INDIAN STATISTICAL INSTITUTE

Student's Brochure

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

(Effective from 2012-13 Academic Year)

203 BARRACKPORE TRUNK ROAD
KOLKATA 700108

1. COURSE STRUCTURE

The two-year Master of Technology programme in Computer Science is divided into four semesters. A student is required to take five courses in Semester I, II and III and two courses in Semester IV, making up a total of seventeen courses plus a dissertation that is equivalent to three credit courses.

The courses offered indifferent semesters are as follows:

Semester I (1st year, 1st semester):

1.1 Discrete Mathematics

1.2 Data and File Structures

1.3-1.5 Three subjects from the following list as advised by the faculty based on the background of the student:

- Al. Introduction to Programming
- A2. Data and File Structures Laboratory
- A3. Computer Organization
- A4. Elements of Algebraic Structures
- A5. Probability and Stochastic Processes
- A6. Principles of Programming Languages

Semester II (1st year, 2nd semester):

2.1. Computer Networks

2.2. Design and Analysis of Algorithms

2.3. Automata, Languages and Computation

2.4. Database Management Systems

2.5. Operating Systems

Semester III (2nd year, 1st semester):

3.1-3.5 Five electives from the 2nd year elective list

(*The teachers' committee determines from the elective list the elective subjects to be offered in any particular semester.)*

3.6 Dissertation (to be continued to Semester IV)

Semester IV (2nd year, 2nd semester):

4.1-4.2 Two electives from the 2nd year elective list

(*The teachers' committee determines from the elective list the elective subjects to be offered in any particular semester.)*

4.3 Dissertation (to be completed) [The dissertation is of 300 marks. The marks break up across semesters will be decided by the dissertation committee.]

Dissertation: A student is required to work for a dissertation on a topic assigned/approved by the teachers' committee under the supervision of a suitable ISI faculty member [see also 13. Practical Training]. The work for a dissertation should be substantial and relate to some important problem in an area of computer science and/or its applications and should have substantial theoretical or practical significance. A critical review of recent advances in an area of computer science and/or its applications with some contribution by the student is also acceptable as a dissertation.

The work should be commenced at the beginning of the third semester and be completed along with the courses of the fourth semester. The dissertation should be submitted by the middle of July of the year of completion. The dissertation will be evaluated by a committee consisting of the supervisor and an external expert. The student has to defend his/her dissertation in an open seminar. The dissertation is considered to be equivalent to three credit courses.

A student is required to submit seven copies of the dissertation. The Dean will reimburse up to a maximum limit of ₹ 1000.00 towards the cost of preparation of seven copies of the dissertation.

A semester-wise break-up of the subjects in the course along with the marks are as follows:

Semester	Subjects
I	Discrete Mathematics
	Data and File Structures
	Any three optional from the following as advised by the faculty based on the background of the student:
	Introduction to Programming
	Data and File Structures Laboratory
	Elements of Algebraic Structures
	Probability and Stochastic Processes
	Principles of Programming Languages
	Computer Organization
Total	**500**
II	Design and Analysis of Algorithms
	Database Management Systems
	Automata, Languages and Computation
	Operating Systems
	Computer Networks
Total	**500**
III	Elective I
	Elective II
	Elective III
	Elective IV
	Elective V
	Dissertation (to be continued to Semester IV)
Total	**500**
IV	Elective VI
	Elective VII
	Dissertation‡ (300)
‡*The marks break up across semesters will be decided by the dissertation committee.*	
Total	**500**
Full Marks	**2000**

2. NON-CREDIT COURSE

A student may also have to successfully complete a non-credit course in some semester on the advice of the faculty in a subject which may be a prerequisite for a course to be taken by him/her in a subsequent semester.

The phrase “successful completion” means securing a composite score of at least 35% marks in which case the score will be recorded in the mark-sheet.

A student may also undergo an extra course in his/her own interest in any semester in which case he/ she must take the tests for the course and the composite score obtained will be recorded in the mark-sheet if it is at least 35%.

A student will be allowed to undergo at most one extra course in each semester whether or not it is recommended by the faculty.

2. METHOD OF EXAMINATION AND AWARD OF DEGREE

For each course there are two examinations: mid-semestral and semestral along with class tests and assignments as may be decided by the respective teacher. The composite score in a course is a weighted average of the scores in the mid-semestral and semestral examinations, assignments, project work, class test, etc. (announced at the beginning of the semester). For courses other than the dissertation, the minimum weight given to the semestral examination is 50.

A student will be declared to have passed a semestral examination if he/she

i) secures at least a composite score of 45% on an average in the credit courses;
ii) does not obtain a composite score of less than 35% in any course;
iii) does not obtain a composite score of less than 45% in more than one credit course;
iv) does not obtain a composite score of less than 35% in any non-credit course recommended by the teachers’ committee; and
v) does not obtain less than 50% in the programming/software assignments for each course where there are such assignments.

If a student misses the mid-semestral or semestral examination due to medical/family emergency, he/she may be allowed to take a supplementary (SU) examination at the discretion of the Teachers’ Committee, on the basis of an adequately documented request from the student. The maximum a student can score in an SU exam is 60%.

Depending on the result of each semester a student may be allowed to take back-paper (BP) examination to satisfy the afore- mentioned passing criteria [(i), (ii), (iii) and (iv)]. The ceiling of the total number of BP examination is 2 in first year and 2 in second year. Teacher’s committee will decide about the time of this examination and until then the concerned student can attend the classes of the next semester. At most one BP examination is allowed in a given course. The post-BP score in a course is equal to the maximum of BP exam score and composite score, subject to a maximum of 45%. A student may take more than the allotted quota of backpaper examinations in a given academic year, and decide at the end of that academic year which of the BP examination scores should be disregarded. If a student obtains less than 50% in programming/software assignments (wherever applicable), he/she may be allowed to take additional assignments.

If a student, even after SU and BP examination or additional assignments stated here, fails to satisfy the passing criteria [(i)-(v)] in the first or the second semester, he/she has to discontinue the course. In case of third or fourth semester, at the discretion of the teachers’ committee, he/she may be allowed to repeat only once the second year of the course without stipend. The scores obtained during the repetition of the final year are taken as the final scores in the final year.

A student admitted to the course is allowed to attend the second semester of the course if he/she passes the first semestral examination.

A student who passes the second semestral examination will be allowed to go in for practical training. A student who completes the practical training satisfactorily, as certified by the supervisor, is promoted to the third semester of the course; otherwise he/she has to discontinue the course.

A student promoted to the third semester of the course will be allowed to attend the fourth semester of the course if he/she passes the third semestral examinations and his/her progress in dissertation is satisfactory as assessed by an internal committee.

A student who submits his/her dissertation within the prescribed time limit and does not obtain less than 45% in the dissertation, and passes the fourth semestral examinations will be declared to have completed the fourth semester of the course.

A student, who has successfully completed the fourth semester of the course and whose conduct is satisfactory, is said to have passed the M. Tech. (Computer Science) examination and is placed in the

i) **First division with Honours** if the student secures an overall average percentage score of at least 75 in the twenty four courses,
ii) **First division** if the student secures an overall average percentage score of at least 60 but less than 75, in the twenty four courses,
iii) **Second division** if the student fails to secure first division with Honours or first division.

Note: Unsatisfactory conduct means copying in the exam, rowdyism or some other breach of discipline or unlawful/ unethical behaviour.

A student passing the M. Tech. (CS) examination is given the degree certificate and the mark-sheet mentioning

i) the twenty four credit courses taken and the composite percentage score in each course,
ii) the non-credit courses taken and the composite percentage score in each such course, and
iii) the division in which the student is placed.

3. ATTENDANCE

The attendance requirement is 75% for each course in a semester. If a student fails to attend classes in any course continuously for one week or more, he/she would be required to furnish explanation to the Dean of Studies or the Class Teacher for such absence. If such explanation is found to be satisfactory by the Teachers' Committee, then the calculation of percentage of attendance is determined disregarding the period for which explanation has been provided by the student and accepted by the Teachers' Committee.

4. DEPOSIT

Each student will have to make a refundable deposit of one month stipend as caution money for use of laboratory and/or Dean's library facilities. The deposit is refundable after deduction of cost of damages or losses, if any, on termination of the course.

5. STIPEND AND CONTINGENCY GRANT

All students other than those sponsored by the employers are awarded full stipend at the time of admission initially for the first semester only. The present rate of stipend is ₹ 2500/- (Rupees two thousand five hundred only) per month. The students (other than those sponsored by the employers) are also eligible to receive a contingency grant of ₹ 3000/- (Rupees three thousand only) per year as reimbursement of cost of text books and supplementary text books, photocopies of required academic materials, a scientific calculator and other required accessories for practical classes. All such expenditure should first be approved by the respective class teacher. The payment of stipend and the reimbursement of contingency grant will, however, be governed by the following terms and conditions:

a. The contingency grant sanctioned will be treated as a limit and the “student concern” will be reimbursed the actual expenditure incurred by him/her on the admissible items within the limit. The grant is not to be paid as an outright payment.
b. The books and non-consumable items purchased or acquired out of the contingency grant allowed will be property of the Institute and the student will have to return them at the end of the course.
c. It will be obligatory for every student concerned to undertake 8 to 10 hours (per week) of work related to teaching and research activities as assigned to him/her by the Institute. This could include tutorials, laboratory classes, development activities undertaken by faculty members, maintenance and operation of computers and other facilities, assistance in Library etc.
d. Wherever Government of India/University Grants Commission/Council of Scientific and Industrial Research/ Industry sponsored projects are undertaken, the services of the students may be used for providing assistance in projects funded by these agencies. In that event, a portion of stipend amount i.e., ₹ 800/- per month, for the time being may be charged to the project funds for the duration a student is engaged on such a project.
e. The Institute will work out specific programmes of work and maintain the record of each student.
f. If however a student fails to secure a first division or equivalent in any semester, his/her stipend and contingency grant will be reduced to ₹ 1250/- and ₹ 750/- respectively in the following semester.
g. A student shall be required to give an undertaking to the effect that he/she would not leave the course midway or appear in any competitive examination, etc., not related to engineering and technology, in order to be eligible to receive this stipend.
h. During the course of studies, the student shall not receive any emoluments, salary, stipend, etc. from any other source.
i. Suitable hostel type accommodation may be provided wherever available.
j. No House Rent Allowance (HRA) is admissible to the M. Tech. students.

No stipend is awarded to:

i. a repeating student
ii. a student whose attendance falls short of 75% overall in the preceding semester.

Stipends may be restored because of improved performance and/or attendance, but no stipend is restored with retrospective effect.

6. CLASS TEACHER

One of the teachers in a semester is designated as the respective class teacher. All students are required to meet their respective class teacher periodically to get their academic performance reviewed, and to discuss any academic problem, if necessary.

7. ISI LIBRARY RULES

Students are allowed to use the reading-room facilities in the library. They have to pay ₹ 250/- as security deposit in order to avail of the borrowing facility. At most four books can be borrowed at a time. Any book from the Text Book Library (TBL) may be issued to a student only for overnight or week-end provided at least two copies of that book are present in the TBL; only one book will be issued at a time to a student. Fine will be charged if any book is not returned by the due date stamped on the issue-slip. The library rules and other details are posted in the library.

8. DEAN'S LIBRARY/STUDENTS' LIBRARY RULES

A student may also borrow text books for the particular semester the student is studying in. Books are usually issued at the beginning of each semester and have to be returned at the end of that semester. Exact date of the issuance and return will be announced by the respective class teacher.

9. PRACTICAL TRAINING

A student who passes the second semester examinations of the first year will be allowed to go in for practical training. The practical training may be organized anywhere in research institutes or in public/private sector organizations. It is generally not possible to arrange training somewhere of the trainee's choice. However, it is recommended that the students should go outside the Institute (ISI).

The duration of the training is about eight weeks. During the period of training, the student is placed under the control of an assigned supervisor belonging to the organization where the training is arranged. A student who completes the practical training satisfactorily, as certified by the supervisor, is promoted to the second year of the course.

A student who undergoes practical 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 own discretion. A student who wishes to receive stipend from the Institute will receive his/her usual monthly stipend. However, the students other than those who are placed in Kolkata for practical training will be paid an additional monthly allowance at the following rates:

i) Fifteen percent (15%) of the monthly stipend for those placed in the suburbs of Kolkata,
ii) Twenty percent (20%) of the monthly stipend for those placed outside of Kolkata and its suburbs, but offered accommodation in the ISI hostel or accommodation by the organization where they are placed,
iii) Forty percent (40%) of the monthly stipend for those placed outside Kolkata and its suburbs, but not offered accommodation as in (ii).
iv) For travel from Kolkata to the city where the students are placed, and back, the students will be reimbursed second class to and fro train fare with sleeper charges, an allowance of ₹ 50.00 for every 24 hours or part thereof during the train journey, and incidental expenditure to cover cost of road travel/coolies to the extent of ₹ 30.00 each way for the whole trip.

Practical training may 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. [For detail please see the proceeding of the 36th meeting of AC held on 1st December 2000.]

10. SYLLABI

Syllabi of the subjects are given in the following pages. Each syllabus is organized as follows.

(a) Topics, (b) pre-requisites/co-requisites if any, (c) number of lectures (tutorials if any) per week, (d) percentage weights for theory and programming/software assignments/mini projects (if any), (e) list of references, (f) supplementary information.

1.1. Discrete Mathematics

(a) *Combinatorics:* Multinomial theorem, principle of inclusion exclusion; Recurrence relations— classification, 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 Kuratowsky's theorem, dual of a planer 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) Nil

(c) Four lectures per week

(d) Theory 80% and 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. E. Mendelsohn: Introduction to Mathematical Logic, 2nd ed. Van-Nostrand, London, 1979.
9. L. Zhongwan: Mathematical Logic for Computer Science, World Scientific, Singapore, 1989.
10. F. S. Roberts: Applied Combinatorics, Prentice Hall, Englewood Cliffs, NJ, 1984.
11. Lewis and Papadimitriou: Elements of Theory of Computation (relevant chapter on Logic), Prentice Hall, New Jersey, 1981.

1.2. Data and File Structures

(a) (i) *Introduction:* Algorithms, programs, storage devices, records and sequential files, compiler and OS (if it is required by the class).

Introduction to complexity of algorithms and different asymptotic notations.

Data structure: Introduction to ADT, formal definitions, implementations of basic data structures, array, stack, queue, dequeue, priority queue, linked list, binary tree and traversal algorithms, threaded tree, m-ary tree, heap, generalized list and garbage collection.

Searching: Binary search, Fibonacci search, binary search tree, height balanced tree, splay tree, digital search tree, trie, hashing techniques.

(ii) *Records and files:* Fixed length/variable length records, pinned/unpinned records, heap file, hashed file, indexed file, relative file, file with dense index, multi-key access file, inverted list, multi-list organization, B-tree, B*-tree, 2-3 tree, storage organization for HSAM, HISAM, HIDAM, HDAM (H= hierarchical, I = indexed, D = direct, AM = access method), need for file reorganization. Management of addition I deletion I overflow for different types of file organization, and introduction to distributed file system, file security, access control and version control.

(b) Nil

(c) Four lectures and one tutorial per week

(d) Theory 80% and non-programming assignments 20%

(e) References:

1. T. A. Standish: Data Structures, Algorithms and Software Principles, Addison-Wesley, Reading, Mass., 1995.
2. L. Nyhoff, C++— An Introduction to Data Structures, Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tannenbaum and M. J. Augesestein: Data Structures Using PASCAL, Prentice Hall, New Jersey, 1981.
4. D. E. Knuth: The Art of Computer Programming. Vol. 1, 2nd ed. Narosa/Addison-Wesley, New Delhi/London, 1973.
5. T. A. Standish: Data Structure Techniques, Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni: Fundamentals of Data Structures, CBS, 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 and R. Rivest: Introduction to Algorithms, McGraw Hill, New York, 1994.

13. S. Sahani: Data Structure, Algorithms and Applications in JAVA, McGraw Hill, New York, 2000.
14. D. Wood: Data Structure, Algorithms and Performance, Addison-Wesley, Reading, Mass., 1993.

A1. Introduction to Programming

(a) *Introduction to computer programming:* Algorithm, storage, file, software-hardware interface (if it is required by the class).

Imperative languages: Introduction to imperative language; syntax and constructs of a specific language (preferably C);

Functions and Recursion: Parameter passing, procedure call, call by value, call by reference, recursion.

Different programming language types: Initial concepts of object-oriented programming, functional programming, logic programming.

Efficiency issues.

(b) Nil

(c) Four lectures and one tutorial per week

(d) Theory 60% and Assignment 40%

(e) References:

1. B. W. Kernighan and D. M. Ritchi: The ‘C’ Programming Language, Prentice Hall, Englewood Cliffs, NJ, 1980.
2. B. Gottfried: Programming in C, Schaum Outline Series, New Delhi, 1996.
3. B. Stroustrup: The C++ Programming Language, 2nd ed., Addison-Wesley, California, 1995.
4. D. M. Arnow and G. Weiss: Introduction to Programming using Java, Addison-Wesley, London, 1999.
5. R. W. Sebesta: Concepts of Programming Languages, 4th ed., Addison-Wesley, Reading, Mass., 1998.
6. R. Sethi and T. Stone (Eds.): Programming Languages: Concepts and Constructs, 2nd ed., Addison-Wesley, Reading, Mass., 1996.
7. T. W. Pratt and M. V. Zelkowitz: Programming Languages: Design and Implementation, 4th ed., Prentice Hall, Engle- wood Cliffs, 2001.
8. J. Bentley: Programming Pearls, Addison-Wesley, Reading, Mass., 1986.
9. J. Bentley: More Programming Pearls, Addison-Wesley, Reading, Mass., 1988.

A2. Data and File Structures 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 the Data and File Structures course. The laboratory sessions should include but are not limited to:

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; AVL tree implementation. Hashing: Hash table implementation, searching, inserting and deleting Searching and sorting techniques

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

(b) Data and File Structures

(c) Four hours per week

(d) Assignments— 60% and Lab Test— 40%

(e) References:

1. T. A. Standish: Data Structures, Algorithms and Software Principles, Addison-Wesley, Reading, Mass., 1995.
2. L. Nyhoff, C++— An Introduction to Data Structures, Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tannenbaum and M. J. Augesestein: Data Structures Using PASCAL, Prentice Hall, New Jersey, 1981.
4. D. E. Knuth: The Art of Computer Programming. Vol. 1, 2nd ed. Narosa/Addison-Wesley, New Delhi/London, 1973.
5. T. A. Standish: Data Structure Techniques, Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni: Fundamentals of Data Structures, CBS, 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 and R. Rivest: Introduction to Algorithms, McGraw Hill, New York, 1994.
13. S. Sahani: Data Structure, Algorithms and Applications in JAVA, McGraw Hill, New York, 2000.
14. D. Wood: Data Structure, Algorithms and Performance, Addison-Wesley, Reading, Mass., 1993.
15. B. W. Kernighan and D. M. Ritchie: The C Programming Language, Prentice Hall of India, 1994.
16. B. Gottfried: Programming in C, Schaum Outline Series, New Delhi, 1996.
17. B. W. Kernighan and R. Pike: The Unix Programming Environment, Prentice Hall of India, 1996.

A3. 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 Kamaughs map, truth tables, duality and complementation, canonical forms, fundamental Boolean operations-AND, OR, NAND, NOR, XOR.

Minimization of Boolean functions: Using fundamental theorems, Kamaughs Maps, Mcclusky's tabular 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.

Processor 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.

Control Unit Design: Hardware control unit design, hardware programming language, microprogramming, horizontal, vertical and encoded-control microprogramming, microprogrammed control unit design.

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) Nil

(c) Four Lectures

(d) Theory 80% and Assignment 20%

(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.

A4. 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.
Linear algebra: Vector spaces, basis and dimension, linear transformations and matrices, rank and nullity determinants, eigenvalues and eigenvectors.
Field extensions: Finite dimensional, algebraic and transcendental; splitting field of a polynomial, existence and uniqueness of finite fields, application to Coding Theory.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and 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. Berstein: Topics in Algebra, Vikas Pub., New Delhi 1987.
8. G. Birkhoff and S. McLane: A Survey of Modem Algebra, 4th ed. Macmillan, New York, 1977.

A5. Probability and Stochastic Processes

(a) *Probability theory:* Probability, conditional probability and independence; Random variables and their distributions (discrete and continuous), bivariate and multivariate distributions; Laws of large numbers, central limit theorem (statement and use only).
Stochastic process: Definition and examples of stochastic processes, weak and strong stationarity; Markov chains with finite and countable state spaces -classification of states, Markov processes, Poisson processes, birth and death processes, branching processes, queuing processes.
Selected applications: Analysis of algorithms, performance evaluation and modelling of computer systems.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. W. Feller: An Introduction to Probability Theory and its Applications (Volume I and II), 3rd ed. John Wiley, New York, 1973.
2. P. G. Hoel, S. C. Port and C. J. Stone: Introduction to Probability Theory, University Book Stall/ Houghton Mifflin, New Delhi/New York, 199811971.

3. K. L. Chung: Elementary Probability Theory and Stochastic Processes, Springer-Verlag, New York, 1974.
4. S. M. Ross: Stochastic Processes, John Wiley, New York, 1983.
5. H. M. Taylor: First Course in Stochastic Processes, 2nd ed. Academic Press, Boston, 1975.
6. H. M. Taylor: Second Course in Stochastic Processes, Academic Press, Boston, 1981.
7. U. N. Bhat: Elements of Applied Stochastic Processes, John Wiley, New York, 1972.
8. K. S. Trivedi: Probability and Statistics with Reliability, Queuing and Computer Science Application, Prentice Hall, Englewood Cliffs, 1982.
9. M. Hofri: Probabilistic Analysis of Algorithms, Springer-Verlag, 1987.
10. G. Latouche (Eds.): Probability Theory and Computer Science, Academic Press London, 1983.

A6. Principles of Programming Languages

This subject is an elective for students with Computer Science background in the first semester. Students with non-Computer Science background can opt for this subject as an elective in the third semester.

(a) Theory:

Introduction: Overview of different programming paradigms e. g. imperative, object oriented, functional, logic and concurrent programming.

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

Imperative and Object Oriented 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 based on time constraints: C++/Java/OCAML/Lisp/Haskell/Prolog.

(b) Pre-requisite: Introduction to Programming.

(c) Three lectures and 1hands-on per week

(d) Theory 70% and Assignment 30%

(e) References:

1. Glynn Winskel, A Formal Semantics of Programming Languages: An Introduction, MIT Press.
2. John C. Mitchell, Foundations for Programming Languages, MIT Press.
3. Benjamin C. Pierce, Types and Programming Languages, MIT Press.

4. Daniel P. Friedman, Mitchell Wand and Christopher T. Haynes, Essentials of Programming Languages, Prentice Hall of India.
5. Ravi Sethi, Programming Languages: Concepts and C

2.1. Computer Networks

(a) *Introduction:* Computer networks and distributed systems, classifications of computer networks, layered network structures.

Data Communication Fundamentals: Channel characteristics, various transmission media, different modulation techniques.

Queuing Theory: M/M queuing systems, M/G/I queuing system; Network performance analysis using queuing theory.

Network Structure: Concepts of subnets, backbone and local access; Channel sharing techniques-FDM, TDM; Polling and concentration, message transport: circuit, message and packet switching, topological design of a network.

Data Link Layers: Services and design issues, framing techniques, error handling and flow control, stop and wait, sliding window and APRPANET protocols, HDCLC standard.

Network Layer: Design issues, internal organization of a subnet, routing and congestion control techniques, network architecture and protocols, concepts in protocol design, CCITT recommendation X. 25

LANs and their Interconnection: Basic concepts, architectures, protocols, management and performance of Ethernet, token ring and token bus LANS; Repeaters and Bridges.

Internet: IP protocol, Internet control protocols— ICMP, APR and RAPP, Internet routing protocols— OSPF, BGP and CIDR.

ATM: ATM switches and AAL layer protocols.

Network Security: Electronic mail, directory services and network management.

Wireless and mobile communication: Wireless transmission, cellular radio, personal communication service, wireless protocol. Network planning, Gigabit and Terabit technology, CDMA, WCDMA, WDM, optical communication networks.

(b) Nil

(c) Three lectures and one tutorial per week

(d) Theory 70% and Assignment 30%

(e) References:

1. A. Tannenbaum: Computer Networks, 3rd ed., Prentice Hall India, 1996.
2. W. Stallings: ISDN and Broadband ISDN With Frame Relay and ATM, Prentice Hall, Englewood Cliffs, 1995.
3. W. Stallings: Local and Metropolitan Area Networks, 4th ed., Macmillan, New York, 1993.
4. Kaufman, R. Perlman and M. Speciner: Network Security, Prentice Hall, Englewood Cliffs, 1995.
5. V. P. Ahuja: Design and Analysis of Computer Communication Networks, McGraw Hill, New York, 1987.
6. L. Gracial and I. Widjaja: Communication Networks, Tata-McGraw Hill, New Delhi, 2000.
7. L. L. Paterson and B. S. Davie: Computer Network, Morgan Kaufman, San Mateo, 2000.

2.2. 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) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(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 and R. L. Rivest: Introduction to Algorithms, Prentice Hall of India, New Delhi, 1998.
4. E. Horowitz and S. Sabni: 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.

(f) At least one assignment involving implementation of several algorithms of same asymptotic complexity for a problem and their empirical comparisons.

2.3. Automata, Languages and Computation

(a) *Automata and Languages:* Finite automata, regular languages, regular expressions, equivalence of 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 CFL, 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.
Complexity: Time complexity of deterministic and nondeterministic Turing machines, P and NP, NP- completeness, Cook's Theorem, other NP -Complete problems.

(b) Nil

(c) Four lectures per week

(d) Theory 100%

(e) References:

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

2.4. 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.
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, trees or graphs, generalization and specialization, aggregation.

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

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

Structured query language: Description an actual RDBMS and SQL.

Normalization: Pitfalls in RDBMS, importance of normalization, functional, multi-valued and join dependencies, INF to 5NF, limitations of RDBMS.

Database tuning: Index selection and clustering, tuning of conceptual schema, denormalization, tuning queries and views. Query optimization: Importance of query processing, equivalence of queries, cost estimation for processing a query, general strategies, bi-relational and multi-relational join algorithms, algebraic manipulation.

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.

(b) Nil

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

(d) Theory 70% and Assignment 30%

(e) References:

1. H. F. Korth and A. Silberschatz: Database System Concepts, McGraw Hill, New Delhi, 1997.
2. R. A. Elmasri and S. B. Navathe: Fundamentals of Database Systems, 3rd ed., Addison-Wesley, 1998.
3. R. Ramakrishnan: Database Management Systems, 2nd ed., McGraw Hill, New York, 1999.

2.5. 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.

Processor management: CPU scheduling— short-term, medium term and long term scheduling, non-pre-emptive and pre-emptive 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 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) Nil

(c) Four lectures per week (including tutorial).

(d) Theory 80% and 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: Modem Operating Systems, Prentice Hall, Englewood Cliffs, 1992.
5. S. E. Madnick and J. J. Donovan: Operating Systems, McGraw Hill, New York, 1974.

Elective List of 2nd Year

The elective list of 2nd year is given below. All the electives of 2nd year are put in a common pool and the subjects to be offered in a particular semester would be decided by the concerned teachers' committee. However, a subject offered in Semester III cannot be offered again in Semester IV.

B1. Optimization Techniques

(a) *Linear Programming:* A brief review of simplex and revised simplex algorithms, Bland's rule, duality theory, large scale linear programmes, computational complexity of simplex method, polynomial time algorithms— ellipsoidal and Karmarkar's methods.
Integer Programming: All integer and mixed integer programming problems, cutting planes and branch and bound algorithms, introduction to the ideas of NP-completeness, travelling salesman and other related problems.
Non-linear Programming: General constrained mathematical programming problems, Kuhn-Tucker-Lagrangian necessary and sufficient conditions, interior point methods, standard algorithms like feasible direction and gradient projections convergence of the methods, convex programming problems, quadratic programming.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:
1. R. J. Vanderbei: Linear Programming Foundations and Extensions, Kluwer Academic Publishers, Boston/London, 1997.
2. D. G. Luenberger: Introduction to Linear and Non-Linear Programming, Addison-Wesley Publishing Co., London/Amsterdam, 198?.
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.

B2. Cryptology

(a) *Introduction:* Brief introduction to number theory, Euclidean algorithm, Euler's totient function, Fermat's theorem and Euler's generalization, Chinese Remainder Theorem, primitive roots and discrete logarithms, Quadratic residues, Legendre and Jacobi symbols.
Basic concepts of cryptology: Cryptography and cryptanalysis, classical cryptosystems, concept of block and stream ciphers, private and public key cryptography.
Information theoretic ideas: Entropy, equivocation, perfect secrecy and unicity distance.
Encryption standard: DES and differential and linear cryptanalysis, Advanced encryption standards.
RSA public key cryptosystems: RSA system, primality testing, survey of factoring algorithms.
Other public key cryptosystems: El Gamal public key cryptosystem, algorithms for discrete log problem, Knapsack public key cryptosystems, cryptanalysis of Knapsack PKC— Shamir's attack and Lenstra, Lenstra and Lovasz algorithm.
Digital signature and hash functions: El Gamal signature scheme, digital signature standard, one-time undeniable and fail-stop signatures, computationally collision-free hash functions, extending hash functions, examples of hash functions.
Security of practical systems: Database, operating system and network security.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:
1. D. E. R. Denning: Cryptography and Data Security, Addison-Wesley, Reading, Mass., 1982.
2. D. M. Bressoud: Factorization and Primality Testing, Springer-Verlag, New York, 1989.
3. R. A. De Millo et al: Foundations of Secure Computations, Academic Press, New York, 1978.
4. E. Kranakis: Primality and Cryptography, B G Teubner, Chichester, 1986.

5. W. Patterson: Mathematical Cryptography for Computer Scientists and Mathematicians, Rowman and Littlefield, 1987.
6. D. Stinson: Cryptography: Theory and Practice, CRC Press, Boca Raton, 1995.
7. W. Leveque: Topics in Number Theory, Vol. l, Addison-Wesley, Mass, 1956.
8. G. A. Jones and J. M. Jones: Elementary Number Theory, Springer-Verlag, London, 1998.
9. B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.
10. A. Menezes, P. C. Van Oorschot and S. A. Vanstone: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.

B3. Advanced Cryptology

(a) *Factoring algorithms:* Recent developments in factoring algorithms, quadratic sieve and number field sieve methods.
Elliptic curves: A detailed study of elliptic curves, factorization of numbers, design of computationally secure elliptic curve cryptosystems.
Identification schemes: Schnorr, Okamoto and Guillou-Quisquater identification schemes.
Authentication codes: computing deception probabilities, combinatorial and entropy bounds.
Secret sharing schemes: Shamir’s threshold schemes, general secret sharing, Brickell vector space construction.
Stream ciphers and S-boxes: cryptographic properties of Boolean functions relevant to stream ciphers and S-boxes, Berlekamp- Massey algorithm, linear complexity, cyclotomic sequences, cryptanalysis of stream ciphers and S-Boxes.
Complexity theoretic foundations of cryptography: One-way functions, pseudo-randomness, interactive proof systems and zero knowledge proofs.
Financial cryptography and e-commerce.
Quantum cryptography: A brief introduction to quantum computing, Shor’s quantum factoring algorithm, quantum key distribution and bit commitment.

(b) Pre-requisite: Cryptology.

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. All references in the course Cryptology-I.
2. D. Stinson: Cryptography, Theory and Practice, CRC Press, Boca Raton, 1995.
3. C. Ding, G. Xiao and W. Shan, The Stability Theory of Stream Ciphers, Springer, LNCS, Berlin, 1991.
4. R. A. Rueppel: Analysis and Design of Stream Ciphers, Springer, Berlin, 1986.
5. O. Goldreich: Foundations of Cryptography— Fragments of a Book, ECCC Reports, Cambridge Univ. Press, London, 2000.
6. N. Koblitz: Introduction to Elliptic Curves and Modular Forms, 2nd ed., Springer, Berlin, 1993.
7. A. Menezes, P. van Oorschot and S. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.

B4. Information and Coding Theory

(a) *Information Theory:* Entropy, its characterization and related properties, Huffman codes, Shannon-Fano coding, robustness of coding techniques, Information measure-noiseless coding, discrete memoryless channel— channel capacity, fundamental theorem of information theory.
Error correcting codes: minimum distance principles, Hamming bound, general binary code, group code, linear group code
Convolution encoding: algebraic structure, Gilbert bound
Threshold decoding: threshold decoding for block codes
Cyclic binary codes: BCH codes, generalized BCH code and decoding, optimum codes, concepts of non-cyclic codes.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. R. Ash: Information Theory, Interscience Publ., Singapore, 1965.
2. E. R. Berlekamp: Algebraic Coding Theory, McGraw Hill, New York, 1986.
3. S. Lin: An Introduction to Error-Correcting Codes, Prentice Hall, Englewood Cliffs, 1970.
4. W. W. Peterson et al: Error Correcting Codes, Wiley, London, 1961.
5. R. W. Hamming: Coding and Information Theory, Prentice Hall, Englewood Cliffs, 1980.
6. A. Khinchin, Mathematical Foundations of Information Theory, Dover Publn., London, 1957.
7. F. J. McWilliams and N. J. Sloane: Theory of Error Correcting Codes, Parts I and II, North-Holland, Amsterdam, 1977.
8. V. Pless: Introduction to the Theory of Error Correcting Codes, 3rd ed., John Wiley, New York, 1982.

B5. Advanced Algorithms for Graph and Combinatorial Optimization Problems

(a) *Shortest path 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, maxfiow-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 (Goldbergarj Ian) 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) Nil

(c) Four lectures per week

(d) Theory 80% and 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.

B6. Multi-dimensional search and Computational geometry

(a) *Multi-dimensional search:* Review of some advanced topics on dynamic unweighted and weighted trees, k-d tree, range tree, priority search tree, finger-tree, interval trees, amortized analysis of dynamically maintaining weight balanced binary trees; Dynamic search trees in secondary storage; Orthogonal range searching, order decomposable problems, multi-dimensional divide and conquer, partial match retrieval in minimum space, few applications to multidimensional searching.

Computational geometry: Point location in monotone subdivision, convex polygons, convex hull of point set and polygon in 2 and 3 dimensions, Voronoi diagram, Delauney triangulation, Arrangement and duality, triangulation of polygons, binary space partitioning, visibility, simplex range searching, isothetic geometry, matrix searching and its applications in different geometric optimization problems, few applications in GIS and robot motion planning, and physical design in VLSI.

(b) Nil

(c) Three lectures and one tutorial per week

(d) Theory 75% and Assignment 25%

(e) References:

1. F. P. Preparata and M. I. Shamos: Computational Geometry: An Introduction, Springer-Verlag, Berlin, 1985.
2. M. De Berg, M. Van Kreveld, M. Overmars and 0. Schwarzkopf: Computational Geometry: Algorithms and Applications, Springer-Verlag, Berlin, 1997,
3. J. O'Rourke: Computational Geometry in C, Cambridge University Press, London, 1995.
4. K. Mehlhom and St. Naher: The LEDA Platform of Combinatorial and Geometric Computing, Cambridge University Press, London, 1999.

(f) Use of LEDA and CGAL library for the standard graph-theoretic and computational geometry problems. (http://www.mpi-sb.mpg.de/mehlhorn/LEDAbook.html)

B7. Combinatorial Geometry

(a) *Basics:* Sylvester-Gallai Problem, Convex Independent Subsets and Erdos-Szekeres Theorem type results.

Theorems: Radon's theorem, Belly's theorem, Ham Sandwich Theorem.

Incidence: Formulation, Szemeredi-Trotter theorems, Unit Distances and point Line Incidences and distinct distances via crossing numbers.

Arrangements of surface: Constructing arrangements, skeletons in arrangements, linear programming, planar point location search, Davenport-Schinzel Sequence.

Epsilon Nets: Epsilon Net theorem, weak epsilon nets and VC-Dimension, Colouring and covering problems.

Crossings: Crossing number of Graphs, Graph drawing and applications. Counting k-Sets and related bounds.

Dowkers theorem, Farys theorem, Fejes Toths theorems, Minkowski-Hlawka theorem, Koebe representation theorem, Erdos, Toran, Ramsey theorems; Szemeredis regularity lemma.

(b) Nil

(c) 4 hours

(d) Theory: 80% and assignment/project: 20%

(e) References:

1. J. Pach and M. Sharir: Combinatorial geometry and its algorithmic applications, Mathematical Surveys and Monographs, vol. 152, AMS, 2009
2. J. Pach and P. K. Agarwal: Combinatorial Geometry, Wiley-Interscience, . New York, 1995.
3. H. Edelsbrunner: Algorithms in Combinatorial Geometry, Series: Monographs in Theoretical Computer Science, EATCS Series, Vol. 10, Springer.
4. J. Matousek: Lectures on Discrete Geometry, vol. 212, Springer GTM Series.

B8. Topics in Algorithms and Complexity

(a) Objective of this course is to introduce a few topics of current interest in Algorithms and Complexity. The course may be organised by selecting a few topics from the list given below.

Randomised algorithms: Review of basic tools from probability theory— moments, deviations, Markov and Chebyshev's inequalities, Chernoff bounds, probabilistic method, Lovasz local lemma, method of conditional probabilities, etc; application to design and analysis of randomized algorithms for selected problems, such as quick-sort, min-cut, 2-SAT, matrix product, primality etc.

Probabilistic analysis of algorithms: Some examples, such as BP heuristics, Karp's partitioning algorithm for Euclidean TSP, etc.

On-line algorithms: On-line algorithms for selected problems, such as paging and load balancing.

Approximability of NP-optimization problems: A brief survey of some important algorithms including primate-dual algorithms, algorithms based on semidefinite programming and randomized rounding for selected problems; approximation classes and completeness.

Probabilistic Computation: Probabilistic Turing Machines and complexity classes, Pseudo-random number generators; Brief introduction to interactive proof systems and PCP Theorem with applications to approximability of NP-optimization problems.

Quantum Computation: A brief introduction to quantum computing and some important quantum algorithms.

Parametric Complexity: A brief introduction to parametric complexity with illustrative examples of fixed parameter tractability.

Computational Biology: A brief introduction to algorithms for molecular biology— basic concepts from molecular biology, algorithms for computational problems related to sequence alignments, DNA sequencing and phylogenetic trees.

(b) Pre-requisite -Probability and Stochastic Processes, Design and Analysis of Algorithms.

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. R. Motwani and P. Raghavan: Randomized Algorithms, Cambridge University Press, Cambridge, 1995.
2. S. Naor: Lecture Notes on Probabilistic Method in computer Science, Technion, 1993.
3. N. Alon and J. Spencer: The Probabilistic Method, Wiley Inter Science, New York, 1992.
4. D. S. Hchbaum (Ed.): Approximate Solutions of NP-hard problems, PWS Publishing Co., New York, 1997,
5. E. W. Mayr, H. T. Promel and A. Steger (Eds.): Lectures on Proof Verification and Approximation Algorithms, Springer- Verlag, Berlin, 1998.
6. M. Goemans: Lecture Notes on Advanced Algorithms, Fall, MIT/LCS/RSS-27, 1994.
7. D. P. Williamson: Lecture Notes on Approximation Algorithms, Cornell, Fall, 1998.
8. G. Ausiello, P, Crescenzi and others: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximation Properties, Springer-Verlag, Berlin, 1999.
9. A. Berthiaume: Quantum Computations, In: Complexity Theory Retrospective II, L. A. Hemaspcendra, and A. L. Selman (Eds.), Springer-Verlag, New York, pp. 23-51, 1997.
10. J. Gruska: Quantum Computing, McGraw Hill, New York, 1999

11. R. G. Downey and M. R. Fellows: Parameterized Complexity, Springer-Verlag, 1999.
12. J. Setubal and J. Meidenis: Introduction to Computational Molecular Biology, PWS Pub. Co., New York, 1997.
13. M. Waterman: Introduction to Computational Biology, Chapman and Hall, London, 1995.
14. R. Shamir and P. Naor: Lecture Notes on Algorithms for Molecular Biology, Tel Aviv Univ., Israel, 1998.
15. Selected recent papers.

B9. Computational Finance

(a) The course will provide a systematic introduction to the development, analysis and implementation of numerical methods for solving financial problems and will focus on computational methods for pricing and hedging equity and fixed-income derivatives. Three main areas are to be covered: pricing by formulas and approximations, pricing using lattices (one-, two-, and n-dimensional) and finite differences and pricing using Monte Carlo simulation. The following topics should be among those that are covered— pricing using exotic derivatives (such as barrier), Asian, lookback, and multi-asset options), pricing interest rate derivatives in the Heath-Jarrow-Morton and Libor Market Models, low discrepancy sequences for financial computations and the pricing of American options using simulation.

(b) Some background on relevant mathematics and programming skills.

(c) Four lectures per week

(d) Theory 70% and 30% for programming assignments.

(e) References:

1. R. Seydel: Tools for Computational Finance, 2nd edition, Springer-Verlag, New York, 2004.
2. P. Glasserman: Monte Carlo Methods in Financial Engineering, Springer-Verlag, New York, 2004.
3. W. Press, S. Teukolsky, W. Vetterling and B. Flannery, Numerical Recipes in C: The Art of Scientific Computing, 1997. Cambridge University Press, Cambridge, UK.
4. Available on-line at: http://www.nr.com/
5. A. Lewis: Option Valuation under Stochastic Volatility, Finance Press, Newport Beach, California, 2000.
6. A. Pelsser: Efficient Methods for Valuing Interest Rate Derivatives, Springer-Verlag, New York, 2000.

(f) Supplementary Information: Based on a course on “Computational Finance” given at Columbia University, http://www.columbia.edu/04/course.html

B10. Algorithmic Game Theory

(a) Algorithmic Game Theory combines algorithmic thinking with game-theoretic, or, more generally, economic concepts. The course will focus on some of the many questions at the interface between algorithms and game theory. Some of the topics that may be covered are as follows.

Games on Networks, congestion games, selfish routing in networks, Nash and Wardrop equilibria,

coordination ratios (price of anarchy), pricing network edges, network design with selfish agents.
Algorithmic Aspects of Equilibria: existence and complexity of equilibria (including Nash and cooperative), complexity of market equilibria, fast algorithms for specific games, games with incomplete information, evolutionary games.
Economic aspects of Internet routing: fairness, charging schemes, and rate control.
Mechanism Design: general principles, algorithmic mechanism design, distributed aspects, specific applications, e. g. multicast pricing, cost-sharing mechanisms.
Auctions: combinatorial auctions, frugality, auctions for digital goods, computational aspects of auctions.

(b) Design and analysis of algorithms, background on relevant mathematics, essentially discrete math and probability.

(c) Four lectures per week.

(d) There need not be any compulsory programming assignments, but, the teacher, at his/her discretion may give out assignments of different nature.

(e) References:

1. M. J. Osborne and A. Rubinstein: A Course in Game Theory, MIT Press, 1994.
2. N. Nisan, T. Roughgarden, E. Tardos and V. Vazirani (ed): Algorithmic Game Theory, Cambridge University Press, 2007.

(f) Supplementary Information: Design based on a course at Cornell University. http://www.cs.cornell.edu/courses/cs684/2004sp/

B11. Computational Complexity

(a) Review of machine models, RAM, Various kinds of Turing machines; Time and space complexities, central complexity classes, hierarchy and simulation results; Reduction between problems and completeness, examples of complete problems; Complexity of optimization problems and their approximability, circuits and non-uniform models of computation; Probabilistic computation and complexity classes;

Introduction to PCP theorem and its application to approximability; Uniform diagonalization, polynomial time hierarchy; Isomorphism conjecture, sparse sets, relativization, resource bounded measure and computational complexity, public-key cryptosystems, one-way functions, trapdoor-functions.

(b) Nil

(c) Four lectures per week

(d) Theory 100%

(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.

B12. Principles of Programming Languages

This subject is an elective for students with Computer Science background in the first semester. Students with non-Computer Science background can opt for this subject as an elective in the third semester.

(a) Theory:

Introduction: Overview of different programming paradigms e. g. imperative, object oriented, functional, logic and concurrent programming.

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

Imperative and Object Oriented 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 based on time constraints: C++/Java/OCAML/Lisp/Haskell/Prolog.

(b) Pre-requisite: Introduction to Programming.

(c) Three lectures and 1hands-on per week

(d) Theory 70% and Assignment 30%

(e) References:

1. Glynn Winskel, A Formal Semantics of Programming Languages: An Introduction, MIT Press.
2. John C. Mitchell, Foundations for Programming Languages, MIT Press.
3. Benjamin C. Pierce, Types and Programming Languages, MIT Press.
4. Daniel P. Friedman, Mitchell Wand and Christopher T. Haynes, Essentials of Programming Languages, Prentice Hall of India.
5. Ravi Sethi, Programming Languages: Concepts and C

B13. 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 Presbnrger arithmetics, incompleteness of first order number theory. Introduction to Modal and Temporal Logic with applications.

(b) Nil

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. C. L. Chang and R. C. T Lee: Symbolic Logic and Mechanical Theorem Proving, Academic Press, New York and London, 1973.
2. H. Enderton: A Mathematical Introduction to Logic, Academic Press, London, 1972.
3. M. Fitting: First-order Logic and Automated Theorem Proving, Springer, Berlin, 1990.
4. H. Gallier: Logic for Computer Science, John Wiley and Sons, New York, 1987.
5. G. E. Hughes and M. J. Cresswell: A New Introduction to Modal Logic Symbolic Logic, Routledge, 1996.
6. E. Mendelson: Introduction to Mathematical Logic, Van Nostrand, 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.

B14. Formal Aspects of Programming Languages and Methodology

(a) *Formal Syntax and Semantics:* Abstract and concrete syntax; Operational, denotational, axiomatic, algebraic and other formal semantics; Domain theory and fixed-point semantics; Lambda-calculus, type theory and functional languages.

Verification of Programs: Reasoning about programs, specification and correctness assertions, soundness and completeness. Verification of deterministic flow and structured programs, methods for proving partial and total correctness and their soundness and completeness.

Verification of nondeterministic programs— guarded commands, deductive system for partial and total correctness, soundness and completeness, proving fair termination.

Verifying program with procedures—recursive procedures with or without parameters.

Verifying concurrent and distributed programs— shared variable language, CSP language, deductive systems and their soundness and completeness, safety and liveness properties, fair termination etc.

(b) Co-requisite: B 13: Logic for Computer Science.

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. K. R. Apt and E. R. Olderog: Verification of Sequential and Concurrent Programs, Springer-Verlag, New York, 1991.
2. C. A. Gunter: Semantics of Programming Languages: Structures and Techniques, The MIT Press, Cambridge, 1992.
3. B. L Kurtz and K. Slonneger; Formal Syntax and Semantics of Programming Languages, Addison-Wesley, Reading, Mass., 1995.
4. J. Loeckx and K. Sieber: Foundations of Program Verification, John Wiley and Sons, Chichester, 1984.
5. N. Francez: Program Verification, Addison-Wesley, Reading, Mass., 1992
6. G. Winskel: Formal Semantics of Programming Languages: An Introduction, The MIT Press, Cambridge, 1993.

B15. Formal Methods in Computer Science— Selected Topics

(a) Objective of this course is to introduce a few topics concerning applications of formal methods for specification, design, development and verification of complex software and hardware systems. The course may be organized by giving an overview of formal methods in Computer Science and treating a few selected topics at some depth. Possible topics are given below.

Introduction and overview: Life cycle and traditional informal approach to software development, need for formal approach, survey of specification and formal software development, methods including algebraic and operational specifications, VDM, Z, various stages in the software development process, program transformation methodology, correctness proofs.

Models of Concurrency: Survey of various models for concurrent and parallel computation, such as labelled transition systems, process algebras like CCS and CSP, synchronization trees and languages, Petri nets, asynchronous transition systems, event structures etc.

Specification and Verification of reactive systems: Fair transition systems, linear time and branching time temporal logics; Omega-automata, Buchi, Muller, Street and Rabin automata; Decision problems, relation to LTL, LTL-satisfiability, tableaux construction and conversion to omega-automata. LTL model-checking using omega-automata; sate-explosion problem and symbolic model checking; deductive model checking—falsification diagram, safety transformation, fairness transformation and well-founded transformation, finding counter examples, generalized verification diagrams.

(b) Co-requisite: Logic for Computer Science.

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. R. D. Dowsing, V. J. Rayward-Smith and C. D. Walter: A First Course in Formal Logic and Its Applications in Computer Science, Computer Sc. Texts, Blackwell Scientific Publications, 1986.
2. C. A. R. Hoare: Communicating Sequential Processes, Prentice Hall, Englewood Cliffs, 1985.
3. M. Hennessy, Algebraic Theory of Processes, MIT Press, Cambridge, 1988.
4. C. B. Jones: Systematic Software Development Using VDM, International Series in Computer

Science, Prentice Hall, Englewood Cliffs, 1990.
5. R. P. Kurshan: Computer Aided Verification of Co-ordinating Processes: The Automata Theoretic Approach, Princeton Univ. Press, 1994.
6. Z. Manna and A Pnueli: The Temporal Logic of Reactive and Concurrent Systems: Specification, Springer-Verlag, Berlin, 1991.
7. Z. Manna and A. Pnueli: Temporal Verification of Reactive and Concurrent Systems: Safety, Springer-Verlag, Berlin, 1995.
8. K. L. McMillan: Symbolic Model Checking, Kluwer Academic Publishers, London, 1993.
9. A. R. G. Milner: Communication and Concurrency, Prentice Hall, Englewood Cliffs, 1989.
10. G. Rattray (Ed.): Specification and Verification of Concurrent Systems, Springer-Verlag, Berlin, 1999.
11. A. W. Roscoe: The Theory and Practice of Concurrency, Prentice Hall Series in Computer Science, Prentice Hall, Englewood Cliffs, 1998.
12. A. Diller: Z: An Introduction to Formal Methods, 2nd ed., Wiley, New York, 1994.
13. C. N. Dean and M. G. Hinchey (Eds.): Teaching and Learning Formal Methods, Academic Press, New York, 1996.
14. M. G. Hinchey and J. B. Bowen (Eds.): Applications of Formal Methods, International Series in Computer Science, Prentice Hall, Englewood Cliffs, 1995.
15. H. A. Partsch: Specification and Transformation of Programs: A Formal Approach to Software Development, Springer, Berlin, 1990.
16. S. Schneider: Concurrent and Real-Time Systems: The CSP Approach, Wiley, New York, 1999.
17. M. Vardi and A. Pnueli: Lecture Notes on Automata-Theoretic Approach to Automated Verification, Weizmann Institute, Fall, 1998.

B16. Logic Programming and Deductive Databases

(a) *Logic programming:* Introduction to four styles of programming -procedural, functional, logic-oriented and object-oriented; Foundation of logic programming review of relevant concepts from Logic-clausal form and Horn clause logic, skolemization, the Herbrand domain, unification, resolution and resolution strategies; Logic programming— data representation, operational views of unification and backtracking, the notion of logical variable, reversibility, non-logical features; Working with PROLOG, implementation issues, meta-level programming, constraint logic programming and other paradigms, practical exercises.

Deductive databases: Introduction to deductive databases— database as a model and a theory, queries and integrity constraints, negation as a failure, top-down and bottom-up query evaluation, semantics of negation and other related issues.

(b) Co-requisite: Logic for Computer Science.

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. S. Ceri, G. Gottlob and L. Tancq: Logic Programming and Databases, Springer, Berlin, 1990.
2. W. F. Clocksin and C. S. Mellish: Programming in PROLOG, 2nd ed., Springer-Verlag, Berlin, 1984.
3. S. K. Das: Deductive Databases and Logic Programming, Addison-Wesley, Reading, Mass.,

1992.
4. H. Gallaire and J. Menkar (Eds.) Logic and Databases, Plenum press, New York, 1978.
5. S. Gregory: Parallel Logic Programming in PARLOG. The Language and its Implementation, Addison-Wesley, England, 1987.
6. C. J. Hogger: Essentials of Logic Programming, Oxford Univ. Press, London, 1991.
7. R. Kowalski: Logic for Problem Solving, Elsevier North Holland, New York, 1979.
8. J. N. Sprinvey: Logic Programming; The Essence of PROLOG, Prentice Hall, New York, 1996.
9. J. W. Lloyd: Foundations of Logic Programming, 2nd ed., Springer, Berlin, 1987.
10. J. Minker (Ed.): Foundation of Deductive Databases, Kaufmann Publisher, New York, 1988.

B17. Topics in Algebraic Computation

(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) Nil

(c) Four lectures per week

(d) Theory 80% and 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-Calculus, Combinators and Functional Programming

(a) *Introduction:* Functional vs. imperative programming; Lambda-calculus the basis of all functional languages, lambda calculus as a formal system free and bound variables; Substitution, conversion rules, equality, extendibility, reduction and reduction strategies, Church-Rosser Theorem (statement only) and consequences, Combinators, lambda-calculus as a programming language-computability, Turing completeness (no proof), representing data and basic operations, truth values, pairs and tuples, natural numbers, predecessor operation, writing recursive functions, fixed-point combinators, Let expressions, lambda-calculus as a declarative language.

Types: Simply typed lambda-calculus, Church and Curry typing, polymorphism, most general types, strong normalization (no proof) and its negative consequence for Turing completeness, adding recursive operators.

Functional programming: Pure/impure functional languages, strict/non-strict functional languages (standard MUHASKELL); Functional programming (using MUHASKELL)— expressions, evaluation, functions and types; Type definitions and built-in types; Recursive definitions and structural reduction, infinite lists, further data structures, examples of functional programmes; Implementation issues.

(b) Nil

(c) Four lectures per week

(d) Theory 100%

(e) References:

1. H. P. Barendregt: The Lambda Calculus: Its Syntax and Semantics, North Holland, Amsterdam, 1984
2. R. S. Bird: Introduction to Functional Programming Using Haskell, Prentice Hall International, New York, 1998
3. R. S. Bird and P. Wadles: Introduction to Functional Programming, Prentice Hall, Englewood Cliffs, 1988

4. C. Hankin: Lambda Calculi A Guide for Computer Scientists, Clarendon Press, Oxford, 1994
5. J. R. Hindley and J. P. Seldin: Introduction to Combinators and Lambda Calculus, Cambridge Univ. Press, London, 1986
6. P. Jones: The Implementation of Functional Programming Languages, Prentice Hall, Englewood Cliffs, 1987
7. L. Paulson: ML for the Working Programmer, Cambridge Univ. Press, London, 1991
8. C. Read: Elements of Functional Programming, International Computer Science Series, Addison-Wesley, California, 1989
9. G. E. Revesz: Lambda Calculus, Combinators and Functional Programming, Cambridge University Press, London, 1988
10. K. Sikdar: Lambda Calculus and Combinators, Lecture Notes for Winter School on Logic Programming and Related Topics, Stat-Math Unit, ISI, Calcutta, January 3-21, 1994
11. J. Ullman: Elements of ML Programming, Prentice Hall, Englewood Cliffs, 1994.

B19. Pattern Recognition and Image Processing

(a) *Pattern Recognition:* Introduction, overview of different approaches, decision boundaries, discriminant functions (linear and non-linear), Bayesian classification, training and test sets, parametric and nonparametric learning, minimum distance classifiers, k-NN rule, unsupervised learning, basic hierarchical and non-hierarchical clustering algorithms, dimensionality reduction, similarity measures, feature selection criteria and algorithms, principal components analysis, some applications.

Image Processing: Introduction, image definition and its representation, neighbourhood metrics, image processing systems, 2-D orthogonal transformations of images (DFT, DCT, HT, KLT), enhancement, contrast stretching, histogram specification, local contrast enhancement, smoothing and sharpening, spatial/ frequency domain filtering, segmentation, pixel classification, greylevel thresholding, global/local thresholding, edge detection operators, region growing, split/merge techniques, image feature/primitive extraction, line detection, border following, Hough transform, medial axis transform, skeletonization/ thinning, shape properties, compression, Huffman coding, block truncation coding, run-length coding, some applications.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. R. O. Duda, P. E. Hart and D. G. Stork: Pattern Classification and Scene Analysis, 2nd ed., Wiley, New York, 2000.
2. J. T. Tou and R. C. Gonzalez: Pattern Recognition Principles, Addison-Wesley, London, 1974.
3. E. Gose, R. Johnsonbaugh and S. Jost: Pattern Recognition and Image Analysis, Prentice Hall of India, New Delhi, 1999.
4. R. C. Gonzalez and R. E. Woods: Digital Image Processing, Addison-Wesley, California, 1993.
5. S. K. Pal and D. Dutta Majumder: Fuzzy Mathematical Approach to Pattern Recognition, Wiley (Halsted Press), New York, 1986.
6. T. Y. Young and K. S. Fu: Handbook of Pattern Recognition and Image Processing, Vols. 1and 2, Academic Press, New York, 1986.

7. W. Niblack: Digital Image Processing, Prentice Hall, Englewood Cliffs, 1986.
8. A. Jain: Fundamentals of Digital Image Processing, Prentice Hall of India, New Delhi, 1989.
9. K. R. Castleman: Digital Image Processing, Prentice Hall, Englewood Cliffs, 1996.
10. B. Chanda and D. Dutta Majumder: Digital Image Processing and Analysis, Prentice Hall of India, New Delhi, 2000.

B20. 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) Nil

(c) Four lectures per week

(d) Theory 80% and 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.

B21. Artificial Intelligence

(a) *Introduction:* Cognitive science and perception problem, a brief history of AI and its applications.
Languages for AI: PROLOG, LISP; Production system and matching: Problem space, natural constraints, problem representation, problem reduction.
Search techniques: Search space, state space search, heuristic search, pattern directed search, planning, control strategies and implementation, constraint satisfaction.
Knowledge representation: Propositional and predicate logic, rule-base, semantic net, conceptual graph, frames, scripts, relational database, knowledge acquisition and learning.
Reasoning: Logical reasoning, theorem proving, probabilistic reasoning, approximate reasoning, fuzzy reasoning, reasoning about action and time, resources bounded reasoning.
Problem solving: Inference engines, expert system, plan generating system, hierarchical planning, game playing, means-ends analysis, deduction system, blackboard approach.
Tools: Fuzzy logic, artificial neural network, genetic algorithm; Some applications: Natural language understanding, vision, speech understanding, MYCIN.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. E. Rich: Artificial Intelligence, McGraw Hill, New Delhi, 1983.
2. P. H. Winston: Artificial Intelligence, Addison-Wesley, Reading, Mass., 1984.
3. E. Chamiak and D. McDermott: Introduction to Artificial Intelligence, Addison-Wesley, Reading, Mass., 1985.
4. S. L. Tanimoto: The Elements of Artificial Intelligence, Computer Science Press, New York, 1987.
5. G. F. Luger: Artificial Intelligence: Structures and Strategies for Complex Problem Solving, The Benjamin Publ. Co., New York, 1993.
6. S. J. Russel and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, Englewood Cliffs, 1994.
7. N. J. Nilsson: Artificial Intelligence: A new synthesis, Morgan Kauffman, San Mateo, 1998.

B22. Internet and Multimedia Technologies

(a) Internet
Introduction: Overview and evolution of Internet programming and application tools, searching and browsing tools.
Markup Languages and its application on the web: HTML, XML and related concepts.
Basic Java programming: Features of Java language— brief history of Java, concept of Java VM, basic structure of Java programming; GUI programming— basics of Java abstract windowing toolkit, event handling and swing programming, applet programming; Java beans and Java IDE.
Communication protocol: TCP/IP, IP addressing and domain registration and related concepts.
Application level technology: HTTP, Web server, browsing, firewall.
Search mechanisms: Search engine, crawler technology, filtering technology, content based searching, agent technology, Internet robot.

Advanced Internet applications: Data and Web mining, e-commerce, distributed objects—component object model, common object request broker architecture; Web security.

Multimedia

Introduction and overview.

User interface design: User interface styles, visual design.

Multimedia authoring: Design principles, use of authoring tools.

Artificial reality and interactive multimedia.

Multimedia and interactive hardware.

Multimedia database: Indexing, searching and retrieval.

Multimedia applications: CBT/CAI, Multimedia e catalogues, encyclopaedia etc.

(b) Nil

(c) Three lectures and two practical/Tutorial per week

(d) Theory 60% and Assignment 40%

(e) References:

1. C. S. Horstmann and G. Cornell: Core Java 2, Vol. 1: Fundamentals, Prentice Hall, Englewood Cliffs, 1998.
2. W. R. Stevens: TCP/IP illustrated, Vol. I: The Protocols, Addison-Wesley, Reading, Mass., 1994.
3. J. D. Foley et. al.: Computer Graphics: Principles and Practice, Addison-Wesley, Reading, Mass., 1993.
4. R. Orfali and D. Harkey: Client/Server Programming with Java and Corba, 2nd ed., John Wiley, New York, 1997.
5. D. S. Ray and E. J. Ray: Mastering HTML 4, Premium, New Delhi, 2000.
6. B. Marchal: XML by Example, 2nd ed., QUE, 1999.
7. R. Abernethy, R. C. Morin and J. Chahin: COM/DCOM Unleashed, SAMS, 1999.
8. Related literatures/tutorials available in Internet and multimedia.

B23. Computer Graphics

(a) *Introduction:* Objective, applications, GKS/PHIGS, normalized co-ordinate system, aspect ratio.

Graphics system: Vector and raster graphics, various graphics display devices, graphics interactive devices, segmented graphics, attribute table.

Raster scan Graphics: Line drawing algorithms, circle/ellipse drawing algorithms, polygon filling algorithms.

Geometric transformation: Homogeneous co-ordinate system, 2D and 3D transformations, projection— orthographic and perspective.

Curve and Surfaces: Curve approximation and interpolation, Lagrange, Hermite, Bezier and B-Spline curves/surfaces and their properties, curves and surface drawing algorithms.

Geometric modelling: 3D object representation and its criteria, edge/vertex list, constructive solid geometry, wire-frame model, generalized cylinder, finite element methods.

Clipping: Window and viewport, 2D and 3D clipping algorithms.

Hidden line and hidden surfaces: Concept of object- and image-space methods, lines and surface removal algorithms.

Intensify and colour models: RGB, YIQ, HLS and HSV models and their conversions, gamma correction, halftoning.

Rendering: illumination models, polygon mesh shading, transparency, shadow, texture.

Some advance topics/applications:

(i) Animation and morphing, (ii) Virtual reality, (iii) User-interface design, (iv) Fractal graphics, (v) Multimedia authoring, (vi) 3D visualization.

(b) Nil

(c) Three lectures and two tutorial per week

(d) Theory 60% and Assignment 40%

(e) References:

1. W. K. Giloi: Interactive Computer Graphics: Data Structure, Algorithms, Languages, Prentice Hall, Englewood Cliffs, 1978.
2. W. M. Newman and R. F. Sproull: Principles of Interactive Computer Graphics, McGraw Hill, New Delhi, 1979.
3. J. D. Foley et al.: Computer Graphics, 2nd ed., Addison-Wesley, Reading, Mass., 1993.
4. D. Hearn and P. M. Baker: Computer Graphics, 2nd ed. Prentice Hall of India, New Delhi, 1997.
5. F. S. Hill: Computer Graphics, McMillan, New York, 1990.
6. D. P. Mukherjee: Fundamentals of Computer Graphics and Multimedia, Prentice Hall of India, New Delhi, 1999.

B24. Computer Vision

(a) *Introduction:* Machine vision systems, optics and lenses, image sensors, human vision and Neuro-visual model; Marr’s paradigm; Imaging geometry— world co-ordinate system and camera co-ordinate system, co-ordinate transformations, projection geometry, camera calibration, radiometry.

Early processing and image filtering: Noise removal, region segmentation, concept of primal sketch, scale space, edge detection and localization, edge linking, Hough transform, corner and junction detection.

Reflectance map and photometric stereo: Image brightness and radiometry, image formation and surface reflectance under different conditions, reflectance map and bidirectional reflectance distribution function, photometric stereo recovering albedo and surface orientation, shape from shading.

Range measurement and recovering scene geometry: Binocular technique— stereo pair, epipolar line and plane, Stereo matching, photogrammetry, monocular technique— texture processing and shape from texture, depth from focusing and symmetry, different range finder (active) -laser range finder, light-stripe method.

Motion estimation: Motion field, optical flow— smoothness, boundary conditions, discontinuities of optical flow, block based method, pre-recursive method, Bayesian method, motion segmentation method, motion from points and lines, token tracking, stereo and motion tracking, use of Kalman filter, focus of expansion, structure from motion, motion compensated filtering and restoration, video compression, active and passive surveillance.

Representation and analysis of polyhedral scene: Understanding line drawings, gradient and dual space, generalized cylinder, volumetric representation, edge and junction labelling; Labelling and recognition of scene objects; Construction of model-base and visual learning, model based recognition system— Acronym, model based recognition from sparse range data, 3D model based vision system, scene understanding.

Special systems for computer vision: Visual information processing architecture, language and control

Some applications (but not restricted to): (i) Automated guided vehicle, (ii) Face and gesture recognition, (iii) Vision based inspection system, (iv) Grasping system, (v) Automated visual inspection.

(b) Pre-requisite: As suggested by the teacher.

(c) Three lectures and one tutorial per week

(d) Theory 80% and Assignment 20%

(e) References:

1. D. H. Ballard and C. M. Brown: Computer Vision, Prentice Hall, New York, 1986.
2. R. M. Haralick, L. G. Shapiro: Computer and Robot Vision, Addison-Wesley Pub Co, reading, Mass., 1992.
3. Y. Shirai: Three-Dimensional Computer Vision, Springer-Verlag, Berlin, 1988.
4. B. K. P. Horn: Robot Vision, MIT Press, Cambridge, 1986.
5. O. Faugeras: Three-Dimensional Computer Vision: A Geometric Viewpoint, MIT Press, Cambridge, 1993.
6. B. K. P. Horn and M. J. Brooks: Shape from Shading, M. I. T. Press, Cambridge, 1989.
7. R. Jain, R. Kasturi and B. Schuck: Machine Vision, McGraw Hill Higher Education, New York, 1995.
8. E. R. Davis: Machine Vision: Theory, Algorithms and Practicalities, Academic Press, New York, 1996.
9. M. Sonka, V. Hlavac and R. Boyle, Image Processing: Analysis and Machine Vision, PWS Pub. Co., London, 1998.

B25. Advanced Image Processing

(a) *Introduction:* Image formation -geometric and photometric models, Digitization.

Enhancement/restoration: Multi-scale/multi-resolution enhancement, edge-preserving smoothing, anisotropic diffusion, least square restoration, constrained least-squares restoration, Wiener filter.

Segmentation: Model-based— facet model, active contour, discrete and probabilistic relaxation, watershade algorithm, edge detection/linking, semantic region grouping, interactive segmentation.

Compression: First/second generation compression, fractal based compression, wavelet based compression.

Properties/Feature extraction: Temporal and textural features— moments, graylevel co-occurrence matrix, pattern spectrum; structural features -Fourier descriptor, polygonal approximation.

Image Analysis and recognition: Shape matching, shape metric, texture analysis, relaxation (probabilistic and fuzzy) technique, graph isomorphism, multi-resolution and multi-scale image analysis, image understanding.

Colour image processing: Colour model, enhancement, segmentation.

Image databases: Attribute list, relational attributes, indexing, storage and retrieval, content based retrieval; Use of advanced tools— fractals, genetic algorithm, Markov model, mathematical morphology, neural networks, wavelets;

Use of Advanced Tools

Some applications (from the following but not restricted to): (i) Document image processing, (ii) Biomedical image processing, (ii) Digital video, (iv) Fingerprint classification, and (v) Digital water-marking.

(b) Pre-requisite: Pattern Recognition and Image Processing.

(c) Three lectures and two tutorials per week

(d) Theory 80% and Assignment 20%

(e) References:

1. A. R. Rao: Taxonomy for Texture Description, Springer-Verlag, Berlin, 1990.
2. R. M. Haralick and L. G. Shapiro; Computer and Robot Vision, Vol. 1and 2, Addison-Wesley, Reading, Mass., 1992.
3. A. Rosenfeld and A. C. Kak; Digital Picture Processing, 2nd ed., (Vol. 1and 2), Academic Press, New York, 1982.
4. B. B. Chaudhuri and D. Dutta Majumder: Two-tone Image Processing and Recognition, Wiley-Eastern, New Delhi, 1993.
5. A. Blake and A. Zisserman: Visual Reconstruction, MIT Press, Cambridge, 1987.
6. B. Bhanu: Genetic learning for Adaptive Segmentation, Kluwer Academic Press, Boston, 1994.
7. W. K. Pratt: Digital Image Processing, 2nd ed., John Wiley, New York, 1992.
8. A. N. Netravall and B. G. Haskell: Digital Pictures, 2nd ed., Plenum Press, 1995.
9. K. Sayood: Data Compression, Morgan Kaufmann, San Mateo, 1986.
10. Y. Fisher: Fractal Image Compression: Theory and Application, Springer-Verlag, Berlin, 1995.
11. H. C. Andrews and B. R. Hunt: Digital Image Restoration, Prentice Hall, Englewood Cliffs, 1977.
12. M. Vetteerli and J. Kovacevic: Wavelet and Sub-Band Coding, Prentice Hall, EC, 1995
13. A. B. Watson: Digital Images and Human Vision, MIT Press, Cambridge, 1993
14. C. A. Glasbey and G. H. Horgen: Image Analysis for Biomedical Sciences, John Wiley, New York, 1995
15. L. Niang: Fractal Imaging, Academic Press, New York, 1997
16. S. Khoshafian and A. B. Baker: Multimedia and Imaging Databases, Morgan Kaufmann, San Mateo, 1996.
17. B. Chanda and D. Dutta Majumder: Digital Image Processing and Analysis, Prentice Hall of India, New Delhi, 2000.
18. S. K. Pal, A. Ghosh and M. K. Kundu: Soft Computing for Image Processing, Physica Verlag (Springer), Heidelberg, 1999.
19. M. Sonka, V. Hlavac and R. Boyle, Image Processing: Analysis and Machine Vision, PWS Pub. Co., London, 1998.

B26. Fuzzy Logic and Applications

(a) Brief overview of crisp sets; the notion of fuzziness; what, why and when to apply fuzzy set; operations on fuzzy sets; fuzzy numbers.

Crisp relations, fuzzy relations, Max*-composition of fuzzy relation; Max*-transitive closure; probability measures of fuzzy events; fuzzy expected value.
Approximate reasoning, different methods of role aggregation and defuzzification.
Fuzzy measures— belief, plausibility and their properties; Dempster's role of combination; consonant body of evidence— possibility, necessity.
Measures of uncertainty— axiomatic formulation of Hartley information, Shannon's entropy, concepts of joint and conditional entropy and their properties; measures of non-specificity; measures of dissonance and confusion; fuzziness measures.
Fuzzy geometry.
Applications to some selected topics like pattern recognition, image processing, computer vision, optimization, control, data mining.
Integration with other computing paradigm.

(b) Nil

(c) Four lectures per week

(d) Theory 70% and Assignment 30%

(e) References:

1. G. J. Klir and T. A. Folger: Fuzzy Sets, Uncertainty, and Information, Prentice Hall, Englewood Cliffs, 1988.
2. A. Kandel: Fuzzy Mathematical Techniques With Applications, Addison-Wesley, Englewood Cliffs, 1986.
3. J. C. Bezdek and S. K. Pal (Eds.): Fuzzy Models for Pattern Recognition— Methods that Search for Structures in Data, IEEE Press, Los Alamos, California, 1992.
4. S. K. Pal and D. Dutta Majumder: Fuzzy Mathematical Approach to Pattern Recognition, John Wiley (Halsted Press), New York, 1986.
5. M. M. Gupta: Fuzzy Mathematical Models with Applications to Engineering and Management Science, North Holland, Amsterdam, 1988.
6. A. Kaufmann: Introduction to Theory of Fuzzy Subsets, Academic Press, New York, 1975.
7. H. Zimmermann: Fuzzy Set Theory and Its Application, 2nd ed., Kluwer, Boston, 1990.
8. T. J. Ross: Fuzzy Logic With Engineering Applications, McGraw Hill, Singapore, 1997.
9. J. C. Bezdek, J. M. Keller, R. Krishnapuram, and N. R. Pal: Fuzzy Models and Algorithms for Pattern Recognition and Image Processing, Kluwer Academic Publisher, Boston, 1999.
10. G. J. Klir and Bo Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice hall, Englewood Cliffs, 1995.

B27. Neural Networks and Applications

(a) Introduction to neural networks, threshold logic, circuit realization.
Introduction to biological neural networks, significance of massive parallelism.
Perceptron, perceptron learning rule and its convergence, multilayered perceptron, learning algorithms, function approximation, generalization, VC-dimension.
Regularization networks, RBF networks.
Recurrent networks; Hopfield model, pattern retrieval process, application to optimization problems, Simulated annealing, mean-field annealing, Boltzmann machine and its learning.
Self-organizing systems, Hebbian and competitive learning, Kohonen's self-organizing map,

learning vector quantization, principal component analysis networks, adaptive resonance theory.
Temporal learning, backpropagation through time, temporal backpropagation, real-time recurrent learning (RTRL).
Architecture optimization; Hardware realization.
Applications in selected topics from pattern recognition, image processing, computer vision, natural language processing, control, forecasting.
Advanced/related topics such as cellular neural networks, support vector machine, neuro-fuzzy computing and other hybridization, independent component analysis.

(b) Nil

(c) Four lectures per week

(d) Theory 70% and Assignment 30%

(e) References:

1. J. Hertz, A. Krogh, and R. G. Palmer: Introduction to the Theory of Neural Computation, Addison-Wesley, California, 1991.
2. Y. H. Pao: Adaptive Pattern Recognition and Neural Networks, Addison-Wesley, Reading, Mass., 1989.
3. P. D. Wassermann: An Introduction to Neural Computing: Theory and Practice, Van Nostrand Reinhold, New York, 1989.
4. J. M. Zurada: Introduction to Artificial Neural Systems, West Publishing Co., St. Paul, Minnesota, 1992.
5. T. Kohonen: Self-Organization and Associative Memory, Springer-Verlag, Berlin, 1988.
6. J. A. Anderson and E. Rosenfeld: Neuro-Computing: Foundation of Research, MIT Press, Cambridge, 1988,
7. Simon Haykin: Neural Networks: A Comprehensive Foundation, 2nd ed., Prentice Hall, New Jersey, 1999.
8. S. K. Pal and S. Mitra: Neuro-Fuzzy Pattern Recognition: Methods in Soft Computing, John Wiley, New York, 1999.
9. B. Yegnarayana: Artificial Neural Networks, Prentice Hall of India, New Delhi, 1999.

B28. Advanced Pattern Recognition

(a) Bayes classification, error probability, error bounds, Bhattacharya bounds, error rates and their estimation, parametric and nonparametric learning, density estimation, estimation of mixture distributions, classification trees.
Unsupervised classification, split/merge techniques, advanced hierarchical clustering algorithms, cluster validity, set estimation, optimal and suboptimal feature selection algorithms, k-NN rule and its error rate.
Syntactic approach to pattern recognition.
Neural network models for pattern recognition; learning, supervised and unsupervised classification, stochastic learning algorithm, feature analysis, fuzzy set theoretic models for pattern recognition.
Some advanced topics with applications, (e.g., neuro-fuzzy approach, genetic algorithms, data mining, case-based reasoning). Use of PR software.

(b) Pre-requisite: Pattern Recognition and Image Processing.

(c) Four lectures per week

(d) Theory 75% and Assignment 25%

(e) References:

1. K. Fukunaga: Introduction to Statistical Pattern Recognition, 2nd eel, Academic Press, New York, 1990.
2. P. A. Devijver and J Kittler: Pattern Recognition: A Statistical Approach, Prentice Hall, London, 1982.
3. J. Sklansky and G. N. Wassel: Pattern classifiers and Trainable Machines, Springer, New York, 1981.
4. M. R. Anderberg: Cluster Analysis for Applications, Academic Press, New York, 1973.
5. A. K. Jain and R. C. Dubes: Algorithms for Clustering Data, Prentice Hall, Englewood Cliffs, 1988.
6. R. C. Gonzalez and M. G. Thomason: Syntactic Pattern Recognition: An Introduction, Addison-Wesley, Reading, Mass., 1978.
7. K. S. Fu: Syntactic Pattern Recognition and Applications, Prentice Hall, Englewood Cliffs, 1982.
8. J. C. Bezdek: Pattern Recognition With Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.
9. G. J. Klir and B. Yuan: Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice Hall, Englewood Cliffs, 1995.
10. B. D. Ripley: Pattern Recognition and Neural Networks, Cambridge University Press, London, 1996.
11. Y-H. Pao: Adaptive Pattern and Neural Networks, Addison-Wesley, New York, 1989.
12. R. J. Schalkoff: Pattern Recognition: Statistical, Structural and Neural Approaches, John Wiley, New York, 1992.
13. S. K. Pal and S. Mitra: Neuro-Fuzzy Pattern Recognition: Methods in Soft Computing, John Wiley, New York, 1999.
14. S. Theodoridis and K. Koutroumbas: Pattern Recognition, Academic Press, San Diego, 1999.
15. S. K. Pal and P. P. Wang (Eds.): Genetic Algorithms for Pattern Recognition, CRC Press, Boca Raton, 1996.
16. S. K. Pal, T. S. Dillon and D. S. Yeung (Eds.): Soft Computing in Case Based Reasoning, Springer, London, 2001.
17. S. K. Pal and A. Pal (Eds.): Pattern Recognition: From Classical to Modern Approaches, World Scientific, Singapore, 2001.

B29. Analysis of Remote Sensing Images

(a) Physics of remote sensing, characteristics of satellites, sensors, earth’s surface, and atmosphere.
Registration of images and maps, sources of errors, like geometric errors and radiometric errors, and their corrections. Methodologies for interpretation of remotely sensed image data — image processing and pattern recognition.
Introduction to GIS, digitization and vectorization, use of GIS packages, analysis of remotely sensed data using GIS. Advanced topics like digital terrain mapping, object identification, multi-source data integration (fusion), hyper-spectral imaging and topics of current research interest.

(b) Pre-requisite: Pattern Recognition and Image Processing.

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. J. A. Richards and X. Jia: Remote Sensing Digital Image Analysis: An Introduction, 3rd ed., Springer-Verlag, Berlin, 1998.
2. C. Elachi: Introduction to the Physics and Techniques of Remote Sensing, Wiley, New York, 1987.
3. R. Laurini and D. Thompson: Fundamentals of Spatial Information Systems, Academic Press, London, 1992.
4. R. A. Schowengerdt: Remote Sensing: Models and Methods for Image Processing, Academic Press, New York, 1997.
5. F. F. Sabins: Remote Sensing, W. H. freeman, New York, 1996.
6. T. Lillesand: Remote Sensing and Image Interpretation, 4th ed., John Wiley and Sons, New York, 1999.
7. D. L. Hall: Mathematical Techniques in Multi-Sensor Data Fusion, Artech House, Boston, 1992.

B30. Document Processing and Retrieval

(a) *Document generation:* Curve drawing -Bezier Polynomial, Splines, Digital Character generation and Fontographics, Analog and Digital Halftoning, Document Layout creation and Page Makeup, Multimedia content creation, Static and Dynamic Website document creation.

Document processing and analysis: Document noise cleaning, Binarization, Automatic Layout recognition and segmentation, Logical structure analysis and recognition. Colour document processing, Text and hypertext data compression, Document image data compression, Graphics analysis and recognition in maps, drawings and diagrams. Lines, curves, logo recognition.

Optical Character Recognition (OCR): Skew detection, Thinning and Skeletonization Line, Word and Character segmentation, Character size normalization, Feature detection, Supervised and unsupervised classification, Tree classifier, Maximum Likelihood method, Minimum distance, K-nearest neighbour classifier, Bayes' classifier, Hidden Markov Model, Support vector machine, Neural Net classifier, Hand- printed Character recognition, Table form document, On-line and offline handwritten character recognition, Multiscript OCR systems.

Applications: Signature verification, Postal Address reading system, Table-form reading system, Mathematical expression, Chemical equation, Table recognition.

Information retrieval (IR): indexing, text-based information retrieval techniques, content based information retrieval (CBIR), multimedia information retrieval, multimodal query formulation/ decomposition, relevance judgment/feedback, evaluation techniques.

(b) Nil

(c) Three classes and one Tutorial per week

(d) Theory 70% and Assignment 30%

(e) References:

1. L. O'Gorman and R. Kasturi: Document Image Analysis, IEEE Computer Society Press, Los Alamitos, 1995.
2. H. S. Hou: Digital document Processing, Wiley-interscience, New York, 1983.
3. H. S. Baird, H. Bunke and K. Yamamoto: Structured document image analysis, Springer-Verlag, Berlin, 1992.
4. B. B. Chaudhuri and D. Dutta Majumder, Two tone Image Processing and Recognition, Wiley Eastern, New Delhi, 1993.
5. R. Baeza-Yates, B. Ribeiro-Neto, Modem Information Retrieval, Addison-Wesley Pub Co, Reading MA, 1999.
6. W. Frakes, R. Baeza-Yates , Information Retrieval: Data Structures and Algorithms, Prentice Hall, New Jersey, 1992.

B31. Data Mining

(a) *Introduction:* Introduction to data mining and knowledge discovery from databases. Scalability issues of data mining algorithms.

Introduction to Data warehousing: General principles, modelling, design, implementation, and optimization.

Data preparation: Preprocessing, sub-sampling, feature selection.

Classification and prediction: Bayes learning, discriminant analysis, decision trees, CART, C4.5 etc, neural learning, support vector machines, active learning. Combination of classifiers/ ensemble learning.

Associations, dependence analysis, correlation, rule generation— a priori algorithm, FP Trees etc. and evaluation.

Cluster analysis and deviation detection: Partitioning algorithms, density based algorithms, hierarchical algorithms, model based algorithms, grid based algorithms, graph theoretic clustering etc.

Temporal and spatial data mining: Mining complex types of data. Visualization of data mining results.

Advanced topics: High performance computing for data mining, distributed data mining, soft-computing tools for data mining.

Applications of data mining in bioinformatics, information retrieval, web mining, image and text mining.

(b) Nil

(c) Four lectures per week

(d) Theory 70% and Assignment 30%

(e) References:

1. J. Han, M. Kamber: Data Mining: Concepts and Techniques, Morgan Kaufmann, 2000
2. D. J. Hand, H. Mannila and P. Smyth: Principles of Data Mining, MIT Press, 2000
3. M. Berry and G. Linoff: Mastering Data Mining, John Wiley & Sons, 2000.
4. A. K. Pujari: Data Mining Techniques, Sangam Books Ltd., 2001.

B32. Computational Molecular Biology and Bioinformatics

(a) *Objectives:* To develop an understanding of the main algorithmic approaches used in solving computational problems that arise in the analysis of biomolecular data (such as DNA/RNA/amino acid sequences, mass spectra of proteins, whole genomes, or gene expression levels).

Specific problems to be covered include sequencing and assembly, multiple sequence alignment, phylogenetic reconstruction, and whole-genome comparisons and evolution. The emphasis throughout is on algorithmic design and analysis, including proofs of correctness and new designs, using both combinatorial and statistical approaches.

Details of topics to be covered: *Sequence Alignments:* Global alignments (Needleman-Wunsh), 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, 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.

(b) an interest in both computational methods and molecular biology and evolution; and a strong background in one of algorithms or (evolutionary) molecular biology and some reasonable acquaintance with the other.

(c) Three lectures per week and three hours of laboratory.

(d) Theory 70% and Assignments 30%

(e) References:

1. C. Setubal and J. Meidanis: Introduction to Computational Molecular Biology, PWS Publishing Company, Boston, 1997.
2. P. A. Pevzner: Computational Molecular Biology -An Algorithmic Approach, MIT Press, 2000.
3. R. Durbin, S. R. Eddy, A. Krogh and G. Mitchison: Biological Sequence Analysis— Probabilistic Models of Proteins and Nucleic Acids, Cambridge University Press, 1998.
4. D. Gusfield: Algorithms on Strings, Trees, and Sequences, Cambridge University Press, USA, 1997.
5. H. Lodish, A. Berk, S. L. Zipursky, P. Matsudaira, D. Baltimore and J. Darnell: Molecular Cell Biology, W. H. Freeman, USA, 2000.
6. C. -I. Branden, J. Tooze: Introduction to Protein Structure, Garland Publishing, 1998.

B33. Computer Architecture

(a) *Introduction:* Evolution of computer architecture, desired properties of the instruction set of a computer, instruction formats, addressing modes, architectural classifications based on multiplicity of data and instruction (SISD, SIMD, MISD and MIMD structures). CISC versus

RISC architectures; Performance metric — different approaches.
Pipelining: Basic concepts, Performance of a static linear pipeline, instruction pipelining, hazards (structural, data and control hazards) and their remedies, instruction level parallelism (ILP). Super pipelining, super scalar processing, vector processing and pipelined vector processing.
Memory System: Memory hierarchy, Cache memory — fundamental concepts, reducing cache misses and cache miss penalty. Interleaved memory, virtual memory.
Interconnection Networks: Static vs. dynamic networks, desirable characteristics, example of popular static interconnection networks, dynamic networks— non-blocking, blocking, blocking rearrangable networks, unique full access multi-stage interconnection networks, examples.
Multiprocessors: Centralized shared-memory architectures, distributed shared-memory architectures, synchronization issues, models of memory consistency; core architecture.

(b) Pre-requisite: Computer Organization.

(c) Four lectures per week (including tutorials).

(d) Theory 90% and Assignment 10%

(e) References:

1. D. A. Patterson and J. L. Hennessey: Computer Organization and Design: Hardware-Software Inter- face, Morgan Kaufmann, San Mateo, 1999.
2. J. L. Hennessey and D. A. Patterson: Computer Architecture: A Quantitative Approach, Morgan Kaufmann, San Mateo, 1999.
3. W. Stallings: Computer Organization and Architecture, Prentice Hall, New Jersey, 1999.
4. K. Hwang and F. A. Briggs: Computer Architecture and Parallel Processing, Tata McGraw Hill, New Delhi, 1984.

B34. VLSI Design and Algorithms

(a) *Introduction:* VLSI design, design styles and parameters, popular technologies.
Logic synthesis: Logic synthesis with nMOS, CMOS, DCVS and PLAs; Pass transistor vs. ratio logic, transit time, clocking, scaling. 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.
Layout synthesis: Design rules, partitioning, placement and floor planning, routing, FPGAs; CAD tools— MAGIC, XACT, VPR, etc.

(b) Pre-requisite: Computer Organization.

(c) Four lectures per week

(d) Theory 75% and 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. A. Mukherjee: Introduction to CMOS VLSI, Prentice Hall, Englewood Cliffs, 1993.

4. N. Weste and K. Eshraghian: Principles of CMOS Design, 2nd ed., Addison-Wesley, Reading, Mass., 1993.
5. C. Mead and L. Conway: Introduction to VLSI Systems, Addison-Wesley, Reading, Mass., 1980.
6. R. K. Brayton et al: Logic Minimization for VLSI Synthesis, Kluwer Academic Publishers, Boston, 1984.
7. D. Gajski, N. Dutt et al: High Level Synthesis: Introduction to Chip and System Design, Kluwer Academic, Boston, 1992.
8. M. Sarrafzadeh and C. K. Wong: AN Introduction to VLSI Physical Design, McGraw Hill, New York, 1996.
9. N. Sherwani: Algorithms for VLSI Physical Design Automation, Kluwer Academic, Boston, 1999.
10. B. T. Preas and M. Lorenzetti: Physical Design automation of VLSI Systems, Benjamin Cummings Pub., 1988.
11. T. Ohtsuki (ed): Layout Design and Verification, North Holland, Amsterdam, 1986.

B35. Parallel Processing: Architectures and Algorithms

(a) *Introduction:* Parallelism in uniprocessor System, memory-interleaving, pipelining and vector processing, parallel computer structures, architectural classifications, parallel computer models: PRAM and VLSI complexity models, program properties: conditions of parallelism, program partitioning and scheduling, granularity and scalability.

System interconnect architectures: Static interconnection networks array, tree, mesh, pyramid, hypercube, cube-connected-cycles, butterfly, Cayley graphs; Dynamic interconnection networks crossbar, Clos network, multistage interconnection networks, blocking, non-blocking and rearrangeable operations, properties and routing.

Networked computers as a multi-computer platform, basics of message-passing, computing using work- station clusters, Software tools.

Parallel algorithms and their mapping on different architectures:

(i) Arithmetic computations: Addition, multiplication, FFT, DFT, Polynomial multiplication, convolution, evaluation and interpolation.
(ii) Matrix operations: Transpose, multiplication, inversion, eigenvalue computation.
(iii) Numerical applications: Solving systems of linear equations, finding roots of non-linear equations, solving partial differential equations.
(iv) Sorting: Theoretical bounds, sorting networks, Batcher's odd-even and bitonic sort, sorting on hypercubic networks, mesh and mesh-like architectures.
(v) Graph algorithms: All-pairs shortest-path (APSP) problem, finding connected components of a graph, minimum spanning tree.
(vi) Computational Geometry: Inclusion problem, intersection problem, proximity problem, construction problem.

(b) Nil

(c) Four lectures per week (including tutorials).

(d) Theory 100%

(e) References:

1. K. Hwang and F. A. Briggs: Computer Architecture and Parallel Processing, McGraw Hill, New Delhi, 1984.
2. K. Hwang: Advanced Computer Architecture, McGraw Hill, New York, 1993.
3. M. J. Quinn: Design of Efficient Algorithms for Parallel Computers, McGraw Hill, New York, 1988.
4. S. G. Aid: Design and Analysis of Parallel Algorithms, Prentice Hall, Englewood Cliffs, 1989.
5. T. Leighton: Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes, Morgan Kauffmann Pub., San Mateo, 1992.
6. S. Lakshmivarahan and Analysis and S. K. Dhall: Design of Parallel Algorithms, McGraw Hill, New York, 1990.
7. J. Jaja: Introduction to Parallel Algorithms, Addison-Wesley, Reading, Mass., 1992.
8. S. G. Aid: Parallel Sorting Algorithms, Academic Press, Orlando, 1985.

B36. Software Engineering

(a) *Part 1 Theory:* Basic concepts of life cycle model, introduction to software specification, its need and importance, formal specification methods including finite state machines, petri nets, state charts, decision tables, program design languages, attributes of good SRS.

Software Design methods and strategies, desirable design attributes, Jackson structured programming, structured systems analysis and structured design, formal approaches to design, object-oriented design, unified modelling language, system design using UML.

Introduction to software verification, software inspection, black box and white box testing, incremental testing, formal proof of correctness, software metrics.

Software reliability, formal definition and its modelling— stochastic models and parameter estimation. Software engineering management, Introduction to capability maturity model, coding standard, software standard, quality assurance, software cost estimation (Delphi, COCOMO).

Part 2: Assignment: **Note** Each student should be given about 3-4 assignments on SRS, design, testing and allied problems. Different students should be asked to use different tools.

(b) Nil

(c) Four lectures and one tutorial per week

(d) Theory 60% and Assignment 40%

(e) References:

1. A. M. Davis: Software Requirements— Objects, Functions and States, Prentice Hall, Englewood Cliffs, 1993
2. G. Booch: Object-Oriented Analysis and Design, Benjamin I Cumming Publishing co., New York, 1994.
3. D. Budgen: Software Design, Addison-Wesley, Reading, Mass., 1994.
4. M. C. Paulk, C. V. Weber, B. Curtis and M. Beth Chrissis: The Capability Maturity Model for Improving the Software Process, Carnegie Mellon University, Addison-Wesley, Reading, Mass., 1999.
5. J. D. Musa, A. Iannino, K. Okumoto: Software Reliability -Measurement, Prediction and Application, McGraw Hill, New Delhi, 1987.
6. R. Fairly: Software Engineering Concepts, Tata McGraw Hill, New Delhi, 1999.

7. P. Jalote: An Integrated Approach to Software Engineering, 2nd ed. Narosa, New Delhi, 1997.
8. R. S. Pressman: Software Engineering: A Practitioner's Approach, 5th ed., McGraw Hill, College Div, Singapore, 2000.
9. P. Oman and S. L. Pfleeger: Applying Software Metrics, IEEE Computer Society Press, Los Alamos, California, 1996.
10. S. L. Pfleeger: Software Engineering -Theory and Practice, Prentice Hall, New York, 1998.
11. C. Larman: Applying UML and Patterns, Addison-Wesley, Reading, Mass., 1998.
12. Capability Maturity Model: The Guidelines for Improving the Software Process, CMU, Software Engg. Inst., 1995.

B37. Compiler Construction

(a) *Introduction:* Compiler, phases and passes, bootstrapping, finite state machines and regular expressions and their applications to lexical analysis, implementation to lexical analysers, lexical-analyser generator; LEX-compiler, formal grammars, and their application to syntax analysis, BNF notation, ambiguity, LL(k) and LR(k) grammar, bottom-up and top-down parsers, operator precedence, simple precedence, recursive descent and predictive parsers, LR(k) parsers, parse table generation, YACC.

Syntax directed translation: Quadruples, triples, 3-address code, code generation for standard constructs with top-down and bottom-up parsers, translation of procedure calls, record structuring.

Code optimization: Loop optimization, DAG analysis, loop identification by flow dominance, depth-first search, reducible flow graphs, legal code motion, induction variables, data flow analysis, u-d and d-u chains, copy propagation, elimination of global sub-expressions, constant folding, code hoisting, forward and backward data flow equations, inter procedural data flow analysis.

Code generation: Problems in code generation, code generator, register assignment and allocation problems, usage count, code generation from DAG, peephole optimization.

Symbol table: Data structure and management, runtime storage administration, error detection and recovery; Lexical, syntactic and semantic errors, case studies with real life compilers.

(b) Nil

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

(d) Theory 70% and Assignment 30%

(e) References:

1. A. V. Aho, R. Sethi and J. Ullman: Compilers: Principles, Techniques and Tools, Addison-Wesley, California, 1986.
2. A. Appel: Modem Compiler Implementation in Java, Cambridge Univ. Press, London, 1997.

B38. Distributed Computing Systems

(a) Introduction, Distributed Programming Model, Theoretical Foundations of Distributed Systems, Balanced Sliding Window Protocol, Routing Algorithms, Deadlock free packet switching, Wave and Traversal Algorithms, Minimal Spanning Tree Algorithms, Logical Clocks and Causal Ordering, Distributed Mutual Exclusion, Distributed Deadlock Detection Algorithms, Termination Detection Algorithms, Communication Protocols, Agreement Protocols, Commit Protocols, Leader Election Algorithms, Self-Stabilization Algorithms, Failure Recovery and Fault tolerance in distributed systems, Distributed File System (DPS), Distributed Shared Memory, Distributed Simulation, Authentication, Introduction to Kerberos, Case studies

(b) Design and Analysis of Algorithms, Data and File Structures, Operating Systems

(c) Three lectures and 1tutorial per week

(d) Theory 70% and Assignment 30%

(e) References:

1. G. Tel: Introduction to Distributed Algorithms, Cambridge University Press, 1994.
2. M. Singhal and N. G. Shivaratri: Advanced Concepts in Operating Systems, Tata McGraw-Hill Publishing Company Limited, 2005
3. H. Attiya and J. Welch: Distributed Computing: Fundamentals, Simulations, and Advanced Topics, John Wiley and Sons, Inc., 2004.
4. N. Lynch: Distributed Algorithms, Elsevier (imprint: Morgan Kaufmann), 1996.
5. S. Ghosh: Distributed Algorithms: An Algorithmic Approach, Chapman and Hall, 2006

B39. Mobile Computing

(a) *Introduction:* Challenges in mobile computing, coping with uncertainties, resource scarcity, bandwidth, energy etc. Cellular architecture, co-channel interference, frequency reuse, capacity increase by cell splitting. Evolution of mobile system: CDMA, FDMA, TDMA, GSM.

Mobility Management: Handoff, types of handoffs; location management, HLR-VLR scheme, hierarchical scheme, predictive location management schemes. Mobile IP, cellular IP.

Publishing and Accessing Data in Air: Pull and push based data delivery models, data dissemination by broadcast, broadcast disks, directory service in air, energy efficient indexing scheme for push based data delivery.

File System Support for Mobility: Distributed file sharing for mobility support, storage manager for mobility support.

Ad hoc and Sensor Networks: Routing Algorithms and Protocols, load-balancing, scheduling for reduced energy, coverage and connectivity problems.

Mobile Transaction and Commerce: Models for mobile transaction. Kangaroo and joey transactions, team transaction. Recovery model for mobile transactions. Electronic payment and protocols for mobile commerce.

(b) Operating systems, Distributed Systems, Computer Architecture and Computer Networks.

(c) Three lectures and 1tutorial

(d) Theory 70% and Assignment 30%

(e) References:

1. T. Imielinski and H. F. Korth: Mobile Computing, Kluwer Academic Publishers, Boston, 1996.
2. A. A. Helal, B. Haskell, J. L. Carter, R. Brice, D. Woelk and M. Rusinkiewicz: Any Time, Anywhere Computing: Mobile Computing Concepts and Technology, Kluwer International Series in Engineering and Computer Science, 1999.

3. C. Perkins, S. R. Alpert and B. Woolf: Mobile IP: Design Principles and Practices, Addison Wesley Longman, 1997.
4. Y. Lin and I. Chlamtac: Wireless and Mobile Network Architecture, Wiley India Pvt. Ltd., New Delhi, 2001.

B40. VLSI Testing and Fault Tolerance

(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 of verification frameworks in EDA.

(b) Pre-requisite: Computer Organization.

(c) Four lectures per week

(d) Theory 75% and 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.

B41. Software Design and Validation

(a) *Theory:* Introduction to Software Design
Modelling notations
Model Validation: Model simulation and model-based testing
Performance validation: Timing analysis and prediction; scheduling methods
Software validation: Trace analysis and Debugging methods; Static property checking of software. Validation of communication behaviour

Assignments: System Modelling assignment using Rhapsody; system Verification assignment using SPIN; performance analysis assignment using Chronos

(b) Discrete Mathematics, Data and File Structures

(c) Three lectures and 1 hands-on per week

(d) Theory 70% and Assignment 30%

(e) References:

1. A. Roychoudhury: Embedded Systems and Software Validation, Morgan Kaufmann Systems-on- Silicon Series, 2009
2. F. Nielson, H. R. Nielson, Chris Hankin: Principles of Program Analysis, 2nd ed., Springer, 2005

B42. Advanced Database Theory and Applications

(a) *Object oriented model:* Nested relations, modelling nested relations as object model, extension of SQL, object definition and query language (ODL, OQL), object relational database model, storage and access methods. Active databases, Advanced trigger structures, SQL extensions.

Security and Integrity: Discretionary and mandatory access control; Facilities in SQL, access control models for RDBMS and OODBMS.

Distributed Database: Basic Structure, fragmentation algorithms, trade-offs for replication, query processing, recovery and concurrency control; Multi-database systems; Design of Web Databases.

Data Mining and Warehousing: Association Rule algorithms, algorithms for sequential patterns; Clustering and classification in data mining; Basic structure of a data warehouse; Extension of ER Model, materialistic view creation; On line analytical processing and data cube. Deductive databases, recursive query construction, logical database design and datalog.

One or more of the following topics:

(i) Temporal database, (ii) Multimedia database, (iii) Text retrieval and mining, (iv) Web mining, and (v) Any topic of current interest.

(b) Nil

(c) Four lectures per week

(d) Theory 80% and Assignment/Seminar 20%

(e) References:

1. H. F. Korth and A. Silberschatz: Database System Concepts, McGraw Hill, New Delhi, 1997.
2. R. A. Elmasri and S. B. Navathe: Fundamentals of Database Systems, 3rd ed., Addison-Wesley, London, 1998.
3. S. Abiteboul, R. Hull and V. Vianu: Foundations of Databases, Addison-Wesley, 1995.
4. Selected papers from different LNCS (Lecture Notes on Computer Science, Springer, Berlin) volumes and different URLs on Internet.

B43. Information Security and Assurance

(a) *Overview of Security Parameters:* Confidentiality, integrity and availability- Security violation and threats-Security policy and procedure- Assumptions and Trust- Security Assurance, Implementation and Operational Issues- Security Life Cycle.

Access Control Models: Discretionary, mandatory, roll-based and task-based models, unified

models, access control algebra, temporal and spatio-temporal models.
Security Policies: Confidentiality policies, integrity policies, hybrid policies, non-interference and policy composition, international standards.
Systems Design: Design principles, representing identity, control of access and information flow, confinement problem. Assurance: Building systems with assurance, formal methods, evaluating systems.
Logic-based System: Malicious logic, vulnerability analysis, auditing, intrusion detection.
Applications: Network security, operating system security, user security, program security.
Special Topics: Data privacy, introduction to digital forensics, enterprise security specification.

(b) Cryptology

(c) Four lecture hours per week

(d) Theory-80%, Assignments-20%.

(e) References:

1. M. Bishop: Computer Security: Art and Science, Pearson Education, 2003.
2. M. Stamp: Information Security: Principles and Practice, John Wiley & Sons, 2005
3. Selected papers as suggested by the teacher.

B44. Advanced Operating System

(a) *Device Drivers.*
Message Passing: Interprocess communication, group communication, broadcasting algorithms.
Remote Procedure Call: 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.
Synchronization: Clock synchronization, event ordering, mutual exclusion, deadlock, election algorithms.
Resource Management: Scheduling algorithm, task assignment, load balancing, load sharing.
Process Management: Process migration, threads.
File Systems.
Protection and Security.
Fault Tolerance.
Real time OS: pSOS, VxWorks.
Naming in distributed systems, directory services, DNS.
Case studies of some distributed OS: Hydra, Mach, Amoeba, etc.

(b) Nil

(c) Four Lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. A. S. Tanenbaum: Distributed Operating Systems, Prentice Hall of India, New Delhi, 1996.
2. G. F. Colouris, J. Dollimore and T. Kindberg: Distributed Systems: Concepts and Design, 2nd ed., Addison-Wesley, Reading, Mass., 1994.
3. S. J. Mullender (Ed.): Distributed Systems: An Advanced Course, 2nd ed., Addison-Wesley, Reading, Mass., 1993.
4. P. K. Sinha: Distributed Operating Systems, IEEE Press, Los Alamos, California, 1997.

B45. Robotics

(a) *Introduction:* Robot anatomy, manipulator mechanics, robot arm kinematics and dynamics, trajectory planning.
Control: Control of robot manipulators, feedback and adaptive control, Kalman filter.
Sensing systems: Range sensing, proximity sensing, tactile, force and torque sensing, grasping strategies.
Robot programming languages.
Navigation: Mobile robots, path planning, navigation.
Vision: Robot vision, scene understanding, recognition, interpretation.
Knowledge engineering and task planning.
Some applications and related advanced topics.

(b) Pre-requisite: As suggested by teacher.

(c) Four lectures per week

(d) Theory 80% and Assignment 20%

(e) References:

1. K. S. Fu, R. C. Gonzalez and C. S. G. Lee: Robotics Control: Sensing, Vision and Intelligence, McGraw Hill, Singapore, 1987.
2. J. J. Craig: Introduction to Robotics: Mechanisms and Control, 2nd ed., Addison-Wesley, Reading, Mass., 1989.
3. M. Sponge and M. Vidyasagar: Robot Dynamics and Control, John Wiley, New York, 1989.
4. R. M. Haralick and L. G. Shapiro: Computer and Robot Vision, Addison-Wesley, Reading, Mass., 1992.
5. B. K. P. Horn, Robot Vision, MIT Press, Cambridge, 1986.
6. Related Internet based literatures/tutorials.

B46. Real-Time Systems

(a) Introduction to real-time system design, real-time operating system -OSE Delta.
Reliable Programming, Fault tolerance, Exception handling, concurrency models, atomic actions, real-time facilities, real-time scheduling, resource control.
Real-time database systems.
High-level and low-level programming issues, real-time languages: Ada.
Turing efficiency, Formal specification methods: Petri nets, Q-models, verification, model checking, theorem proving.
Safety critical real-time control systems, supervisory control theory. Communication under timing

constraints.

(b) Nil

(c) Four lectures per week

(d) Theory 75% and Practical 25%

(e) References:

1. A. Burns and A. Wellings: Real-Time Systems and Programming Languages, 2nd ed., Addison-Wesley, Reading, Mass., 1996.
2. R. A. Buhr and D. L. Bailey: Introduction to Real-Time Systems: From Design to Networking With C/C++, Prentice Hall, Englewood Cliffs, 1998.
3. J. Ellsberger, D. Hogrefe and A. Sarma: SDL: Formal Object-Oriented Language for Communicating Systems. Prentice Hall, Englewood Cliffs, 1997.
4. S. F. Andler, J. Hansson, J. Eriksson, J. Mellin, M. Bemdtsson, and B. D. Eftring: Towards a Distributed Active and Real-Time Database System, SIGMOD Record, Special Section on Real-Time Databases, March 1996.
5. Chakravarthy et al.: HiPAC: A Research Project In Active Time-Constrained Database Management - Final Technical Report, Xerox Advanced Information Technology, Cambridge Center, Technical Report No. XAIT-89-02, Reference Number 187, July, 1989.
6. Eich: A Classification and Comparison of Main Memory Database Recovery Techniques, In Proc. Int'l Conf. Data Engineering, pp. 332-339, 1987.
7. Levy and Silberschatz: Incremental Recovery in Main Memory Database Systems, Technical Report No. TR-92-01, Department of Computer Sciences, The University of Texas at Austin, January, 1992.
8. Singhal and Mukesh: Issues and Approaches to Design of Real-Time Database Systems, Special Issue on Real Time Data Base Systems, SIGMOD Record 17(1), March, 1988.
9. Levi and Agrawala: Real-Time System Design. McGraw Hill New York, 1990.
10. H. W. Lawson: Parallel Processing in Industrial Real-Time Applications, Prentice Hall, Englewood Cliffs, 1992.

B47. Advanced Web Technology/Advanced Internet Programming

(a) *Introduction:* Overview and evolution of Internet programming and application tools, searching and browsing tools.
Markup Languages and its application on the web: HTML, XML and related concepts.
Java programming: Features of Java Language -brief history of Java, concept of Java VM, basic structure of Java programming; GUI programming -basics of Java Abstract Windowing Toolkit, event handling and swing programming, applet programming; Java Beans and Java IDE; I/O in Java;
Network Programming: Client Server programming, Remote Method Invocation; Java database connectivity; Multi-Thread programming in Java.
Communication protocol: TCP/IP, IP addressing and domain registration and related concepts.
Application level technology: HTTP, Web Server, Browsing, Firewall.
Search Mechanisms: Search Engine, Crawler Technology, Filtering Technology Content based Searching, Agent Technology, Internet Robot.
Advance Internet applications: Data and Web mining; e-commerce; Distributed Objects—

component object model, common object request broker architecture, Web security.

(b) Pre-requisite: Nil

(c) Three classes and one Tutorial per week

(d) Theory 70% and Assignment 30%

(e) References:

1. C. S. Horstmann and G. Cornell: Core Java 2, Vol. 1: Fundamentals, Prentice Hall, Englewood Cliffs, 1998.
2. W. R. Stevens: TCP/IP Illustrated, Vol. 1: The Protocols, Addison-Wesley, Reading, Mass., 1994.

B48. Nanotechnology and Biochips

(a) *Description:* Nanotechnology is likely to dominate the present century with its interdisciplinary ramifications into almost all areas of human civilization. This new science attempts to attain control of matter at the molecular or atomic level, i.e., at the single nanometer scale. In the computing domain, it is going to bring a complete paradigm shift at the device level, processor level, architecture level and at the computational modelling level. Other computer-related application areas are nano-biosensors and biochips, the design of which requires knowledge of nanotechnology, microfiuidics, and design automation techniques, among others. This course has been designed to train computer science students in this fascinating area, where their knowledge of algorithms and computer design is needed to solve various challenging interdisciplinary nano-scale engineering problems.

Topics include but not limited to the following:
Nanocomputing Systems: Design and analysis of nano-scale computing systems and devices, e.g., carbon nanotube transistors, molecular electronics, quantum dots, spintronics, etc.), computational paradigms (von Neumann, quantum cellular automata, quantum computing, reversible logic, DNA computing, etc.), microarchitecture and instruction set architecture for nano-scale systems, defect and fault tolerance, fabrication techniques (e. g., nanolithography, self-assemblies), modelling and simulation methods. Nanoscale design automation techniques. Nanobio Systems: Biomaterial based metallic nanowires, networks and circuitry. DNA/protein based nanocircuits, biosensors, MEMS, microfiuidics, microarrays, and biochips. Design algorithms for digital microfluidic lab-on-a-chip platforms. Drug delivery, therapeutic action of nanoparticles and nanodevices. Nano-scale imaging and analysis algorithms.

(b) Computer organization, Data and File Structures, Design and Analysis of Algorithms, VLSI Design

(c) Three hours of lecture and two hours of laboratory (for familiarity with design automation techniques) per week

(d) Theory 70% and Assignment/Term Projects 30%

(e) References:

1. K. Chakrabarty and F. Su: Digital Microfluidic Biochips, CRC Press, 2007
2. T. Xu and K. Chakrabarty: Digital Microfluidic Biochips: Design Automation and Optimization,

CRC Press, 2010
3. M. Wilson, et al.: Nanotechnology, Overseas Press, Reprinted 2008.
4. O. Shoseyyov and I. Levy (Ed.): NanoBioTechnology, Humana Press, NJ, 2008.
5. M. Gross: Travels to the Nanoworld: Miniature Machinery in Nature and Technology, Basic Books, 2001.
6. H. S. Nalwa (Ed.): Encyclopaedia of Nanoscience and Nanotechnology, vol. 1-25, 2010.

B49. Quantum Information Processing and Quantum Computation

(a) *Introduction to Hilbert space:* Linear space, Scalar product, Hilbert space, Self adjoint operator, Projection operator, Unitary operator.
Basic introduction to Quantum mechanics: (i) Postulates of quantum mechanics, Uncertainty principle, Complementary principle, Unitary Dynamics, Detail study of two-level system. (ii) Multipartite quantum system, Quantum entanglement, Schmidt decomposition, Non-unique decomposition of mixed state, Hugston-Jozsa-Wooters theorem, No-Cloning Theorem. (iii) General quantum operations, Kraus representation theorem, various Quantum gates.
Basic quantum information processing: (i) Quantum teleportation, (ii) Quantum dense coding, (iii) Remote state preparation, (iv) Quantum key distribution (Bennett-Brassard- 1984 Protocol, Ekerts entanglement protocol)
Quantum computing: Basic physics of Quantum parallelism, Some basic quantum algorithm; Deutschs algorithm, Deutsch-Jozsa algorithm, Simons algorithm, Grovers search algorithm, Quantum Fourier Transform and Shors factoring algorithm.
Introduction to elementary Quantum error correcting codes

(b) Elements of Algebraic Structures

(c) Two lectures each of two hours duration

(d) Theory 80% and Assignment 20%

(e) References:

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

B50. Functional brain signal processing: EEG and fMRI

(a) *Electroencephalogram (EEG):* Cortical sources of the scalp EEG signals; Linear propagation model (forward problem); Acquisition of EEG; Artefacts of EEG; Preprocessing – filtering, principal component analysis, independent component analysis; Different frequency bands of interest in EEG; Event related potential (ERP); Sleep EEG; Spectral decomposition and feature identification by time frequency analysis – Fourier, wavelet, Hilbert transformation and spectral estimation; Various coherence and synchronization measures. Pattern classification by linear discriminants, support vector machine (SVM) and artificial neural networks (ANN). MATLAB coding assignments; Familiarization with EEGLAB (a MATLAB based open source software for

EEG signal processing).

Functional Magnetic Resonance Imaging (fMRI): Hemodynamic activity in our brain and the basic physics of fMRI; T1, T2 and T2* weighted images; 3D reconstruction of the MR images – K space and Talairach coordinate system, and functional sequencing over time; Fast fMRI acquisition – Echo-Planar Imaging (EPI); Preprocessing – Artefacts removal; Spatial normalization; Spatial smoothing; Statistical analysis – General linear model (GLM) for single subject analysis; GLM for group analysis; Inferencing blood oxygen level dependent (BOLD) patterns of activations; Multi-voxel pattern analysis (MVPA). Familiarization with SPM (MATLAB based open source software for fMRI processing) and FSL (Unix based open source software for fMRI processing).

Simultaneous EEG-fMRI acquisition – an overview.

(b) Prerequisites: Signal or image processing, good programming concept in any one of C, C++, Python and Unix shell script. Knowledge of MATLAB will be a plus but not essential. Familiarity with elementary linear algebra, and basic statistics and probability will be very helpful. However attempts will be made to make the course as self-contained as possible.

Co-requisites: Participants in the course are encouraged to take Image Processing or Signal Processing or Pattern Recognition or Data Mining optional course prescribed in the M. Tech. (CS) curriculum.

(c) The course will be covered in 3 hour lecture for every week for 16 weeks. Hours spent on computer assignments on real data sets will be extra.

(d) 50% weightage will be on written examination and the remaining 50% will be on laboratory assignments.

(e) References

1. A brief survey of quantitative EEG analysis, Kaushik Majumdar (under preparation under a contract with the Taylor & Francis); relevant portions will be distributed during the course.
2. Electroencephalogram processing using neural networks, C. Robert, J. -F. Gaudy and A. M. Limoge, Clinical Neurophysiology, vol. 113, pp. 694–701, 2002.
3. BCI competition 2003 – data set IIb: support vector machines for the P300 speller paradigm, IEEE Transactions on Biomedical Engineering, vol. 51(6), pp. 1073– 1076, 2004.
4. Modern spectral estimation: theory and application, S. M. Kay, Pearson, 1988.
5. Handbook of functional MRI data analysis, R. A. Poldrack, J. A. Mumford and T. E. Nichols, Cambridge University Press, New York, 2009.
6. Beyond mind-reading: multi-voxel pattern analysis of fMRI data, K. A. Norman, S. M. Polyn, G. J. Detre and J. V. Haxby, Trends in Cognitive Science, vol. 10(9), pp. 424–430, 2006.