**FF No. : 654**

**Syllabus Template**

# CS2225::Theory of Computation

**Course Prerequisites:** Discrete Mathematics, Computer Programming.

## Course Objectives:

1. To introduce basic concepts such as alphabet, strings, Languages, Decision problems, etc to work with the abstract formal setup
2. To construct deterministic/nondeterministic automata for regular languages to prove non regularity of languages through application of Pumping Lemma and Myhill-Nerode theorem.
3. To understand the role of non-determinism in Automata theory
4. To design Context free grammars, Push down automata for Context Free Languages
5. To comprehend meaning of undecidability in the context of Turing Machine Model

**Credits: 5**

**Teaching Scheme Theory: 3** Hours/Week

**Tut: 1** Hours/Week

Lab: **2** Hours/Week

## Course Relevance:

This is a foundational course for Computer Science and Engineering. The central theme of the course is to study what makes certain computational problems very hard and the others easy? Is there some concrete theoretical evidence for the exhibited hardness of the problems? The course explores these questions, first by introducing students to the abstract notion of computation and models of computation. Starting from very simple model of state machines to finally cumulating into the Turing machine model (which is a foundation of modern-day computers), several models in between are studied. For every model, questions such as, which computational problems can be/cannot be solved in the model? how efficiently a problem can be solved in a particular model? various closure properties of model are studied. Throughout the course emphasis is given to proving things with concrete mathematical arguments.

The course is very important for understanding the concept of computation in more abstract set-up. Wherever one wants to formally talk about underlying model, the restrictions imposed by the model, what is the power and limitations of the model, the principles learnt in this course are useful. Due to abstract nature of the course, the principles learnt have wide applicability. The course is an essential prerequisite for several advanced courses such as Computational Complexity, Advanced Algorithms, Foundation of Logic, Quantum Computation, Parallel computation, Circuit Complexity etc. On more applied side: The Automata theoretic models, concept of Context Free Grammar and Pushdown Automata studied in the course are very important for Compiler design. The models discussed during the course have direct applications to several machine learning models, Natural Language processing, Artificial Intelligence, Functional Programming.

Once the student gains expertise in thinking abstractly about underlying models of computation it facilitates in systematic study of any other domain (in computer science or otherwise) which demands logical thinking and abstraction.

This course is also relevant for students who want to pursue research career in theory of computing, computational complexity theory, Natural Language Processing, advanced algorithmic research.

## SECTION-1

**Topics and Contents**

**Finite Automata:**
Introduction to Automata, Computability and Complexity theory, Automaton as a model of computation, Central Concepts of Automata Theory: Alphabets, Strings, Languages. Decision Problems Vs Languages. Finite Automata, Structural Representations, Deterministic Finite Automata (DFA)-Formal Definition, Simplified notation: State transition graph, transition table, Language of DFA, construction of DFAs for Languages and proving correctness, Product construction, Nondeterministic finite Automata (NFA), NFA with epsilon transition, Language of NFA, Conversion of NFA with epsilon transitions to DFA, Automata with output. Applications and Limitation of Finite Automata.

**Regular and Non-Regular Languages:**
Regular expression (RE), Definition, Operators of regular expression and their precedence, Algebraic laws for Regular expressions, Kleene’s Theorem: Equivalence Regular expressions and DFAs, Closure properties of Regular Languages (union, intersection, complementation, concatenation, Kleene closure), Decision properties of Regular Languages, Applications of Regular expressions. Myhill-Nerode theorem and applications: proving non-regularity, lower bound on number of states of DFA, State Minimization algorithm, Equivalence testing of DFAs. Non-Regular Languages, Revisiting Pigeon-Hole principle, Pumping Lemma for regular Languages.

**Context Free Grammars (CFG):**
Context Free Grammars: Definition, Examples, Derivation, Languages of CFG, Constructing CFG, correctness proof using induction. Closure properties of CFLs (Union, Concatenation, Kleene closure, reversal). Derivation trees, Ambiguity in CFGs, Removing ambiguity, Inherent ambiguity. Simplification of CFGs, Normal forms for CFGs: CNF and GNF. Decision Properties of CFLs (Emptiness, Finiteness and Membership). Applications of CFG.

## SECTION-1I

**Topics and Contents**

**Push Down Automata:**
Description and definition, Language of PDA, Acceptance by Final state, Acceptance by empty stack, Deterministic, Non-deterministic PDAs, CFG to PDA construction (with proof). Equivalence of PDA and CFG (without proof). Intersection of CFLs and Regular language. Pumping lemma for CFLs, non-Context Free Languages, Context Sensitive Languages, Definition and Examples of Context Sensitive Grammars, Chomsky hierarchy.

**Turing Machines:**
Basic model, definition, and representation, Instantaneous Description, Language acceptance by TM. Robustness of Turing Machine model and equivalence with various variants: Two-way/One-way infinite tape TM, multi-tape TM, non-deterministic TM, Universal Turing Machines. TM as enumerator. Recursive and Recursively Enumerable languages and their closure properties.

**Introduction to Undecidability:**
Church-Turing Thesis and intuitive notion of Algorithm. Introduction to countable and uncountable sets (countability of set of natural numbers, integers, rational numbers. Uncountability of set of real numbers, points in plane, set of all binary strings), Encoding for Turing machines and countability of set of all Turing machines. Existence of Turing unrecognizable languages via Cantor's diagonalization. Undecidability of Halting problem. Examples of undecidable problems: Post Correspondence Problem, Hilbert's 10th Problem, Tiling problem (without proof). Example of Turing unrecognizable language. Decision properties of R, RE languages and Rice’s theorem.

**List of Tutorials: (Any Three)**
1) Problem solving based on deterministic and non-deterministic finite automata.
2) Advanced problem solving based on DFA, NFA.
3) Problem solving based on Regular expressions.
4) Problem solving based on Pumping Lemma.
5) Understanding Myhill-Nerode theorem.
6) Advanced problems on Context Free Grammars.
7) Problem solving on Pushdown Automata.
8) Problem solving on Turing Machines.
9) Problem solving on Contability
10) Problem solving on undecidability

**List of Practicals: (Any Six)**

1.**Problem Solving based on Basic Counting:** Propositional logic, Introduction to proofs: direct, contraposition, contradiction, counterexamples, principle of mathematical induction, strong induction. Proving correctness of programs.

Elementary set theory, relations, functions, basic counting principles, permutations, combinations, generalized permutations and combinations (with/without repetitions, distinguishable/indistinguishable objects), Binomial coefficients and identities. Double counting, combinatorial proof technique, Pigeon-Hole Principle and some applications, Inclusion Exclusion Principle, and applications.

Recurrence relations, modeling using recurrence relations (some examples Fibonacci numbers, Catlan numbers, Derangements, Tower of Hanoi, partitions), generating functions and their application in counting.

**2. Problem Solving based on Basic Discrete Probability:**
Definition of probability, examples, independence of events, conditional probability, union bound, inclusion exclusion, Bayes' rule, discrete random variables, expectation, variance, linearity of expectation, sum of independent random variables, Markov and Chebyshev inequality, weak law of large numbers, standard distributions (Bernoulli, Binomial, Geometric), coupon collector problem, birthday paradox, probabilistic recurrences. Uniform generation of combinatorial structures. Indicator random variables and their role in algorithm analysis.

**3. Problem Solving based on Modular Arithmetic:**
Number theory – Integers, division algorithm, divisibility and congruences, gcd and Euclid's Algorithm, extended Euclid's algorithm, application to modular inversion, prime numbers, Euclid's proof for infinitude of primes, unique factorization, Fermat's little theorem, Euler's phi function, Euler's theorem, Chinese remainder theorem, Fast modular exponentiation.

**4. Problem Solving based on Graph Theory:**
[To be taught in combinatorial perspective] Graphs, different representations, properties of incidence and adjacency matrices, directed/undirected graphs, degree of a vertex, connected components, paths, cycles in graph, Eulerian and Hamiltonian tours, Trees, properties of trees,

Simple combinatorial problem solving based on graphs, bipartite graphs (graph with only odd cycles, 2-colorable graphs), Planar graphs, Euler's theorem for planar graph, Graph colorings, matching in bipartite graphs

**List of Course Seminar Topics:**
1. NFA Vs DFA
2. Pumping Lemma and Applications
3. Closure properties of Regular languages
4. Decision properties of Regular languages
5. Chomsky hierarchy
6. Application of TOC principles in compiler design
7. Hilbert's 10th problem
8. Context Free and Context sensitive grammars
9. Pumping Lemma for CFL and applications
10. Recursive and Recursively enumerable languages

**List of Course Group Discussion Topics:**
1. Applications of Automata theory in Compiler design
2. Applications of Automata theory in Natural language processing
3. Undecidability
4. Software testing, why it is very hard?
5. Robustness of Turing machine model
6. Godel's incompleteness theorem
7. Countable and un-countable sets
8. P Vs NP problem
9. Church Turing thesis
10. Models of computation

**List of Home Assignments:**
**Design:**
1. Solve 5 challenging problems on NFA, DFA
2. Solve 5 challenging problems on non regular, regular languages
3. Solve 5 challenging problems on Context free grammars, PDAs
4. Solve 5 challenging problems on Turing machines
5. Solve 5 challenging problems on undecidability

**Case Study:**
1. Randomized algorithms for pattern matching
2. Myhill-Nerode theorem and applications
3. Chomsky-Schützenberger Theorem and Dyck languages
4. Lambda Calculus
5. Hilbert's 10th Problem

**Blog**
1. Finite Automata
2. Timed Automata and applications

3. Buchi Automata
4. Non-regular languages
5. Contability

**Surveys**

1. Pattern Matching algorithms
2. Parsers
3. Evolution of models of computations
4. Role of nondeterminism in theory of computation
5. Closure and decision properties of Context free languages

**Suggest an assessment Scheme:**

*Suggest an Assessment scheme that is best suited for the course. Ensure 360 degree assessment and check if it covers all aspects of Blooms Taxonomy.*

MSE: 10% + ESE: 10% + Seminar: 15% Group Discussion: 15% + Home Assignments: 10% + Discrete-Maths evaluation 20% + CVV: 20%

**Text Books:** ***(As per IEEE format)***

*1. Hopcroft J, Motwani R, Ullman, Addison-Wesley, "Introduction to Automata Theory, Languagesand Computation", Second Edition, ISBN 81-7808-347-7.*

*2. Michael Sipser, Course Technology, "Introduction to Theory of Computation", Third Edition,ISBN-10: 053494728X.*
*3.. "Discrete Mathematics and its applications" by Kenneth Rosen (William C Brown Publisher)*

**Reference Books:** ***(As per IEEE format)***

*1. J. Martin, "Introduction to Languages and the Theory of Computation", Third edition, Tata McGraw-Hill, ISBN 0-07-049939-x, 2003.*
*2. Daniel I. A. Cohen, "Introduction to Computer Theory", Wiley-Second Edition, ISBN-10 : 04711377*

**Moocs Links and additional reading material:** www.nptelvideos.in

**Course Outcomes:**
The student will be able to –
1. Infer the applicability of various automata theoretic models for recognizing formal languages.
2. Discriminate the expressive powers of various automata theoretic and formal language theoretic computational models.
3. Illustrate significance of non determinism pertaining to expressive powers of various automata theoretic models.
4. Comprehend general purpose powers and computability issues related to state machines and grammars.
5. Explain the relevance of Church-Turing thesis, and the computational equivalence of Turing machine model with the general purpose computers.
6. Grasp the theoretical limit of computation (independent of software or hardware used) via the concept of undecidability.

**CO PO Map**

| CO1 | CO2 | CO3 | CO4 | CO5 | CO6 |
|---|---|---|---|---|---|
| PO1 | PO2 | PO4 | PO9 | PO12 | PSO13 |
| 3 | 3 | 2 | 1 | 2 | 3 |

**CO attainment levels**

| **CO number** | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|
| **Attainment level** | **2** | **3** | **3** | **4** | **5** | **5** |

**Future Courses Mapping:**

*Compiler design, Computational Complexity theory, Computability theory, Advanced Algorithms, Natural Language Processing, Artificial Intelligence*

**Job Mapping:**
*Wherever one wants to formally talk about underlying model, the restrictions imposed by the model, what is the power and limitations of the model, the principles learnt in this course are useful. Due to abstract nature of the course, the principles learnt have wide applicability, let it*

*be domain of Machine learning, Natural Language processing, Compiler design, Parallel computation, for each of them having background of Theory of Computation is very useful. If student wants to pursue higher education/ research in Computer Science, this course is must.*