Formal Languages and Automata Theory

by

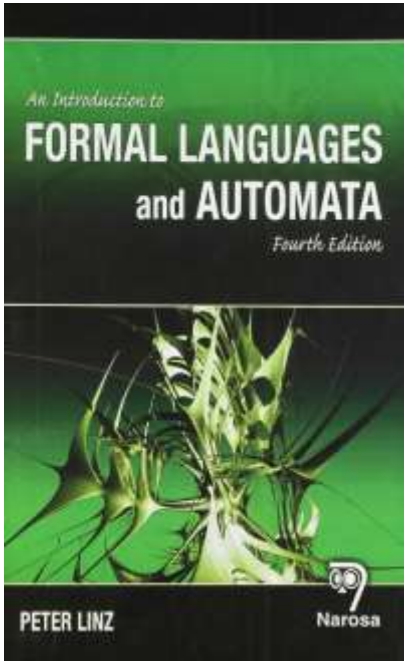
Dr. Pratik Roy

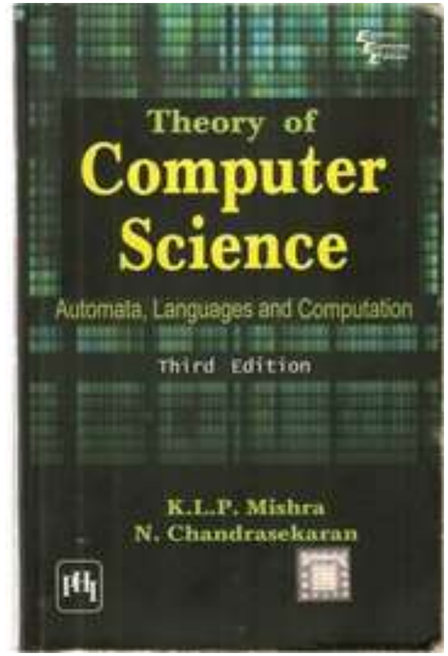
BCSC0011: THEORY OF AUTOMATA & FORMALLANGUAGES

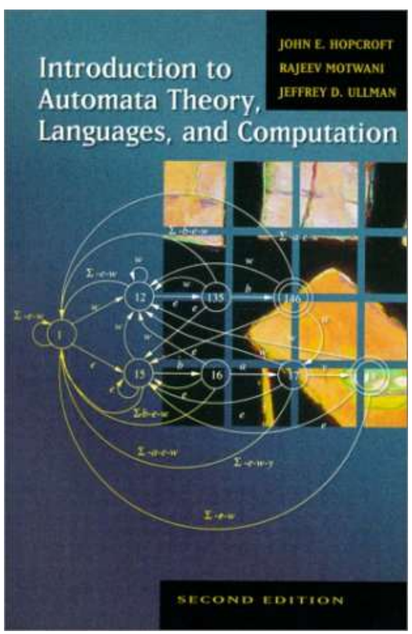
Objective: *The objective of this course is that students will study and compare different models and views of the abstract notion of computation and its various aspects.*

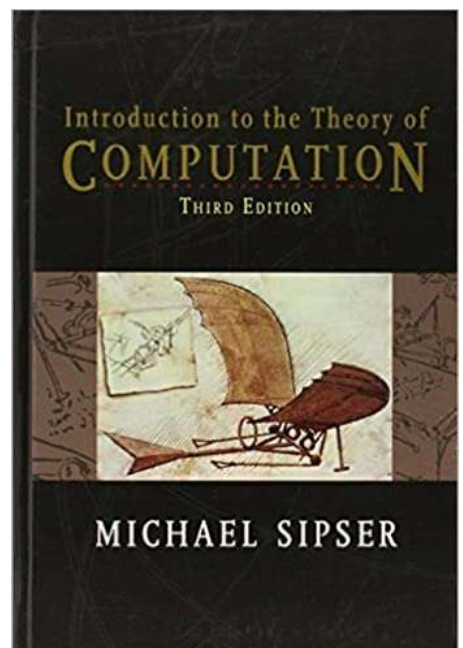
Credits:04 **Semester V** **L-T-P-J:3-1-0-0**

Module No.	Content	Teaching Hours
I	**Introduction:** Alphabets, Strings and Languages; Automata and Grammars, Deterministic Finite Automata (DFA), Nondeterministic Finite Automata (NFA), Equivalence of NFA and DFA, Minimization of Finite Automata, Myhill-Nerode Theorem; FA with Output - Moore and Mealy machine, Applications and Limitations of FA. **Regular expression (RE):** Regular Expression to FA, DFA to Regular Expression, Arden Theorem, Non Regular Languages, Pumping Lemma for Regular Languages, Applications of Pumping Lemma, Closure Properties of Regular Languages. **Push Down Automata (PDA):** Introduction, Language of PDA, Acceptance by Final State, Acceptance by Empty Stack, Deterministic PDA.	20
II	**Context Free Grammar (CFG) and Context Free Languages (CFL):** Introduction, Derivation Trees, Ambiguity in Grammar, Ambiguous to Unambiguous CFG, Simplification of CFGs, Normal Forms for CFGs - CNF and GNF; Pumping lemma for CFLs, Equivalence of PDA and CFG. **Turing machines (TM):** Basic Model, Definition and Representation, Variants of Turing Machine and their equivalence, TM for Computing Integer Functions, Universal TM, Church's Thesis, Recursive and Recursively Enumerable Languages, Halting Problem, Introduction to Computational Complexity.	20


An Introduction to
FORMAL LANGUAGES
and AUTOMATA
Fourth Edition
PETER LINZ
Narosa


Theory of
Computer
Science
Automata, Languages and Computation
Third Edition
K.L.P. Mishra
N. Chandrasekaran


Introduction to
Automata Theory,
Languages, and Computation
JOHN E. HOPCROFT
RAJEEV MOTWANI
JEFFREY D. ULLMAN
SECOND EDITION


Introduction to the Theory of
COMPUTATION
THIRD EDITION
MICHAEL SIPSER

What is Computation

- Computation is the step by step solution to a problem.
- **Example**
  - Multiplication of 2 numbers
  - Finding a word in a dictionary
  - Checking whether there is a path between two vertices in a graph

Preliminaries

❑ Alphabets

- *An alphabet is a finite, non-empty set of symbols*
- Usually, we use the symbol ∑ (sigma) to denote an alphabet
- Examples:
  - Binary: ∑ = {0,1}
  - All lower case letters: ∑ = {a,b,c,...,z}

❑ Strings

- *A string or word is a finite sequence of symbols chosen from some alphabet ∑.*
- Examples:
  - 01101 and 111 are strings from Σ = {0,1}
  - abc and bbb are strings from Σ = {a, b, … , z}

Strings

- **Empty (or null) string** is
  - string with zero occurrences of symbols
  - denoted by ε (“epsilon”)
- **Length of a string** w is
  - the *number of (non- ε) symbols in the string*
  - denoted by $|w|$

    E.g., x = 010100 $|x| = 6$, $|aa| = 2$, $|\varepsilon|=0$
- **Concatenation** of two strings x and y is denoted by xy.
  - E.g., x=abd, y=ce, xy=abdce

$$\varepsilon w = w\varepsilon = w \qquad \forall w$$

ε is the *identity for concatenation*.

- **Reversal**: ω=abd, ω^R = dba

Strings

- Any string of consecutive symbols in some w is said to be a **substring** of w.

E.x., aa$\underline{\text{ab}}$c, $\underline{\text{aaa}}$bc, aaab$\underline{\text{c}}$

- **Powers of an alphabet**

Let ∑ be an alphabet.

- Σ^k = set of all strings of length *k*

  E.g., Σ={a,b}, Σ^2={ab, ba, aa, bb}, Σ^0={ε}

  What is Σ^1? Is Σ^1 different from Σ? How?

- Set of all strings over ∑ is denoted by ∑*.

  Kleene Closure $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots = \cup_{k \geq 0} \Sigma^k$

  E.g., Σ={a,b}, Σ^* = {ε, a, b, ab, aa, ba, bb, aaa, aab, abb, … } is the set of all strings formed by a’s and b’s.

Strings

- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots = \cup_{k>0} \Sigma^k$

i.e., Σ^* without the empty string.

- $\Sigma^+ = \Sigma^* - \{\varepsilon\}$

- $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$

Σ^* and Σ^+ are always infinite.

Languages

- A language is a set of strings over an alphabet.
- *L is said to be a language over alphabet ∑, only if L $\subseteq$ ∑**
  - ➔ this is because ∑* is the set of all strings (of all possible length including 0) over the given alphabet ∑

- A string in a language L is called a **sentence** of L.
  - Examples:
    - Σ={a, b, c, … , z}, the set L of all legal English words is a language over Σ.
    - Σ={0, 1}, L=$\{0^n1^n \mid n \geq 1\}$ is a language over Σ consisting of the strings {01, 0011, 000111, … }

Languages

- Let L be *the* language of all strings consisting of *n* 0's followed by *n* 1's:

  $L = \{\varepsilon, 01, 0011, 000111, \ldots\}$

- Let L be *the* language of all strings with equal number of 0's and 1's:

  $L = \{\varepsilon, 01, 10, 0011, 1100, 0101, 1010, 1001, \ldots\}$

- ∑* is a language for any alphabet ∑.

- Ø denotes the empty language. (have no strings)

- The set {ε} is a language over any alphabet. (consisting of only empty string)

- Is $\phi = \{\varepsilon\}$?

NO

Membership Problem

- In automata theory, a problem is to decide whether a given string is a member of some particular language.

Given a string $w \in \Sigma^*$ and a language L over Σ, decide whether or not $w \in L$.

<u>Example:</u>

Let w = 100011

Q) Is $w \in$ the language of strings with equal number of 0s and 1s?