# INDIAN STATISTICAL INSTITUTE

# Students' Brochure

## PART II

Master of Technology (M.Tech.) in Computer Science

(Effective from Academic Year 2020-21)

(See PART I for general information, rules and regulations)

The Headquarters
203 BARRACKPORE TRUNK ROAD
KOLKATA 700108

INDIAN STATISTICAL INSTITUTE

Master of Technology (M.Tech.) in Computer Science

# Contents

The structure of the M.Tech. in Computer Science programme differs from that of the other degree programmes mentioned in the brochure on general information, rules and regulations. It also has certain variations in the rules that are spelled out in Section 3.

# 1 Eligibility, Admission Test and Stream

## 1.1 Eligibility

In order to be eligible for admission to this programme, an applicant must have one of the following:

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

## 1.2 Admission Test

The admission test has two major components: two written tests conducted on the same day and an interview at a later date. The two written tests usually consist of

**1.** a multiple choice type test on Mathematics at the B.Sc. (pass) level,

**2.** a subjective test having two groups, in which the candidate has to answer questions from any one group but not both:

Group A: Mathematics at the B.Sc. (pass) level

Group B: Computer Science at the B.E./B.Tech. level.

*The specific syllabus of the admission test is declared in due time after the admission test is advertised.*

Based on the performance in the written tests, a candidate is called for an interview. If a candidate fails to appear in the interview, his/her candidature is cancelled.

A candidate may avail of a waiver for the written tests if she/he has a valid GATE score above a threshold announced prior to the date of the written tests. The threshold is to be decided by the admission committee each year. Such candidates have to appear for the interview directly when intimated.

The final selection is based on the performance of the candidates in the written tests and the interview. The detailed policy of selection is announced on the admission webpage.

### 1.3 Stream

Every student who is admitted to the programme is categorised as either a CS-stream student or a non-CS-stream student based on the following criteria.

*CS-stream*: The student has

- either answered questions from Group B (Computer Science) of the subjective written test,
- or has a valid GATE score (above a threshold as decided by the admission committee) in Computer Science and Information Technology and has availed of the waiver from the written tests.

*Non-CS stream*: All other students.

The stream of each student is mentioned in the final selection list by the M.Tech.(CS) admission committee. The courses in this programme are categorised as compulsory, formative and elective. The curricula for the two streams differ in terms of the number of courses in each category a student has to pass, and each student is required to follow the curriculum specified for his/her stream.

## 2 Structure of the Programme

### 2.1 Structure for CS stream

A student of the CS stream has to pass the courses listed below. The requirements of *Compulsory* and *Pool A Formative* courses for the CS stream students, as mentioned in the brochure, have to be completed within the first year of the programme[1].

1. **Compulsory half semester non-credit course:**

   - Introduction to Programming

   This requirement may be waived if a student passes a programming test designed by the Mentor Committee, and conducted within the first two weeks of the first semester.

2. **Two compulsory courses** for the CS stream:

   - Design and Analysis of Algorithms
   - Discrete Mathematics

[1] with effect from Academic Year 2022-23, as decided in the $77^{th}$ meeting of the Academic Council

3. **Five Formative Courses** from the list of formative courses for the CS stream [2]:

| **Pool A** | **Pool B** |
|---|---|
| • Probability and Stochastic Processes<br>• Statistical Methods<br>• Linear Algebra<br>• Elements of Algebraic Structures | • Data and File Structures<br>• Automata Theory, Languages and Computation<br>• Operating Systems<br>• Database Management Systems<br>• Compiler Construction<br>• Computer Networks<br>• Principles of Programming Languages<br>• Computing Laboratory<br>• Computer Architecture |

   At least two formative courses must be from Pool A.

4. **Eight elective courses** from the list of elective courses

5. **Dissertation** (equivalent to three courses)

6. **Minor Project** (*in lieu* of two courses) Optional – see Section 3.5 for eligibility.

## 2.2 Structure for non-CS stream

A student of the non-CS stream has to pass the courses listed below. The requirements of *Compulsory* and *Pool A Formative* courses for the non-CS stream students (both with and without a master's degree in Mathematics/Statistics), as mentioned in the brochure, have to be completed within the first year of the programme[3].

1. **Compulsory half semester non-credit course:**

   - Introduction to Programming

   This requirement may be waived if a student passes a programming test designed by the Mentor Committee, and conducted within the first two weeks of the first semester.

[2] The course on *Data and File Structures* has been included in Pool B for the CS-stream with effect from Academic Year 2022-23, as decided in the $77^{th}$ meeting of the Academic Council.

[3] with effect from Academic Year 2022-23, as decided in the $77^{th}$ meeting of the Academic Council.

2. **Five compulsory courses** for the non-CS stream:

   - Data and File Structures
   - Design and Analysis of Algorithms
   - Discrete Mathematics
   - Computing Laboratory
   - Operating Systems

3. **Four Formative Courses** from the list of formative courses for the non-CS stream:

   **Pool A**

   - Probability and Stochastic Processes
   - Statistical Methods
   - Linear Algebra
   - Elements of 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 are applicable for choosing the formative courses.

   - **For students with a master's degree in Mathematics/Statistics:**
     at least three courses must be from Pool B.
   - **For all other students in non-CS stream:**
     at least two courses must be 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).

## 2.3 Semester-wise layout of the compulsory and formative courses

All compulsory and formative courses are offered at least once in every academic year. The distribution of those subjects according to odd and even semesters is given below. There may be some variation in the semester-wise allocation of courses from time to time.

### 2.3.1 Odd semester (the first semester of an academic year)

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

### 2.3.2 Even semester (the second semester of an academic year)

| Subject | Course Type | |
|---|---|---|
| | CS-stream | non-CS stream |
| Design and Analysis of Algorithms | – | Compulsory |
| Statistical Methods | 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 |

## 2.4 Elective courses

Elective courses are classified into four tracks, namely Theory, Systems, Cryptology and Security, Data Science. An elective course may belong to more than one track. A student who passes a certain number of elective courses in a specific track can obtain a specialisation in that track (details regarding specialisation are noted later) which will be mentioned in the marksheet.

The list of elective courses along with tracks are given in the following table.

| Course | Theory | Data Sc. | Crypto & Sec. | Systems |
|---|---|---|---|---|
| Advanced Operating Systems | | | | ✓ |
| Advanced Logic and Automata Theory | ✓ | | | |
| Algorithms for Big Data | ✓ | ✓ | | |
| Algorithms for Electronic Design Automation | | | | ✓ |
| Coding Theory | ✓ | | ✓ | |
| Computational Algebra and Number Theory | ✓ | | ✓ | |
| Computational Complexity | ✓ | | ✓ | |
| Computational Finance | ✓ | ✓ | | |
| Computational Game Theory | ✓ | ✓ | | |
| Computational Geometry | ✓ | | | |
| Computational Molecular Biology and Bioinformatics | | ✓ | | |
| Computational Topology | ✓ | ✓ | | |
| Computing Systems Security I | | | ✓ | ✓ |
| Computing Systems Security II | | | ✓ | ✓ |
| Computer Graphics | | ✓ | | ✓ |
| Computer Vision | | ✓ | | ✓ |
| Cryptology I | ✓ | | ✓ | |
| Cryptology II | ✓ | | ✓ | |
| Cyber-Physical Systems | | | | ✓ |
| Digital Signal Processing | | ✓ | | ✓ |
| Discrete and Combinatorial Geometry | ✓ | | | |
| Distributed Computing | ✓ | | | ✓ |
| Fault Tolerance and Testing | | | | ✓ |
| Formal Verification of Machine Learning Models | ✓ | ✓ | | |
| Graph Algorithms | ✓ | | | |
| Image Processing I | | ✓ | | ✓ |
| Image Processing II | | ✓ | | ✓ |
| Information Retrieval | | ✓ | | |
| Information Theory | ✓ | ✓ | ✓ | |
| Introduction to Cognitive Science | | ✓ | | |
| Learning Theory | ✓ | ✓ | ✓ | |
| Logic for Computer Science | ✓ | | | |
| Machine Learning I | ✓ | ✓ | | |
| Machine Learning II | ✓ | ✓ | | |
| Mobile Computing | | | | ✓ |
| Natural Language Processing | | ✓ | | |
| Neural Networks | | ✓ | | ✓ |
| Optimization Techniques | ✓ | ✓ | | ✓ |
| Quantum Computation | ✓ | | ✓ | ✓ |
| Quantum Information Theory | ✓ | | ✓ | |
| Randomized and Approximation Algorithms | ✓ | | ✓ | |
| Specification and Verification of Programs | | | | ✓ |
| Statistical Computing | | ✓ | | |
| Topics in Privacy | | | ✓ | |

# 3 Information/rules specific to M.Tech. in CS

The information given in this section supplements, and occasionally supersedes (but is never superseded by) the information given in the general brochure for all non-JRF degree programmes. The sections of the general brochure on promotions and repeating a year do not apply to this degree programme.

## 3.1 Duration of the programme

The expected time for completion of the programme is two years. A student may take upto a maximum of three years for completion. However, after completion of the second year, a student will not be eligible for stipends, contingency grants or hostel facilities.

## 3.2 Waiver of 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 waiver of class attendance, (s)he is required to seek permission from the Mentor Committee.
- The teacher of the course and the Dean's Office need 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 the *Computing Laboratory* course.

## 3.3 Registering for a course

A student has to register for at least four courses in the first semester, at least nine courses in the first two semesters and at least fourteen courses (or at least twelve courses and a minor project for students in the CS-stream) in the first three semesters. Within 2 weeks of the start of a semester, a student will have to inform the Dean's Office through the Mentor Committee of the courses that (s)he is registering for. A student cannot opt out of a course after these 2 weeks. The scores obtained in all the courses thus registered within

2 weeks into the semester, will be recorded in the marksheet. Even after passing a course, if a student opts for any of these courses again in subsequent semester(s) and the Mentor Committee approves, the highest aggregate obtained across all attempts will be recorded[4].

## 3.4 Dissertation

A student is required to work for a dissertation on a topic assigned/approved by the Project and Dissertation Committee under the supervision of a suitable ISI faculty member.

The work for a dissertation should be substantial and related to some important problem in an area of Computer Science and/or its applications. It should have notable theoretical or practical significance. A critical review of recent advances in an area of Computer Science and/or its applications with some contributions by the student is also acceptable as a dissertation.

*Duration*: The work for the dissertation is two-semester long, typically commencing at the beginning of the third semester. It is to be completed along with all other courses of the fourth semester. A student who opts for finishing the degree programme in more than two years but within three years, as per guidelines in the brochure, can commence her/his dissertation at the beginning of the fifth semester and complete it by the end of the following semester. The assignment of dissertation topics and the evaluation of students' work will be coordinated by a Project and Dissertation Committee that is to be formed by the Dean's Office in consultation with the Mentor Committee before the students of that batch are promoted to their second year.

The selection of the supervisor for the dissertation of each student should be done within the first four weeks of the semester in which the student starts her/his dissertation.

*Evaluation*: The interim progress of the dissertation work will be evaluated at the end of the semester in which it was commenced, as either Satisfactory or Unsatisfactory. If the progress is found to be Unsatisfactory, the Project and Dissertation Committee will, in consultation with the supervisor(s), allow the student extra time (not exceeding a month) to make up. If the progress of the student is still found Unsatisfactory, the student will have to start the dissertation afresh in the beginning of the following semester (i.e., the fourth semester) (could be with a different supervisor) and continue for two semesters (i.e., the fourth and fifth semesters). **It is to be noted that there will be no back paper for the dissertation.**

[4]inserted with effect from the 2022-23 academic year as decided in the 77th meeting of the Academic Council

The final dissertation should be submitted at the end of two semesters since its commencement as indicated in the academic calendar (typically by the middle of July of the standard year of completion of the batch). The dissertation will be evaluated out of 300 marks by a subcommittee formed by the Project and Dissertation Committee. The subcommittee will consist of relevant ISI faculty, the supervisor(s) and one or more external expert(s). The student has to defend his/her dissertation in an open seminar.

If the final dissertation is not submitted within the stipulated deadline, or if the subcommittee, as mentioned earlier, evaluates the student to have failed in the dissertation, then the dissertation will be evaluated at the end of the next semester (i.e., the fifth semester). Thus, a student who either had her/his interim progress unsatisfactory or failed the final dissertation evaluation, will have her/his dissertation re-evaluated at the end of the fifth semester.

If a student undertakes her/his dissertation during the fifth and sixth semesters and fails in the interim or the final evaluation organized by the Project and Dissertation Committee, then the Dean and the Mentor Committee will decide on a case-by-case basis whether the student is not to be awarded the degree at all, or to be given one last chance to finish the dissertation if the student has strong reasons for the failure. In these exceptional cases, while continuing to work on the dissertation for another semester, the student will be enrolled as an ISI student but will not be eligible for stipend and hostel facilities. The allowable exceptions are only for valid medical grounds, to be decided upon by the Dean in consultation with the concerned mentor committee.

*Joint supervision*: Joint supervision of a dissertation is possible, with permission from the Mentor Committee. In such a case, the student is allowed to spend considerable time outside the institute, provided his/her course requirements are fulfilled. The primary supervisor of a jointly supervised dissertation needs to be an ISI faculty.

## 3.5 Minor project

The duration of a minor project is one semester and can be opted for in either the third or the fourth semester. It is evaluated out of 200 marks.

A student is eligible to undertake a minor project only if (s)he satisfies the following:

- has passed at least ten courses in the first two semesters,
- the aggregate score of the best ten courses taken in the first two semesters is at least 75%.

An eligible student who does not opt for a minor project, as well as a student who is not eligible for a minor project, has to pass two additional courses from the list of formative or elective courses.

A student who opts for the minor project has to decide on a topic for the project within three weeks from the start of the third or fourth semester. The topic has to be approved by the Project and Dissertation Committee, and should be significantly different from that of the dissertation, as judged by the Project and Dissertation Committee. The Project and Dissertation Committee will also be responsible for the evaluation of the project. The rules for evaluation of the minor projects will be the same as those for the dissertation (see Section 3.4 above).

## 3.6 Internship/industrial training

There would be a mandatory 12 weeks gap between the first and the second year in the academic calendar. During the gap period from the end of the second semester to the start of the third semester, students are allowed to pursue internship/industrial training outside the institute in research institutes or public/private sector organizations in India or abroad. However, internship is not mandatory.

A student who undergoes internship/industrial training somewhere in India during the training period may receive his/her usual monthly stipend/remuneration/emoluments either from the Institute (ISI) or from the host organization at his/her own discretion with appropriate permission from the Dean. The students who are placed outside Kolkata for training will be reimbursed sleeper class to and fro train fare from Kolkata to the place of internship unless the host organization provides travel assistance.

Training may also be arranged at a research/R&D organization abroad. In case a student undergoes practical training at such a place abroad, the Institute (ISI) will not provide any financial support including the monthly stipend for that period.

## 3.7 Specialisation

Among the eight elective courses, if a student passes at least five elective courses from a specific track and does his/her dissertation in a topic which falls under that track, (s)he graduates with a specialisation. The track to which a dissertation belongs is to be decided by the Project and Dissertation Committee during or before the mid-term evaluation.

A student is eligible to obtain a **double specialisation** if (s)he fulfils the following:

- Passes at least 10 elective courses 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.
- Obtains passing marks in a minor project.
- The minor project and the dissertation are in two different tracks for which (s)he wishes to obtain the specialisation.

## 3.8 Final result

Upon passing the minimum number of courses, minor project (if applicable) and dissertation as described in Sections 2.1 and 2.2, the final percentage obtained by a student is to be computed as follows.

- For a student who does not opt for a specialisation, the final score out of 2000 marks is the sum of the best seventeen scores among the courses passed as per requirements[5], and the dissertation. If the student has done a minor project, then her/his score in the minor project may be counted in lieu of two elective or formative courses.
- For a student who opts for a single specialisation, the final score out of 2000 marks is the sum of the best seventeen scores among the courses passed as per requirements[5], and the dissertation. If the student has done a minor project, then her/his score in the minor project may be counted in lieu of two elective or formative courses. The seventeen courses should include at least five elective courses from the track in which the student desires to obtain the specialisation.
- For a student who opts for a double specialisation, the final score out of 2200 marks is the sum of the best seventeen scores among the courses passed as per requirements[5], the minor project and the dissertation. The seventeen courses should include at least ten elective courses with at least five from each of the two tracks in which the student opts for the specialisations.

In all cases, the scores of all the courses a student has registered for, as per rules in Section 3.3, will be reflected in the final mark sheet. The division indicated in the final mark sheet will be determined as per the rules described in the general brochure.

[5]A student who is required to pass $N_c$ compulsory courses, $N_f$ formative courses and $N_e$ elective courses, may, with permission from the Mentor Committee, repeat one or more of the compulsory courses, take more than $N_f$ formative courses, or more than $N_e$ elective courses. In such cases, the final score will be computed using the *best* score obtained in each compulsory course, the best $N_f$ scores obtained in the formative courses, and the best $N_e$ scores obtained in the elective courses. Please also see Section 3.3.

## 3.9 Stipend

On admission, each student will receive the institute specified stipend, subject to the rules described in the general brochure for all non-JRF degree courses, with the following modification in respect of 'Performance in coursework' criterion. A student is eligible for a full stipend

- in the first semester, only if (s)he registers for at least four courses;
- in the second semester, only if (s)he obtains at least 45% in each of at least four courses in the first semester with an average score of 60% in the best four courses;
- in the third semester, only if (s)he obtains at least 45% in each of at least nine courses in the first two semesters with an average score of 60% in the best nine courses;
- in the fourth semester, only if (s)he obtains at least 45% in each of at least fourteen courses (or twelve courses and a minor project) in the first three semesters with an average score of 60% in the best fourteen courses. Additionally, in the mid-term evaluation for the dissertation, her/his progress must be Satisfactory.

Further, a student who is ineligible for a full stipend in the second, third or fourth semester, may be eligible for a half stipend in that semester (i.e., second, third or fourth semester) if (s)he gets at least 45% score in each of the minimum number of individual courses as per the schedule listed above.

## 3.10 Mentor Committee

A group of students entering the M. Tech. (CS) programme in a particular year is treated as a batch. The Mentor Committee for each batch of students is announced at the time of their admission and lasts till all the students leave the programme. This Committee advises the students on their choice of courses (including opting for a course more than once even after passing it) and specialisations, and as a matter of fact any other problem. Students should approach the Mentor Committee if they face any problem. The Chairperson acts as the Class Teacher of the students in a particular batch. All communications on academic matters with other authorities or statutory bodies of the institute by a student should be routed through the Class Teacher. The choice of courses in a semester by a student needs to be approved by the members of the Mentor Committee. The students are required to submit the final course choice list to the Dean's office, signed by the members of the Mentor Committee for her/his batch, within 2 weeks from the start of the semester [6].

[6] inserted with effect from the 2022-23 academic year as decided in the 77th meeting of the Academic Council

## 3.11 Teachers' Committee

The Teachers' Committee in a particular semester consists of all the teachers of the courses (including regular courses and Lab courses) opted for by all students (across all years) studying M. Tech. (CS) in that semester. The Chairperson and Convener of the existing Mentor Committees will be invited members of the Teachers' Committee. The Dean of Studies is the Chair of the Teachers' Committee. Every decision related to the academic performance of a student is taken after deliberation in the relevant Teachers' Committee. A decision can be taken by any higher authority/statutory body only after the recommendations of the relevant Teachers' Committee have been taken into cognizance.

## 3.12 Project and Dissertation Committee

This committee formed for a batch before the start of its second year, is responsible for the following tasks:

- Pool a list of projects and dissertation topics from the relevant faculty members and circulate the list among the students of that batch. However, the students are free to choose a topic outside the list, provided a faculty member agrees to supervise such a project.
- Help a student with finding a Project/Dissertation topic and supervisor, if needed.
- Ascertain for a student opting for both Minor project and Dissertation, that that the two problems are different. An approval of the committee is mandatory before a student is assigned a Minor Project.
- Ascertain for a student who is opting for specialisation in a track, that the topic of the Dissertation is in the area of that track.
- Ascertain for a student who is opting for double specialisation, that the topic of the Minor Project is in the area of the second track of specialisation.
- Conduct evaluation of the minor projects and dissertation at appropriate times.

## 3.13 Prizes

At the end of the four semesters of a particular batch, the Mentor Committee may nominate a few students for their outstanding academic performances during this period, for awards of certain amount of cash as prize money. There will be no consideration of prize for semester-specific performance.

# 4 Detailed Syllabi of the Courses

The courses are categorised as compulsory, formative and elective. Each course has a detailed syllabus, a few prerequisites, the number of hours of class needed, marks distribution, and names of reference books.

**Prerequisites.** Many courses have a few suggested prerequisite(s). The prerequisite courses have to be passed either at ISI or in the undergraduate / postgraduate degrees obtained before joining ISI. The students need to confirm with the concerned teacher if her/his prerequisite courses satisfy the demands of the particular course being taught at ISI. It is the responsibility of the students to verify from the concerned teacher that they satisfy the prerequisites.

**Marks distribution.** The marks distribution given against each course is suggestive for the teacher. A teacher of a course will announce within two weeks of the start of the course the explicit marks distribution for the course. The marks distribution for each course has two components.

**Examination:** This includes mid-semester and end-semester examinations. Except for the *Computing Laboratory* course that has no mid-semester and end semester examinations, all other courses have at least 50% weightage for the end-semester examination.

**Lab/assignment:** This component includes the internal assessment marks and comprises programming assignments, written assignments, class tests, quizzes, etc.

## 4.1 Compulsory and formative courses

**Automata Theory, Languages and Computation**

(a) *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:* Computable functions, primitive and recursive functions, universality, halting problem, recursive and recursively enumerable sets, parameter theorem, diagonalisation, reducibility, Rice's Theorem and its applications. Turing machines and variants; 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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) *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, variables 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) **Prerequisite(s):** Computer organisation; Automata, languages and computation.

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

(d) **Marks Distribution:**

**Examination:** 60%

**Laboratory/assignment:** 40% (inclusive of 10% for assignments and 30% for projects)

(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., 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) 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:**

**Examination:** 60%

**Laboratory/assignment:** 40% (inclusive of projects and 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) *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:**

**Examination:** 70%

**Laboratory/assignment:** 30% (inclusive of a mandatory laboratory 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) *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, Booth's 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:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(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) This laboratory course has to be run in coordination with the Data and File Structures course. The assignments are to be designed based on the coverage in that 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 are not limited to:

**Programming techniques:** Problem solving techniques like divide-and-conquer, dynamic programming, recursion, etc. are to be covered.

**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.
*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 re-

gression; (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) **Prerequisites:** Data and File Structures (can be taken concurrently)

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

(d) **Marks Distribution:**

**Examination:** –

**Laboratory/assignment:** 100% (inclusive of 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.

**Data and File Structures**

(a) *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 ammortized 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, programming implementations should be done as assignments. The Computing Laboratory course acts as a supplement to this course.

(b) **Prerequisites:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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.

**Database Management Systems**

(a) *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 subqueries, inserting, modifying and deleting data, advanced joins, views, transactions, integrity constraints, cascading actions, autorization 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:**

**Examination:** 60%

**Laboratory/assignment:** 40%

(e) **References:**


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


**Design and Analysis of Algorithms**

(a) *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.

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

*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 biconnected components; minimum spanning trees, Kruskal's and Prim's algorithms, Johnson's implementation of Prim's algorithm using priority queue data structures.

*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.

*NP-completeness:* Informal concepts of deterministic and nondeterministic algorithms, P and NP, NP-completeness, statement of Cook's 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:**

**Examination:** 70%

**Laboratory/assignment:** 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. Mehlhom: 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.

**Discrete Mathematics**

(a) *Combinatorics*: 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:**

**Examination:** 80%

**Laboratory/assignment:** 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) *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, Lagrange's 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:**

**Examination:** 80%

**Laboratory/assignment:** 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) 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) **Prerequisite(s):** Nil

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

(d) **Marks Distribution:**

**Examination:** 50% (including programming tests)

**Laboratory/assignment:** 50%

1. **References:**

(a) B.W. Kernighan and D.M. Ritchie: The C Programming Language, Prentice Hall, 1980.
(b) B. Gottfried: Programming in C, Schaum Outline Series, 1996.
(c) B. Stroustrup: The C++ Programming Language, 2nd ed., Addison-Wesley, 1995.

(d) Cay S. Horstmann: Core Java Volume I – Fundamentals, 11th ed., Prentice Hall, 2018.

(e) Joshua Bloch: Effective Java, 3rd ed., Addison-Wesley, 2018.

(f) B.W. Kernighan and R. Pike: The Practice of Programming, Addison-Wesley.

(g) B.W. Kernighan and P.J. Plauger: The Elements of Programming Style, McGraw-Hill.

(h) J. Bentley: Programming Pearls, Addison-Wesley, 1986.

(i) J. Bentley: More Programming Pearls, Addison-Wesley, 1988.

(j) B.W. Kernighan and R. Pike: The Unix Programming Environment, Prentice Hall.

**Linear Algebra**

(a) *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 Gram-Schmidt 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:* Classifiction 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**

1. Steve Leon, Linear Algebra with Applications, Pearson Global edition.

**Operating Systems**

(a) *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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


1 A. Silberschatz and P. B. Galvin: Operating Systems Concepts, 5th ed., John Wiley and Sons, New York, 1998.

2 J. L. Peterson and A. Silberschatz: Operating Systems Concepts, Addison-Wesley, Reading, Mass., 1987.

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

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

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


**Principles of Programming Languages**

(a) **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; in 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 much time permits: C++ / Java / OCAML / Lisp / Haskell / Prolog

(b) **Prerequisites:** Nil

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

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(e) **References:**


1. Glynn Winskel, A Formal Semantics of Programming Languages: An Introduction
2. Benjamin C. Pierce, Types and Programming Languages
3. Daniel P. Friedman, Mitchell Wand and Christopher T. Haynes, Essentials of Programming Languages
4. Terrence W. Pratt, Marvin V. Zelkowit, Programming Languages: Design and Implementation
5. Allen B. Tucker, Robert Noonan, Programming Languages, Principles and Paradigms
6. Robert W. Sebesta, Concepts of Programming Languages


**Probability and Stochastic Processes**

(a) *Introduction:* Sample space, probabilistic models and axioms, conditional probability, Total probability theorem and Bayes' rule, independence.

*Discrete random variables:* basics, probability mass functions, functions of random variables, expectation, variance, idea of moments, joint probability mass functions, conditioning and independence, notions of Bernoulli, Binomial, Poisson, Geometric, etc., covariance and correlation, conditional expectation and variance, some introduction to probabilistic methods.

*Continuous random variables:* basics, probability density functions, cumulative distribution function, normal random variable.

*Moments and deviations:* Markov's inequality, Chebyshev's inequality, Chernoff bounds

*Limit theorems:* Weak law of large numbers, central limit theorem, strong law of large numbers (proofs under some restrictive setups, if possible.)

*Stochastic processes:* Bernoulli and Poisson processes, branching processes. Markov chains, classification of states, ideas of stationary distributions. Introduction to Martingales and stopping times.

*Applications to computer science:* Balls and bins, birthday paradox, hashing, sorting, random walks, etc.

(b) **Prerequisites:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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


**Statistical Methods**

(a) *Review of Descriptive statistics:* Representation of Data (collection, tabulation and diagrammatic representation of different types of univariate and bivariate data); Measures of Central Tendency (Mean, Median, Quartiles and Percentiles, Trimmed Mean,

Mode); Measures of Dispersion (Range, Variance and Standard Deviation, Mean Deviation, Quartile Deviation).

*Correlation:* Pearson's product-moment correlation coefficient, Spearman's rank correlation, Kendall's tau statistic.

*Regression:* Simple and multiple linear regression, Method of least squares.

*Estimation:* Method of moments, Method of maximum likelihood estimation.

*Hypothesis Testing:* Concept of null and alternative hypotheses, Acceptance and critical regions, Probabilities of Type I and Type II errors, Level and Power of a test, Most powerful tests, Tests for mean and variance of a normal distribution, Randomized test, Tests for parameters of Binomial and Poisson distributions. Concept of p-value. Likelihood ratio tests.

*Interval Estimation:* Interval estimation and its connection with hypothesis testing. Interval estimation for parameters of normal, binomial and Poisson distributions.

*ANOVA:* Fixed effect model for one-way classified data, two-way classified data with one observation per cell (if time permits).

Additional topics, if time permits: Nonparametric Regression; Classification & Clustering

(b) **Prerequisite:** None.

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

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(e) **References:**


1. J. M. Tanur (ed.): Statistics: A Guide to the Unknown.
2. D. Freedman, R. Pisani and R. Purves: Statistics.
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.
5. J. F. Kenney and E. S. Keeping: Mathematics of Statistics.
6. G. U. Yule and M. G. Kendall: An Introduction to the Theory of Statistics.
7. C. R. Rao: Linear Statistical Inference and its Applications.

8. C. E. Croxton and D. J. Cowden: Applied General Statistics.
9. A. M. Goon, M. Gupta and B. Dasgupta: Fundamentals of Statistics, Vol I.
10. P. G. Hoel, S. C Port and Charles J. Stone: Introduction to Statistical Theory.
11. W. A. Wallis and H. V. Roberts: Statistics: A New Approach.
12. P. J. Bickel and K. A. Doksum: Mathematical Statistics.
13. L. Wasserman: All of Statistics: A Concise Course in Statistical Inference.
14. C. Casella and R. L. Berger: Statistical Inference

## 4.2 Elective courses

**Advanced Computer Networks**

(a) *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: Circuit-switching 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:**

**Examination:** 60%

**Laboratory/assignment:** 40%

(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) *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, Rabin's 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


1. B. Khoussainov and A. Nerode: Automata Theory and its Applications, Springer, 2001.
2. E. Grädel, 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) 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) **Prerequisite(s):** Computer organisation; Operating Systems.

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

(d) Marks Distribution:

**Examination:** 60%

**Laboratory/assignment:** 40%

(e) **References:**


(a) Thomas Anderson and Michael Dahlin: Operating Systems Principles and Practice, 2nd ed., Recursive Books, 2014.

(b) Daniel P. Bovet and Marco Cesati: Understanding the Linux Kernel, 3rd ed., O'Reilly 2005/2008.

(c) Robert Love: Linux Kernel Development, 3rd ed., Addison-Wesley Professional, 2010.

(d) Jonathan Corbet, Alessandro Rubini and Greg Kroah-Hartman: Linux Device Drivers, 3rd ed., O'Reilly, 2005.

(e) Research articles as prescribed by the instructor.

**Algorithms for Big Data**

(a) *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:**

**Examination:** 60%

**Laboratory/assignment:** 40%

(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) *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) **Prerequisite:** Computer Organization.

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

(d) **Marks Distribution:**

**Examination:** 75%

**Laboratory/assignment:** 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) *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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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) *Polynomial Manipulations:* GCD and Berlekamp-Massey algorithm, factoring polynomials over finite fields, Berlekamp's 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:* Grobner basis, reduced Grobner bases and Buchberger's 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 Grobner 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, nationals and radicals; 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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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. O'shea: 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) *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

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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) *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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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) *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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**

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

**Computational Geometry**

(a) *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:* Plane-sweep line segment intersection, triangulation of monotone subdivisions, plane-sweep triangulation of simple polygons, 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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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) Details of possible topics to be covered (not all):
*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) **Prerequisite:** Design and Analysis of Algorithms.

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

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/assignment:** 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 Processes

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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, Frédéric Chazal and Mariette Yvinec: Geometric and Topological Inference by https://geometrica.saclay.inria.fr/team/Fred.Chazal/papers/CGLcourseNotes/main.pdf


**Computer Graphics**

(a) *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, inbetweening, 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


1. Fundamentals of Computer Graphics, Peter Shirley, et al., CRC Press, 4th Edition, 2016
2. Computer Graphics Principles and Practice, John F. Hughes, et al., Pearson, 3rd Edition, 2014
4. Computer Graphics with Open GL, 4th Edition, Hearn, Baker and Carithers, Pearson, 2014.


**Computer Vision**

(a) 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:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(e) **References:**


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

**Computing Systems Security I**

(a) *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 defense 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 Defense 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:**

**Examination:** 60%

**Laboratory/assignment:** 40%

(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) 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:**

**Examination:** 70%

**Laboratory/assignment:** 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) 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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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) *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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**

1. Oded Goldreich: Foundations of Cryptography Vol 1
2. Oded Goldreich: Foundations of Cryptography Vol 2
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) 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) **Prerequisite:** None.

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

(d) **Marks Distribution:**

**Examination:** 60%

**Laboratory/assignment:** 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 Cyber Physical Systems Approach

**Digital Signal Processing**

(a) *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:**

**Examination:** 80%

**Laboratory/assignment:** 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 Processes

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


1. Jiri Matousek: Lectures on Discrete Geometry, Springer, 2001
2. Jiri Matousek: Geometric Discrepancy, Springer, 1999
3. 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) 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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) 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:**

**Examination:** 75%

**Laboratory/assignment:** 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.


**Formal Verification of Machine Learning Models**

(a) *Quick introduction to logic and automated reasoning:* Expressing properties using propositional, first order and linear temporal logics; Hoare triples; Satisfaction and validity of formulas; Automated reasoning tools (e.g. SAT / SMT / Theorem Provers).

*Quick recap of ML (optional):* Feed-forward Deep Neural Networks (DNN): Structure, loss functions, non-linear functions used in NNs, training basics, NNs as classifiers; Binarized neural networks; Convolutional Neural Networks (CNN), Recurrent

Neural Network (RNN) basic structure; Reinforcement Learning: Agent, environment, reward, strategy, Q-learning; Probably Approximately Correct Learning: Theorem and applications

*Modeling Specifications of Neural Networks:* Formal modeling of specifications of ML models, using different logic fragments to model requirements, properties for safety, output range, adversarial robustness; Correct-by-construction design of ML systems.

*Verification Methods for Neural Networks:* Validation of ML models using simulation or semi-formal methods; Formal Verification: Algorithms and techniques for checking properties against models; Constraint-solving techniques for DNN verification: Modeling NNs as systems of constraints, Output range analysis, In-depth case study of ACAS-Xu safety verification; Introduction to Abstract Interpretation, Abstraction refinement techniques for safety analysis of DNNs, in-depth case study, applications in MNIST classifier verification; Verification of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) using abstraction refinement.

*Generating explainable interpretations with guarantees:* Using formal methods for explainable ML; verified decision trees.

*Reinforcement Learning (RL) through the lens of formal methods:* Shield synthesis for safe RL using Markov Decision Processes (MDP) as models, Safe RL using speculative control, decision tree policy extraction and other techniques.

(b) **Prerequisites:** Discrete mathematics, Data and File Structures, Design and Analysis of Algorithms, Machine Learning I.

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

(d) **Marks Distribution:** Theory 70%, Assignments 30% [1 complete project on Python based ML model development and formal verification of the model for a given set of properties]

(e) **References:**


1. E-book: Introduction to Neural Network Verification, by Aws Albarghouthi (https://verifieddeeplearning.com), 2020.
2. Papers from relevant conferences and journals.

**Graph Algorithms**

(a) *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:**

**Examination:** 80%

**Laboratory/assignment:** 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) *Introduction*: Image processing systems and its applications.

*Image formation*: Geometric and photometric models; Digitization - sampling, quantization; Image definition and its representation, neighborhood metrics; Point spread function and its properties..

*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.

*Compression:* Lossy/lossless compression, error criteria; block truncation compression, Huffman coding, arithmetic coding, run-length coding.

(b) **Prerequisites:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/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) *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* : Transform domain compression; block transform coding, vector quantization, wavelet based compression, JPEG, JBIG.

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

*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:**

**Examination:** 80%

**Laboratory/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) 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) **Prerequisite(s):** Probability and Stochastic Processes, Linear Algebra.

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

(d) **Marks Distribution:**

**Examination:** 60%

**Laboratory/assignment:** 40%

2. **References:**


(a) Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze: 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) 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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.

**Introduction to Cognitive Science** [7]

(a) Topics – Cognitive Science is the study of how the brain, a biological computation device, gives rise to the mind, a functional construct. It is an interdisciplinary field encompassing neuroscience, computer science, psychology, linguistics and mathematics. The objective of this course is to acquaint students with this interdisciplinary field, give students hands-on training on experiment design in addition the introduction to basic topics.

In this introductory course, the following topics will be covered: From AI to real brains – the history of cognitive science, Psychophysics, introduction to experiment visualization using MATLAB (or other software), Perception, Human-Computer interaction, Vision, Face and object recognition, Learning and Development, Brain imaging and brain mapping, Social Cognition, Memory, Consciousness, the Developing Brain, Metacognitive approaches, Intelligence, Ethics, Cognitive Neuroscience and Security systems, Big data cognitive experiments – the Internet.

(b) **Prerequisites:** An interest about scientifically understanding human brain functions, including the human mind. Basic understanding of matrices and MATLAB would be helpful.

(c) **Hours:** Four lectures per week (1 lecture will be more hands-on and interactive).

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(e) **References:**

1. Mind: Introduction to Cognitive Science by Paul Thagard (2nd edition, 2005)

**Learning Theory**

(a) Topics:

i. General introduction

ii. Introduction to Probability theory and Stochastic Processes

iii. PAC model; Occam's razor

[7] inserted with effect from the 2021-22 academic year as decided in the 77th meeting of the Academic Council

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 Processes

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**

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

**Logic for Computer Science**

(a) 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(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) *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, prob. 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:**

**Examination:** 80%

**Laboratory/assignment:** 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) 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:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


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

**Mobile Computing**

(a) *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 pathloss 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, Activitybased); 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:**

**Examination:** 80%

**Laboratory/assignment:** 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. Stüber, 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) 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:** The student should have had a formal course on Programming and Data Structures, and basic familiarity with Probability, Statistics and Neural Networks, as determined by the teacher.

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

(d) **Marks Distribution:**

**Examination:** 75%

**Laboratory/assignment:** 25%

(e) **References:**

1. Christopher D. Manning and Hinrich Schütze, Foundations of Statistical Natural Language Processing.

2. Daniel Jurafsky and James H. Martin, Speech and Language Processing.

3. Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning.

**Neural Networks**

(a) Inspiration and lessons from the brain, introduction to biological neural networks, Models of artificial neurons, threshold logic, binary neurons and pattern dichotomizers, perceptron, it 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 Processes

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

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/assignment:** 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) *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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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 Computations**

(a) *Mathematical foundations:* Linear space, Scalar product, Hilbert space, Self adjoint operator, Projection operator, Unitary operator, Eigen-values and Eigen basis.

*Basics of Quantum Mechanics:* (i) Postulates, Uncertainty principle, Complementary principle, Unitary Dynamics. (ii) Multipartite quantum system, Quantum entanglement, Schmidt decomposition (optional), No-Cloning Theorem, Distinguishing non-orthogonal quantum states. (iii) General quantum operations, Kraus representation theorem (optional).

*Quantum computing in practice:*
Part 1: Introduction to quantum gates and circuits: (i) qubits and physical realisation, (ii) quantum gates as unitary matrices,(iii) circuits, circuit identities and universality, (iv) Introduction to IBM Quantum Experience.

Part 2: Quantum Algorithms — pseudo-code, success probability and complexity: (i) Deutsch's algorithm, (ii) Deutsch-Jozsa algorithm, (iii) Quantum Fourier Transform and Quantum phase estimation, (iv) Shor's algorithm, (v) Simon's algorithm, (vi) Grover's algorithm.

Part 3: Quantum Protocols: (i) Quantum teleportation, (ii) superdense coding, (iii) remote state preparation, (iv) Basics of Quantum key distribution (BB84 and Ekert's entanglement - optional)

Part 4: Applications: (i) Solving linear systems of equations, (ii) solving combinatorial optimization problems using QAOA, (iii) binary classification using VQE.

Part 5: Advanced: Noisy intermediate scale quantum computing (NISQ) era and its challenges, Quantum Error Correcting Codes (QECC), Quantum Memory.

(b) **Prerequisites:** (for Non-CS Stream) Co-curricular semester long courses on Design and Analysis of Algorithms, Data and File Structures, and preferably Computer Organization.

(c) Two lectures each of two hours duration

(d) Theory 60% and Practical Assignment 40%

(e) **References:**


1. Michael A. Nielsen and Isaac L. Chuang: Quantum Computation and Quantum Information, Cambridge University Press, 2010.
2. Phillip Kaye, Raymond Laflamme, and Michele Mosca: An Introduction to Quantum Computing, Oxford U. Press, New York, 2007.
3. John Preskil: Lecture notes, http://theory.caltech.edu/~preskill/ph229/.

4. John Watrous: Quantum Computation lecture notes, https://cs.uwaterloo.ca/~watrous/QC-notes/.

5. N. David Mermin: Quantum Computer Science, Cambridge University Press, 2007.

**Quantum Information Theory**

(a) *Introduction:* A brief history about the emergence of quantum information; a brief review of fundamental concepts of the quantum theory: axioms of Hilbert space quantum mechanics; indeterminism; interference; uncertainty; superposition; entanglement.

*The Noiseless and Noisy Quantum Information:* (i) Noiseless Quantum Information: quantum Bits and qudits; Reversible evolution; Measurement; Schmidt decomposition; HJW theorem, Kraus representation theorem. (ii) Noisy Quantum Information: Density operators; General quantum measurement(POVM); Evolution of quantum states; Quantum channels and their examples; Purification; Isometric evolution.

*Basic Quantum Protocols and Resource Inequalities:* Entanglement Distribution; Super dense coding; Teleportation; Optimality of quantum protocols; Extension to qudits; Quantum nonlocality and contextuality and their applications - Bell theorem and CHSH game.

*Basic Tools and Information Measures in Quantum Information:* Distance Measures: Trace Distance; Fidelity; Purified Distance; Relationship between various distance measures; Gentle measurement lemma. Information Measures and their Properties: Shannon entropy; Relative entropy; Von Neumann entropy; Quantum relative entropy; Coherent information; Hypothesis testing divergence; Max relative entropy; additivity and sub-additivity property of various information measures.

*Quantum Shannon Theory:* Noiseless Tasks: Schumacher Compression; Distillation of entanglement; State merging; State splitting; State redistribution. Noisy Tasks: Classical capacity of quantum channels; Holevo information; Private capacity of quantum Channels; Entanglement assisted classical capacity of quantum channels; Quantum capacity of quantum channel.

*Quantum Cryptography:* Privacy amplification and Information reconciliation; Quantum key distribution; Privacy and Quantum information; Security of quantum key distribution.

(b) **Prerequisites:** Basic familiarity with linear algebra and probability as determined by the teacher.

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 20%

(e) **References:**


1. Michael A. Nielsen and Isaac L. Chuang: Quantum Computation and Quantum Information, Cambridge University Press, 2010.
2. Masahito Hayashi: Quantum Information, Springer, 2017.
3. John Preskil: Lecture notes, http://theory.caltech.edu/~preskill/ph229/.
4. John Watrous: Quantum Computation lecture notes, https://cs.uwaterloo.ca/~watrous/QC-notes/.


**Randomized and Approximation Algorithms**

(a) The course has two parts – Randomized Algorithms and Approximation Algorithms. The instructor can choose from the broad list given against the two parts. 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:**

**Examination:** 80%

**Laboratory/assignment:** 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) *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:** Discrete Mathematics

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

(d) **Marks Distribution:**

**Examination:** 70%

**Laboratory/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) *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) **Prerequisite:** Statistics, Probability and Stochastic Processes.

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

(d) **Marks distribution:**

**Examination:** 70%

**Laboratory/assignment:** 30%

(e) **References:**


1. S. M. Ross: Simulation, Second edition.
2. L. Devroye: Non-uniform Random Variate Generation.
3. R. A. Thisted: Elements of Statistical Computing.
4. L. Breiman et al: Classification and Regression Trees.
5. M. P. Wand and M. C. Jones: Kernel smoothing.
6. D. W. Scott: Multivariate Density Estimation: Theory, Practice, and Visualization.
7. R. O. Duda P. E. Hart D. G. Stork: Pattern Classification.
8. D. Kundu and A. Basu: Statistical Computing.
9. G. McLachlan and T. Krishnan: The EM Algorithm and Extensions.
10. K. Lange: MM Optimization Algorithms.
11. C. de Boor: A Practical Guide to Splines
12. T. Hastie, R. Tibshirani and J. Friedman: The Elements of Statistical Learning: Data Mining, Inference, and Prediction.
13. C. Robert and G. Casella: Monte Carlo Statistical Methods

14. W.R. Gilks, S. Richardson and D. J. Spiegelhater: Markov Chain Monte Carlo in Practice.

**Topics in Privacy**

(a) *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:** Nil

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

(d) **Marks Distribution:**

**Examination:** 80%

**Laboratory/assignment:** 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
4. Internet Resources and research papers.