Unit 3.1Regular Expressions

- Regular Expressions,
- Regular Operators,
- Regular Languages and their applications,
- Algebraic Rules for Regular Expressions

Introduction

- *Regular expressions* are an algebraic expressions used to describe languages.
- Regular Expressions describe exactly the regular languages only.
- If r is any regular expression, then L(r) is a language that is describes by the RE r
- We will describe RE's and their languages recursively.

Regular Operators

- To study about regular language and regular expression we must know about some regular operators.
- The following operators are called regular operators.

  1. **Union(∪):** $L_1 \cup L_2 = \{s \mid s \in L_1 \text{ or } s \in L_2\}$
  2. **Concatenation (.):** $L_1.L_2 = L_1L_2$
     $= \{ st \mid \mathbf{s} \in L_1 \text{ and } \mathbf{t} \in L_2 \}$
  3. **Kleene closure(*):** $L^* = \bigcup_{i=0}^{\infty} L^i$

- NOTE: For regular expression Union operator is replaced by +

Regular Operators: Example

1. $L_1 = \{11,00\}$, $L_2 = \{01, 10\}$ then

   $L_1 \cup L_2 = \{11,00\} \cup \{01,10\} = \{11,00,01,11\}$

2. $L_1.L_2 = \{1101,1110,0001,0010\}$

3. Let $L = \{0,1\}$ then

   $L^* = \{0,1\}^* = L^0 \cup L^1 \cup L^2 \cup$ ....

   $= \{\epsilon\} \cup \{0,1\} \cup \{00,01,10,11\}$ ....

- If ϵ is excluded from Kleene closure of L then it is termed as positive closure and denoted by L^+ and $L^+ = L^* - \{\epsilon\}$

Regular Languages

- **Basic Regular Language**:
  - The langage L={} or ϕ , the empty language is basic regular language.
  - The language L={ϵ}, the language of empty string is basic regular language.
  - For any symbol a ϵ Σ , L={a} is a basic regular language.
- **Recursive Definition of RE:**
  - If L_1 and L_2 are regular languages then,
    - The Union of two regular languages $L_1 \cup L_2$ is regular.
    - The Concatenation of two regular language $L_1.L_2$ or L_1L_2 is also regular.
    - The Kleen closure of L_1 i.e. L_1^* is also regular

Regular Languages : Example

- Below are the examples of regular languages over the alphabets {0,1}
- Languages
  1. { } – The empty Language
  2. {0} – The language for only string 0
  3. {1} – The language for only string 1
  4. {001} Or {{0}{0}{1}} – Concatenation of three language 2,2 and 3
  5. {0,1} or {0} $\cup$ {1} - Union of two languages 2 and 3
  6. {0,1,001} – Union of languages in 2,3 and 4
  7. {0,1}* - Kleen closure of language 5
  8. {0,1}* {001} – Concatenation of Languages 7 and 4

Regular Expression: Definition

- Basis 1: If a ϵ Σ is any symbol, then **a** is a RE, representing language L(**a**) = {a}.
  - Note: {a} is the language containing one string, and that string is of length 1.
- Basis 2: ϵ is a RE, for language L(ϵ) = {ϵ}.
- Basis 3: $\varnothing$ is a RE, for L($\varnothing$) = $\varnothing$.

RE: Definition – (2)

- Induction:

1: If r_1 and r_2 are regular expressions for $L(r_1)$ and $L(r_2)$ respectively then r_1+r_2 is a regular expression for language $L(r_1+r_2) = L(r_1) \cup L(r_2)$.

2: If r_1 and r_2 are regular expressions, then r_1r_2 is a regular expression for language $L(r_1r_2) = L(r_1)L(r_2)$.

3: If r is a RE, then r* is a RE, for language $L(r^*) = (L(r))^*$.

Regular Languages : Example

- Below are the examples of regular languages and corresponding RE's over the alphabets {0,1}
- Languages RE

Languages	RE
1. { }	**ϕ**
2. {ϵ}	ϵ
3. {0}	**0**
4. {1}	**1**
5. {001} Or {{0}{0}{1}}	**001**
6. {0,1}	**0+1**
7. {0,1,001}	**0+1+001**
8. {0,1}*	**(0+1)***
9. {0,1}* {001}	**(0+1)*(001)**
10. All strings of 0's and 1's without two consecutive 1's.	($\mathbf{0}$+$\mathbf{10}$)*(ϵ+$\mathbf{1}$)
11. All strings ending with 00	**(0+1)*00**

Precedence of Regular Operators

- Among the regular operators described in previous slides,
- The Kleen closure * is given highest precedence.
- Then concatenation (.) has next highest precedence.
- The union operator + lowest precedence.
- Parentheses may be used to change the order of precedence wherever needed to influence the grouping of operators.

Precedence of Regular Operators

In regular expressions below:

- **10*** is equivalent to **1(0)*** since * has precedence over .
- **10*** is different from (10)*
- **1+01+10*, (1+0)(1+(10)*) 1+01+(10)*** are different regular expressions
- **01* +1 and (01)*+1** represent two different RE
- **0(1 + 10*) = 0(1 + 1(0)*)** and not equal to **0(1+(10)*)**

Some Examples of RE

- All strings from {0,1}

  (0+1)*

- All strings ending with 01

  (0+1)*01

- All strings starting with 00 and ending with 11

  00(0+1)*11

- All strings starting or ending with 00
  - **Starting with 00**

    00(0+1)*
  - **Ending with 00**

    (0+1)*00
  - Finally starting or ending with 00: **00(0+1)*+(0+1)*00**

- All strings having 000 as substring

  (0+1)*000(0+1)*

Some Examples of RE

- All strings containing 0 or more no of 1’s followed by at least one 0

  1*0+

e.g. { 0, 10,110, 100, 00000,11110000…………..}

- All strings on {0,1} such that 0’s if any must occur before 1’s if any
  - **0*1***

e.g. : {ϵ,0,1,01,00,11,001,011,0011 ……………..}

- Strings over {0,1} that start with 0 and end with 1 and all 0’s are to the left of 1’s.

  $\mathbf{0^+1^+}$

Denotes the set { 01,001,0011,011,………………}

- Strings with length exactly 2

  (00+01+10+11) or (0+1)(0+1)

Some Examples of RE

- Let $\Sigma = \{0,1\}$ and $L \in \Sigma^*$ then
  - R.E. of language L containing even length of strings.
    - Since 0 is even , ϵ belongs to L
    - Any string of even length can be obtained by concatenating zero or more strings of length 2.
    - i.e. $L= \{00,01,10,11\}^*$
    - The corresponding R.E. is $(00+01+10+11)^*$
    - Equivalent RE is $((0+1)(0+1))^*$

Algebraic Rules for RE

- **Commutativity**: The Union of two R.E. is commutative. i.e. if **r** and **s** are two REs representing languages R and S, then

r+s = s+r representing language $R \cup S$ or $S \cup R$

- **Associativity**: The Union and concatenation operation of RE are associative i.e.

  If l, r,s are REs representing languages L, R, and S respectively then $L \cup (R \cup S) = (L \cup R) \cup S$ and corresponding RE is **l+(r+s) = (l+r)+s**

- Similarly, L(RS) = (LR)S and RE is **l(rs) = (lr)s**

Algebraic Rules for RE

- **Identities:**
  - ϕ is the identity for Union operation i.e. $\boldsymbol{\phi+r = r+\phi = r}$ for any RE **r**
  - ϵ is identity for concatenation i.e. $\boldsymbol{\epsilon r = r\epsilon = r}$ for any RE r
- **Annihilator**: An annihilator for an operator is a value such that when operator is applied with that value and another value , the result of operation is the annihilator. ϕ is an annihilator for concatenation
  - i.e. **$\boldsymbol{\phi r = r\phi = \phi}$ for any RE r.**
- **Idempotent Law for Union**: This law states that Union of two same expression can be replaced by the same single expression.
  - i.e. **$\boldsymbol{r + r = r}$ for any RE r.**

Algebraic Rules for RE

- **Laws of closures**: There are different rules involving closure :
  - Kleene Closure of the Kleene closure of a RE is Kleene closure of the RE itself.

    i.e. $\mathbf{(r^*)^* = r^*}$
  - Kleene closure of ϕ is i.e. $\phi^* = \epsilon$
  - Kleene closure of ϵ is i.e. $\epsilon^* = \epsilon$
  - The positive closure of RE r is concatenation of r with its Kleene closure
    - **i.e.** $\mathbf{r^+ = rr^* = r^* r}$
  - The union of positive closure with ϵ is Kleen closure i.e. $\mathbf{r^* = r^+ + \epsilon}$

Algebraic Rules for RE

- **The Distributive Law**: Regular Expressions follow distributive law of concatenation over union.
  - Let l , m and n are REs representing languages L,M, and N respectively then
    - $L(M \cup N) = LM \cup LN$ which is left distributive rule
    - $(L \cup M)N = LN \cup MN$ which is right distributive rule.
- **The algebraic rules described above are very useful for the simplification of the regular expressions.**

Proof of Distributive Rules

- **Theorem**: if L,M, N are any languages

  then $L(M \cup N) = LM \cup LN$
- **Proof:** Let *w* is a string such that $w = xy$. We have to show that w $L(M \cup N)$ iff *w* $LM \cup LN$.
- **(if)** : $w \in LM \cup LN \Rightarrow xy \in LM$ or $xy \in LN$ (by union rule)
  - $xy \in LM \Rightarrow x \in L$ and $y \in M$ (by concatenation rule)
  - $xy \in LN \Rightarrow x \in L$ and $y \in N$ (by concatenation rule)
  - This implies $x \in L$ and $y \in M \cup N$ i.e. $xy \in L(M \cup N)$.

    (concatenation of above)
- **Hence $w \in L(M \cup N)$.**
- **(Only if):** $w \in L(M \cup N) \Rightarrow xy \in L(M \cup N)$
  - i.e. $x \in L$ and $y \in M$ or $y \in N$ (By the Union rule)
  - if $y \in M$ then $xy \in LM$ (by concatenation rule)
  - if $y \in N$ then $xy \in LN$ (by concatenation rule)
  - This implies $xy \in LM \cup LN$ (Union of above)
- Hence **$w \in LM \cup LN$**
- This completes the proof.