L	T	P	C
3	1	0	4

Course Code: CSE301

THEORY OF COMPUTATION

Course Objectives

This course will help the learner to discuss different classes of formal languages in Chomsky hierarchy, their properties and to design the acceptor machines for those languages.

UNIT - I 15 Periods

Introduction: Preliminaries and notations - Basic concepts - applications - **Finite Automata:** Deterministic FA - Non-deterministic FA - Equivalence - Minimization - **Regular languages and Regular grammars:** Regular expressions - Relation between regular languages and regular expressions – Regular grammars **Properties of Regular Languages:** Closure properties - identifying non-regular languages using pumping lemma

UNIT - II 15 Periods

Context Free Languages: Context free grammars - parsing and ambiguity - Context-free grammars and programming languages **Simplification and Normal Forms:** Methods for Transformating grammars - Chomsky and Greibach normal forms - membership algorithm for CFG. **Push Down Automata:** Non-Deterministic PDA - PDA and CFL - Deterministic PDA and deterministic CFL - Grammars for deterministic CFL

UNIT - III 15 Periods

Properties of CFL: Pumping Lemma for CFL, Closure properties and decision algorithm for CFL. **Turing Machines:** The Standard Turing Machine - combining TM for complicated tasks - Turing’s thesis. **Other models of TM:** Minor variations on TM - TM with complex storage - Non-deterministic TM - Universal TM - Linear bounded automata

UNIT - IV 15 Periods

A hierarchy of formal languages and automata: Recursive and recursive enumerable Languages - unrestricted grammars - context sensitive grammars and languages - Chomsky Hierarchy - **Limits of algorithmic computation**: problems that can’t be solved by TM - Undecidable problems for recursively enumerable languages - post correspondence problem - Undecidable problems for CFL - **An overview of computational complexity:** Turing Machine models and complexity - Language families and complexity classes - complexity classes P and NP - Some NP problems - Polynomial time reduction - NP-completeness

TEXTBOOK

1. Peter Linz. *An Introduction to Formal Languages and Automata.* Jones and Bartle Learning International United Kingdom, Sixth Edition, 2016.

REFERENCES

1. John E. Hopcroft, Rajeev Motwani and Jeffery D Ullman. *Introduction to Automata Theory, Languages and Computation,* Third Edition, Pearson Education, 2007.
2. Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeffrey D. Ullman. *Compilers Principles, Techniques & Tools,* Pearson Education, 2007.
3. Susan H. Rodger and Thomas W. Finley. *JFLAP: An Interactive Formal Languages and Automata Package*, Jones & Bartlett Publishers, Sudbury, MA, 2006.

4. Michael Sipser. *Introduction to the theory of computation*, Second Edition, Thomson Course Technology, 2006.

ONLINE MATERIALS

1. http://nptel.ac.in/courses/106104028/
2. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/lecture-notes/

UNITWISE LEARNING OUTCOMES

Upon successful completion of each unit, the learner will be able to

Unit I	• List the properties of regular languages, design regular expressions and construct equivalent automata • Identify and prove a language is regular or not
Unit II	• Design the context-free grammars for context-free languages, transform them into normal forms • Identify a string belongs to the given context-free language or not • Construct PDA for the equivalent context-free grammars
Unit III	• Identify CFL and prove using Pumping Lemma • Demonstrate the properties of CFL • Design Turing Machine for simple and complex tasks • Describe different Turing Machine models
Unit IV	• Summarize Chomsky Hierarchy, and differentiate recursive &recursively enumerable languages • Describe concepts of computational complexity, unsolvable and un-decidable problems

COURSE LEARNING OUTCOMES

Upon successful completion of this course, the learner will be able to

- Design an appropriate automaton for a given language
- Construct equivalent automaton from grammar of languages
- Design an appropriate push down automaton for a given language
- Construct PDA for the equivalent context-free grammars
- Select appropriate Turing Machines for a given problem based on complexity analysis
- Describe concepts of computational complexity, unsolvable and un-decidable problems