COEP Technological University

Department of Computer Science and Engineering

Curriculum Structure

B. Tech. Computer Engineering

(Effective from: A.Y. 2026-27)

List of Abbreviations

Abbreviation	Title
BSC	Basic Science Course
ESC	Engineering Science Course
PCC	Programme Core Course
PEC	Programme Elective Course
OE	Open Elective - other than particular program
MDM	Multidisciplinary Minor
VSEC	Vocational and Skill Enhancement Course
HSMC	Humanities Social Science and Management
AEC	Ability Enhancement Course
IKS	Indian Knowledge System
VEC	Value Education Course
RM	Research Methodology
OJT	Internship
CEA	Community Engagement Activity /Field Project
CCA	Co-curricular & Extracurricular Activities

Final Year B. Tech.
Computer Science and Engineering
[Level 6, UG Degree] Semester -VII

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	CIE	ESE
01	PCC	<TBD>	Compiler Construction	3	0	2	1	4	30	20	50	100	--
02	PCC	<TBD>	Cryptography and Network Security	3	0	2	1	4	30	20	50	100	--
03	PEC	<TBD>	Programme Elective-3	3	0	2	1	4	30	20	50	100	--
04	OJT	<TBD>	Internship	0	0	0	0	1	--	--	--	100	--
05	VSEC	<TBD>	Project Stage-3	0	0	10	0	5	--	--	--	60	40
06	MDM	<TBD>	Multidisciplinary Minor-4	3	0	0	1	3	30	20	50	--	--
			Total	**12**	**00**	**16**	**04**	**21**					

[Level 6, UG Degree] Semester -VIII

Scheme A

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	CIE	ESE
01	PEC	<TBD>	Programme Elective-4	2	0	2	0	3	30	20	50	100	--
02	PEC	<TBD>	Programme Elective-5	3	0	0	0	3	30	20	50	--	--
03	OJT	<TBD>	In-house Internship	0	0	0	0	5	--	--	--	60	40
			Total	**05**	**00**	**02**	**00**	**11**					

Scheme B

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	CIE	ESE
01	PEC	<TBD>	MOOC-1	3	0	0	0	3	--	25	75	--	--
02	PEC	<TBD>	MOOC-2	3	0	0	0	3	--	25	75	--	--
03	OJT	<TBD>	Industry Internship	0	0	0	0	5	--	--	--	60	40
			Total	**06**	**00**	**00**	**00**	**11**					

DEPARTMENT ELECTIVES

	PEC-1	PEC-2	PEC-3	PEC-4	PEC-5
Tracks	**Sem-5**	**Sem-6**	**Sem-7**	**Sem-8**	**Sem-8**
System Software/ Hardware	Cloud Computing Parallel Computer Architecture and Programming	Advanced System Programming	System Administration GPU Computing	Multicore Technology	Embedded Systems Storage and Virtualization
Networking and Security	Mobile and Ad-hoc Networks Remote Sensing	Block Chain Technologies Geographical Information Systems	Internet of Things	Distributed Systems Cyber Security	Computer Forensics and Data Recovery
Artificial Intelligence	Data Visualization	Machine Learning	Deep Learning	Natural Language Processing	Generative AI
Algorithms and Programming	Advanced Data Structures	Functional Programming Advanced Database Management Systems	Parallel Algorithms	Graphics and Multimedia	Software Design Patterns

MULTIDISCIPLINARY MINORS

Semester	Course Code	Course Title	L	T	P	S	Cr
VII	MDM-04	Large Language Models Theory and Deployment	3	0	0	1	3
VII	MDM-04	Quantum Programming and Project	3	0	0	1	3
VII	MDM-04	Advanced Database Management Systems	3	0	0	1	3

(PCC) Compiler Construction

Teaching Scheme
Lectures: 3 Hrs/ Week
Laboratory: 2 Hrs/ Week
Self-Study: 1 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks
Laboratory: CIE: 100 Marks

Course outcomes

1. Demonstrate the understanding of different phases of compilation.
2. Demonstrate the ability to generate and implement lexical and syntax analyzer.
3. Analyze and differentiate different parsing techniques and syntax directed translation schemes and choose the optimal parsing technique.
4. Design and implement intermediate code generation for various program constructs.
5. Identify different code optimization techniques for the various statements

Course content

Introduction **[6 Hrs]**

The role of language translation in the programming process, Translator Issues, Types of translators, Design of an assembler. Introduction of phases of compilation, Front end and backend model of compiler, cross compiler, Incremental compiler

Lexical analysis **[6 Hrs]**

Concept of Lexical analysis, regular expressions, Tokens, Lexemes and Patterns, Block diagram of Lexical analyzer, Revision of finite automata, Introduction to LEX Tool an LEX file specification, Error detection and recovery in LEX

Syntax Analysis **[8 Hrs]**

Context Free Grammers(CFG), concept of parsing, Parsing Techniques- Top-down parsers: Introduction, Predictive Parsing- removal of left recursion, removal of left factoring, Recursive Descent Parsing, Predictive LL(k) Parsing using Tables, Bottom Up parsing: Introduction, Shift-Reduce Parsing Using the ACTION/GOTO Tables, Table Construction, SLR(1), LR(1), and LALR(1) Grammars, Introduction to YACC Tool & YACC file specification, Error detection and recovery in YACC.

Semantic Analysis & Intermediate Representation **[8 Hrs]**

Need of semantic analysis, Abstract Parse trees, Syntax directed definitions, Syntax directed translation schemes, Symbol Tables, Symbol Table management, Data type as set of values and with set of operations Type Checking, Intermediate code generation: Intermediate languages, Design issues, Intermediate code representations: three address code, abstract & concrete syntax trees.

Code optimization **[8 Hrs]**

Introduction, Principal sources of optimization, Machine Independent Optimization, Machine dependent Optimization, Various Optimizations: Function preserving transformation, Common Sub-expressions, Copy propagation, Dead-code elimination, Loop Optimizations, Code Motion, Induction variables and strength reduction, Peephole Optimization, Redundant –instruction elimination.

Run-Time Memory Management & Code generation **[6 Hrs]**

Model of a program in execution, Stack and static allocation, Activation records, Issues in the design of code generation, Target machine description, Basic blocks & flow graphs, Expression Trees, Unified algorithms for

instruction selection and code generation.

Self-study **[12 Hrs]**

Source-to-Source translation, Operator precedence parsing, CLR and LALR parsers, Conflict Resolution in LR parsers, Symbol table organization techniques, Bootstrapping of compiler, Modern compiler construction tools, register allocation and assignment

Textbooks

1. Alfred V. Aho, Monica S. Lam, A. V. R. Sethi and J.D. Ullman, “Compiler Principle, Techniques and Tools” Addison Wesley, Second Edition, ISBN: 978-0321486813.

Reference Books

1. Barrent W. A., J. D. Couch, “Compiler Construction: Theory and Practice”, Computer Science series, Asian student edition, ISBN: 9780574217653.
2. Dhamdhere D.M., “Compiler Construction Principle and Practice”, Macmillan India, New Delhi, 2003, Second Edition, ISBN: 0333904060.
3. Ravendra Singh, Vivek Sharma, Manish Varshney, “Design and Implementation of Compiler”, New Age Publications, 2008, ISBN: 978-81-224-2398-3.
4. Holub, A.J., “Compiler design in C”, Prentice Hall, 1994, ISBN: 978-0133049572.
5. John Levine, Tony Mason & Doug Brown, “Lex and Yacc”, O‟ Reilly, 1995, Second Edition, ISBN: 978-1565920002.

Laboratory Assignments

1. Design a lexical analyzer for a subset of C language using Lex tool.
2. Design a hand-coded lexical analyzer for a subset of C language, draw the transition diagrams and then implement the lexical analyzer in C language.
3. Design a scientific calculator using Lex & Yacc or PLY or ANTLR tools.
4. Write a code for finding FIRST & FOLLOW of a grammar.
5. Design a SQL parser / html parser.
6. Implement a SLR parser for a given grammar.
7. Implement a static semantics analyzer.
8. Implement an intermediate code generator in three-address code form represented in quadruples.
9. Implement different optimization techniques on intermediate code.

(PCC) Cryptography and Network Security

Teaching Scheme
Lectures: 3 Hrs/ Week
Laboratory: 2 Hrs/ Week
Self-Study: 1 Hrs/ Week

Evaluation Scheme
Theory: TA: 20 Marks, MSE: 30 Marks, ESE: 50 Marks
Laboratory: CIE: 100 Marks

Course outcomes

1. Understand various security services, security attacks and classical encryption techniques
2. Apply the concepts of finite mathematics and number theory.
3. Analyze and implement symmetric key cryptographic algorithms using mathematical foundations.
4. Understand and Implement the asymmetric encryption algorithms, the concept of message integrity and the algorithms for checking the integrity of data.
5. Understand and implement various authentication and web security techniques.
6. Demonstrate the understanding of defence mechanisms against network attacks, and cryptographic protection mechanisms.

Course content

Introduction: **[6 Hrs]**

Cryptography, Need of security, Security services, Basic network security terminology, Security attacks, a model for network security, Classical encryption techniques

Mathematical Foundations: **[6 Hrs]**

Prime Number, relatively prime numbers, Modular Arithmetic, Fermat's and Euler's Theorem, Primitive roots, The Euclidean and Extended Euclidean Algorithms, The Chinese Remainder Theorem

Symmetric Cryptography: **[6 Hrs]**

Symmetric Key Ciphers, Feistel block cipher, Modern Block Ciphers: Data Encryption Standard (DES), Block cipher design principles, Advanced Encryption Standard (AES), Block cipher Operations: Electronic Codebook mode, Cipher Block Chaining mode, Cipher Feedback mode, Counter mode., Cryptanalysis of Symmetric Key Ciphers: Linear Cryptanalysis, Differential Cryptanalysis

Asymmetric Cryptography: **[6 Hrs]**

RSA, Diffie-Hellman Key Exchange, Elliptic Curve Cryptography, Key Management and distribution
Cryptographic Hash functions: Message Digest 5 (MD5), Secure Hash algorithm (SHA), Message Authentication Codes, Digital Signature Algorithm

Authentication and Web Security: **[6 Hrs]**

Authentication Protocols, Kerberos, X.509 Digital Certificates, Pretty Good Privacy, Secure Socket Layer

Network Security: **[6 Hrs]**

Intrusion, types of intruders, Intrusion Detection, IDS, Honeypot, Firewalls, Types of firewalls, Firewall Design Principles.

Self-study **[12 Hrs]**

Vulnerabilities in TCP/IP model, Homomorphic Topography

Textbooks

1. "Cryptography and Information Security", V. K. Pachghare, 3rd edition, PHI Learning, ISBN:· 978-93-89-347-10-4.
2. "Cryptography and Network Security, Principles and Practice", William Stallings, Pearson· Education, Seventh Edition, and ISBN: 978-93-325-8522-5

Reference Books

1. "Network Security the Complete Reference", Robert Bragge, Mark Rhodes, Heith Straggberg· Tata McGraw Hill Publication, ISBN: 9780072226973.
2. "Network Security: Private Communication in a Public World", Charlie Kaufman, Radia· Perlman, and Mike Speciner, Prentice Hall, ISBN 0-13-046019-2.

Web references

1. https://www.geeksforgeeks.org/computer-networks/cryptography-tutorial/
2. http://williamstallings.com/Cryptography/

Laboratory assignments

1. Study research papers on cryptography and network security topic and write a study report on any one of the following topics
    a. Wireless Network Security
    b. Key Exchange Protocols
    c. Block Chain
    d. Quantum Computing
2. Write a program for the cryptanalysis of Caesar cipher and identify the plaintext.
   Implement the Playfair cipher for encryption and decryption. Count the time required for encryption/decryption of messages of different length.
3. Implement Chinese Remainder Theorem for n number of congruences.
4. Implement Simplified DES algorithm for encryption and decryption of any message.
   Count the time required for encryption/decryption of messages of different length.
5. Demonstrate how Diffie-Hellman key exchange works with Man-In-The-Middle attack
6. Using Nmap 1)Find Open Ports On A System 2) Find The Machines Which Are Active 3)Find The Version Of Remote Os On Other Systems 4)Find The Version Ofs/W Installed On Other System
7. Write a program to
    a. Check whether the given number is prime or not?
    b. Find GCD (Greatest Common Divisor) of two numbers.
    c. Check whether given numbers are relatively prime or not?
    d. Find multiplicative inverse of given two numbers
    e. Implement the RSA algorithm
    f. Compute the time required for encryption and decryption.

(PEC) System Administration

Teaching Scheme
Lectures: 3 Hrs/ Week
Laboratory: 2 Hrs/ Week
Self-Study: 1 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks
Laboratory: CIE: 100 Marks

Course outcomes

1. Install various Linux distributions and Windows-based servers and desktop systems on commodity hardware, and carry out basic administration tasks of the user, network, process, storage management.
2. Set up a LAN-based laboratory using DHCP.
3. Install and configure a web server, database server, DNS, NFS, NIS, LDAP based system, secure a Desktop and a server system completely using existing tools.
4. Manage Linux file systems and data storage mechanisms.
5. Provide security administration for Linux.

Course content

Basic System Administration **[8 Hrs]**

Partitioning, Installation of multiple operating systems on desktops, Various operating system services: Cron, CPU utilization, User management, Backup, Log management, Boot loader, Process management, File system namespace, Kernel upgrades.

Network Administration **[8 Hrs]**

Configuration of network hardware, Linux router, Managing routes, DHCP - Dynamic Host Configuration Protocol, DNS-Domain Name Server, NFS - Network File System, NIS - Network Information Service, Email Setup-SendMail, Issues with mail services, POP and IMAP Server.

File system Administration **[8 Hrs]**

Formatting, Partitioning, Managing file systems, Defragmentation, Quotas, Journaling file system, Logical volume management, Disk layouts, File system check, Archiving and compressing files, SAN, NAS, Case Studies: ext2, ext4, ntfs, samba, cifs, lvm, fat32.

Security Administration **[8 Hrs]**

Methods of attack, Network security tools- Nmap, Snort, Nessus, Wireshark/tcpdump, Firewall, IPtables, NAT, IP filtering, Setting up linux firewall, Mandatory access control (MAC), SELinux, Cryptographic security tools Authentication mechanisms, LDAP

Server Administration **[6 Hrs]**

Apache webserver configuration, Database servers: MySQL and PostgreSQL

Devices Administration **[6 Hrs]**

Device resources, udev: Device files, Installing and configuring printers, Managing printers via the web

interface, Scanners, PCI devices, LAN cards, Device troubleshooting, Plug and Play devices.

Self-study **[12 Hrs]**

LDAP, Proxy servers.

Textbooks

1. Evi Nemeth, Garth Snyder, Ben Whaley, Trent R. Hein, "UNIX and Linux System Administration Handbook", Pearson Education, Fourth edition, ISBN-13: 978- 8131761779
2. Richard Petersen, "Linux: The Complete Reference, Sixth Edition", Mcgraw-Hill Education Sixth edition,ISBN-13: 978-0070222946
3. Arnold Robbins, Nelson H. F. Beebe, "Classic Shell Scripting", Shroff/O'ReillyFirst edition, ISBN-13: 978-8173668463
4. Wale Soyinka, "Linux Administration: A Beginner's Guide", McGraw-Hill Osborne Media Publication Sixth Edition, ISBN-13: 978-0071767583
5. Olaf Kirch & Terry Dawson, "Linux Network Administrator's Guide", O' Reilly 2nd Edition June 2000,ISBN-10: 1565924002,ISBN-13: 978-1565924000
6. W.Preston, "Using SANs and NAS",O' Reilly; First Edition, February 2002,ISBN-10: 0596001533,ISBN-13: 978-0596001537

Reference Books

1. Richard Blum, Christine Bresnahan, "Linux Command Line and Shell Scripting Bible", Wiley India Pvt. Ltd., Second edition, ISBN-13: 978-8126533831;
2. Jean-Paul Tremblay, Paul. G. Soresan, "An introduction to data structures with Applications", Tata Mc-Graw Hill International Editions, 2nd edition 1984, ISBN-0-07- 462471-7

Web references

1. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9
2. https://wiki.archlinux.org/title/Main_page
3. https://tldp.org/LDP/sag/html/index.html

Laboratory assignments

1. Set a desktop with following software configuration options: triple boot with Windows, Ubuntu and Fedora operating systems; Grub timeout set to 5 seconds with default Fedora Linux; Each step of boot process secured with passwords; Following software installed in each OS: office, internet browser, c compiler, terminal, ssh server, telnet server, web server, database server.
2. Set up a LAN based computer laboratory with 5 computers - using DHCP first and then using static IPs; one of the machines should work as DHCP server; set up LDAP+NFS based authentication for managing single identity on all computers; setup quotas on NFS systems.
3. Demonstrate use of three-tier architecture using LAMP and WAMP suite using same code base.
4. Setup a disk management system using LVM such that all options of LVM are demonstrated.
5. Write a program to browse an ext4 file system and locate data of a deleted file.
6. Setup a proxy server and firewall with set of policies to block video content in the network.

7. Setup apache web server to serve 3 websites, with 3 domain names at a time from a single machine; demonstrate the use of at least 5 configuration options of apache.
8. Setup an email server and demonstrate use of at least 3 email clients to send email using it.
9. Setup SE Linux.

This list is a guideline. The instructor is expected to improve it continuously.

(PEC) GPU Computing

Teaching Scheme

Lectures: 3 Hrs/ Week

Laboratory: 2 Hrs/ Week

Self-Study: 1 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Laboratory: CIE: 100 Marks

Course outcomes

1. Learn and comprehend the differences in CPU & GPU.
2. Demonstrate the need of GPU computing.
3. Analyze different possible performance issues associated with GPU.
4. Program massively parallel processors in different ways.
5. Apply the practices for different problems.

Course content

Introduction to GPU **[6 Hrs]**

History, graphics processors, graphics processing units, GPGPUs. Clock speeds, CPU / GPU comparisons, heterogeneity, Accelerators, Types of cores, Latency hiding architectures, SIMT programming model, Compute Levels, Different microarchitectures, Basics of GPU architectures.

CUDA Architecture **[6 Hrs]**

CUDA Architecture overview, Setting-up C based CUDA environment, General flow of a simple CUDA program & discussions, Grids, Blocks, Threads, Multi-GPU solutions & Streams, Parallel Patterns, Compiling, Debugging & Thread management, Optimizing CUDA applications with different considerations with case studies.

GPU Memory **[6 Hrs]**

Unified memory, Global memory, Shared memory, Cache structure, Constant ,memory & events, Texure memory, Examples of all memory types, Memory handling with CUDA, Memory consistency. Barriers (local versus global), atomics, memory fence. Prefix sum, reduction. Programs for concurrent data structures.

Programming GPU **[6 Hrs]**

pyCUDA, Simple examples using PyCUDA syntax, Differences in C based CUDA & pyCUDA approaches, Heterogeneous programming with pyCUDA, Memory management with pyCUDA, An overview of pyOpenCL approaches & differences.

OpenACC **[6 Hrs]**

Parallel programming with OpenACC, Basics of OpenACC, Loop level parallelism, Programming tools, Best practices for OpenAcc program development, OpenACC & performance portability, Interoperability.

Applications in different domains **[6 Hrs]**

Additional approaches to parallel programming, Docker-Container, Process and workflow of docker-container usages, Different environments- Tensorflow & Pytorch, Deep Learning Accelerations with CUDA, Examples of different domain datasets & Performance observations.

Self Study : Docker Containers & Kubernetes **[12 Hrs]**

Working flow of Docker Containers, Kubernetes architectures and related issues

Textbooks

1. David Kirk, Wen-mei Hwu, "Programming Massively Parallel Processors: A Hands-on Approach", Morgan Kaufman, 2010, ISBN: 978-0123814722
2. Sunita Chandrasekaran, Guido Juckeland, "OpenACC for Programmers Concepts and Strategies", Addison-Wesley, 2018, ISBN-13: 978-0-13-469428-3
3. Giancarlo Zaccone, "Python Parallel Programming Cookbook", Second Edition, Packt Publishing, 2019, ISBN 978-1-78953-373-6

Reference Books

1. Shane Cook, "CUDA Programming: A Developer's Guide to Parallel Computing with GPUs", Morgan Kaufman, 2012, ISBN: 978-0124159334
2. Jaegeun Han, Bharatkumar Sharma"Learn CUDA Programming", Packt Publishing, 2019, ISBN 978-1-78899-624-2
3. Shane Cook, "CUDA Programming: A Developer's Guide to Parallel Computing with GPUs", Morgan Kaufman, 2012, ISBN: 978-0124159334
4. Jaegeun Han, Bharatkumar Sharma"Learn CUDA Programming", Packt Publishing, 2019, ISBN 978-1-78899-624-2

Web references

1. https://docs.nvidia.com/cuda/doc/index.html
2. https://docs.nvidia.com/dgx/index.html

Laboratory assignments

1. Set-up the CUDA environment in Linux & Windows. Finding out the microarchitecture & compute capability of the available graphics card/ device.
2. A simple CUDA program and discussions to all components.
3. A program to implement vector addition to understand the grid, block & thread components.
4. A program to implement shared memory with multiple blocks and multiple threads.
5. A program to implement constant memory & events with examples like ray tracing.
6. A program to implement texture memory with examples like heat transfer simulation.
7. A simple program to understand the pyCUDA environment with examples like prefix sum.
8. A program to implement a sorting algorithm using pyCUDA syntax.
9. A program to observe the OpenACC flow and simple clauses.
10. A program to implement loop level parallelism using OpenACC
11. A program to understand the working flow of Docker-container concept.

(PEC) Internet of Things

Teaching Scheme

Lectures: 3 Hrs/ Week

Laboratory: 2 Hrs/ Week

Self-Study: 1 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Laboratory: CIE: 100 Marks

Course outcomes

1. Demonstrate the application areas of IOT
2. Analyze data and knowledge management and use of devices in IoT technology
3. Analyze gateways and data management in IoT.
4. Analyze real world IoT design constraints, Industrial automation and commercial building automation in IoT.

Course content

Introduction and concepts: **[8 Hrs]**

Origins, Drivers, Applications, Physical and logical design of IoT, IoT enabling technologies: Wireless sensor networks, Cloud computing, Big data analytics, Communication protocols, embedded systems, IoT levels and deployment templates.

IoT and M2M: **[7 Hrs]**

Introduction to M2M, Difference between IoT and M2M, M2M and IoT technology fundamentals: Devices and gateways, Local and wide area networking, Data management, Business processes in IoT, Everything as a service (XaaS), M2M and IoT analytics, Knowledge management.

IoT Architecture-state of the art: **[7 Hrs]**

Architecture reference model: Introduction, Reference model and architecture, IoT Reference model, M2M to IoT-an architectural overview: Building architecture, Main design principles and needed capabilities, State of the art, Standards considerations.

Data management in IoT: **[6 Hrs]**

Managing M2M data, Data collection and analysis (DCA), Big data, Semantic sensor networks and semantic annotation of data, Virtual sensors, Complex event processing.

Security, Privacy & Trust: **[6 Hrs]**

IoT security challenge, Spectrum of security considerations, Unique security challenges of IoT devices, Internet of things privacy background, Unique privacy aspects of internet of things, Trust for IoT.

Developing IoT solutions: **[6 Hrs]**

Introduction to Python, Introduction to different IoT tools, Introduction to Arduino and Raspberry Pi Implementation of IoT with Arduino and Raspberry, Cloud Computing, Fog Computing, Connected Vehicles, Data Aggregation for the IoT in Smart Cities, Case studies in IoT design.

Self-study **[12 Hrs]**

Advanced IoT Communication Protocols (MQTT, CoAP, LoRaWAN, Zigbee, NB-IoT), Cloud Platforms for IoT (AWS IoT, Microsoft Azure IoT, Google Cloud IoT), Artificial Intelligence and Machine Learning in IoT, Blockchain Technology in IoT Security, IoT Cybersecurity, Ethical Hacking, and Secure Device Management, Mobile App Integration with IoT Systems

Textbooks

1. Vijay Madisetti, Arshdeep Bahga, "Internet of Things: A Hands-On Approach", Universities Press (India) Private Limited, 2016, ISBN: 978 81 7371 954 7
2. Jan Holler, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand & David Boyle "From Machine-to-Machine to the Internet of Things", Academic Press, Elsevier, 2014, ISBN: 978-0-12-407684-6

Reference Books

1. Karen Rose, Scott Eldridge, Lyman Chapin, "The Internet of Things: An Overview", Internet Society, 2015
2. Adrian McEwen, Hakim Cassimally, "Designing the Internet of Things", Wiley, 2014, ISBN 978-1-118-43062-0
3. Daniel Kellmereit, "The Silent Intelligence: The Internet of Things", 2013, ISBN 0989973700

Laboratory assignments

1. **Embedded Programming**
    a. Toggling LEDs
    b. Transmitting a string through UART
    c. Controlling LEDs blinking pattern through UART
    d. Echo each character typed on serial terminal.
    e. Digital IO configuration.
    f. Timer based LEDToggle.
    g. On-chip Temperature measurement through ADC.
2. **RF Experiments**
    a. Point-to-Point communication of two Motes over the radio frequency.
    b. Multi-point to single point communication of Motes over the radio frequency.
3. **Experiments on interfacing with Ubisense**
    a. I2C protocol study
    b. Reading Temperature and Relative Humidity value from the sensor.
    c. Reading Light intensity value from light sensor.
    d. Reading of atmospheric pressure value from pressure sensor.

e. Proximity detection with IR LED.
f. Generation of alarm through BuzzerTransmitting the measured physical value from the UbiSense over the Air.

4. **Experiments on interfacing with Ubi – DAQ**
    a. Timestamp with RTC
    b. IO Expander
    c. Relay control.
    d. I2C based 12-channel ADC
    e. EEPROM read and write
5. **WSN Applications**
    a. Demonstration of a Peer-to-Peer network topology using Coordinator and end device network device types.
    b. Demonstration of Peer-to-Peer communication between Coordinator and end device through Router.
    c. Establishing Many-to-One Communication (Star Network Topology)
    d. Establishing Tree Network Topology
    e. Establishing Cluster Tree Network
6. **IOT applications**
    a. Porting 6LoWPAN stack on Ubimote for enabling it with IPv6
    b. 6LoWPAN network formation with motes and PC
    c. IP based lighting control through Data Acquisition Card
    d. IP based sensor monitoring through Ubi-Sense

Each section of any two assignments should be implemented by using an IoT research lab kit as per Lab instructor.

(PEC) Deep Learning

Teaching Scheme
Lectures: 3 Hrs/ Week
Laboratory: 2 Hrs/ Week
Self-Study: 1 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks
Laboratory: CIE: 100 Marks

Pre-Requisite: Linear Algebra, Partial differentiation, Vector Calculus, Probability Theory, Machine Learning, Python programming language

Course outcomes

1. Demonstrate the knowledge of the fundamentals of neural networks.
2. Design feedforward networks with backpropagation.
3. Evaluate the performance of neural network models.
4. Analyze the performance and optimize the deep learning model by undertaking a case study.

Course content

Neural Networks: **[4 Hrs]**

Biological Neural Network, Artificial Neural Network, Idea of computational units, McCulloch–Pitts unit and Thresholding logic, Linear Perceptron, Perceptron Learning Algorithm, Convergence theorem for Perceptron Learning Algorithm, multilayer perceptrons (MLPs), feedforward neural networks, activation functions.

Feedforward Networks and learning Algorithms: **[6 Hrs]**

Single and multilayer perceptrons (MLPs), representation power of MLPs, sigmoid neurons, feedforward neural network representation, Gradient Descent, Delta rule, Backpropagation, learning weights from more than one node, backpropagating errors from more output nodes, backpropagating errors to more layers, working example of weight update.

Optimization and Regularization: **[8 Hrs]**

Types of errors, bias-variance trade-off, overfitting underfitting, brief review of concepts from optimization, variants of gradient descent, momentum-based methods (Gradient Descent, Batch Optimization, Momentum Based GD, Nesterov Accelerated GD, Stochastic GD, AdaGrad, RMSProp, Adam), Saddle point problem in neural networks. Bias Variance Tradeoff, L2 regularization, Early stopping, Parameter sharing and tying, Injecting noise at input, Ensemble methods, Dropout, Greedy Layerwise Pre-training, Better activation functions, weight initialization methods, Batch Normalization.

Autoencoders: **[6 Hrs]**

Unsupervised Learning with Deep Network, Autoencoders (AEs), Regularization in autoencoders, Types of autoencoders (Denoising, sparse, contractive), Variational Autoencoder.

Convolutional Neural Networks (CNNs) and different CNN architectures: **[8 Hrs]**

Building blocks of CNN, Transfer Learning Techniques: VGGNet, GoogLeNet, ResNet, AlexNet, PixelNet, Visualizing CNNs, Guided Backpropagation.

Recurrent Neural Networks (RNN): **[8 Hrs]**

RNN architecture, Backpropagation through time (BPTT), Vanishing and Exploding Gradients, Truncated BPTT, Long Short-Term Memory network and variants of RNN (Gated Recurrent Units, Bidirectional LSTMs), Encoder Decoder

Models, Attention Mechanism, transformers

Self-study **[12 Hrs]**

Dataset Augmentation, LeNet, AlexNet, DenseNet, Bidirectional RNN, Vector Calculus Review

Textbooks

1. Jacek M. Zurada,"Introduction to artificial neural systems", West Publishing Co.1992, ISBN: 0-3 14-93391 -3
2. Goodfellow, Y. Bengio, and A. Courville, “Deep learning” MIT press, 2016. ISBN 978 0262035613

Reference Books

1. R. Rojas, “Neural networks: a systematic introduction”. Springer Science and Business Media, 2013.
2. C. M. Bishop and N. M. Nasrabadi, “Pattern recognition and machine learning”, vol. 4, no. 4. Springer, 2006. ISBN: 978-0387310732
3. J. Krohn, G. Beyleveld, and A. Bassens, “Deep learning illustrated: a visual, interactive guide to artificial intelligence”. Addison-Wesley Professional, 2019. ISBN: 978-0135116692

Web references

1. DeepLearning.AI (Coursera Specialization): https://www.deeplearning.ai
2. https://onlinecourses.nptel.ac.in/e-learning/preview/noc20_cs62
3. https://nptel.ac.in/courses/106105215
4. https://nptel.ac.in/courses/106106184
5. https://www.datacamp.com/tutorial/exploratory-data-analysis-python
6. https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/
7. CS231n: Convolutional Neural Networks for Visual Recognition (Stanford): http://cs231n.stanford.edu/

Laboratory assignments

1. Write a program to build a logistic regression classifier with a Neural Network mindset. Consider following
    a. Guidelines.
    b. Consider any convenient dataset (Cats dataset etc.) and pre-process the dataset.
    c. Define the appropriate model structure.
    d. Evaluate the model performance.
    e. Analyze the results obtained
2. Implement a multilayer perception (MLP) model for prediction such as house prices.
    a. Perform Exploratory Data Analysis
    b. Prepare dataset
    c. Build MLP model
    d. Evaluate Model performance
    e. Predict for test data
3. Design RNN or its variant including LSTM or GRU
    a. Select a suitable time series dataset. Example – predict sentiments based on product reviews
    b. Apply for prediction
4. Build a Multiclass classifier using the CNN model. Use CIFAR-10/ MNIST or any other suitable dataset.
    a. Perform Data Pre-processing
    b. Define Model and perform training
    c. Evaluate Results using confusion matrix
5. Implement Autoencoders for any of the tasks including:

    a. Data Compression
    b. Image de-noising
    c. Dimensionality reduction
6. Apply a pre-trained network and apply it to a new task using transfer learning
    a. Use any three pre-trained models including AlexNet, GoogleNet, VGGNet, MobileNet, ResNet etc.
    b. Fine-tune the hyper-parameters and compare their performance for a suitable application.
7. Design and implement Deep Convolutional GAN to generate images of faces/digits from a set of given images.
    a. Students will select a project from a provided list or propose their own. The project should involve a complex problem requiring a deep learning solution, with a focus on problem formulation, implementation, and a comprehensive final report.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Note: Those who opt for this subject as an elective cannot take as Honors (can opt for another honors course)

(PEC) Parallel Algorithms

Teaching Scheme

Lectures: 3 Hrs/ Week

Laboratory: 2 Hrs/ Week

Self-Study: 1 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Laboratory: CIE: 100 Marks

Course outcomes

1. To understand different array processors and parallel algorithms for multiprocessors.
2. To Perform the various operation on the PRAM Model
3. To perform merging and sorting operations on different models
4. To solve linear equations using parallel algorithms for basic problems.
5. To study graph Algorithms

Course content

Parallel Algorithms architectural perspective **[6 Hrs]**

Structures and algorithms for array processors: SIMD Array Processors, Interconnection networks, Parallel algorithms for Array processors. Multiprocessor architecture-and Interconnection networks-multiprocessor control algorithms- parallel algorithms for multiprocessors.

Search and Sort **[6 Hrs]**

Selection – broadcast- all sums- parallel selection. Searching a random sequence, sorted sequence on PRAM models, Tree and Mesh

Merging **[6 Hrs]**

Merging – A network for merging – merging on PRAM models. Sorting on a linear array, EREW, CREW and CRCW SIMD models, MIMD Enumeration sort.

Matrix operations **[6 Hrs]**

Matrix operations- Transposition, Matrix by matrix multiplication, matrix by vector multiplication. Numerical problems- solving systems of linear equations, finding roots of nonlinear equations on PRAM models.

Graphs **[6 Hrs]**

Graphs – Connected components- dense graphs- sparse graphs. Minimum spanning tree- Solli's algorithm, Biconnected components, Ear decomposition, Directed graphs

Applicability and Case Studies **[6 Hrs]**

Weather forecasting, Drug discovery, Medical Imaging etc.

Self-study: Applications and Performance Improvement **[12 Hrs]**

Different aspects of performance improvement through parallel algorithmic design

Textbooks

1. S. G. Akl, “Design and Analysis of Parallel Algorithms”, Prentice Hall Inc., 1992.
2. Selim G. Akl “Parallel Sorting Algorithms”, Academic Press, 1985.

Reference Books

1. Joseph Jaja, “An Introduction to Parallel Algorithms”, Addison Wesley, 1992.
2. Herlihy, M., Shavit, N., Luchangco, V., & Spear, M., The art of multiprocessor programming, Second Edition, 2021, Morgan Kaufmann.
3. Michael McCool, James Reinders, Arch Robison, Structured Parallel Programming: Patterns for Efficient Computation, 2012, Morgan Kaufmann.
4. Peter S. Pacheco, “An Introduction to Parallel Programming”, Morgan Kaufmann, Elsevier Series, 2011, ISBN: 978-0-12-374260-5.

Web references

1. https://www.cs.purdue.edu/homes/ayg/book/Slides/
2. https://ftp.utcluj.ro/pub/users/civan/CPD/3.RESURSE/9.Book_2018_Introduction%20to%20ParallelComputing_Trobek.pdf

Laboratory assignments

1. Implementation of Searches
2. Implementation of Sorting Algorithms
3. Implementation of Matrix based and Graph Algorithms

This is a suggested list. The instructor is expected to continuously update it.

(PEC) Multicore Technology

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

1. Learn & analyze the working principles of multicore architecture.
2. Optimize performance of multicore systems.
3. Specify the necessity of SMP & AMP.
4. Comprehend and differentiate between SMP & AMP.
5. Identify and demonstrate the need of domain specific architectures.

Course content

Introduction to Multicore Systems **[6 Hrs]**

Fundamentals, The Era of Multicore Machines, Unicore vs Multicore - Understanding Performance, Shared memory Multicore Systems, Distributed Memory Multicore Systems, Hybrid Systems Symmetric and Asymmetric Multicore Systems, Overview of Multithreading, Multithreading in different forms, Homogeneous and Heterogeneous Multicore systems, Examples of different Multicore Systems

Cache Memory **[6 Hrs]**

Large Cache Design: Shared vs. Private Caches, Centralized vs. Distributed, Shared Caches, Coherence: Snooping-based cache coherence protocol, directory-based cache coherence protocol, Uniform Cache Access, Non-Uniform Cache Access, S-NUCA, D-NUCA, Inclusion, Exclusion, Examples of different Cache Organization, Consistency Models, Case Study

Performance and Optimizations for Multicore Systems **[6 Hrs]**

Select the right “core” - Improve serial performance - Achieve proper load balancing - Improve data locality - Reduce or eliminate false sharing - Use of affinity scheduling - Lock granularity and frequency - Remove synchronization barriers - Minimize communication latencies - Use of thread pools - Managing thread count - Use of parallel libraries.

Programming Multicore Systems **[6 Hrs]**

Programming models for Multicore Systems – Shared Memory Programming using pthreads - Shared Memory Programming using OpenMP – Use of OpenMP compiler directives – #pragma with different clauses – Understanding parallelized loops – Synchronization Constructs towards dependencies – Function parallel program - OpenMP Library Functions, OpenMP Environment Variables, Compilation, Debugging, Performance

Domain Specific-Architectures **[6 Hrs]**

Guidelines for domain specific architectures – Deep Learning Architecture - Google's Tensor Processing Unit (TPU) for Deep Neural Networks (DNNs) - Pixel Visual Core, a Personal Mobile Device Image Processing Unit

Open source Simulators **[6 Hrs]**

Different types of simulators for architectures, Classification parameters, Different modes of simulators, Building benchmarks and running sample codes to observe different parameters, Case Study: Any one of the simulator like gem5 or Tejas

Self-study: Secured architecture practices **[12 Hrs]**

Hardware security approaches

Textbooks

1. Gerassimos Barlas, "Multicore and GPU Programming: An Integrated Approach", Morgan Kaufmann, 2015, ISBN: 978-0-12-417137-4.
2. Rob Oshana, "Multicore Application Development Techniques: Applications, Tips and Tricks", Elsevier, 2016, ISBN: 978-0-12-800958-1.
3. John L Hennessy, David A Patterson, "Computer Architecture: A Quantitative Approach", Sixth Edition, Morgan Kaufmann, 2018, ISBN: 978-0-12-811905-1.

Reference Books

1. Rajeev Balasubramonian, Norman P. Jouppi, and Naveen Muralimanohar, "Multicore Cache Hierarchies", Morgan & Claypool Publishers, 2011,ISBN: 9781598297546.
2. Daniel J. Sorin, Mark D. Hill, David A. Wood "A Primer on Memory Consistency and Cache Coherence", Morgan & Claypool Publishers, 2011, ISBN: 9781608455652.

Web references

1. https://ftp.utcluj.ro/pub/users/civan/CPD/3.RESURSE/9.Book_2018_Introduction%20to%20ParallelComputing_Trobek.pdf

A few research papers reading to analyze the performance related issues

(PEC) Distributed Systems

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

Students will be able:

1. Explain the fundamental concepts, architecture, communication models, and design issues of distributed systems.
2. Analyze distributed algorithms related to synchronization, mutual exclusion, deadlock handling, and clock coordination.
3. Apply distributed system models, inter-process communication techniques, and process management concepts in problem solving.
4. Examine distributed shared memory, consistency models, fault tolerance, recovery mechanisms, and distributed file systems.
5. Explore emerging trends and applications in distributed systems, including cloud computing, peer-to-peer systems, and distributed object-based systems.

Course content

Introduction **[6 Hrs]**

Motivation, Goals, Advantages and disadvantages of distributed systems, Design issues in distributed systems, Middleware, Overview of distributed systems, Message passing: Desirable features of good message passing systems, Issues in IPC by message passing.

Communication **[6 Hrs]**

Client server model, Relation of Network models with distributed system (TCP/IP, OSI, ATM etc.) , Remote Procedure Call, group communication and its protocol (IS-IS)

Synchronization **[8 Hrs]**

Clock synchronization, Logical clocks, Lamport's algorithm, Global state, Vector algorithm, Election algorithms, Mutual exclusion algorithms, Dead locks in distributed systems, Deadlock avoidance, Prevention and Detection.

Distributed Systems Models **[8 Hrs]**

Threads, system models, processor allocation, workstation model, Processor Pool Model, Hybrid Model, real time distributed systems, time triggered systems, event driven systems, distributed shared memory, consistency models, page based distributed shared memory.

Distributed File System **[6 Hrs]**

Design, Implementation, & Trends in distributed file systems, File-Accessing Models, File-Sharing Semantics, File-caching Schemes, File Replication, Fault Tolerance.

Applications and Emerging Trends in Distributed Systems **[6 Hrs]**

Distributed object based systems, Distributed co-ordination based systems, Distributed document based systems.

Self-study **[12 Hrs]**

Issues in IPC by message passing, Load balancing approach, Load sharing approach, Process migration, Google File System (GFS), Hadoop Distributed File System (HDFS), Case Studies of Modern Distributed Systems.

Textbooks

2. S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*. New Delhi, India: PHI, ISBN: 978-81-203-2215.
3. P. K. Sinha, *Distributed Operating Systems: Concepts and Design*. New Delhi, India: PHI Publications, ISBN: 978-81-203-1380-4.

Reference Books

1. S. Tanenbaum, *Distributed Operating Systems*. New Delhi, India: Pearson Education, ISBN: 978-81-7758-179-9.
2. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, *Distributed Systems: Concepts and Design*, 5th ed. Harlow, England: Pearson Education, 2011, ISBN: 978-0-13-214301-1.
3. D. L. Galli, *Distributed Operating Systems: Concepts and Practice*. Upper Saddle River, NJ, USA: Prentice-Hall, 2000, ISBN: 978-0-13-080599-7.
4. S. Mullender, *Distributed Systems*, 2nd ed. Reading, MA, USA: Addison-Wesley, 1993, ISBN: 978-0-201-62427-6.

Web References
https://pages.cs.wisc.edu/~remzi/OSTEP/

(PEC) Cyber Security

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

Students will be able to:

1. Define the need of Cyber Security.
2. Explain the IT act, Application Security vulnerabilities and its mitigation techniques.
3. Demonstrate the knowledge of penetration testing, and social networking security.
4. Analyse the malwares, social networking websites and impact of cyber-crime on e-commerce

Course content

Introduction **[6 Hrs]**

Nature and scope of computer crime, Understanding how cyber criminals and hackers work, Different types of cyber-crimes, Introduction to digital signatures, Cryptography, Digital certificate and public key infrastructure, IT Act.

Malware reverse engineering **[6 Hrs]**

Overview of malware reverse engineering, Types of malware, Malicious code families, Latest trends in malware analysis, Basic static and dynamic analysis, Malware analysis techniques, Case study.

Web application security **[8 Hrs]**

Introduction to web application security: Attacks, vulnerabilities and mitigation, Client-side security, Server-side security, Application security: HTTPS, HSTS etc., Security engineering: Passwords and their limitations, Attacks on passwords: CAPTCHA, OTP.

Advanced security topics **[6 Hrs]**

Secure email systems: PGP, SMIME, DKIM, DMARC, DNSSec, SMTP STS etc., Privacy and security for online social networks, Database security, Browser security, Mobile device security.

Ethical hacking and penetration testing **[8 Hrs]**

Security Technologies: IDS, IPS, Ethical hacking, Penetration testing fundamentals: Reconnaissance, scanning, gaining access, maintaining access, Covering tracks.

Case studies **[8 Hrs]**

Cloud security, Operating system security, Security of social networking websites, IoT devices security, E-commerce websites security.

Self-study **[12 Hrs]**

Impact of cyber-crime on e-governance and e-commerce, Network Malware Analysis and Anti-Analysis

Techniques, OWASP Top 10 and Web Security Tools, Scanning and Enumeration, Exploitation Frameworks and Scripting, Mobile and IoT Security

Textbooks

1. Hossein, “Handbook of Information Security, Threats, Vulnerabilities, Prevention, Detection, and Management”, Wiley, Volume 3 edition, ISBN-13: 978-0470323069.
2. Georgia Weidman, “Penetration testing: A Hands-On Introduction to Hacking”, No Starch Press, 2014, ISBN-13: 978-1593275648.
3. Michael Sikorski and Andrew Honig, “ Practical Malware Analysis”, No Starch Press, 1st Edition, 2012, ISBN-13: 978-1593272906

Reference Books

1. “Practical Internet of Things Security” by Brian Russell, Drew Van Duren, Packt publishing, 2016, ISBN: 9781785889639
2. T. Mather, S. Kumaraswamy, S. Latif, “Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance”, O'Reilly Series, 2009, ISBN-13: 978-0596802769.
3. “Cyberlaw: the Indian perspective”; Pavan Duggal; Saakshar Law Publications, 1st edition, 2002, ISBN: 8189121022, 9788189121020.

Web references

1. https://www.cybrary.it/
2. https://portswigger.net/web-security
3. https://owasp.org/
4. https://www.nist.gov/cyberframework
5. https://www.netacad.com/

(PEC) Natural Language Processing

Teaching Scheme
Lectures: 3 Hrs/ Week
Laboratory: 2 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks
Laboratory: CIE: 50 Marks, ESE: 50 Marks

Course outcomes
Understand linguistic structures and language processing techniques.

1. Apply basic text pre-processing techniques for processing natural language text data.
2. Apply and analyze text representation techniques for transforming unstructured language data into machine-readable formats.
3. Implement POS tagging and parsing methods to perform syntactic analysis of language.
4. Design and implement Word Sense Disambiguation methods using lexical knowledge resources.
5. Implement relation extraction techniques to identify semantic relationships between entities in text.

Course content

Introduction to NLP and Morphology **[4 Hrs]**

What is NLP, Fundamental and Scientific goals, Engineering goals, Levels of Language Analysis, Ambiguity in Natural Language, Applications of NLP, Empirical Laws of language, Zipf's law, Heap's law. Morphology-morphemes, types, Inflectional and Derivational morphology.

Basic Text Processing **[8 Hrs]**

Basic Text Processing- Tokenization, word tokens, types of tokenization -Sentence, word, character, sub-word level., Tokenization algorithms - byte pair encoding, word-piece, sentence-piece, unigram language modeling., Issues in tokenization for different languages, Text normalization, Stemming, lemmatization, Porter Stemmer, stop word removal, Smoothing techniques, Spelling Correction-minimum edit distance, N-gram Language Modeling-probabilistic language model, Evaluation of Language Model -Perplexity, Cross-Entropy Loss, Accuracy, Blue Score, Rough Score.

Text Representation **[4 Hrs]**

Text Representation- Need of text representation, Vector Space Model, types of representations- one-hot vectors, BOW (bag of words), TF-IDF (term frequency-inverse document frequency), n-gram representation, word embedding- Skip gram , CBOW, Glove, Fast-text models, contextual embeddings.

POS Tagging **[6 Hrs]**

Sequence labeling tasks of NLP, POS tagging, POS tag sets, Introduction to Hidden Markov Models (HMM), POS tagging algorithms - Viterbi Algorithm, BaumWelch, Maximum Entropy Model, Conditional Random Field.

Syntax and Parsing **[10 Hrs]**

Syntax Parsing, Context Free Grammar, Parsing strategies, Constituency based parsing, Dynamic Programming parsing methodologies- CKY (Cocke Kasami Younger), Chart Parser, Earley Parsers, Issues in parsing,
Probabilistic Parsing –PCFG, Inside Algorithm, inside-outside probabilities.
Dependency parsing- Dependency structure, dependency graphs, formal conditions on dependency graphs, dependency relations, dependency parsing methodologies- Arc Eager Parsing, Transition-Based Dependency Parsing, Classifier based dependency parsing, Evaluation

Semantics and Relation Extraction **[8 Hrs]**

Semantics- Introduction to Semantics, Ontology, WordNet: Word Senses, Word relations, Word similarity and thesaurus methods, Word Sense Disambiguation, Novel Word Sense Detection Algorithm, Applications. Relation Extraction - Introduction, named-entity recognition, types of relations, relation extraction algorithms- using patterns, supervised learning.

Self-study **[12 Hrs]**

Ontologies types, co-reference resolution techniques, named entity extraction techniques, mention detection, research on Marathi and Hindi Indic languages.

Textbooks

1. Daniel Jurafsky and James H. Martin, "Speech and Language Processing", Second Edition, Prentice Hall, 2008, ISBN: 978-0131873216.
2. Allen James, "Natural Language Understanding", Second Edition, Benjamin/Cumming, 1994, ISBN: 978-0805303346.
3. Chris Manning and Hinrich Schuetze, "Foundations of Statistical Natural Language Processing", MIT Press, ISBN: 978-0262133609

Reference Books

1. Journals: Computational Linguistics, Natural Language Engineering, Machine Learning, Machine Translation, Artificial Intelligence.
2. Conferences: Annual Meeting of the Association of Computational Linguistics (ACL), Computational Linguistics (COLING), European ACL (EACL), Empirical Methods in NLP (EMNLP), Annual Meeting of the Special Interest Group in Information Retrieval (SIGIR), Human Language Technology (HLT)

Web references

1. NPTEL course on NLP offered by Prof. Pawan Goyal (IIT Kharagpur) - (https://onlinecourses.nptel.ac.in/noc19_cs56/)
2. Stanford course on Speech and Language Processing (https://web.stanford.edu/~jurafsky/slp3/)

Laboratory assignments

1. Basic Text preprocessing and Normalization of Social Media Data.
2. Implementation of Spell checking Algorithm .
3. Building n-gram language models.
4. Implement the Viterbi Decoding/ MEM Algorithm for Parts of Speech tagging.
5. Building a dependency parser for English Language input sentence.
6. Building word embeddings for English/Hindi/Marathi language.
7. Using Hindi/ Marathi WordNet for Word sense disambiguation of Hindi/Marathi Language Sentence.
8. To extract semantic relationships between entities given input text data.
9. Mini-project

(PEC) Graphics and Multimedia

Teaching Scheme
Lectures: 3Hrs/Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

1. Understand and apply the fundamental concepts of computer graphics including raster scan graphics, polygon processing, transformations, and clipping techniques.
2. Design and implement graphics algorithms such as line drawing, circle generation, polygon filling, curve generation, and hidden surface removal techniques.
3. Perform 2D and 3D geometric transformations, viewing transformations, and projections for graphical object manipulation and visualization.
4. Apply lighting, shading, color models, reflections, transparency, and ray tracing techniques to generate realistic computer-generated images.
5. Use multimedia technologies and commercial tools for media preparation, integration, communication, and entertainment applications.

Course content

Raster Scan Graphics: [6 Hrs]

Introduction to computer graphics, lines, line segments, vectors, pixels and frame buffers, vector generation. DDA and Bresenham's line and circle drawing algorithms, antialiasing, thick lines. Character generation: Stroke Principle, Starburst Principle, Bit map method, display of frame buffer.
Polygons: Introduction, representation, entering Polygons, Polygon filling: Seed fill, Edge fill, scan conversion algorithm, filling with patterns.

Transformations: [6 Hrs]

Basic 2D & 3D transformations - Translation, Scaling, Rotation, Reflection, Shearing, Multiple Transformations, Rotation about an axis parallel to a coordinate axis, Rotation about an arbitrary axis in space, Affine and Perspective Geometry, Orthographic projections and Axonometric projections

Viewing and Clipping: [6 Hrs]

Segments: Introduction, segment table, segment creation, closing, deletion, renaming. Image transformations, raster techniques.
Windowing and clipping: Introduction, viewing transforms, 2D clipping, Cohen-Sutherland algorithm, Midpoint subdivision algorithm.

Curves and Surfaces [6 Hrs]

Curve Representation, Non-parametric and parametric curves, representation of space curves, Cubic Spline, Parabolic Blended curves, Bezier curves and B-spline curves, Z- buffer, Warnock algorithm. Fractals, fractal lines and surfaces.

Light, Color and Shading: [6 Hrs]

Introduction, Diffuse illumination, point-source illumination, Specular reflection. Shading algorithms, transparency, reflections, shadows, ray tracing, Colour models and tables.

Multimedia Applications **[6 Hrs]**

Media preparation, composition, integration, communication, entertainment using commercial tools

Self-study **[12 Hrs]**

Cyrus Beck algorithm, Interior and Exterior clipping, Text Clipping. Polygon Clipping, Sutherland-Hodgman algorithm, Generalized clipping. Hidden Surface removal

Textbooks

1. D. Hearn, M. Baker, "Computer Graphics – C Version", 2nd Edition, Pearson Education, 2002, ISBN 81 – 7808 – 794 – 4
2. V. K. Pachghare, "Computer Graphics", 2nd edition, Laxmi Publication, 2007, ISBN 81– 318–0061 – X

Reference Books

1. J. Foley, Van Dam, S. Feiner, J. Hughes, "Computer Graphics Principles and Practice", 2nd Edition, Pearson Education, 2003, ISBN 81 – 7808 – 038 – 9
2. "Procedural Elements for Computer Graphics" by David F. Rogers, McGraw-Hill,

Web References

1. https://www.geeksforgeeks.org/computer-graphics-2/
2. https://www.scratchapixel.com/
3. https://www.tutorialspoint.com/computer_graphics/index.htm

(PEC) Embedded Systems

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

1. Explain Characteristics and recent trends in Embedded Systems
2. Discuss PIC and ARM families
3. Explain the communication interface for wired and wireless protocols
4. Design a system using the concepts of RISC architecture and ARM processors
5. Decide the hardware and/or software components best suited for a given problem in context of prototype development

Course content

Overview of Embedded Systems **[6 Hrs]**

Introduction, Definition, Characteristics & Salient Features, Classification, Application Areas, Overview of Embedded System Architecture & Recent Trends.

Hardware Architecture **[6 Hrs]**

Embedded Hardware based on Microprocessors, Microcontrollers & DSPs. Study of PIC Microcontrollers: PIC16C6X/7X Family & Applications. Study of ARM Family: ARM 7,9,10 &11: Overview & Architecture Comparison, Detailed Study of ARM7-TDMI including Core Architecture, ARM/Thumb State, On-Chip Debug and Development Support, AMBA Bus, Applications.

Communication Interface **[6 Hrs]**

Serial, Parallel, Wired Wireless Protocols Wired: CAN, I2C, USB, FireWire Wireless: BlueTooth, IrDA, IEEE802.11.

Software Architecture **[6 Hrs]**

Concepts: Embedded OS, Real-Time Operating Systems (RTOS), Detailed Study of RT Linux, Hand Held OS, Windows CE. & Development Tools.

Embedded Systems for Automotive Sector **[6 Hrs]**

Electronic Control Units (ECU) for Engine Management, Antilock Braking System (ABS), Crusie Control, Design Challenges, Legislative Emission Norm, Interface Standards, Developmental Tools Navigation Systems: Global Positioning System (GPS): Detailed Study and Applications.

Smart Cards **[6 Hrs]**

Classifications, Interfacing, Standards & Applications. RFID Systems: Technology, RFID Tag, RFID Reader, Applications.

Self-study **[12 Hrs]**

Embedded System for Mobile Applications, DSP Based Embedded System, Networked Embedded System and Digital Camera.

Textbooks

1. K.V.K. Prasad, Embedded / Real Time Systems: Concepts, Design and Programming Black Book, Dreamtech Press, 2005.
2. Vahid F. and Givargies T., Embedded Systems Design, John Wiley X. Sons, 2002

Reference Books

1. Daniele Lacamera, Embedded Systems Architecture: Design and write software for embedded devices to build safe and connected systems, Second Edition 2024
2. J. Staunstrup, Wayne Wolf, “Hardware/Software Co-Design: Principles and Practice”, Prentice Hall.

Web references

1. https://users.ece.utexas.edu/~valvano/Volume1/IntroToEmbSys/

(PEC) Storage & Virtualization

Teaching Scheme
Lectures: 3 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes
Students will be able to:

1. Apply, compare and contrast basic concepts of hardware storage, NAS-SAN, virtualization concepts.
2. Analyze cloud computing setup with its vulnerabilities and applications using different architectures.
3. Assess cloud Storage systems and cloud security, the risks involved, its impact and develop suitable cloud application.
4. Design and implement Virtualization environment and apply cloud resource management.

Course content

Basic concepts for Systems and Storage **[6 Hrs]**

Storage Challenges, Issues- Data sources, challenges of data growth and data availability, Performance and manageability requirements, Data virtualization, OS and device drivers - Kernel, firmware, RDMA, boot sector, device partitioning, UNIX file system, virtual memory, namespace, metadata, buffer cache, defragmentation.

Storage Hardware and SCSI Protocol **[6 Hrs]**

Storage hardware and SCSI Protocol, SAN, NAS, Fibre Channel and iSCSI, Storage Hardware Building Blocks - Device types, HBA, switches, hubs, routers, GBIC, Introduction to various Storage protocols, Overview of IDE, SAS, SATA, SCSI, FC, FCoE, iSCSI, Infiniband, FCP, FC-IP, iFCP, Fibre Channel Protocol Stack & Concepts- FC ,Mapping Protocols - iSCSI, FCP- SCSI mapping to underlying transport, Connection Management, PDU,TOE.

SAN & NAS **[6 Hrs]**

Fibre Channel Protocol Stack & Concepts- FC (Protocol stack, Exchange, Sequence, Frames, Port types, Topologies, Login, FC-ID) Mapping Protocols - iSCSI, FCP- SCSI mapping to underlying transport, Connection Management, PDU, TOE SAN Concepts and Advanced Topics- DAS, SAN architecture, Concepts (zoning, name server, SCN, WWN, routing), FC-SAN, IP-SAN and Applications NAS Concepts and Appliances- NAS architecture, Protocols (CIFS, NFS), Performance, Scalability and Usability, Appliances.

Storage Virtualization Concepts: **[6 Hrs]**

Data Centre end-to-end view, Overview of stack, clustering, virtual Machines, cloud storage and virtualization, RAID levels, I/O stack, OS abstraction, Storage Pooling, Storage Provisioning, Online Grow/Shrink Storage Virtualization, Metadata management, Transaction consistency, I/O maps, I/O path considerations, Data consistency, Crash recovery, Application interfaces, Linux container.

Applications and Use Cases for Storage Virtualization **[6 Hrs]**

Storage Virtualization use cases, applications, data replication, RPO/RTO, Snapshots, Data protection, Restore, Archival, Compliance considerations. Capacity Management, Storage provisioning, De-duplication, thin provisioning, Storage Tier, ILM, Data classification, Storage grid.

Cloud Computing **[6 Hrs]**

Virtualized environments features, taxonomy of virtualization techniques, virtualization services and applications, Cloud Computing mechanisms, Cloud architecture & Services, MapReduce programming model, Cloud resource management, Cloud Platforms in industry, Case study of AWS, Google Cloud, Microsoft Azure and IBM Softlayer.

Self-Study **[12 Hrs]**

Advanced Storage Management, Data Protection and Disaster Recovery, Storage Optimization Techniques, Cloud Storage and Virtualization, Emerging Trends in Storage Systems, Industrial Cloud Platforms.

Textbooks

1. Storage Networks: The complete Reference. Robert Spalding TMH.
2. Designing Storage Area Networks: A Practical Reference for Implementing Fibre Channel and IP SANs, Second Edition Publisher: Addison-Wesley Author: Tom Clarks.
3. Cloud Computing: Concepts, Technology & Architecture, Thomas Erl, Prentice Hall, 2013.

Reference Books

3. Cloud Computing: Theory and Practice – Latest available edition: 2017 (2nd Edition), Morgan Kaufmann/Elsevier.
4. Computer Systems: A Programmer's Perspective – Latest edition: 3rd Edition (2015**)**, Pearson.
5. Cloud Computing: Principles and Paradigms, Wiley Series on Parallel and Distributed Computing, Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, John Wiley and Sons, 2011,ISBN 978-0-470-88799-8.
6. Cloud Computing: Theory and Practice, Dan C Marinescu, Elsevier, 2013.
7. Cloud Computing Bible, Barrie Sosinsky, Wiley Publishing, 2011. · Virtualization Essentials, Matthew Portnoy, John Wiley and Sons, 2012.
8. Computer Systems - A Programmer‟s Perspective, Randall Bryant and David O‟Hallaron Pearson Education, 2003.
9. The Design and Implementation of the 4.4 BSD Operating System, McKusick, Bostic, Karels, Quaterman, 1996.

Web references

1. AWS Storage & Cloud Documentation
   Covers cloud computing, storage systems, virtualization, backup, replication, and scalability.
2. Linux Kernel Documentation
   Reference for kernel, device drivers, file systems, virtual memory, and boot concepts.
3. SNIA Storage Networking Tutorials
   SAN, NAS, RAID, Fibre Channel, iSCSI, storage protocols, and data protection.
4. VMware Documentation
   Storage virtualization, clustering, snapshots, HA, multipathing, and cloud infrastructure.
5. Kubernetes Documentation
   Distributed systems, containers, orchestration, scalability, and cloud-native architecture.

(PEC) Computer Forensics and Recovery

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

Students will be able to:

1. Identify legal issues of performing digital forensics based on investigator duty/position.
2. Discuss various digital forensics techniques.
3. Analyze different file system used in different operating system.
4. Apply various tools during real world forensics investigation.
5. Aware of the state of the practice and gaps in the technology, policy and legal issues.

Course content

Issues in Digital Forensics: [6 Hrs]

Computer Crime and Law, Legal Aspects of Digital Forensics, Forensic Investigation Procedure, Investigation Techniques, Digital Forensic Evidence, Anti-forensics, Computer Forensic Model, Maintaining Professional Conduct, Data recovery commands , Forensic tools: ProDiscover, SluethKit, CAINE.

Forensic Evidence and Investigations [6 Hrs]

Functions, Categorization, Order of Volatility, Admissibility of Evidence, Acquisition and seizure of evidence, Chain of Custody, Storage formats, Multimedia Forensics: Image Capturing Process, Image Validation, Steganography, Tools for Multimedia Forensics.

MS Windows Forensics [8 Hrs]

Windows artifacts, Program Execution artifacts, Windows Registry, Structure, Registry Analysis Tools, Taskbar Jump Lists, Automatic Destination, Custom Destination, Jump List Extract tools: Structured Storage Viewer, Windows Event Logging Service, Events Structure, Eventvwr Tool, Volume Shadow Copies, Analysis Tools, Windows Shell Bags, BagMRU keys, Prefetch Files, File Deletion, Recovery Mechanisms.

Windows File Systems [6 Hrs]

Clusters and Sectors, FAT File System, FAT Boot Sector, Interpretation using WinHex, FAT Directories, File Allocation Table, File Slack, New Technology File System (NTFS), Comparison to FAT, NTFSWalker tool, Partition Boot Sector, Boot Sector in WinHex, Master File Table (MFT), MFT File Attributes, Directory Files (Index Nodes), $INDEX_ROOT, NTFS Encrypting File System (EFS), Whole Disk Encryption, NTFS Compressed Files

Linux File System: [8 Hrs]

Examining Linux File Structures, Ext4, Superblocks, Directory entries, Inodes, Data blocks, Acquiring file system images using dd, dcfldd, Write blocking options, Mounting images, Leveraging The Sleuth Kit (TSK) and Autopsy, fsslat, mmls, Forensic data from /etc, /usr, /var, /dev, /proc, Timeline Analysis

Network Forensics: [8 Hrs]

Email Structure, Email Server Examination, Tracing emails, Email Forensics Tools, Cloud Forensics and Cloud Forensics Tools.

Self-study **[12 Hrs]**

Digital Evidence, Computer Forensic Investigation Process, Data Acquisition Techniques, Linux Forensics: Log analysis, Shell history, User activity tracking, and File permissions, Memory and Malware Forensics, Email and Browser Forensics, Mobile and Cloud Forensics (Introductory).

Textbooks

1. “Guide to Computer Forensics and Investigations”, Bill Nelson, Amelia Phillips, Christopher Steuart, 6th Edition, Course Technology, Cengage Learning, ISBN-13: 978-1-435-49883.
2. "Digital Forensic: The Fascinating World of Digital Evidences", Nilakshi Jain and Dhananjay R. Kalbande, Wiley 2016, ISBN 879-94756567

Reference Books

1. “File System Forensic Analysis”, Brian Carrier, Pearson education, 1st Edition, ISBN13:978-0321268174.
2. “Handbook of Digital Forensics and Investigation”, E. Casey, Academic Press, 1st Edition, 2010, ISBN-13: 978-0123742674.
3. "Cyber Forensics", Dejey and S. Murugan, Oxford University Press, ISBN 9780199489442.Dejey, Murugan, Cyber Forensics, Oxford Higher Education, 2018

Web references

1. https://github.com/asiamina/a-course-on-digital-forensics
2. https://www.nist.gov/itl/ssd/digital-forensics
3. https://onlinecourses.swayam2.ac.in/e-learning/preview/nou25_cs05
4. https://onlinecourses.swayam2.ac.in/e-learning/preview/cec20_lb06

(PEC) Generative Artificial Intelligence

Teaching Scheme
Lectures: 3 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes
Students will be able to:

1. Understand the principles of Generative AI models.
2. Utilize prompt engineering, fine-tuning to customize large language models.
3. Evaluate the performance, robustness, and quality of generative models using appropriate metrics
4. Apply generative models for text, image, and multimodal data generation.
5. Design Generative AI applications using APIs and frameworks.

Course content

Foundations of Generative AI **[6 Hrs]**

Introduction; Generative vs. Discriminative Models; Probabilistic Generative Models; Challenges of Generative modelling; Taxonomy of Modern Generative Models — GANs, VAEs, Normalizing Flows, Diffusion Models, LLMs

Variational Autoencoders and Normalizing Flows **[6 Hrs]**

Autoencoders, VAE Architecture - Latent variable models; Nonlinear latent variable model; Training, ELBO properties, Variational approximation, reparameterization trick; Types of VAE; Applications of VAE
Normalizing Flows — Change of Variables, Jacobian Determinant; coupling layers; Types of flows; Other Normalizing Flow Models

Generative Adversarial Networks **[8 Hrs]**

GAN Framework, GAN Training - Components, Challenges; DCGAN; Wasserstein GAN; Conditional GAN; CycleGAN; Progressive GAN; StyleGAN; Data Augmentation and Semi-supervised GANs; GAN Applications; GAN Evaluation

Diffusion Models **[6 Hrs]**

Denoising Diffusion Models; Forward Diffusion Process, Reparameterization Trick, Diffusion Schedules, Reverse Diffusion Process, Training the Diffusion Model, Sampling and Analysis of the Diffusion Model

Large Language Models and Prompt Engineering **[8 Hrs]**

LLM Tokenization, Token Embeddings; Transformer Architecture and Models; Pretrained language models; Modern LLM Families;
Prompt Engineering - Basic Ingredients of a Prompt, Instruction-Based Prompting; Complexity of a Prompt; Chain Prompting; Reasoning with Generative Models - Chain-of-Thought, Self-Consistency, Tree-of-Thought; Output verification

Multimodal Generation and RAG **[8 Hrs]**

Transformers for Vision, Multimodal Embedding Models, CLIP; Making Text Generation Models Multimodal - BLIP-2; Preprocessing Multimodal Inputs - Image Captioning, Multimodal Chat-Based Prompting;
Semantic Search with Language Models, Dense Retrieval, Reranking; RAG - Overview, Advanced RAG Techniques, Applications

Self-study **[12 Hrs]**

Monte Carlo methods, importance sampling, and variational inference beyond ELBO; Energy-Based Models — contrastive divergence, Boltzmann machines, their relationship to diffusion; Hugging Face Ecosystem Walkthrough — Diffusers, PEFT, TRL, LangChain, LlamaIndex; practical environment setup; Agentic AI systems

Textbooks

1. Foster, D. (2023). Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play (2nd ed.). O'Reilly Media.
2. Alammar, J., & Grootendorst, M. (2024). Hands-On Large Language Models: Language Understanding and Generation. O'Reilly Media.

Reference Books

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
2. Prince, S. J. D. (2023). Understanding Deep Learning. MIT Press.
3. Chollet, F. (2021). Deep Learning with Python (2nd ed.). Manning Publications.

Web references

1. MIT 6.S978 — Deep Generative Modeling (MIT OpenCourseWare) - https://mit-6s978.github.io/
2. NPTEL — Introduction to Large Language Models (IIT Madras, 2024) - https://nptel.ac.in/courses/106106220
3. Generative AI with LLMs (Coursera) - https://www.coursera.org/learn/generative-ai-with-llms

(PEC) Software Design Patterns

Teaching Scheme
Lectures: 3 Hrs/Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes
Students will be able to:

1. Understand the principles of scalability and maintainability in Object-Oriented design.
2. Implement creational design patterns for class instantiation and construct structural design patterns for object composition.
3. Apply behavioral design patterns to organize and facilitate communication among objects.
4. Evaluate existing codebases and propose refactoring strategies to restructure methods, package code appropriately, and assign effective class responsibilities.

Course content

Introduction **[6 Hrs]**

What is a design Pattern? Design patterns in smalltalk MVC, Describing Design patterns, the catalog of design patterns, organizing the catalog, How design patterns solve design problems, how to select a design pattern, how to use a design pattern.

Designing a document editor **[6 Hrs]**

Case Study on Design Problems, Document Structure, formatting, Embellishing the User Interface, Supporting Multiple Look-and-Feel Standards, supporting multiple window system, user operations, spelling checking and hyphenation.

Creational Patterns **[6 Hrs]**

Abstract Factory – Creates families of related objects without specifying concrete classes,Builder – Constructs complex objects step by step,Factory Method – Creates objects through subclasses instead of direct instantiation,Prototype – Creates new objects by cloning existing objects,Singleton – Ensures only one instance of a class exists.

Structural Patterns: **[6 Hrs]**

Adapter –Converts one interface into another compatible interface,Bridge– Separates abstraction from implementation for flexibility,Composite – Treats individual objects and groups uniformly,Decorator – Adds new functionality to objects dynamically,Façade – Provides a simplified interface to a complex system,Flyweight – Reduces memory usage by sharing common objects,Proxy – Controls access to another object through a substitute.

Behavioral Patterns **[6 Hrs]**

Chain of Responsibility – Passes requests through multiple handlers,Command – Encapsulates requests as objects,Interpreter – Interprets language expressions and grammar,Iterator – Traverses collection elements sequentially,Mediator – Controls communication between objects,Memento – Stores and restores object state,Observer – Notifies objects about state changes,State – Changes behavior based on state,Strategy – Uses interchangeable algorithms dynamically,Template Method – Defines algorithm structure with customizable steps,Visitor – Adds operations without modifying object structure.

Patterns and Software Architecture **[6 Hrs]**

Architectural pattern, Difference between design patterns and architectural patterns, different architectural patterns, relation between software architecture and patterns.

Self-Study **[12 Hrs]**

Real-world applications using design patterns diagrams for selected patterns, creational, structural, and behavioral patterns.case studies on MVC and layered architecture, examples of Singleton and Observer patterns.

Textbooks

1. "Design Patterns: Elements of Reusable ObjectOriented Software", Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Addison-Wesley Professional 1st Edition. ISBN-13: 978-0201633610
2. "Clean Architecture: A Craftsman's Guide to Software Structure and Design", Robert C. Martin, Pearson 1st Edition. ISBN-13: 978-0134494166
3. "Software Architecture in Practice", Len Bass, Paul Clements, Rick Kazman, Addison-Wesley 4th Edition. ISBN-13: 978-0136886099

Reference Books

1. "Design Patterns Explained- A New Perspective on Object Oriented Design", Allan Shalloway, James Trott, Addison Wesley 2nd Edition, ISBN-13: 978-0321247148
2. Head First Design PatternsLatest Edition: 2nd Edition (2020), O'Reilly Media.ISBN-13: 978-1492078005
3. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions", Gregor Hohpe, Bobby Woolf, Addison-Wesley 1st Edition. ISBN-13: 978-0321200686
4. Applied Java PatternsLatest Available Edition: 2002, Prentice Hall.ISBN-13: 978-0130935380
5. Java Design Patterns: A TutorialLatest Available Edition: 2000, Addison-Wesley.ISBN-13: 978-0201485394
6. Refactoring to PatternsLatest Edition: 2005, Addison-Wesley Professional.ISBN-13: 978-0321213358

Web references

1. Refactoring Guru – Design Patterns
   Complete explanation of creational, structural, and behavioral design patterns with UML and examples.
2. Microsoft Software Architecture Guide
   Architectural patterns, cloud architecture, scalability, reliability, and software design principles.
3. Oracle Java Documentation
   Java OOP concepts, GUI frameworks, MVC, event handling, and software design basics.
4. SourceMaking Design Patterns
   Tutorials and examples for Abstract Factory, Singleton, Adapter, Proxy, Observer, Strategy, etc.
5. GeeksforGeeks Design Patterns
   Simple explanations and implementation examples for all GoF patterns and architecture topics.
6. Martin Fowler Architecture Articles
   Software architecture, enterprise patterns, refactoring, MVC, and system design concepts.

(MDM) Large Language Models Theory and Deployment

Teaching Scheme
Lectures: 3 Hrs/ Week

Evaluation Scheme
Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

1. Understand the fundamentals of Large Language Models (LLMs), Transformer architecture, and pre-training techniques such as Masked Language Modeling and Causal Language Modeling.
2. Apply fine-tuning methods to adapt pre-trained LLMs for domain-specific and real-world applications.
3. Learn deployment techniques for LLMs, including infrastructure management, scalability, latency optimization, and cloud deployment using AWS, Google Cloud, and Azure.
4. Develop practical skills in deploying LLMs for real-time applications and implementing Retrieval-Augmented Generation (RAG) techniques.
5. Design and implement an industrial capstone project involving end-to-end development and deployment of an LLM-based real-world solution.

Course content

Build your own GPT from Scratch **[10 Hrs]**

Introduction to LLMs: Understanding Transformer Architecture Pre-training Techniques for LLMs: Masked Language Modeling, Causal Language Modeling

Fine-tuning LLMs **[6 Hrs]**

Adapting Pre-trained Models to Specific Tasks

LLM Deployment **[8 Hrs]**

Model Deployment Basics: Infrastructure, Scalability, and Latency Reduction Deploying LLMs on Cloud: AWS, Google Cloud, and Azure

LLM Practical Deployment Lab **[8 Hrs]**

Hands-on Lab: Deploying an LLM for Real-time Applications, Understanding RAG for supplementing LLM with additional information

Capstone Project **[6 Hrs]**

Industrial Capstone Project: Deploying an LLM in a Real-world Application

Self-Study **[12 Hrs]**

Transformers, GPT & BERT, Fine-tuning, Prompt Engineering, Cloud Deployment

Textbooks

1. **Natural Language Processing with Transformers** – Lewis Tunstall, Leandro von Werra, Thomas Wolf
2. **Hands-On Large Language Models** – Jay Alammar, Maarten Grootendorst

Reference Books

1. **Deep Learning** – Ian Goodfellow, Yoshua Bengio, Aaron Courville
2. **Speech and Language Processing** – Daniel Jurafsky, James H. Martin

Web references

1. https://developers.openai.com/api/docs

(MDM) Quantum Programming and Project

Teaching Scheme

Lectures: 2Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

1. Understand the fundamental principles of quantum computing, qubits, and quantum mechanics used in computation.
2. Develop quantum programs using frameworks such as Qiskit and execute them on simulators or real quantum hardware.
3. Evaluate quantum circuit performance considering complexity, noise, optimization, and error mitigation techniques.
4. Design mini-projects and research-oriented solutions using quantum computing concepts for real-world applications.
5. Demonstrate teamwork, presentation, and technical documentation skills through project-based learning.
6. Explore recent advancements and interdisciplinary applications of quantum computing in AI, cryptography, optimization, healthcare, and engineering.

Course content

Mathematical Foundations of Quantum Mechanics **[6 Hrs]**

Quantum Computing Concepts, Limitations of classical physics, Origin of quantum mechanics, Wave-particle duality, Black body radiation, Photoelectric effect, De Broglie wavelength, Heisenberg uncertainty principle, Complex numbers, Linear algebra, Vector spaces, Hilbert spaces, Inner and outer products, Eigenvalues and eigenvectors, Unitary matrices, Hermitian operators, Tensor products

Quantum Programming Frameworks **[6 Hrs]**

Quantum programming paradigms, Quantum assembly language, OpenQASM, Quantum SDKs, Hybrid quantum-classical computing, Variational quantum circuits, Parameterized quantum circuits, Quantum workflow automation Applications to number theory problems, Cross-platform implementation, Cloud quantum execution

Quantum Machine Learning and Generative AI **[6 Hrs]**

Quantum neural networks, Quantum perceptrons, Variational Quantum Eigensolver (VQE), Quantum Approximate Optimization Algorithm (QAOA), Quantum GANs, Quantum Support Vector Machines, Quantum reinforcement learning, Quantum kernels, Quantum generative AI, Hybrid AI architectures, QML model development, Hybrid classical-quantum AI implementation

Quantum Error Correction and Noise **[6 Hrs]**

Quantum decoherence, Quantum noise channels, Shor Code, Steane Code, Surface codes, Noise simulation, Error mitigation techniques, Fidelity analysis,

Quantum Cryptography and Security **[6 Hrs]**

Quantum key distribution, BB84 protocol, E91 protocol, Quantum teleportation, Quantum steganography, Post-quantum cryptography, Quantum internet concepts, Quantum repeaters, Quantum blockchain, BB84 simulation, Secure communication protocol implementation

Research Methodology and Project Theory **[6 Hrs]**

Literature survey techniques, Research paper analysis, Problem identification, Research gap analysis, Hypothesis formulation, Experimental design, Benchmarking quantum systems, Dataset selection, Simulation methodologies, , Result analysis, Technical paper writing, Patent drafting, IEEE publication standards, Research Project Selection, Research Review, Implementation Project, Conference-style Presentation, Research Paper Submission

Practical Applications **[4 Hrs]**

Different application areas

Textbooks

1. Quantum Computation and Quantum Information — Authors: Michael A. Nielsen, Isaac L. Chuang
2. Programming Quantum Computers — Authors: Eric Johnston, Nic Harrigan, Mercedes Gimeno-Segovia
3. Introduction to Quantum Mechanics — Author: David J. Griffiths
4. Quantum Machine Learning — Author: Peter Wittek
5. Quantum Computing for Computer Scientists — Authors: Noson S. Yanofsky, Mirco A. Mannucci
6. Machine Learning with Quantum Computers — Authors: Maria Schuld, Francesco Petruccione

Reference Books

1. Lipton, Richard J., & Regan, Kenneth W. Introduction to Quantum Algorithms via Linear Algebra (2nd ed.). Cambridge, MA: MIT Press, 2021. ISBN 978-0262045254 (hardcover); ISBN 978-0262362153 (eBook).
2. Portugal, Renato. Quantum Walks and Search Algorithms. 2nd edition, Springer, 2018. ISBN-13: 978-3-319-97812-3 (Hardcover); ISBN-13: 978-3-319-97813-0 (eBook). Part of the Quantum Science and Technology series.
3. Pittenger, Arthur O. An Introduction to Quantum Computing Algorithms. Boston, MA: Birkhäuser (Progress in Computer Science and Applied Logic, Vol. 19), 2000. ISBN-13: 978-0-8176-4127-6. DOI: 10.1007/978-1-4612-1390-1

Web references

1. https://onlinecourses.nptel.ac.in/e-learning/preview/noc26_cs89
2. https://quantum.cloud.ibm.com/learning/en/courses

(MDM) Advanced Database Management Systems

Teaching Scheme

Lectures: 3 Hrs/ Week

Evaluation Scheme

Theory: MSE: 30 Marks, TA: 20 Marks, ESE: 50 Marks

Course outcomes

Students will be able to:

1. Understand the concept and working of a parallel database system.
2. Study different types of distributed databases.
3. Analyze Distributed Transactions and Query processing.
4. Describe the key characteristics of a data warehouse.
5. Exploit Big Data platforms such as Hadoop and NoSQL databases.

Course content

Database system Architectures **[6 Hrs]**

Centralized and client server architectures ,Server system architecture, parallel systems, Distributed systems, Network Types

Parallel Databases **[6 Hrs]**

I/O Parallelism, Inter-query Parallelism , Intra-query Parallelism, Inter-operational Parallelism , Intra-operational Parallelism

Distributed Databases **[6 Hrs]**

Homogeneous and Heterogeneous databases, Distributed data storage, Directory systems

Distributed Transactions **[9 Hrs]**

System Structure , Commit protocols, Concurrency control in Distributed databases, Distributed Query processing: Non join queries and joins

Data Analysis and Mining **[9 Hrs]**

Decision support system, Data Analysis and OLAP: Multidimensional data Model, OLAP Queries, Database design for OLAP, Implementation Techniques for OLAP , Data Warehousing, Introduction to data Mining

Unit name **[4 Hrs]**

Features of NoSQL databases, Types of No SQL databases,

Self-study **[12 Hrs]**

Design of Parallel systems , Distributed catalog management ,Updating distributed data : Synchronous and asynchronous replication , Views and decision support, Classification, Regression, Implementation of NoSQL

Textbooks

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, “Database system concepts”, 5th Edition, McGraw Hill International Edition.
2. Raghu Ramkrishnan, Johannes Gehrke, “Database Management Systems”, Second Edition, McGraw Hill International Editions.

Reference Books

1. Rob Coronel, Database systems: “Design implementation and management”, 4th Edition, Thomson Learning Press.
2. J. D. Ullman, Principles of Database Systems, Galgotia Publication, 2nd Edition, 1999.
3. R. Elmasri, and S. Navathe, Fundamentals of Database Systems, Benjamin Cummings, Pearson, 6th Edition, 2010.

Web references

1. https://onlinecourses.nptel.ac.in/e-learning/preview/noc22_cs91