MSc in Computer Science

Department of Computer Science

RKMVERI, Belur Campus

Program Outcomes

Program Specific Outcomes

Course Outcomes

Program outcomes

- Inculcate critical thinking to carry out scientific investigation objectively without being biased with preconceived notions.
- Equip the student with skills to analyze problems, formulate an hypothesis, evaluate and validate results, and draw reasonable conclusions thereof.
- Prepare students for pursuing research or careers in industry in mathematical sciences and allied fields
- Imbibe effective scientific and/or technical communication in both oral and writing.
- Continue to acquire relevant knowledge and skills appropriate to professional activities and demonstrate highest standards of ethical issues in mathematical sciences.
- Create awareness to become an enlightened citizen with commitment to deliver ones responsibilities within the scope of bestowed rights and privileges.

Program Specific Outcomes

- Understanding the theoretical underpinnings in computing and computing systems.
- Knowledge of the synergy between s/w and h/w through the study of computer architecture, compilers, and systems programming.
- Knowledge about storage, organization, and manipulation of structured data.
- Knowledge and application of various algorithms, algorithmic methods, and data structures in solving computational problems drawn from various fields such computer graphics, computational geometry, distributed systems, data mining, mobile computing.
- Understanding the linkages that optimization has with machine learning, deep learning, data mining, computer vision etc.
- Knowledge of complexity classes and its appearance in algorithm design.
- Develop workable solutions for problems drawn either from social context or from research corpus.
- Develop s/w applications for handheld devices in Android.
- Use software development tools, software systems in modern computing platforms.
- Communicate computer science concepts, designs, and solutions effectively and professionally.

CS241
Design and Analysis of Algorithms

Joydeep Mukherjee joydeep.m1981@gmail.com

Course Description: This course deals with topics in design and analysis of algorithms. In particular, the course will cover different techniques of algorithm design illustrating them with several examples and also highlight some of the lower bounding techniques in algorithm design such as NP-Completeness.

Prerequisite(s): (1) High School Mathematics.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url:
Credit Hours: 4

Text(s):

Introduction to Automata Theory, Languages, and Computation, third edition
John E. Hopcroft, Rajeev Motwani & Jeffery D. Ullman
ISBN-13: 978-8131720479

Introduction to Algorithms, third edition
Thomas H. Cormen, Charles E. Leiserson, Ronald Rivest, Clifford Stein
ISBN: 9788120340077

Algorithm Design, first edition
Eva Tardos, Jon Kleinberg
ISBN: 9789332518643

Course Objectives:

Knowledge acquired: (1) Asymptotic analysis of running time of algorithms,
(2) different techniques of algorithm design, and,
(3) polynomial time reducibility.

Skills gained: The students will be able to
(1) compare different algorithms in terms of their running time ,
(2) design algorithms for some practical problems, and,
(3) do polynomial time reductions based on knowledge gained in the class.

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Different order notations like $O, \Theta, \Omega, o, \theta, \omega$ and compare two different functions using order notation. • Methods to calculate and state running time of algorithms using order notations.
Week 2	• Introduction of the Divide and Conquer paradigm of algorithm design. • Devising algorithms using divide and conquer for merge sort, counting inversions, finding closest pair of points in a plane, fast integer multiplication etc. • Home assignment 1
Week 3	• Fast Fourier Transform and its application. • Quiz 1
Week 4	• Introducing the concept of Dynamic Programming and use of memoization. • Devising algorithms using dynamic programming for the problems like longest increasing subsequence, edit distance, knapsack, matrix chain multiplication, independent sets in trees etc.
Week 5	• Greedy methods of algorithm design. • Studying few techniques for proving the correctness of greedy algorithm.
Week 6	• Devising greedy algorithm for various problems like minimum spanning tress, Huffman codes, Horn clauses etc. • Home assignment 2 • Quiz 2
Week 7	• Breadth First Search (BFS)in graphs. • Depth First Search (DFS) in graphs.
Week 8	• Topological sorting of a directed acyclic graph. • Finding all strongly connected components of a directed graph. • Finding articulation points,bridges and biconnected component of a graph. • Finding Eulerian tour in a Eulerian graph. • Home assignment 3
Week 9	• Union Find data structure. • Kruskal and Prim's algorithm for minimum spanning trees. • Home assignment 4
Week 10	• Algorithms for single source shortest paths in a directed graph like Bellman-Ford algorithm, Dijkstra's algorithm. • Home assignment 5 • Quiz 4
Week 11	• Few applications of Single Source Shortest Paths algorithms • Home assignment 6
Week 12	• Algorithms for all pair shortest paths. • Matrix multiplication based procedure. • Floyd-Warshall algorithm. • Johnson's algorithm for sparse graphs. • Home assignment 7 • Quiz 5
Week 13	• String Matching algorithms • Home assignment 8
Week 14	• Introduction to the concept of P, NP, NP-Completeness, • Circuit satisfiability, Boolean satisfiability
Week 15	• NP-Completeness reduction for few problems. • Review for Final Exam

CS123
Concepts of Programming Languages

Time: TBA

Place: MB212

Instructor: Dhyanagamyananda

`dhyangamyananda@gmail.ac.in, swathyprabhu@gmail.com`
url: `http://cs.rkmvu.ac.in/~swat/`
Office: MB205, Medhabhavan, RKMVERI, Belur
Office Hours: 10 pm—12 noon, 3 pm—5 pm
`(+91) 033-2654 9999`

Course Description: CS123 deals with analysing the relevance, benefit, and limitations of various features that have been implemented in important and widely used programming languages. It introduces the student to various programming paradigms. With C programming language as a case study, the student is introduced to the different stages in compilation, namely Lexical analysis, Semantic Analysis, and intermediate code generation.

Prerequisite(s): (1) Good working knowledge of C, and C++/Java
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Moodle url: `http://moodle.rkmvu.ac.in/course/view.php?id=58`
Credit Hours: 4

Text(s):

Principles of programming languages, third edition
Kenneth Louden

Understanding Programming Languages -ebook
M. Ben-Ari

The anatomy of programming languages
Alice. E. Fisher, & Frances. S. Grodzinsky

Compilers: Principles, Techniques, & Tools
Aho, Lam, Seith, & Ullman

Course Objectives:

Knowledge acquired: (1) Different models of computation,
(2) their associated complexity classes, and,
(3) reducibility.

Skills gained: The students will be able to

1. classify different langauges based on the programming paradigms, like imperative, functional, logic, procedural, object oriented, declarative.
2. critically analyse the programming language design criterion like readability, writeability, orthogonality, generality etc.
3. differentiate between the syntactic and semantic notions of programming languages.
4. discern the relative merit and demerit in the choice of programming language to solve a given computing problem.
5. explain equivalence checking, conversion, polymorphism for PL Data types.
6. conceptualize the PL Procedure environments, activations and allocations.
7. understand how memory is dynamically managed, and exception handling is implemented.
8. understand the differences among operational semantics, denotational semantics, and axiomatic semantics.

Grade Distribution:

Assignments	20%
Quizzes	10%
Midterm Exam	20%
Final Exam	40%

Grading Policy: There will be relative grading such that the cutoff for A grade will not be less than 75% and cutoff for F grade will not be more than 34.9%. Grade distribution will follow normal bell curve (usually, A: $\geq \mu+3\sigma/2$, B: $\mu+\sigma/2 \ldots \mu+3\sigma/2$ C: $\mu-\sigma/2 \ldots \mu+\sigma/2$, D: $\mu - 3\sigma/2 \ldots \mu - \sigma/2$, and F: $< \mu - 3\sigma/2$)

Approximate grade assignments:

>= 90.0	A+
75.0 – 89.9	A
60.0 – 74.9	B
50.0 – 59.9	C
about 35.0 – 49.9	D
<= 34.9	F

Course Policies:

- **General course policies, Grades, Labs and assignments, Attendance and Absences** These clauses are common to all courses. And it can be found in the program schedule.

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Definition of programming languages, their elements, environments, and design criteria, • Reading assignment: Chapter 1,2, KL
Week 2	• Lexical structure of PL, scope of lexical analysis, tools for implementing lexical analysis. • Reading assignment: Chapter 6, KL, Ref: Ch 5 ASUL • Programming assignment 1: Building a lexical analyser for C–. • Quiz 1
Week 3	• Context free grammars, Parse trees, Abstract Syntax trees, Ambiguity, Associativity and precedence of operators. Understanding the C grammar. • Reading assignment: Chapter 6: KL, Chapter 4.2,3 ASUL, C-Grammar from KR
Week 4	• Overview of various Parsing Techinques, Top-Down parsing • Reading assignment: Chapter 2.4, 4.4 ASUL, • Programming assignment: Building a top down parser for expression grammar.
Week 5	• Bottom-up parsing: Reductions, Handle pruning, Shift-reduce parsing, handling conflicts. • Reading assignment: Chapter 4.5 ASUL • Quiz 2
Week 6	• LR Parsing: Items, LR(0) Automaton, SLR parsing tables, Viable prefixes • Reading assignment: Chapter 4.6 ASUL
Week 7	• LR(1) items, construction of LR(1) automaton, LR(1) parsing tables, LALR parsing tables. • Reading assignment: Chapter 4.7, ASUL • Quiz 3
Week 8	• Parser generator tool: Yacc/Bison • Reading Assignment: Internet resources, Ref: Bison, Shroff Publishers. • Proramming assignment: Building AST for C– using yacc/bison • Review for Midterm Exam

Week	Content
Week 9	• Syntax directed translation: Inherited and Synthesized attributes, S-attributed and L-attributed definitions • Reading assignment: Chapter 5.1,2 ASUL • Home assignment 4
Week 10	• SDT-contd: structure of a Type, postfix translation schemes, Parser-stack implementation of postfix SDT's. • Reading assignment: Chapter 5.3,4 ASUL • Home assignment 5 • Quiz 4
Week 11	• Intermediate code generation: Tranlation of expressions, Type checking • Reading assignment: Chapter 6.4,5, ASUL • Home assignment 6
Week 12	• ICG-contd: Boolean expressions, short-circuit code, flow-of-control statements, avoiding redudant gotos, boolean values, and jumping code. • Reading assignment: Chapter 6.6, ASUL • Home assignment 7 • Quiz 5
Week 13	• ICG-contd: Backpatching, switch statements, procedures. • Reading assignment: Chapter 6.7,8, ASUL • Home assignment 8
Week 14	• Types revisited: Type Constructors, Type Equivalence, Type Checking, Type Conversion, Hindley-milner Polymorphic Type Checking. • Reading assignment: Chapter 8, KL
Week 15	• Dynamic memory management • Reading assignment: Chapter 7, AB • Review for Final Exam
Week 16,17	• Logic programming: Horns clauses, resolution and unification, Prolog: a case study. • Reading assignment: Chapter 7, AB • Programming assignment:
Week 18,19	• Functional programming: A study of Haskell • Reading assignment: Programming in Haskell. • Programming assignment:

CS211
Graph algorithms and Combinatorial optimization

Instructor: Dhyanagamyananda swathyprabhu@gmail.com

Course Description: CS211 is the first course to deal with the topic of this course. This course is a mixed bag of graph algorithms. Some of these algorithms are relevant in the context of optimization. The field of graph algorithms is vast and the kind of problem studied in CS211 are those that are in general difficult to solve but has easy solutions for a sub-class of them.

Prerequisite(s): Design and Analysis of Algorithms, Data and File Structures.
Credit Hours: 4

Text(s):

Algorithm Design, PHI
Kleinberg & Targos

Introduction to Graph Theory
Douglas West

Lecture Notes from University of Waterloo

Draft on Discharging techinque by Douglas West

Course Objectives:

Knowledge acquired: .

(1) Flow networks.
(2) Planar graph theory
(3) Algorithm design and analysis

Grade Distribution:

Assignments 20%, Quizzes 10%, Midterm Exam 20%, Final Exam 40%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Network Flow: Definition, Basic Idea, Algorithm, Maxflow mincut theorem, Ford Fulkerson Algorithm Analysis, LP formulation of maxflow and proof. • Reading assignment: Chapter 3, KT
Week 2	• Layered Network: Definition, Theorem, Computation of blocking flow (Edmonds, Dinics, MPM) • Reading assignment: XBitmap from Wiki. • Programming assignment 1: • Quiz 1
Week 3	• Student presentation of Tarzan's algorithm
Week 4	• Bipartite matching: Definition, Application, Using Ford Fulkerson Algorithm bipartite matching is obtained in O(—V——E—) time Edge connecting problem. The augmenting path algorithm for bipartite matching. • Reading Assignment:
Week 5	• Matching for Non-Bipartite Graph: Theorem and proof (Edmonds blossom shrinking) • Reading Assignment: • Quiz 2
Week 6, 7	• Max-Cut: NP-Hard problem and its proof, 2-Approximation algorithm, Randomized algorithm for max-cut, De-randomization LP based approximation algo for maxcut • Reading assignment:
Week 8,9	• Interval Graph: Intersection graph, Perfect elimination order (PEO), Chordal graph (Triangulated Graph), Simplicial vertex, Algorithm MIS, vertex cover, coloring, clique cover for interval graph, Finding a PEO Comparability graph • Reading assignment: Waterloo Lecture Notes • Home assignment: • Quiz 3

Week	Content
Week 10,11,12	• Trees and Friends, Trees, Treewidth, Tree decomposition, Closure properties, Partial k-trees, Partial k-trees to tree decomposition, Tree decomposition to partial k-trees, Dynamic programming MIS algo for partial k-tree • Home assignment 4 • Quiz at the end of three weeks.
Week 13,14	• Perfect Graph, Definition and properties, Perfect graph theorem, Triangulated graph is a perfect graph • Home assignment 7 • Quiz 5
Week 15	• Discharging method • Reading assignment: DW on discharging • Home assignment 8

CS312
Approximation and Online Algorithms

Instructor
Prof. Subir Kumar Ghosh

Prerequisite(s): CS241: DAA

Credit Hours: 4

Text(s):

1. M. R. Garey and D. S. Johnson, Computers and Intractibility: A guide to the theory of NP-completeness, W. H. Freeman, 1979.
2. R. Motwani, Lecture Notes on Approximation Algorithms, Volume 1, No. STAN-CS-92-1435, Stanford University, 1992.
3. D. P. Williamson and D. B. Shmoys, The Design of Approximation Algorithms, Cambridge University Press, 2011.
4. Vijay Vazirani, Approximation algorithms, Springer-Verlag, 2001.
5. S. Albers, Competitive Online Algorithms, Lecture notes, Max Plank Institute, Saarbrucken, 1996.
6. S. K. Ghosh and R. Klein, Online algorithms for searching and exploration in the plane, Computer Science Review, vol. 4, pp. 189-201, 2010.

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Approximation Algorithm:
Performance Measure, Greedy Algorithm, Unweighted Vertex Cover Problem
Minumum-Degree Spanning Tree, Minimum Weight Spanning Tree, The Traveling-Salesman Problem,The k-Center Problem, Multiway Cut and K-Cut Problems, Scheduling Jobs with Deadlines on a Single Machine, Scheduling Jobs on Identical Parallel Machines, The Set Cover Problem, An Application of Set Cover to Art Gallery problems, Shortest Superstring Problem
Rounding Data and Dynamic Programming, The Knapsack Problem, The Bin-Packing Problem,
The Primal-Dual Method, Weighted Vertex Cover Problem

Online Algorithms:
Competitive Analysis, The Paging Problem, Amortized Analysis, List Update Problem, Scheduling Jobs on Identical Parallel Machines, Graph Colouring, Machine Learning, K-Server Problem, Target Searching in an Unbounded Region and Target Searching in Streets

DA103
Linear Algebra

Course Description: CS301 deals with topics in linear algebra. In particular, the course will cover linear equations, vector spaces, linear transformations, eigenvalues and eigenvectors, bilinear forms, introduction to linear programming and related topics.

Prerequisite(s): (1) Highschool mathematics.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Linear Algebra,
K. M. Hoffmann, R. Kunze
Prentice Hall.

Algebra,
M. Artin
Prentice Hall.

Introduction to Linear Algebra,
G. Strang
Wellesley-Cambridge Press.

Linear Programming,
L. I. Gass
Tata McGraw Hills.

Linear Programming,
G. Hadley
Narosa Publishing House.

Course Objectives:

Knowledge acquired: (1) systems of linear equations, their associated matrices and their properties,
(2) characteristic polynomial, eigenvalues and eigenvectors,
(3) bilinear forms, and,
(4) linear programming.
Skills gained: The students will be able to
(1) analyze system of linear equations,
(2) solving linear recurrences, and,
(3) formulating linear programming problems and finding their feasible and optimal solutions.

Grade Distribution:

Assignments	10%
Quizzes	10%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:
The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	**Content**
Week 1	• Systems of linear equations, Matrices and elementary row operations, Row reduced Echelon matrices,
Week 2	• Matrix multiplication, Invertible matrices, Transpose of a matrix,
Week 3	• Systems of homogeneous equations, Equivalence of row rank and column rank of a matrix, Determinant and volume of the fundamental parallelepiped,
Week 4	• Permutation matrices, Cramers rule, • Home assignment 1
Week 5	• Vector spaces and subspaces, Bases and dimensions, Coordinates and change of bases, Direct sums, • Home assignment 2
Week 6	• The Rank-Nullity theorem, Matrix of a linear transformation, Linear operators and isomorphism of vector spaces, Determinant of a linear operator,
Week 7	• Linear functionals, Annihilators, The double dual, • Home assignment 3
Week 8	• Eigenvalues and eigenvectors of matrices, The characteristic polynomial, Algebraic and geometric multiplicities of eigenvalues,
Week 9	• Diagonalizability, Cayley-Hamilton theorem, Solving linear recurrences, • Home assignment 4
Week 10	• Matrix of a bilinear form, Symmetric and positive definite bilinear forms, Normed spaces,
Week 11	• Cauchy-Schwarz inequality and triangle inequality, Angle between two vectors, Orthogonal complement, Projection,
Week 12	• Gram-Schmidt orthogonalization, Hermitian operators, The Spectral theorem, • Home assignment 5
Week 13	• Bounded and unbounded sets, Convex functions, Convex cone, Interior points and boundary points, Extreme points or vertices,
Week 14	• Convex hulls and convex polyhedra, Supporting and separating hyperplanes, Formulating linear programming problems,
Week 15	• Feasible solutions and optimal solutions, Graphical method, The basic principle of Simplex method, Big-M method, • Home assignment 6

CS250 Database Management Systems

Instructor:

Course Description: CS250 deals with a detailed study of principles of RDBMS.

Prerequisite(s): The student must know about a typical file system, data types like integer, float, and string, basic computer arithmetic, venn diagram regpresentation of union, intersection, and complement of sets.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url:
Credit Hours: 4

Text(s):

- H. F. Korth and A. Silberschatz: Database System Concepts, McGraw Hill, New Delhi, 1997.
- R. A. Elmasri and S. B. Navathe: Fundamentals of Database Systems, 3rd ed., Addison-Wesley, 1998.
- R. Ramakrishnan: Database Management Systems, 2nd ed., McGraw Hill, New York, 1999.
- C. J. Date, A. Kannan and S. Swamynathan, An Introduction to Database Systems, Pearson Education, Eighth Edition, 2009
- J D Ullman : Principles of Database Systems, Computer Science Press; 2nd edition (December 1982)

Course Objectives:

Knowledge acquired: At the finish of this course, students will be quite empowered and will know
(1) basic concepts of the database approach, the underlying models and organizational issues
(2) the relational database model takes a logical view of data
(3) data modelling
(4) the theoretical underpinnings of the relational database, including concepts like functional dependence, entity integrity, and relational integrity.
(5) how a flawed data model can impact relational database implementation and manipulation
(6) relational database operators, the data dictionary, and the system catalog
(7) the various relational algebra operations that provide the basis for relational database manipulation
(8) concurrency control and locking protocols.

Skills gained: The students will be able to
(1) interpret the modeling symbols for the most popular ER modeling tools.
(2) model the RDBMS schema with the help of ER models given a problem statement in English.
(3) construct queries in SQL to manipulate live RDBMS
(4) analyze database requirements and determine the entities involved in the system and their relationship to one another sophisticated database applications

Competence Developed: The student will be able to
(1) tackle the design, development, and implementation of databases in an organization.
(2) assume any role in the database design and implementation process
(3) identify computational bottlenecks in the performance of an algorithm

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

1. Introduction and Conceptual Modeling Database system concepts, three-schema Architecture, data independence, database administrator, database user, Client/Server Architecture, E-R diagram, mapping constraints, Keys, Generalization, Aggregation, Reducing E-R diagram to tables.

2. Relational Model: Concepts, constraints and Languages . Structure of Relational database, Entity Integrity, Referential Integrity, Foreign Keys, Query languages, Relational algebra and relational calculus, SQL, views.

3. Database Design Theory and Methodology Functional dependencies, Closure of a set of functional dependencies, Canonical cover, closure of attribute sets, Lossless decomposition, Dependency preservation, 1 NF, 2 NF, 3 NF, BCNF, Multivalued dependencies and 4 NF, Join dependencies and 5 NF.

4. Data Storage, Indexing and Query Processing File organization, Sequential file, B+ tree index files, B-tree index file, Static hash Functions, Dynamic hash functions, Query processing and Query optimization.

5. Transaction Processing Concepts Transaction, Properties of transaction, database recovery, shadow paging, recoverable schedule, serializable schedule; Concurrency control: Lock-Based protocol, Timestamp-Based protocol, Multiple granularity, Multiversion schemes; Deadlock Handling.

6. Database Security Discretionary access control, Mandatory access control and multi-level security, statistical database security, Introduction to flow control, Encription and public key infrastructures, privacy issues and preservation.

DA220 Machine Learning

Instructor: **Tanmay Basu**

Course Description:
DA220 deals with topics in supervised and unsupervised learning methodologies. In particular, the course will cover different advanced models of data classification and clustering techniques, their merits and limitations, different use cases and applications of these methods. Moreover, different advanced unsupervised and supervised feature engineering schemes to improve the performance of the learning techniques will be discussed.

Prerequisite(s): (1) Linear Algebra and (2) Probability and Stochastic processes
Credit Hours: 4

Text(s):

Introduction to Machine Learning E. Alpaydin ISBN: 978-0262-32573-8
The Elements of Statistical Learning J. H. Friedman, R. Tibshirani, and T. Hastie ISBN: 978-0387-84884-6
Pattern Recognition S. Theodoridis and K. Koutroumbas ISBN: 0-12-685875-6
Pattern Classification R. O. Duda, P. E. Hart and D. G. Stork ISBN: 978-0-471-05669-0
Introduction to Information Retrieval C. D. Manning, P. Raghavan and H. Schutze ISBN: 978-0-521-86571-5

Course Objectives:

Knowledge Acquired:

1) The background and working principles of various supervised learning techniques viz., linear regression, logistic regression, bayes and naive bayes classifiers, support vector machine etc. and their applications.
2) The importance of cross validation to optimize the parameters of a classifier.
3) The idea of different kinds of clustering techniques e.g., k-means, k-medoid, single-linkage, DBSCAN algorithms and their merits and demerits.
4) The significance of feature engineering to improve the performance of the learning techniques and overview of various supervised and unsupervised feature engineering techniques.
5) The essence of different methods e.g., precision, recall etc. to evaluate the performance of the machine learning techniques.

Skills Gained: The students will be able to

1) pre-process and analyze the characteristics of different types of standard data,
2) work on scikit-learn, a standard machine learning library,
3) evaluate the performance of different machine learning techniques for a particular application and validate the significance of the results obtained.

Competence Developed:

1) Build skills to implement different classification and clustering techniques as per requirement to extract valuable information from any type of data set.
2) Can train a classifier on an unknown data set to optimize its performance
3) Develop novel solutions to identify significant features in data e.g., identify the feedback of potential buyers over online markets to increase the popularity of different products.

Evaluation:
Assignments 50% Midterm Exam 25% Endterm Exam 25%

Course Outline (tentative) and Syllabus:
The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Week	Contents
Week 1	• Overview of machine learning: idea of supervised and unsupervised learning, regression vs classification, concept of training and test set, classification vs clustering and significance of feature engineering • Linear regression: least square and least mean square methods
Week 2	• Bayes decision rule: bayes theorem, bayes classifier and error rate of bayes classifier • Minimum distance classifier and linear discriminant function as derived from Bayes decision rule
Week 3	• Naive bayes classifier: gaussian model, multinomial model, bernoulli model • k-Nearest Neighbor (kNN) decision rule: idea of kNN classifier, distance weighted kNN decision rule and other variations of kNN decision rule
Week 4	• Perceptron learning algorithm: incremental and batch version, proof of convergence • XOR problem, two layer perceptrons to resolve XOR problem, introduction to multilayer perceptrons
Week 5	• Discussion on different aspects of linear discriminant functions for data classification • Logistic regression and maximum margin classifier
Week 6	• Support vector machine (SVM): hard margin • Soft margin SVM classifier
Week 7	• Cross validation and parameter tuning • Different techniques to evaluate the classifiers e.g., precision, recall and f-measure
Week 8	• The basics to work with Scikit-learn: a machine learning repository in python • How to implement different classifiers in scikit-learn, tune the parameters and evaluate the performance
Week 9	• Text classification(case study for data classification): overview of text data, stemming and stopword removal, tf-idf weighting scheme and n-gram approach. • How to work with text data in scikit-learn
Week 10	• Assignment 2: Evaluate the performance of different classifiers to classify a newswire e.g., Reuters-21578. • Review for midterm exam • Data clustering: overview, cluster validity index
Week 11	• Partitional clustering methods: k-means, bisecting k-means • k-medoid, buckshot clustering techniques
Week 12	• Hierarchical clustering techniques: single linkage, average linkage and group average hierarchical clustering algorithms • Density based clustering technique e.g., DBSCAN
Week 13	• Feature engineering: overview of feature selection, supervised and unsupervised feature selection techniques • Overview of principal component analysis for feature extraction
Week 14	• How to work with Wordnet, an English lexical database • Sentiment analysis (case study for data clustering): overview, description of a data set of interest for sentiment identification, sentiment analysis using Wordnet
Week 15	• Assignment 2: Sentiment analysis from short message texts • Practice class for the second assignment • Review for endterm exam

Syllabus for the Computer Architecture Course

Class.no	Course Materials to be taught
1	Fundamental Concepts and ISA The von Neumann Model Von Neumann vs Dataflow ISA vs. Microarchitecture
2	ISA Tradeoffs -I
3	ISA Tradeoffs -II
4	Intro to Microarchitecture: Single-Cycle
5	Multi-Cycle and Microprogrammed Microarchitectures
6	Pipelining
7	Introduction to Verilog
8	LAB
9	Branch Prediction I
10	Introduction to TEJAS simulatorr
11	LAB
12	Branch Prediction II
13	Out-of-Order Execution
14	Memory Hierarchy and Caches
15	High Performance Caches
16	Virtual Memory

- Few Homework and Lab assignment are also included.
- Few topics after *pipelining* are very intense , so it may be that if students are not very comfortable ,due to limited class and time , we may trim down the syllabus.

CS301

Theory of Computation

Time: Wed & Fri (12 noon—2 pm)
Place: MB215

Sarvottamananda

`sarvottamananda@rkmvu.ac.in, sarvottamananda@gmail.com`
url: `http://cs.rkmvu.ac.in/s̃arvottamananda/`
Office: MB115, Medhabhavan, RKMVERI, Belur
Office Hours: 11 pm—12 noon, 4 pm—5 pm
`(+91) 98740 94516`

Course Description: CS301 deals with topics in computability theory and computational complexity. In particular, the course will cover different models of computation, their associated complexity classes, undecidability, intractability, space and time complexity classes, oracle turing machines, circuit complexity, and related topics.

Prerequisite(s): (1) Discrete Mathematics and (2) Automata Theory.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url: `http://cs.rkmvu.ac.in/courses/cs301/`
Credit Hours: 4

Text(s):

Introduction to Automata Theory, Languages, and Computation, third edition
John E. Hopcroft, Rajeev Motwani & Jeffery D. Ullman
ISBN-13: 978-8131720479

Introduction to the Theory of Computation, second edition
Michael Sipser
ISBN-10: 81-315-0162-0

Computational Complexity: A Modern Approach, first edition
Sanjeev Arora & Boaz Barak
ISBN-13: 978-0-521-42426-4

Course Objectives:

Knowledge acquired: (1) Different models of computation,
(2) their associated complexity classes, and,
(3) reducibility.
Skills gained: The students will be able to
(1) analyze the complexity classes of problems closely related to those discussed in the class,
(2) analyze intractability and undecidability of some practical problems, and,
(3) do reductions based on knowledge gained in the class.

Grade Distribution:

Assignments	20%
Quizzes	20%
Midterm Exam	20%
Final Exam	40%

Grading Policy: There will be relative grading such that the cutoff for A grade will not be less than 75% and cutoff for F grade will not be more than 34.9%. Grade distribution will follow normal bell curve (usually, A: $\geq \mu + 3\sigma/2$, B: $\mu + \sigma/2 \ldots \mu + 3\sigma/2$ C: $\mu - \sigma/2 \ldots \mu + \sigma/2$, D: $\mu - 3\sigma/2 \ldots \mu - \sigma/2$, and F: $< \mu - 3\sigma/2$)

Approximate grade assignments:

>= 90.0	A+
75.0 – 89.9	A
60.0 – 74.9	B
50.0 – 59.9	C
about 35.0 – 49.9	D
<= 34.9	F

Course Policies:

- **General**

  1. Computing devices are not to be used during any exams unless instructed to do so.
  2. Quizzes and exams are closed books and closed notes.
  3. Quizzes are unannounced but they are frequently held after a topic has been covered.
  4. **No makeup quizzes or exams will be given.**

- **Grades**

  Grades in the **C** range represent performance that **meets expectations**; Grades in the **B** range represent performance that is **substantially better** than the expectations; Grades in the **A** range represent work that is **excellent**.

- **Labs and Assignments**

  1. Students are expected to work independently. **Offering** and **accepting** solutions from others is an act of dishonesty and students can be penalized according to the *Academic Honesty Policy*. Discussion amongst students is encouraged, but when in doubt, direct your questions to the professor, tutor, or lab assistant. Many students find it helpful to consult their peers while doing assignments. This practice is legitimate and to be expected. However, it is not acceptable practice to pool thoughts and produce common answers. To avoid this situation, it is suggested that students not write anything down during such talks, but keep mental notes for later development of their own.
  2. **No late assignments will be accepted under any circumstances**.

- **Attendance and Absences**

  1. Attendance is expected and will be taken each class. Students are not supposed to miss class without prior notice/permission. Any absences may result in point and/or grade deductions.
  2. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to get all missing notes or materials.

Course Outline (tentative) and Syllabus:
The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	**Content**
Week 1	• Finite Automata: Basic definitions, equivalence of finite automata, mille and muller automata, definition and acceptance criteria of timed and hybrid automata • Reading assignment: Chapter 2, HMU
Week 2	• Regular Expressions and Languages: definition of regular expressions and regular languages, relationship with finite automata, regulation expression algebra • Reading assignment: Chapter 3, HMU • Home assignment 1 • Quiz 1
Week 3	• Properties of Regular Languages: Pumping lemma for regular languages, Myhill-Nerode theorem and minimization of finite automata, closure properties of regular languages, decision problems and algorithms for regular languages • Reading assignment: Chapter 4: HMU
Week 4	• Context Free Grammar and Languages: Definition of context free grammars and context free languages, parse trees, ambiguity in grammars and inherent ambiguity in languages, context sensitive languages • Reading assignment: Chapter 5, HMU
Week 5	• Pushdown Automata: Definition of pushdown automata, languages of pushdown automata, equivalence of pushdown automata and context free grammars, deterministic pushdown automata and its language class • Reading assignment: Chapter 6, HMU • Home assignment 2 • Quiz 2
Week 6	• Properties of Context Free Languages: Normal forms, pumping lemma for context free languages, closure properties of context free languages, decision properties of context free languages • Reading assignment: Chapter 7, HMU
Week 7	• Turing Machines: Halting problem, definition of Turing machines, its extensions, restrictions and their equivalences, linear bounded automata and its relationship with context sensitive languages • Reading assignment: Chapter 8, HMU • Home assignment 3 • Quiz 3
Week 8	• Undecidability and Intractability: class of recursive languages and recursively enumerable languages, non recursively enumerable diagonalization language L_d, undecidable recursively enumerable language L_u, Rice's theorem, Post's correspondence problem, reductions, classes P and NP, NP-complete problem 3SAT • Reading Assignment: Chapter 9 & 10, HMU • Review for Midterm Exam

Week	Content
Week 9	• Computational Model for space and time complexity classes: Defining Turing Machine model, efficiency and running time, machine representation, universal turing machine, efficient simulation of universal turing machine, class P • Reading assignment: Chapter 1, AB • Home assignment 4
Week 10	• NP and NP-completeness: Definition of class NP, reducibility, NP-completeness, Cook-Levin theorem, web of reductions, definitions of coNP, EXP, and NEXP • Reading assignment: Chapter 2, AB • Home assignment 5 • Quiz 4
Week 11	• Diagonalization: Time hierarchy theorem, nondeterministic time hierarchy theorem, Ladner's theorem, oracle machines and limits of diagonalization, Baker-Gill-Solovay theorem • Reading assignment: Chapter 3, AB • Home assignment 6
Week 12	• Space complexity: Space hierarchy thorems, class PSPACE, PSPACE-completeness, class NL and coNL, NL-completeness, NL=coNL, Savitch'e theorem • Reading assignment: Chapter 4, AB • Home assignment 7 • Quiz 5
Week 13	• Polynomial Hierarchy and Alternations: Class Σ_2^p, polynomial hierarchy, class PH, alternating Turing machines • Reading assignment: Chapter 5, AB • Home assignment 8
Week 14	• Boolean Circuits: Boolean circuits, class $P_{/poly}$, uniformly generated circuits, Turing machines with advice, circuit lower bounds, nonuniform hierarchy theorem, circuits of exponential size • Reading assignment: Chapter 6, AB
Week 15	• Randomized Computation: Probabilistic Turing machines, classes RP, coRP, ZPP, BPP, relationships between BPP and other classes, randomized reductions, randomized space bound computations • Reading assignment: Chapter 7, AB • Review for Final Exam

CS220
Data and File Structures

Course Description: This course introduces the study of internal and external data structures and algorithms with an on-going emphasis on the application of software engineering principles. Trees, graphs and the basic algorithms for creating, manipulating and using them will be covered. Various types of hash and indexed random access file structures will be discussed and implemented. B-trees and external file sorting will be introduced. Internal and external data/file organizations and algorithms will be compared and analyzed.

Prerequisite(s): (1) Programming in C/C++/JAVA/Python.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Credit Hours: 4

Text(s): *Data Structures and Algorithms in JAVA*
Michael T. Goodrich, Roberto Tamassia, and Michael H. Goldwasser

Fundamentals of Data structures
Horowitz, E., and Sahni.S:

File Structures in C++
Folk & Zoellick & Riccardi

Data structures and algorithm analysis in C
Mark Allen Weiss

Course Objectives: Having completed this course successfully, the student should:

1. Be familiar with the use of data structures as the foundational base for computer solutions to problems.
2. Become introduced to and investigate the differing logical relationships among various data items.
3. Understand the generic principles of computer programming as applied to sophisticated data structures.
4. Comprehend alternative implementations using the differing logical relationships and appreciate the significance of choosing a particular logical relationship for implementation within real-world setting.
5. Demonstrate the ability to plan, design, execute and document sophisticated technical programs to handle various sorts of data structures.
6. become introduced the most important high-level file structures tools which include indexing, co-sequential processing, B trees, Hashing.

7. know the techniques for organization and manipulation of data in secondary storage including the low level aspects of file manipulation which include basic file operations, secondary storage devices and system software.

Course Outline (tentative) and Syllabus:
The weekly coverage might change as it depends on the progress of the class. Each week assumes 4 hour lectures.

Week	Content
Week 1	• Introduction to algorithm analysis: pseudo code, algorithm efficiency, asymptotic and empirical analysis of algorithms.
Week 2	• Introduction to data structures. Linear data structures: arrays, stacks, queues, linked lists (operations, implementations, applications.)
Week 3	• Non-linear data structures: binary trees and general trees (operations, implementations and applications). Binary search trees.
Week 4	• Priority queues and heaps: using a heap to implement a priority queue. Heap sort.
Week 5	• Balanced search trees: AVL trees
Week 6	• (2,4) and red-black trees.
Week 7	• B-trees and B^+ trees
Week 8	• Amortized Analysis, Splay tree
Week 9	• Hashing, Hash fucntions and collision resolution techniques -linear probing
Week 10	• Hashing and collision resolution techniques - quadratic probing, Double hashing
Week 11	• Graphs and elementary Graph operations - Breadth First Search, Depth First Search
Week 12	• Spanning Trees, Shortest paths
Week 13	• File Structure: Concepts of fields, records and files, Sequential, Indexed and Relative/Random File Organization
Week 14	• Indexing structure for index files, hashing for direct files
Week 15	• Multi-Key file organization and access methods.

CS 244 : Introduction to Optimization Techniques

Course Overview: The process of making optimal judgement according to various criteria is known as the science of decision making. A mathematical programming problem, also known as an optimization problem, is a special class of problem where we are concerned with the optimal use of limited resources to meet some desired objective(s). Mathematical models (simulation based and/or analytical based) are used in providing guidelines for making effective decisions under constraints. This course covers three major analytical topics in mathematical programming [linear, nonlinear and integer programming]. On each topic, the theory and modeling aspects are discussed first, and subsequently solution techniques or algorithms are covered.

Prerequisite(s): Linear Algebra
Credit Hours: 4
Course Objectives: Optimization techniques are used in various fields like machine learning, graph theory, VLSI design and complex networks. In all these applications/fields, mathematical programming theory supplies the notion of optimal solution via the optimality conditions, and mathematical programming algorithms provide tools for training and/or solving large scale models. Students will have knowledge of theory and applications of several classes of math programs.

Text(s): The course material will be drawn from multiple book chapters, journal articles, reviewed tutorials etc. However, the following two books are recommended texts for this course.

- *Linear programming and Network Flows*, Wiley-Blackwell; 4th Edition, 2010
  M. S. Bazaraa, John J. Jarvis and Hanif D. Sheral, **ISBN-13:** 978-0470462720
- *Nonlinear Programming: Theory and Algorithms*, Wiley-Blackwell; 3rd Edition (2006)
  M. S. Bazaraa, Hanif D. Sherali, C. M. Shetty, **ISBN-13:** 978-0471486008

Course Policies:

- **Grades**

  Grades in the **C** range represent performance that **meets expectations**; Grades in the **B** range represent performance that is **substantially better** than the expectations; Grades in the **A** range represent work that is **excellent**.

- **Assignments**

  1. Students are expected to work independently. Discussion amongst students is encouraged but offering and accepting solutions from others is an act of dishonesty and students can be penalized according to the *Academic Honesty Policy*.
  2. **No late assignments will be accepted under any circumstances**.

- **Attendance and Absence**

  Students are not supposed to miss class without prior notice/permission. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to get all missing notes or materials.

Grade Distribution:

Assignments	40%
Midterm Exam	20%
Final Exam	40%

Grading Policy: *Approximate grade assignments:*

>= 90.0 %	A+
75.0 – 89.9 %	A
60.0 – 74.9 %	B
50.0 – 59.9 %	C
about 35.0 – 49.9 %	D
<= 34.9%	F

Table 1: **Topics Covered**

Mathematical Preliminaries

- Theory of Sets and Functions,
- Vctor spaces,
- Matrices and Determinants,
- Convex sets and convex cones,
- Convex and concave functions,
- Generalized concavity

Linear Programming

- The (Conventional) Linear Programming Model
- The Simplex Method: Tableau And Computation
- Special Simplex Method And Implementations
- Duality And Sensitivity Analysis

Integer Programming

- Formulating Integer Programing Problems
- Solving Integer Programs (Branch-and-Bound Enumeration, Implicit Enumeration, Cutting Plane Methods)

Nonlinear Programming: Theory

- Constrained Optimization Problem (equality and inequality constraints)
- Necessary and Suffiecent conditions
- Constraint Qualification
- Lanrangian Duality and Saddle Point Optimality Criteria

Nonlinear Programming: Algorithms

- The concept of Algorithm
- Algorithms for Uconstrained Optimization
- Constraint Qualification
- Algorithms for Constrained Optimization (Penalty Function, Barrier Function, Feasible Direction)

Special Topics (if time permits)

- Semi-definite and Semi-infinte Programs
- Quadratic Programming
- Linear Fractional programming
- Separable Programming

DA230
Enabling Technologies for Big Data Computing

Instructor

Sudeep Mallick, Ph.D.
Sudeep.mallick@gmail.com

Course Description:

DA230 deals with technologies and engineering solutions for enabling big data processing and analytics . More specifically, it deals with the tools for data processing, data management and programming in the distributed programming paradigm using techniques of MapReduce programming, NoSQL distributed databases, streaming data processing, data injestion, graph processing and distributed machine learning for big data use cases.

Prerequisite(s): (1) Basic knowledge of python and Java programming languages (2) Tabular data processing / SQL queries. (3) Basic knowledge of common machine learning algorithms.
Credit Hours: 4

Text(s):

Hadoop: The Definitive Guide, fourth edition
Tom White
ISBN: 978-1-491-90163-2

Hadoop in Action, edition: 2011
Chuck Lam
ISBN: 978-1-935-18219-1

Spark in Action, edition: 2017
Petar Zecevic & Marko Bonaci
ISBN: 978-93-5119-948-9

Data-Intensive Text Processing with MapReduce, edition: 2010
Jimmy Lin & Chris Dyer
ISBN: 978-1-608-45342-9

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. Each week assumes 4 hour lectures.

Week	Content
Week 1	• Big data computing paradigm and Hadoop: big data, hadoop architecture • Reading assignment: Chapter 1, LD & Chapter 1, TW • Lab: setting up Hadoop platform in standalone mode
Week 2	• Hadoop MapReduce (MR): Lab session with simple MR algorithms in Hadoop standalone mode • Reading assignment: Chapter 2, LD & Chapter 2, TW
Week 3	• Hadoop Distributed File System (HDFS), YARN and MR architecture, daemons, serialization concept, command line parameters: Lab session • Reading assignment: Chapter 3-5 & 7, TW
Week 4	• Implementing algorithms in MR - joins, sort, text processing, etc.: Lab session • Reading assignment: Chapter 3, LD & Chapter 7, TW • Lab assignment 1
Week 5	• Hadoop operations in Cluster Mode, Hadoop on AWS Cloud: Lab session • Reading assignment: Instructor notes
Week 6	• Understanding NoSQL using Pig: Lab Session • Reading assignment: Chapter 16, TW • Lab assignment 2
Week 7	• Introduction to Apache Spark platform and architecture, RDD, • Reading assignment: Chapters 1-3, ZB
Week 8	• Mapping, joining, sorting, grouping data with Spark RDD: Lab session • Reading assignment: Chapter 4, ZB • Review for Mid term exam
Week 9	• Advanced usage of Spark API: Lab session • Reading assignment: Chapter 4, ZB • Lab assignment 3
Week 10	• NoSQL queries using Spark DataFrame and Spark SQL: Lab session • Reading assignment: Chapter 5, ZB
Week 11	• Using SQL Commands with Spark: Lab session • Reading assignment: Chapter 5, ZB
Week 12	• Machine Learning using Spark MLib: Lab session • Reading assignment: Chapter 7, ZB
Week 13	• Machine Learning using Spark ML: Lab session • Reading assignment: Chapter 8, ZB • Lab assignment 4
Week 14	• Spark operations in Cluster Mode, Spark on AWS Cloud: Lab session • Reading assignment: Chapter 11, ZB
Week 15	• Graph processing with Spark GraphX: Lab session • Reading assignment: Chapter 9, ZB

DA104 Probability and Stochastic Processes

Instructor

Dr. Arijit Chakraborty (ISI Kolkata)

Course Description:

DA104 deals with technologies and engineering solutions for enabling big data processing and analytics . More specifically, it deals with the tools for data processing, data management and programming in the distributed programming paradigm using techniques of MapReduce programming, NoSQL distributed databases, streaming data processing, data injestion, graph processing and distributed machine learning for big data use cases.

Prerequisite(s): (1) Basic knowledge of python and Java programming languages (2) Tabular data processing / SQL queries. (3) Basic knowledge of common machine learning algorithms.
Credit Hours: 4

Text(s):

1. Introduction to time series analysis; PJ Brockwell and RA Davis
2. Time Series Analysis and Its Applications; Robert H. Shumway and David S. Stoffer
3. Introduction to Statistical time series; WA Fuller
4. A first course in Probability, Sheldon Ross, Pearson Education, 2010
5. Time Series Analysis; Wilfredo Palma
6. P. G. Hoel, S. C. Port and C. J. Stone: Introduction to Probability Theory, University Book Stall/Houghton Mifflin, New Delhi/New York, 1998/1971.

Syllabus

1. Basic Probability
a. Introduction
b. Sample Spaces
c. Probability Measures
d. Computing Probabilities: Counting Methods
  i. The Multiplication Principle
  ii. Permutations and Combinations
e. Conditional Probability
f. Independence

2. Random Variables
a. Discrete Random Variables
  i. Bernoulli Random Variables
  ii. The Binomial Distribution
  iii. Geometric and Negative Binomial Distributions
  iv. The Hypergeometric Distribution
  v. The Poisson Distribution
b. Continuous Random Variables

i. The Exponential Density
ii. The Gamma Density
iii. The Normal Distribution
iv. The Beta Density
c. Functions of a Random Variable

3. Joint Distributions
a. Introduction
b. Discrete Random Variables
c. Continuous Random Variables
d. Independent Random Variables
e. Conditional Distributions
i. The Discrete Case
ii. The Continuous Case
f. Functions of Jointly Distributed Random Variables
i. Sums and Quotients
ii. The General Case

4. Expected Values

a. The Expected Value of a Random Variable
i. Expectations of Functions of Random Variables
ii. Expectation of Linear Combinations of Random Variables
b. Variance and Standard Deviation
c. Covariance and Correlation
d. Conditional Expectation
e. Definitions and Examples
f. The Moment-Generating Function

5. Limit Theorems
a. Introduction
b. The Law of Large Numbers
c. Convergence in Distribution and the Central Limit Theorem

6. Stochastic Process
a. Markov chain
i. State transition matrix
ii. Hitting time
iii. Different States
b. Poisson process

CS211
Combinatorial optimization and Graph Algorithms

Instructor: Dhyanagamyananda swathyprabhu@gmail.com

Course Description: CS211 is the first course to deal with the topic of this course. This course is a mixed bag of graph algorithms. Some of these algorithms are relevant in the context of optimization. The field of graph algorithms is vast and the kind of problem studied in CS211 are those that are in general difficult to solve but has easy solutions for a sub-class of them.

Prerequisite(s): Design and Analysis of Algorithms, Data and File Structures.
Credit Hours: 4

Text(s):

Algorithm Design, PHI
Kleinberg & Targos

Introduction to Graph Theory
Douglas West

Lecture Notes from University of Waterloo

Draft on Discharging techinque by Douglas West

Course Outcome:

Knowledge gained: At the end of the course the student will know about
(1) flow networks
(2) the theory of various graph classes like planar graphs, perfect graphs, comparability graphs, etc
(3) algorithm design and analysis related to coloring, matching, perfectness, shortest distance, spanning trees etc.
(4) linear optimization in graph problems
(5) structural properties of graphs

Skills acquired: The students will be able to
(1) perform graph modelling for computing parameters in real life problems
(2) encode graphs in the programming context
(3) apply techniques like discharging method in analysis of planar graphs
(4) graph programming using native python libraries, and specialized sagemath tool.
(5) apply dynamic pogramming in solving specific graph problems

Competence developed: The student will be able to.
(1) read and understand research contributions in the field of structural graph theory and ask pertinent questions
(2) work on research problems related to graph algorithms, and structural graph theory

Grade Distribution:

Assignments 20%, Quizzes 10%, Midterm Exam 20%, Final Exam 40%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Network Flow: Definition, Basic Idea, Algorithm, Maxflow mincut theorem, Ford Fulkerson Algorithm Analysis, LP formulation of maxflow and proof. • Reading assignment: Chapter 3, KT
Week 2	• Layered Network: Definition, Theorem, Computation of blocking flow (Edmonds, Dinics, MPM) • Reading assignment: XBitmap from Wiki. • Programming assignment 1: • Quiz 1
Week 3	• Student presentation of Tarzan's algorithm
Week 4	• Bipartite matching: Definition, Application, Using Ford Fulkerson Algorithm bipartite matching is obtained in O(—V——E—) time Edge connecting problem. The augmenting path algorithm for bipartite matching. • Reading Assignment:
Week 5	• Matching for Non-Bipartite Graph: Theorem and proof (Edmonds blossom shrinking) • Reading Assignment: • Quiz 2
Week 6, 7	• Max-Cut: NP-Hard problem and its proof, 2-Approximation algorithm, Randomized algorithm for max-cut, De-randomization LP based approximation algo for maxcut • Reading assignment:
Week 8,9	• Interval Graph: Intersection graph, Perfect elimination order (PEO), Chordal graph (Triangulated Graph), Simplicial vertex, Algorithm MIS, vertex cover, coloring, clique cover for interval graph, Finding a PEO Comparability graph • Reading assignment: Waterloo Lecture Notes • Home assignment: • Quiz 3

Week	**Content**
Week 10,11,12	• Trees and Friends, Trees, Treewidth, Tree decomposition, Closure properties, Partial k-trees, Partial k-trees to tree decomposition, Tree decomposition to partial k-trees, Dynamic programming MIS algo for partial k-tree • Home assignment 4 • Quiz at the end of three weeks.
Week 13,14	• Perfect Graph, Definition and properties, Perfect graph theorem, Triangulated graph is a perfect graph • Home assignment 7 • Quiz 5
Week 15	• Discharging method • Reading assignment: DW on discharging • Home assignment 8

CS214 Applied Computer Graphics

Session: 2017-2018, Semester-II

Instructor: **Sarvottamananda**

Course Description: The course CS214—*Applied Computer Graphics* aims to teach modern OpenGL programming fundamentals to students. We mainly stress on 3D programming including modeling, lighting, shadow, animationa techniques. This course also teaches how to make use of modern hardware to efficiently render 3D scenes.

Prerequisite(s): Introduction to Computer Graphics, Linear Algebra, C++/C Programming
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url: `http://cs.rkmvu.ac.in/cs214`
Credit: 4 (four), approximately 60 credit hours

Text(s):

Computer Graphics with OpenGL by Donald Hearn and M Pauline Baker, third edition

Computer Graphics: Principles and Practice by John F Hughes, Andries Van Dam, Morgan Mcguire, David F Sklar, James D Foley, Steven K Feiner, and Kurt Akeley, third edition

Learn OpenGL: An offline transcript of learnopengl.com Joey de Vries, web resource, Jan 2018

Learning Modern 3D Graphics Programming by Jason L. McKesson, web resource, Jan 2018

OpenGL Programming Guide — the red book by John Kessenich Graham Sellers and Dave Shreiner, ninth edition

The OpenGL Graphics System: A specification (v4.5) — the blue book by Mark Segal and Kurt Akeley

Course Objectives:

Knowledge acquired :

- OpenGL graphics pipeline for different versions
- Concepts of 3D programming
- Mathematical foundations of 3D computer graphics
- Lighting and shadow models
- Ray tracing method
- Theory of animation

Skills gained :

- 3D models with loading, saving and display in OpenGL
- 3D Lights, creation and manupulation
- 3D Shadows modeling
- Rendering of 3D environments
- Basic 3D animation
- Basic 3D effects

General Competence :

- Present the solution to a defined computer graphics problem orally, and answer question about the solution
- Read and integrate academic material from various online sources
- Improved software development skills
- Learning version control and static code analysis
- Improvement in asking quality questions

Other goals : Understanding basic GPU hardware

Course Grading Policy:

There will be flexi-relative grading such that the cutoff for 'A' grade will not be less than 75% and cutoff for 'F' grade will not be more than 34.9%. Grade distribution will follow the normal bell curve.

Approximate grade assignments:

>= 90.0	A+	$\geq \mu + 2\sigma$
75.0 – 89.9	A	$\mu + \sigma \ldots \mu + 2\sigma$
60.0 – 74.9	B	$\mu \ldots \mu + \sigma$
50.0 – 59.9	C	$\mu - \sigma \ldots \mu$
about 35.0 – 49.9	D	$\mu - 2\sigma \ldots \mu - \sigma$
<= 34.9	F	$< \mu - 2\sigma$

Approximate weightage of different components in evaluation:

Assignments (8)	20%
Quizzes (6)	20%
Midterm Exam	20%
Final Exam	40%

Course Policies:

- **General**
  1. Computing devices are not to be used during any exams unless instructed to do so.
  2. Quizzes and exams are closed books and closed notes.
  3. Quizzes are unannounced but they are frequently held after a topic has been covered.
  4. **No makeup quizzes or exams will be given.**
- **Grades**

  Grades in the **C** range represent performance that **meets expectations**; Grades in the **B** range represent performance that is **substantially better** than the expectations; Grades in the **A** range represent work that is **excellent**.
- **Labs and Assignments**
  1. Students are expected to work independently. **Offering** and **accepting** solutions from others is an act of dishonesty and students can be penalized according to the *Academic Honesty Policy*. Discussion amongst students is encouraged, but when in doubt, direct your questions to the professor, tutor, or lab assistant. Many students find it helpful to consult their peers while doing assignments. This practice is legitimate and to be expected. However, it is not acceptable practice to pool thoughts and produce common answers. To avoid this situation, it is suggested that students not write anything down during such talks, but keep mental notes for later development of their own.
  2. **No late assignments will be accepted under any circumstances**.

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 lecture hours. Quizzes will be unannounced, so students should maintain close to 100% attendance.

Week	**Content**
Week 1	• Introduction to OpenGL • Reading assignment: Introductions in all references
Week 2	• OpenGL pipeline 1.x, 2.x, 3.x and 4.x, specifications and differences • Reading assignment: Chapter 1, redbook • Home assignment 1 • Quiz 1
Week 3	• Vertex and Fragment Shaders, fundamentals and basic programming Analysis of a simple beginner OpenGL 4.x program • Reading assignment: Chapter 2 & Chapter 3 of LO
Week 4	• OpenGL data representation Data structures for graphics • Reading assignment: Chapter 2, blue book
Week 5	• Mathematics for 3D computer graphics Transformations • Reading assignment: Relevant chapters in CG w/OpenGL • Home assignment 2 • Quiz 2
Week 6	• Model, view and projection matrices local space, world space, view space and clip space • Reading assignment: Chapter 8,9 LO
Week 7	• Textures, storing and using • Reading assignment: Chapter 7, LO • Home assignment 3 • Quiz 3
Week 8	• Basic color theory, physical, physiological and psychological explanations of color perception • Reading Assignment: Relevant Chapters of CG w/OpenGL • Review for Midterm Exam

Week	Content
Week 9	• Basic lighting and materials point lights, spot lights, directional light • Reading assignment: Chapter 13,14 LO • Home assignment 4
Week 10	• Shadows mapping, techniques and approximations • Reading assignment: Chapter 35, LO • Home assignment 5 • Quiz 4
Week 11	• Cubemaps, usage and programming • Reading assignment: Chapter 27, LO • Home assignment 6
Week 12	• Basic animation, data structures and programming • Reading assignment: Chapter 4, AB • Home assignment 7 • Quiz 5
Week 13	• Geometry Shaders • Reading assignment: Chapter 30, LO • Home assignment 8
Week 14	• Tessalation Shaders • Reading assignment: Chapter 9, redbook
Week 15	• Compute Shaders • Reading assignment: Chapter 12, redbook • Review for Final Exam

CS229 Programming Handheld Devices

Instructor: **Sudeep Mallick**

Course Description: CS128 is the first course in design and programming of handheld devices using Android technology platform covering the essential components of the Android platform. The course enables students to develop and publish apps developed with Java/Kotlin based language using Android Studio IDE and Android programming API from Google following Android platform best practices. It enables appreciation of the architecture of the android platform and various enabling components for web access, data persistence, multi-threading, UI management among others

Prerequisite(s): (1) Basic knowledge of Java programming language or any other programming language (2) Software analysis and design concepts.
Credit Hours: 4

Text(s):

1. Head First: Android Development, first edition (2015) Dawn Griffiths & David Griffiths
2. Android Programming: The Big Nerd Ranch Guide, edition: 2013 Bill Phillips & Brian Hanrdy ISBN: 10-0321804333
3. Android Developer `https://developer.android.com/`
4. Android Studio `https://developer.android.com/studio/`
5. Android Developer Fundamentals `https://developers.google.com/training/courses/android-fundamentals`
6. Android Developer Guide `https://developer.android.com/guide/`
7. Documentation for App Developers `https://developer.android.com/docs/`

Course Objectives:

Knowledge gained: At the finish of the course the student will know

1) Architecture of a typical mobile app using MVC architectural pattern.
2) Architecture of the Android platform.
3) App development using Android platform APIs and Android platform best practices for app development software lifecycle.
4) Android component life cycle management and navigation styles.
5) Importance of the concept of fragments, layouts, inter-component communication, taskbars, menus, navigation drawers

Skills acquired: The students will be able to

1) design and build app components, layouts, and navigation structure based on given set of application features (requirements).
2) build useful Android applications like location-based app, map-based app, content management app, multimedia app, web based and social media app along with rich user interfaces by utilizing existing or custom Android Views and layouts.
3) take advantage of Android background services, AsyncTask, SQLite database, Preferences, content providers and notifications.
4) implement object oriented design patterns, and architectural patterns.
5) Implement an app by going through software development lifecycle of analysis, design, implementation, integration and testing.

6) build their own Android apps and deploy in Google Playstore.
7) Ability to design and develop user interface (UI) using Android development best practices

Competence developed: The student develop the

1. Ability to transform an idea of an app to an implemented code base using Android platform by following an engineering approach.
2. Ability to take advantage of latest, advanced features of android platform such as location service, maps, broadcasting, material design and such others.

Evaluation:

Midterm Lab Exam 20%   Term Project 40%   Endterm Theory Exam 40%

Course Outline (tentative) and Syllabus:

Week 1	Getting started - introduction to Android Studio; Android Project structure; App basics; creating the First Basic App and using Emulator and device; Android MVC - using MVC for building app
Week 2	Activities and Lifecycle
Week 3	Communication among activities and Intents
Week 4	Android UI Fragments
Week 5	UI Fragment design best practices
Week 6	Android Lists and Adapters- RecyclerView, ListView, etc.
Week 7	Android Layouts and Widgets
Week 8	Managing master-detail interfaces, Navigation Drawers Review for mid-term exam
Week 9	ViewPager
Week 10	Dialogs and Fragments
Week 11	Toolbar and Action Bars
Week 12	SQLite Databases, Cursors and Asynctasks
Week 13	Implicit Intents, background Tasks and background Services
Week 14	HTTP tasks Review for end-term exam
Week 15	Project Presentations

CS323 Discrete Event Systems

Instructor:

Course Description: A discrete event system is a mathematical model of a system (such as computational device) that communicates with its environment by atomic actions (called events). For example, a user of the system pressing a button could send a signal to a controller. These events are assumed to be discrete in the sense that they occur instantaneously (as opposed to over a period of time). The module will present an overview of various modelling and analysis techniques for discrete event systems. We start by looking at sequential systems (where no two events can occur simultaneously). Systems of this kind will be modelled by finite automata. This class is then extended to allow for events occurring simultaneously; these are modelled by Petri nets. Subsequently, we will study techniques that allow us to extract quantitative information about the behaviour of systems. This gives rise to the class of probabilistic systems (where we assume that a certain event occurs with a given probability) and we can then estimate the likelihood of situations such as system failure. Included in this section is an introduction to queuing theory.

Prerequisite(s): CS200 Automata theory.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url:
Credit Hours: 4

Text(s):

- Introduction to Discrete Event Systems , Christos G.Cassandras, Stphane Lafortune

Course Objectives:

Knowledge acquired: At the finish of this course, students will know how to employ some basic formalisms of behavioural modelling (such as automata and Petri nets) to model real world examples;

Skills gained: The students will be able to apply mathematical formalisms to model and analyse event driven systems

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments.

1. Systems and Models, Supervisory Control, Petri Nets, Timed Automata.

2. Systems and Control basics, Goal of system theory, Feedback control with supervisors, Control with partial controllability, Non-blocking control, Control with modular specifications, Control with partial observation, Decentralized control, Basics-Analysis-and-control of Petri Nets, Comparison of Petri Nets with automata, Timed Automata.

CS301 Computational Complexity

Instructor:

Course Description: CS250 deals with a detailed study of principles of RDBMS.

Prerequisite(s): CS300: Theory of NP-Completeness, CS200: Theory of Computation
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url:
Credit Hours: 4

Text(s):

- Computational Complexity: A Modern Approach by Sanjeeve Arora and Boaz Barak
- Computational Complexity by Papadimitrou

Course Objectives:

Knowledge acquired: At the finish of this course, students will be quite empowered and will know
Skills gained: The students will be able to
Competence Developed:

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments.

1. Basic resources for computation (time, space, nondeterminism) and their associated complexity classes (P, NP, PSPACE and more).
2. Relationships among resources (P versus NP, time versus space, and more.
3. Reductions and completeness (NP completeness, PSPACE completeness, and more.
4. Counting problems,$\tilde{\#}$P. Randomness as a computational resource; associated complexity classes.
5. Nonuniform models of computation; circuit complexity; lower bounds.
6. Communication complexity.
7. Interactive proofs & IP=PSPACE.
8. Probabilistically checkable proofs (PCP) and inapproximability.