# UNIT II
# REGULAR EXPRESSIONS AND REGULAR SETS

# UNIT II SYLLABUS

- **REGULAR EXPRESSIONS AND REGULAR SETS :** Regular Expressions and Identities for Regular Expressions, Finite Automata and Regular Expressions: Transition System Containing null moves, NDFA with null moves and Regular Expressions, Conversion of Non-deterministic Systems to Deterministic Systems, Algebraic Methods using Arden's Theorem, Construction of Finite Automata Equivalent to a Regular Expression, Equivalence of Two Finite Automata and Two Regular Expressions, Closure Properties of Regular Sets, Pumping Lemma for Regular Sets and its Application, Equivalence between regular languages: Construction of Finite Automata Equivalent to a Regular Expression, Properties of Regular Languages, Non-deterministic Finite Automata with Null Moves and Regular Expressions, Myhill-Nerode Theorem

# REGULAR EXPRESSIONS

- The language accepted by finite automata can be easily described by simple expressions called Regular Expressions. It is the most effective way to represent any language.

- The languages accepted by some regular expression are referred to as Regular languages.

- A regular expression can also be described as a sequence of pattern that defines a string.

- Regular expressions are used to match character combinations in strings. String searching algorithm used this pattern to find the operations on a string.

- In a regular expression, $x^*$ means **zero or more** occurrence of x.
  - It can generate {e, x, xx, xxx, xxxx, .....}

- In a regular expression, $x^+$ means **one or more** occurrence of x.
  - It can generate {x, xx, xxx, xxxx, .....}

# REGULAR EXPRESSIONS

- A **Regular Expression** can be recursively defined as follows –
- **ε** is a Regular Expression indicates the language containing an empty string. **(L (ε) = {ε})**
- **ϕ** is a Regular Expression denoting an empty language. **(L (ϕ) = { })**
- **x** is a Regular Expression where **L = {x}**

# OPERATIONS ON REGULAR LANGUAGE

- **Union:** If L and M are two regular languages then their union L U M is also a union.

  L U M = {s | s is in L or s is in M}

- **Intersection:** If L and M are two regular languages then their intersection is also an intersection.

  L ∩ M = {st | s is in L and t is in M}

- **Kleen closure:** If L is a regular language then its Kleen closure L1* will also be a regular language.

  L* = Zero or more occurrence of language L.

# Some RE Examples

| Regular Expressions | Regular Set |
|---|---|
| $(0 + 10^*)$ | L = { 0, 1, 10, 100, 1000, 10000, … } |
| $(0^*10^*)$ | L = {1, 01, 10, 010, 0010, …} |
| $(0 + \varepsilon)(1 + \varepsilon)$ | L = {ε, 0, 1, 01} |
| $(a+b)^*$ | Set of strings of a's and b's of any length including the null string. So L = { ε, a, b, aa , ab , bb , ba, aaa…….} |
| $(a+b)^*abb$ | Set of strings of a's and b's ending with the string abb. So L = {abb, aabb, babb, aaabb, ababb, …………..} |
| $(11)^*$ | Set consisting of even number of 1's including empty string, So L= {ε, 11, 1111, 111111, ……….} |
| $(aa)^*(bb)^*b$ | Set of strings consisting of even number of a's followed by odd number of b's , so L = {b, aab, aabbb, aabbbbb, aaaab, aaaabbb, …………..} |
| $(aa + ab + ba + bb)^*$ | String of a's and b's of even length can be obtained by concatenating any combination of the strings aa, ab, ba and bb including null, so L = {aa, ab, ba, bb, aaab, aaba, …………..} |

# REGULAR SETS

- Any set that represents the value of the Regular Expression is called a **Regular Set.**

- **Properties of Regular Sets**

- **Property 1**: *The union of two regular set is regular.*
- **Property 2:***The intersection of two regular set is regular.*
- **Property 3:** *The complement of a regular set is regular.*
- **Property 4:** *The difference of two regular set is regular.*
- **Property 5:***The reversal of a regular set is regular.*
- **Property 6:***The closure of a regular set is regular.*
- **Property 7:** *The concatenation of two regular sets is regular.*

# Identities Related to Regular Expressions

Given R, P, L, Q as regular expressions, the following identities hold –

- $\varnothing^* = \varepsilon$
- $\varepsilon^* = \varepsilon$
- $RR^* = R^*R$
- $R^*R^* = R^*$
- $(R^*)^* = R^*$
- $RR^* = R^*R$
- $(PQ)^*P = P(QP)^*$
- $(a+b)^* = (a^*b^*)^* = (a^*+b^*)^* = (a+b^*)^* = a^*(ba^*)^*$
- $R + \varnothing = \varnothing + R = R$ (The identity for union)
- $R\,\varepsilon = \varepsilon\,R = R$ (The identity for concatenation)
- $\varnothing\,L = L\,\varnothing = \varnothing$ (The annihilator for concatenation)
- $R + R = R$ (Idempotent law)
- $L\,(M + N) = LM + LN$ (Left distributive law)
- $(M + N)\,L = ML + NL$ (Right distributive law)
- $\varepsilon + RR^* = \varepsilon + R^*R = R^*$

# EXAMPLES OF REGULAR EXPRESSION

1. Write the regular expression for the language accepting all the string containing any number of a's and b's.

2. Write the regular expression for the language accepting all combinations of a's, over the set $\sum = \{a\}$.

3. Write the regular expression for the language accepting all combinations of a's except the null string, over the set $\sum = \{a\}$.

4. Write the regular expression for the language accepting all the string which are starting with 1 and ending with 0, over $\sum = \{0, 1\}$.

5. Write the regular expression for the language starting with a but not having consecutive b's.

6. Write the regular expression for the language starting and ending with a and having any combination of b's in between.

7. Write the regular expression for the language accepting all the string in which any number of a's is followed by any number of b's is followed by any number of c's.

8. Write the regular expression for the language over $\sum = \{0\}$ having even length of the string.

9. Write the regular expression for the language L over $\sum$ ={0,1} such that all the string do not contain the substring 01.

# POLLING QUESTIONS

1. Regular expression for all strings starts with ab and ends with bba is:

   A) aba*b*bba
   B) ab(ab)*bba
   C) ab(a+b)*bba
   D) All of the mentioned

# Conversion of RE to FA

- To convert the RE to FA, we are going to use a method called the subset method. This method is used to obtain FA from the given regular expression. This method is given below:

- **Step 1:** Design a transition diagram for given regular expression, using NFA with ε moves.
- **Step 2:** Convert this NFA with ε to NFA without ε.
- **Step 3:** Convert the obtained NFA to equivalent DFA.

# Examples

1. Design a FA from given regular expression

$$10 + (0 + 11)0^{*}\ 1$$

2. Design a NFA from given regular expression

$$1\ (1^{*}\ 01^{*}\ 01^{*})^{*}$$

# CONVERSION OF FA TO RE
# ARDEN’S THEOREM

- The Arden's Theorem is useful for checking the equivalence of two regular expressions as well as in the conversion of DFA to a regular expression.

# ALGORITHM

- **Arden's theorem** state that:

  "If P and Q are two regular expressions over ∑, and if P does not contain epsilon, then the following equation in R given by R = Q + RP has an unique solution i.e., R = QP*."

  That means, whenever we get any equation in the form of R = Q + RP, then we can directly replaced by R = QP*.

# PUMPING LEMMA FOR REGULAR EXPRESSIONS

- **Theorem**
- Let L be a regular language. Then there exists a constant **'c'** such that for every string **w** in **L** –

$$|w| \geq c$$

- We can break **w** into three strings, **w = xyz**, such that –

$$|y| >= 1$$

$$|xy| \leq c$$

For all $k \geq 0$, the string $xy^kz$ is also in L.

# Applications of Pumping Lemma

- Pumping Lemma is to be applied to show that certain **languages are not regular.** It should never be used to show a language is regular.

- If **L** is regular, it satisfies Pumping Lemma.
- If **L** does not satisfy Pumping Lemma, it is non-regular.

# Method to prove that a language L is not regular

- At first, we have to assume that **L** is regular.
- So, the pumping lemma should hold for **L**.
- Use the pumping lemma to obtain a contradiction –
  - Select **w** such that **|w| ≥ c**
  - Select **y** such that **|y| ≥ 1**
  - Select **x** such that **|xy| ≤ c**
  - Assign the remaining string to **z.**
  - Select **k** such that the resulting string is not in **L.**
- **Hence L is not regular.**

- Thus, if a language is regular, it always satisfies pumping lemma.

- If there exists at least one string made from pumping which is not in L, then L is surely not regular.

# Practice Question

1. Prove $L_{01} = \{a^n b^n \mid n \geq 1\}$ is irregular.

# POLLING QUESTIONS

1. While applying Pumping lemma over a language, we consider a string w that belong to L and fragment it into ___________ parts.
   a) 2
   b) 5
   c) 3
   d) 6

2. If we select a string w such that w∈L, and w=xyz. Which of the following portions cannot be an empty string?

a) x

b) y

c) z

d) all of the mentioned

3. Answer in accordance to the third and last statement in pumping lemma:

For all ________ $xy^iz \in L$

a) i>0

b) i<0

c) i<=0

d) i>=0