St. PETER'S UNIVERSITY

St. Peter's Institute of Higher Education and Research
(Declared under section 3 of UGC Act 1956)
Avadi, Chennai – 600 054.

M.E. (COMPUTER SCIENCE AND ENGINEERING) PROGRAMME

(I TO IV SEMESTERS)

REGULATIONS AND SYLLABI

(REGULATIONS – 2013)

(Effective from the Academic Year 2013-'14)

M.E. (COMPUTER SCIENCE AND ENGINEERING) PROGRAMME

Regulations and Syllabi

(Effective from the Academic Year 2013-'14)

1. **Eligibility:** Candidates who passed B.E. / B.Tech.(CSE / IT / EEE / ECE /Electronics / E&I / ICE / Instrumentation)/M.C.A. / M.Sc.(Computer Science /Computer Technology / Software Engineering / Information Technology / Electronics) of the University or any other equivalent examination thereto are eligible for admission to Two Year M.E.(Computer Science and Engineering) Programme.

2. **Duration:** Two Years Comprising 4 Semesters. Each semester has a minimum 90 working days with a minimum of 5 hours a day.

3. **Medium:** English is the medium of instruction and examination.

4. **Weightage for Internal and End Assessment:** The weightage for Continuous Assessment (CA) and End Assessment (EA) be 25:75 unless the ratio is specifically mentioned in the scheme of Examinations.

5. **Credit System:** Credit system be followed with 18 credits for each semester and each credit is equivalent to 25-30 hours of effective study provided in the Time Table.

6. **Scheme of Examinations (for I to IV Semesters)**

I Semester

Code No.	Course Title	Credit	Marks		
Theory			**CA**	**EA**	**Total**
113CSPT01	Applied Probability and Statistics	2	25	75	100
113CSPT02	Design and Management of Computer Networks	2	25	75	100
113CSPT03	Advanced Data Structures and Algorithms	2	25	75	100
113CSPT04	Multicore Architectures	3	25	75	100
113CSPT09	**Elective I:** Image Processing and Analysis	3	25	75	100
113CSPT14	**Elective II:** Software Requirements Engineering	3			
Practical					
113CSPP01	Advanced Data Structures Laboratory	2	25	75	100
113CSPP02	Case Study - Network Design (Team Work)	1	25	75	100
	Total	**18**	**175**	**525**	**700**

II Semester

Code No.	Course Title	Credit	Marks		
Theory			**CA**	**EA**	**Total**
213CSPT01	Theoretical Foundations of Computer Science	2	25	75	100
213CSPT02	Advanced Databases	2	25	75	100
213CSPT03	Principles of Programming Languages	2	25	75	100
213CSPT04	Advanced Operating Systems	3	25	75	100
	Elective III:	3	25	75	100
	Elective IV:	3	25	75	100
Practical					
213CSPP01	Advanced Databases Laboratory	2	25	75	100
213CSPP02	Case Study - Operating Systems Design (Team Work)	2	25	75	100
	Total	**18**	**200**	**600**	**800**

III Semester

Code No.	Course Title	Credit	Marks		
Theory			**CA**	**EA**	**Total**
313CSPT01	Software Process and Project Management	3	25	75	100
313CSPT07	**Elective V:** Software Design	3	25	75	100
313CSPT13	**Elective VI:** Software Quality Assurance	3	25	75	100
313CSPT15	**Elective VII:** Enterprise Application Integration	3	25	75	100
Practical					
313CSPP01	Project Work(Phase I)	6	25	75	100
	Total	**18**	**125**	**375**	**500**

IV Semester

Code No.	Course Title	Credit	Marks		
Practical			**CA**	**EA**	**Total**
413CSPP01	Project Work(Phase II)	18	25	75	100
	Total	**18**	25	75	100

LIST OF ELECTIVES

I Semester

Course Code	Course Title	Credit
Elective I		
113CSPT05	Formal Models of Software Systems	3
113CSPT06	Performance Evaluation of Computer Systems	3
113CSPT07	Probabilistic Reasoning Systems	3
113CSPT08	Data Analysis and Business Intelligence	3
113CSPT09	**Image Processing and Analysis**	3
113CSPT10	Sensing Techniques and Sensors	3
Elective II		
113CSPT11	Randomized Algorithms	3
113CSPT12	Mobile and Pervasive Computing	3
113CSPT13	Parallel Programming Paradigms	3
113CSPT14	**Software Requirements Engineering**	3
113CSPT15	Speech Processing and Synthesis	3
113CSPT16	Machine Learning Techniques	3

II Semester

Course Code	Course Title	Credit
Elective III		
213CSPT05	Concurrency Models	3
213CSPT06	Real Time Systems	3
213CSPT07	Computer Vision	3
213CSPT08	Network and Information Security	3
213CSPT09	Design and Analysis of Parallel Algorithms	3
213CSPT10	Software Architectures	3
Elective IV		
213CSPT11	Model Checking and Program Verification	3
213CSPT12	Embedded Software Development	3
213CSPT13	Cloud Computing	3
213CSPT14	Data Visualization Techniques	3
213CSPT15	Protocols and Architecture for Wireless Sensor Networks	3
213CSPT16	Language Technologies	3

III Semester

Course Code	Course Title	Credit
Elective V		
313CSPT02	Social Network Analysis	3
313CSPT03	Managing Big Data	3
313CSPT04	Mobile Application Development	3
313CSPT05	Bio-inspired Computing	3
313CSPT06	Medical Image Processing	3
313CSPT07	Software Design	3
Elective VI		
313CSPT08	Reconfigurable Computing	3
313CSPT09	Energy Aware Computing	3
313CSPT10	Information Retrieval Techniques	3
313CSPT11	Data Mining Techniques	3
313CSPT12	Bio Informatics	3
313CSPT13	Software Quality Assurance	3
Elective VII		
313CSPT14	Multiobjective Optimization Techniques	3
313CSPT15	Enterprise Application Integration	3
313CSPT16	Information Storage Management	3
313CSPT17	Robotics	3
313CSPT18	Compiler Optimization Techniques	3

7. **Passing Requirements:** The minimum pass mark (raw score) be 50% in End Assessment (EA) and 50% in Continuous Assessment (CA) and End Assessment (EA) put together. No minimum mark (raw score) in Continuous Assessment (CA) be prescribed unless it is specifically mentioned in the scheme of Examination.

8. **Grading System:** Grading System on a 10 Point Scale be followed with 1 mark = 0.1 Grade point to successful candidates as given below.

CONVERSION TABLE

(1 mark = 0.1 Grade Point on a 10 Point Scale)

Range of Marks	Grade Point	Letter Grade	Classification
90 to 100	9.0 to 10.0	O	First Class
80 to 89	8.0 to 8.9	A	First Class
70 to 79	7.0 to 7.9	B	First Class
60 to 69	6.0 to 6.9	C	First Class
50 to 59	5.0 to 5.9	D	Second Class
0 to 49	0 to 4.9	**F**	**Reappearance**

Procedure for Calculation

Cumulative Grade Point Average (CGPA)	=	$\frac{\text{Sum of Weighted Grade Points}}{\text{Total Credits}}$
	=	$\frac{\sum (CA+EA)\ C}{\sum C}$
Where Weighted Grade Points in each Course	=	Grade Points (CA+EA) multiplied by Credits
	=	(CA+EA)C
Weighted Cumulative Percentage of Marks(WCPM)	=	CGPAx10

C- Credit, CA-Continuous Assessment, EA- End Assessment

9. **Pattern of the Question Paper:** The question paper for End Assessment will be set for three hours and for the maximum of 100 marks with following divisions and details.

Part A: 10 questions (with equal distribution to all units in the syllabus). Each question carries 2 marks.

Part B: 5 question with either or type (with equal distribution to all units in the syllabus). Each question carries 16 marks.
The total marks scored by the candidates will be reduced to the maximum prescribed in the Regulations.

10. **Effective Period of Operation for the Arrear Candidates :** Two Year grace period is provided for the candidates to complete the arrear examination, if any.

Registrar

11. Syllabus

I Semester

113CSPT01 - APPLIED PROBABILITY AND STATISTICS

OBJECTIVES:

- To develop the ability to apply the concepts of Matrix theory and Liner programming in Engineering problems.
- To familiarize the students in calculus of variations
- To introduce the concepts of one dimensional and two dimensional random variables.

UNIT I CALCULUS OF VARIATION

Introduction – Eucler's equation – several dependent variables Lagrange's equation of dynamics – integrals involving derivatives higher than the first – problem with constraints – direct methods and Eigen value problems.

UNIT II MATRIX THEORY

Eigen value using OR transformation – generalized eigen vectors – canonical forms singular value decomposition and applications- pseudo inverse- least square approximation

UNIT III LINEAR PROGRAMMING PROBLEM

Graphical method – simplex method – Big M technique- Integer programming

UNIT IV ONE DIMENSIONAL RANDOM VARIABLES

Random variables - Probability function – Moments – Moment generating functions and their properties – Binomial, Poisson, Geometric, Uniform, Exponential, Gamma and Normal distributions – Functions of a Random Variable.

UNIT V TWO DIMENSIONAL RANDOM VARIABLES

Joint distributions – Marginal and Conditional distributions – Functions of two dimensional random variables – Regression Curve – Correlation.

REFERENCES:

1. Gupta, A.S Calculus of Variations with Applications, Prentice – Hall of India New Delhi 1997

2. Bronson.R. " Matrix Operation " schaum outline series, Mc Graw Hill, Newyork,1989.

3. Taha H.A, "Operation Research – An Introduction " Prentice hall of India ,2001

4. Jay L. Devore, "Probability and Statistics For Engineering and the Sciences",Thomson and Duxbury, 2002

5. Gupta S.C. and Kapoor V.K."Fundamentals of Mathematical Statistics", Sultan an Sons, 2001.

113CSPT02 - DESIGN AND MANAGEMENT OF COMPUTER NETWORKS

UNIT I INTRODUCTION TO NETWORK MANAGEMENT

Overview of Analysis, Architecture and Design Process-System Methodology, Service methodology, Service Description - Service characteristics - Performance Characteristics - Network supportability - Requirement analysis – User Requirements – Application Requirements – Device Requirements – Network Requirements – Other Requirements - Requirement specification and map.

UNIT II REQUIREMENTS ANALYSIS

Requirement Analysis Process – Gathering and Listing Requirements- Developing service metrics – Characterizing behavior – Developing RMA requirements – Developing delay Requirements - Developing capacity Requirements - Developing supplemental performance Requirements – Requirements mapping – Developing the requirements specification

UNIT III FLOW ANALYSIS

Individual and Composite Flows – Critical Flows - Identifying and developing flows – Data sources and sinks – Flow models- Flow prioritization – Flow specification algorithms – Example Applications of Flow Analysis

UNIT IV NETWORK ARCHITECTURE

Architecture and design – Component Architectures – Reference Architecture – Architecture Models – System and Network Architecture – Addressing and Routing Architecture – Addressing and Routing Fundamentals – Addressing Mechanisms – Addressing Strategies – Routing Strategies – Network Management Architecture – Network Management Mechanisms Performance Architecture – Performance Mechanisms – Security and Privacy Architecture – Planning security and privacy Mechanisms

UNIT V NETWORK DESIGN

Design Concepts – Design Process - Network Layout – Design Traceability – Design Metrics – Logical Network Design – Topology Design – Bridging, Switching and Routing Protocols- Physical Network Design – Selecting Technologies and Devices for Campus and Enterprise Networks – Optimizing Network Design

REFERENCES:

1. Network Analysis, Architecture, and Design By James D. McCabe, Morgan Kaufmann, Third Edition, 2007.ISBN-13: 978-0123704801
2. Computer Networks: A Systems Approach by Larry L. Peterson, Bruce S. Davie - 2007, Elsevier Inc.
3. Top-down Network Design: [a Systems Analysis Approach to Enterprise Network Design] By Priscilla Oppenheimer, Cisco Press , 3rd Edition, ISBN-13: 978-1-58720- 283-4 ISBN-10: 1-58720-283-2
4. Integrated Management of Networked Systems: Concepts, Architectures, and Their Operational Application (The Morgan Kaufmann Series in Networking), Heinz-Gerd Hegering, Sebastian Abeck, and Bernhard Neumair, 1999.
5. "Network Design and Management" – by Steven T.Karris, Orchard publications, Second edition, Copyright 2009, ISBN 978-1-934404-15-7
6. "Network Design, Management and Technical Perspective", Teresa C. Mann-Rubinson and Kornel Terplan, CRC Press, 1999
7. "Ethernet Networks-Design, Implementation, Operation and Management by Gilbert Held, John Wiley and sons, Fourth Edition
8. James Kurose and Keith Ross, "Computer Networking: A Top-Down Approach Featuring the Internet", 1999

113CSPT03 - ADVANCED DATA STRUCTURES AND ALGORITHMS

OBJECTIVES:

- To understand the principles of iterative and recursive algorithms.
- To learn the graph search algorithms.
- To study network flow and linear programming problems.
- To learn the hill climbing and dynamic programming design techniques.
- To develop recursive backtracking algorithms.
- To get an awareness of NP completeness and randomized algorithms.
- To learn the principles of shared and concurrent objects.
- To learn concurrent data structures.

UNIT I ITERATIVE AND RECURSIVE ALGORITHMS

Iterative Algorithms: Measures of Progress and Loop Invariants-Paradigm Shift: Sequence of Actions versus Sequence of Assertions- Steps to Develop an Iterative Algorithm-Different Types of Iterative Algorithms--Typical Errors-Recursion-Forward versus Backward- Towers of Hanoi-Checklist for Recursive Algorithms-The Stack Frame-Proving Correctness with Strong Induction- Examples of Recursive Algorithms-Sorting and Selecting Algorithms-Operations on Integers- Ackermann's Function- Recursion on Trees-Tree Traversals-Examples- Generalizing the Problem - Heap Sort and Priority Queues-Representing Expressions.

UNIT II OPTIMISATION ALGORITHMS

Optimization Problems-Graph Search Algorithms-Generic Search-Breadth-First Search-Dijkstra's Shortest-Weighted-Path -Depth-First Search-Recursive Depth-First Search-Linear Ordering of a Partial Order- Network Flows and Linear Programming-Hill Climbing-Primal Dual Hill Climbing- Steepest Ascent Hill Climbing-Linear Programming-Recursive Backtracking-Developing Recursive Backtracking Algorithm- Pruning Branches-Satisfiability

UNIT III DYNAMIC PROGRAMMING ALGORITHMS

Developing a Dynamic Programming Algorithm-Subtle Points- Question for the Little Bird-Subinstances and Subsolutions-Set of Substances-Decreasing Time and Space-Number of Solutions-Code. Reductions and NP-Completeness**-**Satisfiability-Proving NP-Completeness-3-Coloring- Bipartite Matching. Randomized Algorithms**-**Randomness to Hide Worst Cases-Optimization Problems with a Random Structure.

UNIT IV SHARED OBJECTS AND CONCURRENT OBJECTS

Shared Objects and Synchronization -Properties of Mutual Exclusion-The Mora l- The Producer–Consumer Problem -The Readers–Writers Problem-Realities of Parallelization-Parallel Programming- Principles- Mutual Exclusion-Time- Critical Sections--Thread Solutions-The Filter Lock-Fairness-Lamport's Bakery Algorithm-Bounded Timestamps-Lower Bounds on the Number of Locations-Concurrent Objects- Concurrency and Correctness-Sequential Objects-Quiescent Consistency- Sequential Consistency-Linearizability- Formal Definitions- Progress Conditions- The Java Memory Model

UNIT V CONCURRENT DATA STRUCTURES

Practice-Linked Lists-The Role of Locking-List-Based Sets-Concurrent Reasoning- Coarse- Grained Synchronization-Fine-Grained Synchronization-Optimistic Synchronization- Lazy Synchronization-Non-Blocking Synchronization-Concurrent Queues and the ABA Problem- Queues-A Bounded Partial Queue-An Unbounded Total Queue-An Unbounded Lock-Free Queue-Memory Reclamation and the ABA Problem- Dual Data Structures- Concurrent Stacks and Elimination- An Unbounded Lock-Free Stack- Elimination-The Elimination Backoff Stack

OUTCOMES:

Upon completion of the course, the students will be able to

- Design and apply iterative and recursive algorithms.
- Design and implement optimisation algorithms in specific applications.
- Design appropriate shared objects and concurrent objects for applications.
- Implement and apply concurrent linked lists, stacks, and queues.

REFERENCES:

1. Jeff Edmonds, "How to Think about Algorithms", Cambridge University Press, 2008.
2. M. Herlihy and N. Shavit, "The Art of Multiprocessor Programming", Morgan Kaufmann, 2008.
3. Steven S. Skiena, "The Algorithm Design Manual", Springer, 2008.
4. Peter Brass, "Advanced Data Structures", Cambridge University Press, 2008.
5. S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, "Algorithms" , McGrawHill, 2008.
6. J. Kleinberg and E. Tardos, "Algorithm Design", Pearson Education, 2006.
7. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithms", PHI Learning Private Limited, 2012.
8. Rajeev Motwani and Prabhakar Raghavan, "Randomized Algorithms", Cambridge University Press, 1995.
9. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms**",** Addison-Wesley, 1975.
10. A. V. Aho, J. E. Hopcroft, and J. D. Ullman,"Data Structures and Algorithms", Pearson,2006.

113CSPT04 - MULTICORE ARCHITECTURES

OBJECTIVES:

- To understand the recent trends in the field of Computer Architecture and identify performance related parameters
- To appreciate the need for parallel processing
- To expose the students to the problems related to multiprocessing
- To understand the different types of multicore architectures
- To expose the students to warehouse-scale and embedded architectures

UNIT I FUNDAMENTALS OF QUANTITATIVE DESIGN AND ANALYSIS

Classes of Computers – Trends in Technology, Power, Energy and Cost – Dependability – Measuring, Reporting and Summarizing Performance – Quantitative Principles of Computer Design – Classes of Parallelism - ILP, DLP, TLP and RLP - Multithreading - SMT and CMP Architectures – Limitations of Single Core Processors - The Multicore era – Case Studies of Multicore Architectures.

UNIT II DLP IN VECTOR, SIMD AND GPU ARCHITECTURES

Vector Architecture - SIMD Instruction Set Extensions for Multimedia – Graphics Processing Units - Detecting and Enhancing Loop Level Parallelism - Case Studies.

UNIT III TLP AND MULTIPROCESSORS

Symmetric and Distributed Shared Memory Architectures – Cache Coherence Issues - Performance Issues – Synchronization Issues – Models of Memory Consistency - Interconnection Networks – Buses, Crossbar and Multi-stage Interconnection Networks.

UNIT IV RLP AND DLP IN WAREHOUSE-SCALE ARCHITECTURES

Programming Models and Workloads for Warehouse-Scale Computers – Architectures for Warehouse-Scale Computing – Physical Infrastructure and Costs – Cloud Computing – Case Studies.

UNIT V ARCHITECTURES FOR EMBEDDED SYSTEMS

Features and Requirements of Embedded Systems – Signal Processing and Embedded Applications – The Digital Signal Processor – Embedded Multiprocessors - Case Studies.

OUTCOMES:

Upon completion of the course, the students will be able to

- Identify the limitations of ILP and the need for multicore architectures
- Discuss the issues related to multiprocessing and suggest solutions
- Point out the salient features of different multicore architectures and how they exploit parallelism
- Critically analyze the different types of inter connection networks
- Discuss the architecture of GPUs, warehouse-scale computers and embedded processors

REFERENCES:

1. John L. Hennessey and David A. Patterson, " Computer Architecture – A Quantitative Approach", Morgan Kaufmann / Elsevier, 5th edition, 2012.
2. Kai Hwang, "Advanced Computer Architecture", Tata McGraw-Hill Education, 2003
3. Richard Y. Kain, "Advanced Computer Architecture a Systems Design Approach", Prentice Hall, 2011.
4. David E. Culler, Jaswinder Pal Singh, "Parallel Computing Architecture : A Hardware/ Software Approach" , Morgan Kaufmann / Elsevier, 1997.

113CSPP01 - ADVANCED DATA STRUCTURES LABORATORY

OBJECTIVES:

- To learn to implement iterative and recursive algorithms.
- To learn to design and implement algorithms using hill climbing and dynamic programming techniques.
- To learn to implement shared and concurrent objects.
- To learn to implement concurrent data structures.

LAB EXERCISES:

Each student has to work individually on assigned lab exercises. Lab sessions could be scheduled as one contiguous four-hour session per week or two two-hour sessions per week. There will be about 15 exercises in a semester. It is recommended that all implementations are carried out in Java. If C or C++ has to be used, then the threads library will be required for concurrency. Exercises should be designed to cover the following topics:

- Implementation of graph search algorithms.
- Implementation and application of network flow and linear programming problems.
- Implementation of algorithms using the hill climbing and dynamic programming design techniques.
- Implementation of recursive backtracking algorithms.
- Implementation of randomized algorithms.
- Implementation of various locking and synchronization mechanisms for concurrent linked lists, concurrent queues, and concurrent stacks.
- Developing applications involving concurrency.

OUTCOMES:

Upon completion of the course, the students will be able to

1. Design and apply iterative and recursive algorithms.
2. Design and implement algorithms using the hill climbing and dynamic programming and recursive backtracking techniques.
3. Design and implement optimisation algorithms for specific applications.
4. Design and implement randomized algorithms.
5. Design appropriate shared objects and concurrent objects for applications.
6. Implement and apply concurrent linked lists, stacks, and queues.

REFERENCES:

1. Jeff Edmonds, "How to Think about Algorithms", Cambridge University Press, 2008.
2. M. Herlihy and N. Shavit, "The Art of Multiprocessor Programming", Morgan Kaufmann, 2008.
3. Steven S. Skiena, "The Algorithm Design Manual", Springer, 2008.
4. Peter Brass, "Advanced Data Structures", Cambridge University Press, 2008.
5. S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, "Algorithms" , McGrawHill, 2008.
6. J. Kleinberg and E. Tardos, "Algorithm Design", Pearson Education, 2006.
7. T. H. Cormen, C.2 E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithms", PHI Learning Private Limited, 2012.
8. Rajeev Motwani and Prabhakar Raghavan, "Randomized Algorithms", Cambridge University Press, 1995.
9. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms**",** Addison-Wesley, 1975.
10. A. V. Aho, J. E. Hopcroft, and J. D. Ullman,"Data Structures and Algorithms", Pearson,2006.

CASE STUDY : 1

Analyzing the performance of various configurations and protocols in LAN.

1.1. Establishing a Local Area Network (LAN): The main objective is to set up a Local Area Network,concepts involved in this network are IP addressing and the Address Resolution Protocol (ARP). The required equipments are192.168.1.1 ,192.168.1.2, 192.168.1.3,Host A Host B Host C, Switch/HUB, three PC`s equipped with at least one NIC, one HUB or Switch and the necessary cables. Once the physical LAN is set up the hosts need to be configured using the ifconfig command. To verify communication among the machines the ping command is used. Next, to manipulate the routing tables at the hosts to understand how machines know where to send packets. Since the ifconfig command places a default route into the routing tables this route must be deleted. to ´blindfoldˇ the machine. The ping command is used again to show that communication is no longer available. To re-establish communication the routes are put back into the routing table one host at a time. Communication is once again verified using the ping command.

1.2. Connecting two LANs using multi-router topology with static routes: The main objective is to extend routing connection by using multiple routers. The concepts include IP addressing and basic network routing principles. Connect two LANs topology. During router configuration attention is paid to the types of interfaces as additional issues are involved with set-up. For example, the serial interfaces require clocking mechanisms to be set correctly. Once the interfaces are working the ping command is used to check for communication between LANs. The failure of communication illustrates the need for routes to be established inside the routing infrastructure. Static routes are used to show how packets can be transported through any reasonable route. It is run trace route on two different configurations to demonstrate the implementation of different routes.

1.3 Analyzing the performance of various configurations and protocols Original TCP versus the above modified one: To compare the performance between the operation of TCP with congestion control and the operation of TCP as implemented . The main objective is for students to examine how TCP responds to a congested network. The concepts involved in the lab include network congestion and the host responsibilites for communicating over a network. This lab requires three PC’s connected to a switch.One PC is designated as the target host and the other two PC’s will transfer a file from the targethost using FTP. A load is placed on the network to simulate congestion and the file is transfered, first by the host using the normal TCP and then by the host using the modified version. This procedure is performed multiple times to determine average statistics. The students are then asked to summarize the results and draw conclusions about the performance differences and the underlying implications for hosts operating in a network environment.

Case Study 2:

RIP and OSPF Redistribution

This case study addresses the issue of integrating Routing Information Protocol (RIP) networks with Open Shortest Path First (OSPF) networks. Most OSPF networks also use RIP to communicate with hosts or to communicate with portions of the internetwork that do not use OSPF. This case study should provide examples of how to complete the following phases in redistributing information between RIP and OSPF networks, including the following topics:

- Configuring a RIP Network
- Adding OSPF to the Center of a RIP Network
- Adding OSPF Areas
- Setting Up Mutual Redistribution

‘Case Study 3:
Dial-on-Demand Routing
This case study should describe the use of DDR to connect a worldwide network that consists of a central site located in Mumbai and remote sites located in Chennai, Bangalore, and Hyderabad. The following scenarios should be considered:
• Having the Central Site Dial Out
Describe the central and remote site configurations for three setups: a central site with one interface per remote site, a single interface for multiple remote sites, and multiple interfaces for multiple remote sites. Include examples of the usage of rotary groups and access lists.
• Having the Central and Remote Sites Dial In and Dial Out
Describe the central and remote site configurations for three setups: central site with one interface per remote site, a single interface for multiple remote sites, and multiple interfaces for multiple remote sites. Also describes the usage of Point-to-Point Protocol (PPP) encapsulation and the Challenge Handshake Authentication Protocol (CHAP).
• Having Remote Sites Dial Out
A common configuration is one in which the remote sites place calls to the central site but the central site does not dial out. In a “star” topology, it is possible for all of the remote routers to have their serial interfaces on the same subnet as the central site serial interface.
• Using DDR as a Backup to Leased Lines
Describes the use of DDR as a backup method to leased lines and provides examples of how to use floating static routes on single and shared interfaces.
• Using Leased Lines and Dial Backup
Describes the use of Data Terminal Ready (DTR) dialing and V.25bis dialing with leased lines.

Case Study 4:
Network Security
This case study should provide the specific actions you can take to improve the security of your network. Before going into specifics, however, you should understand the following basic concepts that are essential to any security system:
• Know your enemy
This case study refers to attackers or intruders. Consider who might want to circumvent your security measures and identify their motivations. Determine what they might want to do and the damage that they could cause to your network. Security measures can never make it impossible for a user to perform unauthorized tasks with a computer system. They can only make it harder. The goal is to make sure the network security controls are beyond the attacker’s ability or motivation.
• Count the cost
Security measures almost always reduce convenience, especially for sophisticated users. Security can delay work and create expensive administrative and educational overhead. It can use significant computing resources and require dedicated hardware. When you design your security measures, understand their costs and weigh those costs against the potential benefits. To do that, you must understand the costs of the measures themselves and the costs and likelihoods of security breaches. If you incur security costs out of proportion to the actual dangers, you have done yourself a disservice.
• Identify your assumptions
Every security system has underlying assumptions. For example, you might assume that your network is not tapped, or that attackers know less than you do, that they are using standard software, or that a locked room is safe. Be sure to examine and justify your assumptions. Any hidden assumption is a potential security hole.
• Control your secrets
Most security is based on secrets. Passwords and encryption keys, for example, are secrets. Too often, though, the secrets are not really all that secret. The most important part of keeping secrets is knowing the areas you need to protect. What knowledge would enable someone to circumvent your system? You should jealously guard that knowledge and assume that everything else is known to your adversaries. The more secrets you have, the harder it will be to keep all of them. Security systems should be designed so that only a limited number of secrets need to be kept.
• Know your weaknesses
Every security system has vulnerabilities. You should understand your system’s weak

points and know how they could be exploited. You should also know the areas that present the largest danger and prevent access to them immediately. Understanding the weak points is the first step toward turning them into secure areas.

- Limit the scope of access

You should create appropriate barriers inside your system so that if intruders access one part of the system, they do not automatically have access to the rest of the system. The security of a system is only as good as the weakest security level of any single host in the system.

- Remember physical security Physical access to a computer (or a router) usually gives a sufficiently sophisticated user total control over that computer. Physical access to a network link usually allows a person to tap that link, jam it, or inject traffic into it. It makes no sense to install complicated software security measures when access to the hardware is not controlled.

Case Study 5:
Controlling Traffic Flow

In this case study, the firewall router allows incoming new connections to one or more communication servers or hosts. Having a designated router act as a firewall is desirable because it clearly identifies the router's purpose as the external gateway and avoids encumbering other routers with this task. In the event that the internal network needs to isolate itself, the firewall router provides the point of isolation so that the rest of the internal network structure is not affected. Connections to the hosts are restricted to incoming file transfer protocol (FTP) requests and email services. The incoming Telnet, or modem connections to the communication server are screened by the communication server running TACACS username authentication.

Case Study 6:
Defining Access Lists

Access lists define the actual traffic that will be permitted or denied, whereas an access group applies an access list definition to an interface. Access lists can be used to deny connections that are known to be a security risk and then permit all other connections, or to permit those connections that are considered acceptable and deny all the rest. For firewall implementation, the latter is the more secure method. In this case study, incoming email and news are permitted for a few hosts, but FTP, Telnet, and rlogin services are permitted only to hosts on the firewall subnet. IP extended access lists (range 100 to 199) and transmission control protocol (TCP) or user datagram protocol (UDP) port numbers are used to filter traffic. When a connection is to be established for email, Telnet, FTP, and so forth, the connection will attempt to open a service on a specified port number. You can, therefore, filter out selected types of connections by denying packets that are attempting to use that service. An access list is invoked after a routing decision has been made but before the packet is sent out on an interface. The best place to define an access list is on a preferred host using your favorite text editor. You can create a file that contains the access-list commands, place the file (marked readable) in the default TFTP directory, and then network load the file onto the router.

Case Study 7:
Configuring a fire wall

Consider a Fire wall communication server with single inbound modem. Configure the modem to ensure security for LAN

Case Study 8:
Integrating EIGRP (Enhanced Interior Gateway Routing Protocol) into Existing Networks:

The case study should provide the benefits and considerations involved in integrating Enhanced IGRP into the following types of internetworks:

- IP—The existing IP network is running IGRP
- Novell IPX—The existing IPX network is running RIP and SAP
- AppleTalk—The existing AppleTalk network is running the Routing Table Maintenance Protocol (RTMP)

When integrating Enhanced IGRP into existing networks, plan a phased implementation. Add Enhanced IGRP at the periphery of the network by configuring Enhanced IGRP on a boundary router on the backbone off the core network. Then integrate Enhanced IGRP into the core network.

213CSPT01 - THEORETICAL FOUNDATIONS OF COMPUTER SCIENCE

OBJECTIVES:

- To review sets, relations, functions, and other foundations
- To understand propositional and predicate logics and their applications
- To understand lambda calculus and functional programming
- To understand graph structures and their applications
- To understand formal models of computation, computability, and decidability

UNIT I FOUNDATIONS

Sets – relations – equivalence relations – partial orders – functions – recursive functions – sequences – induction principle – structural induction – recursive algorithms – counting – pigeonhole principle – permutations and combinations – recurrence relations

UNIT II LOGIC AND LOGIC PROGRAMMING

Propositional logic – syntax – interpretations and models – deduction theorems – normal forms – inference rules – SAT solvers - predicate logic – syntax – proof theory – semantics of predicate logic – undecidability of predicate logic – inferences in first-order logic – logic programming – definite programs – SLD resolution – normal programs – SLDNF resolution – introduction to Prolog

UNIT III LAMBDA CALCULUS AND FUNCTIONAL PROGRAMMING

Lambda notation for functions – syntax – curried functions – parametric polymorphism – lambda reduction – alpha reduction – beta reduction – beta abstraction – extensionality theorem – delta reduction – reduction strategies – normal forms – Church-Rosser Theorems – pure lambda calculus – constants – arithmetic – conditionals – Iteration – recursion – introduction to functional programming

UNIT IV GRAPH STRUCTURES

Tree Structures – Graph structures – graph representations – regular graph structures – random graphs – Connectivity – Cycles – Graph Coloring – Cliques, Vertex Covers, Independent sets – Spanning Trees – network flows – matching

UNIT V STATE MACHINES

Languages and Grammars – Finite State Machines – State machines and languages – Turing Machines – Computational Complexity – computability – Decidability – Church's Thesis

OUTCOMES:

Upon Completion of the course,☐the students will be able

- To explain sets, relations, functions
- To conduct proofs using induction, pigeonhole principle, and logic
- To apply counting, permutations, combinations, and recurrence relations
- To apply recursive functions and lambda calculus
- To explain logic programming and functional programming principles
- To apply sequential structures, tree structures, and graph structures
- To explain computational models, computability, and complexity

REFERENCES:

1. Uwe Schoning, "Logic for Computer Scientists", Birkhauser, 2008.
2. M. Ben-Ari, "Mathematical logic for computer science", Second Edition, Springer, 2003.
3. John Harrison, "Handbook of Practical Logic and Automated Reasoning", Cambridge University Press, 2009.
4. Greg Michaelson, "An introduction to functional programming through lambda calculus", Dover Publications, 2011.
5. Kenneth Slonneger and Barry Kurtz, "Formal syntax and semantics of programming languages", Addison Wesley, 1995.
6. Kenneth H. Rosen, "Discrete Mathematics and its applications", Seventh Edition, Tata McGraw Hill, 2011.
7. Sriram Pemmaraju and Steven Skiena, "Computational Discrete Mathematics", Cambridge University Press, 2003.
8. M. Huth and M. Ryan, "Logic in Computer Science – Modeling and Reasoning about systems", Second Edition, Cambridge University Press, 2004.
9. Norman L. Biggs, "Discrete Mathematics", Second Edition, Oxford University Press, 2002.
10. Juraj Hromkovic, "Theoretical Computer Science", Springer, 1998.
11. J. E. Hopcroft, Rajeev Motwani, and J. D. Ullman, "Introduction to Automata Theory, Languages, and Computation", Third Edition, Pearson, 2008.

213CSPT02 - ADVANCED DATABASES

OBJECTIVES:

- To learn the modeling and design of databases.
- To acquire knowledge on parallel and distributed databases and its applications.
- To study the usage and applications of Object Oriented database
- To understand the principles of intelligent databases.
- To understand the usage of advanced data models.
- To learn emerging databases such as XML, Cloud and Big Data.
- To acquire inquisitive attitude towards research topics in databases.

UNIT I PARALLEL AND DISTRIBUTED DATABASES

Database System Architectures: Centralized and Client-Server Architectures – Server System Architectures – Parallel Systems- Distributed Systems – Parallel Databases: I/O Parallelism – Inter and Intra Query Parallelism – Inter and Intra operation Parallelism – Design of Parallel Systems- Distributed Database Concepts - Distributed Data Storage – Distributed Transactions – Commit Protocols – Concurrency Control – Distributed Query Processing – Case Studies

UNIT II OBJECT AND OBJECT RELATIONAL DATABASES

Concepts for Object Databases: Object Identity – Object structure – Type Constructors – Encapsulation of Operations – Methods – Persistence – Type and Class Hierarchies – Inheritance – Complex Objects – Object Database Standards, Languages and Design: ODMG Model – ODL – OQL – Object Relational and Extended – Relational Systems: Object Relational features in SQL/Oracle – Case Studies.

UNIT III INTELLIGENT DATABASES

Active Databases: Syntax and Semantics (Starburst, Oracle, DB2)- Taxonomy- Applications- Design Principles for Active Rules- Temporal Databases: Overview of Temporal Databases- TSQL2- Deductive Databases: Logic of Query Languages – Datalog- Recursive Rules- Syntax and Semantics of Datalog Languages- Implementation of Rules and Recursion- Recursive Queries in SQL- Spatial Databases- Spatial Data Types- Spatial Relationships- Spatial Data Structures-Spatial Access Methods- Spatial DB Implementation.

UNIT IV ADVANCED DATA MODELS

Mobile Databases: Location and Handoff Management - Effect of Mobility on Data Management - Location Dependent Data Distribution - Mobile Transaction Models - Concurrency Control - Transaction Commit Protocols- Multimedia Databases- Information Retrieval- Data Warehousing- Data Mining- Text Mining.

UNIT V EMERGING TECHNOLOGIES

XML Databases: XML-Related Technologies-XML Schema- XML Query Languages- Storing XML in Databases-XML and SQL- Native XML Databases- Web Databases- Geographic Information Systems- Biological Data Management- Cloud Based Databases: Data Storage Systems on the Cloud- Cloud Storage Architectures-Cloud Data Models- Query Languages- Introduction to Big Data-Storage-Analysis.

OUTCOMES:

Upon completion of the course, the students will be able to

- Select the appropriate high performance database like parallel and distributed database
- Model and represent the real world data using object oriented database
- Design a semantic based database to meaningful data access
- Embed the rule set in the database to implement intelligent databases
- Represent the data using XML database for better interoperability
- Handle Big data and store in a transparent manner in the cloud
- To solve the issues related to the data storage and retrieval

REFERENCES:

1. R. Elmasri, S.B. Navathe, "Fundamentals of Database Systems", Fifth Edition, Pearson Education/Addison Wesley, 2007.
2. Thomas Cannolly and Carolyn Begg, "Database Systems, A Practical Approach to Design, Implementation and Management", Third Edition, Pearson Education, 2007.
3. Henry F Korth, Abraham Silberschatz, S. Sudharshan, "Database System Concepts", Fifth Edition, McGraw Hill, 2006.
4. C.J.Date, A.Kannan and S.Swamynathan, "An Introduction to Database Systems", Eighth Edition, Pearson Education, 2006.
5. Raghu Ramakrishnan, Johannes Gehrke, "Database Management Systems", McGraw Hill, Third Edition 2004.

213CSPT03 - PRINCIPLES OF PROGRAMMING LANGUAGES

OBJECTIVES:

- To understand and describe syntax and semantics of programming languages
- To understand data, data types, and basic statements
- To understand call-return architecture and ways of implementing them
- To understand object-orientation, concurrency, and event handling in programming languages
- To develop programs in non-procedural programming paradigms

UNIT I SYNTAX AND SEMANTICS

Evolution of programming languages – describing syntax – context-free grammars – attribute grammars – describing semantics – lexical analysis – parsing – recursive-decent – bottomup parsing

UNIT II DATA, DATA TYPES, AND BASIC STATEMENTS

Names – variables – binding – type checking – scope – scope rules – lifetime and garbage collection – primitive data types – strings – array types – associative arrays – record types – union types – pointers and references – Arithmetic expressions – overloaded operators – type conversions – relational and boolean expressions – assignment statements – mixedmode assignments – control structures – selection – iterations – branching – guarded statements

UNIT III SUBPROGRAMS AND IMPLEMENTATIONS

Subprograms – design issues – local referencing – parameter passing – overloaded methods – generic methods – design issues for functions – semantics of call and return – implementing simple subprograms – stack and dynamic local variables – nested subprograms – blocks – dynamic scoping

UNIT IV OBJECT-ORIENTATION, CONCURRENCY, AND EVENT HANDLING

Object-orientation – design issues for OOP languages – implementation of object-oriented constructs – concurrency – semaphores – monitors – message passing – threads – statement level concurrency – exception handling – even handling

UNIT V FUNCTIONAL AND LOGIC PROGRAMMING LANGUAGES

Introduction to lambda calculus – fundamentals of functional programming languages – Programming with Scheme – Programming with ML – Introduction to logic and logic programming – Programming with Prolog – multi-paradigm languages

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Describe syntax and semantics of programming languages
- Explain data, data types, and basic statements of programming languages
- Design and implement subprogram constructs
- Apply object-oriented, concurrency, and event handling programming constructs
- Develop programs in Scheme, ML, and Prolog
- Understand and adopt new programming languages

REFERENCES:

1. Robert W. Sebesta, "Concepts of Programming Languages", Tenth Edition, Addison Wesley, 2012.
2. Michael L. Scott, "Programming Language Pragmatics", Third Edition, Morgan Kaufmann, 2009.
3. R. Kent Dybvig, "The Scheme programming language", Fourth Edition, MIT Press, 2009.
4. Jeffrey D. Ullman, "Elements of ML programming", Second Edition, Prentice Hall, 1998.
5. Richard A. O'Keefe, "The craft of Prolog", MIT Press, 2009.
6. W. F. Clocksin and C. S. Mellish, "Programming in Prolog: Using the ISO Standard", Fifth Edition, Springer, 2003.

213CSPT04 - ADVANCED OPERATING SYSTEMS

OBJECTIVES:

- To learn the fundamentals of Operating Systems
- To gain knowledge on Distributed operating system concepts that includes architecture, Mutual exclusion algorithms, Deadlock detection algorithms and agreement protocols
- To gain insight on to the distributed resource management components viz. the algorithms for implementation of distributed shared memory, recovery and commit protocols
- To know the components and management aspects of Real time, Mobile operating systems

UNIT I FUNDAMENTALS OF OPERATING SYSTEMS

Overview – Synchronization Mechanisms – Processes and Threads - Process Scheduling – Deadlocks: Detection, Prevention and Recovery – Models of Resources – Memory Management Techniques.

UNIT II DISTRIBUTED OPERATING SYSTEMS

Issues in Distributed Operating System – Architecture – Communication Primitives – Lamport's Logical clocks – Causal Ordering of Messages – Distributed Mutual Exclusion Algorithms – Centralized and Distributed Deadlock Detection Algorithms – Agreement Protocols.

UNIT III DISTRIBUTED RESOURCE MANAGEMENT

Distributed File Systems – Design Issues - Distributed Shared Memory – Algorithms for Implementing Distributed Shared memory–Issues in Load Distributing – Scheduling Algorithms – Synchronous and Asynchronous Check Pointing and Recovery – Fault Tolerance – Two-Phase Commit Protocol – Nonblocking Commit Protocol – Security and Protection.

UNIT IV REAL TIME AND MOBILE OPERATING SYSTEMS

Basic Model of Real Time Systems - Characteristics- Applications of Real Time Systems – Real Time Task Scheduling - Handling Resource Sharing - Mobile Operating Systems – Micro Kernel Design - Client Server Resource Access – Processes and Threads - Memory Management - File system.

UNIT V CASE STUDIES

Linux System: Design Principles - Kernel Modules - Process Management Scheduling - Memory Management - Input-Output Management - File System - Interprocess Communication. iOS and Android: Architecture and SDK Framework - Media Layer - Services Layer - Core OS Layer - File System.

OUTCOMES:

Upon Completion of the course, the students should be able to:

- Discuss the various synchronization, scheduling and memory management issues
- Demonstrate the Mutual exclusion, Deadlock detection and agreement protocols of Distributed operating system
- Discuss the various resource management techniques for distributed systems
- Identify the different features of real time and mobile operating systems
- Install and use available open source kernel
- Modify existing open source kernels in terms of functionality or features used

REFERENCES:

1. Mukesh Singhal and Niranjan G. Shivaratri, "Advanced Concepts in Operating Systems – Distributed, Database, and Multiprocessor Operating Systems", Tata McGraw-Hill, 2001.
2. Abraham Silberschatz; Peter Baer Galvin; Greg Gagne, "Operating System Concepts", Seventh Edition, John Wiley & Sons, 2004.
3. Daniel P Bovet and Marco Cesati, "Understanding the Linux kernel", 3rd edition, O'Reilly, 2005.
4. Rajib Mall, "Real-Time Systems: Theory and Practice", Pearson Education India, 2006.
5. Neil Smyth, "iPhone iOS 4 Development Essentials – Xcode", Fourth Edition, Payload media, 2011.

213CSPP01 - ADVANCED DATABASE LABORATORY

OBJECTIVES:

- To learn to work on distributed data bases
- To understand and work on object oriented databases
- To gain knowledge in parallel data base by experimenting it
- To learn to work on active database
- To study and explore deductive database
- To work on the data mining tool *weka*
- To represent and work with the database using XML

DISTRIBUTED DATABASE:

1. Consider a distributed database for a bookstore with 4 sites called S1, S2, S3 and S4. Consider the following relations:
Books (ISBN, primary Author, topic, total Stock, price)
Book Store (store No, city, state, zip, inventoryValue)
Stock (store No, ISBN, Qty)
Total Stock is the total number of books in stock and inventory Value is the total inventory value for the store in dollars.
Consider that Books are fragmented by price amounts into:
F1: Books: price up to $20
F2: Books: price from $20.01 to $50
F3: Books: price from $50.01 to $100
F4: Books: price $100.01 and above
Similarly, Book Stores are divided by ZIP codes into:
S1: Bookstore: Zip up to 25000
S2: Bookstore: Zip 25001 to 50000
S3: Bookstore: Zip 50001 to 75000
S4: Bookstore: Zip 75001 to 99999
Task: Write SQL query for the following
1. Insert and Display details in each table.
2. Find the total number of books in stock where price is between $15 and $55.
3. Update the book price of book No=1234 from $45 to $55 at site S3.
4. Find total number of book at site S2.

2. Implement deadlock detection algorithm for distributed database using wait-for graph and test with the following information.
Consider five transactions T1, T2, T3, T4 and T5 with
T1 initiated at site S1 and spawning an agent at site S2
T2 initiated at site S3 and spawning an agent at site S1
T3 initiated at site S1 and spawning an agent at site S3
T4 initiated at site S2 and spawning an agent at site S3
T5 initiated at site S3
The locking information for these transactions is shown in the following table

Transactions	Data items locked by transactions	Data items transaction is waiting for	Site involved in operations
T1	X1	X8	S1
T1	X6	X2	S2
T2	X4	X1	S1
T2	X5	-	S3
T3	X2	X7	S1
T3	-	X3	S3
T4	X7	-	S2
T4	X8	X5	S3
T5	X3	X7	S3

Produce local wait for graph for each of the sites and construct global wait for graph and check for dead lock.

OBJECT ORIENTED DATABASE:

3. A University wants to track persons associated with them. A person can be an Employee or Student. Employees are Faculty, Technicians and Project associates. Students are Full time students, Part time students and Teaching Assistants.
a) Design an Enhanced Entity Relationship (EER) Model for university database.
Write OQL for the following
i. Insert details in each object.
ii. Display the Employee details.
iii. Display Student Details.
iv. Modify person details.
v. Delete person details.
b) Extend the design by incorporating the following information.
Students are registering for courses which are handled by instructor researchers (graduate students). Faculty are advisors to graduate students. Instructor researchers' class is a category with super class of faculty and graduate students. Faculty are having sponsored research projects with a grant supporting instruction researchers. Grants are sanctioned by different agencies. Faculty belongs to different departments. Department is chaired by a faculty. Implement for the Insertion and Display of details in each class.

PARALLEL DATABASE:

4. Consider the application for University Counselling for Engineering Colleges. The college, department and vacancy details are maintained in 3 sites. Students are allocated colleges in these 3 sites simultaneously. Implement this application using parallel database [State any assumptions you have made].
5. There are 5 processors working in a parallel environment and producing output. The output record contains college details and students mark information. Implement parallel join and parallel sort algorithms to get the marks from different colleges of the university and publish 10 ranks for each discipline.

ACTIVE DATABASE:

6. Create triggers and assertions for Bank database handling deposits and loan and admission database handling seat allocation and vacancy position. Design the above relational database schema and implement the following triggers and assertions.
a. When a deposit is made by a customer, create a trigger for updating customers account and bank account
b. When a loan is issued to the customer, create a trigger for updating customer's loan account and bank account.
c. Create assertion for bank database so that the total loan amount does not exceed the total balance in the bank.
d. When an admission is made, create a trigger for updating the seat allocation details and vacancy position.

DEDUCTIVE DATABASE:

7. Construct a knowledge database for kinship domain (family relations) with facts. Extract the following relations using rules.
Parent, Sibling, Brother, Sister, Child, Daughter, Son, Spouse, Wife, husband, Grandparent, Grandchild, Cousin, Aunt and Uncle.

WEKA TOOL:

8. Work with Weka tool classification and clustering algorithms using the given training data and test with the unknown sample. Also experiment with different scenarios and large data set

RID	Age	Income	Student	Credit_ rating	Class: buys_ Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	Youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

QUERY PROCESSING

9. Implement Query Optimizer with Relational Algebraic expression construction and execution plan generation for choosing an efficient execution strategy for processing the given query.

Also design employee database and test the algorithm with following sample queries.

a) Select empid, empname from employee where experience > 5

b) Find all managers working at London Branch

XML

10. Design XML Schema for the given company database

Department (deptName, deptNo, deptManagerSSN, deptManagerStartDate, deptLocation)

Employee (empName, empSSN, empSex, empSalary, empBirthDate, empDeptNo, empSupervisorSSN, empAddress, empWorksOn)

Project (projName, projNo, projLocation, projDeptNo, projWorker)

a. Implement the following queries using XQuery and XPath

i. Retrieve the department name, manager name, and manager salary for every department'

ii. Retrieve the employee name, supervisor name and employee salary for each employee who works in the Research Department.

iii. Retrieve the project name, controlling department name, number of employees and total hours worked per week on the project for each project.

iv. Retrieve the project name, controlling department name, number of employees and total hours worked per week on the project for each project with more than one employee working on it

b. Implement a storage structure for storing XML database and test with the above schema.

OUTCOMES:

- Work on distributed databases
- Create and work on object oriented databases
- Create and work with parallel database
- Experiment on active database
- Explore the features of deductive database
- To work on weka tool for clustering and classification
- Represent the database using XML and work on it

213CSPP02 - CASE STUDY – OPERATING SYSTEMS DESIGN (Team Work)

OBJECTIVES:

1. To develop capabilities to work at systems level
2. To learn about issues in designing and implementing modern operating systems
3. To understand team formation, team issues, and allocating roles and responsibilities
4. To make effective presentations on the work done
5. To develop effective written communication skills

LAB EXERCISES:

A team of three or four students will work on assigned case study / mini-project. Case Study / Mini-project can be designed on the following lines:

1. Development of a reasonably sized dynamically loadable kernel module for Linux kernel
2. Study educational operating systems such as Minix (http://www.minix3.org/), Weenix (http://weenix.cs.brown.edu/mediawiki/index.php/Weenix) and develop reasonably sized interesting modules for them
3. Study the Android open source operating system for mobile devices (http://source.android.com/) and develop / modify some modules.
4. Study any embedded and real-time operating system such as eCos (http://ecos.sourceware.org/) and develop / modify some modules.

OUTCOMES:

Upon completion of the course, the students will be able to

- Develop assigned modules of operating systems design carrying out coding, testing, and documentation work involved.
- Describe team issuses and apply suitable methods to resolve the same.
- Demonstrate individual competence in building medium size operating system components.
- Demonstrate ethical and professional attributes of a computer engineer.
- Prepare suitable plan with clear statements of deliverables, and track the same.
- Make individual presentation of the work carried out.
- Prepare well-organized written documents to communicate individual work accomplished.

REFERENCES:

1. Watts S. Humphrey, "Introduction to Team Software Process", Addison-Wesley, SEI Series in Software Engineering, 1999.
2. Mukesh Singhal and Niranjan G. Shivaratri, "Advanced Concepts in Operating Systems – Distributed, Database, and Multiprocessor Operating Systems", Tata McGraw-Hill, 2001.
3. T. W. Doeppner, "Operating Systems in Depth: Design and Programming", Wiley, 2010.
4. S. Tanenbaum and A. S. Woodhull, "Operating Systems Design and Implementation", Third Edition, Prentice Hall, 2006.
5. Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, "Operating System Concepts", Ninth Edition, John Wiley & Sons, 2012.
6. Daniel P. Bovet and Marco Cesati, "Understanding the Linux kernel", 3rd edition, O'Reilly, 2005.
7. Rajib Mall, "Real-Time Systems: Theory and Practice", Pearson Education India, 2006.

313CSPT01 - SOFTWARE PROCESS AND PROJECT MANAGEMENT

OBJECTIVES:

1. To understand overall SDLC and adopt suitable processes
2. To elicite, analyze, prioritize, and manage both functional and quality requirements
3. To estimate efforts required, plan, and track the plans
4. To understand and apply configuration and quality management techniques
5. To evaluate, manage, and design processes

(A mini-project can be chosen by the instructor and use it as a context for the tutorials)

UNIT I DEVELOPMENT LIFE CYCLE PROCESSES

Overview of software development life cycle – introduction to processes – Personal Software Process (PSP) – Team software process (TSP) – Unified processes – agile processes – choosing the right process Tutorial: Software development using PSP

UNIT II REQUIREMENTS MANAGEMENT

Functional requirements and quality attributes – elicitation techniques – Quality Attribute Workshops (QAW) – analysis, prioritization, and trade-off – Architecture Centric Development Method (ACDM) – requirements documentation and specification – change management – traceability of requirements
Tutorial: Conduct QAW, elicit, analyze, prioritize, and document requirements using ACDM

UNIT III ESTIMATION, PLANNING, AND TRACKING

Identifying and prioritizing risks – risk mitigation plans – estimation techniques – use case points – function points – COCOMO II – top-down estimation – bottom-up estimation – work breakdown structure – macro and micro plans – planning poker – wideband delphi – documenting the plan – tracking the plan – earned value method (EVM)
Tutorial: Estimation, planning, and tracking exercises

UNIT IV CONFIGURATION AND QUALITY MANAGEMENT

identifying artifacts to be configured – naming conventions and version control – configuration control – quality assurance techniques – peer reviews – Fegan inspection – unit, integration, system, and acceptance testing – test data and test cases – bug tracking – causal analysis
Tutorial: version control exercises, development of test cases, causal analysis of defects

UNIT V SOFTWARE PROCESS DEFINITION AND MANAGEMENT

Process elements – process architecture – relationship between elements – process modeling – process definition techniques – ETVX (entry-task-validation-exit) – process baselining – process assessment and improvement – CMMI – Six Sigma
Tutorial: process measurement exercises, process definition using ETVX

OUTCOMES:

Upon Completion of the course, ☐the students will be able to

1. Explain software development life cycle
2. Adopt a suitable process for software development
3. Elicit functional and quality requirements
4. Analyze, prioritize, and manage requirements
5. Perform trade-off among conflicting requirements
6. Identify and prioritize risks and create mitigation plans
7. Estimate the efforts required for software development
8. Perform planning and tracking activities
9. Control the artifacts during software development
10. Perform various tests to ensure quality
11. Define new processes based on the needs
12. Adopt best practices for process improvement

REFERENCES:

1. Pankaj Jalote, "Software Project Management in Practice", Pearson, 2002.
2. Chris F. Kemerer, "Software Project Management – Readings and Cases", McGraw Hill, 1997.
3. Watts S. Humphrey, "PSP: A self-improvement process for software engineers", Addison-Wesley, 2005.
4. Watts S. Humphrey, "Introduction to the Team Software Process", Addison-Wesley, 2000.
5. Orit Hazzan and Yael Dubinsky, "Agile software engineering", Springer, 2008.
6. James R. Persse, "Process Improvement Essentials", O'Reilly, 2006.
7. Roger S. Pressman, "Software Engineering – A Practitioner's Approach", Seventh Edition, McGraw Hill, 2010.

113CSPT05 - FORMAL MODELS OF SOFTWARE SYSTEMS

OBJECTIVES:

- To understand the basic elements of Z
- To understand relations, functions, and logical structures in Z
- To understand Z schemas and schema calculus
- To learn selected Z case studies
- To understand Z schema refinement

UNIT I FOUNDATIONS OF Z

Understanding formal methods – motivation for formal methods – informal requirements to formal specifications – validating formal specifications – Overview of Z specification – basic elements of Z – sets and types – declarations – variables – expressions – operators – predicates and equations

UNIT II STRUCTURES IN Z

Tuples and records – relations, tables, databases – pairs and binary relations – functions – sequences – propositional logic in Z – predicate logic in Z – Z and boolean types – set comprehension – lambda calculus in Z – simple formal specifications – modeling systems and change

UNIT III Z SCHEMAS AND SCHEMA CALCULUS

Z schemas – schema calculus – schema conjunction and disjunction – other schema calculus operators – schema types and bindings – generic definitions – free types – formal reasoning – checking specifications – precondition calculation – machine-checked proofs

UNIT IV Z CASE STUDIES

Case Study: Text processing system – Case Study: Eight Queens – Case Study: Graphical User Interface – Case Study: Safety critical protection system – Case Study: Concurrency and real time systems

UNIT V Z REFINEMENT

Refinement of Z specification – generalizing refinements – refinement strategies – program derivation and verification – refinement calculus – data structures – state schemas – functions and relations – operation schemas – schema expressions – refinement case study

OUTCOMES:

Upon Completion of the course,□the students will be able to

- Apply the basic elements of Z
- Develop relational, functional, and logical Z structures
- Develop Z schema as models of software systems
- Perform verifications and conduct proofs using Z models
- Refine Z models towards implementing software systems

REFERENCES:

1. Jonathan Jacky, "The way of Z: Practical programming with formal methods", Cambridge University Press, 1996.
2. Antoni Diller, "Z: An introduction to formal methods", Second Edition, Wiley, 1994.
3. Jim Woodcock and Jim Davies, "Using Z – Specification, Refinement, and Proof", Prentice Hall, 1996.
4. J. M. Spivey, "The Z notation: A reference manual", Second Edition, Prentice Hall, 1992.
5. M. Ben-Ari, "Mathematical logic for computer science", Second Edition, Springer, 2003.
6. M. Huth and M. Ryan, "Logic in Computer Science – Modeling and Reasoning about systems", Second Edition, Cambridge University Press, 2004.

113CSPT06 - PERFORMANCE EVALUATION OF COMPUTER SYSTEMS

OBJECTIVES:

- To understand the mathematical foundations needed for performance evaluation of computer systems
- To understand the metrics used for performance evaluation
- To understand the analytical modeling of computer systems
- To enable the students to develop new queueing analysis for both simple and complex systems
- To appreciate the use of smart scheduling and introduce the students to analytical techniques for evaluating scheduling policies.

UNIT I OVERVIEW OF PERFORMANCE EVALUATION

Need for Performance Evaluation in Computer Systems – Overview of Performance Evaluation Methods – Introduction to Queueing – Probability Review – Generating Random Variables for Simulation – Sample Paths, Convergence and Averages – Little's Law and other Operational Laws – Modification for Closed Systems.

UNIT II MARKOV CHAINS AND SIMPLE QUEUES

Discrete-Time Markov Chains – Ergodicity Theory – Real World Examples – Google, Aloha – Transition to Continuous-Time Markov Chain – M/M/1 and PASTA.

UNIT III MULTI-SERVER AND MULTI-QUEUE SYSTEMS

Server Farms: M/M/k and M/M/k/k – Capacity Provisioning for Server Farms – Time Reversibility and Burke's Theorem – Networks of Queues and Jackson Product Form – Classed and Closed Networks of Queues.

UNIT IV REAL-WORLD WORKLOADS

Case Study of Real-world Workloads – Phase-Type Distributions and Matrix-Alalytic Methods – Networks with Time-Sharing Servers – M/G/1 Queue and the Inspection Paradox – Task Assignment Policies for Server Farms.

UNIT V SMART SCHEDULING IN THE M/G/1

Performance Metrics – Scheduling Non-Preemptive and Preemptive Non-Size-Based Policies - . Scheduling Non-Preemptive and Preemptive Size-Based Policies – Scheduling - SRPT and Fairness.

OUTCOMES:

Upon completion of the course, ☐the students will be able to

- Identify the need for performance evaluation and the metrics used for it
- Discuss open and closed queueing networks
- Define Little'e law and other operational laws
- Apply the operational laws to open and closed systems
- Use discrete-time and continuous-time Markov chains to model real world systems
- Develop analytical techniques for evaluating scheduling policies

REFERENCES:

1. Mor Harchol - Balter, "Performance Modeling and Design of Computer Systems – Queueing Theory in Action", Cambridge University Press, 2013.
2. Raj Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling", Wiley-Interscience, 1991.
3. Lieven Eeckhout, "Computer Architecture Performance Evaluation Methods", Morgan and Claypool Publishers, 2010.
4. Paul J. Fortier and Howard E. Michel, "Computer Systems Performance Evaluation and Prediction", Elsevier, 2003.
5. David J. Lilja, "Measuring Computer Performance: A Practitioner's Guide", Cambridge University Press, 2000.
6. Krishna Kant, "Introduction to Computer System Performance Evaluation", McGraw-Hill,1992.
7. K. S. Trivedi, "Probability and Statistics with Reliability, Queueing and Computer Science Applications", John Wiley and Sons, 2001.

113CSPT07 - PROBABILISTIC REASONING SYSTEMS

OBJECTIVES:

- To construct and reason with Bayesian networks
- To reason with temporal models
- To make exact and approximate inferences with graphical models
- To understand learning of parameters for probabilistic graphical models
- To understand actions and decisions with probabilistic graphical models

UNIT I REPRESENTATION

Probability Theory, Graphs, Bayesian network representation: Bayes networks, Independence in graphs – Undirected graphical models: Parameterization, Markov Network independencies – Conditional Bayesian networks.

UNIT II TEMPLATE BASED REPRESENTATION

Temporal models (Dynamic Bayesian networks , Hidden Markov Models) – Directed probabilistic models for object-relational domains – Inference in temporal models: Kalman filters.

UNIT III INFERENCE

Exact inference: Variable elimination – Exact inference: Clique trees (Junction trees) – Approximate inference: Forward sampling, Importance sampling, MCMC – MAP inference: Variable elimination for MAP, Max-product in clique trees.

UNIT IV LEARNING

Learning graphical models – Parameter estimation: maximum-likelihood estimation, MLE for Bayesian networks, Bayesian parameter estimation – Structure learning in Bayesian networks: Constraint based, structure scores, structure search – Partially observed data: Parameter estimation, Learning models with hidden variables – Learning undirected models: Maximum likelihood

UNIT V ACTIONS AND DECISIONS

Causality – Utilities and decisions – Structured decision problems

OUTCOMES:

Upon Completion of the course,□the students will be able to

- Construct Bayesian networks
- Reason with Bayesian networks
- Reason with Dynamic networks and Hidden Markov Models
- Conduct inferences with Bayesian networks
- Implement algorithms to learn probabilistic graphical models
- Explain actions and decisions with probabilistic graphical models

REFERENCES:

1. Daphne Koller and Nir Friedman, "Probabilistic Graphical Models: Principles and Techniques", MIT Press, 2009.
2. David Barber, "Bayesian Reasoning and Machine Learning", Cambridge University Press, 2012.
3. Adnan Darwiche, "Modeling and Reasoning with Bayesian networks", Cambridge University Press, 2009.
4. Kevin P. Murphy, "Machine Learning: A Probabilistic Perspective", MIT Press, 2012.
5. Stuart Russel and Peter Norvig, "Artificial Intelligence: A Modern Approach", Third Edition, Prentice Hall, 2009.

113CSPT08 - DATA ANALYSIS AND BUSINESS INTELLIGENCE

OBJECTIVES:

- To understand linear regression models
- To understand logistic regression models
- To understand generalized linear models
- To understand simulation using regression models
- To understand causal inference
- To understand multilevel regression
- To understand data collection and model understanding

UNIT I LINEAR REGRESSION

Introduction to data analysis – Statistical processes – statistical models – statistical inference – review of random variables and probability distributions – linear regression – one predictor – multiple predictors – prediction and validation – linear transformations – centering and standardizing – correlation – logarithmic transformations – other transformations – building regression models – fitting a series of regressions

UNIT II LOGISTIC AND GENERALIZED LINEAR MODELS

Logistic regression – logistic regression coefficients – latent-data formulation – building a logistic regression model – logistic regression with interactions – evaluating, checking, and comparing fitted logistic regressions – identifiability and separation – Poisson regression – logistic-binomial model – Probit regression – multinomial regression – robust regression using *t* model – building complex generalized linear models – constructive choice models

UNIT III SIMULATION AND CAUSAL INFERENCE

Simulation of probability models – summarizing linear regressions – simulation of non-linear predictions – predictive simulation for generalized linear models – fake-data simulation – simulating and comparing to actual data – predictive simulation to check the fit of a timeseries model – causal inference – randomized experiments – observational studies – causal inference using advanced models – matching – instrumental variables

UNIT IV MULTILEVEL REGRESSION

Multilevel structures – clustered data – multilevel linear models – partial pooling – grouplevel predictors – model building and statistical significance – varying intercepts and slopes – scaled inverse-Wishart distribution – non-nested models – multi-level logistic regression – multi-level generalized linear models

UNIT V DATA COLLECTION AND MODEL UNDERSTANDING

Design of data collection – classical power calculations – multilevel power calculations – power calculation using fake-data simulation – understanding and summarizing fitted models – uncertainty and variability – variances – R2 and explained variance – multiple comparisons and statistical significance – analysis of variance – ANOVA and multilevel linear and general linear models – missing data imputation

OUTCOMES:

Upon Completion of the course,□the students will be able to

- Build and apply linear regression models
- Build and apply logistic regression models
- Build and apply generalized linear models
- Perform simulation using regression models
- Perform casual inference from data
- Build and apply multilevel regression models
- Perform data collection and variance analysis

REFERENCES:

1. Andrew Gelman and Jennifer Hill, "Data Analysis using Regression and multilevel/Hierarchical Models", Cambridge University Press, 2006.
2. Philipp K. Janert, "Data Analysis with Open Source Tools", O'Reilley, 2010.
3. Wes McKinney, "Python for Data Analysis", O'Reilley, 2012.
4. Davinderjit Sivia and John Skilling, "Data Analysis: A Bayesian Tutorial", Second Edition, Oxford University Press, 2006.
5. Robert Nisbelt, John Elder, and Gary Miner, "Handbook of statistical analysis and data mining applications", Academic Press, 2009.
6. Michael Minelli, Michelle Chambers, and Ambiga Dhiraj, "Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses", Wiley, 2013.
7. John Maindonald and W. John Braun, "Data Analysis and Graphics Using R: An Example-based Approach", Third Edition, Cambridge University Press, 2010.
8. David Ruppert, "Statistics and Data Analysis for Financial Engineering", Springer, 2011.

OBJECTIVES:

- To understand the basics of digital images
- To understand noise models
- To understand spatial domain filters
- To understand frequency domain filters
- To learn basic image analysis --- segmentation, edge detection, and corner detection
- To learn morphological operations and texture analysis
- To understand processing of color images
- To understand image compression techniques

UNIT I SPATIAL DOMAIN PROCESSING

Introduction to image processing – imaging modalities – image file formats – image sensing and acquisition – image sampling and quantization – noise models – spatial filtering operations – histograms – smoothing filters – sharpening filters – fuzzy techniques for spatial filtering – spatial filters for noise removal

UNIT II FREQUENCY DOMAIN PROCESSING

Frequency domain – Review of Fourier Transform (FT), Discrete Fourier Transform (DFT), and Fast Fourier Transform (FFT) – filtering in frequency domain – image smoothing – image sharpening – selective filtering – frequency domain noise filters – wavelets – Haar Transform – multiresolution expansions – wavelet transforms – wavelets based image Processing

UNIT III SEGMENTATION AND EDGE DETECTION

Thresholding techniques – region growing methods – region splitting and merging – adaptive thresholding – threshold selection – global valley – histogram concavity – edge detection – template matching – gradient operators – circular operators – differential edge operators – hysteresis thresholding – Canny operator – Laplacian operator – active contours – object segmentation

UNIT IV INTEREST POINTS, MORPHOLOGY, AND TEXTURE

Corner and interest point detection – template matching – second order derivatives – median filter based detection – Harris interest point operator – corner orientation – local invariant feature detectors and descriptors – morphology – dilation and erosion – morphological operators – grayscale morphology – noise and morphology – texture – texture analysis – co-occurrence matrices – Laws' texture energy approach – Ade's eigen filter approach

UNIT V COLOR IMAGES AND IMAGE COMPRESSION

Color models – pseudo colors – full-color image processing – color transformations – smoothing and sharpening of color images – image segmentation based on color – noise in color images. Image Compression – redundancy in images – coding redundancy – irrelevant information in images – image compression models – basic compression methods – digital image watermarking.

OUTCOMES:

Upon completion of the course, the students will be able to

- Explain image modalities, sensing, acquisition, sampling, and quantization
- Explain image noise models
- Implement spatial filter operations
- Explain frequency domain transformations
- Implement frequency domain filters
- Apply segmentation algorithms
- Apply edge detection techniques
- Apply corner and interest point detection algorithms
- Apply morphological operations
- Perform texture analysis
- Analyze color images
- Implement image compression algorithms

REFERENCES:

1. E. R. Davies, "Computer & Machine Vision", Fourth Edition, Academic Press, 2012.
2. W. Burger and M. Burge, "Digital Image Processing: An Algorithmic Introduction using Java", Springer, 2008.
3. John C. Russ, "The Image Processing Handbook", Sixth Edition, CRC Press, 2011.
4. R. C. Gonzalez and R. E. Woods, "Digital Image Processing", Third Edition, Pearson, 2008.
5. Mark Nixon and Alberto S. Aquado, "Feature Extraction & Image Processing for Computer Vision", Third Edition, Academic Press, 2012.
6. D. L. Baggio et al., "Mastering OpenCV with Practical Computer Vision Projects", Packt Publishing, 2012.
7. Jan Erik Solem, "Programming Computer Vision with Python: Tools and algorithms for analyzing images", O'Reilly Media, 2012.

OBJECTIVES:

- To study the sensor characteristics and the fundamental principles of sensing
- To understand the sensor interface electronics
- To study selected motion-related sensors
- To study light and radiation detectors
- To study selected temperature sensors
- To study selected chemical sensors

UNIT I PRINCIPLES OF SENSING

Data Acquisition – sensor characteristics – electric charges, fields, potentials – capacitance – magnetism – inductance – resistance – piezoelectric – pyroelectric – Hall effect – thermoelectric effects – sound waves – heat transfer – light – dynamic models of sensors

UNIT II OPTICAL COMPONENTS AND INTERFACE ELECTRONICS

Radiometry – Photometry – mirrors – lenses – fibre optics – concentrators – Interface circuits – amplifiers – light-to-voltage – excitation circuits – ADC – Digitization – Capacitance-to-voltage – bridge circuits – data transmission – noise in sensors and circuits – calibration – low power sensors

UNIT III MOTION RELATED SENSORS

Occupancy and motion detectors: ultrasonic – microwave – capacitive detectors – triboelectric – optoelectronic motion sensors – optical presence sensor – Pressure Gradient sensorsVelocity and acceleration sensors: Accelerometer characteristics – capacitative accelerometers – piezoelectric accelerometers – piezoresistive accelerometers – thermal accelerometers – Gyroscopes – piezoelectric cables – gravitational sensors

UNIT IV LIGHT AND RADIATION DETECTORS

Light Detectors: Photo diodes – photo transistor – photo resistor – cooled detectors – CCD and CMOS image sensors – thermal detectors – optical design – gas flame detectors Radiation Detectors: scintillating detectors – ionization detectors – cloud and bubble chambers

UNIT V TEMPERATURE AND CHEMICAL SENSORS

Temperature Sensors: coupling with objects – temperature reference points – thermo resistive sensors – thermo electric contact sensors – semiconductor sensors – acoustic sensors – piezoelectric sensors Chemical sensors: characteristics – classes of chemical sensors – biochemical sensors – multi-sensor arrays – electronic noses and tongues

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Explain sensor characteristics
- Explain the physics of sensors
- Explain optical components of sensors
- Apply sensor interface electronics
- Choose and use appropriate motion-related sensors
- Choose and use appropriate light and radiation detectors
- Choose and use appropriate temperature sensors
- Choose and use appropriate chemical sensors

REFERENCE:

1. Jacob Fraden, "Handbook of Modern Sensors: Physics, Designs, and Applications", Fourth Edition, Springer, 2010.

113CSPT11 - RANDOMIZED ALGORITHMS

OBJECTIVES:

- To understand the mathematical foundations needed for understanding and designing randomized algorithms
- To appreciate the need for randomized algorithms
- To expose the students to probabilistic methods
- To understand the concept of random walk
- To expose the students to different types of applications of randomized algorithms

UNIT I INTRODUCTION TO RANDOMIZED ALGORITHMS

Introduction to Randomized Algorithms - Min-cut – Elementary Probability Theory – Models of Randomized Algorithms – Classification of Randomized Algorithms – Paradigms of the Design of Randomized Algorithms - Game Theoretic Techniques – Game Tree Evaluation – Minimax Principle – Randomness and Non Uniformity.

UNIT II PROBABILISTIC METHODS

Moments and Deviations – occupancy Problems – Markov and Chebyshev Inequalities – Randomized Selection – Two Point Sampling – The Stable Marriage Problem – The Probabilistic Method – Maximum Satisfiability – Expanding Graphs – Method of Conditional Probabilities – Markov Chains and Random Walks – 2-SAT Example – Random Walks on Graphs – Random Connectivity.

UNIT III ALGEBRAIC TECHNIQUES AND APPLICATIONS

Fingerprinting Techniques – Verifying Polynomial Identities – Perfect Matching in Graphs – Pattern Matching – Verification of Matrix Multiplication - Data Structuring Problems – Random Treaps – Skip Lists – Hash Tables.

UNIT IV GEOMETRIC AND GRAPH ALGORITHMS

Randomized Incremental Construction – Convex Hulls – Duality – Trapezoidal Decompositions – Linear Programming – Graph Algorithms – Min-cut – Minimum Spanning Trees.

UNIT V HASHING AND ONLINE ALGORITHMS

Hashing – Universal Hashing - Online Algorithms – Randomized Online Algorithms - Online Paging – Adversary Models – Relating the Adversaries – The k-server Problem.

OUTCOMES:

Upon completion of the course, ☐the students will be able to

- Identify the need for randomized algorithms
- Discuss the classification of randomized algorithms
- Present the various paradigms for designing randomized algorithms
- Discuss the different probabilistic methods used for designing randomized algorithms
- Apply the techniques studied to design algorithms for different applications like matrix multiplication, hashing, linear programming

REFERENCES:

1. Rajeev Motwani and Prabhakar Raghavan, "Randomized Algorithms", Cambridge University Press, 1995.
2. Juraj Hromkovic,"Design and Analysis of Randomized Algorithms", Springer, 2010.
3. Michael Mitzenmacher and Eli Upfal, "Probabilty and Computing – Randomized Algorithms and Probabilistic Analysis", Cambridge University Press, 2005.

113CSPT12 - MOBILE AND PERVASIVE COMPUTING

OBJECTIVES :

- To understand the basics of Mobile Computing and Personal Computing
- To learn the role of cellular networks in Mobile and Pervasive Computing
- To expose to the concept of sensor and mesh networks
- To expose to the context aware and wearable computing
- To learn to develop applications in mobile and pervasive computing environment

UNIT I INTRODUCTION

Differences between Mobile Communication and Mobile Computing – Contexts and Names – Functions – Applications and Services – New Applications – Making Legacy Applications Mobile Enabled – Design Considerations – Integration of Wireless and Wired Networks – Standards Bodies – Pervasive Computing – Basics and Vision – Principles of Pervasive Computing – Categories of Pervasive Devices

UNIT II 3G AND 4G CELLULAR NETWORKS

Migration to 3G Networks – IMT 2000 and UMTS – UMTS Architecture – User Equipment – Radio Network Subsystem – UTRAN – Node B – RNC functions – USIM – Protocol Stack – CS and PS Domains – IMS Architecture – Handover – 3.5G and 3.9G a brief discussion – 4G LAN and Cellular Networks – LTE – Control Plane – NAS and RRC – User Plane – PDCP, RLC and MAC – WiMax IEEE 802.16d/e – WiMax Internetworking with 3GPP

UNIT III SENSOR AND MESH NETWORKS

Sensor Networks – Role in Pervasive Computing – In Network Processing and Data Dissemination – Sensor Databases – Data Management in Wireless Mobile Environments – Wireless Mesh Networks – Architecture – Mesh Routers – Mesh Clients – Routing – Cross Layer Approach – Security Aspects of Various Layers in WMN – Applications of Sensor and Mesh networks

UNIT IV CONTEXT AWARE COMPUTING & WEARABLE COMPUTING

Adaptability – Mechanisms for Adaptation - Functionality and Data – Transcoding – Location Aware Computing – Location Representation – Localization Techniques – Triangulation and Scene Analysis – Delaunay Triangulation and Voronoi graphs – Types of Context – Role of Mobile Middleware – Adaptation and Agents – Service Discovery Middleware Health BAN- Medical and Technological Requirements-Wearable Sensors-Intra-BAN Communications

UNIT V APPLICATION DEVELOPMENT

Three tier architecture - Model View Controller Architecture - Memory Management – Information Access Devices – PDAs and Smart Phones – Smart Cards and Embedded Controls – J2ME – Programming for CLDC – GUI in MIDP – Application Development ON Android and iPhone

OUTCOMES:

At the end of the course the student should be able to

- Design a basic architecture for a pervasive computing environment
- Design and allocate the resources on the 3G-4G wireless networks
- Analyze the role of sensors in Wireless networks
- Work out the routing in mesh network
- Deploy the location and context information for application development
- Develop mobile computing applications based on the paradigm of context aware computing and wearable computing

REFERENCES:

1. Asoke K Talukder, Hasan Ahmed, Roopa R Yavagal, "Mobile Computing: Technology, Applications and Service Creation", 2nd ed, Tata McGraw Hill, 2010.
2. Reto Meier, "Professional Android 2 Application Development", Wrox Wiley,2010.
3. .Pei Zheng and Lionel M Li, 'Smart Phone & Next Generation Mobile Computing', Morgan Kaufmann Publishers, 2006.
4. Frank Adelstein, 'Fundamentals of Mobile and Pervasive Computing', TMH, 2005
5. Jochen Burthardt et al, 'Pervasive Computing: Technology and Architecture of Mobile Internet Applications', Pearson Education, 2003
6. Feng Zhao and Leonidas Guibas, 'Wireless Sensor Networks', Morgan Kaufmann Publishers, 2004
7. Uwe Hansmaan et al, 'Principles of Mobile Computing', Springer, 2003
8. Reto Meier, "Professional Android 2 Application Development", Wrox Wiley,2010.
9. Mohammad s. Obaidat et al, "Pervasive Computing and Networking",John wiley
10. Stefan Poslad, "Ubiquitous Computing: Smart Devices, Environments and Interactions", Wiley, 2009
11. Frank Adelstein Sandeep K. S. Gupta Golden G. Richard III Loren Schwiebert "Fundamentals of Mobile and Pervasive Computing, ", McGraw-Hill, 2005

OBJECTIVES:

- To understand models of and issues in concurrency in computing
- To develop message-passing parallel programs using MPI
- To develop shared-memory parallel programs using Pthreads
- To develop shared-memory parallel programs using OpenMP
- To use GPU for parallel programming using OpenCL and CUDA

UNIT I FOUNDATIONS OF PARALLEL PROGRAMMING

Motivation for parallel programming - Concurrency in computing – basics of processes, multiprocessing, and threads – cache – cache mappings – caches and programs – virtual memory – instruction level parallelism – hardware multi-threading – SIMD – MIMD – interconnection networks – cache coherence – shared-memory model – issues in sharedmemory model – distributed-memory model – issues in distributed-memory model – hybrid model – I/O – performance of parallel programs – parallel program design

UNIT II MESSAGE PASSING PARADIGM

Basic MPI programming – MPI_Init and MPI_Finalize – MPI communicators – SPMD programs – message passing – MPI_Send and MPI_Recv – message matching – MPI I/O – parallel I/O – collective communication – MPI_Reduce – MPI_Allreduce – broadcast – scatter – gather – allgather – derived types – remote memory access – dynamic process management – MPI for grids – performance evaluation of MPI programs

UNIT III SHARED MEMORY PARADIGM: PTHREADS

Basics of Pthreads – thread synchronization – critical sections – busy-waiting – mutexes – semaphores – barriers and condition variables – read-write locks – Caches, cache coherence and false sharing – thread safety – Pthreads case study

UNIT IV SHARED MEMORY PARADIGM: OPENMP

Basic OpenMP constructs – scope of variabls – reduction clause – parallel for directive – loops in OpenMP – scheduling loops – synchronization in OpenMP – Case Study: Producer-Consumer problem – cache issues – threads safety in OpenMP – OpenMP best Practices

UNIT V GRAPHICAL PROCESSING PARADIGMS: OPENCL AND CUDA

Introduction to CUDA – CUDA programming examples – CUDA execution model – CUDA memory hierarchy – CUDA case study - introduction to OpenCL – OpenCL programming examples – Programs and Kernels – Buffers and Images – Event model – OpenCL case study

OUTCOMES:

Upon completion of the course, the students will be able to

- Explain models of parallel programming
- Explain hardware level support for concurrency
- Explain issues in parallel programming
- Develop message-passing parallel programs using MPI framework
- Develop shared-memory parallel programs using Pthreads
- Develop shared-memory parallel programs using OpenMP
- Develop CUDA programs
- Develop OpenCL programs

REFERENCES:

1. Peter S. Pacheco, “An introduction to parallel programming”, Morgan Kaufmann, 2011.
2. M. J. Quinn, “Parallel programming in C with MPI and OpenMP”, Tata McGraw Hill, 2003.
3. W. Gropp, E. Lusk, and R. Thakur, “Using MPI-2: Advanced features of the message passing interface”, MIT Press, 1999.
4. W. Gropp, E. Lusk, and A. Skjellum, “Using MPI: Portable parallel programming with the message passing interface”, Second Edition, MIT Press, 1999.
5. B. Chapman, G. Jost, and Ruud van der Pas, “Using OpenMP”, MIT Press, 2008.
6. D. R. Butenhof, “Programming with POSIX Threads”, Addison Wesley, 1997.
7. B. Lewis and D. J. Berg, “Multithreaded programming with Pthreads”, Sun Microsystems Press, 1998.
8. A. Munshi, B. Gaster, T. G. Mattson, J. Fung, and D. Ginsburg, “OpenCL programming guide”, Addison Wesley, 2011.
9. Rob Farber, “CUDA application design and development”, Morgan Haufmann, 2011.

113CSPT14 - SOFTWARE REQUIREMENTS ENGINEERING

OBJECTIVES:

1. Understand system requirements
2. Identify different types of requirement
3. Generate requirements be elicitation
4. Develop requirements documentation
5. Evaluate the requirements

UNIT I DOMAIN UNDERSTANDING

Introduction – Types of requirements – Requirements engineering process – Validating requirements – Requirements and design – Requirements and test cases – introduction to business domain – Problem analysis – Fish bone diagram – Business requirements – Business process modeling – Business use cases – Business modeling notations – UML Activity diagrams.

UNIT II REQUIREMENTS ELICITATION

Introduction – Understanding stakeholders' needs – Elicitation techniques – interviews, questionnaire, workshop, brainstorming, prototyping – Documenting stakeholders' needs

UNIT III FUNCTIONAL REQUIREMENTS

Introduction – Features and Use cases – Use case scenarios – Documenting use cases – Levels of details – SRS documents.

UNIT IV QUALITY ATTRIBUTES AND USER EXPERIENCE

Quality of solution – Quality attributes – Eliciting quality attributes – Quality attribute workshop (QAW) – Documenting quality attributes – Six part scenarios – Usability requirements – Eliciting and documenting usability requirements – Modeling user experience – Specifying UI design

UNIT V MANAGING REQUIREMENTS

Defining scope of the project – Context diagram – Managing requirements – Requirements properties – Traceability – Managing changes – Requirements metrics – Requirements management tools.

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Define a process for requirments engineering
- Execute a process for gathering requirments through elicitation techniques.
- Validate requirements according to criteria such as feasibility, clarity, preciseness etc.
- Develop and document functional requirements for different types of systems.
- Develop and document quality attributes of the system to be implemented
- Communicate the requirments to stakeholders
- Negotiate with stakeholders in order to agree on a set of requirements.
- Detect and resolve feature interactions

REFERENCES:

1. Axel van Lamsweerde, "Requirements Engineering", Wiley, 2009
2. Gerald Kotonya, Ian Sommerville, "Requirements Engineering: Processes and Techniques", John Wiley and Sons, 1998
3. Dean Leffingwell and Don Widrig, "Managing Software Requirements: A Use Case Approach (2nd Edition) ", Addison-wesley, 2003
4. SEI Report, "Quality Attributes Workshop", http://www.sei.cmu.edu/library/abstracts/reports/03tr016.cfm , 2003
5. J Nielsen, "Usability Engineering", Academic Press, 1993

113CSPT15 - SPEECH PROCESSING AND SYNTHESIS

OBJECTIVES:

- To understand the mathematical foundations needed for speech processing
- To understand the basic concepts and algorithms of speech processing and synthesis
- To familiarize the students with the various speech signal representation, coding and recognition techniques
- To appreciate the use of speech processing in current technologies and to expose the students to real– world applications of speech processing

UNIT I FUNDAMENTALS OF SPEECH PROCESSING

Introduction – Spoken Language Structure – Phonetics and Phonology – Syllables and Words – Syntax and Semantics – Probability, Statistics and Information Theory – Probability Theory – Estimation Theory – Significance Testing – Information Theory.

UNIT II SPEECH SIGNAL REPRESENTATIONS AND CODING

Overview of Digital Signal Processing – Speech Signal Representations – Short time Fourier Analysis – Acoustic Model of Speech Production – Linear Predictive Coding – Cepstral Processing – Formant Frequencies – The Role of Pitch – Speech Coding – LPC Coder.

UNITIII SPEECH RECOGNITION

Hidden Markov Models – Definition – Continuous and Discontinuous HMMs – Practical Issues – Limitations. Acoustic Modeling – Variability in the Speech Signal – Extracting Features – Phonetic Modeling – Adaptive Techniques – Confidence Measures – Other Techniques.

UNITIV TEXT ANALYSIS

Lexicon – Document Structure Detection – Text Normalization – Linguistic Analysis – Homograph Disambiguation – Morphological Analysis – Letter-to-sound Conversion – Prosody – Generation schematic – Speaking Style – Symbolic Prosody – Duration Assignment – Pitch Generation

UNIT V SPEECH SYNTHESIS

Attributes – Formant Speech Synthesis – Concatenative Speech Synthesis – Prosodic Modification of Speech – Source-filter Models for Prosody Modification – Evaluation of TTS Systems.

OUTCOMES:

Upon completion of the course, □the students will be able to

- Identify the various temporal, spectral and cepstral features required for identifying speech units – phoneme, syllable and word
- Determine and apply Mel-frequency cepstral coefficients for processing all types of signals
- Justify the use of formant and concatenative approaches to speech synthesis
- Identify the apt approach of speech synthesis depending on the language to be processed
- Determine the various encoding techniques for representing speech.

REFERENCES:

1. Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, "Spoken Language Processing – A guide to Theory, Algorithm and System Development", Prentice Hall PTR, 2001.
2. Thomas F.Quatieri, "Discrete-Time Speech Signal Processing", Pearson Education, 2002.
3. Lawrence Rabiner and Biing-Hwang Juang, "Fundamentals of Speech Recognition", Prentice Hall Signal Processing Series, 1993.
4. Sadaoki Furui, "Digital Speech Processing: Synthesis, and Recognition, Second Edition, (Signal Processing and Communications)", Marcel Dekker, 2000.
5. Joseph Mariani, "Language and Speech Processing", Wiley, 2009.

113CSPT16 - MACHINE LEARNING TECHNIQUES

OBJECTIVES:

1. To understand the machine learning theory
2. To implement linear and non-linear learning models
3. To implement distance-based clustering techniques
4. To build tree and rule based models
5. To apply reinforcement learning techniques

UNIT I FOUNDATIONS OF LEARNING

Components of learning – learning models – geometric models – probabilistic models – logic models – grouping and grading – learning versus design – types of learning – supervised – unsupervised – reinforcement – theory of learning – feasibility of learning – error and noise – training versus testing – theory of generalization – generalization bound – approximationgeneralization
tradeoff – bias and variance – learning curve

UNIT II LINEAR MODELS

Linear classification – univariate linear regression – multivariate linear regression – regularized regression – Logistic regression – perceptrons – multilayer neural networks – learning neural networks structures – support vector machines – soft margin SVM – going beyond linearity – generalization and overfitting – regularization – validation

UNIT III DISTANCE-BASED MODELS

Nearest neighbor models – K-means – clustering around medoids – silhouttes – hierarchical clustering – k-d trees – locality sensitive hashing – non-parametric regression – ensemble learning – bagging and random forests – boosting – meta learning

UNIT IV TREE AND RULE MODELS

Decision trees – learning decision trees – ranking and probability estimation trees – regression trees – clustering trees – learning ordered rule lists – learning unordered rule lists – descriptive rule learning – association rule mining – first-order rule learning

UNIT V REINFORCEMENT LEARNING

Passive reinforcement learning – direct utility estimation – adaptive dynamic programming – temporal-difference learning – active reinforcement learning – exploration – learning an action-utility function – Generalization in reinforcement learning – policy search – applications in game playing – applications in robot control

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- To explain theory underlying machine learning
- To construct algorithms to learn linear and non-linear models
- To implement data clustering algorithms
- To construct algorithms to learn tree and rule-based models
- To apply reinforcement learning techniques

REFERENCES:

1. Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, "Learning from Data", AMLBook Publishers, 2012.
2. P. Flach, "Machine Learning: The art and science of algorithms that make sense of data", Cambridge University Press, 2012.
3. K. P. Murphy, "Machine Learning: A probabilistic perspective", MIT Press, 2012.
4. C. M. Bishop, "Pattern Recognition and Machine Learning", Springer, 2007.
5. D. Barber, "Bayesian Reasoning and Machine Learning", Cambridge University Press, 2012.
6. M. Mohri, A. Rostamizadeh, and A. Talwalkar, "Foundations of Machine Learning", MIT Press, 2012.
7. T. M. Mitchell, "Machine Learning", McGraw Hill, 1997.
8. S. Russel and P. Norvig, "Artificial Intelligence: A Modern Approach", Third Edition, Prentice Hall, 2009.

213CSPT05 - CONCURRENCY MODELS

OBJECTIVES:

- To model concurrency in FSP
- To specify and check safety and liveness properties
- To understand concurrency architectures and design
- To apply linear temporal logic to safety and liveness analysis
- To apply Petri nets for concurrency modeling and analysis

UNIT I FSP AND GRAPH MODELS

Concurrency and issues in concurrency – models of concurrency – graphical models – FSP & LTSA – modeling processes with FSP – concurrency models with FSP – shared action – structure diagrams – issues with shared objects – modeling mutual exclusion – conditional synchronization – modeling semaphores – nested monitors – monitor invariants

UNIT II SAFETY AND LIVENESS PROPERTIES

Deadlocks – deadlock analysis in models – dining philosophers problem – safety properties – single-lane bridge problem – liveness properties – liveness of the single-lane bridge – readers-writers problem – message passing – asynchronous message passing models – synchronous message passing models – rendezvous

UNIT III CONCURRENCY ARCHITECTURES AND DESIGN

Modeling dynamic systems – modeling timed systems – concurrent architectures – Filter pipeline – Supervisor-worker model – announcer-listener model – model-based design – from requirements to models – from models to implementations – implementing concurrency in Java – program verification

UNIT IV LINEAR TEMPORAL LOGIC (LTL)

Syntax of LTL – semantics of LTL – practical LTL patterns – equivalences between LTL statements – specification using LTL – LTL and FSP – Fluent proposition – Temporal propositions – Fluent Linear Temporal Logic (FLTL) – FLTL assertions in FSP – Database ring problem

UNIT V PETRI NETS

Introduction to Petri nets – examples – place-transition nets – graphical and linear algebraic representations – concurrency & conflict – coverability graphs – decision procedures – liveness – colored Petri nets (CPN) – modeling & verification using CPN – non-hierarchical CPN – modeling protocols – hierarchical CPN – timed CPN – applications of Petri Nets

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Develop concurrency models and FSP
- State safety and liveness properties in FSP
- Verify properties using LTSA tool
- Explain concurrency architectures
- Design concurrent Java programs from models
- Apply Linear Temporal Logic to state safety and liveness properties
- Assert LTL properties in FSP and check using LTSA tool
- Model and analyze concurrency using Petri nets

REFERENCES:

1. Jeff Magee & Jeff Kramer, "Concurrency: State Models and Java Programs", Second Edition, John Wiley, 2006.
2. M. Huth & M. Ryan, "Logic in Computer Science – Modeling and Reasoning about Systems", Second Edition, Cambridge University Press, 2004.
3. B. Goetz, T. Peierls, J. Bloch, J. Bowbeer, D. Holmes, and D. Lea, "Java Concurrency in Practice", Addison-Wesley Professional, 2006.
4. Wolfgang Reisig, "Petri Nets: An Introduction", Springer, 2011.
5. K. Jensen and L. M. Kristensen, "Colored Petri Nets: Modeling and Validation of Concurrent Systems", Springer, 2009.
6. Wolfgang Reisig, "Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies", Springer, 2013.

213CSPT06 - REAL TIME SYSTEMS

OBJECTIVES:

- To provide good understanding of fundamental concepts in real time systems.
- To provide understanding of advanced topics in real time systems.
- To provide understanding on basic multi-task scheduling algorithms for periodic, aperiodic, and sporadic tasks as well as understand the impact of the latter two on scheduling
- To expose to understand capabilities of commercial off-the-shelf R-T kernel.
- To expose to real time communications and databases.

UNIT I INTRODUCTION

Real-time systems – Applications – Basic Model – Characteristics – Safety and Reliability – Real-Time tasks – Timing Constraints – Modelling Timing Constraints.

UNIT II SCHEDULING REAL-TIME TASKS

Concepts – Types of RT Tasks and their Characteristics – Task Scheduling – Clock-Driven Scheduling – Hybrid Schedulers - Event-Driven Scheduling – EDF Scheduling – RMA – Issues with RMA – Issues in Using RMA in Practical Situations

UNIT III RESOURCE SHARING AMONG RT TASKS & SCHEDULING RT TASKS

Resource Sharing Among RT Tasks – Priority Inversion – PIP – HLP – PCP – Types of Priority Inversions Under PCP – Features of PCP – Issues in using Resource Sharing Protocol – Handling Task Dependencies – Multiprocessor Task Allocation – Dynamic Allocation of Tasks – Fault-Tolerant Scheduling of Tasks – Clocks in Distributed RT Systems – Centralized and Distributed Clock Synchronization.

UNIT IV COMMERCIAL RT OPERATING SYSTEMS

Time Services – Features of RT OS – Unix as a RT OS – Unix Based RT OS – Windows as a RT OS – POSIX – Survey of RTOS: PSOS – VRTX – VxWorks – QNX - µC/OS-II – RT Linux – Lynx – Windows CE – Benching RT Systems.

UNIT V RT COMMUNICATION & DATABASES

Examples of Applications Requiring RT Communication – Basic Concepts – RT Communication in a LAN – Soft & Hard RT Communication in a LAN – Bounded Access Protocols for LANs – Performance Comparison – RT Communication Over Packet Switched Networks – QoS Framework – Routing – Resource Reservation – Rate Control – QoS Models - Examples Applications of RT Databases – RT Databases – Characteristics of Temporal Data – Concurrency Control in RT Databases – Commercial RT Databases.

OUTCOMES:

- Understand the basics and importance of real-time systems
- Generate a high-level analysis document based on requirements specifications
- Generate a high-level design document based on analysis documentation
- Generate a test plan based on requirements specification
- Generate a validation plan based on all documentation
- Understand basic multi-task scheduling algorithms for periodic, aperiodic, and sporadic tasks as well as understand the impact of the latter two on scheduling
- Understand capabilities of at least one commercial off-the-shelf R-T kernel

REFERENCES:

1. Rajib Mall, "Real-Time Systems: Theory and Practice," Pearson, 2008.
2. Jane W. Liu, "Real-Time Systems" Pearson Education, 2001.
3. Krishna and Shin, "Real-Time Systems," Tata McGraw Hill. 1999.
4. Alan C. Shaw, "Real-Time Systems and Software", Wiley, 2001.
5. Philip Laplante, "Real-Time Systems Design and Analysis", 2nd Edition, Prentice Hall of India.
6. Resource Management in Real-time Systems and Networks, C. Siva Ram Murthy and G. Manimaran, MIT Press, March 2001.

213CSPT07 - COMPUTER VISION

OBJECTIVES:

- To review image processing techniques for computer vision
- To understand shape and region analysis
- To understand Hough Transform and its applications to detect lines, circles, ellipses
- To understand three-dimensional image analysis techniques
- To understand motion analysis
- To study some applications of computer vision algorithms

UNIT I IMAGE PROCESSING FOUNDATIONS

Review of image processing techniques – classical filtering operations – thresholding techniques – edge detection techniques – corner and interest point detection – mathematical morphology – texture

UNIT II SHAPES AND REGIONS

Binary shape analysis – connectedness – object labeling and counting – size filtering – distance functions – skeletons and thinning – deformable shape analysis – boundary tracking procedures – active contours – shape models and shape recognition – centroidal profiles – handling occlusion – boundary length measures – boundary descriptors – chain codes – Fourier descriptors – region descriptors – moments

UNIT III HOUGH TRANSFORM

Line detection – Hough Transform (HT) for line detection – foot-of-normal method – line localization – line fitting – RANSAC for straight line detection – HT based circular object detection – accurate center location – speed problem – ellipse detection – Case study: Human Iris location – hole detection – generalized Hough Transform (GHT) – spatial matched filtering – GHT for ellipse detection – object location – GHT for feature collation

UNIT IV 3D VISION AND MOTION

Methods for 3D vision – projection schemes – shape from shading – photometric stereo – shape from texture – shape from focus – active range finding – surface representations – point-based representation – volumetric representations – 3D object recognition – 3D reconstruction – introduction to motion – triangulation – bundle adjustment – translational alignment – parametric motion – spline-based motion – optical flow – layered motion

UNIT V APPLICATIONS

Application: Photo album – Face detection – Face recognition – Eigen faces – Active appearance and 3D shape models of faces

Application: Surveillance – foreground-background separation – particle filters – Chamfer matching, tracking, and occlusion – combining views from multiple cameras – human gait analysis

Application: In-vehicle vision system: locating roadway – road markings – identifying road signs – locating pedestrians

OUTCOMES:

Upon completion of the course, the students will be able to

- Implement fundamental image processing techniques required for computer vision
- Perform shape analysis
- Implement boundary tracking techniques
- Apply chain codes and other region descriptors
- Apply Hough Transform for line, circle, and ellipse detections
- Apply 3D vision techniques
- Implement motion related techniques
- Develop applications using computer vision techniques

REFERENCES:

1. E. R. Davies, "Computer & Machine Vision", Fourth Edition, Academic Press, 2012.

2. R. Szeliski, “Computer Vision: Algorithms and Applications”, Springer 2011.
3. Simon J. D. Prince, “Computer Vision: Models, Learning, and Inference”, Cambridge University Press, 2012.
4. Mark Nixon and Alberto S. Aquado, “Feature Extraction & Image Processing for Computer Vision”, Third Edition, Academic Press, 2012.
5. D. L. Baggio et al., “Mastering OpenCV with Practical Computer Vision Projects”, Packt Publishing, 2012.
6. Jan Erik Solem, “Programming Computer Vision with Python: Tools and algorithms for analyzing images”, O'Reilly Media, 2012.

213CSPT08 - NETWORK AND INFORMATION SECURITY

OBJECTIVES:

- To understand the fundamentals of Cryptography
- To acquire knowledge on standard algorithms used to provide confidentiality, integrity and authenticity.
- To understand the various key distribution and management schemes.
- To understand how to deploy encryption techniques to secure data in transit acrossdata networks
- To design security applications in the field of Information technology

UNIT I INTRODUCTION

An Overview of Computer Security-Security Services-Security Mechanisms-Security Attacks-Access Control Matrix, Policy-Security policies, Confidentiality policies, Integrity policies and Hybrid policies.

UNIT II CRYPTOSYSTEMS & AUTHENTICATION

Classical Cryptography-Substitution Ciphers-permutation Ciphers-Block Ciphers-DESModes of Operation- AES-Linear Cryptanalysis, Differential Cryptanalysis- Hash Function - SHA 512- Message Authentication Codes-HMAC - Authentication Protocols –

UNIT III PUBLIC KEY CRYPTOSYSTEMS

Introduction to Public key Cryptography- Number theory- The RSA Cryptosystem and Factoring Integer- Attacks on RSA-The ELGamal Cryptosystem- Digital Signature Algorithm-Finite Fields-Elliptic Curves Cryptography- Key management – Session and Interchange keys, Key exchange and generation-PKI

UNIT IV SYSTEM IMPLEMENTATION

Design Principles, Representing Identity, Access Control Mechanisms, Information Flow and Confinement Problem

Secure Software Development: Secured Coding - OWASP/SANS Top Vulnerabilities - Buffer Overflows - Incomplete mediation - XSS - Anti Cross Site Scripting Libraries - Canonical Data Format - Command Injection - Redirection - Inference – Application Controls

UNIT V NETWORK SECURITY

Secret Sharing Schemes-Kerberos- Pretty Good Privacy (PGP)-Secure Socket Layer (SSL)-Intruders – HIDS- NIDS - Firewalls - Viruses

OUTCOMES:

Upon Completion of the course, the students will be able to

- Implement basic security algorithms required by any computing system.
- Analyze the vulnerabilities in any computing system and hence be able to design a security solution.
- Analyze the possible security attacks in complex real time systems and their effective countermeasures
- Identify the security issues in the network and resolve it.
- Evaluate security mechanisms using rigorous approaches, including theoretical derivation, modeling, and simulations
- Formulate research problems in the computer security field

REFERENCES:

1. William Stallings, "Cryptography and Network Security: Principles and Practices",

Third Edition, Pearson Education, 2006.
2. Matt Bishop ,"Computer Security art and science ", Second Edition, Pearson Education, 2002
3. Wade Trappe and Lawrence C. Washington, "Introduction to Cryptography with Coding Theory" Second Edition, Pearson Education, 2007
4. Jonathan Katz, and Yehuda Lindell, Introduction to Modern Cryptography, CRC Press, 2007
5. Douglas R. Stinson, "Cryptography Theory and Practice", Third Edition, Chapman & Hall/CRC, 2006
6. Wenbo Mao, "Modern Cryptography – Theory and Practice", Pearson Education, First Edition, 2006.
7. Network Security and Cryptography, Menezes Bernard, Cengage Learning, New Delhi, 2011
8. Man Young Rhee, Internet Security, Wiley, 2003
9. OWASP top ten security vulnerabilities: http://xml.coverpages.org/OWASPTopTen.pdf

OBJECTIVES:

- To understand the need for parallel algorithms
- To expose the students to different models of parallel computation
- To expose the students to parallel sorting and searching algorithms
- To understand the application of the concepts studied to different types of problems
- To analyze parallel algorithms

UNIT I INTRODUCTION

Introduction to Parallel Algorithms – Models of Parallel Computation – Sorting on an EREWSIMD PRAM Computer – Relation between PRAM Models – SIMD Algorithms – MIMD Algorithms – Selection – Desirable Properties for Parallel Algorithms - Parallel Algorithm for Selection – Analysis of Parallel Algorithms.

UNIT II SORTING AND SEARCHING

Merging on the EREW and CREW Models - Fast Merging on EREW - Sorting Networks – Sorting on a Linear Array – Sorting on CRCW, CREW, EREW Models – Searching a Sorted Sequence – Searching a Random Sequence.

UNIT III ALGEBRAIC PROBLEMS

Generating Permutations and Combinations in Parallel – Matrix Transpositions – Matrix by Matrix Multiplications – Matrix by Vector multiplication.

UNIT IV GRAPH THEORY AND COMPUTATIONAL GEOMETRY PROBLEMS

Connectivity Matrix – Connected Components – All Pairs Shortest Paths – Minimum Spanning Trees – Point Inclusion – Intersection, Proximity and Construction Problems - Sequential Tree Traversal - Basic Design Principles – Algorithm – Analysis.

UNITV DECISION AND OPTIMIZATION PROBLEMS

Computing Prefix Sums – Applications - Job Sequencing with Deadlines – Knapsack Problem- The Bit Complexity of Parallel Computations**.**

OUTCOMES:

Upon completion of the course, ☐the students will be able to

- Identify the need for parallel algorithms
- Discuss the classification of parallel architectures and identify suitable programming models
- Perform sorting on Sorting on CRCW, CREW, EREW Models
- Search a sorted as well as random sequence
- Develop and analyze algorithms for different applications like matrix multiplication, shortest path, job sequencing and the knapsack problem.

REFERENCES:

1. Selim G. Akl, "The Design and Analysis of Parallel Algorithms", Prentice Hall, New Jersey, 1989.
2. Michael J. Quinn, "Parallel Computing : Theory & Practice", Tata McGraw Hill Edition, 2003.
3. Justin R. Smith, "The Design and Analysis of Parallel Algorithms", Oxford University Press, USA , 1993.
4. Joseph JaJa, "Introduction to Parallel Algorithms"**,** Addison-Wesley, 1992.

OBJECTIVES:

- Understand architectural requirements
- Identify architectural structures
- Develop architectural documentation
- Generate architectural alternatives
- Evaluate the architecture against the drivers

UNIT I ARCHITECTURAL DRIVERS

Introduction – Standard Definitions of Software Architecture– Architectural structures – Influence of software architecture on organization – Architecture Business Cycle – Functional requirements – Technical constraints – Quality Attributes – Quality Attribute Workshop (QAW) – Documenting Quality Attributes – Six part scenarios

UNIT II ARCHITECTURAL VIEWS AND DOCUMENTATION

Introduction – Standard Definitions for views – Structures and views- Perspectives: Static, dynamic and physical and the accompanying views – Representing views-available notations – Good practices in documentation– Documenting the Views using UML – Merits and Demerits of using visual languages – Need for formal languages - Architectural Description Languages – ACME

UNIT III ARCHITECTURAL STYLES

Introduction – Data flow styles – Call-return styles – Shared Information styles – Event styles – Case studies for each style

UNIT IV ARCHITECTURAL DESIGN

Approaches for architectural design – System decomposition – Attributes driven design – Architecting for specific quality attributes – Performance, Availability – Security – Architectural conformance

UNIT V ARCHITECTURE EVALUATION AND SOME SPECIAL TOPICS

Need for evaluation – Scenario based evaluation against the drivers – ATAM and its variations – Case studies in architectural evaluations – SOA and Web services – Cloud Computing – Adaptive structures

OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Explain key architectural drivers
- Explain the influence of architecture on business and technical activities
- Identify key architectural structures
- Adopt good practices for documenting the architecture
- Develop alternative architectures for a given problem
- Explain how to use formal languages to specify architecture
- Evaluate the architecture against the drivers
- Describe the recent trends in software architecture

REFERENCES:

1. Len Bass, Paul Clements, and Rick Kazman, "Software Architectures Principles and

Practices", 2n Edition, Addison-Wesley, 2003.
2. Anthony J Lattanze, "Architecting Software Intensive System. A Practitioner's Guide", Auerbach Publications, 2010.
3. Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Paulo Merson, Robert Nord, and Judith Stafford, "Documenting Software Architectures. Views and Beyond", 2nd Edition, Addison-Wesley, 2010.
4. Paul Clements, Rick Kazman, and Mark Klein, "Evaluating software architectures: Methods and case studies.", Addison-Wesley, 2001.
5. David Garlan and Mary Shaw, "Software architecture: Perspectives on an emerging discipline", Prentice Hall, 1996.
6. Rajkumar Buyya, James Broberg, and Andrzej Goscinski, "Cloud Computing. Principles and Paradigms", John Wiley & Sons, 2011
7. Mark Hansen, "SOA Using Java Web Services", Prentice Hall, 2007
8. David Garlan, Bradley Schmerl, and Shang-Wen Cheng, "Software Architecture-Based Self-Adaptation," 31-56. Mieso K Denko, Laurence Tianruo Yang, and Yan Zang (eds.), "Autonomic Computing and Networking". Springer Verlag, 2009.

213CSPT11 - MODEL CHECKING AND PROGRAM VERIFICATION

OBJECTIVES:

- To understand automata for model checking
- To understand LTL, CTL, and CTL*
- To understand timed automata, TCTL, and PCTL
- To understand verification of deterministic and recursive programs
- To understand verification of object-oriented programs
- To understand verification of parallel, distributed, and non-deterministic programs

UNIT I AUTOMATA AND TEMPORAL LOGICS
Automata on finite words – model checking regular properties – automata on infinite words – Buchi automata – Linear Temporal Logic (LTL) – automata based LTL model checking – Computational Tree Logic (CTL) – CTL model checking – CTL* model checking

UNIT II TIMED AND PROBABILISTIC TREE LOGICS
Timed automata – timed computational tree logic (TCTL) – TCTL model checking – probabilistic systems – probabilistic computational tree logic (PCTL) – PCTL model checking – PCTL* - Markov decision processes

UNIT III VERIFYING DETERMINISTIC AND RECURSIVE PROGRAMS
Introduction to program verification – verification of "while" programs – partial and total correctness – verification of recursive programs – case study: binary search – verifying recursive programs with parameters

UNIT IV VERIFYING OBJECT-ORIENTED AND PARALLEL PROGRAMS
Partial and total correctness of object-oriented programs – case study: Insertion in linked lists – verification of disjoint parallel programs – verifying programs with shared variables – case study: parallel zero search – verification of synchronization – case study: the mutual exclusion problem

UNIT V VERIFYING NON-DETERMINISTIC AND DISTRIBUTED PROGRAMS
Introduction to non-deterministic programs – partial and total correctness of nondeterministic programs – case study: The Welfare Crook Problem – syntax and semantics of distributed programs – verification of distributed programs – case study: A Transmission Problem – introduction to fairness

OUTCOMES:
Upon Completion of the course,□the students will be able to
- Perform model checking using LTL
- Perform model checking using CTL
- Perform model checking using CTL*
- Perform model checking using TCTL and PCTL
- Verify deterministic and recursive programs
- Verify object-oriented programs
- Verify parallel, distributed, and non-deterministic programs

REFERENCES:
1. C. Baier, J.-P. Katoen, and K. G. Larsen, "Principles of Model Checking", MIT Press, 2008.
2. E. M. Clarke, O. Grumberg, and D. A. Peled, "Model Checking", MIT Press, 1999.
3. M. Ben-Ari, "Principles of the SPIN Model Checker", Springer, 2008.
4. K. R. Apt, F. S. de Boer, E.-R. Olderog, and A. Pnueli, "Verification of Sequential and Concurrent Programs", Third Edition, Springer, 2010.
5. M. Huth and M. Ryan, "Logic in Computer Science --- Modeling and Reasoning about Systems", Second Edition, Cambridge University Press, 2004.
6. B. Berard et al., "Systems and Software Verification: Model-checking techniques and tools", Springer, 2010.
7. J. B. Almeida, M. J. Frade, J. S. Pinto, and S. M. de Sousa, "Rigorous Software Development: An Introduction to Program Verification", Springer, 2011.

213CSPT12 - EMBEDDED SOFTWARE DEVELOPMENT

OBJECTIVES:

- To understand processors and their instruction sets for embedded systems
- To understand hardware platform for embedded systems
- To design and analyze programs for embedded systems
- To design multi-tasking embedded systems with RTOS
- To understand overall embedded systems development lifecycle
- To understand distributed and multi-processor embedded systems

UNIT I PROCESSORS AND INSTRUCTION SETS
Introduction to embedded computing – overview of embedded system design process – instruction sets of processors: ARM, PIC, TI C55x, TI C64x – programming I/O – modes and exceptions – co-processors – memory system – CPU performance – CPU power consumption

UNIT II EMBEDDED COMPUTING PLATFORM
Basic computing platforms – CPU Bus – memory devices and systems – choosing a platform – development environments – debugging – consumer electronics architecture – platform-level performance analysis – design example: Audio Player

UNIT III PROGRAM DESIGN AND ANALYSIS
Components for embedded programs – models of programs – Assembly, linking, and loading – compiler optimizations – program-level performance analysis – performance optimization – program-level energy optimization – optimizing program size – program validation and testing – design example: Digital Still Camera

UNIT IV PROCESSES AND OPERATING SYSTEMS
Multiples tasks and multiple processes – multirate systems – pre-emptive RTOS – prioritybased scheduling – inter-process communication – evaluating OS performance – processes and power optimization – Case study: Real-time and embedded Linux – design example: Telephone answering machine

UNIT V SYSTEM DESIGN, NETWORKS, AND MULTIPROCESSORS
System design methodologies – requirements analysis – specifications – architecture design – quality assurance – distributed embedded systems – shared-memory multiprocessors – design example: Video accelerator

OUTCOMES:
Upon Completion of the course, the students will be able to
- Develop assembley code for processors such as ARM, PIC Microcontroller, TI C55x, TI C64x, etc.
- Choose appropriate hardware platform for a given application
- Perform platform-level performance analysis
- Design, develop, and debug embedded programs optimized for size or performance
- Develop embedded applications using an RTOS
- Perform OS-level performance analysis
- Employ best practices in embedded software engineering
- Develop distributed embedded systems and systems with shared-memory concurrency

REFERENCES:
1. Marilyn Wolf, "Computers as Components: Principles of Embedded Computing Systems Design", Third Edition, Morgan Kaufmann, 2012.
2. Christopher Hallinan, "Embedded Linux Primer: A Practical Real-World Approach", Second Edition, Prentice Hall, 2010.
3. Karim Yaghmour et al., "Building Embedded Linux Systems", O'Reilly, 2008.
4. Arnold S. Berger, "Embedded Systems Design: An Introduction to Processes, Tools, and Techniques", CMP Books, 2001.
5. David E. Simon, "An embedded Software Primer", Addison-Wesley, 1999.

213CSPT13 - CLOUD COMPUTING

OBJECTIVES:

- To introduce the broad perceptive of cloud architecture and model
- To understand the concept of Virtualization
- To be familiar with the lead players in cloud.
- To understand the features of cloud simulator
- To apply different cloud programming model as per need.
- To be able to set up a private cloud.
- To understand the design of cloud Services.
- To learn to design the trusted cloud Computing system

UNIT I CLOUD ARCHITECTURE AND MODEL

Technologies for Network-Based System – System Models for Distributed and Cloud Computing – NIST Cloud Computing Reference Architecture.
Cloud Models:- Characteristics – Cloud Services – Cloud models (IaaS, PaaS, SaaS) – Public vs Private Cloud –Cloud Solutions - Cloud ecosystem – Service management – Computing on demand.

UNIT II VIRTUALIZATION

Basics of Virtualization - Types of Virtualization - Implementation Levels of Virtualization - Virtualization Structures - Tools and Mechanisms - Virtualization of CPU, Memory, I/O Devices - Virtual Clusters and Resource management – Virtualization for Data-center Automation.

UNIT III CLOUD INFRASTRUCTURE

Architectural Design of Compute and Storage Clouds – Layered Cloud Architecture Development – Design Challenges - Inter Cloud Resource Management – Resource Provisioning and Platform Deployment – Global Exchange of Cloud Resources.

UNIT IV PROGRAMMING MODEL

Parallel and Distributed Programming Paradigms – MapReduce , Twister and Iterative MapReduce – Hadoop Library from Apache – Mapping Applications - Programming Support - Google App Engine, Amazon AWS - Cloud Software Environments -Eucalyptus, Open Nebula, OpenStack, Aneka, CloudSim

UNIT V SECURITY IN THE CLOUD

Security Overview – Cloud Security Challenges and Risks – Software-as-a-Service Security – Security Governance – Risk Management – Security Monitoring – Security Architecture Design – Data Security – Application Security – Virtual Machine Security - Identity Management and Access Control – Autonomic Security.

COURSE OUTCOMES:

Upon Completion of the course,☐the students will be able to

- Compare the strengths and limitations of cloud computing
- Identify the architecture, infrastructure and delivery models of cloud computing
- Apply suitable virtualization concept.
- Choose the appropriate cloud player
- Choose the appropriate Programming Models and approach.
- Address the core issues of cloud computing such as security, privacy and interoperability
- Design Cloud Services
- Set a private cloud

REFERENCES:

1. Kai Hwang, Geoffrey C Fox, Jack G Dongarra, "Distributed and Cloud Computing, From Parallel Processing to the Internet of Things", Morgan Kaufmann Publishers, 2012.
2. John W.Rittinghouse and James F.Ransome, "Cloud Computing: Implementation, Management, and Security", CRC Press, 2010.
3. Toby Velte, Anthony Velte, Robert Elsenpeter, "Cloud Computing, A Practical Approach", TMH, 2009.
4. Kumar Saurabh, " Cloud Computing – insights into New-Era Infrastructure", Wiley India,2011.
5. George Reese, "Cloud Application Architectures: Building Applications and Infrastructure in the Cloud" O'Reilly
6. James E. Smith, Ravi Nair, "Virtual Machines: Versatile Platforms for Systems and Processes", Elsevier/Morgan Kaufmann, 2005.
7. Katarina Stanoevska-Slabeva, Thomas Wozniak, Santi Ristol, "Grid and Cloud Computing – A Business Perspective on Technology and Applications", Springer.
8. Ronald L. Krutz, Russell Dean Vines, "Cloud Security – A comprehensive Guide to Secure Cloud Computing", Wiley – India, 2010.
9. Rajkumar Buyya, Christian Vecchiola, S.Tamarai Selvi, 'Mastering Cloud Computing", TMGH,2013.
10. Gautam Shroff, Enterprise Cloud Computing, Cambridge University Press, 2011
11. Michael Miller, Cloud Computing, Que Publishing,2008
12. Nick Antonopoulos, Cloud computing, Springer Publications, 2010

213CSPT14 - DATA VISUALIZATION TECHNIQUES

COURSE OBJECTIVES:

- To introduce visual perception and core skills for visual analysis

- To understand visualization for time-series analysis
- To understand visualization for ranking analysis
- To understand visualization for deviation analysis
- To understand visualization for distribution analysis
- To understand visualization for correlation analysis
- To understand visualization for multivariate analysis
- To understand issues and best practices in information dashboard design

UNIT I CORE SKILLS FOR VISUAL ANALYSIS

Information visualization – effective data analysis – traits of meaningful data – visual perception – making abstract data visible – building blocks of information visualization – analytical interaction – analytical navigation – optimal quantitative scales – reference lines and regions – trellises and crosstabs – multiple concurrent views – focus and context – details on demand – over-plotting reduction – analytical patterns – pattern examples

UNIT II TIME-SERIES, RANKING, AND DEVIATION ANALYSIS

Time-series analysis – time-series patterns – time-series displays – time-series best practices – part-to-whole and ranking patterns – part-to-whole and ranking displays – best practices – deviation analysis – deviation analysis displays – deviation analysis best practices

UNIT III DISTRIBUTION, CORRELATION, AND MULTIVARIATE ANALYSIS

Distribution analysis – describing distributions – distribution patterns – distribution displays – distribution analysis best practices – correlation analysis – describing correlations – correlation patterns – correlation displays – correlation analysis techniques and best practices – multivariate analysis – multivariate patterns – multivariate displays – multivariate analysis techniques and best practices

UNIT IV INFORMATION DASHBOARD DESIGN I

Information dashboard – categorizing dashboards – typical dashboard data – dashboard design issues and best practices – visual perception – limits of short-term memory – visually encoding data – Gestalt principles – principles of visual perception for dashboard design

UNIT V INFORMATION DASHBOARD DESIGN II

Characteristics of dashboards – key goals in visual design process – dashboard display media – designing dashboards for usability – meaningful organization – maintaining consistency – aesthetics of dashboards – testing for usability – case studies: sales dashboard, CIO dashboard, Telesales dashboard, marketing analysis dashboard

COURSE OUTCOMES:

Upon completion of the course, the students will be able to

- Explain principles of visual perception
- Apply core skills for visual analysis
- Apply visualization techniques for various data analysis tasks
- Design information dashboard

REFERENCES:

1. Stephen Few, "Now you see it: Simple Visualization techniques for quantitative analysis", Analytics Press, 2009.
2. Stephen Few, "Information dashboard design: The effective visual communication of data", O'Reilly, 2006.
3. Edward R. Tufte, "The visual display of quantitative information", Second Edition, Graphics Press, 2001.
4. Nathan Yau, "Data Points: Visualization that means something", Wiley, 2013.
5. Ben Fry, "Visualizing data: Exploring and explaining data with the processing environment", O'Reilly, 2008.
6. Gert H. N. Laursen and Jesper Thorlund, "Business Analytics for Managers: Taking business intelligence beyond reporting", Wiley, 2010.
7. Evan Stubbs, "The value of business analytics: Identifying the path to profitability", Wiley,2011.

213CSPT15 - PROTOCOLS AND ARCHITECTURE FOR WIRELESS SENSOR NETWORKS

UNIT I INTRODUCTION AND OVERVIEW OF WIRELESS SENSOR NETWORKS
Background of Sensor Network Technology, Application of Sensor Networks, Challenges for Wireless Sensor Networks, Mobile Adhoc NETworks (MANETs) and Wireless Sensor Networks, Enabling Technologies For Wireless Sensor Networks.

UNIT II ARCHITECTURES
Single-node Architecture, Hardware Components & Design Constraints, Operating Systems and Execution Environments, Introduction to TinyOS and nesC, Network Architecture, Sensor Network Scenarios, Optimization Goals and Figures of Merit, Design Principles for WSNs, Service Interfaces of WSNs, Gateway Concepts.

UNIT III DEPLOYMENT AND CONFIGURATION
Localization and Positioning, Coverage and Connectivity, Single-hop and Multi-hop Localization, Self Configuring Localization Systems, Sensor Management
Network Protocols: Issues in Designing MAC Protocol for WSNs, Classification of MAC Protocols, S-MAC Protocol, B-MAC Protocol, IEEE 802.15.4 Standard and Zig Bee, Dissemination Protocol for Large Sensor Network.

UNIT IV ROUTING PROTOCOLS AND DATA MANIPULATION
Issues in Designing Routing Protocols, Classification of Routing Protocols, Energy-Efficient Routing, Unicast, Broadcast and Multicast, Geographic Routing.
Data Centric and Content based Routing, Storage and Retrieval in Network, Compression Technologies for WSN, Data Aggregation Technique.

UNIT V SENSOR NETWORK PLATFORMS AND TOOLS
Sensor Node Hardware – Berkeley Motes, Programming Challenges, Node-level Software Platforms, Node-level Simulators, State-centric Programming.

REFERENCES:
1. Holger Karl & Andreas Willig, "Protocols And Architectures for Wireless Sensor Networks", John Wiley, 2005.
2. Feng Zhao & Leonidas J. Guibas, "Wireless Sensor Networks- An Information Processing Approach", Elsevier, 2007.
3. Raghavendra, Cauligi S, Sivalingam, Krishna M., Zanti Taieb, "Wireless Sensor Network", Springer 1st Ed. 2004 (ISBN: 978-4020-7883-5).
4. Kazem Sohraby, Daniel Minoli, & Taieb Znati, "Wireless Sensor Networks- Technology, Protocols, and Applications", John Wiley, 2007.
5. N. P. Mahalik, "Sensor Networks and Configuration: Fundamentals, Standards, Platforms, and Applications" Springer Verlag.
6. Anna Hac, "Wireless Sensor Network Designs", John Wiley, 2003.

OBJECTIVES:

- To understand the mathematical foundations needed for language processing
- To understand the representation and processing of Morphology and Part-of Speech Taggers
- To understand different aspects of natural language syntax and the various methods used for processing syntax
- To understand different methods of disambiguating word senses
- To know about various applications of natural language processing
- To learn the indexing and searching processes of a typical information retrieval system and to study NLP based retrieval systems
- To gain knowledge about typical text categorization and clustering techniques

UNIT I INTRODUCTION

Natural Language Processing – Mathematical Foundations – Elementary Probability Theory – Essential information Theory - Linguistics Essentials - Parts of Speech and Morphology – Phrase Structure – Semantics – Corpus Based Work.

UNIT II WORDS

Collocations – Statistical Inference – n-gram Models – Word Sense Disambiguation – Lexical Acquisition.

UNIT III GRAMMAR

Markov Models – Part-of-Speech Tagging – Probabilistic Context Free Grammars - Parsing.

UNIT IV INFORMATION RETRIEVAL

Information Retrieval Architecture – Indexing - Storage – Compression Techniques – Retrieval Approaches – Evaluation - Search Engines - Commercial Search Engine Features – Comparison - Performance Measures – Document Processing - NLP based Information Retrieval – Information Extraction.

UNIT V TEXT MINING

Categorization – Extraction Based Categorization – Clustering - Hierarchical Clustering - Document Classification and Routing - Finding and Organizing Answers from Text Search – Text Categorization and Efficient Summarization using Lexical Chains – Machine Translation - Transfer Metaphor - Interlingual and Statistical Approaches.

OUTCOMES:

Upon completion of the course,□ the students will be able to

- Identify the different linguistic components of given sentences
- Design a morphological analyser for a language of your choice using finite state automata concepts
- Implement a parser by providing suitable grammar and words
- Discuss algorithms for word sense disambiguation
- Build a tagger to semantically tag words using WordNet
- Design an application that uses different aspects of language processing.

REFERENCES:

1. Christopher D.Manning and Hinrich Schutze, " Foundations of Statistical Natural Language Processing ", MIT Press, 1999.
2. Daniel Jurafsky and James H. Martin, " Speech and Language Processing" , Pearson, 2008.
3. Ron Cole, J.Mariani, et.al "Survey of the State of the Art in Human Language Technology", Cambridge University Press, 1997.
4. Michael W. Berry, " Survey of Text Mining: Clustering, Classification and Retrieval", Springer Verlag, 2003.

313CSPT02 - SOCIAL NETWORKS ANALYSIS

OBJECTIVES:

- To understand the components of the social network
- To model and visualize the social network
- To mine the users in the social network
- To understand the evolution of the social network
- To mine the interest of the user

UNIT I INTRODUCTION

Introduction to Web - Limitations of current Web – Development of Semantic Web – Emergence of the Social Web – Statistical Properties of Social Networks -Network analysis - Development of Social Network Analysis - Key concepts and measures in network analysis - Discussion networks - Blogs and online communities - Web-based networks

UNIT II MODELING AND VISUALIZATION

Visualizing Online Social Networks - A Taxonomy of Visualizations - Graph Representation - Centrality- Clustering - Node-Edge Diagrams - Visualizing Social Networks with Matrix-Based Representations- Node-Link Diagrams - Hybrid Representations - Modelling and aggregating social network data **–** Random Walks and their Applications –Use of Hadoop and Map Reduce - Ontological representation of social individuals and relationships.

UNIT III MINING COMMUNITIES

Aggregating and reasoning with social network data, Advanced Representations - Extracting evolution of Web Community from a Series of Web Archive - Detecting Communities in Social Networks - Evaluating Communities – Core Methods for Community Detection & Mining - Applications of Community Mining Algorithms - Node Classification in Social Networks.

UNIT IV EVOLUTION

Evolution in Social Networks – Framework - Tracing Smoothly Evolving Communities - Models and Algorithms for Social Influence Analysis - Influence Related Statistics - Social Similarity and Influence - Influence Maximization in Viral Marketing - Algorithms and Systems for Expert Location in Social Networks - Expert Location without Graph Constraints - with Score Propagation – Expert Team Formation - Link Prediction in Social Networks - Feature based Link Prediction - Bayesian Probabilistic Models - Probabilistic Relational Models

UNIT V TEXT AND OPINION MINING

Text Mining in Social Networks -Opinion extraction – Sentiment classification and clustering - Temporal sentiment analysis - Irony detection in opinion mining - Wish analysis - Product review mining – Review Classification – Tracking sentiments towards topics over time

OUTCOMES:

Upon Completion of the course,□the students will be able to

- Work on the internals components of the social network
- Model and visualize the social network
- Mine the behaviour of the users in the social network
- Predict the possible next outcome of the social network
- Mine the opinion of the user

REFERENCES:

1. Charu C. Aggarwal, "Social Network Data Analytics", Springer; 2011
2. Peter Mika, "Social Networks and the Semantic Web", Springer, 1st edition, 2007.
3. Borko Furht, "Handbook of Social Network Technologies and Applications", Springer, 1st edition, 2010.
4. Guandong Xu , Yanchun Zhang and Lin Li, "Web Mining and Social Networking – Techniques and applications", Springer, 1st edition, 2011.
5. Giles, Mark Smith, John Yen, "Advances in Social Network Mining and Analysis", Springer, 2010.
6. Ajith Abraham, Aboul Ella Hassanien, Václav Snášel, "Computational Social Network Analysis: Trends, Tools and Research Advances", Springer, 2009.
7. Toby Segaran, "Programming Collective Intelligence", O'Reilly, 2012

313CSPT03 - MANAGING BIG DATA

OBJECTIVES:

- Understand big data for business intelligence
- Learn business case studies for big data analytics
- Understand nosql big data management
- Perform map-reduce analytics using Hadoop and related tools

UNIT I UNDERSTANDING BIG DATA

What is big data – why big data – convergence of key trends – unstructured data – industry examples of big data – web analytics – big data and marketing – fraud and big data – risk and big data – credit risk management – big data and algorithmic trading – big data and healthcare – big data in medicine – advertising and big data – big data technologies – introduction to Hadoop – open source technologies – cloud and big data – mobile business intelligence – Crowd sourcing analytics – inter and trans firewall analytics

UNIT II NOSQL DATA MANAGEMENT

Introduction to NoSQL – aggregate data models – aggregates – key-value and document data models – relationships – graph databases – schemaless databases – materialized views – distribution models – sharding – master-slave replication – peer-peer replication – sharding and replication – consistency – relaxing consistency – version stamps – mapreduce – partitioning and combining – composing map-reduce calculations

UNIT III BASICS OF HADOOP

Data format – analyzing data with Hadoop – scaling out – Hadoop streaming – Hadoop pipes – design of Hadoop distributed file system (HDFS) – HDFS concepts – Java interface – data flow – Hadoop I/O – data integrity – compression – serialization – Avro – file-based data structures

UNIT IV MAPREDUCE APPLICATIONS

MapReduce workflows – unit tests with MRUnit – test data and local tests – anatomy of MapReduce job run – classic Map-reduce – YARN – failures in classic Map-reduce and YARN – job scheduling – shuffle and sort – task execution – MapReduce types – input formats – output formats

UNIT V HADOOP RELATED TOOLS

Hbase – data model and implementations – Hbase clients – Hbase examples – praxis.Cassandra – cassandra data model – cassandra examples – cassandra clients – Hadoop integration.
Pig – Grunt – pig data model – Pig Latin – developing and testing Pig Latin scripts.
Hive – data types and file formats – HiveQL data definition – HiveQL data manipulation – HiveQL queries.

OUTCOMES:

Upon Completion of the course, the students will be able to

- Describe big data and use cases from selected business domains
- Explain NoSQL big data management
- Install, configure, and run Hadoop and HDFS
- Perform map-reduce analytics using Hadoop
- Use Hadoop related tools such as HBase, Cassandra, Pig, and Hive for big data analytics

REFERENCES:

1. Michael Minelli, Michelle Chambers, and Ambiga Dhiraj, "Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses", Wiley,2013.
2. P. J. Sadalage and M. Fowler, "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence", Addison-Wesley Professional, 2012.
3. Tom White, "Hadoop: The Definitive Guide", Third Edition, O'Reilley, 2012.
4. Eric Sammer, "Hadoop Operations", O'Reilley, 2012.
5. E. Capriolo, D. Wampler, and J. Rutherglen, "Programming Hive", O'Reilley, 2012.
6. Lars George, "HBase: The Definitive Guide", O'Reilley, 2011.
7. Eben Hewitt, "Cassandra: The Definitive Guide", O'Reilley, 2010.
8. Alan Gates, "Programming Pig", O'Reilley, 2011.

313CSPT04 - MOBILE APPLICATION DEVELOPMENT

OBJECTIVES:
1. Understand system requirements for mobile applications
2. Generate suitable design using specific mobile development frameworks
3. Generate mobile application design
4. Implement the design using specific mobile development frameworks
5. Deploy the mobile applications in marketplace for distribution

UNIT I INTRODUCTION
Introduction to mobile applications – Embedded systems - Market and business drivers for mobile applications – Publishing and delivery of mobile applications – Requirements gathering and validation for mobile applications

UNIT II BASIC DESIGN
Introduction – Basics of embedded systems design – Embedded OS - Design constraints for mobile applications, both hardware and software related – Architecting mobile applications – User interfaces for mobile applications – touch events and gestures – Achieving quality constraints – performance, usability, security, availability and modifiability.

UNIT III ADVANCED DESIGN
Designing applications with multimedia and web access capabilities – Integration with GPS and social media networking applications – Accessing applications hosted in a cloud computing environment – Design patterns for mobile applications.

UNIT IV TECHNOLOGY I - ANDROID
Introduction – Establishing the development environment – Android architecture – Activities and views – Interacting with UI – Persisting data using SQLite – Packaging and deployment – Interaction with server side applications – Using Google Maps, GPS and Wifi – Integration with social media applications.

UNIT V TECHNOLOGY II - IOS
Introduction to Objective C – iOS features – UI implementation – Touch frameworks – Data persistence using Core Data and SQLite – Location aware applications using Core Location and Map Kit – Integrating calendar and address book with social media application – Using Wifi - iPhone marketplace.

OUTCOMES:
Upon the students will be able to□Completion of the course,
1. Describe the requirements for mobile applications
2. Explain the challenges in mobile application design and development
3. Develop design for mobile applications for specific requirements
4. Implement the design using Android SDK
5. Implement the design using Objective C and iOS
6. Deploy mobile applications in Android and iPone marketplace for distribution

REFERENCES:
1. http://developer.android.com/develop/index.html
2. Jeff McWherter and Scott Gowell, "Professional Mobile Application Development", Wrox, 2012
3. Charlie Collins, Michael Galpin and Matthias Kappler, "Android in Practice", DreamTech, 2012
4. James Dovey and Ash Furrow, "Beginning Objective C", Apress, 2012
5. David Mark, Jack Nutting, Jeff LaMarche and Frederic Olsson, "Beginning iOS 6 Development: Exploring the iOS SDK", Apress, 2013.

313CSPT05 - BIO-INSPIRED COMPUTING

OBJECTIVES:

- Learn evolutionary theory and algorithms
- Understand Cellular Automata and artificial life
- Learn artificial neural systems and related learning algorithms
- Learn developmental and artificial immune systems
- Understand behavioral systems especially in the context of Robotics
- Understand collective systems such as ACO, PSO, and swarm robotics

UNIT I EVOLUTIONARY AND CELLULAR SYSTEMS

Foundations of evolutionary theory – Genotype – artificial evolution – genetic representations – initial population – fitness functions – selection and reproduction – genetic operators – evolutionary measures – evolutionary algorithms – evolutionary electronics – evolutionary algorithm case study

Cellular systems – cellular automata – modeling with cellular systems – other cellular systems – computation with cellular systems – artificial life – analysis and synthesis of cellular systems

UNIT II NEURAL SYSTEMS

Biological nervous systems – artificial neural networks – neuron models – architecture – signal encoding – synaptic plasticity – unsupervised learning – supervised learning – reinforcement learning – evolution of neural networks – hybrid neural systems – case study

UNIT III DEVELOPMENTAL AND IMMUNE SYSTEMS

Rewriting systems – synthesis of developmental systems – evolutionary rewriting systems – evolutionary developmental programs

Biological immune systems – lessons for artificial immune systems – algorithms and applications – shape space – negative selection algorithm – clonal selection algorithm - examples

UNIT IV BEHAVIORAL SYSTEMS

Behavior is cognitive science – behavior in AI – behavior based robotics – biological inspiration for robots – robots as biological models – robot learning – evolution of behavioral systems – learning in behavioral systems – co-evolution of body and control – towards self reproduction – simulation and reality

UNIT V COLLECTIVE SYSTEMS

Biological self-organization – Particle Swarm Optimization (PSO) – ant colony optimization (ACO) – swarm robotics – co-evolutionary dynamics – artificial evolution of competing systems – artificial evolution of cooperation – case study

OUTCOMES:

Upon completion of the course, the students will be able to

- Implement and apply evolutionary algorithms
- Explain cellular automata and artificial life
- Implement and apply neural systems
- Explain developmental and artificial immune systems
- Explain behavioral systems
- Implement and apply collective intelligence systems

REFERENCES:

1. D. Floreano and C. Mattiussi, "Bio-Inspired Artificial Intelligence", MIT Press, 2008.
2. F. Neumann and C. Witt, "Bioinspired Computation in combinatorial optimization: Algorithms and their computational complexity", Springer, 2010. 3. A. E. Elben and J. E. Smith, "Introduction to Evolutionary Computing", Springer, 2010.
4. D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning", Addison-Wesley, 1989.
5. Simon O. Haykin, "Neural Networks and Learning Machines", Third Edition, Prentice Hall, 2008.
6. M. Dorigo and T. Stutzle, "Ant Colony Optimization", A Bradford Book, 2004.
7. R. C. Ebelhart et al., "Swarm Intelligence", Morgan Kaufmann, 2001.

313CSPT06 - MEDICAL IMAGE PROCESSING

COURSE DESCRIPTION:

An advanced graduate level course on medical imaging and medical image analysis. The course includes topics in medical image formation, medical imaging techniques, such as XRay, Computed Tomography, Magnetic Resonance Imaging, and Nuclear Imaging, image segmentation, registration, statistical modeling, visualization, and applications of computational tools for medicine.

OBJECTIVES:

The course will provide the participants with an up-to-date background in current state-ofthe-art in medical imaging and medical image analysis. The aim of the course is to show how to extract, model, and analyze information from medical data and applications in order to help diagnosis, treatment and monitoring of diseases through computer science.

UNIT I INTRODUCTION

Introduction to medical imaging technology, systems, and modalities. Brief history; importance; applications; trends; challenges. Medical Image Formation Principles: X-Ray physics; X-Ray generation, attenuation, scattering; dose Basic principles of CT; reconstruction methods; artifacts; CT hardware.

UNIT II STORAGE AND PROCESSING

Medical Image Storage, Archiving and Communication Systems and Formats Picture archiving and communication system (PACS); Formats: DICOM Radiology Information Systems (RIS) and Hospital Information Systems (HIS). Medical Image Processing, Enhancement, Filtering Basic image processing algorithms Thresholding; contrast enhancement; SNR characteristics; filtering; histogram modeling.

UNIT III VISUALIZATION

Medical Image Visualization Fundamentals of visualization; surface and volume rendering/visualization; animation; interaction. Magnetic Resonance Imaging (MRI) Mathematics of MR; spin physics; NMR spectroscopy; imaging principles and hardware; image artifacts.

UNIT IV SEGMENTATION AND CLASSIFICATION

Medical Image Segmentation - Histogram-based methods; Region growing and watersheds; Markov Random Field models; active contours; model-based segmentation. Multi-scale segmentation; semi-automated methods; clustering-based methods; classification-based methods; atlas-guided approaches; multi-model segmentation. Medical Image Registration Intensity-based methods; cost functions; optimization techniques.

UNIT V NUCLEAR IMAGING

PET and SPECT Ultrasound Imaging methods; mathematical principles; resolution; noise effect; 3D imaging; positron emission tomography; single photon emission tomography; ultrasound imaging; applications. Medical Image Search and Retrieval Current technology in medical image search, content-based image retrieval, new trends: ontologies. Applications. Other Applications of Medical Imaging Validation, Image Guided Surgery, Image Guided Therapy, Computer Aided Diagnosis/Diagnostic Support Systems.

REFERENCES:

1. Paul Suetens, "Fundamentals of Medical Imaging", Second Edition, Cambridge University Press, 2009.
2. J. Michael Fitzpatrick and Milan Sonka, "Handbook of Medical Imaging, Volume 2. Medical Image Processing and Analysis", SPIE Publications, 2009.
3. Kayvan Najarian and Robert Splinter, "Biomedical Signal and Image Processing", Second Edition, CRC Press, 2005.
4. Geoff Dougherty, "Digital Image Processing for Medical Applications", First Edition, Cambridge University Press, 2009.
5. Jerry L. Prince and Jonathan Links, "Medical Imaging Signals and Systems", First Edition, Prentice Hall, 2005.
6. John L. Semmlow, "Biosignal and Medical Image Processing", Second Edition, CRC Press, 2008.

313CSPT07 - SOFTWARE DESIGN

OBJECTIVES:

- Analyze specifications
- Describe approaches to design
- Develop design documentation
- Evaluate the design

UNIT I SOFTWARE DESIGN PRINCIPLES

Introduction – Design process – Managing complexity – Software modeling and notations – Abstraction – Modularity – Hierarchy – Coupling - Cohesion – Design guidelines and checklists – Refactoring

UNIT II OO DESIGN

Object model – Classes and objects – Object oriented analysis – Key abstractions and mechanisms – Object oriented design – Identifying design elements – Detailed design – Case studies.

UNIT III DESIGN PATTERNS

Introduction to patterns – Design context – Reusable solutions – Documenting reusable solutions – Standard patterns from GOF book.

UNIT IV FUNCTION AND SERVICE ORIENTED DESIGNS

Structural decomposition – Detailed Design – Function oriented design Case study – Services – Service identification – Service design – Service composition – choreography and orchestration – Service oriented design Case study

UNIT V USER CENTERED DESIGN AND DESIGN REVIEW

Introduction to user centered design – Use in context – Interface and interaction – User centered design principles – Task analysis – Evaluation – Introduction to design review– Testing the design – Walk throughs – Review against check lists.

OUTCOMES:

Upon completion of the course, the students will be able to

- Describe different approaches to desigining a software application
- Analyze specifications and identify appropriate design strategies.
- Develop an appropriate design for a given set of requirements
- Identify applicatble design patterns for the solution
- Abstract and document reusable design patterns
- Evaluate a given design against the specifications

REFERENCES:

1. Grady Booch et al., "Object Oriented Analysis and Design with Applications", 3rd Edition, Pearson, 2010.
2. Carlos Otero, "Software Engineering Design: Theory and Practice", CRC Press, 2012
3. David Budgen, "Software Design", 2nd Edtion, Addison Wesley, 2003
4. Alan Shalloway and James R Trott, "Design Patterns Explained: A New Perspective on Object-Oriented Design", 2nd Edition, Addison-Wesley Professional, 2004
5. Hassan Gomaa, "Software Modeling and Design", Cambridge University Press, 2011
6. Eric Gamma et al., "Design Patterns: Elements of Reusable Object-Oriented Software", Addison-Wesley Professional, 1994
7. Ian Sommerville, "Software Engineering", 9th Edition, Addison-Wesley, 2010
8. M B Rosson and J M Carroll, "Usability Engineering: Scenario-Based Development of Human-Computer Interaction"**,** Morgan Kaufmann, 2002

313CSPT08 - RECONFIGURABLE COMPUTING

OBJECTIVES:

- To understand the need for reconfigurable computing
- To expose the students to various device architectures
- To examine the various reconfigurable computing systems
- To understand the different types of compute models for programming reconfigurable architectures
- To expose the students to HDL programming and familiarize with the development environment
- To expose the students to the various placement and routing protocols
- To develop applications with FPGAs

UNIT I DEVICE ARCHITECTURE

General Purpose Computing Vs Reconfigurable Computing – Simple Programmable Logic Devices – Complex Programmable Logic Devices – FPGAs – Device Architecture - Case Studies.

UNIT II RECONFIGURABLE COMPUTING ARCHITECTURES AND SYSTEMS

Reconfigurable Processing Fabric Architectures – RPF Integration into Traditional Computing Systems – Reconfigurable Computing Systems – Case Studies – Reconfiguration Management.

UNIT III PROGRAMMING RECONFIGURABLE SYSTEMS

Compute Models - Programming FPGA Applications in HDL – Compiling C for Spatial Computing – Operating System Support for Reconfigurable Computing.

UNIT IV MAPPING DESIGNS TO RECONFIGURABLE PLATFORMS

The Design Flow - Technology Mapping – FPGA Placement and Routing – Configuration Bitstream Generation – Case Studies with Appropriate Tools.

UNIT V APPLICATION DEVELOPMENT WITH FPGAS

Case Studies of FPGA Applications – System on a Programmable Chip (SoPC) Designs.

OUTCOMES:

Upon completion of the course, the students will be able to

- Identify the need for reconfigurable architectures
- Discuss the architecture of FPGAs
- Point out the salient features of different reconfigurable architectures
- Build basic modules using any HDL
- Develop applications using any HDL and appropriate tools
- Design and build an SoPC for a particular application

REFERENCES:

1. Maya B. Gokhale and Paul S. Graham, "Reconfigurable Computing: Accelerating Computation with Field-Programmable Gate Arrays", Springer, 2005.
2. Scott Hauck and Andre Dehon (Eds.), "Reconfigurable Computing – The Theory and Practice of FPGA-Based Computation", Elsevier / Morgan Kaufmann, 2008.
3. Christophe Bobda, "Introduction to Reconfigurable Computing – Architectures, Algorithms and Applications", Springer, 2010.

313CSPT09 - ENERGY AWARE COMPUTING

OBJECTIVES:

This course examines the design of power efficient architecture, power and performance tradeoffs, restructuring of software and applications and standards for energy aware Hardware and Software. The objective of this course is:

- To know the fundamental principles energy efficient devices
- To study the concepts of Energy efficient storage
- To introduce energy efficient algorithms
- To enable the students to know energy efficient techniques involved to support realtime systems.
- To study Energy aware applications

UNIT I INTRODUCTION

Energy efficient network on chip architecture for multi core system-Energy efficient MIPS CPU core with fine grained run time power gating – Low power design of Emerging memory technologies.

UNIT II ENERGY EFFICIENT STORAGE

Disk Energy Management-Power efficient strategies for storage system-Dynamic thermal management for high performance storage systems-Energy saving technique for Disk storage systems

UNIT III ENERGY EFFICIENT ALGORITHMS

Scheduling of Parallel Tasks – Task level Dynamic voltage scaling – Speed Scaling – Processor optimization- Memetic Algorithms – Online job scheduling Algorithms.

UNIT IV REAL TIME SYSTEMS

Multi processor system – Real Time tasks- Energy Minimization – Energy aware scheduling-Dynamic Reconfiguration- Adaptive power management-Energy Harvesting Embedded system

UNIT V ENERGY AWARE APPLICATIONS

On chip network – Video codec Design – Surveillance camera- Low power mobile storage.

OUTCOMES:

Upon completion of the course, the students will be able to

- Design Power efficient architecture Hardware and Software.
- Analyze power and performance trade off between various energy aware storage devices.
- Implement various energy aware algorithms.
- Restructure the software and Hardware for Energy aware applications.
- Explore the Energy aware applications

REFERENCES:

1. Ishfaq Ah mad, Sanjay Ranka, Handbook of Energy Aware and Green Computing, Chapman and Hall/CRC, 2012
2. Chong-Min Kyung, Sungioo yoo, Energy Aware system design Algorithms and Architecture, Springer, 2011.
3. Bob steiger wald ,Chris:Luero, Energy Aware computing, Intel Press,2012.

313CSPT10 - INFORMATION RETRIEVAL TECHNIQUES

OBJECTIVES:

- To understand the basics of Information Retrieval with pertinence to modeling, query operations and indexing
- To get an understanding of machine learning techniques for text classification and clustering
- To understand the various applications of Information Retrieval giving emphasis to Multimedia IR, Web Search
- To understand the concepts of digital libraries

UNIT I INTRODUCTION

Motivation – Basic Concepts – Practical Issues - Retrieval Process – Architecture - Boolean Retrieval –Retrieval Evaluation – Open Source IR Systems–History of Web Search – Web Characteristics–The impact of the web on IR ––IR Versus Web Search–Components of a Search engine

UNIT II MODELING

Taxonomy and Characterization of IR Models – Boolean Model – Vector Model - Term Weighting – Scoring and Ranking –Language Models – Set Theoretic Models - Probabilistic Models – Algebraic Models – Structured Text Retrieval Models – Models for Browsing

UNIT III INDEXING

Static and Dynamic Inverted Indices – Index Construction and Index Compression Searching - Sequential Searching and Pattern Matching. Query Operations -Query Languages–Query Processing - Relevance Feedback and Query Expansion - Automatic Local and Global Analysis – Measuring E ffectiveness and Efficiency.

UNIT IV CLASSIFICATION AND CLUSTERING

Text Classification and Naïve Bayes – Vector Space Classification – Support vector machines and Machine learning on documents. Flat Clustering – Hierarchical Clustering – Matrix decompositions and latent semantic indexing – Fusion and Meta learning

UNIT V SEARCHING AND RANKING

Searching the Web –Structure of the Web –IR and web search – Static and Dynamic Ranking - Web Crawling and Indexing – Link Analysis - XML Retrieval Multimedia IR: Models and Languages – Indexing and Searching Parallel and Distributed IR – Digital Libraries

OUTCOMES:

Upon completion of the course, the students will be able to

- Build an Information Retrieval system using the available tools
- Identify and design the various components of an Information Retrieval system
- Apply machine learning techniques to text classification and clustering which is used for efficient Information Retrieval
- Analyze the Web content structure
- Design an efficient search engine

REFERENCES:

1. Ricardo Baeza – Yates, BerthierRibeiro – Neto, Modern Information Retrieval: The concepts and Technology behind Search (ACM Press Books), Second Edition 2011
2. Christopher D. Manning, PrabhakarRaghavan, HinrichSchutze, Introduction to Information Retrieval, Cambridge University Press, First South Asian Edition 2012
3. Stefan Buttcher, Charles L. A. Clarke, Gordon V. Cormack, Information Retrieval Implementing and Evaluating Search Engines, The MIT Press, Cambridge, Massachusetts London, England, 2010

313CSPT11 - DATA MINING TECHNIQUES

UNIT I INTRODUCTION TO DATA MINING

Introduction to Data Mining – Data Mining Tasks – Components of Data Mining Algorithms – Data Mining supporting Techniques – Major Issues in Data Mining – Measurement and Data – Data Preprocessing – Data sets

UNIT II OVERVIEW OF DATA MINING ALGORITHMS

Overview of Data Mining Algorithms – Models and Patterns – Introduction – The Reductionist viewpoint on Data Mining Algorithms – Score function for Data Mining Algorithms- Introduction – Fundamentals of Modeling – Model Structures for Prediction – Models for probability Distributions and Density functions – The Curve of Dimensionality – Models for Structured Data – Scoring Patterns – Predictive versus Descriptive score functions – Scoring Models with Different Complexities – Evaluation of Models and Patterns – Robust Methods.

UNIT III CLASSIFICATIONS

Classifications – Basic Concepts – Decision Tree induction – Bayes Classification Methods – Rule Based Classification – Model Evaluation and Selection – Techniques to Improve Classification Accuracy – Classification: Advanced concepts – Bayesian Belief Networks- Classification by Back Propagation – Support Vector Machine – Classification using frequent patterns.

UNIT IV CLUSTER ANALYSIS

Cluster Analysis: Basic concepts and Methods – Cluster Analysis – Partitioning methods – Hierarchical methods – Density Based Methods – Grid Based Methods – Evaluation of Clustering – Advanced Cluster Analysis: Probabilistic model based clustering – Clustering High – Dimensional Data – Clustering Graph and Network Data – Clustering with Constraints.

UNIT V ASSOCIATION RULE MINING AND VISUALIZATION

Association Rule Mining – Introduction – Large Item sets – Basic Algorithms – Parallel and Distributed Algorithms – Comparing Approaches – Incremental Rules – Advanced Association Rule Techniques – Measuring the Quality of Rules – Visualization of Multidimensional Data – Diagrams for Multidimensional visualization – Visual Data Mining – Data Mining Applications – Case Study: WEKA.

REFERENCE BOOKS:

1. Jiawei Han, Micheline Kamber , Jian Pei, "Data Mining: Concepts and Techniques", Third Edition (The Morgan Kaufmann Series in Data Management Systems), 2012.
2. David J. Hand, Heikki Mannila and Padhraic Smyth "Principles of Data Mining" (Adaptive Computation and Machine Learning), 2005
3. Margaret H Dunham, "Data Mining: Introductory and Advanced Topics", 2003
4. Soman, K. P., Diwakar Shyam and Ajay V. "Insight Into Data Mining: Theory And Practice", PHI, 2009.

313CSPT12 - BIO INFORMATICS

OBJECTIVES:

- To get exposed to the domain of bioinformatics
- To understand the role of data warehousing and data mining for bioinformatics
- To learn to model bioinformatics based applications
- To understand how to deploy the pattern matching and visualization techniques in bioinformatics
- To study the Microarray technologies for genome expression

UNIT I INTRODUCTION
Need for Bioinformatics technologies – Overview of Bioinformatics technologies – Structural bioinformatics – Data format and processing – secondary resources- Applications – Role of Structural bioinformatics - Biological Data Integration System.

UNIT II DATAWAREHOUSING AND DATAMINING IN BIOINFORMATICS
Bioinformatics data – Data ware housing architecture – data quality – Biomedical data analysis – DNA data analysis – Protein data analysis – Machine learning – Neural network architecture- Applications in bioinformatics

UNIT III MODELING FOR BIOINFORMATICS
Hidden markov modeling for biological data analysis – Sequence identification – Sequence classification – multiple alignment generation – Comparative modeling – Protein modeling – genomic modeling – Probabilistic modeling – Bayesian networks – Boolean networks - Molecular modeling – Computer programs for molecular modeling

UNIT IV PATTERN MATCHING AND VISUALIZATION
Gene regulation – motif recognition and motif detection – strategies for motif detection – Visualization – Fractal analysis – DNA walk models – one dimension – two dimension – higher dimension – Game representation of Biological sequences – DNA, Protein, Amino acid sequences

UNIT V MICROARRAY ANALYSIS
Microarray technology for genome expression study – image analysis for data extraction – preprocessing – segmentation – gridding , spot extraction , normalization, filtering – cluster analysis – gene network analysis – Compared Evaluation of Scientific Data Management Systems – Cost Matrix – Evaluation model ,Benchmark , Tradeoffs

OUTCOMES:
Upon Completion of the course, the students will be able to
- Deploy the data warehousing and data mining techniques in Bioinformatics
- Model bioinformatics based applications
- Deploy the pattern matching and visualization techniques in bioinformatics
- Work on the protein sequences
- Use the Microarray technologies for genome expression

REFERENCES:
1. Yi-Ping Phoebe Chen (Ed), "Bio Informatics Technologies", First Indian Reprint, Springer Verlag, 2007.
2. N.J. Chikhale and Virendra Gomase, "Bioinformatics- Theory and Practice", Himalaya Publication House, India, 2007
3. Zoe lacroix and Terence Critchlow, "Bio Informatics – Managing Scientific data", First Indian Reprint, Elsevier, 2004
4. Bryan Bergeron, "Bio Informatics Computing", Second Edition, Pearson Education, 2003.
5. Arthur M Lesk, "Introduction to Bioinformatics", Second Edition, Oxford University Press, 2005
6. Burton. E. Tropp, "Molecular Biology: Genes to Proteins ", 4th edition, Jones and Bartlett Publishers, 2011
7. Dan Gusfield, "Algorithms on Strings Trees and Sequences", Cambridge University Press, 1997.
8. P. Baldi, S Brunak , Bioinformatics, "A Machine Learning Approach ", MIT Press, 1998.

313CSPT13 - SOFTWARE QUALITY ASSURANCE

OBJECTIVES:

- Describe approaches to quality assurance
- Understand quality models
- Evaluate the system based on the chosen quality model

UNIT I INTRODUCTION

Introduction – Views on quality – Cost of quality - Quality models – Quality frameworks – Verification and Validation – Defect taxonomy – Defect management – Statistics and measurements – IEEE standards – Quality assurance and control processes

UNIT II VERIFICATION

Introduction – Verification techniques – Inspections, reviews, walk-throughs – Case studies

UNIT III TEST GENERATION

Software testing- Validation – Test plan – Test cases - Test Generation – Equivalence partitioning – Boundary value analysis – Category partition method – Combinatorial generation - Decision tables – Examples and Case studies

UNIT IV STRUCTURAL TESTING

Introduction – Test adequacy criteria – Control flow graph – Coverages: block, conditions, multiple conditions, MC/DC, path – Data flow graph – Definition and use coverages – C-use, P-use, Def-clear, Def-use – Finite state machines – Transition coverage – Fault based testing – Mutation analysis – Case studies

UNIT V FUNCTIONAL TESTING

Introduction – Test adequacy criteria - Test cases from use cases – Exploratory testing - Integration, system, acceptance, regression testing – Testing for specific attributes: Performance, load and stress testing – Usability testing – Security testing - Test automation – Test oracles

OUTCOMES:

Upon Completion of the course, the students will be able to

- Describe different approaches to testing software applications
- Analyze specifications and identify appropriate test generation strategies
- Develop an appropriate test design for a given test object
- Identify applicatble measurements for the verification and validation effort
- Execute the test design
- Evaluate the testing effort based on adequate measures

REFERENCES:

1. Boriz Beizer, "Software Testing Techniques", 2nd Edition, DreamTech, 2009.
2. Aditya P. Mathur, "Foundations of Software Testing", Pearson, 2008
3. Mauro Pezze and Michal Young, "Software Testing and Analysis. Process, Principles, and Techniques", John Wiley 2008
4. Stephen H. Kan, "Metrics and Models in Software Quality Engineering", 2nd Edition, Pearson, 2003
5. Kshirasagar Naik and Priyadarshi Tripathy (Eds), "Software Testing and Quality Assurance: Theory and Practice", John Wiley, 2008
6. "Combinatorial Methods in Software Testing", ttp://csrc.nist.gov/groups/SNS/acts/index.html

313CSPT14 - MULTI OBJECTIVE OPTIMIZATION TECHNIQUES

OBJECTIVES:

- Learn fundamental principles of Multiobjective Optimization (MOP)
- Survey different Multiobjective Optimization algorithms
- Introduce various design issues of MOP
- Develop and Evaluate MOP Algorithms
- Learn Parallel and hybrid MOP Algorithms
- Learn other Metaheuristics

UNIT I INTRODUCTION AND CLASSICAL APPROACHES

Multiobjective optimization: Introduction - Multiobjective optimization problem-principles – Difference between single and multiobjective optimization – Dominance and Pareto Optimality , Classical Methods – Weighted Sum - □ Constraint method – Weighted Metric methods – Benson's method - Value Function - Goal Programming methods – Interactive Methods

UNIT II MOP EVOLUTIONARY ALGORITHMS

Generic MOEA - Various MOEAs: MOGA, NSGA-II, NPGA, PAES, SPEA2, MOMGA, micro GA - Constrained MOEAs: Penalty Function approach - Constrained Tournament – Ray – Tai –Seow's Method.

UNIT III THEORETICAL ISSUES

Fitness Landscapes - Fitness Functions - Pareto Ranking - Pareto Niching and Fitness Sharing - Recombination Operators - Mating Restriction - Solution Stability and Robustness - MOEA Complexity - MOEA Scalability - Running Time Analysis - MOEA Computational Cost - No Free Lunch Theorem.

UNIT IV MOEA TESTING, ANALYSIS, AND PARALLELIZATION

MOEA Experimental Measurements – MOEA Statistical Testing Approaches – MOEA Test Suites - MOEA Parallelization: Background – Paradigms – Issues - MOEA Local Search Techniques.

UNIT V APPLICATIONS AND ALTERNATIVE METAHEURISTICS

Scientific Applications: Computer Science and Computer Engineering - Alternative Metaheuristics: Simulated Annealing – Tabu Search and Scatter Search – Ant System – Distributed Reinforcement Learning – Particle Swarm Optimization – Differential Evolution – Artificial Immune Systems - Other Heuristics.

OUTCOMES:

Upon Completion of the course, students will be able to

- Explain MOP principles
- Explain classical methods to solve MOP problems
- Be familiar with and explain structures of different MOP algorithms
- Solve constrained MOP problems
- Explain various design issues of MOP algorithms
- Perform a evaluation and analysis of MOP algorithm results
- Explain parallelization of MOP algorithms
- Develop parallel and hybrid MOP algorithms
- Identify various real time MOP applications
- Explain other search algorithms

REFERENCES:

1. Carlos A. Coello Coello, Gary B. Lamont, David A. Van Veldhuizen, "Evolutionary

Algorithms for Solving Multi-objective Problems", Second Edition, Springer, 2007.
2. Kalyanmoy Deb, " Multi-Objective Optimization Using Evolutionary Algorithms", John Wiley, 2002.
3. Aimin Zhoua, Bo-Yang Qub, Hui Li c, Shi-Zheng Zhaob, Ponnuthurai Nagaratnam Suganthan b, Qingfu Zhangd, "Multiobjective evolutionary algorithms: A survey of the state of the art", Swarm and Evolutionary Computation (2011) 32–49.
4. E Alba, M Tomassini, "Parallel and evolutionary algorithms", Evolutionary Computation, IEEE Transactions on 6 (5), 443-462.
5. Crina Grosan, Ajith Abraham, "Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews", Studies in Computational Intelligence, Vol. 75, Springer, 2007.
6. Christian Blum and Andrea Roli. 2003. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.* 35, 3 (September 2003), 268-308.

OBJECTIVES:

- Describe approaches to enterprise application integration
- Understand the integration middleware
- Evaluate the integration approaches suitable for a given problem

UNIT I INTRODUCTION

Requirements for EAI - Challenges in EAI – Integration with legacy systems – Integration with partners - Heterogeneous environment – Implementation approaches – Web services, messaging, ETL, direct data integration – Middleware requirements – Approaches to integration – services oriented and messaging.

UNIT II INTEGRATION PATTERNS

Introduction to integration patterns – Architecture for application integration – Integration patterns – Point to point, broker, message bus, publish/subscribe, Challenges in performance, security, reliability - Case studies

UNIT III SERVICE ORIENTED INTEGRATION

Business process integration - Composite applications-services – Web services – Service choreography and orchestration - Business process modeling - BPMN, Business process execution - BPEL – Middleware infrastructure - Case studies

UNIT IV MESSAGING BASED INTEGRATION

Messaging – Synchronous and asynchronous – Message structure – Message oriented middleware – Reliability mechanisms – Challenges – Messaging infrastructure – Java Messaging Services – Case studies

UNIT V ENTERPRISE SERVICE BUS

Enterprise Service Bus – routing, scalable connectivity, protocol and message transformations, data enrichment, distribution, correlation, monitoring – Deployment configurations – Global ESB, Directly connected, Federated, brokered ESBs – Application server based – Messaging system based – Hardware based ESBs – Support to SOA, message based and event based integrations - Case studies.

OUTCOMES:

Upon Completion of the course, □the students will be able to

- Describe different approaches to integration enterprise applications
- Analyze specifications and identify appropriate integration approaches
- Develop a suitable integration design for a given problem
- Identify appropriate integration middleware for a given problem
- Evaluate the integration approaches against specified requirements

REFERENCES:

1. George Mentzas and Andreas Frezen (Eds), "Semantic Enterprise Application Integration for Business Processes: Service-oriented Frameworks", Business Science Reference, 2009
2. Waseem Roshen, "SOA Based Enterprise Integration", Tata McGrawHill, 2009.
3. G Hohpe and B Woolf,"Enterprise Integration Patterns:Designing, Building, and Deploying Messaging Solutions", Addison Wesley Professional, 2003
4. D Linthicum, "Next Generation Application Integration: From Simple Information to Web Services", Addison Wesley, 2003
5. Martin Fowler, "Patterns of Enterprise Application Architecture", Addison- Wesley, 2003
6. Kapil Pant and Matiaz Juric, "Business Process Driven SOA using BPMN and BPEL: From Business Process Modeling to Orchestration and Service Oriented Architecture", Packt Publishing, 2008

313CSPT16 - INFORMATION STORAGE MANAGEMENT

UNIT I INTRODUCTION TO STORAGE TECHNOLOGY

Review data creation and the amount of data being created and understand the value of data to a business, challenges in data storage and data management, Solutions available for data storage, Core elements of a data center infrastructure, role of each element in supporting business activities

UNIT II STORAGE SYSTEMS ARCHITECTURE

Hardware and software components of the host environment, Key protocols and concepts used by each component ,Physical and logical components of a connectivity environment ,Major physical components of a disk drive and their function, logical constructs of a physical disk, access characteristics, and performance Implications, Concept of RAID and its components, Different RAID levels and their suitability for different application environments: RAID 0, RAID 1, RAID 3, RAID 4, RAID 5, RAID 0+1, RAID 1+0, RAID 6, Compare and contrast integrated and modular storage systems ,Iligh-level architecture and working of an intelligent storage system

UNIT III INTRODUCTION TO NETWORKED STORAGE

Evolution of networked storage, Architecture, components, and topologies of FC-SAN, NAS, and IP-SAN, Benefits of the different networked storage options, understand the need for long-term archiving solutions and describe how CAS full fill the need, understand the appropriateness of the different networked storage options for different application environments

UNIT IV INFORMATION AVAILABILITY, MONITORING & MANAGING DATACENTER

List reasons for planned/unplanned outages and the impact of downtime, Impact of downtime - Differentiate between business continuity (BC) and disaster recovery (DR) ,RTO and RPO, Identify single points of failure in a storage infrastructure and list solutions to mitigate these failures, Architecture of backup/recovery and the different backup/ recovery topologies, replication technologies and their role in ensuring information availability and business continuity, Remote replication technologies and their role in providing disaster recovery and business continuity capabilities. Identify key areas to monitor in a data center, Industry standards for data center monitoring and management, Key metrics to monitor for different components in a storage infrastructure, Key management tasks in a data center

UNIT V SECURING STORAGE AND STORAGE VIRTUALIZATION

Information security, Critical security attributes for information systems, Storage security domains, List and analyzes the common threats in each domain, Virtualization technologies, block-level and file-level virtualization technologies and processes

REFERENCE BOOKS:

1. EMC Corporation, Information Storage and Management, Wiley, India.
2. Robert Spalding, "Storage Networks: The Complete Reference", Tata McGraw Hill , Osborne, 2003.
3. Marc Farley, "Building Storage Networks", Tata McGraw Hill ,Osborne, 2001.
4. Additional resource material on www.emc.com/resource-library/resource-library.esp

313CSPT17 - ROBOTICS

OBJECTIVES:

- To understand robot locomotion and mobile robot kinematics
- To understand perception in robotics
- To understand mobile robot localization
- To understand mobile robot mapping
- To understand simultaneous localization and mapping (SLAM)
- To understand robot planning and navigation

UNIT I LOCOMOTION AND KINEMATICS

Introduction to Robotics – key issues in robot locomotion – legged robots – wheeled mobile robots – aerial mobile robots – introduction to kinematics – kinematics models and constraints – robot maneuverability

UNIT II ROBOT PERCEPTION

Sensors for mobile robots – vision for robotics – cameras – image formation – structure from stereo – structure from motion – optical flow – color tracking – place recognition – range data

UNIT III MOBILE ROBOT LOCALIZATION

Introduction to localization – challenges in localization – localization and navigation – belief representation – map representation – probabilistic map-based localization – Markov localization – EKF localization – UKF localization – Grid localization – Monte Carlo localization – localization in dynamic environments

UNIT IV MOBILE ROBOT MAPPING

Autonomous map building – occupancy grip mapping – MAP occupancy mapping – SLAM – extended Kalman Filter SLAM – graph-based SLAM – particle filter SLAM – sparse extended information filter – fastSLAM algorithm

UNIT V PLANNING AND NAVIGATION

Introduction to planning and navigation – planning and reacting – path planning – obstacle avoidance techniques – navigation architectures – basic exploration algorithms

OUTCOMES:

Upon Completion of the course, the students will be able to

- Explain robot locomotion
- Apply kinematics models and constraints
- Implement vision algorithms for robotics
- Implement robot localization techniques
- Implement robot mapping techniques
- Implement SLAM algorithms
- Explain planning and navigation in robotics

REFERENCES:

1. Roland Seigwart, Illah Reza Nourbakhsh, and Davide Scaramuzza, "Introduction to autonomous mobile robots", Second Edition, MIT Press, 2011.
2. Sebastian Thrun, Wolfram Burgard, and Dieter Fox, "Probabilistic Robotics", MIT Press, 2005.
3. Howie Choset et al., "Principles of Robot Motion: Theory, Algorithms, and Implementations", A Bradford Book, 2005.
4. Gregory Dudek and Michael Jenkin, "Computational Principles of Mobile Robotics", Second Edition, Cambridge University Press, 2010.
5. Maja J. Mataric, "The Robotics Primer", MIT Press, 2007.

313CSPT18 - COMPILER OPTIMIZATION TECHNIQUES

OBJECTIVES:

- To understand the optimization techniques used in compiler design.
- To be aware of the various computer architectures that support parallelism.
- To become familiar with the theoretical background needed for code optimization.
- To understand the techniques used for identifying parallelism in a sequential program.
- To learn the various optimization algorithms.

UNIT I INTRODUCTION

Language Processors - The Structure of a Compiler – The Evolution of Programming Languages- The Science of Building a Compiler – Applications of Compiler Technology Programming Language Basics - The Lexical Analyzer Generator -Parser Generator - Overview of Basic Blocks and Flow Graphs - Optimization of Basic Blocks - Principle Sources of Optimization.

UNIT II INSTRUCTION-LEVEL PARALLELISM

Processor Architectures – Code-Scheduling Constraints – Basic-Block Scheduling –Global Code Scheduling – Software Pipelining.

UNIT III OPTIMIZING FOR PARALLELISM AND LOCALITY-THEORY

Basic Concepts – Matrix-Multiply: An Example - Iteration Spaces - Affine Array Indexes – Data Reuse Array data dependence Analysis.

UNITIV OPTIMIZING FOR PARALLELISM AND LOCALITY – APPLICATION

Finding Synchronization - Free Parallelism – Synchronization Between Parallel Loops – Pipelining – Locality Optimizations – Other Uses of Affine Transforms.

UNIT V INTERPROCEDURAL ANALYSIS

Basic Concepts – Need for Interprocedural Analysis – A Logical Representation of Data Flow – A Simple Pointer-Analysis Algorithm – Context Insensitive Interprocedural Analysis - Context-Sensitive Pointer-Analysis - Datalog Implementation by Binary Decision Diagrams.

REFERENCES:

1. Alfred V. Aho, Monica S.Lam, Ravi Sethi, Jeffrey D.Ullman, "Compilers:Principles, Techniques and Tools", Second Edition, Pearson Education,2008.
2. Randy Allen, Ken Kennedy, "Optimizing Compilers for Modern Architectures: A Dependence-based Approach", Morgan Kaufmann Publishers, 2002.
3. Steven S. Muchnick, "Advanced Compiler Design and Implementation",Morgan Kaufmann Publishers - Elsevier Science, India, Indian Reprint 2003.

Registrar