I

Regular Expression

Regular Expression are used for representing certain sets of strings in an algebraic fasion.

① Any terminal symbol (i.e) symbols $\in \Sigma$ are regular expressions, including $\wedge$ and ϕ

a, b, c $\quad \wedge, \phi$

② The Union of two regular expressions is also a regular expression.

Let R_1 & $R_2 \rightarrow$ Regular $(R_1 \cup R_2) \rightarrow$ Regular

③ The concatenation of two regular expressions is also a regular expression.

$R_1, R_2 \rightarrow$ regular $\quad (R_1 \cap R_2) \rightarrow$ Regular

④ The iteration (or closure) of a regular expression is also a regular expression

$R_1 \rightarrow R_1^*$ (closure) $\Rightarrow$ Regular $\quad a^* \Rightarrow \wedge, a, aa, aaa, \ldots$ $\rightarrow$ Every thing that includ a.

⑤ The regular expression over Σ are precisely those obtained recursively by the application of the above rules once or several times.

RE obtained by above ①-④ rules once or more times $\Rightarrow$ Regular Expression.

Examples Of RE

Describe the following sets as Regular Expressions.

1) $\{0, 1, 2\}$ It can contain 0 or 1 or 2

$R = 0 + 1 + 2$ '+' symbol for 'or'

2) $\{\wedge, ab\}$

$R = \wedge\ ab$ if it is empty symbol '$\wedge$' => no need to use '+'.

3) $\{abb, a, b, bba\}$

$R = abb + a + b + bba$

4) $\{\wedge, 0, 00, 000 \ldots\}$ closure of 0

$R = 0^*$

5) $\{1, 11, 111, 1111 \ldots\}$ not a closure of 1 bcaz this is not a ~~closure~~ containing '$\wedge$'/'ϵ'

$R = 1^+$ → when empty symbol is not present.

Identities of RE

1) $\phi + R = R$ // Union of R with ϕ => Regular Expression
2) $\phi R + R\phi = \phi$ // Concatenate ϕ with R => ϕ (null set)
3) $\epsilon R = R\epsilon = R$ // Concatenate of ϵ with R => Regular.
4) $\epsilon^* = \epsilon$ and $\phi^* = \epsilon$ //
5) $R + R = R$
6) $R^* R^* = R^*$ // concatenating closure of two Regular Expressions => Regular expression closure
7) $RR^* = R^* R$
8) $(R^*)^* = R^*$ // (closure RE) of ~~R~~ => (closure RE)

9) $\epsilon + RR^* = \epsilon + R^*R = R^*$

$RR^* \Rightarrow R^+$

$\Rightarrow \epsilon + R^+ \Rightarrow \underline{\underline{R^*}}$ proved

10) $(PQ)^*P = P(QP)^*$

11) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$

12) $(P+Q)R = PR + QR$ and

$P(P+Q) = RP + RQ$.

III An Example proof using Identities of Regular Expressions.

1) Prove that $(1+00^*1) + (1+00^*1)(0+10^*1)^*(0+10^*1)$ is equal to $0^*1(0+10^*1)^*$

LHS $= \underbrace{(1+00^*1) + (1+00^*1)}(0+10^*1)^*(0+10^*1)$

$= (1+00^*1)[\epsilon + (0+10^*1)^*(0+10^*1)]$ // $\epsilon + R^*R = R^*$

$= (1+00^*1)[(0+10^*1)^*]$ // $\epsilon \cdot R = R$

$= (1+00^*1)(0+10^*1)^*$

$= (\epsilon \cdot 1 + 00^*1)(0+10^*1)^*$ // $\epsilon + R^*R = \epsilon + RR^* = R^*$.

$= (\epsilon + 00^*)1(0+10^*1)^*$

$= (0^*)1(0+10^*1)^*$

$=$ RHS //

11 Designing Regular Expressions - Examples

① Design Regular Expression for the following language over $\{a, b\}$

a) Language accepting strings of length exactly 2

$L = \{aa, ab, ba, bb\}$ // strings in the language

$R = aa + ab + ba + bb$

$= a(a+b) + b(a+b)$

$\boxed{R = (a+b)(a+b)}$ //

Note :- If L accepts strings with length exactly 3 ⇒ the answer can be $(a+b)(a+b)(a+b)$.
4 ⇒ $(a+b)(a+b)(a+b)(a+b)$

ⓑ Language accepting strings of length at least 2 (two or more than 2)

$L_1 = \{aa, ab, ba, bb, aaa, aab \ldots\}$

$R = \underbrace{(a+b)(a+b)}_{\text{strings of length two}} \underbrace{(a+b)^*}_{\text{length two or more than two}}$

Hence soln :

$\boxed{R = (a+b)(a+b)(a+b)^*}$

ⓒ Language accepting strings of length atmost 2.

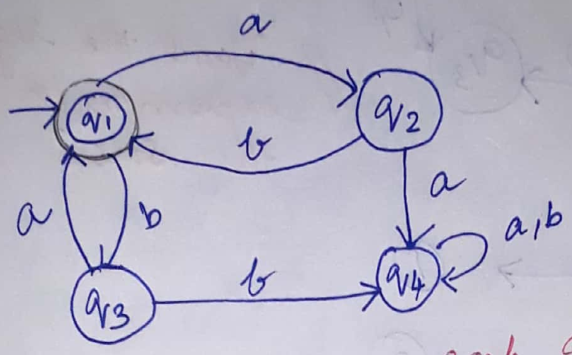
$L = \{\epsilon, a, b, aa, ab, ba, bb\}$

$R = \epsilon + a + b + aa + ab + ba + bb$

$\boxed{R = (\epsilon + a + b)(\epsilon + a + b)}$

V. Designing Regular Expression Examples.

(a) Finding the Regular Expression for the following DFA



⇒ q_1 is the initial & final states

Equations for the each of the states.

$$q_1 = \epsilon + q_2 b + q_3 a \rightarrow (1)$$

$$q_2 = q_1 a \rightarrow (2)$$

$$q_3 = q_1 b \rightarrow (3)$$

$$q_4 = q_2 a + q_3 b + q_4 (a + b) \rightarrow (4)$$

(1) ⇒ $q_1 = \epsilon + q_2 b + q_3 a$

Putting values of q_2 and q_3 from (2) & (3)

$$q_1 = \epsilon + q_1 a b + q_1 b a$$

$$q_1 = \epsilon + q_1 (ab + ba) \quad // R = Q + RP$$

$$\underbrace{q_1}_{R} = \underbrace{\epsilon}_{Q} + \underbrace{q_1}_{R} \underbrace{(ab + ba)}_{P} \quad // R = QP^* \text{ (Arden's Theorem)}$$

$$q_1 = \epsilon (ab + ba)^* \quad // \epsilon R = R$$

$$\boxed{q_1 = (ab + ba)^*}$$

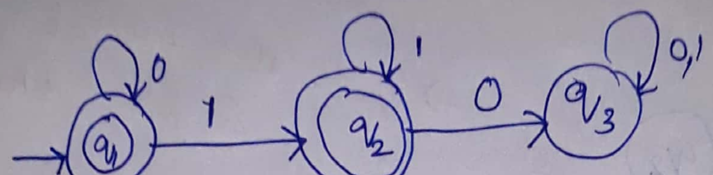
// q_1 is the final state, so we got the expression for q_1.

Hence this is the required regular expression.

VI

Designing Regular Expression – Examples (Part-4)

(when there are multiple final states)



// Find the regular expressions for all the states.

$q_1 = \epsilon + q_1 0 \longrightarrow (1)$

$q_2 = q_1 1 + q_2 1 \longrightarrow (2)$

$q_3 = q_2 0 + q_3 0 + q_3 1 \longrightarrow (3)$

Final state

(1) $\Rightarrow$

$\underbrace{q_1}_{R} = \underbrace{\epsilon}_{Q} + \underbrace{q_1}_{R}\underbrace{0}_{P}$ // $R = Q + RP$

$R = QP^*$ {Arden's Theorem}

$q_1 = \epsilon 0^*$ // $\epsilon R = R$

$\boxed{q_1 = 0^*} \longrightarrow (4)$

$\Rightarrow$ Regular Expression for final state q_1

Take eqn (2) {final state q_2}

$q_2 = q_1 1 + q_2 1$ // sub. the value of q_1

$\underbrace{q_2}_{R} = \underbrace{0^* 1}_{Q} + \underbrace{q_2}_{R}\underbrace{1}_{P}$ // $R = Q + RP$

$= QP^*$

$\boxed{q_2 = 0^* 1 (1)^*} \longrightarrow (5)$ // Regular expression of q_2 final states.

If there are two final states, the Regular expression will be the Union of both final states

$$R = 0^* + 0^* 1 (1)^*$$

$$= 0^* (\epsilon + 1 1^*) \qquad // \ \epsilon + RR^* = R^*$$

$$= 0^* (1^*)$$

$$\boxed{R = 0^* 1^*}$$ // Required Regular expression for DFA.