Learning Outcomes based Curriculum

Department of Computer Science and Engineering

Tezpur University

Master of Technology in Computer Science

Preamble

The M.Tech. in Computer Science and Engineering is a four semester full-time programme. It is an AICTE approved programme which covers major aspects of Computer Science and related subjects through 14 courses (core :7, elective: 4, open elective: 2, and seminar: 1). The student must carry out a dissertation work to demonstrate the core competency in the development of enhancements to the knowledge base in the chosen area of interest in computing. Students having completed B.E./B.Tech./AMIE/AMIETE in CSE or MCA with a minimum of 60% marks in aggregate are admitted to the programme. Candidates selected under GATE must possess a valid GATE score in CS. Admission is either based on valid GATE score in CS or based on the performance in the University entrance examinations.

1. Introduction

The M.Tech(CS) programme in Tezpur University was started in the year 2019 with the intention to train the students in both core and specialized topics in emerging fields of Computer Science. It is oriented for research and advance training in Computer Science. With new innovations constantly on the horizon, the demand of skilled computer science professional keeps rising both in the industries and academia. Our graduates of the M.Tech.(CS) programme can work independently as researchers, or as in-house employees with companies and educational institutions. The curriculum of the M.Tech.(CS) programme has been designed and evolved keeping this in mind and according to the model prescribed by the AICTE. Courses offered include Mathematical Foundation for Computer Science, Advanced Algorithms and Data Structures, Advanced Computer Architectures, Advanced Operating Systems, Selected Topics in Computer Networks, Modern Compiler Design, Pattern Recognition, Image Processing and Computer Vision, Distributed Computing, Parallel Algorithms, Advanced Database System, Bioinformatics, Machine Learning, Natural Language Processing etc. The project work in the final year is intended to equip the students to go deeper into her/his area of specialization. The curriculum is organized with few core courses and many electives to give the students enough choice for specialization like Soft Computing, Networking, Algorithms, Cyber Physical Systems, and Bioinformatics etc.

2. Qualification descriptors for the graduates

Knowledge & Understanding

1. Graduates develop an in-depth knowledge in the fundamentals of Computer Science.
2. Graduates develop the ability and confidence to analyse a problem in topics of interest critically using state of the art tools and techniques.
3. Graduates acquire expert awareness and competency to supervise and moderate the computer science applications in various domains.

Skills & Techniques

1. Graduates develop skill set for using knowledge in computer science to create the ability to configure and operate complex software systems, packages, tools, and applications for sustainability in various domains.
2. Graduates have the right communication skills required for success in their profession.
3. Graduates equip themselves with techniques of design of experiments, analysis and interpretation of data and synthesis of information to provide a valid conclusion.

Competence

1. Graduates develop the competency to adapt to the changing trends of computer science application.
2. Graduates are ready to work individually as well as in teams, in industry, academia, research, and entrepreneurship.
3. Graduates are ready for pursuing lifelong learning through higher studies, research and development and professional training.

3. Graduates Attributes

1. Graduates have a solid foundation in the fundamental of computer science, with the ability and confidence to specialize in topics of interest in the field.
2. Graduates have developed skill set for using knowledge in computer science to create configure and operate software systems, packages, and tools.
3. Graduates have developed core competency to adapt to changing trends and career opportunities in the field of computer science.
4. Graduates have skills for effective technical communication/collaboration in multidisciplinary teams providing technical leadership to comprehend, analyze, design, and create innovative computing solutions for challenging real-life problems.

5. Graduates will pursue lifelong learning through such activities as higher studies, research & development, distance education, professional training, and membership in professional societies to be able to adapt to challenges of changing environment.

6. Program Outcomes

1. Graduates will gain adequate theoretical and practical knowledge of fundamentals of computer science to groom the ability to design computational systems containing functionality, aesthetics, safety, cost effectiveness and sustainability.
2. Graduate will have the ability to design and conduct experiments as well as to analyze and interpret data and the ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.
3. Graduates will have competency to take up higher studies, research and development activities.
4. Graduates will gain the ability to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
5. Graduate has acquired some subject domain specific competence through elective and open-elective courses and project works.

1. Programme structure

Credit Requirements:
2.

i.	Minimum Credit requirement	: 68
ii.	Core Courses	: 24
iii.	Electives	: 12
iv.	Term Projects	: 24
v.	Seminar	: 02
vi.	Open Electives	: 06

Structure of the curriculum

Core Courses/Seminar/Projects					
Course Code	**Title**	**Credit Structure L-T-P**			**Total Credit**
CS541	Mathematical Foundation of Computer Science	3	1	0	4
CS 542	Advanced Algorithms and Data Structures	2	1	0	3
CS 606	Computer Architecture and Parallel Processing	3	0	0	3

CS543	Advanced Programming Lab I	0	1	2	3
IT510	Advanced Operating Systems	3	0	1	4
CS 634	Selected Topics in Computer Networks	3	0	1	4
CS 544	Advanced Programming Lab II	0	1	2	3
CS 545	Seminar	0	2	0	2
CS 640	Term Project I	0	0	8	8
CS 641	Term Project-II	0	0	16	16
				Total Credit	**50**
		Elective Courses			
	Elective 1				3/4
	Elective 2				3/4
	Elective 3				3/4
	Elective 4				3/4
				Total Credit	**12/16**
		Open Electives			
	Open Elective 1	3	0	0	3
	Open Elective 2	3	0	0	3
				Total Credit	**6**
				Total Credit	**68/72**

2. SEMESTER-WISE SCHEDULE

Course Code	**Title**	**Credit Structure L-T-P**			**Total Credit**
	1st Semester				
CS 541	Mathematical Foundation for Computer Science	3	1	0	4
CS 542	Advanced Algorithms and Data Structures	2	1	0	3
CS 634	Selected Topics in Computer Networks	3	0	1	4
CS543	Advanced Programming Lab I	0	1	2	3
IT510	Advanced Operating Systems	3	0	1	4
				Total Credit	**18**

	2nd Semester				
CS 606	Computer Architecture and Parallel Processing	3	0	0	3
CS 544	Advanced Programming Lab II	0	1	2	3
CS 545	Seminar	0	2	1	2
	Elective 1				3
	Elective 2				3/4
	Open Elective 1				3/4
	Total Credit				**17/19**
	3rd Semester				
CS 640	Term Project I	0	0	8	8
	Elective 3				3/4
	Elective 4				3/4
	Open Elective 2	3	0	0	3
	Total Credit				**17/19**
	4th Semester				
CS 641	Term Project II	0	0	0	16
	Total Credit				**16**

List of Elective Courses

Course Code	**Title**	**Credit Structure**	**Total Credit**
Elective -1			
IT 610	Advanced Database System	3-0-1	4
CS 622	Modern Compiler Design	3-0-0	3
CS 525	Artificial Intelligence	3-0-0	3
CS 659	An Introduction to Probability in Computing	3-0-0	3
IT 507	Computer Security and Cryptography	3-0-0	3
CS 664	Parallel Algorithms	3-0-0	3
CS 613	Image Processing and Computer Vision	3-0-0	3
CS 638	Software Defined Networking and Network Function Virtualization	3-0-0	3
Elective 2			
IT 611	Distributed Systems	3-0-0	3

CS 602	Image Processing	3-0-0	3
CS 619	Privacy and Security in Online Social Network	3-0-0	3
CS 616	Machine Learning	3-0-0	3
CS 614	Graph Theoretic Algorithms	3-0-0	3
CS 610	Bioinformatics	3-0-0	3
CS 618	Information Theory and Coding	3-0-0	3
IT 517	Pattern Recognition	3-0-1	4
Elective 3			
CS 621	Mobile Computing	4-0-0	4
CS 725	Knowledge Representation and Reasoning	4-0-0	4
CS 529	Embedded Systems	3-0-1	4
CS 653	Introduction to Internet of Things	3-0-0	3
CS 538	Computational Geometry	3-0-0	3
IT 509	Data Mining and Data Warehousing	3-0-1	4
CS 668	Blockchain Architecture and Use cases	3-0-0	3
CS 617	VLSI Design	3-0-0	3
CS 504	Natural Language Processing	3-0-0	3
CS 524	Theory of Computation	3-0-0	3
Elective 4			
CS 612	Advanced Computer Graphics	3-0-1	4
IT 503	Multimedia Systems	3-0-0	3
CS 637	Topics on Cognitive Radio and Networks	3-0-0	3
CS 615	Robotics	3-0-0	3
CS 620	Data Science	3-0-0	3
CS 607	Optimization Techniques	3-0-0	3
CO 503	Fuzzy Logic and Neural Networks	3-0-0	3
Elective Courses (from SWAYAM MOOCs of UGC/NPTL)			
CS 650	Introduction to Machine Learning		2
CS 651	AI: Search Methods for Problem Solving		3
CS 652	Privacy and Security in Online Social Media		2
CS 653	Introduction to Internet of Things		3
CS 654	Programming, Data Structures and Algorithms using Python		2
CS655	Scalable Data Science		2
CS656	Introduction to R Software		2

CS657	Cloud Computing		2
CS658	Social Networks		3
CS659	An Introduction to Probability in Computing		3
CS660	Programming in Java		3
CS 661	Data Science for Engineers		2
CS 662	Machine Learning for Engineering and Science Applications		3
CS 663	Randomized Algorithms		3
CS 664	Parallel Algorithms		3
CS 665	AI: Knowledge Representation and Reasoning		3
CS 666	Embedded System Design with ARM		2
CS 667	Introduction to Soft Computing		2
CS 668	Block chain Architecture and Use Cases		3
CS 669	Introduction to Industry 4.0 and Industrial Internet of Things		3

3. **Mapping of course with program outcomes (POs)**

Course code	Course title	PO1	PO2	PO3	PO4	PO5
CS541	Mathematical Foundation of Computer Science	50%	20%	15%	15%	-
CS 542	Advanced Algorithms and Data Structures	40%	30%	30%	-	-
CS 606	Computer Architecture and Parallel Processing	40%	30%	30%	-	-
CS543	Advanced Programming Lab I	40%	30%	20%	10%	-
IT510	Advanced Operating Systems	30%	30%	20%	20%	-
CS 634	Selected Topics in Computer Networks	40%	20%	20%	20%	-
CS 544	Advanced Programming Lab II	40%	30%	20%	10%	-
CS 545	Seminar	20%	20%	20%	40%	-
CS 640	Term Project I	20%	20%	20%	20%	20%
CS 641	Term Project-II	20%	20%	20%	20%	20%
	Elective 1	40%	20%	10%	10%	20%

	Elective 2	40%	20%	10%	10%	20%
	Elective 3	40%	20%	10%	10%	20%
	Elective 4	40%	20%	10%	10%	20%
	Open Elective 1	20%	20%	20%	20%	20%
	Open Elective 2	20%	20%	20%	20%	20%

4. **Evaluation plan:**
   There shall be minimum two Sessional Tests and two Examinations for each Theory Course, and two Examinations for a Practical Course having L-T-P structure. Details as follows:

Evaluation plan for Theory Courses:

Sessional Test/ Examination		Course Credit ≤ 2		Course Credit ≥ 3		Semester period
Nomenclature	Type	Marks	Duration	Marks	Duration	
Sessional Test-I	Written	20	30 min	25	45 min	Within 5th week
Mid-Semester Examination	Written	30	90 min	40	2 hours	Within 10th week
Sessional Test-II	Written/Assignment/Seminar etc.	20	XX	25	XX	Within 14th week
End-Semester Examination	Written	50	2 hours	60	3 hours	Within 18th week

Evaluation plan for Practical Courses:

Examination		L-T-P Structure-wise Marks		Semester period
		L-T-P: 0-0-z	L-T-P: x-y-z	
Nomenclature	Type	Marks	Marks	
Mid-Semester (Practical) Examination	Viva, Report	30	-	Before Mid Semester
End-Semester (Practical) Examination	Practical Examination, Viva, Report	70	50	Before End Semester

5. DETAILED SYLLABUS

***

Course Name: Mathematical Foundation for Computer Science Course Code: CS541

Credits: 4 (L-3 T-1 P-0)

PREREQUISITES

Undergraduate course in Discrete Mathematics/Discrete Structures

COURSE OBJECTIVES

- Have the basic statistical methodology of data analysis including; graphs, descriptive statistics
- Understand random variables and its distributions
- Use statistical tests in testing hypotheses on data
- To provide the foundations of probabilistic and statistical analysis that can be helpful in modeling applications like gene expression, stock market prediction, pattern recognition, computer networks etc.
- To provide concepts of Linear Algebra that are mandatory in the areas including graphics, image processing, cryptography, machine learning, computer vision, optimization, graph algorithms, quantum computation, computational biology, information retrieval and web search

COURSE CONTENT

Review of Discrete Mathematics:logic, set theory, functions and relations, , proof techniques, algorithms and integers, counting, generating functions, graphs, combinatorics

Basic probability: Basic probability: basic idea, expectation, probability calculus, Bayes' theorem, conditional probability, Data summaries and descriptive statistics, central tendency, variance, covariance, correlation, Probability distribution functions: uniform, normal, binomial, Poisson, chi-square, student's t-distribution, central limit theorem, Stochastic processes, Markov chains

Mathematical Statistics: Sampling, measurement, error, random number generation, Hypothesis testing, A/B testing, confidence intervals, p-values, regression and analysis of variance

Linear Algebra: Basic properties of matrix and vectors: scalar multiplication, linear transformation, transpose, conjugate, rank, determinant; Inner and outer products, matrix multiplication rule and various algorithms, matrix inverse; Special matrices: square matrix, identity matrix, triangular matrix, idea about sparse and dense matrix, unit vectors, symmetric matrix, Hermitian, skew-Hermitian and unitary matrices

Applications of Algebra:Linear Systems. Matrices. Gauss elimination. Echelon form. Matrix multiplication. Elementary matrices. Inverse matrix.; Vector space, basis, span, orthogonality, orthonormality, linear least squareEigen values, eigenvectors, characteristic polynomial, diagonalization, singular value decomposition

Dealing with intractability: Recent trends in the use of statistics and algebra in various fields of computer science including bioinformatics, machine learning, computer vision, pattern recognition etc.

COURSE OUTCOMES

After completion of course, students would be able to:

1. read a real world problem,
2. realize the uncertainty that is involved in a situation described,
3. select a suitable probability model,
4. estimate and test its parameters on the basis of real data,
5. compute probabilities of interesting events and other vital characteristics, and make appropriate conclusions and forecasts
6. apply linear algebra concepts in two-dimensional graphics transformations, face morphing, face detection, image transformations (e.g., blurring and edge detection, image perspective removal), classification of tumors as malignant or benign, integer factorization, error-correcting codes, and secret-sharing.

Books/References:

Text Books:

1. John Vince, Foundation Mathematics for Computer Science, Springer.
2. K. Trivedi. Probability and Statistics with Reliability, Queuing, and Computer Science Applications. Wiley.
3. M. Mitzenmacher and E. Upfal. Probability and Computing: Randomized Algorithms and Probabilistic Analysis.
4. Alan Tucker, Applied Combinatorics, Wiley
5. Sheldon Axler. Linear Algebra Done Right, Springer
6. Gilbert Strang. Linear Algebra and Its Applications. Cengage Learning
7. David A. Forsyth, Probability and Statistics for Computer Science, Springer
8. Sheldon M. Ross, Probability Models for Computer Science, Academic Press

Course Name: Advanced Algorithms and Data Structure Course

Code :CS542

Credits: 3 (L-2 T-1 P-0)

PREREQUISITES

Undergraduate course each in algorithms and data structures

COURSE OBJECTIVES

- Develop 'students' algorithmic thinking and their ability to analyse the efficiency of algorithms;
- Enable students to find different approaches for dealing with challenging computational problems;
- Provide insight into cutting-edge research-led teaching in modern subfields of algorithms theory.

COURSE CONTENT

Algorithm Review:Basic algorithms: Asymptotic notation, amortized analysis, average case analysis

Review of Basic Data Structures: arrays, heap, lists, binary search trees, hashing, graphs

AlgorithmicTechniques: recursion, divide-and-conquer, greedy, dynamic programming,

Advanced Data structures:Disjoint sets / union-find, Fibonacci heaps, splay trees, dynamic trees, Red-Black trees, Skip lists, B-tree, dictionaries, geometric data structures

Advanced Algorithms:Fast Fourier transform, network flow, Linear programming, Matching algorithms, String matching, Number theoretic algorithms + RSA, geometric algorithms

Dealing with intractability:NP-Completeness, Approximation algorithms, LP based approximations, randomized algorithm

COURSE OUTCOMES

After completion of course, students would be able to:

1. Appreciate what solution to a computational problem efficient or inefficient
2. Compare between different data structures. Pick an appropriate data structure for a design situation.
3. Analyse the efficiency of advanced algorithmic techniques e.g., approximation(randomized and LP-based rounding etc), randomization algorithms (expected running time, probability of error).

4. Identify and apply design principles for the design of advanced efficient algorithms
5. Understand some pieces of current research on algorithms.

Books/References:

Text Books:

1. Rajeev Motwani and Prabhakar Raghavan, *Randomized Algorithms*, Cambridge University Press, 2000.
2. Michael Mitzenmacher and Eli Upfal, *Probability and Computing*, Cambridge University Press, 2005.
3. David Williamson and David Shmoys, *The Design of Approximation Algorithms*, Cambridge University Press, 2011.
4. Jon Kleinberg and Eva Tardos, *Algorithm Design*, Addison-Wesley, 2006.
5. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd edition, 2001.

Course Name: Computer Architecture and Parallel Processing
Course Code : CS 606

Credit :L-T-P:Cr:: 3-0-0:3

PREREQUISITES

Knowledge in basic Computer Organization & Architecture, Compiler Design and Operating System

COURSE OBJECTIVES

- To understand the state-of-the-art in computer architectures and parallel computing
- Have the knowledge of different architectural developments to improve the efficiency of a computer system
- Develop the skill to measure the performance of a computer system
- To know the performance bottlenecks in typical Von-Neumann based architecture
- Be familiar with how to exploit different kinds of parallelism possible in the underlying execution model
- Be familiar with architectures used in Mobile devices, Cloud and High-performance computing devices
- To train parallel programming techniques

COURSE CONTENT

Definitions of Computer Architecture - Abstract Architecture & Concrete Architecture. Concepts in Parallel Processing - Available Parallelism and Utilized Parallelism.

ParallelProgramming Models – PRAM, Shared Variable, Message Passing, Data Parallel.

Classification of Computer Architectures – Flynn's Classification – Classification of ParallelArchitectures.

Instruction Level Parallel (ILP) Processors – Pipelined, VLIW,

Super Scalar Processors –Instruction Dependencies, their Effect on Performance and Techniques to overcome them.Basic Concepts and Techniques in Vector, Systolic and Dataflow architectures.

Multiprocessor and Multicore Architectures – Synchronization and Cache Coherence Issues.

Multicomputer and cloud Architectures – Interconnection Networks, Routing and Data CommunicationAlgorithms.

COURSE OUTCOMES

At the end of the course, the studentsshould be able to

1. Understand the state-of-the-art in computer architectures and processing
2. Evaluate the performance of computer Systems
3. Determine the key architectural elements to improve the performance of a computer system
4. Understand the Instruction Level Parallelism and the techniques to handle different types of data and control hazards
5. Design Memory Hierarchy for improving performance of computer system
6. Understand the Vector Processor and Data Flow Architecture for exploiting parallelism
7. Understand the architectures used in mobile, cloud and high performance computing devices
8. Write programs for Parallel Processors

Books/References:

1. J. L. Hennessey and D. Patterson, Computer Architecture: A Quantitative Approach, Elsevier

2. K. Huang, F. A. Briggs, Computer Architecture and Parallel Processing, McGraw Hill.

3. V. Kumar et el. Parallel Computing, Klewer Publishers.

4. D. Sima, T. Fountain, P. Kacsuk, Advanced Computer Architectures – A Design Space Approach, Addision-Wesley.

Course Name :Advanced Programming Lab I Course Code :CS543

Credits: 3(L-0 T-1 P-2)

PREREQUISITES

Undergraduate courses in procedural programming and object oriented programming

COURSE OBJECTIVES

- Be competent with use of basic constructs provided by high-level programming languages
- Be able to use computational thinking to design solutions
- Be competent with use of computational approaches to solve problems in science and engineering
- Develop proficiencyin implementation of Algorithmic techniques using appropriate data structures
- Be proficient in using advanced data structures
- Develop competency in implementation of advanced algorithms

COURSE CONTENT

This course will involve the students extensively in writing programs for assignments in contemporary programming languages. Topics include:

- Matrix operations like inverse, rank, linear equations, Eigen values and vectors
- Implementing algorithms using recursions, divide-and-conquer, greedy, dynamic programming concepts
- Linear programming using Cplex
- Implement some commonly used graph algorithms
- Implement network flow algorithms/matching
- Implement advanced data structures like B-Tree, dynamic trees, skip lists, suffix tree, red black tree, tries, R-tree, geometric data structures (KD-tree, interval tree, Quad tree, range tree etc.)

COURSE OUTCOMES

After completion of course, students would be able to:

1. Master in use of procedural programming language C and object oriented language C++ to implement basic algorithms and use of data structures

2. Master in development and implementation of algorithms for different problems in Computer Science
3. Master in implementation of graph algorithms for some well known problems
4. Master in solving some linear programming problems using Cplex
5. Master in implementation of advanced data structures for solving problems related to cryptography, networks, machine learning, text processing, internet programming, compiler construction etc.

Books/References:

Text Books:

1. Introduction to Scientific Computation and Programming, Daniel T. Kaplan
2. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to Algorithms, 2nd edition, 2001.
3. Fischer and Leblanc, Crafting a Compiler with C
4. Clean Code: A Handbook of Agile Software Craftsmanship by Robert C Martin
5. Programming Pearls by Jon Bently
6. Algorithms + Data Structures = Programs by Niklaus Wirth
7. Thinking in C++ by Bruce Eckel
8. Thinking in Java by Bruce Eckel
9. Effective Java by Joshua Bloch
10. Python in Practice. Mark Summerfield
11. Python Cookbook. David Beazley and Brian K. Jones

Course Name : Advanced Operating System Course Code : IT501

Credits: 4 (L-2 T-1 P-1)

PREREQUISITES

(i) The students should have basic understanding of principles of operating systems design and implementation and principles of computer hardware design and implementation covered in under graduate level.

(ii) Students should have detailed knowledge of C programming language and use of Java language.

(iii) Working knowledge of the UNIX programming environment

COURSE OBJECTIVES

- At the end of the course, the student will have become exposed to classic and current operating systems literature, gained the experience of conducting research in the area of operating systems, developed state-of-the-art research projects that lead to publishable results.
- The course will be instrumental to familiarize and build the confidence in the students to develop the knowledge of design and develop mobile operating system based knowledge and application on it.

COURSE CONTENT

Concurrent Execution: threads, event systems,async/sync I/O,Parallelism, Ordering, and Races, Dynamic Data Race Detector for Multi-Threaded Programs, Discussions of synchronization with an emphasis on monitors, On Optimistic Methods for Concurrency Control, Concurrency Control and Recovery, Communication using lightweight remote procedure call (RPC)

Memory Management: virtual memory, NUMA machines, memory allocators – Hoard Scalable Memory Allocator, Memory Resource Management in VMware, Global Memory Management in Cluster machines

Scalability: Multicore processing, locking, lock-free data structures, The Scalable Commutativity Rule: Designing Scalable Software for Multicore Processorsetc.

OS Architecture:The structure and design of an operating system, OS Architecture and Extensibility: SPIN and the Exo-kernel, OS architecture for scalable multicore systems: Multi-kernelMobile OS Architecture: Android, iOS

Virtualization:Machine virtualization, binary instrumentation, VMware design etc.

File Systems and Disk: file system interfaces, networked file systems, AFS, The Design and Implementation of a Log-Structured File System, File system extensibility, non-disk file systems, A Case for Redundant Arrays of Inexpensive Disks (RAID), Using Model Checking to Find Serious File System Errors

Big Data System:Google File system, OceanStore, Facebook Photo Storage, SPOCA, Corfu, RAMCloud, Dynamo: Amazon's Highly Available Key-value Store,MapReduce and Spark, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems

Security:Operating systems security fundamentals, data security and integrity, authentication, authorization, etc.

Reliability Computing:Operating systems reliability, failure oblivious computing, recovery-oriented computing

History/Experience: historically important papers and experience reports

COURSE OUTCOMES

After completion of course, students would be conversant with:

1. Concepts of some of the advanced topics in operating system design, issues involved in memory, concurrency, and file management techniques, familiar with some advanced topics and emerging type of operating system design concepts.
2. Ability to design and analyze mobile OS based applications and to customize kernel tuning.

Books/References:

Text Book:

1. Advanced Concepts in Operating Systems: Distributed, Database, and Multiprocessor Operating Systems by Mukesh Singhal, Niranjan Shivaratri, 2017, McGraw Hill
2. There is no specific textbook for this course. The course is based on a collection of journal and conference papers

Reference Books:

1. Modern Operating Systems, Tanenbaum
2. Operating Systems: A Modern Perspective, Gary Nutt,
3. Advanced Operating Systems, by Silberschatz, 1999
4. Gary Nutt, Kernel Projects for Linux, Gary Nutt
5. Advanced Programming in the Unix Environment by W. Richard Stevens, Addison-Wesley, 1993
6. Unix Network Programming: Networking APIs: Sockets and XTI (Volume 1) by W. Richard Stevens.
7. The Art of Computer Systems Performance Analysis, by Raj Jain.

Course Name: Selected Topics in Computer Networks Course

Code: CS 634

Credits: 4 (L-2 T-1 P-1)

PREREQUISITES

(i) The students should have some basic knowledge in Data Communication, Computer Networks and TCP/IP protocol stack covered in under graduate level.

(ii) Students should have detailed knowledge of C programming language and use of Java language.

COURSE OBJECTIVES

- This course intends to build up some of important topics on computer networks and the design and implementation details for performance improvement of the existing network and some advanced topics.
- To familiarizeby covering topics with case studies giving opportunity to explore practical/simulation of some advanced concepts using network simulator and Linux operating system environment.
- The course will be instrumental to build the confidence in the students to develop the knowledge of design and develop selected topics in computer network.

COURSE CONTENT

Layering abstraction, Network architecture, Packet switching

Performance of Networks: delay, jitter and throughput

Application layer: HTTP and other application protocols, DNS, HTTP transport recent developments, Socket programming

Overlay networks and peer-to-peer (P2P) systems, wrap-up application layer
Transport layer: introduction, TCP congestion control

Analysis of TCP: General congestion control and queuing, Random Early detection(RED), BBR: Congestion-Based Congestion Control, QUIC protocol, Active Queue Management TCP implementation details, multipath TCP.

QoS and fairness, traffic shaping

Router scheduling algorithms: Implementation and analysis of fair queuing algorithms.

Network layer: Linux IP networking, advanced topics in IP

Router architectures, MPLS, routing protocols, IP and MPLS routers, BGP: Introduction, BGP advanced topics with BGP security

Link layer: introduction, link layer functions, link layer addresses, ARP, shared broadcast, medium access protocols

Network and link layer practical topics: TCP traversal trough NATs and firewalls
Multimedia Networking: RSVP and Integrated Services (IntServ), Differentiated Services (DiffServ), Real-Time Protocol (RTP), RTP Control Protocol (RTCP), Session Initiation Protocol (SIP), Real-time Streaming Protocol (RTSP), Internet video

Data center networking: Data center network architecture, Load balancing issues

Software defined networking: introduction, OpenFlow, B4: Software defined WAN, Google

Network and Virtualization: introduction, NFV, NetVM

More recent topic in networking: few selected recent papers will be discussed

COURSE OUTCOMES

After completion of course, students would be conversant with:

1. Concepts of some selected topics in computer network, issues involved in network design, familiar with some advanced topics and emerging type of network design concepts.
2. Ability to design and analyze some of the protocols behaviour

Books/References:

Text Book:

1. Computer Networking, A Top-Down Approach Featuring the Internet, Second Edition, J.F. Kurose and K.W. Ross, Adition-Wesely, 2002.
2. Computer Networks: A Systems Approach, Third Edition, L. Peterson and B.S. Davie, Morgam Kaufmann, 2003.
3. Computer Networks, Fourth Edition, A. Tanenbaum, Prentice-Hall, 2002.

Reference Books:

4. Unix Network Programming, W. Richard Stevens
5. Data Networks, Bertsekas and Gallager
6. Selected Research papers published in journal and conferences will be used to cover some of the recent topics.

Course Name: Advanced Programming Lab II Course Code: CS544

Credits: 3 (L-0 T-1 P-2)

PREREQUISITES

Undergraduate courses in procedural programming and object oriented programming

COURSE OBJECTIVES

- Be competent with Linux shell programming
- Be competent in processing different types of textual data using Python/Perl/Java/Lex/Yacc

- Be competent with writing simple MATLAB/Octave programs for performing numerical calculations
- Be competent to use ML approaches to solve problems in Bioinformatics, Computer Visions, Pattern Recognition, Image Processing, Cyber security, knowledge discovery, NLP etc
- Be competent in using language and tools to solve problems of Graphics, Computer Vision and use of simulation tools
- Be proficient in high quality document preparation using packages including LaTex, GNU plot etc

COURSE CONTENT

This course will involve the students extensively in writing programs for assignments involve programming in contemporary programming languages. Topics include:

- Unix/Linux shell programming,
- Documentation using LaTex
- Text processing using Python/Perl/Java Script
- Text parsing using Lex/Yacc
- Use of ML approaches for solving computational problems
- Writing codes for image processing, graphics, computer vision related problems C/C++/Java/MatLab/Octave

COURSE OUTCOMES

After completion of course, students would be able to:

1. Master an understanding of scripting and the contributions of scripting languages.
2. Master an understanding of Python especially the object-oriented concepts,
3. Master an understanding of the built-in objects of Python,
4. Master in using ML approaches in computational problems
5. Master in preparation of high quality documents using LaTex and different graphics packages
6. Master in using simulation tools

Books/References:

Reference Books:

1. LaTeX Beginner's Guide, Stefan Kottwitz
2. Latex Document Preparation System Users, Leslie Lamport
3. Introduction to Computation and Programming Using Python: With Application to Understanding Data. MIT Press, Guttag, John.
4. Crafting a Compiler with C, Fischer and Leblanc

5. Programming Pearls, Jon Bently
6. Flex & Bison, John R. Levine
7. A Guide To Matlab: For Beginners And Experienced Users, Hunt, Lipsman and Rosenberg
8. Matlab for Machine Learning: Functions, algorithm and use cases, Giuseppe Ciaburro
9. Thinking in C++, Bruce Eckel
10. Thinking in Java, Bruce Eckel
11. The C programming Language, K&R
12. Advanced Programming in Unix Environment, R. Stevens
13. Eloquent JavaScript A Modern Introduction to Programming,MarijnHaverbeke
14. Unix Shell Programming, Sams Publishing,Stephen G. Kochan and Patrick Wood
15. Programming Perl: Unmatched power for text processing and scripting, Tom Christiansen, Brian DFoy, Larry Wall and Jon Orwant
16. Perl Best Practices,O'Reilly, Damian Conway
17. John W. Eaton, David Bateman, Soren Itauberg, Rik Wehbring (2019). GNU Octave version 5.1.0 manual: a high-level interactive language for numerical computations. https://www.gnu.org/software/octave/doc/v5.1.0/

Course Name: Seminar

Course Code: CS545

Credits: L-0-T-2-P-0: 2

PREREQUISITES

Core courses of M.Tech CSE program

COURSE OBJECTIVES

- to create an environment to engage students in delivering and listening to interesting talks that promotes discussion
- to provide students with opportunity to learn new concepts and skills acquired in core courses and further extend these ideas to solve research/industry related problems
- know how to read research papers critically and efficiently
- to learn fundamental principles, generalizations and important theories of Computer Science
- to enable students to find their own field of interest in academia, industry or entrepreneurship
- to help students develop their own learning and teaching styles and communication skills

COURSE CONTENT:

Student presentations:

- Each student will present one paper during the term

Class evaluations:

- Each week each student is asked to write a short evaluation of one of the papers being presented

Class Discussion:

- Discuss the papers – expose the flaws, analyse the writing, what was the impact?

COURSE OUTCOMES

Because the curriculum is about individuals, there are no specific course level learning outcomes. Nevertheless, after completion of the course, students would be able to:

1. Explain factual knowledge (terminology, classifications, methods, trends). of current areas of research.
2. State and explain some fundamental principles, generalizations, or theories the student has learned in this course.
3. To apply gained knowledge in thinking, problem solving, or decisions making process.
4. To achieve specific skills, competencies, and points of view needed by computing professionals.
5. to judge the value of different contributions
6. to identify promising new directions

Books/References:

A list of works will be posted by mentors/teachers at the start of the course. The students also have the option of choosing works according to his/her own areas of interest.

Course Name: Term Project I Course

Code : CS640

Credit : L-0:T-0:P-8: 8

PREREQUISITE: NIL

COURSE OUTCOMES

1. Demonstrate sound fundamentals in a chosen area of computing
2. Identify and formulate a problem of research interest in the chosen area of computing
3. Analyze the computing problem and propose solutions
4. Apply the emerging technologies like – Blockchain, IoT, Robotics, ML, AI, Datamining, Big Data Analytics in solving some challenging problem in chosen area
5. Effectively communicate the work at all stages of the project

COURSE CONTENT

The student is expected to carry out supervised research in this course. An intensive literature in thechosen area, should result in sound knowledge in the area and result in the identification of a suitable research problem, and its formulation and analysis. Study of relevant supplementary literature, such mastering useful programming languages and tools for the problem, are also expected at this stage of the project. The student is expected to present three reports at different evaluation points duringthe semester, with clearly defined achievements and plans for further steps.

References

Relevant literature and software tools for the chosen problem

Course Name: Term Project II Course

Code : CS641

Credit: L-0: T-0:P-16: 16

PREREQUISITE: Term Project I

COURSE OUTCOMES

- Identify a suitable problem to be solved computationally
- Reflectively analyze proposed solutions to the identified computing problem
- Design and develop solutions to the problem and analyze results
- Prepare a thesis and defend the thesis on the work done
- Augment the knowledge base in the chosen area of computing, adhering to ethical practices at every stage

COURSE CONTENTS

The studentsare expected to demonstrate the core competency in the development of enhancements to the knowledge base in the area of interest in computing. The secondary competencies include the management of time bound projects involving research, analysis of problem complexities, design and development of effective solutions and communication of the project's progress, adhering to ethical practices at every stage. This stage of the project evaluates the state of maturity of these competencies. The students are expected to present two reports at intermediate stages, as well as prepare and defend a thesis on their research work.

The students will usually continue the project work in Term Project I (CS 641) or optionally can take a new research-oriented project in consultation with the assigned project supervisor.

References

Relevant literature and software tools for the

***