, validity and satisfiability, tautology; Adequate set of connectives; Equivalence and normal forms; Compactness and resolution; Formal reducibility, natural deduction system and axiom system; Soundness and completeness. Introduction to Predicate Calculus: Syntax of first order language; Semantics structures and interpretation; Formal deductibility; First order theory, models of a first order theory (definition only), validity, soundness, completeness, compactness (statement only), outline of resolution principle.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. J. L. Mott, A. Kandel and T. P. Baker: Discrete Mathematics for Computer Scientists, Reston, Virginia, 1983.
2. D. F. Stanat and D. E. McAllister: Discrete Mathematics in Computer Science, Prentice Hall, Englewood Cliffs, 1977.
3. C. L. Liu: Elements of Discrete Mathematics, 2nd ed., McGraw Hill, New Delhi, 1985.
4. R. A. Brualdi: Introductory Combinatorics, North-Holland, New York, 1977.
5. Reingold et al.: Combinatorial Algorithms: Theory and Practice, Prentice Hall, Englewood Cliffs, 1977.
6. J. A. Bondy and U. S. R. Murty: Graph Theory with Applications, Macmillan Press, London, 1976.
7. N. Deo: Graph Theory with Applications to Engineering and Computer Science, Prentice Hall, Englewood Cliffs, 1974.
8. Douglas B. West: Introduction to Graph Theory, Pearson, 2000
9. Reinhard Diestel: Graph Theory, Springer, 2010

10 Frank Harary: Graph Theory, Narosa Publishing House, 2001

11. E. Mendelsohn: Introduction to Mathematical Logic, 2nd ed. Van-Nostrand, London, 1979.
12. L. Zhongwan: Mathematical Logic for Computer Science, World Scientific, Singapore, 1989.
13. Fred S. Roberts, Barry Tesman: Applied Combinatorics, Chapman and Hall/CRC; 2 edition, 2008.
14. Lewis and Papadimitriou: Elements of Theory of Computation (relevant chapter on Logic), Prentice Hall, New Jersey, 1981.

Elements of Algebraic Structures

(a) **Topics:** *Introduction:* Sets, operations on sets, relations, equivalence relation and partitions, functions, induction and inductive definitions and proofs, cardinality of a set, countable and uncountable sets, diagonalisation argument.

Groups: Binary operations, groupoids, semi-groups and monoids, groups, subgroups and cosets, Lagranges theorem, cyclic group, order of an element, normal subgroups and quotient groups, homomorphism and isomorphism, permutation groups and direct product.

Rings and sub-rings: Introduction to rings, sub-rings, ideals and quotient rings, homomorphism and isomorphism, integral domains and fields, field of fractions, ring of polynomials.

Field extensions: Finite dimensional, algebraic and transcendental; splitting field of a polynomial, existence and uniqueness of finite fields, application to Coding Theory.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. D. F. Stanat and D. E. McAllister: Discrete Mathematics in Computer Science, Prentice Hall, Englewood Cliffs, 1977.
2. C. S. Sims: Abstract Algebra: A Computational Approach, John Wiley, New York, 1984.
3. K. H. Kim, F. W. Kim and F. W. Rough: Applied Abstract Algebra, Ellis Horwood, Chichester, 1983.
4. C. H. Sah: Abstract Algebra, Academic Press, London, 1967.
5. L. L. Domhoff and F. E. Hohn: Applied Modem Algebra, Macmillan, New York, 1978.
6. J. B. Fraleigh: First Course in Abstract Algebra, Narosa/Addison-Wesley, New Delhi/Reading, 1982/1978.
7. I. N. Herstein: Topics in Algebra, Vikas Pub., New Delhi 1987.
8. G. Birkhoff and S. McLane: A Survey of Modern Algebra, 4th ed. Macmillan, New York, 1977.
9. M. Artin: Algebra, Pearson, 2010

Introduction to programming (non-credit)

(a) **Topics:**

Writing, compiling, and running basic programs.

Introduction to an imperative language (preferably C); syntax and constructs.

Functions and parameter passing, call by value, call by reference; recursion.

Basics of object-oriented programming: introduction to object oriented programming, classes and methods, polymorphism, inheritance; basics of C++ or Java.

Basics of functional programming, logic programming.

Efficiency issues.

(b) **Prerequisites:** None.

(c) **Hours:** Two three-hour hands-on sessions per week

(d) **Marks Distribution:** This is a non-credit course, the instructor may give assignments for practice but the students do not obtain any credit for this course.

(e) **References:**

1. B.W. Kernighan and D.M. Ritchie: The C Programming Language, Prentice Hall, 1980.
2. B. Gottfried: Programming in C, Schaum Outline Series, 1996.
3. B. Stroustrup: The C++ Programming Language, 2nd ed., Addison-Wesley, 1995.
4. Cay S. Horstmann: Core Java Volume I – Fundamentals, 11th ed., Prentice Hall, 2018.
5. Joshua Bloch: Effective Java, 3rd ed., Addison-Wesley, 2018.

6. B.W. Kernighan and R. Pike: The Practice of Programming, Addison-Wesley, .
7. B.W. Kernighan and P.J. Plauger: The Elements of Programming Style, McGraw-Hill, .
8. J. Bentley: Programming Pearls, Addison-Wesley, 1986.
9. J. Bentley: More Programming Pearls, Addison-Wesley, 1988.
10. B.W. Kernighan and R. Pike: The Unix Programming Environment, Prentice Hall, .

Linear Algebra

(a) **Topics:** *Matrices and System of Equations:* System of Linear Equations, Row Echelon Form, Matrix Arithmetic, Matrix Algebra, Elementary Matrices, Partitioned Matrices, Determinant and its properties.

Vector Spaces: Definition and Examples, Subspaces, Linear Independence, Basis and Dimension, Change of Basis, Row Space, Column Space, Null space

Inner product spaces: The Euclidean dot product, Orthogonal Subspaces, Least Squares Problems, Orthonormal Sets, The GramSchmidt Orthogonalization Process, Orthogonal Polynomials

Linear Transformations: Definition and Examples, Matrix Representations of Linear Transformations, Similarity

Eigenvalues and Eigenvectors: System of Linear Differential Equations, Diagonalization, Hermitian Matrices, Singular Value Decomposition.

Quadratic Forms: Classification and characterisations, Optimisation of quadratic forms.

Algorithms: Gaussian Elimination with different Pivoting Strategies; Matrix Norms and Condition Numbers; Orthogonal Transformations; The Eigenvalue Problem; Least Squares Problems.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Steve Leon: Linear Algebra with Applications, Pearson Global edition.
2. Jin Ho Kwak and Sungpyo Hong: Linear Algebra, 2nd ed., Birkhäuser, 2004.
3. T. Banchoff and J. Wermer: Linear Algebra Through Geometry, 2nd edition, Springer, 2011.

Operating Systems

(a) **Topics:** *Introduction:* Basic architectural concepts, interrupt handling, concepts of batch-processing, multiprogramming, time-sharing, real-time operations; Resource Manager view, process view and hierarchical view of an OS.
Memory management: Partitioning, paging, concepts of virtual memory, demand-paging – page replacement algorithms, working set theory, load control, segmentation, segmentation and demand-paging, Cache memory management.
Process management: CPU scheduling – short-term, medium term and long term scheduling, non-preemptive and preemptive algorithms, performance analysis of multiprogramming, multiprocessing and interactive systems; Concurrent processes, precedence graphs, critical section

problem - 2-process and n-process software and hardware solutions, semaphores; Classical process co-ordination problems, Producer-consumer problem, Reader-writer problem, Dining philosophers problem, Barber's shop problem, Interprocess communication.

Concurrent Programming: Critical region, conditional critical region, monitors, concurrent languages (eq. concurrent Pascal), communicating sequential process (CSP); Deadlocks: prevention, avoidance, detection and recovery.

Device Management: Scheduling algorithms – FCFS, shortest-seek-time-first, SCAN, C-SCAN, LOOK, C-LOOK algorithms, spooling, spool management algorithm.

Information Management: File concept, file support, directory structures, symbolic file directory, basic file directory, logical file system, physical file system, access methods, file protection, file allocation strategies.

Protection: Goals, policies and mechanisms, domain of protection, access matrix and its implementation, access lists, capability lists, Lock/Key mechanisms, passwords, dynamic protection scheme, security concepts and public and private keys, RSA encryption and decryption algorithms.

A case study: UNIX OS file system, shell, filters, shell programming, programming with the standard I/O, UNIX system calls.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week (including tutorial/lab)

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1 A. Silberschatz and P. B. Galvin: Operating Systems Concepts, 9-th ed., John Wiley and Sons, New York, 2012.

2 P. B. Hansen: Operating System Principles, Prentice Hall, Englewood Cliffs, 1980.

3 A. S. Tannenbaum: Modern Operating Systems, Prentice Hall, Englewood Cliffs, 1992.

4 S. E. Madnick and J. J. Donovan: Operating Systems, McGraw Hill, New York, 1974.

Principles of Programing Languages

(a) **Topics:** *Introduction:* Overview of different programming paradigms e.g. imperative, object oriented, functional, logic and concurrent programming.

Syntax and semantics of programming languages: A quick overview of syntax specification and semiformal semantic specification using attribute grammar.

Imperative and OO Languages: Names, their scope, life and binding. Control-flow, control abstraction; subprogram and exception handling. Primitive and constructed data types, data abstraction, inheritance, type checking and polymorphism.

Functional Languages: Typed-calculus, higher order functions and types, evaluation strategies, type checking, implementation.

Logic Programming: Computing with relation, first-order logic, SLD-resolution, unification, sequencing of control, negation, implementation, case study.

Concurrency: Communication and synchronization, shared memory and message passing, safety and liveness properties, multithreaded program.

Formal Semantics: Operational, denotational and axiomatic semantics, languages with higher order constructs and types, recursive type, subtype, semantics of non-determinism and concurrency.

Assignments: Using one or more of the following as time permits: C++ / Java / OCAML / Lisp / Haskell / Prolog

(b) **Prerequisites:** None

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 70%, Programming assignments 30%

(e) **References:**

1. Glynn Winskel: A Formal Semantics of Programming Languages: An Introduction, MIT Press Cambridge, MA, USA, 1993.
2. Benjamin C. Pierce: Types and Programming Languages, MIT Press Cambridge, MA, USA, 2002.
3. Daniel P. Friedman, Mitchell Wand and Christopher T. Haynes: Essentials of Programming Languages, MIT Press Cambridge, MA, USA, 1992.
4. Terrence W. Pratt and Marvin V. Zelkowit: Programming Languages: Design and Implementation, Prentice Hall, 2001.
5. Allen B. Tucker and Robert Noonan: Programming Languages, Principles and Paradigms, Tata McGraw Hill Education Pvt. Ltd., 2007.
6. Robert W. Sebesta: Concepts of Programming Languages, 10th ed., Pearson, 2013.

Probability and Stochastic Processes

(a) **Topics:** Sample space and Probability theory; Discrete random variables; Continuous random variables; Functions of random variables; Distributions; Expectation and moments; moment generating function; some tail inequalities like Markov, Chebyshev, Chernoff bounds; ideas of Limit theorems; Balls and bins framework; Martingales; Markov chains; Random walks on graphs; Branching processes.

(b) **Prerequisites:** None

(c) **Hours:** Two lectures each of two hours duration

(d) **Marks Distribution:** Theory 80%, assignments 20%

(e) **References:**

1. Dimitri P. Bertsekas and John N. Tsitsiklis: Introduction to Probability (2nd Edition), Athena Scientific, 2008.
2. William Feller: An Introduction to Probability and its Applications: Volume I (3rd Edition), Wiley, 2008
3. William Feller: An Introduction to Probability and its Applications: Volume II (2nd Edition), John Wiley & Sons, Inc, 1971
4. Sheldon Ross: A First Course in Probability (11th Edition), Academic Press, 2014

5. Noga Alon and Joel Spencer: The Probabilistic Method (4th Edition), Wiley-Blackwell, 2016
6. Michael Mitzenmacher and Eli Upfal: Probability and Computing (2nd Edition), Cambridge University Press, 2017

Statistics

(a) **Topics:**

Representation of Data:

Measures of central Tendency:

Measures of Dispersion:

Correlation: Product moments; Rank correlation

Regression: Simple and multiple

Estimation: method of moments, maximum likelihood estimation.

Hypothesis Testing:

ANOVA:

Additional topics: Nonparametric Regression; Classification & Clustering

(b) **Prerequisites:** None.

(c) **Hours:** Four lectures per week including a tutorial.

(d) **Marks Distribution:** Theory 70%, Assignments 30%

(e) **References:**

1. J. M. Tanur (ed.): Statistics: A Guide to the Unknown. Pacific Grove, Calif.: Wadsworth & Brooks/Cole, Advanced Books & Software, 1989.
2. D. Freedman, R. Pisani and R. Purves: Statistics. 4th ed., Viva Books, 2011.
3. M. Tanner: An Investigation for a Course in Statistics.
4. M. G. Kendall and A. Stuart: The Advanced Theory of Statistics, Vol. I and II. 5th ed., Oxford University Press, 1991.
5. J. F. Kenney and E. S. Keeping: Mathematics of Statistics. 3rd ed., Van Nostrand, 1964.
6. G. U. Yule and M. G. Kendall: An Introduction to the Theory of Statistics. 14th ed., Charles Griffin, 1950.
7. C. R. Rao: Linear Statistical Inference and its Applications. 2nd ed., Wiley.
8. C. E. Croxton and D. J. Cowden: Applied General Statistics. Prentice-Hall Inc.
9. A. M. Goon, M. Gupta and B. Dasgupta: Fundamentals of Statistics, Vol I. World Press.
10. P. G. Hoel, S. C. Port and Charles J. Stone: Introduction to Statistical Theory. Boston: Houghton-Mifflin, 1971.
11. W. A. Wallis and H. V. Roberts: Statistics: A New Approach. Methuen & Co., 1960.
12. P. J. Bickel and K. A. Doksum: Mathematical Statistics. Chapman and Hall/CRC, 2015.
13. L. Wasserman: All of Statistics: A Concise Course in Statistical Inference. Springer.
14. C. Casella and R. L. Berger: Statistical Inference. 2nd ed., Cengage Learning, 2001.

Electives

Advanced Computer Networks

(a) **Topics:** *Introduction:* Overview and motivation, Characteristics of communication networks, Protocol design issues, Protocol stacks and layering

Transmission fundamentals: Analog and digital transmissions, Different transmission media, Different transmission impairments, Different modulation techniques, Channel capacity, Basic concept of spread spectrum and frequency hopping, Asynchronous and synchronous transmission, Multiplexing.

Communication networks: Introduction to LANs, MANs, and WANs; Switching techniques: Circuitswitching and Packet-switching; Topological design of a network, LAN topologies, Ethernet, Performance of Ethernet, Repeaters and bridges, Asynchronous Transfer Mode.

Queuing theory: Introduction to queuing theory and systems, Elementary queuing systems, Network performance analysis using queuing systems.

Data link layer: Services and design issues, Framing techniques, Error detection and correction, Flow control: Stop-and-wait and Sliding window; Performance analysis of stop-and-wait and sliding window protocols, MAC Protocols: ALOHA, CSMA, CSMA/CD, Collision free protocols, Limited contention protocol; Wireless LAN protocols: MACA, CSMA/CA; Comparative analysis of different MAC protocols.

Internetworking and IP: Design issues, Organization of the subset, Routing: Static and dynamic routing, Shortest path routing, Flooding, Unicast and multicast routing, Distance-vector routing, Linkstate routing; Congestion control: choke packets, leaky bucket, token bucket; IP protocol, IPV4, IPV6, IP addressing, CIDR, NAT, Internet control protocols: ICMP, ARP, RARP.

Transport and Reliable Delivery: Design issues, Port and socket, Connection establishment and release, TCP, UDP, TCP congestion control, TCP timer management, RPC.

(b) **Prerequisites:** Computer Organization, Computer Networks

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 50%, Assignments/Labs 50%

(e) **References:**

1. Larry L. Peterson and Bruce S. Davie, Computer Networks: A Systems Approach, Morgan Kaufmann Publishers.
2. Andrew S. Tanenbaum, "Computer Networks", Prentice Hall.
3. William Stallings, Data and Computer Communcations, Prentice Hall.
4. Bertsekas and Gallagher, Data Networks, Prentice Hall.
5. W. R. Stevens, Unix Network Programming, PHI, 2009
6. W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley Professional

Advanced Logic and Automata Theory

(a) **Topics:** *Monadic second order logic:* syntax, semantics, truth, definability, relationship between logic and languages, Büchi-Elgot-Trakhtenbrot theorem.

Automata on infinite words: Büchi automata, closure properties, Müller automata, Rabin automata, Streett automata, determinization, decision problems, Linear temporal logic and Büchi automata, Finite and infinite tree automata, closure properties, decision problems, complementation problem for automata on infinite trees, alternation, Rabins theorem. Modal mu-calculus: syntax, semantics, truth, finite model property, decidability, Parity Games, model checking problem, memoryless determinacy, algorithmic issues, bisimulation, Janin/Walukiewicz theorem.

(b) **Prerequisites:** Discrete Mathematics, Automata Theory, Languages and Computation, Logic for Computer Science

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. B. Khoussainov and A. Nerode: Automata Theory and its Applications, Springer, 2001.
2. E. Gradel, W. Thomas and T. Wilke (Eds.): Automata, Logics, and Infinite Games, LNCS 2500, Springer, 2002.
3. D. Perrin and J.-E. Pin: Infinite Words: Automata, Semigroups, Logic and Games, Elsevier, 2004.
4. H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, M. Tommasi: Tree Automata Techniques and Applications, (open source: `http://tata.gforge.inria.fr`), 2008.
5. P. Blackburn, M. de Rijke and Y . V enema: Modal Logic, Cambridge University Press, 2001.
6. Y. Venema: Lectures on the Modal mu-calculus, (available at `https://staff.science.uva.nl/y.venema/teaching/ml/mu/mu20121116.pdf`), 2012.

Advanced Operating Systems

(a) **Topics:** The instructor may select only some of the following topics, and include other topics of current interest

Operating systems structures: monolithic, microkernel, ExoKernel, multi kernel.

System calls, interrupts, exceptions.

Symmetric Multi Processor (SMP) systems: scheduling, load balancing, load sharing, process migration; synchronisation in SMP systems.

Interprocess communication: signals, message passing.

Naming in distributed systems: directory services, DNS.

Remote Procedure Calls (RPC): model, stub generation, server management, parameter passing, call semantics, communication protocols, client-server binding, exception handling, security, optimization.

Distributed shared memory: architecture, consistency model, replacement strategy, thrashing, coherence.

File systems: Fast File System (FFS), Virtual File System (VFS), log-structured file systems and journalling, RAID; Distributed File Systems (DFS), stateless and stateful DFS, Andrew File System (AFS), Network File Systems (NFS).

Virtualisation: introduction, nested virtualisation, case study.

Device drivers.

Fault tolerance.

Clusters, cloud computing.

Protection and security.

Projects and real systems implementations

(b) **Prerequisites:** Computer organisation; Operating systems.

(c) **Hours:** Three lectures and one lab-session per week.

(d) **Marks Distribution:** Theory 50%; Labs 50%.

(e) **References:**

1. Thomas Anderson and Michael Dahlin: Operating Systems Principles and Practice, 2nd ed., Recursive Books, 2014.
2. Daniel P. Bovet and Marco Cesati: Understanding the Linux Kernel, 3rd ed., O'Reilly 2005/2008.
3. Robert Love: Linux Kernel Development, 3rd ed., Addison-Wesley Professional, 2010.
4. Jonathan Corbet, Alessandro Rubini and Greg Kroah-Hartman: Linux Device Drivers, 3rd ed., O'Reilly, 2005.
5. Research articles as prescribed by the instructor.

Algorithms for Big Data

(a) **Topics:** *Review of Linear Algebra and Probability*

Sketching and Streaming algorithms for basic statistics: Distinct elements, heavy hitters, frequency moments, p-stable sketches.

Dimension Reduction: Johnson Lindenstrauss lemma, lower bounds and impossibility results

Graph stream algorithms: connectivity, cut/spectral sparsifiers, spanners, matching, graph sketching.

Lower bounds for Sketching and Streaming.

Communication complexity: Equality, Index and Set-Disjointness.

Locality Sensitive Hashing: similarity estimation, approximate nearest neighbor search, data dependent hashing.

Fast Approximate Numerical Linear Algebra: matrix multiplication, low-rank approximation, subspace embeddings, least squares regression

(b) **Prerequisites:** Probability and Stochastic processes, Linear Algebra.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Avrim Blum, John Hopcroft, and Ravindran Kannan: Foundations of Data Sciences by, `https://www.cs.cornell.edu/jeh/book.pdf`
2. Dimitri P. Bertsekas and John N. Tsitsiklis: Introduction to Probability (2nd Edition), Athena Scientific, 2008.
3. Michael Mitzenmacher and Eli Upfal: Probability and Computing (2nd Edition), Cambridge University Press, 2017
4. Noga Alon and Joel Spencer: The Probabilistic Method (4th Edition), Wiley-Blackwell, 2016

Algorithms for Electronic Design Automation

(a) **Topics:** *Introduction:* VLSI design, design styles and parameters, popular technologies.

Logic synthesis: PLA minimization, folding, testing. Role of BDDs. Logic design tools- ESPRESSO, SIS, OCTOOLS.

High level synthesis: Design description languages introduction to features in VHDL, Verilog; Scheduling algorithms; Allocation and Functional binding.

Layout synthesis: Design rules, partitioning, placement and floor planning, routing in ASICs, FPGAs; CAD tools

Advanced Topics: Design for Hardware Security and IP Protection; Design of Manufacturability

(b) **Prerequisites:** Computer Organization.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 75%, assignments 25%

(e) **References:**

1. D. Pucknell and K. Eshraghian: Basic Principles of VLSI Design, Prentice Hall, Englewood Cliffs, 1985.
2. E. D. Fabricius: Introduction to VLSI Design, McGraw Hill, New York, 1990.
3. N. Weste and K. Eshraghian: Principles of CMOS Design, 2nd ed., Addison-Wesley, Reading, Mass., 1993.
4. C. Mead and L. Conway: Introduction to VLSI Systems, Addison-Wesley, Reading, Mass., 1980.
5. R. K. Brayton et al: Logic Minimization for VLSI Synthesis, Kluwer Academic Publishers, Boston, 1984.
6. D. Gajski, N. Dutt et al: High Level Synthesis: Introduction to Chip and System Design, Kluwer Academic, Boston, 1992.
7. M. Sarrafzadeh and C. K. Wong: AN Introduction to VLSI Physical Design, McGraw Hill, New York, 1996.
8. N. Sherwani: Algorithms for VLSI Physical Design Automation, Kluwer Academic, Boston, 1999.
9. B. T. Preas and M. Lorenzetti: Physical Design automation of VLSI Systems, Benjamin Cummings Pub., 1988.
10. T. Ohtsuki (ed): Layout Design and Verification, North Holland, Amsterdam, 1986.
11. Bhunia, Swarup, Ray, Sandip, Sur-Kolay, Susmita (Eds.): Fundamentals of IP and SoC Security: Design, Verification, and Debug

Coding Theory

(a) **Topics:** *Introduction:* Basic definitions: codes, dimension, distance, rate, error correction, error detection.

Linear Codes: Properties of linear codes; Hamming codes; Efficient decoding of Hamming codes; Dual of a linear code

Gilbert Varshamov bound; Singleton bound; Plotkin bound

Shannon's Theorems: Noiseless coding; Noisy Coding; Shannon Capacity

Algebraic codes: Reed-Solomon codes; Concatenated codes; BCH codes; Reed-Muller codes; Hadamard codes; Dual BCH codes.

Algorithmic issues in coding: Decoding Reed-Solomon Codes; Decoding Concatenated Codes

List Decoding: List decoding; Johnson bound; List decoding capacity; List decoding from random errors. List decoding of Reed-Solomon codes.

Advanced Topics: Graph Theoretic Codes; Locality in coding: Locally decodable codes, locally testable codes; codes and derandomization.

(b) **Prerequisites:** Design and Analysis of Algorithms; Elements of Algebraic Structures; Linear Algebra

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. Venkatesan Guruswami, Atri Rudra, Madhu Sudan: Essential Coding Theory, internet draft available at: `https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/web-coding-book.pdf`
2. F.J. MacWilliams and Neil J.A. Sloane: Theory of Error-Correcting Codes, North Holland Publishing Co., 1977.
3. Jacobus H. van Lint: Introduction to Coding Theory, Springer, 1973.
4. Vera S. Pleaa and W. Cary Huffman (Eds.), Handbook of Coding Theory

Computational Algebra and Number Theory

(a) **Topics:** *Polynomial Manipulations:* GCD and Berlekamp-Massey algorithm, factoring polynomials over finite fields, Berlekamps algorithm and fast probabilistic algorithm; Factoring polynomials over the integers, p-adic methods and lattice reduction, deterministic algorithms.

Matrix Computations: Asymptotically fast matrix multiplication algorithms; Symbolic and exact solutions of linear systems, and Diophantine analyses, normal forms over fields, algorithms for large sparse matrices, co-ordinate recurrence methods.

Solving Systems of Non-linear Equations: Gröbner basis, reduced Gröbner bases and Buchbergers algorithm; Dimensions of ideals, the number of zeros of an ideal, decomposition of ideals, approximating zeros of real polynomial systems; Applications to word problem, automatic theorem proving, term rewriting systems, complexity of Gröbner basis computation.

Computer Algebra Systems: Issues of data representation – sparse, dense, canonical, normal; Representations of polynomials, matrices and series; Simplification of expressions and systems -

canonical simplification of polynomials; Knuth-Bendix critical pair and completion algorithms; Cylindrical decompositions.

Algebraic Complexity Theory: Uniform and non-uniform models, straight-line and branching programs; Survey of lower bound results for polynomial, matrix and bilinear computations.

(b) **Prerequisites:** Elements of Algebraic Structures, Design and Analysis of Algorithms.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. A. V. Aho, J. E. Hopcroft and J. D. Ullman: The Design and Analysis of Computer Algorithms, AWL International Students Edition, Singapore, 1998.
2. T. Becker and V. Weispfenning: Grobner Bases: A Computational Approach to Commutative Algebra, Springer-Verlag, New York, 1991.
3. A. Borodin and L. Munro: The Computational Complexity of Algebraic and Numeric Problems, American Elsevier Publishing Co., New York, 1975.
4. B. Buchberger, G. E. Collins and R. Loas (Eds.): Computer Algebra: Symbolic and Algebraic Computing, Computing Supplement 4, Springer-Verlag, Berlin, 1982.
5. H. Cohen: A Course in Computational Number Theory, Springer-Verlag, Berlin, 1993.
6. D. A. Cox, J. B. Little, and D. Oshea: Ideals, Varieties, and Algorithms: An Introductions to Computational Algebraic Geometry and Commutative Algebra, Springer-Verlag, Berlin, 1996.
7. J. H. Davenport, Y. Siret, E. Tournier and F. Tournier: Computer Algebra: Systems and Algorithms for Algebraic Computations, 2nd ed., Academic Press, New York, 1993.
8. K. Gedder, S. Czapor and Q. Labalin: Algorithms for Computer Algebra, Kluwer Academic Publishers, Boston, 1992.
9. D. E. Knuth: The Art of Computer Programming; Semi-Numerical Algorithms, Vol. 2, 3rd ed., Addison-Wesley Publishing Company, Reading, Mass., 1997.
10. R. Lidl and H. Niederreiter: Introduction to Finite Fields and their Applications, Cambridge University Press, London, 1994
11. J. D. Lipson: Elements of Algebra and Algebraic Computing, Addison- Wesley, Reading, Mass., 1981.
12. B. Peter, C. Michael, and S. M. Amin: Algebraic Complexity Theory, Springer, Berlin, 1997.
13. J. Von zur Gathen: Algebraic Complexity Theory, In: Ann. Rev. of Computer Science 3, pp. 317- 347, 1988.
14. J. Von zur Gathen and J. Gerhard: Modem Computer Algebra, Cambridge University Press, London, 1999.
15. S. Winograd: The Arithmetic Complexity of Computations, SIAM, 1980.
16. R. Zippel: Effective Polynomial Computation, Kluwer Academic Press, Boston, 1993. B18. Lambda-

Computational Complexity

(a) **Topics:**

Introduction: Review of machine models, Turing machines and its variants, reduction between problems and completeness, time and space complexity classes

Structural Results: Time and space hierarchy theorems, polynomial hierarchy, Ladner's theorem, relativization, Savitch's theorem

Circuit Complexity: Circuits and non-uniform models of computation, parallel computation and NC, P-completeness, circuit lower bounds, AC^0 and parity not in AC^0, Hastad's Switching Lemma, introduction to natural proof barrier

Random Computation: Probabilistic computation and complexity classes and their relations with other complexity classes, BPP=P?

Interactive proofs: Introduction to Arthur-Merlin Games, IP=PSPACE, multiprover interactive proofs, introduction to PCP theorem

Complexity of counting: Complexity of optimization problems and counting classes, Toda's theorem, inapproximability, application of PCP theorem to inapproximability and introduction to unique games conjecture

Cryptography: Public-key cryptosystems, one-way functions, trapdoor-functions, application to derandomization

(b) **Prerequisites:** Discrete Mathematics, Design and Analysis of Algorithms, Automata Theory Languages and Computation

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. J. Balcazar, J. Diaz and J. Gabarro: Structural Complexity – I, Springer-Verlag, Berlin, 1988. Structural Complexity – II, Springer-Verlag, Berlin, 1990.
2. D. P. Bovet ad P. Crescenzi: Introduction to the Theory of Complexity, Prentice Hall, Englewood Cliffs, 1994.
3. M. Sipser: Introduction to Theory of Computation, PWS Pub.Co, New York, 1999.
4. C. H. Papadimitriou: Computational Complexity, Addison-Wesley, Reading, Mass., 1994.
5. J. E. Hopcroft and J. D. Ullman: Introduction to Automata Theory, Languages and Computation, Addison-Wesley, Reading, Mass., 1979.
6. O. Goldreich: Lecture Notes on Computational Complexity, Tel Aviv Univ., 1999.
7. S. Arora and B. Barak: Computational Complexity: A Modern Approach, Cambridge University Press, 2009.

Computational Finance

(a) **Topics:** *Basic Concepts*: (i) Arbitrage, Principle of no arbitrage, Law of one price; Frictionless / Efficient market, Transaction cost, Contingent contracts, Concept of complete market (ii) Time value of money, discounting: deterministic and stochastic; Martingale, Risk neutral valuation, Equivalent martingale measure; (iii) Mean Variance utility / Normal distributed returns; Capital Asset pricing Model (CAPM), Extensions, test for efficiency

Contracts: Forwards, Futures, Options (Call, Put, European, American, Exotics), Combinations; Risk neutral portfolio construction

Valuation of contracts in discrete time models. Computation using Binomial tree. Link with the continuous time model: Brownian motion, Black Scholes option pricing and hedging.

High frequency trading (Machine learning, Neural networks), Algorithmic trading

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Pliska, S.R.: Introduction to Mathematical Finance: Discrete Time Models
2. Hull, J.C.: Options, Futures, and Other Derivatives
3. Prisman, E.: Pricing Derivative Securities
4. Oksendal, B.: Stochastic Differential Equations, An Introduction with Applications
5. Selected research papers

Computational Game Theory

(a) **Topics:** *Computing in Games:* Basic Solution Concepts and Computational Issues, Strategies, Costs and Payoffs, Basic Solution Concepts, Equilibria and Learning in Games.

Refinement of Nash: Games with Turns and Subgame, Perfect Equilibrium. Nash Equilibrium without Full Information: Bayesian Games, Cooperative Games, Markets and their algorithmic issues.

The Complexity of Finding Nash Equilibria

Equilibrium Computation for Two-Player Games in Strategic and Extensive Form

Learning, Regret Minimization, and Equilibria: External Regret Minimization, Generic Reduction from External to Swap Regret, Partial Information Model

Combinatorial Algorithms for Market Equilibria

Introduction to Mechanism Design: Social Choice, Mechanisms with Money

Cost Sharing: Cooperative Games and Cost Sharing, Group-Strategy proof Mechanisms and Cross-Monotonic Cost-Sharing Schemes, The Shapley Value and the Nash Bargaining Solution

Online Mechanisms

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Reference: Nisan, Roughgarden, Tardos, Vazirani (eds), Algorithmic Game Theory

Computational Geometry

(a) **Topics:** *Preliminaries:* Basic Euclidean geometry

Grids and Hulls: Fixed-radius near neighbors, convex hull algorithms, dominance and applications.

Linear Programming: Half-plane intersection and randomized LP, backwards analysis, applications of low-dimensional LP.

Intersections and Triangulation: Line segment intersection using plane-sweep, triangulation of monotone subdivisions, triangulation of simple polygons using plane-sweep, art gallery problems.

Point Location: Trapezoidal decompositions and analysis, history DAGs.

Voronoi Diagrams: Basic definitions and properties, Fortune's algorithm.

Geometric Data Structures: kd-trees, range trees and orthogonal range searching, segment trees, ham-sandwich cut tree, simplex range searching.

Delaunay Triangulations: Point set triangulations, basic definition and properties, randomized incremental algorithm and analysis.

Arrangements and Duality: Point/line duality, incremental construction of arrangements and the zone-theorem, applications.

Geometric Approximation: Dudley's theorem and applications, well-separated pair decompositions and geometric spanners, VC dimension, epsilon-nets and epsilon-approximations, Polynomial Time Approximation Schemes (Shifting Strategy of Hochbaum and Maass)

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Computational Geometry: An Introduction, F. P. Preparata and M. I. Shamos:, Springer-Verlag, Berlin, 1985.
2. Computational Geometry: Algorithms and Applications(3rd Edition), M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, Springer-Verlag, 2008.
3. Geometric Approximation Algorithms, S. Har-Peled, American Mathematical Society, 2010.
4. Lectures on Discrete Geometry, J. Matousek, Springer, 2002.
5. David Mount's Lecture notes on Computational Geometry (CMSC 754)

Computational Molecular Biology and Bioinformatics

(a) **Topics:**

The instructor can choose from the broad list of topics given against the following heads.

Sequence Alignments: Global alignments (Needleman-Wunsch), Local alignments (Smith-Waterman), k-mer based methods (BLAST), Advanced alignment methods (Gibbs sampling, suffix trees).

Genome: NOVA on genomics, Genetic mapping, Physical mapping, Recombinant DNA and Sequencing technologies, Whole-genome shotgun (Arachne) and clone-by-clone sequencing (Walking), Population genomics, SNP discovery, discovery of copy number variation and other structural variations, disease mapping, Gene recognition (Genscan) and cross-annotation (Rosetta).

Transcriptome and Evolution: Regulation - Transcription regulation, microarray technology, expression clustering, DNA binding sites, location analysis, regulatory motif prediction, Ribozymes, RNA World, RNA secondary structure, non-coding RNAs, Evolution: RNA world, multiple alignments, phylogeny. Protein Structure: Introduction to protein structure, Protein motifs - hidden Markov models for MSA, prediction (coiled-coil and beta-helix motifs), Threading.

Protein Dynamics: Molecular mechanics, Side-chain packing, Drug discovery tools, Lattice models for protein folding, Simulating virus shell assembly. Biochemical Pathways: Systems Biology and analysis of biochemical pathways, Kinetic modeling, Network based modeling, Flux balance analysis, Omics and Multiomics Data Analysis.

(b) **Pre-requisite:** Design and Analysis of Algorithms.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 70% and Assignments 30%

(e) **References:**

1. C. Setubal and J. Meidanis: Introduction to Computational Molecular Biology, PWS Publishing Company, Boston, 1997.
2. P. A. Pevzner: Computational Molecular Biology An Algorithmic Approach, MIT Press, 2000.
3. R. Durbin, S. R. Eddy, A. Krogh and G. Mitchison: Biological Sequence Analysis Probabilistic Models of Proteins and Nucleic Acids, Cambridge University Press, 1998.
4. D. Gusfield: Algorithms on Strings, Trees, and Sequences, Cambridge University Press, USA, 1997.
5. H. Lodish, A. Berk, S. L. Zipursky, P. Matsudaira, D. Baltimore and J. Darnell: Molecular Cell Biology, W. H. Freeman, USA, 2000.
6. C.-I. Branden, J. Tooze: Introduction to Protein Structure, Garland Publishing, 1998.
7. A. Kowald, C.hristoph Wierling, E. Klipp, and W. Liebermeister: Systems Biology, Wiley-VCH, 2016.
8. B.O. Palsson: Systems Biology - Constraint based Reconstruction and Analysis, Cambridge Univer- sity Press, 2015.

Computational Topology

(a) **Topics:**

i. Planar graphs

ii. Introduction to classification of surfaces, and graphs embedded on surfaces.

iii. Introduction to homotopy and homology

iv. Computational problems in homotopy and homology

v. Introduction to computational 3-manifold theory

vi. Complexity issues in high dimensional computational topology

vii. Persistent homology and its applications

viii. Distance to a measure

(b) **Prerequisites:** Design and Analysis of Algorithms, Probability and Stochastic process

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. Êric Colin de Verdiŕe: Algorithms for embedded graphs, `http://monge.univ-mlv.fr/~colinde/cours/all-algo-embedded-graphs.pdf`
2. Herbert Edelsbrunner and John Hared: Computational Topology: An Introduction, American Mathematical Society, 2009
3. Bojan Mohar and Carsten Thomassen: Graphs on Surfaces, Johns Hopkins University Press, 2001
4. Joel Hass, J. C. Lagarias, Nicholas Pippenger: The Computational Complexity of Knot and Link Problems, J. ACM 46(2), pp. 185-211, 1999
5. Francis Lazarus and Arnaud de Mesmay: Lecture Notes on Computational Topology
6. Jean-Daniel Boissonnat, Frdric Chazal and Mariette Yvinec: Geometric and Topological Inference by `https://geometrica.saclay.inria.fr/team/Fred.Chazal/papers/CGLcourseNotes/main.pdf`

Computer Graphics

(a) **Topics:** *Overview of Graphics Systems:* displays, input devices, hard copy devices, GPU, graphics software, graphics programming language, e.g. openGL

Line drawing algorithms, circle and ellipse drawing algorithms, polygon filling, edge based fill algorithms, seed fill algorithms

2D and 3D camera geometry, Affine and Projective transformations, Orthographic and Perspective view transformations, object to image projection, pin-hole camera model, 3D scene reconstruction, epipolar geometry

2D and 3D clipping, subdivision line-clipping algorithms, line clipping for convex boundaries. Sutherland-Hodgman algorithm, Liang-Barsky algorithm

Hidden line and hidden surfaces algorithms, ray tracing and z-buffer algorithm, Floating horizon algorithm, list priority and backface culling algorithms

2D and 3D object representation and visualization, Bezier and B-Spline curves and surfaces, 2D and 3D surface mesh representation and drawing, sweep representations, constructive solid geometry methods, Octrees, BSP trees, Fractal geometry methods, Visualization of datasets - visual representations for scalar, vector, and tensor fields

Different colour representations, transformation between colour models, halftoning

Rendering, Illumination models, Gouraud shading, Phong shading, transparency, shadows, image and texture mapping and synthesis, Radiosity lighting model

Raster animations, key frame systems, in-betweening, morphing, motion and pose interpolation and extrapolation

Graphical user interface and interactive input methods, interactive picture construction techniques, virtual reality environments.

Projects and Assignments: At least two assignments and one class project, assignments should include implementation of graphics algorithm using a programming language

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Steve Marschner and Peter Shirley: Fundamentals of Computer Graphics, CRC Press, 4th Edition, 2016
4. D.D. Hearn, M.P. Baker and W. Carithers: Computer Graphics with Open GL, 4th Edition, Pearson, 2014.
5. John F. Hughes, Andries van Dam, Morgan McGuire, David F. Sklar, James D. Foley: Computer Graphics: Principles and Practice (3rd Edition), Addison-Wesley Professional, 2013.

Computer Vision

(a) **Topics:** Machine vision systems, introduction to low, mid and high level vision, low and mid level image processing, edge detection, image segmentation, image and texture features

Camera geometry, object to image geometric transformations, orthographic and perspective view transformations, camera calibration

Binocular vision system, epipolar geometry, 3D scene reconstruction, recovering shape from stereo

Human vision structure, neurovisual model, scale space representation

Motion estimation and tracking, active contours, recovering shape from motion, video processing

Reflectance map and photometric stereo, surface reflectance model, recovering albedo and surface orientation, recovering shape from shading

Machine learning for computer vision, Classification models for vision, deep learning architectures for vision, Model based recognition system

Object recognition, recognition of arbitrary curved object sensed either by stereo or by range sensor, Recognition under occlusion, Aspect graph of an arbitrary 3D object viewed from different directions, Recognition of 3D objects based on 2D projections

Projects and Assignments: At least two assignments and one class project, assignments should include implementation of computer vision algortihm using a programming language

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 70%, Assignments 30%

(e) **References:**

1. David Forsyth and Jean Ponce: Computer Vision, A Modern Approach, Prentice Hall Pearson, 2015
2. Richard Szeliski: Computer Vision: Algorithms and Applications, Springer 2011
3. Richard Hartley and Andrew Zisserman: Multiple View Geometry in Computer Vision, Cambridge University Press, 2004
4. B.K.P. Horn: Robot Vision, MIT Press, Cambridge, 1986

Computing Systems Security I

(a) **Topics:** *Introduction to basic security services:* Confidentiality, integrity, availability, non-repudiation, privacy.

Anatomy of an Attack: Network Mapping using ICMP queries, TCP Pings, traceroutes, TCP and UDP port scanning, FTP bounce scanning, stack fingerprinting techniques, Vulnerability scanning, System and Network Penetration, Denial of Service.

Network Layer Protocols attacks and defence mechanisms: Hacking Exploits in ARP, IP4, IPv6, ICMP based DOS, ICMP covert Tunneling, Network Controls against flooding, Network Monitoring, SSL, IPSEC.

Transport Layer Protocols Attacks and Defence mechanisms: Covert TCP, TCP Syn flooding DOS, TCP Sequence Number Prediction attacks, TCP session hijacking, UDP Hacking Exploits, Network security controls for defense mechanism, OS hardening, kernel parameter tuning, DDOS & DDOS Mitigation, Stateful firewall, application firewalls, HIDS, NIDS and IPS.

Application Layer Protocol Attacks and Defense mechanisms: DNS spoofing attacks, DNS cache poisoning attacks, organization activity finger printing using DNS, SMTP vulnerability and Hacking Exploits, Mails relays, SMTP Security and Controls, HTTP hacking, Buffer Overflow Attacks, SQL Injection, Cross Side Scripting HTTP security and controls.

Malware detection and prevention

(b) **Prerequisites:** Operating Systems, Computer Networks, Discrete Mathematics

(c) **Hours:** Three lectures and one lab per week

(d) **Marks Distribution:** Theory 50%, Lab 50%

(e) **References:**

1. Ross Anderson: Security Engineering, 2nd ed., Wiley. Available online: `http://www.cl.cam.ac.uk/~rja14/book.html`.
2. C.P. Pfleeger, S.L. Pfleeger, J. Margulies: Security in Computing, 5th ed., Prentice Hall, 2015.
3. David Wheeler: Secure Programming HOWTO. Available online: `https://www.dwheeler.com/secure-programs/`.
4. Michal Zalewski: Browser Security Handbook, Michael Zalewski, Google. Available online: `https://code.google.com/archive/p/browsersec/wikis/Main.wiki`.
5. B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.
6. A. Menezes, P. C. Van Oorschot and S. A. Vanstone: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.

Computing Systems Security II

(a) **Topics:** Cellular networks, Access Technologies, GSM, CDMA, GPRS, 3G networks, Wireless LAN, WLAN security.

Operating Systems Security: (i) Access Control Fundamentals (ii) Generalized Security Architectures (iii) Analysis of security in Unix/Linux and problems with the design of its security architecture (iv) Analysis of security in Windows and problems with its security architecture (v) Security Kernels: SCOMP design and analysis, GEM-SOS design (vi) Difficulties with securing Commercial Operating Systems (Retrofitting Security) (vii) Security issues in Virtual Machine Systems (viii) Security issues in sandboxing designs: design and analysis of Android.

Database Security: (i) Introduction: Security issues faced by enterprises (ii) Security architecture (iii) Administration of users (iv) Profiles, password policies, privileges and roles (v) Database auditing

(b) **Prerequisites:** Computing Systems Security I, Discrete Mathematics

(c) **Hours:** Two lectures each of two hours duration

(d) **Marks Distribution:** Theory 70%, Assignments 30%

(e) **References:**

1. Ross Anderson: Security Engineering, 2nd ed., Wiley. Available online: `http://www.cl.cam.ac.uk/~rja14/book.html`.
2. C.P. Pfleeger, S.L. Pfleeger, J. Margulies: Security in Computing, 5th ed., Prentice Hall, 2015.
3. David Wheeler: Secure Programming HOWTO. Available online: `https://www.dwheeler.com/secure-programs/`.
4. Michal Zalewski: Browser Security Handbook, Michael Zalewski, Google. Available online: `https://code.google.com/archive/p/browsersec/wikis/Main.wiki`.
5. B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.
6. A. Menezes, P. C. Van Oorschot and S. A. Vanstone: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.

Cryptology-I

(a) **Topics:**

Classical ciphers and their cryptanalysis; Information Theoretic Security, one time pad; Stream ciphers; Block ciphers; Cryptanalysis of Block and Stream Ciphers; Formal models for block and stream ciphers: Pseudorandom generators, Pseudorandom functions and permutations; Symmetric key encryption: Notion of CPA and CCA security with examples; Cryptographic hash functions; Symmetric key authentication; Modern modes of operations: Authenticated Encryption, Tweakable Enciphering schemes.

Introduction to public key encryption; computational security and computational assumptions; The Diffie Hellman key exchange; The RSA, ElGamal, Rabin and Paillier encryption schemes; Digital Signatures; Introduction to Elliptic Curve Cryptosystems; Public key infrastructure.

(b) **Prerequisites:** Discrete Mathematics, Elements of Algebraic Structures

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Jonathan Katz, Yehuda Lindell: Introduction to Modern Cryptography, Chapman & Hall/CRC, 2007.
2. Douglas R. Stinson: Cryptography Theory and Practice, 3rd ed., Chapman & Hall/CRC, 2006.
3. Dan Boneh, Victor Shoup: A Graduate Course in Applied Cryptography, online draft available at `http://toc.cryptobook.us/`.
4. B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.
5. A. Menezes, P. C. Van Oorschot and S. A. Vanstone: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.

Cryptology-II

(a) **Topics:** *Theoretical construction of pseudorandom objects:* One way functions, pseudorandom generators, pseudorandom functions and pseudorandom permutations.

Secure Multiparty Computations: Zero knowledge proofs; Oblivious Transfer; Yao's two party protocol

Elliptic curves and bilinear pairings: The group of points on an elliptic curve; Elliptic curves over finite fields, Mongomery and Edwards curves; The curve 25519, the curve P256; Bilinear pairings; Signature schemes from bilinear pairings; Identity based encryption; Broadcast encryption.

Lattice Based Cryptology: Integer lattices; hard problems on lattices: SIS, LWE and ring LWE problems; Trapdoor sampling from lattices; signatures and encryption schemes from lattices

(b) **Prerequisites:** Cryptology I

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Oded Goldreich: Foundations of Cryptography, Vol 1, Cambridge University Press, 2001
2. Oded Goldreich: Foundations of Cryptography, Vol 2, Cambridge University Press, 2004
3. Dan Boneh, Victor Shoup: A Graduate Course in Applied Cryptography, online draft available at `http://toc.cryptobook.us/`.
4. Steven D. Galbraith: Mathematics of Public Key Cryptography, Cambridge University Press, 2012
5. Rafael Pass and Abi Shelat: A Course in Cryptography, Lecture notes. Available online: `https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf`
6. Daniele Micciancio, Shafi Goldwasser, Complexity of Lattice Problems: A Cryptographic Perspective, Kluwer, 2002.
7. Lawrence C. Washington, Elliptic Curves: Number Theory and Cryptography, Second Edition, CRC Press 2008.
8. S. Chatterjee, P. Sarkar: Identity-Based Encryption, Springer, 2011.

Cyber-Physical Systems

(a) **Topics:**

Cyber-Physical Systems (CPS) in the real world, Basic principles of design and validation of CPS, AUTomotive Open System Architecture (AutoSAR), Industrial Internet-of-things (IIoT) implications, Building Automation, Medical CPS

CPS - Platform components: CPS Hardware platforms - Processors, Sensors, Actuators, CPS Network Control Area Network (CAN), Automotive Ethernet, CPS Software stack Real Time Operating Systems (RTOS), Scheduling Real Time control tasks

Principles of Automated Control Design (basic control theory): Dynamical Systems and Stability, Controller Design Techniques, Stability Analysis: Common Lyapunov Function (CLF), Multiple Lyapunov Function (MLF), stability under slow switching, Performance under Packet drop and Noise

CPS implementation: From features to software components, Mapping software components to Electronic Control Units (ECU), CPS Performance Analysis - effect of scheduling, bus latency, sense and actuation faults on control performance, network congestion

Safety and Security Assurance of Cyber-Physical Systems: Advanced Automata based modelling and analysis, Timed and Hybrid Automata, Definition of trajectories, Zenoness, Formal Analysis, Flow-pipe construction, reachability analysis, Analysis of CPS Software, Weakest Pre-conditions, Bounded Model checking

Secure Deployment of CPS: Attack models, Secure Task mapping and Partitioning, State estimation for attack detection, Automotive Case Study

CPS Case studies and Tutorials

- Matlab toolboxes - Simulink, Stateflow
- Control, Bus and Network Scheduling using Truetime
- Hybrid Automata Modeling : Flowpipe construction using Flowstar, SpaceX and Phaver tools
- CPS Software Verification: Frama-C, CBMC
- Automotive and Avionics : Software controllers for Antilock Braking Systems (ABS), Adaptive Cruise Control (ACC), Lane Departure Warning, Suspension Control, Flight Control Systems
- Heathcare: Artificial Pancreas/Infusion Pump/Pacemaker
- Green Buildings : automated lighting, Air-Condition (AC) control

(b) **Prerequisites:** None.

(c) **Hours:** Four lectures per week including a tutorial.

(d) **Marks Distribution:** Theory 60% and Assignments 40%

(e) **References:**

1. Rajeev Alur, Principles of Cyber-Physical Systems, MIT Press 2015
2. Andre Platzer, Lecture Notes on Cyber-Physical Systems, available online
3. Edward Lee and Sanjit Seshia, Introduction to Embedded Systems A CyberPhysical Systems Approach

Digital Signal Processing

(a) **Topics:** *Introduction:* Applications of signal processing, elements of analog signal processing.

Discrete time signals and systems: Causal and stable systems, linear time invariant systems, difference equation representations, Fourier transform of sequences, transfer function.

Random signals: Stationary signals, autocorrelation function, power spectral density.

Sampling of continuous time signals: Frequency domain interpretation of sampling, reconstruction of band limited signals from samples.

The z-transform: Region of convergence, properties of z-transform, inverse z-transform, relation with other transforms.

Transfer function: Poles and zeroes, interpretation of causality and stability, frequency response for rational transfer functions, minimum phase and all-pass systems.

Transform analysis of discrete signals: Discrete Fourier series, discrete Fourier transform, relationships with Fourier transform of sequences.

Structures for discrete time systems: Block diagrams, signal flow graphs, direct, cascade and parallel forms, transposed forms, structures for FIR filters, lattice filters.

Effects of finite precision: Coefficient quantization, round-off noise, analysis of various structural forms, limit cycles in IIR filters.

Filter design: Filter specifications, design using analog filters, impulse invariance, bilinear transformation, frequency transformation of low-pass IIR filters, computer-aided design, FIR filter design by windowing.

Computation of DFT: Direct computation, FFT and other implementations, finite precision effects.

Applications of DFT: Fourier analysis of signals using DFT, DFT analysis of sinusoidal signals, spectral estimation, analysis of non-stationary signals.

Some advanced topics.

Practical exercises using MATLAB or other software.

(b) **Prerequisites:** Nil

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. A. V. Oppenheim and R. W. Schafer: Discrete Time Signal Processing, Prentice Hall, Englewood Cliffs, 1989.
2. S. K. Mitra: Digital Signal Processing, McGraw Hill, New York, 1998.
3. S. K. Mitra: Digital Signal Processing Laboratory Using MATLAB, McGraw Hill, New York, 1999.
4. A. Peled and B. Liu: Digital Signal Processing, Wiley, New York, 1976.

Discrete and Combinatorial Geometry

(a) **Topics:**

i. Planarity and Crossing Number
ii. Convexity and Lattices
iii. Extremal problems in Discrete Geometry
iv. Arrangement of Geometric and Algebraic Objects
v. Range Spaces and VC dimension
vi. Cuttings and Simplicial Partitions
vii. Incidence Geometry (Geometric Approach) and its Applications
viii. Algebraic Approach to Combinatorial Geometry
ix. Applications of Topological Methods
x. Additional Topics: Measure Concentration in High Dimensions and its Applications
xi. Additional Topics: Metric Embedding

(b) **Prerequisites:** Design and Analysis of Algorithms, Discrete Mathematics, Probability and Stochastic process

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. Jiri Matousek: Lectures on Discrete Geometry, Springer, 2001
2. Jiri Matousek: Geometric Discrepancy, Springer, 1999
3. by Bernard Chazelle: The Discrepancy Method, Cambridge University Press, 2002
5. Pankaj K. Agarwal and Janos Pach: Combinatorial Geometry, Wiley-Interscience, 1995
6. Larry Guth: Polynomial Methods in Combinatorics, American Mathematical Society,

Distributed Computing

(a) **Topics:** Distributed Computing Environments. Decentralized Computational Systems, Principles of Decentralized Computations in Networked Systems, Cooperative Tasks, Knowledge, Communication.

Information Diffusion, Topological considerations, Subnet Construction, Acyclic Computations.

Message Routing, Decentralized Control, Distributed Reset, Token Circulation, Distributed Set operations.

Symmetry Breaking, Leader Election.

Synchronous Computations, Computational Use of Time.

Computing in presence of Faults, Stable properties (Deadlock, termination etc.)

Continuous Computation (Virtual Time, Mutual Exclusion)

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Two lectures each of two hours duration

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. N. Santoro: Design and Analysis of Distributed Algorithms, Wiley 2006
2. A. Kshemkalyani, M. Singhal: Distributed Computing: Principles, Algorithms, and Systems, Cambridge University Press 2011
3. N. Lynch: Distributed Algorithms. Elsevier India 2005

Fault Tolerance and Testing

(a) **Topics:** Origin of fault-tolerant computing, reliability, maintainability, testability, dependability;

Faults, errors and fault models stuck-at, bridging, delay, physical, component level;

Design techniques for fault tolerance, triple-modular redundancies, m-out-of-n codes, check sums, cyclic codes, Berger codes, etc;

Fault tolerant design of VLSI circuits and systems; Concepts of t-diagnosability, self-checking, BIST, LSSD, etc; Testing and Design for testability, fault equivalence, dominance, checkpoints, test generation, D-algorithm, PODEM, FAN, Boolean difference, testability analysis, fault sampling, random pattern testability, testability-directed test generation, scan path, syndrome and parity testing, signature analysis; CMOS and PLA testing, delay fault testing, system-on-a chip testing, core testing; BDDs.

Formal verification: Introduction, Overview of Digital Design and Verification, Verilog HDL, Simulators, Test Scenarios and Coverage, Assertions, Binary Decision Diagrams (BDD), State Machines and Equivalence Checking, Model Checking, Bounded Model Checking, Counter Example Guided Abstraction Refinement; case studies.

(b) **Prerequisites:** Computer Organization.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 75%, Assignments 25%

(e) **References:**

1. D. K. Pradhan: Fault Tolerant Computing, Vols. 1and 2, Prentice Hall, Englewood Cliffs, 1986.
2. B. W. Johnson: Design and Analysis of Fault-Tolerant System, Addison-Wesley, Reading, Mass., 1989.
3. V. D. Agrawal and S. C. Seth: Tutorial: Test Generation for VLSI Chips, IEEE Computer Society Press, Los Alamos, California, 1988.
4. M. Abramovici et al: Digital Systems Testing and Testable Design, IEEE Press, Los Alamos, California, 1993.

Foundations of Data Science

(a) **Topics:**

i. Basics of Probability theory and Stochastic Process
ii. Basic of High-Dimensional Geometry
iii. Singular Value Decomposition and its applications in Computer Science
iv. Basics of Random Graphs, Random Walks and Markov Chains with applications
v. Basics of Learning Theory
vi. Algorithms for Big Data: Streaming, Sketching, and Sampling
vii. Clustering
viii. Wavelets

(b) **Prerequisites:** Design and Analysis of Algorithms and Probability and Stochastic process

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. Foundations of Data Science by A. Blum, J. Hopcroft and R. Kannan.
2. Sketching as a Tool for Numerical Linear Algebra by David P Woodruff
3. Foundations of Machine Learning by M. Mohri, A. Rostamizadeh and A. Talwalkar.
4. An Introduction to Computational Learning Theory by M. Kearns and U. Vazirani

Graph Algorithms

(a) **Topics:** *Shortest path (SP) problems:* Single source SP problem, SP tree, Ford's labelling method, labelling and scanning method, efficient scanning orders – topological order for acyclic networks, shortest first search for non-negative networks (Dijkstra), BFS search for general networks, correctness and analysis of the algorithms; All pair SP problem – Edmond-Karp method, Floyd's algorithm and its analysis.
Flows in Networks: Basic concepts, maxflow-mincut theorem, Ford and Fulkerson augmenting path method, integral flow theorem, maximum capacity augmentation, Edmond-Karp method, Dinic's method and its analysis, Malhotra-Kumar-Maheswari method and its analysis, Preflow-push method (Goldberg Tarjan) and its analysis; Better time bounds for simple networks.
Minimum cost flow: Minimum cost augmentation and its analysis.
Matching problems: Basic concepts, bipartite matching – Edmond's blossom shrinking algorithm and its analysis; Recent developments.
Planarity: Basic fact about planarity, polynomial time algorithm.
Graph isomorphism: Importance of the problem, backtrack algorithm and its complexity, isomorphism complete problems, polynomial time algorithm for planar graphs, group theoretic methods.
NP-hard optimization problems: Exponential algorithms for some hard problems – dynamic programming algorithm for TSP, recursive algorithm for maximum independent set problem; Review of NP-completeness of decision problems associated with TSP, bin packing, knapsack, maximum clique, maximum independent set, minimum vertex cover, scheduling with independent task, chromatic number etc; Formulation of the concept of NP-hard optimization problem, perfect graphs and polynomial time algorithms for hard problems on graphs, approximation algorithms and classification of NP-optimization problems with respect to approximability.

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. G. Ausiello et al: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximation Properties, Springer, Berlin, 1999.
2. T. H. Cormen, C. E. Leisarsen and R. L. Rivest: Introduction to Algorithms, Prentice Hall of India, New Delhi, 1998
3. M. R. Garey and D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completeness, W, H. Freeman, New York, 1979
4. D.S. Hochbaum (Ed.): Approximate Solution of NP-Hard Problems, PWS Pub.Co., New York, 1947.
5. D. Jungnickel: Graphs, Networks and Algorithms, Springer-Verlag, Berlin, 1999
6. K. Mehlhorn: Data Structures and Algorithms, Vol.2., Springer-Verlag, Berlin 1984.
7. M. Golumbic: Algorithmic Graph Theory and Perfect Graphs, Academic Press, New York, 1980.
8. C. M. Hoffman: Group Theoretic Algorithms and Graph Isomorphisms, Springer-Verlag, Berlin, 1982.
9. C. H. Papadimitriou and K. Stiglitz: Combinatorial Optimization: Algorithms and Complexity, Prentice Hall of India, New Delhi, 1997.
10. R. E. Tarjan: Data Structures and Network Algorithms, SIAM, Philadelphia, 1983
11. E. Horowitz and S. Sahni: Fundamentals of Computer Algorithms, Galgotia Pub, New Delhi, 1985.

Image Processing I

(a) **Topics:** *Introduction*: Image processing systems and its applications.

Image formation: Geometric and photometric models; Digitization - sampling, quantization; Image definition and its representation, neighborhood metrics.

Intensity transformations and spatial filtering: Enhancement, contrast stretching, histogram specification, local contrast enhancement; Smoothing, linear and order statistic filtering, sharpening, spatial convolution, Gaussian smoothing, DoG, LoG; Fuzzy techniques for intensity transformations and spatial filtering.

Segmentation: Pixel classification; Grey level thresholding, global/local thresholding; Optimum thresholding - Bayes analysis, Otsu method; Derivative based edge detection operators, edge detection/linking, Canny edge detector; Region growing, split/merge techniques, line detection, Hough transform.

Image/Object features extraction: Textural features - gray level co-occurrence matrix; Moments; Connected component analysis; Convex hull; Distance transform, medial axis transform, skeletonization/thinning, shape properties.

Registration : Monomodal/multimodal image registration; Global/local registration; Transform and similarity measures for registration; Intensity/pixel interpolation.

Color image processing: Fundamentals of different color models - RGB, CMY, HSI, YCbCr, Lab; False color; Pseudocolor; Enhancement; Segmentation.

Image databases: Attribute list, relational attributes, indexing, storage and retrieval.

(b) **Prerequisites:** None

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80% and Assignment 20%

(e) **References:**

1. R. C. Gonzalez and R. E. Woods: Digital Image Processing, Prentice Hall, 1993.
2. Maria Petrou and Panagiota Bosdogianni: Image Processing: The Fundamentals, John Wiley & Sons, Ltd, 1999.
3. B. Chanda and D. Dutta Majumder: Digital Image Processing and Analysis, Prentice Hall of India, New Delhi, 2000.
4. A. Jain: Fundamentals of Digital Image Processing, Prentice Hall of India, New Delhi, 1989.
5. K. R. Castleman: Digital Image Processing, Prentice Hall, Englewood Cliffs, 1996.
6. A. Rosenfeld and A. C. Kak; Digital Picture Processing, 2nd ed., (Vol. 1 and 2), Academic Press, New York, 1982.
7. A. Blake and A. Zisserman: Visual Reconstruction, MIT Press, Cambridge, 1987.
8. W. K. Pratt: Digital Image Processing, 2nd ed., John Wiley, New York, 1992.
9. A. N. Netravali and B. G. Haskell: Digital Pictures, 2nd ed., Plenum Press, 1995.
10. A. B. Watson: Digital Images and Human Vision, MIT Press, Cambridge, 1993

Image Processing II

(a) **Topics:** *2-D transformations of images and frequency filtering*: Frequency domain analysis, discrete Fourier transform, fast Fourier transform, convolution and correlation in frequency domain, frequency domain filtering; Walsh transform; Hadamard transform; Discrete cosine transform; Hotelling transform.

Enhancement/restoration: Edge-preserving smoothing, anisotropic diffusion; Least square restoration, constrained least-squares restoration, Wiener filter; Blind deconvolution; Superresolution.

Morphological image processing : Erosion, dilation, opening, closing, Hit-or-Miss transformation; Gray-scale morphology, area morphology; Watershade algorithm.

Segmentation : Model-based - facet model, active contour, semantic (saliency based) region grouping; Interactive segmentation - growcut, graphcut.

Image analysis : Pattern spectrum; Structural features - Fourier descriptor, polygonal approximation; Shape matching, template matching, shape metric, image understanding.

Multi-resolution image analysis : Spatial/frequency domain analysis, Gabor transform; Continuous wavelet analysis, dyadic wavelet, fast wavelet analysis, fast inverse wavelet analysis, 1D/2D wavelets; Wavelet packets.

Compression : Lossy/lossless compression, error criteria; Huffman coding, arithmetic coding, run-length coding, block transform coding, JBIG; transform domain compression, vector quantization, block truncation compression, JPEG, wavelet based compression.

Some applications (from the following but not restricted to): (i) Biomedical image processing; (ii) Document image processing; (iii) Fingerprint classification; (iv) Digital water-marking; (v) Image fusion; (vi) Image dehazing; (vii) Face detection; (viii) Face recognition; (ix) Content aware resizing; (x) Content based image retrieval.

(b) **Prerequisites:** Image Processing I

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80% and Assignment 20%

(e) **References:**

1. R. C. Gonzalez and R. E. Woods: Digital Image Processing, Prentice Hall, 1993.
2. Maria Petrou and Pedro Garcfa Sevilla: Image Processing: Dealing with Texture, John Wiley & Sons, Ltd, 2006.
3. B. Chanda and D. Dutta Majumder: Digital Image Processing and Analysis, Prentice Hall of India, New Delhi, 2000.
4. A. Jain: Fundamentals of Digital Image Processing, Prentice Hall of India, New Delhi, 1989.
5. K. R. Castleman: Digital Image Processing, Prentice Hall, Englewood Cliffs, 1996.
6. A. R. Rao: Taxonomy for Texture Description, Springer-Verlag, Berlin, 1990.
7. R. M. Haralick and L. G. Shapiro; Computer and Robot Vision, Vol. 1and 2, Addison-Wesley, Reading, Mass., 1992.
8. A. Rosenfeld and A. C. Kak; Digital Picture Processing, 2nd ed., (Vol. 1and 2), Academic Press, New York, 1982.
9. B. B. Chaudhuri and D. Dutta Majumder: Two-tone Image Processing and Recognition, Wiley-Eastern, New Delhi, 1993.
10. A. Blake and A. Zisserman: Visual Reconstruction, MIT Press, Cambridge, 1987.
11. W. K. Pratt: Digital Image Processing, 2nd ed., John Wiley, New York, 1992.
12. A. N. Netravali and B. G. Haskell: Digital Pictures, 2nd ed., Plenum Press, 1995.
13. K. Sayood: Data Compression, Morgan Kaufmann, San Mateo, 1986.
14. H. C. Andrews and B. R. Hunt: Digital Image Restoration, Prentice Hall, Englewood Cliffs, 1977.
15. M. Vetteerli and J. Kovacevic: Wavelet and Sub-Band Coding, Prentice Hall, EC, 1995.
16. A. B. Watson: Digital Images and Human Vision, MIT Press, Cambridge, 1993.
17. C. A. Glasbey and G. H. Horgen: Image Analysis for Biomedical Sciences, John Wiley, New York, 1995.
18. S. Khoshafian and A. B. Baker: Multimedia and Imaging Databases, Morgan Kaufmann, San Mateo, 1996.
19. S. K. Pal, A. Ghosh and M. K. Kundu: Soft Computing for Image Processing, Physica Verlag (Springer), Heidelberg, 1999.
20. M. Sonka, V. Hlavac and R. Boyle, Image Processing: Analysis and Machine Vision, PWS Pub. Co., London, 1998.

Information Retrieval

(a) **Topics:** The instructor may select only some of the following topics, and include other topics of current interest.

Introduction: overview of applications, brief history; text representation, indexing: tokenisation, stopword removal, stemming, phrase identification; index structures, index creation.

Models: Boolean retrieval, ranked retrieval, vector space model: term weighting; probabilistic models for IR; language modeling for IR: query likelihood, KL divergence, smoothing.

Evaluation: recall, precision, average precision, NDCG, other commonly used metrics; test collections, evaluation forums, sound experimental methods.

Query expansion: query expansion in the vector space model: relevance feedback, Rocchio's method, variations, pseudo relevance feedback; query expansion in the probabilistic model; relevance based language models and variations; automatic thesaurus generation; matrix decompositions; latent semantic analysis.

Web search: Web document preprocessing: parsing, segmentation, deduplication, shingling; crawling, focused crawling, metacrawlers; link analysis: hubs and authorities, Google PageRank; query auto completion; search log analysis; search result diversification; computational advertising.

Machine learning in IR: text categorisation: vector space methods, nearest neighbours, naive Bayes, support vector machines, feature selection; text clustering: agglomerative clustering, k-means, search result clustering; learning to rank.

1. *Other applications:* opinion mining, sentiment analysis; automatic text summarisation.

(b) **Prerequisites:** Probability, Linear algebra.

(c) **Hours:** Four lectures per week, one lab-session per week during the second half of the course.

(d) **Marks Distribution:** Theory 50%; project 50%.

(e) **References:**

(a) Christopher D. Manning, Prabhakar Raghavan and Hinrich Schtze: Introduction to Information Retrieval, Cambridge University Press, 2008.

(b) Stefan Büttcher, C.L.A. Clarke and G.V. Cormack: Information Retrieval Implementing and Evaluating Search Engines, MIT Press, 2010.

(c) W. Bruce Croft, D. Metzler, T. Strohman: Search Engines: Information Retrieval in Practice, Pearson, 2010.

(d) ChengXiang Zhai and Sean Massung: Text Data Management: A Practical Introduction to Information Retrieval and Text Mining, ACM and Morgan & Claypool Publishers, 2016.

(e) ChengXiang Zhai: Statistical Language Models for Information Retrieval A Critical Review, NOW Publishers, .

(f) Christopher Olston, Marc Najork: Web Crawling, NOW Publishers, .

(g) Fei Cai, Maarten de Rijke: A Survey of Query Auto Completion in Information Retrieval, NOW Publishers, .

(h) Fabrizio Silvestri: Mining Query Logs: Turning Search Usage Data into Knowledge, NOW Publishers, .

(i) R.L.T. Santos, Craig Macdonald, Iadh Ounis: Search Result Diversification, NOW Publishers, .

(j) Jun Wang, Weinan Zhang, Shuai Yuan: Display Advertising with Real-Time Bidding (RTB) and Behavioural Targeting, NOW Publishers, .

(k) Tie-Yan Liu: Learning to Rank for Information Retrieval, NOW Publishers, .

(l) Bo Pang, Lillian Lee: Opinion Mining and Sentiment Analysis, NOW Publishers, .

(m) Ani Nenkova, Kathleen McKeown: Automatic Summarization, NOW Publishers, .

Information Theory

(a) **Topics:** Introduction: Historical perspective; Entropy; Mutual Information; Chain rule; Relative entropy and its non-negativity

Compression: Asymtotic Equipartition Property(AEP); Markov Models; AEP for Markov Models; Krafts Inequality; Prefix Codes; Huffman Codes; Arithmetic Codes; Lempel-Ziv Codes

Communication: Communication over noisy channels; Channel capacity; Converse to the noisy channel coding theorem; Sphere packing view of the coding theorem; Polar codes; Gaussian channel; Information measures for continuous variables; Entropy maximization

Kolmogorov Complexity: Models of computation; Definition and examples of Kolmogorov Complexity; Kolmogorov Complexity and Entropy; Algorithmically Random and Incompressible sequences; Universal Probability; Minimum description length principle.

Applications to Statistics and Machine Learning

(b) **Prerequisites:** Probability and Stochastic Processes

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. R. B. Ash, Information Theory, Dover, 1990.
2. T. Berger, Rate Distortion Theory: A Mathematical Basis for Data Compression, Prenice Hall, 1971.
3. T. M. Cover and J. A. Thomas, Elements of Information Theory, John Wiley, 1991.
4. I. Csiszar and J. Korner, Infromation Theory: Coding Theorems for Discrete Memoryless Systems, Academic Press, 1981.
5. R. G. Gallager, Information Theory and Reliable Communication, Wiley, 1968.
6. Abbas El Gamal and Young-Han Kim, Network Information Theory, Cambridge University Press, 2012.

Learning Theory

(a) **Topics:**

i. General introduction

ii. Introduction to Probability theory and Stochastic process

iii. PAC model; Occam's razor

iv. Sample complexity: VC dimension, Sauer-Shelah Lemma, infinite hypothesis spaces, supervised learning, agnostic learning, lower bounds, and Rademacher complexity

v. Computational hardness results

vi. Online learning

vii. Boosting and margins theory

viii. Support-vector machines and kernels

ix. Mistake-bounded algorithms, halving algorithm and weighted majority algorithm

x. Learning and game theory

xi. Linear-threshold algorithms

xii. Maximum entropy modeling

xiii. Portfolio selection; Cover's algorithm

xiv. Introduction to Active learning

xv. Introduction to semi-supervised learning

xvi. Introduction to Distributed learning

xvii. Introduction to Differential privacy and Statistical query learning

(b) **Prerequisites:** Design and Analysis of Algorithms, Probability and Stochastic process

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. M. Mohri, A. Rostamizadeh and A. Talwalkar: Foundations of Machine Learning by, MIT Press.
2. M. Kearns and U. Vazirani: An Introduction to Computational Learning Theory, MIT Press.

Logic for Computer Science

(a) **Topics:** Syntax and semantics of first order logic; Proof procedures – Hilbert system, natural deduction and sequent calculus, resolution methods, soundness and completeness; Prenex normal form and skolemization; Compactness, Lowenheim Skolem theorem, Herbrand's theorem, undecidability and incompleteness; Peano and Presburger arithmetics, incompleteness of first order number theory. Introduction to Modal and Temporal Logic with applications.

(b) **Prerequisites:** Discrete Mathematics

(c) **Hours:** Two lectures each of two hours duration

(d) **Marks Distribution:** Theory 100%

(e) **References:**

1. C. L. Chang and R. C. T Lee: Symbolic Logic and Mechanical Theorem Proving, Academic Press, New York and London, 1973.

2. H. Enderton: A Mathematical Introduction to Logic, Academic Press, London, 1972.
3. M. Fitting: First-order Logic and Automated Theorem Proving, Springer, Berlin, 1990.
4. H. Gallier: Logic for Computer Science, John Wiley and Sons, New York, 1987.
5. G.E. Hughes and M.J. Cresswell: A New Introduction to Modal Logic Symbolic Logic, Routledge, 1996.
6. E. Mendelson: Introduction to Mathematical Logic, Van Northand, London, 1979.
7. A Nerode and R.A. Shore: Logic for Applications, Springer, Berlin, 1993.
8. V. Sperschneider and G. Antonio: Logic: A Foundation for Computer Science, Addison-Wesley, California, 1991.
9. I.S. Torsun: Foundations of Intelligent Knowledge-Based Systems, Academic Press, New York, 1995.
10. L.Zhongwan: Mathematical Logic for Computer Science, World Scientific, Singapore, 1989.

Machine Learning I

(a) **Topics:** *Review of basic Mathematical and Statistical concepts:* (i) Metric, Positive definite matrix, Eigen values and eigen vectors, mean, median, mode, variance, co-variance, correlation, dispersion matrix, binomial distribution, normal distribution, multi-variate normal distribution, basic concepts in probability theory such as Bayes theorem, Chebyshev's inequality, Laws of large numbers, Central limit theorem, (ii) Unbiased estimate, consistent estimate, maximum likelihood estimate.

Classification: (i) Bayes decision rule, examples, normal distribution cases, training and test sets, probability of misclassification, estimation of parameters for normal distribution, minimum distance classifier, standardization, normalization, Mahalanobis distance, Naive-Bayes rule, (ii) K-NN decision rule, its properties, (iii) Density estimation, (iv) Perceptron (linear separable case), MLP, (v) Assessment of classifiers

Clustering: Similarity measures, minimum within cluster distance criterion, K-means algorithm, Hierarchical clustering, Density based clustering, FCM, cluster validation.

Dimensionality reduction: (i) Feature selection: Different criterion functions, Algorithms, BBA (ii) Feature extraction: PCA (iii) LDA

Decision trees, Random forests

(b) **Prerequisites:** Probability and Stochastic processes, Linear Algebra.

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Tom M. Mitchell: Machine Learning , McGraw-Hill International Edition, 1997.
2. Richard O. Duda, Peter E. Hart and David G. Stork: Pattern Classification, Wiley 2000.
3. Trevor Hastie Robert Tibshirani and Jerome Friedman: The Elements of Statistical Learning (2nd Edition), Springer
4. Christopher M. Bishop: Pattern recognition and Machine Learning, Springer, 2006

Machine Learning II

(a) **Topics:**

Some methods of optimization like Genetic algorithms, Simulated Annealing

Hilbert Space

Kernels, KPCA

VC dimension, Linear SVMs

PAC Learning

Gram Matrix, Mercer's theorem, classification using kernels

Linear regression, multiple correlation coefficient,

Logistic regression

EM algorithm, mixture models

Ensemble learning: Bagging, boosting

Regression using kernel functions

Deep Learning, Recurrent NNs

Semi-supervised and active learning; reinforcement learning.

Markov Random fields, Hidden Markov models

Any latest topic

(b) **Prerequisites:** Machine Learning I

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Tom M. Mitchell: Machine Learning , McGraw-Hill International Edition, 1997.
2. Richard O. Duda, Peter E. Hart and David G. Stork: Pattern Classification, Wiley 2000.
3. Trevor Hastie Robert Tibshirani and Jerome Friedman: The Elements of Statistical Learning (2nd Edition), Springer
4. Christopher M. Bishop: Pattern recognition and Machine Learning, Springer, 2006

Mobile Computing

(a) **Topics:** *Introduction:* Overview of wireless and mobile systems; Basic cellular concepts and architecture; Design objectives and performance issues; Radio resource management; Radio interface; Radio propagation and path loss models; Channel interference and frequency reuse; Cell splitting; Channel assignment strategies; Overview of generations of cellular systems:- 1G to 5G.

Location and handoff management: Introduction to location management (HLR and VLR); Mobility models characterizing individual node movement (Random walk, Fluid flow, Markovian, Activity based); Mobility models characterizing the movement of groups of nodes (Reference point based group mobility model, Community based group mobility model); Static location management schemes (Always vs. Never update, Reporting Cells, Location Areas); Dynamic location management schemes (Time, Movement, Distance, Profile Based); Terminal Paging (Simultaneous paging, Sequencial paging); Location management and Mobile IP; Introduction to handoffs; Overview of handoff process; Factors affecting handoffs and performance evaluation metrics; Handoff strategies; Different types of handoffs (soft, hard, horizontal, vertical).

Wireless transmission fundamentals: Introduction to narrowband and wideband systems; Spread spectrum; Frequency hopping; Introduction to MIMO; MIMO Channel Capacity and diversity gain; Introduction to OFDM; MIMO-OFDM system; Multiple access control (FDMA, TDMA, CDMA, SDMA); Wireless local area network; Wireless personal area network (Bluetooth and zigbee).

Mobile Ad hoc networks: Characteristics and applications; Coverage and connectivity problems; Routing in MANETs.

Wireless sensor networks: Concepts, basic architecture, design objectives and applications; Sensing and communication range; Coverage and connectivity; Sensor placement; Data relaying and aggregation; Energy consumption; Clustering of sensors; Energy efficient Routing (LEACH).

Cognitive radio networks: Fixed and dynamic spectrum access; Direct and indirect spectrum sensing; Spectrum sharing; Interoperability and co-existence issues; Applications of cognitive radio networks.

D2D communications in 5G cellular networks: Introduction to D2D communications; High level requirements for 5G architecture; Introduction to the radio resource management, power control and mode selection problems; Millimeterwave communication in 5G.

Labs: Development and implementation of different network protocols using network simulators such as NS-3 and OMNET++.

(b) **Prerequisites:** Computer Networks

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Theodore Rappaport, "Wireless Communications: Principles and Practice", Pearson Education.
2. Jochen Schiller, Mobile Communications, Pearson Education.
3. Andrea Goldsmith, Wireless Communications, Cambridge University Press.
4. Ezio Biglieri, MIMO Wireless Communications, Cambridge University Press.
5. Ivan Stojmenovic, Handbook of Wireless Networking and Mobile Computing, Wiley.
6. James Cowling, "Dynamic Location Management in Heterogeneous Cellular Networks," MIT Thesis. `http://people.csail.mit.edu/cowling/hons/jcowling-dynamic-Nov04.pdf`
7. Travis Keshav, Location Management in Wireless Cellular Networks `https://www.cse.wustl.edu/~jain/cse574-06/ftp/cellular_location.pdf`
8. Fahd A. Batayneh, Location Management in Wireless Data Networks `https://www.cse.wustl.edu/~jain/cse574-06/ftp/wireless_location.pdf`
9. Gordon L. Stber, Principles of Mobile Communication, Springer.
10. Lingyang Song, Dusit Niyato, Zhu Han, and Ekram Hossain, Wireless Device-to- Device Communications and Networks, Cambridge University Press.

11. Ezio Biglieri, Andrea J. Goldsmith, Larry J. Greenstein, Narayan Mandayam and H. Vincent Poor, Principles of Cognitive Radio, Cambridge University Press.
12. Edgar H. Callaway, Jr. and Edgar H. Callaway, "Wireless Sensor Networks: Architectures and Protocols," CRC Press.
13. `https://www.nsnam.org/docs/manual/html/index.html`

Natural Language Processing

(a) **Topics:** Introduction to NLP and language engineering, Components of NLP systems; Basics of probability; language modelling, smoothing; Hidden Markov Model (HMM) and its use in POS tagging; EM algorithm, IBM models (Model 1 and 2) for machine translation; probabilistic CFG, constraint Grammar- bracketed corpus, tree banks; discussion of different NLP tools: chunker, NER tagger, stemmer, lemmatizer, word sense disambiguation (WSD), anaphora resolution, etc.; neural language processing: word embedding, use of word embeddings in designing NLP tools, Recurrent Neural Nets, GRU, LSTM, sequence-to-sequence learning; Social media text analysis, Noisy text analysis.

(b) **Prerequisites:** NIL

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 75%, Programming assignments 25%

(e) **References:**

1. Christopher D. Manning and Hinrich Schütze: Foundations of Statistical Natural Language Processing, MIT Press.
2. Daniel Jurafsky and James H. Martin: Speech and Language Processing, Prentice Hall; 2nd edition, 2008.
3. Ian Goodfellow and Yoshua Bengio and Aaron Courville: Deep Learning, MIT Press, 2016.

Neural Networks

(a) **Topics:** Inspiration and lessons from the brain, introduction to biological neural networks, Models of artificial neurons, threshold logic, binary neurons and pattern dichotomizers, perceptron: its learning rule and convergence.

Multilayered perceptron, learning algorithms, function approximation, generalization, regularization networks, Radial Basis Function (RBF) network and learning. VC-dimension, Structural Risk minimization, support vector machines (regression and classification).

Recurrent neural networks, simulated annealing, mean-field annealing, Boltzmann machine, restricted Boltzmann machine (RBM), and their learning. Temporal learning, backpropagation through time, temporal backpropagation, real-time recurrent learning (RTRL).

Self-organizing maps, Hebbian and competitive learning, learning vector quantization, principal component analysis networks.

Deep learning, deep neural networks, architectures, autoencoder, stacked autoencoder, denoising autoencoder, activation function, learning, contrastive divergence, deep belief network, Long Short term memory LSTM, Sequence modeling, word2vec.

Convolutional Neural network, architecture, activation function, learning, popular convolutional networks like AlexNnet / GoogleNet.

(b) **Prerequisites:** Design and Analysis of Algorithms, Probability and Stochastic process

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 70%, Projects 30%

(e) **References:**

1. Satish Kumar, Neural Networks: a classroom approach, Tata McGraw-Hill Education, 2004
2. I Goodfellow, Y. Bengio, A. Courville, Deep learning. Cambridge: MIT press; 2016.
3. C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995
4. Simon Haykin: Neural Networks and learning machines, 3rd ed, Pearson, 2009.
5. T. Kohonen: Self-Organization and Associative Memory, Springer Science & Business Media, 2012.
6. M. H. Hassoun, Fundamentals of artificial neural networks. MIT press, 1995
7. J. Hertz, A. Krogh, and R. G. Palmer: Introduction to the Theory of Neural Computation, Addison- Wesley, California, 1991.

Optimization Techniques

(a) **Topics:** *Linear Programming:* Theory of LP — geometric interpretation of LP; basic feasible solution; feasible region of LP, convexity and convex polyhedra; vertices of the convex polyhedron, linear independence and basic feasible solution; Algorithms for LP — a brief review of simplex and revised simplex algorithms, Bland's rule, polynomial time algorithms – ellipsoidal and interior point methods; Duality — duality of LP, weak duality and strong duality theorems; Farkas lemma; Applications of LP — some applications from graph theory, game theory, LP relaxation to be done.

Integer Programming: Integer and mixed integer programming problems, cutting planes and branch and bound algorithms, NP-completeness of integer programming and ILP, travelling salesman and other related problems.

Non-linear Programming: Quadratic programming, convex programming problems; Unconstrained and constrained optimization problems; Karush-Kuhn-Tucker-Lagrangian necessary and sufficient conditions, interior point methods, standard algorithms like gradient descent, steepest descent, Newton's method, etc., ideas of convergence of the methods;

Semidefinite Programming

(b) **Prerequisites:** Design and Analysis of Algorithms

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. R. J. Vanderbei: Linear Programming Foundations and Extensions, Kluwer Academic Publishers, Boston/London, 1997.
2. D. G. Luenberger and Y. Ye: Linear and Non-Linear Programming, Springer, 2010.
3. C. H. Papadimitriou and K. Steiglitz: Combinational Optimization, Prentice Hall, Englewood Cliffs, 1982.

4. R. Garfinkel and G. Nemhauser: Integer Programming, John Wiley, New York, 1976.
5. G. Nemhauser and L. Wolsey: Integer and Combinational Optimization, Wiley, New York, 1988.
6. D. Bertsekas: Non-Linear Programming. Athena Scientific, Belmont, Mass., 1995.
7. S. Nash and A. Sofer: Linear and Non-Linear Programming, McGraw Hill, New York, 1996.
8. F. Hillier and G. Liebermann: Introduction to Mathematical Programming, McGraw Hill, 1995.
9. K. G. Murty: Linear and Combinatorial Programming, John Wiley, New York, 1976.
10. M. Bazaraa, J. Jarvis and H. Sherali: Linear Programming and Network Flows, Wiley, New York, 1977.
11. W. I. Zangwill: Non-Linear Programming, Prentice Hall, New Jersey, 1969.
12. R. Fletcher: Practical Methods of Constrained Optimization, John Wiley, Chichester, 1981.
13. J. Matoušek and Bernd Gärtner: Understanding and Using Linear Programming, Springer, 2007.
14. S. Boyd and L. Vandenberghe: Convex Optimization, Cambridge University Press, 2009.
15. V. Chvátal: Linear Programming, W. H. Freeman & Co. Ltd., 1983.
(a) A. Schrijver: Theory of Linear and Integer Programming, Wiley, 1998.
16. G. M. Ziegler: Lectures on Polytopes, Springer, 2012.
17. A. Schrijver: Combinatorial Optimization (3 volume A, B and C), Springer, 2003.

Quantum Information Processing and Quantum Computations

(a) **Topics:** *Introduction to Hilbert space:* Linear space, Scalar product, Hilbert space, Self adjoint operator, Projection operator, Unitary operator.

Basic introduction to Quantum mechanics: (i) Postulates of quantum mechanics, Uncertainty principle, Complementary principle, Unitary Dynamics, Detail study of two-level system. (ii) Multipartite quantum system, Quantum entanglement, Schmidt decomposition, Non-unique decomposition of mixed state, Hugston-Jozsa-Wooters theorem, No-Cloning Theorem, Distinguishing non-orthogonal quantum states. (iii) General quantum operations, Kraus representation theorem, various Quantum gates.

Basic quantum information processing: (i) Quantum teleportation, (ii) Quantum dense coding, (iii) Remote state preparation, (iv) Quantum key distribution (Bennett-Brassard–1984 Protocol, Ekerts entanglement protocol)

Quantum computing: Basic physics of Quantum parallelism, Some basic quantum algorithm; Deutschs algorithm, Deutsch-Jozsa algorithm, Simons algorithm, Grovers search algorithm, Quantum Fourier Transform and Shors factoring algorithm.

Introduction to elementary Quantum error correcting codes

Preliminary introduction to Quantum Cryptanalysis on Classical Cryptosystems

(b) **Prerequisites:** Linear Algebra, Elements of Algebraic Structures

(c) **Hours:** Two lectures each of two hours duration

(d) **Marks Distribution:** Theory 80% and Assignment 20%

(e) **References:**

1. Quantum Computation and Quantum Information, Michael A. Nielsen and Isaac L. Chuang, Cambridge University Press, 2002.
2. An Introduction to Quantum. Computing, Phillip Kaye,. Raymond Laflamme, and Michele Mosca. Oxford U. Press, New York, 2007.
3. Presskil Lecture notes `http://www.theory.caltech.edu/~preskill/ph229/`.
4. Quantum Computer Science, N. David Mermin, Cambridge University Press 2007.

Randomized and Approximation Algorithms

(a) **Topics:** The course has two parts – Randomized and Approximation Algorithms. The instructor can choose from the broad list given against Randomized and Approximation Algorithms. The course can, if needed, start with a brief introduction to (i) NP completeness, strong NP completeness; (ii) Linear programs – strong and weak duality;

Randomized Algorithms: The syllabus consists of several tools from the theory of randomization and its application to several branches of computer science like graphs, geometry, discrepancy, metric embedding, streaming, random graphs, etc.

Tools: Linearity of expectations; moments and deviations, tail inequalities – Markov's inequality, Chebyshev's inequality, Chernoff and Hoeffding bounds; concentration of measure; Sampling techniques – (Vitter, Knuth, Reif-Vitter, reservoir sampling, D2-sampling); Martingales – tail inequalities, Azuma Hoeffding inequality, Talagrand's inequality, Kim-Vu theorem; Markov chains; Random walks; Poisson process, branching process; Monte Carlo methods; Pairwise independence; Probabilistic methods;

Topics: Applications (can be chosen from the following list):

(1) Computational Geometry – Randomized incremental construction; backward analysis; random sampling – VC dimension, epsilon-nets; convex polytopes; geometric data structures

(2) Streaming algorithms – estimating the number of distinct elements; estimating frequency moments; geometric streams and core-sets; metric stream and clustering; graph streams; proving lower bounds from communication complexity;

(3) Metric embedding and dimension reduction – Johnson-Lindenstrauss lemma, Noga's lower bound, Bourgain embedding, Bartal's result

(4) Discrepancy – Combinatorial discrepancy for set systems; VC dimension and discrepancy;

(5) Probabilistic methods – Linearity of expectation; alteration; second moment; Lovasz local lemma – existential and constructive proofs; derandomization techniques; expander graphs; random graphs

(6) Miscellaneous topics – Data structures; Hashing and its variants; Primality testing; approximate counting; graph algorithms; randomized rounding; etc.

Approximation Algorithms:

(1) Greedy algorithms and local search – k-center problem; TSP; minimum degree spanning tree;

(2) Rounding and Dynamic Programming – knapsack; bin-packing; scheduling jobs on identical parallel machines

(3) Deterministic rounding of linear programs – solving large linear programs in polynomial time via ellipsoid method; prize collecting Steiner tree; uncapacitated facility location

(4) Random Sampling and randomized rounding of linear programs – derandomization; linear and non-linear randomized rounding; integrality gap; MAX-CUT, MAX-SAT; prize collecting Steiner tree; uncapacitated facility location; integer multicommodity flows

(5) Semidefinite programming – introduction; randomized rounding in semidefinite programming; finding large cuts; approximating quadratic programs

(6) Primal Dual method – introduction; feedback vertex set; shortest s-t path; Lagrangean relaxation and k-median problem

(7) Cuts and metrics – multiway cut, multiple cut, balanced cut, probabilistic approximation of metrics by tree metrics; spreading metrics, tree metrics and linear arrangement

(8) Iterative rounding – generalized assignment problem, discrepancy based methods, etc.

(9) Geometric approximation algorithms – well separated pair decomposition; VC dimension, epsilon-net, epsilon sampling, discrepancy; random partition via shifting; Euclidean TSP; approximate nearest neighbor search; core-sets

(10) Hardness of approximation – approximation preserving reduction; use of PCP; unique games conjecture

(b) **Prerequisites:** Discrete Mathematics, Design and Analysis of Algorithms, Probability and Stochastic Processes

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Michael Mitzenmacher and Eli Upfal: Probability and Computing – Randomized Algorithms and Probabilistic Analysis, Cambridge University Press, 2005.
2. Rajeev Motwani and Prabhakar Raghavan: Randomized Algorithms, Cambridge University Press, 2004.
3. Noga Alon and Joel H. Spencer: The Probabilistic Method, Wiley, 2008.
4. Ketan Mulmuley: Computational Geometry - An Introduction through Randomized Algorithms, Prentice Hall, 1994.
5. Jiri Matousek: Geometric Discrepancy: An Illustrated Guide, Springer.
6. S. Muthukrishnan: Data Streams: Algorithms and Applications, Now Publishers, Foundations & Trends in Theoretical Computer Science.
7. B. Chazelle: The Discrepancy Method: Randomness and Computation, Cambridge University Press.
8. Vijay V. Vazirani: Approximation Algorithms, Springer.
9. David P. Williamson and David B. Shmoys: The Design of Approximation Algorithms, Cambridge University Press.
10. Sariel Har-Peled: Geometric Approximation Algoithms, American Mathematical Society.
11. Lap Chi Lau, R. Ravi and Mohit Singh: Iterative Methods in Combinatorial Optimization.

Specification and Verification of Programs

(a) **Topics:** *Modeling of Systems:* Modeling of concurrent systems, timed systems, hybrid systems and probabilistic systems.

Specification Languages: Linear time properties, Linear Temporal Logic (LTL), Computation Tree Logic (CTL), Timed Computation Tree Logic (TCTL), Probabilistic Computational Tree Logic (PCTL) and their variants.

Abstract Interpretation, Weakest Precondition, Floyd-Hoare Logic, Separation Logic; Shape Analysis

Techniques for verification: Explicit-State Model Checking, Symbolic Model Checking, Bounded Model Checking, Equivalence checking, Partial Order Reduction, Symbolic execution, Counterexample guided abstraction refinement, probabilistic model checking.

Program Testing: program testing basics, automatic test-case generation, directed testing.

Decision Diagrams, SAT Solvers, Satisfiability Modulo Theories (SMT) Solvers.

Software Tools: Popular formal methods tools such as Spin, NuSMV, SAL, UPPAAL, SpaceX, Prism, Z3 and CUDD.

(b) **Prerequisites:** Switching Theory and Logic Design

(c) **Hours:** Three lectures and one tutorial (hands-on) per week

(d) **Marks Distribution:** Theory 70% and Assignment 30%

(e) **References:**

1. C. Baier and J.-P. Katoen. Principles of Model Checking. The MIT Press, 2008.
2. E. M. Clarke, Jr., O. Grumberg, and D. A. Peled. Model Checking. MIT Press, 1999.
3. C. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli. Satisfiability modulo theories. In Armin Biere, Hans van Maaren, and Toby Walsh, editors, Handbook of Satisfiability, IOS Press, 2009.
4. Michael Huth and Mark Ryan, Logic in Computer Science: Modelling and Reasoning about Systems. Cambridge University Press

Statistical Computing

(a) **Topics:**

Random number generation & randomness tests.

Nonparametric density estimation: Histogram, Kernel, Nearest Neighbors Density estimates

EM & MM Algorithms

Nonparametric regression: Kernel, Splines, Nearest Neighbors, Trees

Classification: Nearest neighbor classifiers, KDA, Tree, random forests

Markov Chains and Monte Carlo Methods

(b) **Pre-requisite:** Statistics, Probability and Stochastic processes.

(c) **Hours:** Four lectures per week including a tutorial.

(d) **Marks Distribution:** Theory 70% and Assignments 30%

(e) **References:**

1. S. M. Ross: Simulation, Second edition, Academic Press, 2012.
2. L. Devroye: Non-uniform Random Variate Generation, Springer, 2013.
3. R. A. Thisted: Elements of Statistical Computing, Routledge, 2017.
4. L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone: Classification and Regression Trees, Chapman and Hall/CRC, 1984
5. M. P. Wand and M. C. Jones: Kernel smoothing, Chapman and Hall/CRC, 1995.
6. D. W. Scott: Multivariate Density Estimation: Theory, Practice, and Visualization, Wiley, 2015.
7. R. O. Duda P. E. Hart D. G. Stork: Pattern Classification, Wiley-Interscience, 2000.
8. D. Kundu and A. Basu: Statistical Computing.
9. G. McLachlan and T. Krishnan: The EM Algorithm and Extensions, Wiley-Interscience, 2008.
10. K. Lange: MM Optimization Algorithms, SIAM-Society for Industrial and Applied Mathematics, 2016.
11. C. de Boor: A Practical Guide to Splines, Springer, 2001.
12. T. Hastie, R. Tibshirani and J. Friedman: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer 2016.
13. C. Robert and G. Casella: Monte Carlo Statistical Methods, Springer, 2005.
14. W.R. Gilks, S. Richardson and D. J. Spiegelhater: Markov Chain Monte Carlo in Practice, Chapman and Hall/CRC;, 1996.

Topics in Privacy

(a) **Topics:** *Cryptographic foundations:* Homomorphic Encryption, group signatures, blind signatures, anonymous credential management, commitment schemes, zero-knowledge proofs, proof of knowledge, SNARK, oblivious transfer, secure multiparty computation, Oblivious RAM, private set intersections, private information retrieval.

Perturbation, K-anonymity, L-diversity

Differential privacy

De-anonymization techniques

Privacy preserving analytics

Applications: Mixnets, Onion Routing (TOR), e-cash, e-voting, location privacy, profiling, Web Privacy (Online tracking and advertising), Bitcoin, Zerocash etc.

Privacy for outsourced data

Privacy risk analysis

Ethical Aspects of privacy: Privacy compliance, GDPR, HIPAA etc.

(b) **Prerequisites:** None

(c) **Hours:** Four lectures per week

(d) **Marks Distribution:** Theory 80%, Assignments 20%

(e) **References:**

1. Cynthia Dwork, Aaron Roth: The Algorithmic Foundations of Differential Privacy (Foundations and Trends in Theoretical Computer Science).
2. Jonathan Katz and Yehuda Lindell: Introduction to Modern Crypography, Second Edition, CRC Press.
3. Rafael Pass and Abi Shelat, A Course in Cryptography, Lecture notes. Available online: https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf