Regular Expression

A regular expression is a concise way to describe a regular language. These are symbolic notations used to define search patterns in strings.

For a given character set Σ of a language L, the regular expression is defined by the following rules:

Rule 1: Every character belonging to Σ is a regular expression. $r = a$.

Rule 2: Null string ϵ is a regular expression. $r = \epsilon$.

Rule 3: Empty language is a regular expression. $r = \phi$.

Rule 4: If R_1 & R_2 are two regular expressions, then $R_1 R_2$ is also a regular expression (concatenation of two regular expressions is a RE).

Rule 5: If R_1 & R_2 are two RE, then $R_1 + R_2$ is a RE (Union of two R.E. is a R.E.).

Rule 6: If R is a R.E., then (R) is a R.E.

Rule 7: If R is a R.E., then R^*, R^+, R^2 is also a R.E. (Power of a R.E. is a R.E.).

Rule 8: Any combination of the preceding rules is also a R.E.

i.e. there are three types of basic (atomic) R.E.
'a', 'ϵ', 'ϕ'

& there are 4 types of operations on R.E.
'+', '.', Power ('+', '*', exact value), Parantheses()
Union, concatenation

__/__/____

High to low Precedence of Operators:

( ), Power, . , +

Highest → Lowest.

Regular Set: Set of strings generated by regular expression is known as Regular Set. It is the language corresponding to a regular expression & is also known as regular language.

Some Regular Language corresponding to R.E.

R.E.	Regular Set
a	{a}
a+b+c	{a, b, c}
ab+ba	{ab, ba}
ab(d+e)	{abd, abe}
(aa)*	{ε, aa, aaaa, aaaaaa, ---}
(a+b)(c+d+e)u	{acu+adu+aeu+bcu+bdu+beu}
abc*de	{abde, abcde, abccde, ---}
(a+b)c*	{a, b, ac, bc, acc, ---}
(abc)*	{ε, abc, abcabc, ---}
(0+11)2(33+1)	{021, 0233, 11233, 1121}
a*a	{a, aa, aaa, ---}
(abc)*k	{k, abck, abcabck, ---}
ϕ	{ }
(a+b+ε)	{ε, a, b}
ε	{ε}
(abc)k*	{abc, abck, abckk, ---}
ab(c+d)e*	{abc, abd, abce, abde, ---}

__/__/____

Regular expression corresponding to regular lang

Language	R.E.
set of odd length of a, min length 3. $L = \{aaa, aaaaa, ----\}$	$(aa)^+a$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{0,1\}$, w begins with 00 & ends with 11$\}$	$00(0+1)^*11$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{0,1\}$, such that w contains alternates 0s & 1s$\}$	$0(10)^* + 0(10)^*1 + 1(01)^* + 1(01)^*0$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{0,1\}$, w ends with 11$\}$	$(0+1)^*11$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{a,b,c\}$, 'ccc' is a substring of w$\}$	$(a+b+c)^*ccc(a+b+c)^*$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{a,b\}$, w contains exactly 2 a's	$b^*ab^*ab^*$
atleast 2 a's	$(a+b)^*a(a+b)^*a(a+b)^*$
atmost 2 a's	$b^*(a+\epsilon)b^*(a+\epsilon)b^*$
$L = \{0^i1^j2^k$, where $i,j,k \in N\}$	$0^+1^+2^+$
$L = \{0^i1^j2^k$, where $i,j,k \geq 0\}$	$0^*1^*2^*$
$L = \{0^i1^j2^k$, where $i,j \geq 1, k \geq 0\}$	$0^+1^+2^*$
$L = \{w \mid w \in \Sigma^*, \Sigma = \{0,1\}$, w contains no. of 1's divisible by 3$\}$	$0^*(10^*10^*10^*)^*$

__/__/____

Identities related to Regular Expression :

Two regular expressions R_1 & R_2 are equivalent if their corresponding languages are same, i.e. $L(R_1) = L(R_2)$. following are some such Regular expressions which are equivalent :

1.) $\phi + R = R + \phi = R$
2.) $\epsilon + R = R + \epsilon$
3.) $\epsilon R = R\epsilon = R$
4.) $R + R = R$
5.) $\epsilon^* = \epsilon$
6.) $RR^* = R^*R = R^+$
7.) $\epsilon + RR^* = \epsilon + R^*R = R^*$
8.) $(P+Q)R = PR + QR$
9.) $(PQ)^*P = P(QP)^*$
10.) $(R^*)^* = R^*$
11.) $(P+Q)^* = (P^* + Q^*)^* = (P^*Q^*)^*$
12.) $\phi R = R\phi = \phi$

Q. Which of the following statements is correct for R_1 & R_2. $R_1 = (aa)^* + a^*)^*$, $R_2 = (a^*)^+$

a) $L(R_1) \subset L(R_2)$ b) $L(R_1) \supset L(R_2)$
c) $L(R_1) \neq L(R_2)$ d) $L(R_1) = L(R_2)$

Solution : d) since R_1 & R_2 both are a^*.

Q. Identify which of the following R.E. can generate all the strings which do not contain the substring 100.

a) $0^*(0+1)^*$ b) 0^*1010^*
c) 0^*1^*01 d) $0^*(10+1)^*$

Solution: d) although c) also does not create strings containing 100, but it also cannot create ϵ, 0, 1 strings which also does not contain 100.

Q. Identify which of the R.E. cannot generate a string 1101.

a) $110^*(0+1)^*$ b) $1(0+1)^*101$
c) $(10)^*(01)^*(00+11)^*$ d) $(00+(11)^*01^*)^*$

Solution: c)

Q. Identify which of the following R.E. correctly represent $L = \{w \in \{0+1\}^* \mid w \text{ has even no. of 1's}\}$.

a) $(0^*10^*1)^*$ b) $0^*(10^*10^*)^*$
c) $0^*(10^*1)^*0^*$ d) $0^*1(10^*1)^*10^*$

Solution: b), a) cannot generate only 0's.
c) can't generate 11011.
d) cannot generate ϵ, 0, 00 ... .

Q. Identify which of the following R.E. represents a language which contains atleast one a and one b.

a) $(a+b)^+$ b) $(a+b)^*a(a+b)^*b(a+b)^*$
c) $(a+b)^*b(a+b)^*a(a+b)^*$ d) $(a+b)^*a(a+b)^*b(a+b)^* + (a+b)^*b(a+b)^*a(a+b)^*$

Sol. d)

__/__/____

Regular Expression From Finite Automata

A finite automata contains following types of paths: serial path, parallel path, loop, cycle. corresponding to these path the regular expression will be: concatenation for serial path, Union for parallel path.
Klnee closure for loop. Similarly for cycle also it is a klnee closure.
For converting a regular expression, we need to find path from initial state to final state.
There are two methods to convert FA to Regular Expression:

1.) Arden's Theorem.
2.) State Elimination.

Arden's Theorem: Let P, Q, and R be the regular expressions over Σ.
If P is not ϵ, then the equation
$R = Q + RP$
has a unique solution QP^*.

Proof: Let's first check QP^* is actually a sol".
Let's substitute the value of QP^* in $R = Q + RP$.

$$QP^* = \underbrace{Q + (QP^*)P}_{Q(\epsilon + P^*P) = Q(P^*) = QP^*}$$

L.H.S = R.H.S.
So yes QP^* is a sol" of $R = Q + RP$.

__/__/___

Now, let's check if it is the only sol^n.

$R = Q + RP$

$= Q + (Q + RP)P = Q + QP + RP^2$

$= Q + QP + (Q + RP)P^2 = Q + QP + QP^2 + RP^3$

$\vdots$

$\vdots$

$= Q + QP + QP^2 + \dots QP^i + RP^{i+1}$

$= Q(\epsilon + P^1 + \dots P^i) + RP^{i+1}$

$= QP^* + RP^{i+1}$

$\Rightarrow R = Q + RP$

has two solutions

QP^* and

RP^{i+1}

Since $P \neq \epsilon$, and $i \geq 0$ hence

$|R| < |RP^{i+1}|$ {length of R is less than RP^{i+1}

& that is not possible, so RP^{i+1} cannot be a solution to R.

Hence the only feasible solution for R is QP^*.

$\therefore$ $R = QP^*$ is a unique sol^n for $R = Q + RP$

Conditions to apply Arden's theorem:

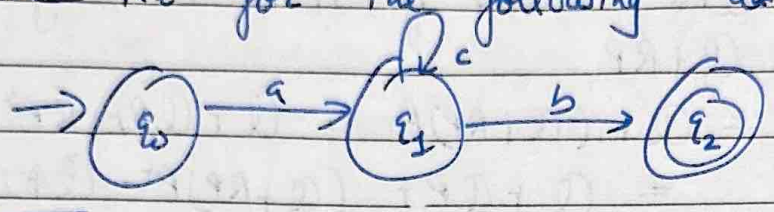
1.) The finite automata should not contain null (ϵ) moves

2.) There should be exactly one initial state.

For finding regular expression using arden's theorem, first ~~con~~ create equations for each state, i.e. find out all the ways to reach at that state. for initial state, ϵ is always a way to reach there.

__/__/____

Q. Find RE for the following automata.



Sol. The transitions required to reach various states in the automata.

$q_0 = \epsilon$ —(1)

$q_1 = q_0 a + q_1 c$ —(2)

$q_2 = q_1 b$ —(3)

Regular expression of the FA will be same as that of final state, so we need to find the value of q_2.

Putting value of (1) in (2).

$q_1 = (\epsilon) a + q_1 c = a + q_1 c$

$(R = Q + RP)$ Applying Arden's Theorem

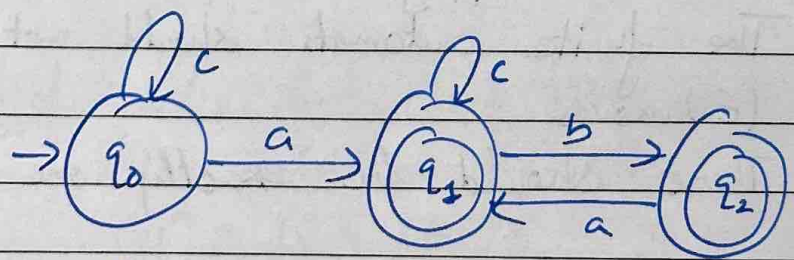
$R = QP^*$ $q_1 = a c^*$ —(4)

Put the value of (4) in (3)

$q_2 = (ac^*) b = ac^*b$

↳ R.E. of FA.

Q.



Sol.

$q_0 = \epsilon + q_0 c$ —(1)

$q_1 = q_0 a + q_1 c + q_2 a$ —(2)

$q_2 = q_1 b$ —(3)

Applying Arden's on (1)

$$q_0 = \epsilon(c)^* = c^* \quad -(4)$$

Put value of q_0 from (4) in (2)

$$q_1 = c^*a + q_1c + q_2a \quad -(5)$$

Put value of q_2 from (3) in (5)

$$q_1 = c^*a + q_1c + (q_1b)a$$

$$q_1 = c^*a + q_1(c+ba) \quad -(6)$$

Applying Arden's theorem on (6)

$$q_1 = c^*a(c+ba)^* \quad -(7)$$

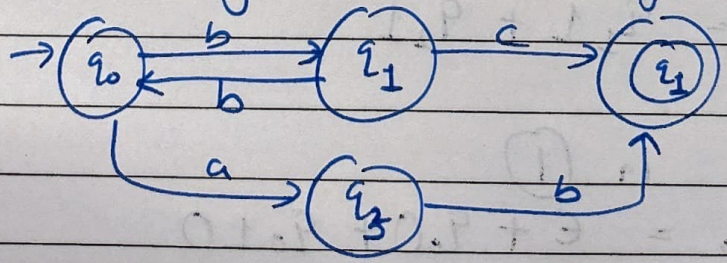
Putting value of (7) in (3)

$$q_2 = c^*a(c+ba)^*b$$

Since there are two final states, so R.E. for the automata is union of R.E. of both the final states. i.e.

$$\underline{c^*a(c+ba)^* + c^*a(c+ba)^*b}$$

Q. Find the regular expression for



Sol.

$$q_0 = \epsilon + q_1b \quad -(1)$$

$$q_1 = q_0b \quad -(2)$$

$$q_2 = q_1c + q_2b \quad -(3)$$

$$q_3 = q_0a \quad -(4)$$

Putting the value of (2) in (1)

$q_0 = \epsilon + (q_0 b) b$

$q_0 = \epsilon + q_0 bb$

Applying Arden's

$q_0 = (bb)^*$ — (3)

Putting the value of (3) in (2) & (4)

$q_1 = (bb)^* b$

$q_3 = (bb)^* a$

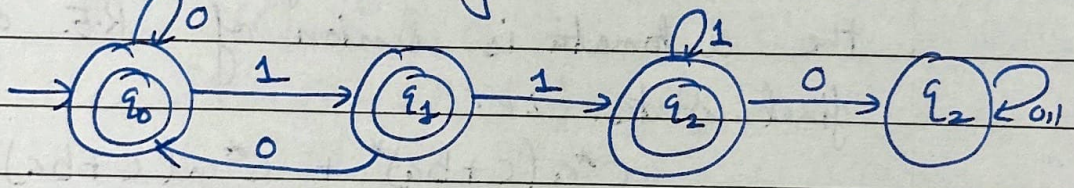
Putting these values in (3).

$q_2 = (bb)^* bc + (bb)^* ab$

since q_2 is the only final state, the R.E. for FA is $\underline{(bb)^* bc + (bb)^* ab}$

Q. Create a regular expression for '110' is not a subset using Arden's theorem.

Sol



$q_0 = \epsilon + q_0 0 + q_1 0$ — (1)

$q_1 = q_0 1$ — (2)

$q_2 = q_1 1 + q_2 1$ — (3)

(2) in (1)

$q_0 = \epsilon + q_0 0 + q_0 10$

$q_0 = \epsilon + q_0 (0 + 10)$

Applying Arden's

$q_0 = \epsilon (0+10)^* = (0+10)^*$ — (4)

Put it in (2)

$q_1 = (0+10)^* 1$ — (5)

Put value of (5) in (3)

$q_2 = (0+10)^* 11 + q_2 1$

Applying Arden's.

$q_2 = (0+10)^* 11 1^*$

Since q_0, q_1 & q_2 all are final states,

Regular expression of automata is

$q_0 + q_1 + q_2$

$(0+10)^* + (0+10)^* 1 + (0+10)^* 11 1^*$

$= (0+10)^* (\epsilon + 1 + 11 1^*)$

State Elimination Method