DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

Scheme of Instructions and Syllabus
for Post Graduate Studies

M.Tech. in
Computer Science & Engineering
2015 - 2016

Visvesvaraya National Institute of Technology, Nagpur

July 21, 2015

MISSION AND VISION
OF
VISVESVARAYA NATIONAL INSTITUTE OF TECHNOLOGY, NAGPUR

MISSION

The Mission of VNIT is to achieve high standards of excellence in generating and propagating knowledge in engineering and allied disciplines. VNIT is committed to providing an education that combines rigorous academics with joy of discovery. The Institute encourages its community to engage in a dialogue with society to be able to effectively contribute for the betterment of humankind.

VISION

To contribute effectively to the national endeavor of producing quality human resource of world class standard by developing a sustainable technical education system to meet the changing technological needs of the Country, incorporating relevant social concerns and to build an environment to create and propagate innovative technologies for the economic development of the Nation.

MISSION AND VISION
OF
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING, VNIT, NAGPUR

MISSION

To produce highly qualified and motivated graduates through a rigorous curriculum of theory and application that develops the ability to solve problems, individually and in teams.

Creating knowledge of fundamental principles and innovative technologies through research within the core areas of computer science and also in inter-disciplinary topics.

Serving the communities to which we belong at local and national levels, combined with a deep awareness of our ethical responsibilities to our profession and to society.

VISION

To contribute effectively to the important national endeavour to produce quality human resource in the information technology and related areas for sustainable development of the country's IT industry needs.

To advance the state of the art in computer science and engineering by working on cutting edge research topics, publishing quality research papers and filing enduring patents.

To serve the local and the national community by creating awareness about IT related products and to

impress upon then the importance of knowledge management.

Department of Computer Science & Engineering came into being in 1994. It offers under-graduate and post-graduate programs. The department has well qualified and motivated faculty members and support staff. The laboratories are adequately equipped with state-of-the-art facilities. The department is actively involved in R&D as well as consultancy projects and has collaborations with several industries, academic institutes and R&D organizations in the country.

Department of Computer Science & Engineering a M.Tech. program, namely, M.Tech. Computer Science Engineering. This is a four semester program, wherein student has to complete certain number of credits as indicated in Table 1. Each subject (or course) has certain number of credits. There are two types of subjects: Core and elective. Core courses are compulsory and some courses from electives are to be taken to complete the required credits.

TABLE 1 : CREDIT REQUIREMENTS FOR POST GRADUATE STUDEIS

Program Core (PC)		**Program Elective (PE)**	
Category	**Credit**	**Category**	**Credit**
Basic Science (BS)	00	Departmental Electives (DE)	13-14
Engineering Science (ES)		Humanities & Management (HM)	0
Humanities (HU)	-	Open Courses (OC)	0
Departmental Core (DC)	39		
Total	**39**	**Total**	**13-14**
Grand Total PC+PE			**52-53**

The number of credits attached to a subject depends on number of classes in a week. For example a subject with 3-1-0 (L-T-P) means it has 3 Lectures, 1 Tutorial and 0 Practical in a week. This subject will have four credits (3x1+1x1+0x1 = 4). If a student is declared pass in a subject, then he / she gets the credits associated with that subject. Depending on marks scored in a subject, student is given a Grade. Each grade has got certain grade points as follows :

Grades	AA	AB	BB	BC	CC	CD	DD	FF
Grade Points	10	09	08	07	06	05	04	Fail

The performance of a student will be evaluated in terms of two indices, viz. the Semester Grade Point Average (SGPA) which is the Grade Point Average for a semester and Cumulative Grade Point Average (CGPA) which is the Grade Point Average for all the completed semesters at any point in time. SGPA & CGPA are :

$$\text{SGPA} = \frac{\sum_{\text{semester}}(\text{Course credits x Grade points}) \text{ for all courses except audit}}{\sum_{\text{semester}} (\text{Course credits}) \text{ for all courses except audit}}$$

Students can Audit a few subjects i.e. they can attend the classes and do home work and given exam also,

but they will not get any credit for that subject. Audit subjects are for self enhancement of students.

Programme : M.Tech. in Computer Science Engineering

Programme Educational Objectives of M.Tech. in Computer Science Engineering

1. Gain the ability to analyze and solve computer science and engineering problems through application of fundamental knowledge of mathematics and algorithms.

2. Learn to apply modern skills, techniques, and engineering tools to create computational systems. Understand the state of the art in the recent areas of research in computer science and engineering and to formulate problems from them and perform original work to contribute in the advancement of the state of the art.

3. To be able to adapt to the evolving technical challenges and changing career opportunities. Learn to effectively communicate ideas in oral, written, or graphical form and to promote collaboration with other members of engineering teams.

Programme Outcomes of M.Tech. in Computer Science Engineering

a) To obtain sound knowledge in the theory, principles and applications of computer systems.
b) Apply knowledge of mathematics and algorithms in the design and development of software systems.
c) Configure recent software tools, apply test conditions, and deploy and mange them on computer systems.
d) Perform experiments on different software packages either obtain from external parties or developed by themselves and analyse the experimental results.
e) Design and develop software projects given their specifications and within performance and cost constraints.
f) Identify, formulate and solve software engineering problems and understand the software project management principles.
g) Ability to understand the computing needs of inter-disciplinary scientific and engineering disciplines and design and develop algorithms and techniques for achieving these.
h) Acquire and understand new knowledge, use them to develop software products, and to understand

the importance of lifelong learning.

i) Ability to extend the state of art in some of the areas of interest and create new knowledge.

j) Communicate effectively in oral, written and graphical form.

k) Understand and formulate research problems and explore the current research being done.

l) Extend the state of art and explore new problems and solution techniques.

Scheme of Instructions for M.Tech. in Computer Science Engineering

I Semester				II Semester			
Code	**Course**	**L-T-P**	**Cr**	**Code**	**Course**	**L-T-P**	**Cr**
Core				**Core**			
CSL 514	Advances in Algorithms	3-0-0	3	CSL 519	Distributed Systems	3-0-2	4
CSL 522	Advances in Compiler Construction	3-0-2	4	CSL 520	Advanced DBMS	3-0-2	4
CSL 523	Advanced Computer Architecture	3-0-0	3	CSL 528	Cryptography & Information Security	3-0-0	3
CSP 502	Software Lab-I	1-0-2	2	CSP 504	Software Lab II	1-0-2	2
CSP 529	Technical Writing & Publishing	1-0-2	2				
Elective (Any one)				**Elective (Any one)**			
CSL 517	Pattern Recognition	3-0-2	4	CSL516	Soft Computing Techniques	3-0-0	3
CSL509	Cloud Computing	3-0-2	4	CSL521	Software Architecture	3-0-0	3
ECL412	Advanced Digital Signal Processing	3-0-2	4	CSL511	Advances in Data Mining	3-0-0	3
Total No. of Credits			**18**	**Total No. of Credits**			**16**

III Semester				IV Semester			
Code	**Course**	**L-T-P**	**Cr**	**Code**	**Course**	**L-T-P**	**Cr**
Core				**Core**			
CSD 501	Project Phase I		3	CSD 502	Project Phase II		9
Elective (Any Two)				**Elective**			
CSL 524	Real Time Systems	3-0-0	3				
CSL 539	Formal Methods in Program Design	3-0-0	3				

CSL 443	Information Retrieval	3-0-2	4				
CSL530	Topics in Bioinformatics	3-0-0	3				
CSL 515	Mobile Communication Systems	3-0-0	3				
Total No. of Credits			**9-10**	**Total No. of Credits**			**9**

Course Code and Title

CSL 514: Advances in Algorithms

1. **Course Description : Core Course**

   Asymptotic complexity, Study of the design technique for algorithms such as divide-and-conquer, greedy method, dynamic programming, backtracking and branch-and-bound, convex hull, closest pair of points, FFT, polynomial multiplication, primality testing, text pattern matching, study of the theory of NP-completeness, approximation algorithms, randomized algorithms

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite: Data Structures and Program Design**

3. **Detailed Description of the Course**

   - Asymptotic Notations-Big-Oh, Big-omega and Big-theta notations (1 week)
   - Recurrence relations –Substitution method (2 weeks)
     Change of variables, Master Method, Characteristic Equation, Generating Functions
   - Analysis of Algorithms-Best case, Worst case and Average case (1 weeek)
   - Divide-and-conquer-skeleton of the technique, binary search, quick sort, merge sort, (2 weeks)
   - FFT, closest pair of points, polynomial multiplication (1 week)
   - Greedy method-basic technique, job sequencing with deadlines, minimum spanning trees, all point shortest paths (1 week)
   - Dynamic Programming-basic fundamentals of the technique, application to multistage graph problem, longest common subsequence problem, travelling salesman problem (2 weeks)
   - Backtracking-basic fundamentals of the technique, application to n-queens problem, graph colouring problem, Hamiltonian cycles problem (1 week)
     Branch-and-bound-description of the technique, illustration through suitable Examples

- Theory of NP-completeness- Definition of the terms NP, NP-hard and NP-complete, showing NP-completeness of a problem (2 weeks)
- Approximation algorithms (1 week)
- Randomised algorithms (1 week)

4. Text books and/or other required material

- E. Horowitz, S. Sahni, S. Rajasekaran, Fundamentals of Computer Algorithms, University Press, Second Edition.
- Thomas H. Cormen et al., Introduction to Algorithms, PHI, Second Edition.

5. Course Objectives

- Appreciate the need for analysis of algorithms.
- How to analyze the best-case, average-case and worst-case running times of algorithms using asymptotic analysis.
- Know the standard design techniques of algorithms and know the conditions in which each particular technique is to be applied.
- Design efficient algorithms for problems encountered in common engineering design situations.
- Know the limitations on the time complexity of algorithms i.e. the theory of NP-completeness.
- Study approximation algorithms and randomized algorithms to address the limitations on the time complexity of complexity

6. Class Schedule

Lectures : Three 1-hr lectures per week

7. Contribution of Course to Professional Component

Students learn techniques for design of algorithms. They also learn to apply these techniques to various problems. They also learn how to analyze the algorithms; they know the limitations on the time complexity of algorithms and how to address the same.

8. Evaluation of Students

The students are evaluated through 2 sessional exams, end-semester examination.

Course Code and Title

CSL522 : Advances in Compiler Construction

1. **Course Description : Core Course**

   The study of construction of compilers and compiler optimizations based on data flow analysis and parallelizing compilers.

   Credit scheme - (L-T-P-C: 3-0-2-4)

2. **Required Background or Pre-requisite:** Basic Compilers course in B.Tech.

3. **Detailed Description of the Course**

   - Review of compiler fundamentals – lexical analysis, parsing, semantic analysis and intermediate code generation, error recovery, run time storage management, code generation. (4 weeks)

   - Code optimization – Peephole optimization, control flow analysis, data flow analysis, dependence analysis, redundancy elimination, loop optimization, procedural and interprocedural optimization, instruction scheduling. (3 weeks)

   - Compiling for High performance architectures, Compiling for scalar pipeline, compiling for vector pipeline, superscaler and VLIW processors, compiling for multiple issue processors, compiling for memory hierarchy.Parallelization and Vectorization, Dependence and dependence testing. (3 weeks)

   - Loop Normalization, Induction variable Exposure, Enhancing Fine Grained Parallelism, Loop Interchange, Scalar Expansion, Scalar and Array Renaming, Node splitting, Index-set splitting, Loop skewing. (4 weeks)

   Typical Laboratory Experiments: Assignments based on techniques covered.

4. **Text books and/or other required material**

   - Optimizing Compiler for Modern Architecture: A dependence based approach, Randy Allen, Kennedy
   - Advanced Compiler Design and implementation : Steven S. Muchnick
   - Engineering & Compiler : Keith D. Cooper & Linda Torczon: Morgan Kaufmann

5. Course Objectives

- Appreciation of parsing and code generation techniques
- Understanding of optimizations problems and issues, data flow analysis framework and mathematical modeling
- Appreciation of role of machine specific issues in compiler construction, the choice of instructions, the availability of registers etc.
- Ability to combine different optimization techniques to achieve the overall objective of program efficiency
- Appreciation of optimization techniques for multi-processor machines and parallelizing optimization schemes

6. Class/Laboratory Schedule

Lecture: Three 1-hour lectures per week
Lab: One 2 hrs session per week

7. Contribution of Course to Professional Component

Lecture: Students learn about different compiler optimization techniques, the role of processor architecture and applications of optimization techniques.

Lab: Students learn to implement different techniques to construct a compiler.

8. Evaluation of Students

The instructor uses the following methods: home-work assignments, 2 sessional exams, end-semester exam and course project, one-on-one discussions during office hours, laboratory experiments and programming assignments.

Course Code and Title

CSL 523 Advanced Computer Architecture

1. **Course Description : Core Course**

- To make aware the students about taxonomy of computer design, computer architectures, the technological trends that drive the industry. Appreciation of quantitative principles of computer design
- Appreciation of various issues related to performance enhancement. Understand memory organization, memory hierarchy, cache performance issues
- Appreciation of architectures of multi/many core processors, superscalar and vector processors, GPGPU architectures.

3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite: Course in Computer Architecture, Operating Systems**

3. **Detailed Description of the Course**

	Topic	Sub topic	Duration
1	Introduction & Instruction Set Principles	Fundamentals of Computer Design, Trends in Computer Design, performance, Quantitative Principles of Computer Design, Amdahl's Law, Processor Performance Equations, Instruction Set Architecture, ISA – Encoding, RISC vs CISC approach, The role of compilers	8
2	Pipelining: Basics & concepts	Pipelining – Minimal RISC processor, pipeline design and implementation, Pipelining Hazards, Exceptions, Interrupt, Multi cycle operations, Score Board Technique	10
3	Memory Hierarchy Design	Cache Organization, Cache Optimization Issues, basic, & advanced	3
4	Instruction Level Parallelism	Instruction Level Parallelism: Issues, dependences and loop unrolling, predictors, Introduction to Tomasulo's algorithm, H/W speculation, Value prediction, Introduction to VLIW and Superscalar processors,	8

6	Thread Level Parallelism	Multiprocessors, Architecture, Thread Level Parallelism, Shared memory processors, Distributed memory processors and Architecture, Performance, Synchronization issues, Cache coherence, snooping protocol, Directory based cache-coherence protocol	6
7	Data Level Parallelism, GPGPU Architecture	Vector Architecture, Example program execution, SIMD Instruction set extensions to multimedia applications, Introduction to GPU architecture	4

4. Text books and/or other reference material

1. Computer Architecture : A Quantitative Approach, Hennessy and Patterson, Morgan Kaufmann
2. Advanced Computer Architecture and Parallel Processing, Kai Hwang , McGraw Hill
3. Advanced Computer Architectures : A design space approach, Sima D, Fountain T. and Kacsuk P, Pearson Education

5. Course Objective

Expose the ideas and techniques that define the art of computer architecture, organization, and design. Provide the students with architectural framework and foundation needed to understand future trends in the design.

6. Course Outcome

1. Make the students aware about various trends in computer design, architecture of advanced processors.
2. Realization about issues related to instruction level, thread level, data level parallelism in multi/many core systems, memory organization & optimization techniques.
3. Understand the design issues with shared/distributed memory systems, multi / many core / GPGPU architecture

7. Class Schedule

Lectures : Three 1-hr lectures per week

8. Contribution of Course to Professional Component

Get a fare idea about various types of modern computer architectures from single processors to multi, many core processors, their design issues, implementation details. Get aware about programming tools and techniques to exploit architectural features offered by modern computer architectures.

9. Evaluation of Students

2 sessional exams, end-semester exam, assignments, seminars on current trends in architecture

Course Code and Title

CSP541 : Software Lab - I

1. **Course Description : Core Course**

   Understanding Linux operating system. Getting insight into object oriented programming, advanced data structures and web programming.

   1 practical slot (2 hrs), per week.

   Credit scheme - (L-T-P-C: 1-0-2-2)

2. **Required Background or Pre-requisite: None**

3. **Detailed Description of the Course**

   This lab course prepares students for understanding Linux operating system environment. It also teaches them advanced data structures and object oriented programming in Java and C++ languages. Exercises also include programming for web using languages like HTML, XML, and Python. Network monitoring tools are also introduced.

   Typical Laboratory Experiments:

   1. Unix/Linux Lab – 1

      a. Common Commands – ls, passwd, wc, chdir, mkdir, chmod, cd, mv, df, du, netstat, ps, more, set, env, setenv, chgrp, man, rm, rmdir, grep, vi, tar, untar, uuencode, find, cat, history, ping, ifconfig, traceroute, cksum, cmp, ln, lynx, gzip, gunzip

      b. Piping and redirection

      c. Editing, Scripting and Pattern Matching – vi, emacs, awk, sed, bash script – variables, conditionals, and loops

      d. Parameter passing to C program from shell (argc / argv)

      e. Introduction to using different tools for identification of possible errors in C program – gdb, concepts of “core dump”, backtracing using “bt”, using “info” to dump all registers, creating watch-list / watch variables.

      f. DDD (Data Display Debugger) – introduction and usage

2. Web Technologies and Networking Lab

    a. Creating your own homepage

    b. HTML, XML, XSD and HTML / XML parsing

    c. J2EE/.Net introduction: Using Eclipse and VisualStudio to create webpages

    d. JavaScript and JavaScript debugging

    e. PHP/Perl/Python/Ruby scripting

    f. Networking Commands – inetd, host, ifconfig, netstat, nslookup, ping, ssh, traceroute

    g. Network Monitoring tools – Nagios, Wireshark, OpenNMS

3. Advanced Programming and Data Structures Lab – I

    a. Arrays: Searching, Sorting, 2D/3D arrays

    b. Functions: Pass-by-Value, Pass-by-Reference, Recurrence Functions

    c. Generating permutation/combinations, Generating truth-table for a logical formula

4. Object Oriented Lab

    a. OO Concepts – Classes, Objects, Inheritance, Overloading

    b. Exceptions and Error Handing

    c. Threading and Synchronization in Java/C++

    d. UML and Design Patterns

    e. Profilers –static and dynamic profiling, Code coverage tools, memory leak tools and usage

4. Text books and/or other required material

1. Head First Java, 2nd Edition by Kathy Sierra, Bert Bates, Publisher: O'Reilly Media
2. Linux in a Nutshell, 6th Edition By Ellen Siever, Stephen Figgins, Robert Love, Arnold Robbins Publisher: O'Reilly Media

5. Course Objectives

Given the knowledge of basic programming practices, a first semester M. Tech. CSE student will be able to design and solve basic programming problems and work in Linux environment.

6. Course Outcomes

- The students should in-depth understanding of the Linux operating system and reveals critical commands and options of using the Linux for any application. To understand application development, debugging and code management under Linux platform. Learn how to develop for and port applications to the Linux environment. Get to know the necessary tools for Linux application development and learn about special features offered by Linux.
- To inform students about various elements that make the web work, and the way they relate to PHP, Python, HTML, XML and JavaScript. Show some of the ways that can be used to inspect and change a web form and a web-page.
- To substantially strengthen students' programming ability by requiring them to program a number of large, interesting problems.
- To improve students understanding of object-oriented principles.
- To provide exposure to a broad range of programming areas including multi-threaded programs, communication between processes, and interacting with databases.

7. Laboratory Schedule

Lab: One 2-hrs session per week

8. Contribution of Course to Professional Component

Lab: Students learn to program and use Linux operating system. They understand object oriented programming and web based programming. They also learn about advanced data structures.

9. Evaluation of Students

The instructor uses the following methods: several lab projects, and one-on-one discussions during office hours.

Course Code and Title
CSP529 : Technical Writing and Publishing

1. **Course Description : Core Course**

   Understanding and using English language for communication, documentation and presentation.
   1 practical slot (2 hrs), per week.
   Credit scheme - (L-T-P-C: 1-0-2-2)

2. **Required Background or Prerequisite: None**

3. **Detailed Description of the Course**

   Understanding and using English language for communication, documentation and presentation. Using Document management tools. Gaining expertise on Microsoft Office tools like Word, Excel and PowerPoint. Using Linux and Latex for creating documents for sharing. Summarizing, Abstracting and Elucidating technical documentation. Understanding paper reading, paper reviewing and paper publishing.

   Typical Laboratory Experiments:

   1. Presenting a book chapter using powerpoint slides
   2. Data Analysis: Maintaining multiple results obtained over time and reporting them using charts and graphs
   3. Technical Documentation – Requirement/specification documentation, Design documentation, Test-cases documentation, Use-cases documentation
   4. Writing an installation/instruction manual
   5. Writing an abstract of a technical article – summarizing an article in 300 words
   6. Summarizing 3 papers into a report and its presentation

4. **Text books and/or other required material:**

   1. Strunk and White : The Elements of Style
   2. Gretchen Hargis et. al. : Developing Quality Technical Information: A Handbook for Writers and Editors, Second Edition, IBM, 2004.

3. Leslie Lamport : LaTex

5. Course Objectives

Given the knowledge of technical communication skills, a first semester M. Tech. CSE student will be able to communicate effectively using verbal and written communication skills.

6. Course Outcomes

1. To inform students about writing by engineers and computer scientists, to engineers, engineering managers, and technical writers.
2. Assignments and projects include job application and resume, in-code documentation, algorithm description, survey article, proposal, progress report, formal technical report, summarization and oral presentation.
3. Students would be able to effectively use Google search, understand and use manuals, create installation/instruction manuals, understand the standards and RFC's, and Present a book chapter using power point slides

7. Laboratory Schedule

Lab: One 2-hr session per week

8. Contribution of Course to Professional Component

Lab: Students learn to communicate effectively using different platform.

9. Evaluation of Students

The instructor uses the following methods: several lab projects, and one-on-one discussions during office hours.

Course Code & Title
CSL517: Pattern Recognition

1. Course Description : Elective Course

The course is introduced to teach concepts of pattern recognition which is used in the applications like document analysis, medical imaging. The concepts like statistical and syntactic pattern recognition, feature selection, feature generation, error analysis, and various methods of designing classifiers are important from industry perspective.

3 lectures per week. Credit scheme – (L-T-P-C: 3-0-2-4)

2. Required Background or Pre-requisite: Probability theory, Linear Algebra

3. Detailed Description of the Course

- Introduction: Pattern Recognition, Challenges, advance concepts**(**01 Week)
- Probability Theory: Basic concepts, Random variable, Discrete distribution, Continuous distribution, Continuous distribution(01 Week)
- Probability generating and moment generating functions, : Functions and operations on random variable, Estimation of distribution parameters(01 Week)
- Confidence interval estimation, Hypothesis testing and type I and II errors(01 Week)
- Goodness of Fit test, Joint distribution and correlation analysis(01 Week)
- Classifiers: Building classifier for single feature using Bay's rule and error analysis, decision boundaries for various cases **(**01 Week)
- Building classifiers using KNN, Parzen windows**(**01 Week)
- Building linear classifiers(01 Week)
- Building linear classifiers using SVM(01 Week)
- SVM optimization(01 Week)
- Building classifiers using syntactic methods(01 Week)
- Context dependant classification(01 Week)
- Feature generation(01 Week)
- Unsupervised learning(01 Week)

4. Text books and/or other required material

- Probability and Statistics with Reliability, Queuing, and Computer Science Applications, Kishore Trivedi, John Wiley and Sons, New York, 2001.
- Pattern Recognition, 4th Edition from Sergios Theodoridis, Konstantinos Koutroumbas. Elsevier ,ISBN-9781597492720, Printbook , Release Date: 2008.
- Pattern Classification, 2nd Edition, Richard O. Duda, Peter E. Hart, David G. Stork. Wiley, ISBN: 978-0-471-05669

5. Course Objective:

Upon successful completion of this course, each student should be able to

- Understand how to generate pattern features using various transforms based on data.
- Understand how to analyze pattern features using probability theory.
- Understand how to build classifiers using known probability distribution.
- Understand how to build classifiers using non parametric methods.
- Understand how to build linear classifiers using perception model.
- Understand how to build linear, nonlinear classifiers using SVM model.
- Understand how to build classifiers using syntactic model.
- Understand theory of unsupervised learning.

6. Lab Experiments

- Generating features for two classes and analyzing them.
- Building classifiers for two classes using Bay's rule.
- Building classifiers for multiple classes using Bay's rule.
- Building classifiers for two classes using linear classifier.
- Building classifiers for two classes using SVM
- Implementation of clustering of patterns

7. Class and Lab Schedule

Lecture: Three 1-hour lectures per week

Lab: One per week(120min)

8. Contribution of Course to Professional Component

Lecture: Student learns about theory of classification, building classifiers, analyzing accuracy of classifiers.

Lab: Student learns about implementation of various classifiers for various types of applications.

9. Evaluation of students:

The instructor evaluates outcomes using the following methods:

- Assignments
- Midterm exams
- Quizzes
- Laboratory assignments

The student grades are decided based on the following factors:

- Assignment
- Midterm exam
- Final exam
- Lab viva

Course Code & Title
CSL 509 - Cloud Architecture, Infrastructure and Technology

1. **Course Description : Elective Course**

   To study cloud architecture, cloud types, and cloud infrastructure. To explore different aspects of cloud technology like authentication, sharing, virtualization and data storage management.

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-2-4)

2. **Required Background or Pre-requisite :**

3. **Detailed Description of the course :**

   - Cloud system architectures, Cloud programming frameworks and what is "infrastructure-as-a-service", “platform-as-a-service” and “software-as-a-service” , Cloud computing delivery models – public, private and hybrid clouds, cloud-in-a-box
   - "Big data" concepts, storage and management, Security, scalability, privacy, lock-in, and other risks (and mitigations) for individuals and companies
   - Virtualization, clustering and resource management, HPC in cloud computing
   - Cloud applications considerations for updates, backups, disaster recovery and fault tolerance, Data center networks and Energy use in data centers
   - Introduction to cloud enabling technologies: Introduction to Hadoop, Map-reduce, NoSQL, MongoDB, Cassandra, Web Servers, Encryption techniques, SSL
   - Case Study: Design of a cloud system

4. **Text Book :**

   Resse G., Cloud Application Architectures: Building Applications and Infrastructure in the Cloud,O’ Reilly.

   Reference Book :

   Buyya R., Broberg J., Goscinski A. M., Cloud Computing – Principles and Paradigms,Wiley.

5. **Course Objectives :**

   Given the knowledge of languages with tools to process them, a first semester M.Tech. CSE student will be able to understand current cloud computing technologies, including technologies for

Infrastructure as a Service, Platform as a Service, and Software as a Service.

6. Course Outcomes

- Understanding of different layers of cloud computing, infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). Introduction of practical IaaS, PasS, SaaS, including Amazon ECS, GAE, Force.com, Microsoft Azure, etc.
- Gain knowledge of cloud storage system design issues, including directory management, data placement, and consistency issues. Practical cloud storage system solutions including GFS, Big Table, HDFS, etc.
- Student should have good knowledge of authentication, authorization and secure access in the cloud. Introduction to cloud security. Secure computation in the cloud.
- Insights into the virtualization technologies: Hypervisor, emulation, and application VM. Platform virtualization, storage virtualization, and network virtualization.

7. Class/Laboratory Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

Lecture: Students learn about ways of delivery of enterprise applications and software as a service (SaaS) over different cloud platforms.

9. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, end-semester exam and one-on-one discussions during office hours.

Course Code & Title
ECL 412 - Advanced Digital Signal Processing

1. Course Description : Elective Course

To study the advances in digital signal processing mechanism. To study various data compression mechanism such as DCT, DWT, LZW, etc. To study the DSP processors used for digital signal processing applications.

3 lectures per week. Credit scheme - (L-T-P-C: 3-0-2-4)

2. Required Background or Pre-requisite

3. Detailed Description of the course

Introduction to Speech processing, Speech production model, Linear predictive coding for speech, Yuleualker equations, Short Time Fourier Transform (STFT) , analysis of speech signals using STFT.Multi rate signal processing, decimator, interpolator, poly-phase decomposition, Noble identities, application to Discrete multi-carrier transmission, sigma-delta ADC Data compression, lossy and lossless compression, LZW compression, Arithmetic coding, Discrete Cosine Transform (DCT) and its application to still image compression, audio compression Introduction to Wavelet transform: Properties of wavelet transform,
DWT, filter implementation of DWT, applications of DWT for image denoising and Scaling functions as signaling pulses in communication Introduction to commercial DSP processors & DSP architecture

4. Text / References

1.Introduction to data compression,Khalid Sayood,Elsevier,Second
2.Digital signal processing a computer based approach,S.K.Mitra,TMH,Third
3.Digital signal processing & applications,Dagstranneby and William walker,Elsevier,Second
4.Wavelet Transforms: Introduction to Theory and Applications,A.M.Rao & A.S.Bopardikar, Pearson Edition

5. Course Objectives

To study the advances in digital signal processing mechanism. To study various data compression mechanism such as DCT, DWT, LZW, etc. To study the DSP processors used for digital signal processing applications.

Course Code and Title
CSL519 : Distributed Systems

1. **Course Description : Core Course**

   The study of basic techniques in the design and development of Distributed Systems and understanding solutions of the fundamental problems in distributed systems like mutual exclusion, deadlock detection, termination detection, leader election, fault tolerance, etc.
   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-2-4)

2. **Required Background or Pre-requisite: Operating Systems Course**

3. **Detailed Description of the Course**

   - Introduction and motivation to Distributed Systems,Characteristics, Applications, Challenges, Architecture types, Fundamental models. (2 weeks)
   - Inter-process and inter-node communication using Sockets – connection oriented and connection-less, Remote Procedure Calls, Remote Method Invocation (1 week)
   - Distributed File System Design and Implementation, Case Studies of NFS, Andrew File Systems, HDFS, Distributed Resource Management. (1 week)
   - Clock Synchronization Techniques, Network Time Protocol, Logical Clocks, Vector Clocks. (1 week)
   - Causally Ordered Broadcast and Unicast, Termination Detection – Ring based and Dijkstra Scholten algorithms, Leader Election – Ring based, Franklin's algorithm and Bully Algorithm (2 weeks)
   - Distributed Mutual Exclusion – Token based algorithms – Lamport's, Ricart-Agarwala, Maekawa's algorithms, Non Token based Algorithms – Suzuki Kasami, Raymond's algorithms, comparison of different algorithms. (2 weeks)
   - Distributed Deadlock Detection, Resource and Communication Deadlocks – Centralized technique, Distributed technique - edge chasing and path pushing algorithms, Hierarchical technique, Recovery from Deadlocks. (1 week)
   - Fault Tolerance, Handling Crash faults –Two phase commit protocol, Non-blocking three phase commit protocol, Birman-Joseph Atomic Broadcast Protocol, Voting techniques for fault tolerance. (2 weeks)
   - Recovery – forward and backward recovery, undo-redo logs, Coordinated and Uncoordinated Checkpointing and Recovery algos (1 week)
   - Agreement protocols – LSP Oral Messages, Agreement using Signed Messages (1 week)

   Typical Laboratory Experiments

   Socket Programming, RPC, Using Sockets to implement a rudimentary Distributed File System,

Implementation and Performance Evaluation of any Distributed Mutual Exclucion algorithm.

4. Text/References

- Singhal and Shivratri, “Advanced concepts in Operating Systems”, McGraw Hill
- Coulouris, “Distributed Systems”, AWL Press. Pearson Education
- Tanenbaum, “Modern Operating Systems”, PHI

5. Course Objective

Given the knowledge of operating systems and sequential program design, the students of the second semester M. Tech. CSE will be able to design and develop fault tolerant and efficient distributed algorithms to solve large problems where data and control is distributed over different nodes.

6. Course Outcomes

1. Identify the advantages and challenges in designing distributed algorithms for different primitives like mutual exclusion, deadlock detection, agreement, etc.
2. Design and develop distributed programs using sockets and RPC/RMI.
3. Differentiate between different types of faults and fault handling techniques in order to implement fault tolerant systems.
4. Analyze different algorithms and techniques for the design and development of distributed systems subject to specific design and performance constraints.

7. Class/Laboratory Schedule

Lecture: Three 1-hour lectures per week
Lab: One 2 hr session per week

8. Contribution of Course to Professional Component

Lectures: Students learn about the design and development of distributed systems. They are able to understand the various primitives used in implementing distributed systems and can analyze different algorithms for their implementation.

Lab: Students learn to implement distributed programs using sockets and RPC/RMI. They understand the issues about scalability by doing performance evaluation of their experiments.

9. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, end-semester exam, a course project, one-on-one discussions during office hours, laboratory experiments and programming assignments.

Course Code and Title

CSL520 ADVANCED DBMS

1. **Course Description : Core Course**

   To teach database internals and implementation .

   To teach concepts in distributed databases, parallel databases and columnar databases.

   Credit scheme - (L-T-P-C: 3-2-0-4)

2. **Required Background or Pre-requisite:**
   Data structure and algorithm, Operating Systems, Database Management Systems

3. **Detailed Description of the Course**
   - Introduction: DBMS (1 week)
   - SQL: Overview (1 week)
   - Anatomy of DBMS, process, memory and disk structure (1 week)
   - ACID properties: Implementation (1 week)
   - Concurrency control and Isolation level (1 week)
   - Logical and Physical storage (1 week)
   - Query optimization (1 week)
   - Security and auditing (1 week)
   - Materialized Views and Query rewriting (1 week)
   - Distributed database model (1 week)
   - ACID properties in distributed database (1 week)
   - Parallel database architecture (1 week)
   - Hadoop Architecture (1 week)
   - Columnar databases (1 week)

4. **Text books and/or other required material**

   - Database System Concepts **Sixth Edition**. Avi Silberschatz · Henry F.**Korth** · S. Sudarshan. McGraw-Hill ISBN 0-07-352332-1
   - Fundamentals of Database Systems **5th Edition**. Textbook authors: Shamkant B. **Navathe**,

Ramez Elmasri Addison-Wesley ISBN: 9780321369574

- **Distributed Databases**. Author, Stefano CERI. Publisher, McGraw-Hill, 1988. ISBN, 0070265119, 9780070265110.
- Principles of Distributed Database Systems 3rd Edition Author **Özsu**, M. Tamer, **Valduriez**, Patrick Springer
- Readings in Database Systems, Third Edition (The Morgan Kaufmann Series in Data Management Systems) [Paperback]
- Michael Stonebraker (Author), Joseph Hellerstein (Author)
- Oracle 11g Concepts guide
- Oracle 11g Administration Guide
- Oracle 11g Performance and Tuning

5. Lab Experiments

1. Advanced SQL queries
2. Query optimization
3. Establishing distributed environment
4. Establishing parallel environment
5. Solving problems using Hadoop
6. Columnar database installation and experimentation.

6. Course Objectives

Upon successful completion of this course, each student should be able to

- Understand how to optimize database management system.
- Understand how to handle large concurrent operations.
- Understand how to ensure durability of data.
- Understand how to handle distributed database.
- Understand how to design systems for parallel databases.
- Understand new concepts like BIGDATA and columnar databases.

7. Class Schedule

Lecture: Three 1-hour lectures per week
Lab: One per week(120min each)

8. Contribution of Course to Professional Component

Lecture: Student learn about database internals, database optimization, distributed and parallel environment and new technologies like Hadoop and columnar databases.

Lab: Student learns about writing advanced queries, query optimization, establishing different environments like parallel and distributed databases.

9. Evaluation of Students

The instructor evaluates outcomes using the following methods:

- Assignments
- Midterm exams
- Quizzes
- Laboratory assignments

The student grades are decided based on the following factors:

- Assignment
- Midterm exam
- Final exam
- Lab viva

Course Code and Title

CSL 528 Cryptography and Information Security

1. Course Description : Core Course

This course covers the major aspects of cryptography and information security. Various topics such as classical encryption techniques, data encryption standard, advanced encryption standard and public key cryptography algorithms such as RSA, ElGamal etc. are studied. Key management, message authentication code, hash algorithm and digital signatures are also studied. Security applications are also studied.

3 lectures per week. Credit scheme – (L-T-P-C: 3-0-2-4)

2. Required Background and Pre-requisites

3. Detailed Description of the Course

- Introduction and Classical Ciphers (1 week)
- Block Ciphers and DES (2 weeks)
- Algebraic Structures: Groups, Rings, Finite and Galois Fields (2 weeks)
- AES, 2DES and 3DES (1 weeks)
- Block Modes of Operation: ECB, CBC (1 week)
- Basic Number Theory: Primes, Congruences, CRT, Modular Exponentiation (2 weeks)
- Asymmetric Key Cryptosystems: RSA, Elgamal (1 week)
- Hash and MAC: SHA-512 (1 weeks)
- Digital Signatures: RSA, Elgamal. Schnorr, DSS (2 weeks)
- Key Management: Kerberos, Diffie-Hellman, Digital Certificates (1 week)
- Email Security: PGP, Viruses, Worms, and other Malware , Firewalls (1 week)

4. Text books and other required material

- William Stallings, Cryptography and Network Security, PHI

- Behrouz A. Forouzan, Cryptography and Network Security, McGraw Hill

5. **Courses Objectives**
   - Study of symmetric key cryptography and related mathematical background.
   - Study of public key cryptography and related mathematical background.
   - Know the applications of information security

6. **Course Outcomes**
   1. The student gains knowledge of mathematical background of symmetric key cryptography.
   2. The student gains knowledge of mathematical background of public key cryptography.
   3. The student gains knowledge about digital signatures.
   4. The student gains knowledge about applications of information security.

7. **Class Schedule**

   Lectures: Three 1-hr lectures per week

8. **Contribution of Course to Professional Component**

   Student learns the important aspects of cryptography and information security which is very important in his professional career as a user or as an advisor.

9. **Evaluation of students**

   The students are evaluated through 2 sessional exams, one assignment and an end-demester examination.

Course Code and Title
CSP542 : Software Lab - II

1. **Course Description : Core Course**

   Understanding Advanced Linux operating system. Getting insight into open source tools.

   1 practical slot (2 hrs), per week.

   Credit scheme - (L-T-P-C: 1-0-2-2)

2. **Required Background or Pre-requisite: S/W Lab-I**

3. **Detailed Description of the Course**

   This lab course prepares students for understanding advanced Linux operating system environment. It also teaches them advanced data structures and open source technologies like web servers, database servers, etc. Network monitoring tools are also introduced.

   Typical Laboratory Experiments:

   1. Unix/Linux Lab – II
      a. Makefile – writing Makefile, compilation via make, compilation of C programs distributed in multiple files
      b. Versioning system like CVS – versioning and branching
      c. Linux System Administration
         - Installing Linux in a virtual machine
         - Mounting/Unmounting Disks
         - Setting and using Path and Environment Variables
         - Starting telnet, ftp, smtp services and using them
         - rcp, rsh, rlogin
         - Super user commands/priviledges – su, sudo, install/uninstall of packages, updating linux system
      d. OS related exercises – Accessing iNode, Creation of threads, fork / join, creation of semaphore / mutex, assignments on synchronizing threads. Pthreads and Java threading APIs.

   2. Advanced Programming and Data Structures Lab – II
      a. Pointers: Linked List, Queue, Stack, Sparse Matrix

b. Trees: Binary, B-Trees, B+ Trees
c. Graphs: Matrix representation, Shortest Path, Connectivity, Max-cut/Min-Flow, Matchings

3. Open Sources Lab
   a. Using automatic testing tools – Junit, NUnit
   b. Installing and using Apache Web server to develop a website, Installing and using mysql to store some data
   c. Use of open source cloud platforms:
      - Integration of gmail with google calendar – from gmail you should be able to schedule an appointment with all the recipients of the mail.
      - Creating a website on Salesforce cloud for tracking inventory from east, west, north, south regions in India separately.
      - Accessing google-map via google-map APIs
   d. Java Native Interface (JNI) – Calling C / C++ code from Java and vice versa
   e. Downloading and Installing Hadoop on 3 to 4 machines and writing a distributed sorting program on the same.
   f. Creating web-services using Axis-2 (Java) or gSoap library (C / C++)
   g. Introduction to SSL. Use digital certificates to encrypt / decrypt data in transfers Notes - Keytool in Java allows to create / store / manipulate certificates. Also, refer www.thawte.com for free download/creation of a certificate
   h. Introduction to Android Platform and APIs / libraries provided. A sample game / application on Android.
   i. Use of MySQL server:
      - Installing and using mysql servers in load sharing manner (Configuration of 2 MySQL instances in master-slave mode).
      - Database operations via programs written in C/C++ or Java.

4. Text books and/or other required material:

1. The cathedral and the bazaar, Eric Raymond, http://www.catb.org/~esr/writings/cathedral-bazaar/
2. Linux in a Nutshell, 6th Edition By Ellen Siever, Stephen Figgins, Robert Love, Arnold Robbins Publisher: O'Reilly Media
3. http://en.wikibooks.org/wiki/Open_Source

5. Course Objectives

Given the knowledge of advanced linux and open source tools, a second semester M. Tech. CSE student will be able to design and solve advanced programming problems and work in web-based environment.

6. Course Outcomes

- The students should get understanding of the advanced Linux operating system. Get to know the necessary tools for Linux administration and script development and learn about special features offered by Linux.
- To inform students about various elements that make the advanced web work, and the way they relate to open-source platforms.
- To substantially strengthen students' programming ability by requiring them to program a number of complex programming problems.
- To provide exposure to a broad range of open-source technologies including encryption and working with databases.

7. Laboratory Schedule

Lab: One 2-hrs session per week

8. Contribution of Course to Professional Component

Lab: Students learn to program and use Linux operating system. They understand object oriented programming and web based programming. They also learn about advanced data structures. They learn about open source tools.

9. Evaluation of Students

The instructor uses the following methods: several lab projects, and one-on-one discussions during office hours.

Course Code and Title

CSL516 : Soft Computing Techniques

1. **Course Description : Elective Course**

   This course provides an introduction to the basic concepts of Soft Computing methodology and covers three main components - Neural Networks, Fuzzy Logic and Evolutionary Computation. The course combines theoretical foundations with practical applications using different tools and techniques.

   Topics include Neural Networks, Fuzzy Logic, Evolutionary Computation and Recent developments and applications of Soft Computing in various areas.

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite:** Data Structure

3. **Detailed Description of the Course**

Topics	**Duration**
Overview of course and Basic of Soft Computing	1 week
Introduction of Neural Networks	1week
Learning Process and Learning Task	1week
Supervised Learning – Single and Multi Layer Network	2 weeks
Associative Memory	1 week
Self organizing Maps	1 week
Neuro-Dynamics, Hopfield Network	1 week
Fuzzy Logic and Systems-Fuzzy Sets and Membership Functions, Operations on Fuzzy Sets, Fuzzification.	1 week
Fuzzy Numbers- Uncertain Fuzzy Values, Fuzzy Numbers and its L-R representation, Operations on Fuzzy Numbers.	1 week
Fuzzy Relations	1 week

Fuzzy Inference Systems- Architecture of Fuzzy Inference System, Fuzzy Inference Rules and Reasoning, Defuzzification.	1 week
Applications of Fuzzy Logic	1 week
Genetic algorithms and evolutionary computation.	1 week
Applications of Genetic Algorithms & Hybrid Systems	1 week

4. Text books and/or other required material

- Soft Computing and Its Applications : R.A. Aliev, R.R. Aliev
- Neuro-Fuzzy and Soft Computing: A computational Approach to Learning & Machine Intelligence; Roger Jang, Tsai Sun, Eiji Mizutani, PHI.
- Neural Network: A Comprehensive Foundation; Simon Haykin, PHI.
- Elements of artificial Neural Networks; Kishan Mehtrotra, S. Ranka, Penram International Publishing (India).
- Fuzzy Logic with Engineering Applications; Timothy Ross, McGraw-Hill.
- Neural Networks and Fuzzy Systems: Bar Kosko , PHI.

5. Course Objectives

Introduce students to soft computing concepts and techniques and foster their abilities in designing and implementing soft computing based solutions for real-world problems.

6. Course Outcomes

1. Appreciation of the unified and exact mathematical basis as well as the general principles of various soft computing techniques.
2. Appreciation of basic knowledge about the theory and key algorithms that form the foundation for artificial neural network and practical knowledge of learning algorithms and methods
3. Appreciation of basic knowledge about the theory and key algorithms that form the foundation for fuzzy logic and become aware of the use of fuzzy inference systems in the design of intelligent or humanistic systems.
4. Ability to understand the principles, advantages, limitations and possible applications of learning.
5. Ability to identify and apply the appropriate learning technique to classification, pattern recognition, optimization and decision problems.

7. Class Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

- Understand the need for Soft Computing;
- Understand different uses of Soft Computing in various areas;
- Understand the steps involved in the development of Soft Computing;
- Acquire a working knowledge of some popular tools for Soft Computing;
- Design, implement and verify computing systems by using appropriate Soft Computing techniques and tools

9. Evaluation of Students

The instructor uses the following methods: Two sessional exams, one end-semester examination and programming assignments.

Course Code and Title

CSL521 : Software Architecture (SA)

1. **Course Description : Elective Course**

   The course gives the students understanding of the concept of software architecture and how this phase in the development between requirement specification and detailed design is vital for the success of a software system. The students will get knowledge of some well-known architecture patterns, and be able to design, construct and evaluate architectures for software systems. In addition, the students should get some understanding of how the developers experiences and the technical and organizational environment will influence on the choice of architecture.
   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite: CSL308 Software Engineering or equivalent**

3. **Detailed Description of the Course**

   - Software process and the role of modeling and analysis, software architecture and software design. (1 week)
   - Software Modeling and Analysis: Analysis modeling and best practices, traditional best practice diagrams such as DFDs and ERDs (2 weeks)
   - Software Architecture: architectural styles, architectural patterns, analysis of architectures, formal descriptions of software architectures , Architectural description languages and tools (3 weeks)
   - Software Design: design best practices, design patterns, design case studies, component technology, object oriented frameworks, distributed objects, interoperability standards, case studies., software quality (3 weeks)
   - UML diagrams and UML analysis modeling, analysis case studies, analysis tools, analysis patterns, documenting software architecture, reconstructing software architecture. (2 weeks)
   - Middleware components, programming models, implementation, systems qualities Moving from qualities to architecture and views Components and COTS, Economics- Driven Architecture, Software product line, Software architecture future. (2 weeks)
   - Issues in Software Architecture: Scalability and interoperability issues, web application architectures, case studies. (2 weeks)

4. Text/Reference material

- M. Shaw, "Software Architecture Perspectives on an Emerging Discipline", PHI
- Len Bass, Paul Clements, Rick Kazman, "Software Architecture in Practice", Pearson Education Asia
- R. Taylor, N. Medvidovic, E. Dashofy, "Software Architecture – Foundations, Theory, and Practice", Wiley India
- Jan Bosch, "Design and Use of Software Architectures", Addision-Wesley-Pearson Education
- Christine Hofmeister, Robert Nord, Dilip Soni, "Aoolied Software Architecture", Addision-Wesley Pearson Education
- Dikel, D.Met Al, "Software Architecture: Organizational Principles and Pattern", Prentice Hall

5. Course Objective

The students should know how to develop different architectural views of an architecture, addressing specific concerns of stakeholders. It involves making a large number of trade-offs between concerns of different stakeholders. There may be different acceptable solutions, and the solution eventually chosen depends on how the balancing between stakeholder concerns is made. The students should know how to do an assessment of an architecture.

6. Course Outcomes:

On completion of this course students will be able to:

1. Design and understand software architecture for large scale software systems.
2. Recognize major software architectural styles, design patterns, and frameworks
3. Describe a software architecture using various documentation approaches and architectural description languages
4. Develop architectural alternatives for a problem and select among them
5. Use well-understood paradigms for designing new systems

7. Class Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

The students get the understanding of software reuse, reuse of corporate and domain knowledge through case studies. Rapid system implementation and many modeling concepts contribute to professional. The practice of architecture-centric development makes the students to think on different paradigms of software quality and usage.

9. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, end-semester exam and case-study based assignments.

Course Code and Title

CSL511 : Advances in Data Mining

1. **Course Description : Elective Course**

   Advanced Concepts of Data mining and its application to real life examples. Various data mining algorithms .

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite:** CSL403 : Database Management Systems

3. **Detailed Description of the Course**

   - Association Rules : A priori Algorithms, Fp-trees Algorithms,Constraints and solutions, Constrained association rules. Maximal frequent items mining using GENMAX and MAFIA. (2 week)
   - Sequential data mining rules and algorithms for extracting sequential association rules. Graph Mining (2 week)
   - Cluster Analysis – Paradigms , DBSCAN , Cluster algorithms, Agglomerative Hierarchical Clustering, Graph-Based Clustering , Density-Based Outlier Detection, Density-Based Outlier Detection (2 week)
   - Anomaly Detection, Statistical Approaches, Proximity-Based Outlier Detection(2 week)
   - Decision Trees and applications , Data discreatetization (2 week)
   - Data mining algorithms for streams.Concept drift detection and model building(2 week)

4. **Text books and/or other required material**

   - Data mining - Concepts & Techniques, Jiawei Han, Micheline Kamber, Morgan Kaufmann ,2nd Ed.2006.
   - Introduction to Data Mining : Vipin Kumar , Tan, Steinbach , Pearson 2014
   - Data Mining : Witten, Frank , Hall Morgan Kaiffman 2014

5. Course Objectives

- Identify the scope and necessity of Data Mininn
- Designing of Data Mining algorithms so that students can able to solve the root problems.
- To understand various tools of Data Mining and their techniques to solve the real time problems.
- To develop further interest in research and design of new Data Mining Techniques.

6. Class Schedule

Lectures : Three 1-hr lectures per week

7. Contribution of Course to Professional Component

Lecture: Students should know about design issues of various mining algorithms . To identify the real time problems and able to design solution using various mining tools, further take the R&D interest and try to contribute some new methods to the area.

8. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, end-semester exam, class test, some real time problems as programming assignments.

Course Code and Title
CSL524 : Real Time Systems

1. **Course Description : Elective Course**

   The study of scheduling algorithms and schedulability tests for different algorithms in the design of Hard Real-Time Systems (RTS). Understanding execution of periodic, sporadic, and aperiodic jobs. Taking into account the issues of resource allocation and overview of features of different commercially available Real Time Operating Systems (RTOS).

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite:** Operating Systems Course

3. **Detailed Description of the Course**

   - Introduction to RTS, WCET notion, Types of RTS, Task Types, Jobs – Periodic, Sporadic, Aperiodic, Applications of RTS, Predictability, Reference Model, Types of schedulers, Cyclic and Priority based Schedulers. (1 week)

   - Cyclic, priority based schedulers – static/dynamic – RM, EDF, LST, Optimality of EDF, Non-optimality of EDF, Scheduling with precedence constraints, Multiprocessor scheduling – static and dynamic systems, Problems of Predictability in multi-processor systems, Predictability of preemptive priority based scheduling in uniprocessor systems, Performance Measure of validation techniques. (2 weeks)

   - Cyclic scheduling, frame size constraints, Job Slicing, Aperiodic job scheduling using Slack stealing, Sporadic job scheduling, Practical considerations, Disadv of cyclic scheduling. (2 weeks)

   - Priority Based Sched, Static-Dynamic Systems, Fixed, Variable Priorities, Schedulable Utilization, Schedulable Utilization of EDF, Schedulability Test of EDF, Unpredictability of Dynamic Priority in Overload, Liu-Layland Theorem, Optimality of RM in Simply-Periodic Systems, Concept of Critical Instants, Time Demand Analysis, Practical factors - Non-preemption, self-suspension, context switch time, Limited priority levels, Mapping techniques, Impact on schedulability, Tick Scheduling. (3 weeks)

   - Aperiodic jobs in Priority based systems, Polling Server, Combining with background server,Polling Server Parameters, Deferrable Server (DS), Combining with Background Server, Deferrable Server parameters, Disadv of DS, Simple Sporadic Server Rules, Combining Background time, Proof of Simple Sporadic Server as a periodic task, Constant Utilization Server for Deadline Driven systems, Total Bandwidth Server, Starvation free CU/Background Server,

Preemptive Weighted Fair Queuing Server, Scheduling of Sporadic Jobs in Fixed Priority and Dynamic Priority Systems. (3 weeks)

- Resource Control, Model, Priority Inversion, Uncontrolled Priority Inversion, Anomalies, NPCS, Blocking Time, Disadvantages of NPCS, Priority Inheritance Protocol, Deadlocks due to Priority Inheritance Protocol, Priority Ceiling Protocol, Deadlock Avoidance,Analysis of Priority Ceiling Protocol, Blocking time, context switches, Stack Sharing Priority Ceiling Protocol, example, Priority Ceiling Protocol in Dynamic Priority Systems, Preemption Levels, Fixed Preemption Level Systems like EDF, Basic Preemption Ceiling Protocol, Multiple units of resources, Priority ceiling, Preemption ceiling and stack based preemption ceiling protocols for multiple unit resources. (3 weeks)

4. Text / Reference

- Real-Time Systems :Jane W.S. Liu, Pearson Education
- Real Time Systems : C.M. Krishna & Kang G. Shin : McGraw Hill

5. Course Objective

Given the knowledge of operating systems, a third semester M. Tech. CSE student, will be able to design and implement systems that can be verified to meet the hard timing requirements.

6. Course Outcomes

1. Enumerate the need and the challenges in the design of hard and soft real time systems.
2. Compare different scheduling algorithms and the schedulability criteria.
3. Determine schedulability of a set of periodic tasks given a scheduling algorithm.
4. Develop algorithms to decide the admission criterion of sporadic jobs and the schedule of aperiodic jobs.
5. Integrate resource access mechanisms with the scheduling techniques and develop integrated schedulibility criteria.

7. Class Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

Lecture: Students learn about designing and developing real-time systems. They are able to analyze the different scheduling approaches and are able to make informed design choices.

9. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, a class test, end-semester exam, and programming assignments.

Course Code & Title

CSL539: Formal Methods in program Design

1. **Course Description: Elective Course**

   3 lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite**

3. **Detailed Description of the Course**

   - Foundations of parallel programming, Nondeterminism, Absence of control flow, Synchrony and Asynchrony, States and assignments, extricating proofs from program text, separation of concerns : correctness and complexity, programs and implementation.
   - UNITY program structure, Assignemnet statement, Assign section, Initially-section, Always-section, Proving assertions about Assignment statement, Quantified assertions, conventions about priorities of logical relations.
   - Fundamental concepts, proofs and theorems about: Unless / Ensures / Leads-to / Fixed-point. Proving bounds on progress.
   - Introduction about Architecture and Mappings. All pairs shortest path problem: solution strategy, formal description, proof of correctness, creating the program. Implementation on sequential architectures, parallel synchronous architectures, asynchronous shared-memory architecture, and distributed architecture. Complexities on each of the architectures.
   - Formal description and programs for saddle-point-of-a-matrix, reachability in directed graphs, prime number generation, comparing two ascending sequences, computing the maximum of a set of numbers, Boolean matrix multiplication.
   - Program structuring, program composition by Union, Union theorem, composing specifications, substitution axiom, hierarchical program structures, superposition and superposition theorem, design specifications.
   - Introduction to communicating processes.

4. **Text / References**

1. Pararllel Program Design, A foundation : K. Mani Chandy, Jaidev Misra, Addison-Wesley Publishing.
2. The Science of Programming : David Gries, Springer.

Reference Books:

1. Logic for Computer Science: Foundations of Automatic Theorem Proving : Jean Gaullier, Harper & Row Computer Science Technology Series.

5. Course Objectives :

To study basic concepts used in program design like determinism / non-determinism, synchrony/asynchrony, separation of concerns like correctness and complexity, programs and implementation. To study the concept of progress and proofs thereof. To study architecture and mappings by taking different case studies. To study program structuring and program composition.

Course Code and Title
CSL436 : Information Retrieval

1. **Course Description : Elective Course**

   The study of basic Information Retrieval techniques, data structures and algorithms in Information retrieval
   3 lectures (1 hour each) per week
   Credit scheme - (L-T-P-C: 3-0-0-6)

2. **Required Background or Pre-requisite**: CSL 213: Data Structures and Program Design I and Data Structures and Program Design II

3. **Detailed Description of the Course**

   Boolean retrieval (1 week)
   the term vocabulary and postings lists, (1 week)
   Dictionaries and tolerant retrieval, (1 week)
   Introduction to index-construction and index-compression (1 week)

   Scoring, term weighting and the vector space model (2 weeks)
   Computing scores in a complete search system, Evaluation in information retrieval, Introduction to Relevance feedback and query expansion. (2 weeks)

   Probabilistic information retrieval, review of basic probability theory, the probability ranking principle, the binary independence model (2 weeks)

   Language models for information retrieval, Language modeling versus other approaches to IR, Text classification and Naive Bayes, Bayesian Network approaches to IR. (2 weeks)

   Vector space classification, Support vector machines and machine learning on documents, Fl at clustering, Hierarchical clustering, Matrix decomposition and latent semantic indexing. (2 weeks)

   Introduction to Web search basics, Web crawling and indexes, Link analysis (1 weeks)

   Typical Assignments : Based on techniques studied, implementation of those techniques, study of research papers.

4. **Text books and/or other required material**

   Text Books

   1. An Introduction to Information Retrieval: Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, Cambridge University Press, Cambridge, England, 2009

2. Information Retrieval: Implementing and evaluating search engines: Stefan Büttcher, Charles L. A. Clarke, Gordon V. Cormack, MIT Press, 2010

Reference Books

1. Information Retrieval: Algorithms and Heuristics : David A. Grossman, Ophir Frieder, Springer.
2. Information Retrieval: Data Structures and Algorithms by Frakes, Pearson.

5. Course Objectives

Enable students to understand the various aspects of an Information retrieval system and its evaluation and to be able to design such a system from scratch

6. Course Outcomes

- Understanding the basics of Information retrieval like what is a corpus, what is precision and recall of an IR system
- Understanding the data structures like Inverted Indices used in Information retrieval systems
- Understanding the basics of web search
- Understanding the different techniques for compression of an index including the dictionary and its posting list
- Understanding the different components of an Information retrieval system
- Developing the ability of develop a complete IR system from scratch

7. Class/Laboratory Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

Lecture: Students learn about developing the various aspects of a complete Information Retrieval System

9. Evaluation of Students

The instructor uses the following methods: home-work assignments, 2 sessional exams, end-semester exam and Study of research papers or implementations of learnt techniques in Information retrieval, one-on-one discussions during office hours, internal evaluation test.

Course Code and Title
CSL 530 : Topics in Bioinformatics

1. Course Description : Elective Course

The study of computer applications for Biology. This course also talks about different algorithms that are designed for solving biological problems.
3 lectures (3 hrs) per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. Required Background or Pre-requisite: CSL 313 – Analysis of Algorithms

3. Detailed Description of the Course

1. Basics of biology (1 Week)
2. Sequences: Problem statement, Edit distance and substitution matrices, Global and local alignments, Spliced alignment, Space-efficient sequence alignment, Multiple alignment, (2 Weeks)
3. Structures: Protein alignment, Protein structure prediction (1 Week)
4. Phylogenetic trees: Large parsimony and small parsimony problems, Probabilistic approaches, Grammar-based approaches (1 Weeks)
5. Overview of Gene Control, Working of Genetic Switches, Introductory Systems Biology, The biochemical paradigm, genetic paradigm and the systems paradigm (2 Weeks)
6. Building an Organism Starting From a Single Cell -Quorum Sensing – Programmed Population Control by Cell-Cell Communication and Regulated Killing; Gene regulation at a single cell level-Transcription Networks -basic concepts -coherent Feed Forward Loop (FFL) and delay gate -The incoherent FFL -Temporal order, Signaling networks and neuron circuits -Aspects of multi-stability in gene networks. (3 Weeks)
7. Modeling biological systems, Hidden Markov models (2 Weeks)
8. Miscellaneous topics: Pathways and networks, Microarrays, Biomedical images, Genetic Algorithms and applications, (2 weeks)

4. Text books and/or other required material:

1. "An Introduction to Bioinformatics Algorithms" by Jones, Pevzner. MIT Press.
2. "Algorithms on Strings, Trees and Sequences" by Gusfield. Cambridge University Press.

3. "An Introduction to Systems Biology: Design Principles of Biological Circuits" by Alon. Chapman & Hall/CRC Press.

5. Course Objective

Given the knowledge of string-based and statistical algorithms in Bioinformatics, a second semester M. Tech. CSE student will be able to model biological problems as theoretical computational formulations and reapply the solutions back to the biological world.

6. Course Outcomes

1. This course introduces students to the basic string based computational methods and algorithms that can be used to understand the cell and biological systems.
2. Students will know algorithms and programming techniques like dynamic programming, hashing, and suffix trees.
3. The course focuses on computational approaches to: genetic and physical mapping; genome sequencing, assembly, and annotation.
4. This course will help students develop multidisciplinary approach to the systematic analysis and modeling of complex biological phenomena.
5. Students are also made aware of the currently emerging research areas in the fields of computational and systems biology.

7. Class Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

Lecture: Students learn about computational methods and algorithms in bioinformatics and systems biology. They are also exposed to current emerging research areas in these fields.

9. Evaluation of Students

The instructor uses the following methods: 2 sessional exams, end-semester exam, class test and assignments, one-on-one discussions during office hours.

Course Code and Title
CSL515: Mobile Communication Systems

1. **Course Description : Elective Course**

   Introduction to fundamental computer communication concepts underlying and supporting modern mobile networks. The focus is on wireless communications and media access layers of network protocol stack. Systems include cellular networks (GSM), wireless LANs (IEEE 802.11) and ad hoc wireless and personal area networks (e.g., Bluetooth, ZigBee).
   Three lectures per week. Credit scheme - (L-T-P-C: 3-0-0-3)

2. **Required Background or Pre-requisite**

   CSL 313: Computer Networks

3. **Detailed Description of the course**

Topics	Duration
Radio Propogation, Multiplexing in space, time, frequency and spread spectrum	2 weeks
PCS architecture, cellular concept, cordless telephony, mobility management, handoff	1 week
GSM system overview	1 week
GSM Network signaling	1 week
GSM mobility management	1 week
GSM SMS, international roaming, administration and maintenance, GPRS	2 weeks
3G systems, Wireless LANs	2 weeks
MANETs features, routing, sensor networks	2 weeks
Mobility support at transport and network layers	1 week

4. **Text books and/or other required material**

   - Mobile Communications - J. Schiller -Pearson Education
   - Introduction to Wireless and Mobile Systems- Agrawal and Zeng -Cengage Learning
   - Wireless and Mobile Network Architecture - Lin and Chlamtac- Wiley
   - Wireless Communications: Principles and Practice- Theodore Rappaport - Prentice Hall
   - Wireless and Mobile All-IP Networks - Lin and Pang- Wiley

5. **Course Objective**

   Given the knowledge of computer networks, a third semester M. Tech. CSE student will be able to understand

the cellular radio concepts and describe current and future mobile communication systems.

6. Course Outcomes

- Understand the cellular radio concepts such as frequency reuse, handoff and how interference between mobiles and base stations affects the capacity of cellular systems.
- Identify the techno-political aspects of wireless and mobile communications such as the allocation of the limited wireless spectrum by government regulatory agencies.
- Understand propagation effects such as fading, time delay spread, and Doppler spread, and describe how to measure and model the impact that signal bandwidth and motion have on the instantaneous received signal through the multipath channel.
- Understand the information theoretical aspects (such as the capacity) of wireless channels and basic spread spectrum techniques in mobile wireless systems
- Describe current and future cellular mobile communication systems (GSM, IS95, WCDMA, etc), wireless LANs, adhoc and sensor networks

7. Class Schedule

Lecture: Three 1-hour lectures per week

8. Contribution of Course to Professional Component

Lecture: This is an introductory course in Mobile Communication designed for students to become familiar with the various mobile technologies and their application to computer engineering problems. It provides a vast exposure to engineering professionals for understanding the mobile computing paradigm.

9. Evaluation of students

The instructor uses the following methods: Two sessional exams, one end-semester examination and assignments.