Unit-2

Regular Expression: are used to represent the strings accepted by Finite Automata (Regular languages) in an algebraic manner

Features of Regular expression:

(1) Any terminal symbol, λ and ϕ are regular expression

(2) The union of two regular expression R_1 and R_2 written as R_1+R_2 is also regular expression

(3) The Concatenation of two regular expression R_1 and R_2, written as R_1R_2 is also regular expression.

(4) The Closure of a regular expression R, written as R^* is also regular expression.

(5) If R is a regular expression, then (R) is also a regular expression.

Regular Set: Any set represented by a regular expression is called "Regular set".

for $a, b \in \Sigma$

(1) Regular expression a is represented by regular set $\{a\}$.

(2) Regular expression $a+b$ is represented by regular set $\{a, b\}$.

(3) Regular expression ab is represented by regular set $\{ab\}$.

(4) Regular expression a^* is represented by regular set $= \{\lambda, a, aa, aaa, aaaa, aaaaa, aaaaaa, --\}$

(5) Regular expression $(a+b)^*$ is represented by regular set $\{a, b\}^*$

Represent the following regular set by regular expressions: (2)

S.No.	Regular Set	Regular expression
1.	{101}	101
2.	{abba}	abba
3.	{01, 10}	01+10
4.	{λ, ab}	λ+ab
5.	{abb, a, b, bba}	abb+a+b+bba
6.	{λ, 0, 00, 000, ---}	0*

Describe the following set by regular expressions

(a) The set of all strings of 0's and 1's ending in 00.

Ans $(0+1)^*00$

(b) The set of all strings of 0's and 1's beginning with 0 and ending with 1.

Ans $0\,(0+1)^*\,1$

(c) $L = \{\lambda, 11, 1111, 111111, ----\}$

Ans $L = \{\lambda, 11, (11)^2, (11)^3 ------\}$

$= (11)^*$

(d) The set of all strings in which every 0 is immediately followed by at least two 1's. $\Sigma = \{0, 1\}$

Ans $(1+011)^*$

Find the regular expression corresponding to each of the following subset of $\{a,b\}$.

(a) The set of all strings containing exactly two a's.

Ans (a) $b^* a b^* a b^*$

(b) The set of all strings containing at least 2 a's.

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

(c) The set of all strings containing atmost 2 a's.

$b^* + b^* a b^* + b^* a b^* a b^*$

(zero a) (one a) (two a)

(d) The set of all strings containing the substring 'aa'.

$(a+b)^* aa (a+b)^*$.

(e) $L = \{ a, aa, aaa, aaaa, \text{---} \}$

$= a\{ \lambda, a, aa, aaa, \text{---} \}$

$= a(a)^*$

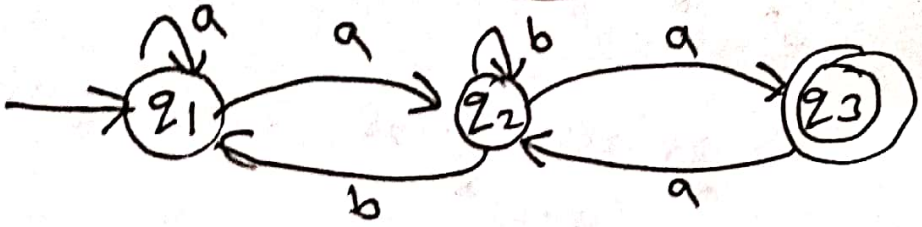
Arden's theorem to Convert NFA to regular expression (Re)

Arden's theorem is used to convert NFA into regular expression.

Arden's theorem: If P and Q are two regular expression over Σ. If P does not λ, then the following equation in R,

$R = Q + RP$ has a unique solution given by $R = QP^*$.

Example Show using Arden's theorem that strings recognized by given transition system is $(a+a(b+aa)^*b)^*a(b+aa)^*a$ (4)



Ans (i) for each state, we need to find the inputs that are coming to that state.

~~(ii) Initial State also contains a regular expression~~

(ii) Kindly ensure that NFA does not have λ moves for using Arden's theorem.

$$q_1 = q_1a + q_2b + \lambda \quad \text{——— (1)}$$

↓ only added in initial state equation.

$$q_2 = q_1a + q_2b + q_3a \quad \text{——— (2)}$$

$$q_3 = q_2a \quad \text{——— (3)}$$

putting the value of q_3 from eq^n (3) to eq^n (2)

$$q_2 = q_1a + q_2b + q_2aa$$

$$\frac{q_2}{R} = \frac{q_1a}{Q} + \frac{q_2(b+aa)}{R\ \ P}$$

(using Arden's theorem $R = Q + RP$ has solution, $R = QP^*$)

$$R = QP^*$$

$$q_2 = q_1a(b+aa)^* \quad \text{— (4)}$$

Putting the value of q_2 from eq^n (4) to eq^n (1)

$$q_1 = q_1a + q_1a(b+aa)^*b + \lambda$$

$$\frac{q_1}{R} = \frac{q_1(a+a(b+aa)^*b)}{P} + \frac{\lambda}{Q}$$

(Use arden's theorem) $R = Q + RP$ has solution $R = QP^*$

$$q_1 = \lambda\cdot(a+a(b+aa)^*b)^*$$

$$q_1 = (a+a(b+aa)^*b)^* \quad \text{— (5)}$$

Putting the value of q_1 from eqn (5) to eqn (4) (5)

$$q_2 = q_1 a (b+aa)^* \quad \text{—(4)}$$

$$q_2 = (a + a(b+aa)^* b)^* a(b+aa)^*$$

Putting the value of q_2 in eqn (3)

$$q_3 = q_2 a$$

$$q_3 = (a + a(b+aa)^* b)^* a (b+aa)^* a$$

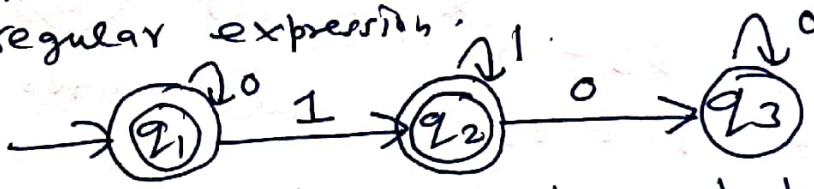
Since q_3 is the final state, so it shows the set of strings accept in regular expression form

$$(a + a(b+aa)^* b)^* a(b+aa)^* a$$

Ans

---

ex:2 Convert the given transition system (NFA) into regular expression.



Ans Since it does not have λ-transition, so we can apply arden's theorem.

$$q_1 = q_1 0 + \lambda \quad \text{—(1)}$$

$$q_2 = q_1 1 + q_2 1 \quad \text{—(2)}$$

$$q_3 = q_2 0 + q_3 (0+1) \quad \text{—(3)}$$

We can directly apply arden's theorem in this $R = Q + RP$ has solution $R = QP^*$

$$q_1 = q_1 0 + \lambda$$

$$R = RP + Q$$

$$q_1 = \lambda \cdot 0^* = 0^* \quad \text{—(4)}$$

So $q_1 = 0^*$

Putting value of q_1 in eq^n (2)

$q_2 = q_1 1 + q_2 1$

$\frac{q_2}{R} = \frac{0^* 1}{Q} + \frac{q_2 1}{RP}$ with solution $R = QP^*$

$q_2 = 0^* 1 1^*$ — (5)

Putting value of q_2 in eq^n (3)

$q_3 = q_2 0 + q_3(0+1)$

~~$q_3 = q_2 0$~~

$q_3 = 0^* 11^* 0 + q_3(0+1)$

$R = Q + RP$

$R = QP^*$

$q_3 = 0^* 11^* 0 (0+1)^*$ — (6)

Since q_1 & q_2 are final states, So solution is given by $q_1 + q_2$

$0^* + 0^* 11^*$

Identities for regular expression

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

Represent the following sets by regular expressions

(a) $\{0,1,2\}$

$0+1+2$

(b) $\{1^{2n+1} \mid n>0\}$

$= \{1^{2\times1+1}, 1^{2\times2+1}, 1^{2\times3+1}, 1^{2\times4+1}, 1^{2\times5+1} \text{---}\}$

$= \{111, 11111, 1111111, 111111111, 11111111111, \text{---}\}$

$= 111\{\Lambda, 11, 1111, 111111, 11111111, \text{---}\}$

$= 111(11)^*$

(c) $\{w \in \{a,b\}^* \mid w \text{ has only one } a\}$

$= b^*ab^*$

(d) The set of all strings over $[0,1]$ which has at most two zeros.

$1^* + 1^*01^* + 1^*01^*01^*$

(e) $\{a^2, a^5, a^8, \text{---}\}$

$= a^2\{\Lambda, a^3, a^6, \text{---}\}$

~~$aa(aaa)^*$~~

$= a^2(a^3)^*$

$= aa(aaa)^*$

(f) $\{a^n \mid n \text{ is divisible by } 2 \text{ or } 3 \text{ or } n=5\}$

$= (aa)^* + (aaa)^* + aaaaa$

(g) The set of all strings over $[a,b]$ beginning & ending with a.

$a(a+b)^*a$

Proof of Arden's theorem

Arden's theorem: "Let P and Q be two regular expressions over Σ. If P does not contain λ, then the following equation in R

$$R = Q + RP$$

has a unique solution given by $R = QP^*$.

Proof: Since $R = Q + RP$

$= Q + (Q+RP)P$ (Putting $R = Q+RP$)

$= Q + QP + RP^2$

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

$= Q + QP + QP^2 + RP^3$

$= Q + QP + QP^2 + QP^3 + QP^4 + \cdots$

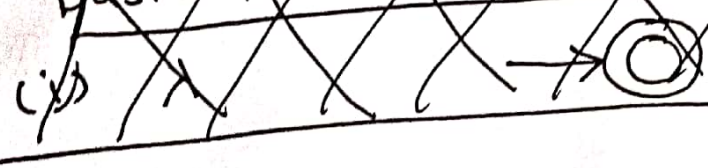
$= Q(\lambda + P + P^2 + P^3 + P^4 + \cdots)$

$R = QP^*$

Conversion from Finite Automata (FA) to Regular expression (RE)

Basic Rules for Conversion:

(i)



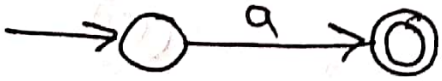
Conversion from Regular Expression (RE) to Finite Automata (FA)

Rules for Conversion:

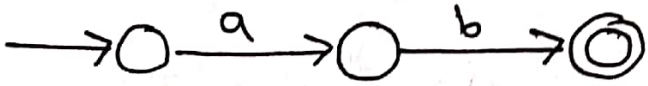
(i) $R = \lambda$

(ii) $R = \phi$

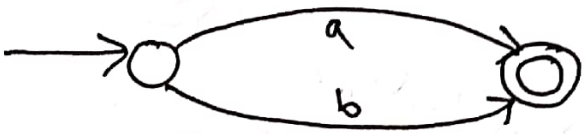
(iii) $R = a$



(iv) $R = ab$

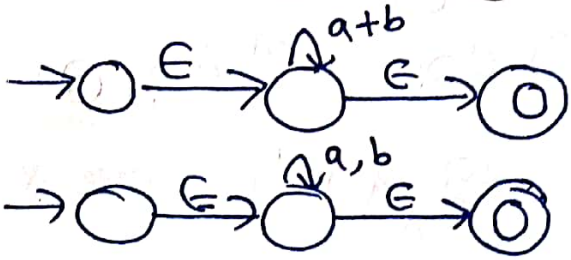


(v) $R = a+b$

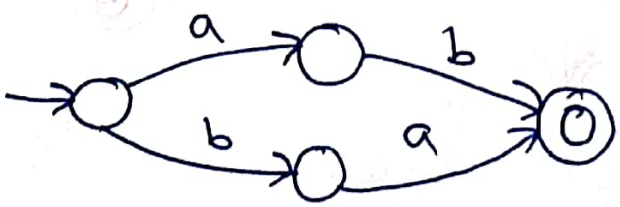


(vi) $R = a^*$

(VII) $R = (a+b)^*$



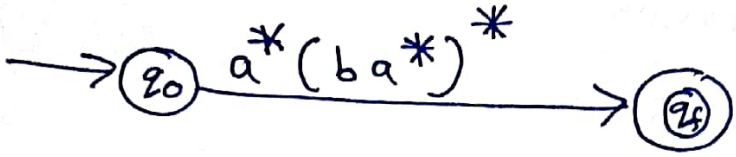
(VIII) $R = \underline{ab}+ba$

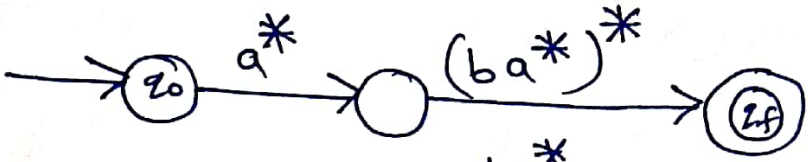


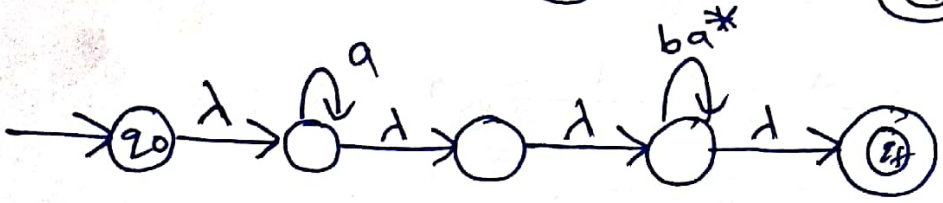
Kleene's theorem: If R is a regular expression then there exist an NDFA with λ-moves

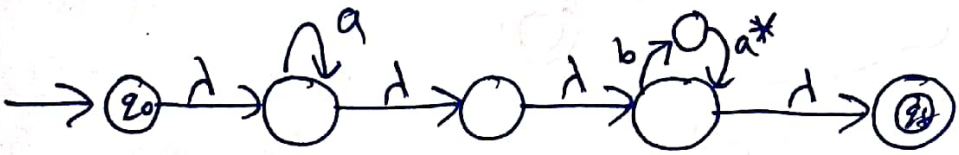
Example 1:

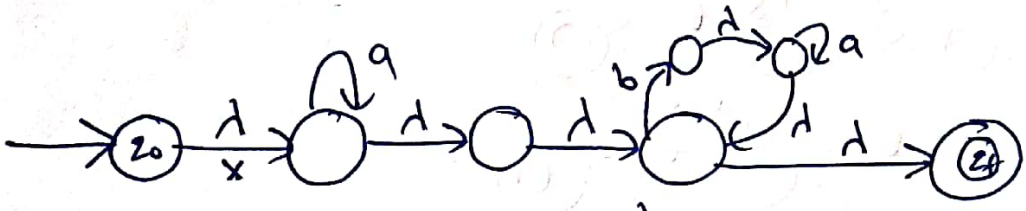
Convert $a^*(ba^*)^*$ into Finite Automaton (FA)

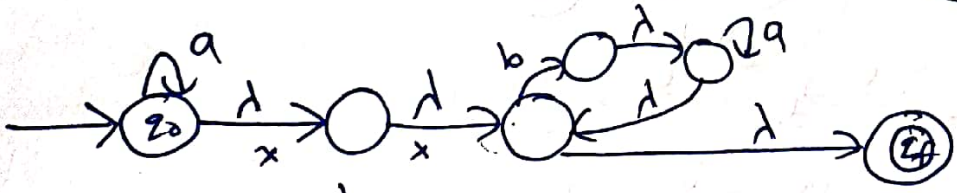


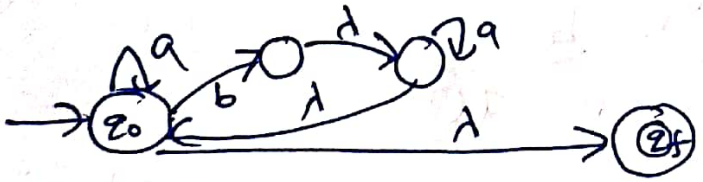


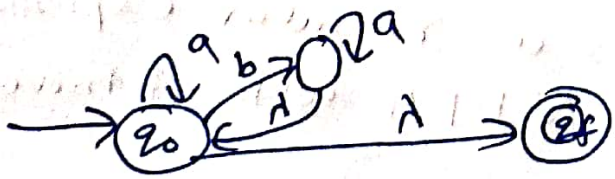


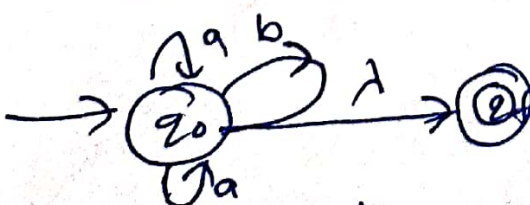


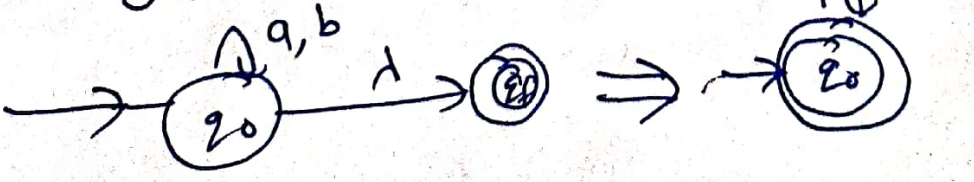








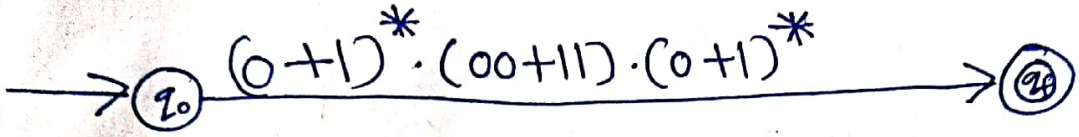


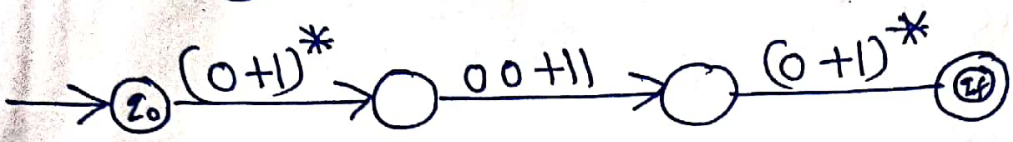


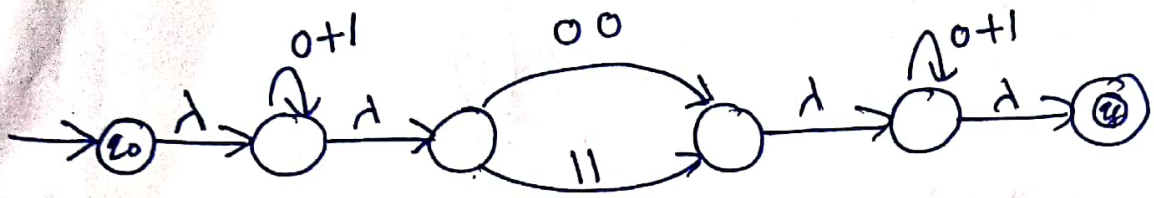
Example 2

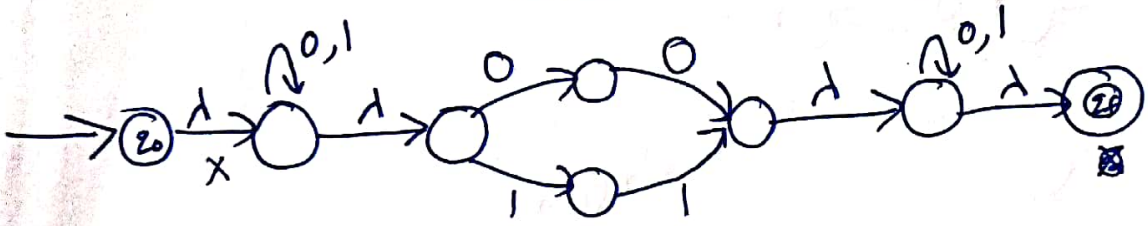
Construct the finite automaton equivalent to regular expression

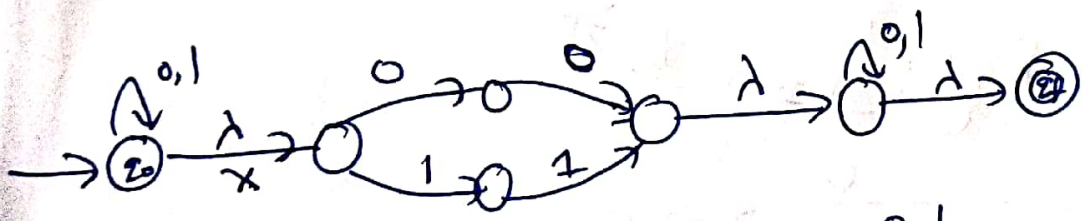
$(0+1)^*(00+11)(0+1)^*$

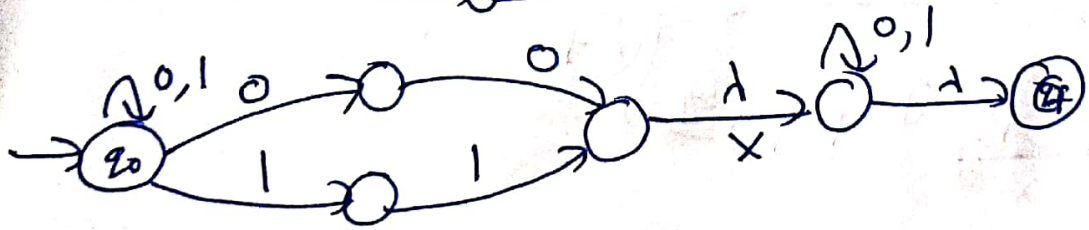


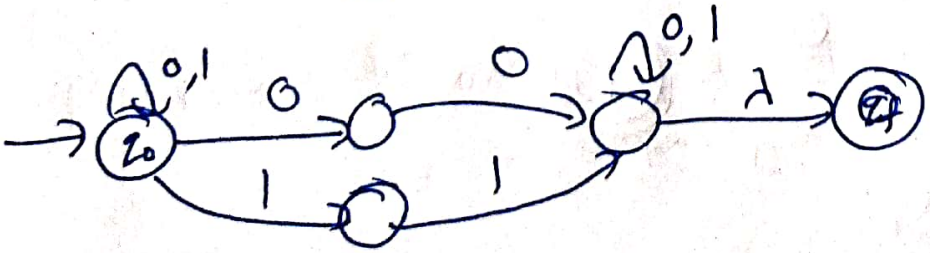


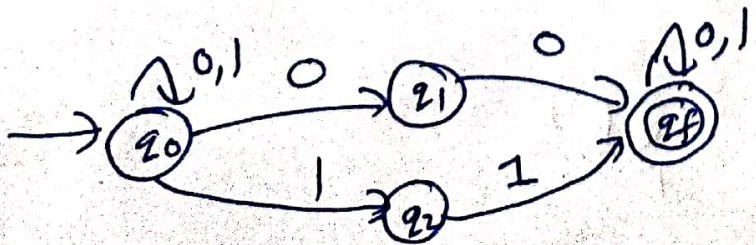




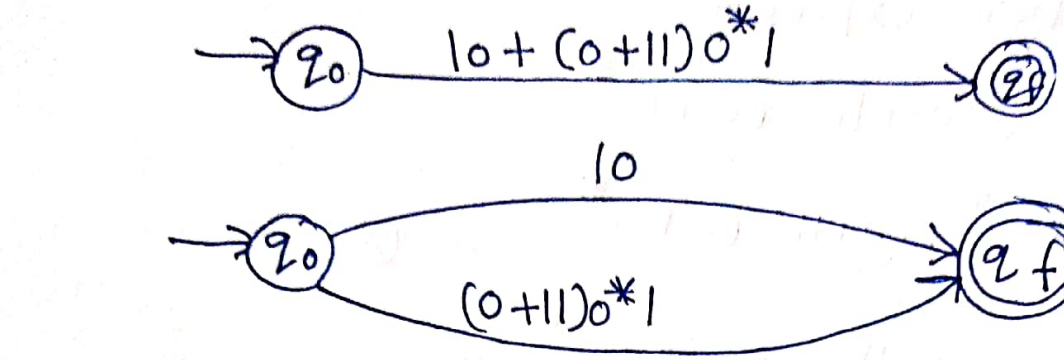


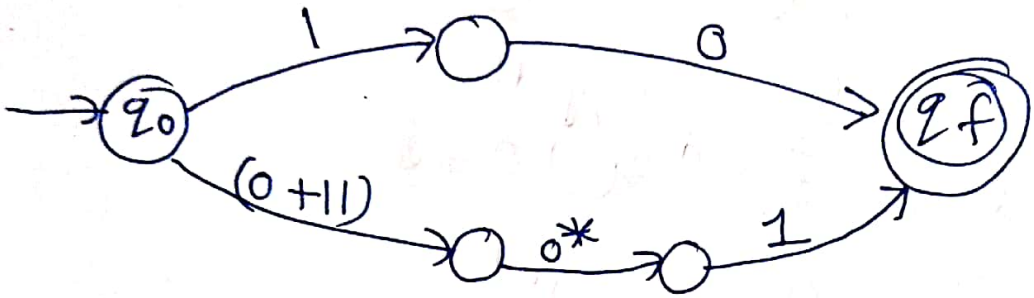


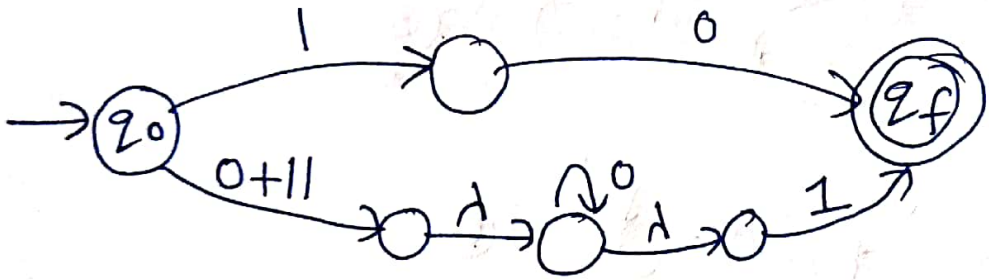


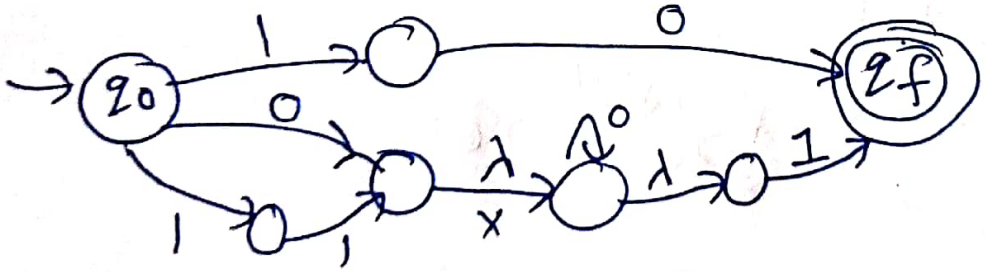


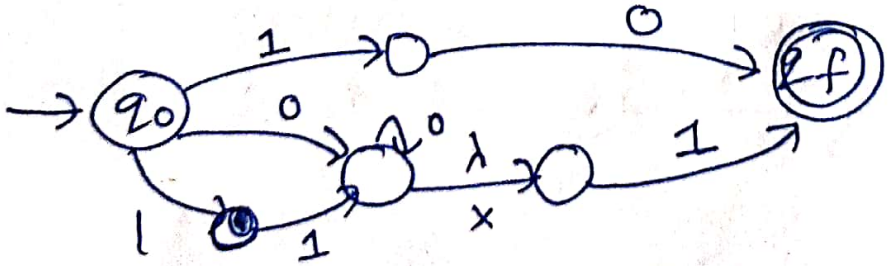
Ex:2 Construct DFA for the following regular expression $10+(0+11)0^*1$ (12)

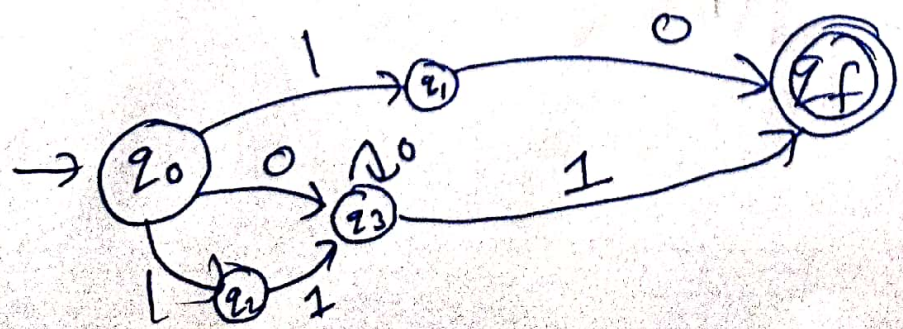










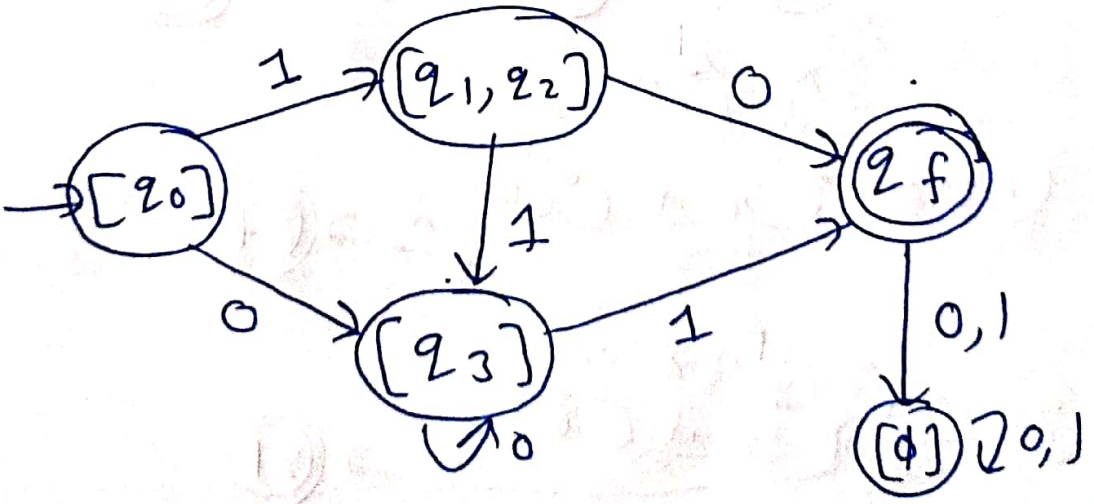


Transition Table for given NDFA

Q \ Σ	0	1
→ q_0	q_3	q_1, q_2
q_1	q_f	
q_2		q_3
q_3	q_3	q_f
(q_f)		

⇓ NDFA to DFA

Q \ Σ	0	1
→[q_0]	[q_3]	[q_1, q_2]
[q_3]	[q_3]	[q_f]
[q_1, q_2]	[q_f]	[q_3]
[q_f]	[ϕ]	[ϕ]
[ϕ]	[ϕ]	[ϕ]

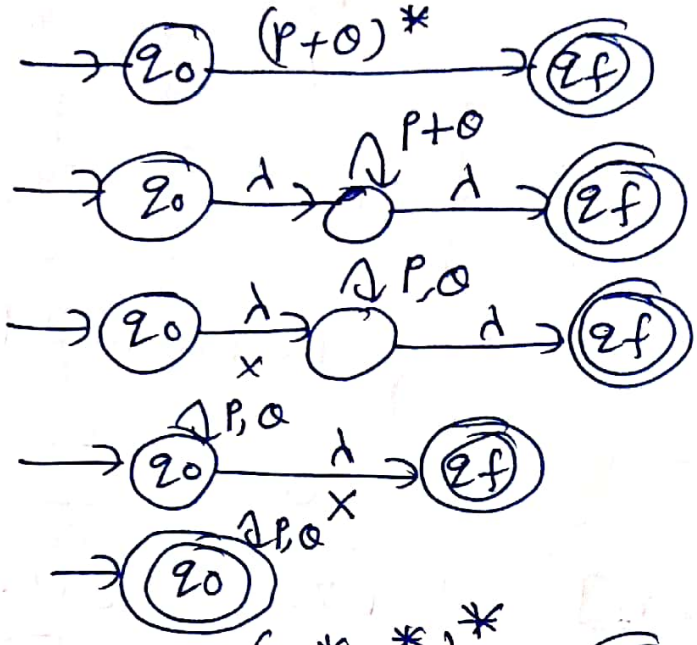


DFA

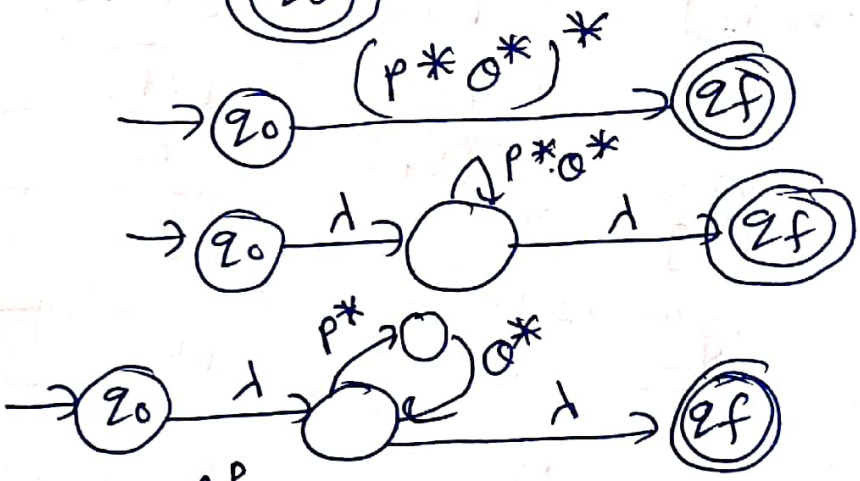
Prove that $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$

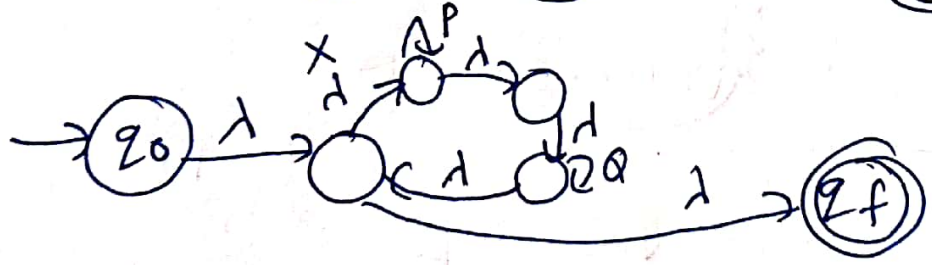
Ans Draw FA for all three Regular expressions

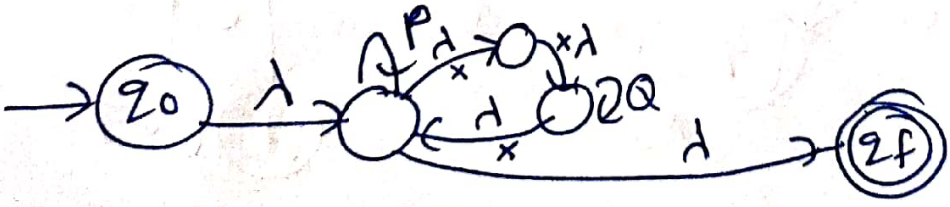
$(P+Q)^*$

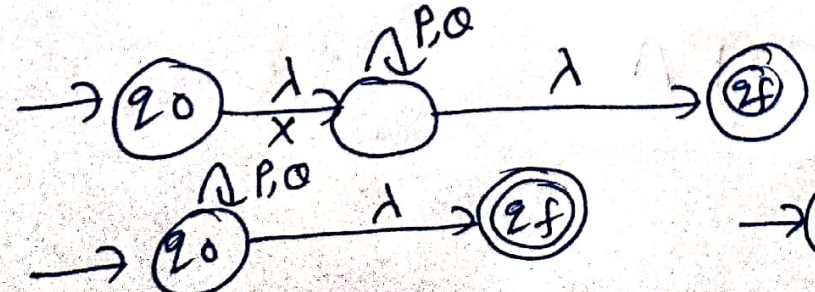


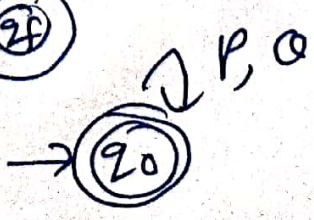
$(P^* Q^*)^*$



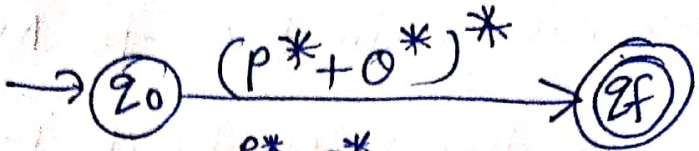


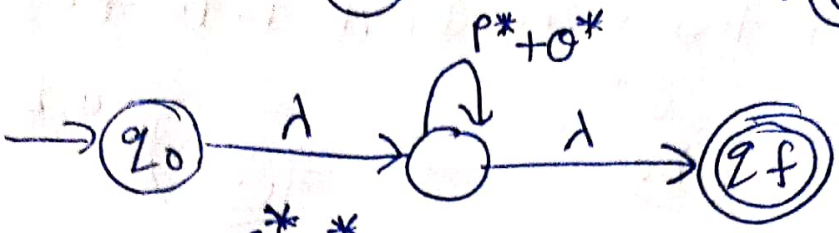


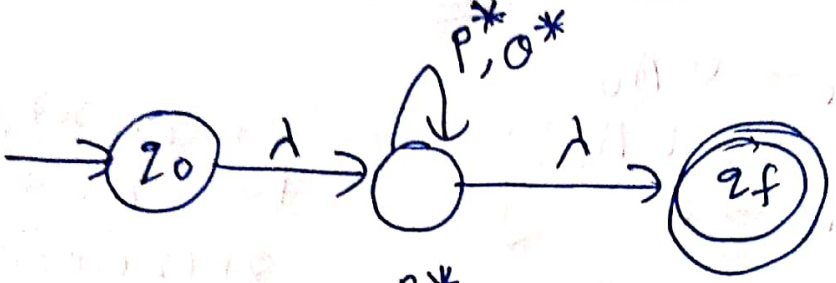


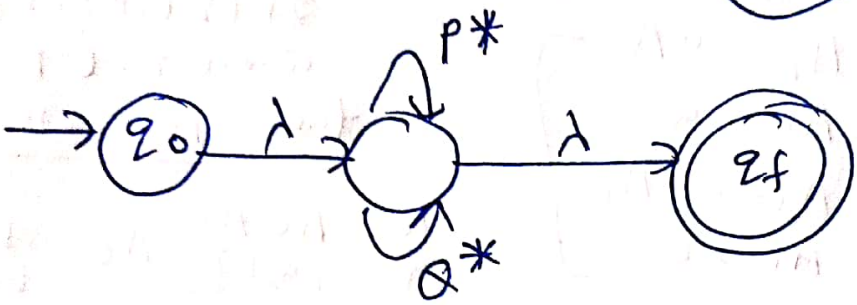


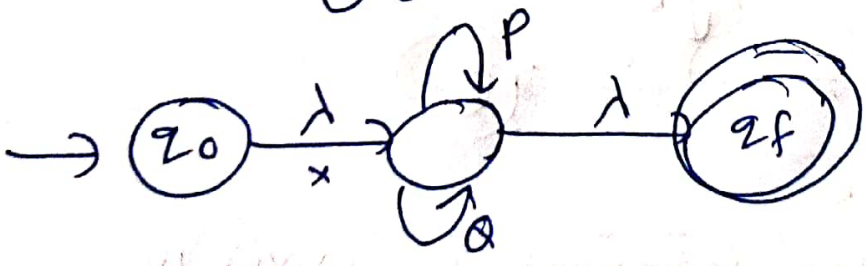
$(p^* + o^*)^*$

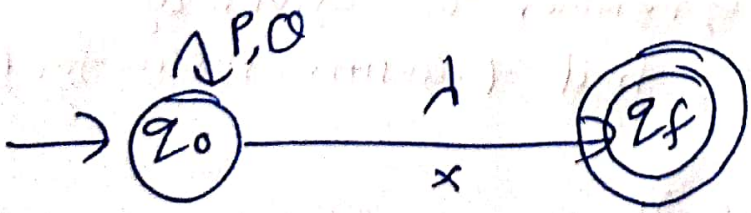


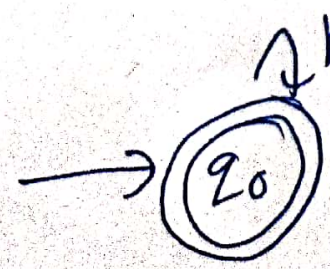






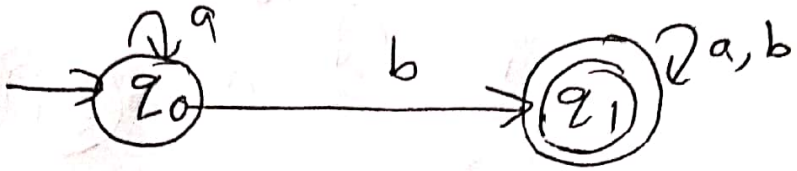






Conversion from FA to Regular Grammar

ex:1 Convert the given DFA into Regular Grammar



$$G = \begin{bmatrix} A_0 \to a A_0 \\ A_0 \to b A_1 \\ A_0 \to b \end{bmatrix} \quad \begin{bmatrix} A_1 \to a A_1 \\ A_1 \to a \\ A_1 \to b A_1 \\ A_1 \to b \end{bmatrix}$$

if $(q_i, x) \to q_j$ then $A_i \to x q_j$

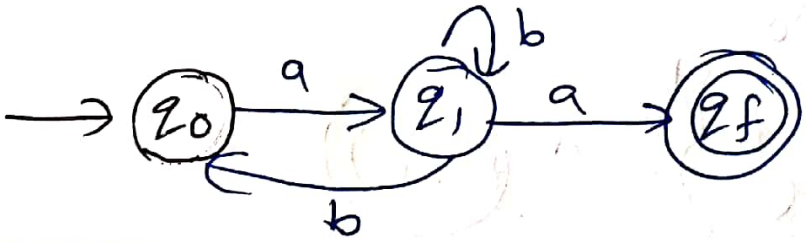
if $(q_i, x) \to q_j$ where $q_j \in F$ then $A_i \to x$

$N = A_0, A_1$, A_0 = start variable
$T = \{a, b\}$,
~~initial state~~

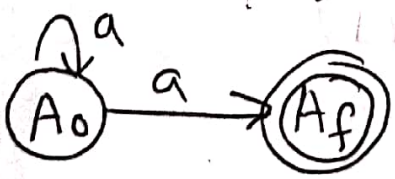
Conversion from Regular Grammar to FA

ex:2 Convert the following regular grammar to FA

$A_0 \to a A_1$, $A_1 \to b A_1$, $A_1 \to a$, $A_1 \to b A_0$



ex:3 If a regular grammar is given by $S \to aS \mid a$, find transition system for it.



Numericals on Regular expressions

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

Ans

L.H.S $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$

$\Rightarrow (1+00^*1)[\lambda + \underbrace{(0+10^*1)^*}_{R^*}\underbrace{(0+10^*1)}_{R}$

$\Rightarrow (1+00^*1)(0+10^*1)^*$ $\quad [\lambda + R^*R = R^*]$ from Ig

$\Rightarrow 1[\lambda + \frac{00^*}{RR^*}](0+10^*1)^*$ $\quad [\lambda + RR^* = R^*]$

$\Rightarrow 10^*(0+10^*1)^*$

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

---

② Prove that $P+PQ^*Q = a^*bQ^*$, where $P = b+aa^*b$ and Q is any regular expression.

Ans **L.H.S** $P+PQ^*Q$

$\Rightarrow P(\lambda + Q^*Q)$ $\quad [\lambda + R^*R = R^*]$ from Ig

$\Rightarrow P.Q^*$

$\Rightarrow (b+aa^*b)Q^*$

$\Rightarrow b(\lambda + aa^*)Q^*$ $\quad [\lambda + aa^* = a^*]$ from Ig

$\Rightarrow b.a^*Q^*$

$\Rightarrow \underline{a^*bQ^*}$

Pumping Lemma for Regular Languages (18)

(1) Pumping Lemma for regular languages are used to prove that certain languages are not regular.

(2) In this, from a given string, we create new strings by pumping (generating) a specific part of a string.

(3) Lemma are parts of a theorem, that helps in the proof of a theorem. They are the steping stones in the proof of a theorem.

(4) "If L is a regular language and w is a string of L. If w is divided into 3 parts x, y and z, then it must satisfy the following conditions

① $xy^iz \in L$ for $i \geq 0$

② $|y| > 0$

③ $|xy| \leq p$ (where p is the pumping length)

example: Prove that $\{L = a^n b^n \mid n \geq 1\}$ is not regular

Ans: Assume $L = a^n b^n$ is regular and p = pumping length

then $w = a^p b^p$, $L = a^n b^n = \{ab, aabb, aaabbb, aaaabbbb, \ldots\}$

Let $w = a^3 b^3$ $(p = 3)$

$w = aaabbb$

divide into 3 parts such that $\begin{cases} |y| > 0 \\ |xy| \leq 3 \text{ since } p = 3 \text{ is pumping length} \end{cases}$

Case 1: $w = \underbrace{a}_{x}\underbrace{aa}_{y}\underbrace{bbb}_{z}$ $\begin{cases} |y| > 0 \\ |xy| \leq 3 \end{cases}$

$\begin{cases} x = a \\ y = aa \\ z = bbb \end{cases}$

$xy^iz \in L$ for $i \geq 0$

for $i = 0$, $xy^0z = a(aa)^0bbb = a \cdot 1 \cdot bbb = abbb \notin L$

for $i = 1$, $xy^1z = a(aa)bbb = aaabbb \in L$

for $i = 2$, $xy^2z = a(aa)^2bbb = aaaaabbb \notin L$

If rules violates for any value of i, then we can say that Language is not regular

Case 2: $\omega = \underbrace{aa}_{x}\underbrace{a}_{y}\underbrace{bbb}_{z}$ $\left[\begin{matrix}|y|>0 \\ |xy|\leq 3\end{matrix}\right]$

$\{x=aa,\ y=a,\ z=bbb\}$

$xy^iz \in L$ for $i \geq 0$

for i=0 $xy^0z \in L$

$= (aa)(a)^0(bbb) = aa.1.bbb = aabbb \notin L$

for i=1 xy^1z

$(aa)(a)bbb = aaabbb \in L$

for i=2 xy^2z

$(aa)(a)^2bbb = aaaabbb \notin L$

for i=3 xy^3z

$(aa)(a)^3bbb = aaaaabbb \notin L$

for i=4 xy^4z

$(aa)(a)^4bbb = aaaaaabbb \in L$

only for $i=1$, it belongs to regular language.

Since $\omega = a^p b^p$ for $p=4$

$\omega = a^4b^4$

$\omega = a^4b^4$

$\omega = \underbrace{aa}_{x}\underbrace{aa}_{y}\underbrace{bbbb}_{z}$ $\left[\begin{matrix}|y|>0 \\ |xy|\leq 3\end{matrix}\right]$ $\{x=a,\ y=aa,\ z=abbbb\}$

Case 1:

$xy^iz \in L$ for $i \geq 0$

for i=0 $xy^0z = a(aa)^0abbbb = aabbbb \notin L$

for i=1 $xy^1z = a(aa)^1abbbb = aaaabbbb \in L$

for i=2 $xy^2z = a(aa)^2abbbb$

$= aaaaaabbbb \notin L$

∴ Since some of the strings does not belongs to language L, so $\{L = a^nb^n \mid n>1\}$ is not regular.

Ex:2 Show that $\{L = a^p \mid p$ is a prime No. $\}$ is not regular

Ans $L = a^p \mid p$ is a prime No.

$= \{aa, aaa, aaaaa, aaaaaaa, \ldots\}$

since $p = 2, 3, 5, 7, 11, 13 \ldots$ are prime No.

Using pumping Lemma

1. $xy^iz \in L$ for $i \geq 0$
2. $|y| > 0$
3. $|xy| \leq q$ (where q is the pumping length)

Case 1:

Let $\omega = a^q$ $(q = 3)$

$\omega = a^3$

$\omega = \underset{x}{a}\,\underset{y}{a}\,\underset{z}{a}$ $\left\{\begin{matrix} |y| > 0 \\ |xy| \leq q \\ 2 \leq 3 \end{matrix}\right\}$

$\begin{bmatrix} x=a \\ y=a \\ z=a \end{bmatrix}$ ↓ divide into 3 parts

for $i=0$, $xy^iz = xy^0z = a \cdot 1 \cdot a = aa \in L$

for $i=1$, $xy^iz = xy^1z = a(a)^1a = aaa \in L$

for $i=2$, $xy^iz = xy^2z = a(a)^2a = aaaa \notin L$

for $i=3$, $xy^iz = xy^3z = a(a)^3a = aaaaa \in L$

for $i=4$, $xy^iz = xy^4z = a(a)^4a = aaaaaa \notin L$

Since some of the value are not belonging to Language, so for Case 1, it is not regular

Case 2

$\omega = \underset{x}{aa}\,\underset{y}{a}\,\underset{z}{a} = a^3 = a^q$

Let $x = aa$, $y = a$, $z = \epsilon$ $\left\{\begin{matrix} |y| > 0 \\ |xy| \leq q \end{matrix}\right.$

Case 2: $L = xy^iz$

$i=0$, $xy^0z = aa(a)^0 \in$
$= aa.1 \in = aa \in L$

$i=1$, $xy^1z = aa(a) \in$
$= aaa \in = aaa \in L$

$i=2$ $xy^2z = aa(a)^2 \in$
$= aaaa \notin L$

$i=3$ $xy^3z = aa(a)^3 \in$
$= aaaaa \in L$

for, $i=2$, string does not belong to Language.

Since we got some strings that does not belong to language L, so L is not regular
i.e. $\{L = a^b \mid b$ is a prime No.$\}$ is not regular

Question for Practice

(a) $L = a^n b^{2n} \mid n>0$

(b) $L = \{a^n b^m \mid 0<n<m\}$

(c) $L = \{a^{2n} \mid n \geq 1\}$

Using pumping Lemma, prove that following languages are not regular.

More examples on Regular grammar to DFA

ex: Construct a DFA equivalent to the grammar

$S \to aS \mid bS \mid aA$

$A \to bB$

$B \to aC$

$C \to \lambda$

Sol: First remove λ by placing the value of $C \to \lambda$ in $B \to aC$,

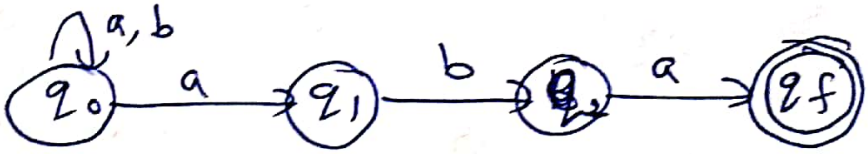
the modified grammar is

$$G = \begin{cases} S \to aS \\ S \to bS \\ S \to aA \\ A \to bB \\ B \to a \end{cases}$$

(putting $C \to \lambda$ in $B \to aC$)

So transition system (NFA) is

Let q_0, q_1, q_2 are states corresponding to S, A, B non-terminals



NFA table =

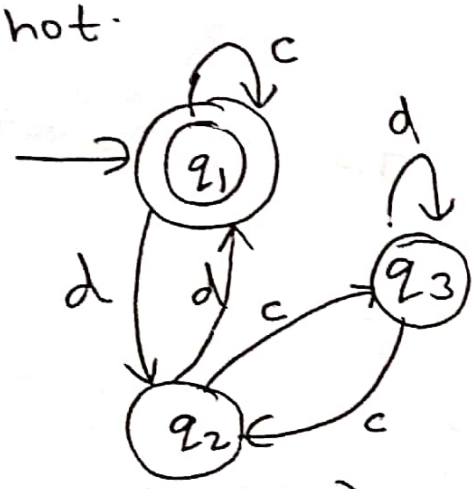
Q \ Σ	a	b
→ q_0	q_0, q_1	q_0
q_1	-	q_2
q_2	q_f	-
(q_f)	-	-

⇓ NDFA to DFA

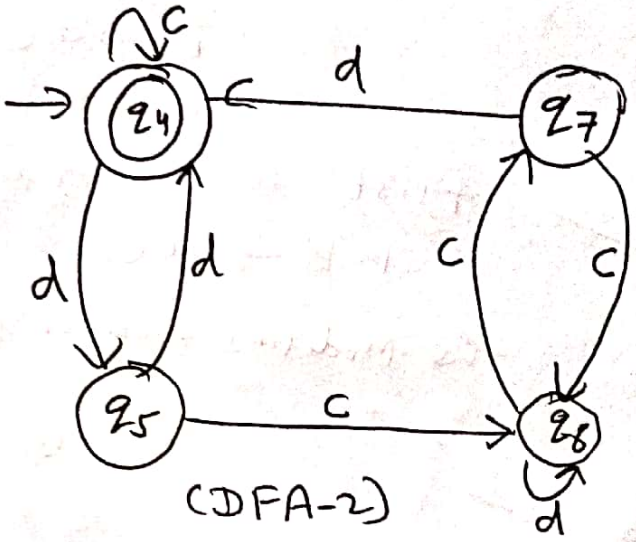
Q \ Σ	a	b
→ $[q_0]$	$[q_0, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_2]$
$[q_0, q_2]$	$[q_0, q_1, q_f]$	$[q_0]$
$[q_0, q_1, q_f]$	$[q_0, q_1]$	$[q_0, q_2]$

Equivalence of two Finite Automata using Comparison Method

Ex: Consider the following two DFA shown in the given figure. Using Comparison method, check whether the two DFA are equivalent or not.



(DFA-1)



(DFA-2)

Ans Comparison Table Use starting state of each DFA to initiate table.

States \ Inputs	c	d
(q_1, q_4)	(q_1, q_4)	(q_2, q_5)
(q_2, q_5)	(q_3, q_6)	(q_1, q_4)
(q_3, q_6)	(q_2, q_7)	(q_3, q_6)
(q_2, q_7)	(q_3, q_6)	(q_1, q_4)

As we did not get a single pair of states, in which for a given input (cord), one next state is final & other is non-final (or vice versa) at every row.

So both DFA are equivalent

Right Linear Grammar

A grammar is said to right linear grammar, if the productions are of the form

$$A \to xB$$
$$A \to x$$

Where $A, B \to$ Non terminal
$x \to$ terminal including null string

Left Linear Grammar:

A grammar is said to be left linear Grammar, if the productions are of the form

$$A \to Bx$$
$$A \to x$$

Where $A, B \to$ Non-terminal
$x \to$ terminal including null string.

Regular Grammar: A regular grammar is either right linear grammar or left linear grammar

Closure properties of regular languages

① If L_1 and L_2 are regular languages, then $L_1 \cup L_2$, $L_1 \cdot L_2$ and L_1^* are also regular languages.

Proof

If L_1 and L_2 are regular languages, then there exist regular expression r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$

As by definition we know that

(i) $r_1 + r_2$ is regular expression, so correspondingly $L_1 \cup L_2$ is also regular languages

(ii) $r_1 \cdot r_2$ is regular expression, so correspondingly $L_1 \cdot L_2$ is also regular regular languages.

(iii) r_1^* is regular expression, so correspondingly L_1^* is also regular language.

② If L is a regular language, then its complement $\bar{L}$ is also regular language.

Proof Since L is a regular language, so there exist a DFA for it.

Let 

Lets Name it M

M = DFA for regular language L

Now if we Complement M, then all Non-final states will become final states and all final states will be non-final states

$\overline{M}$ = 

Still a DFA it is

Since $\bar{M}$ is still DFA, So its Language $\bar{L}$ is also regular language.

So we can say that if L is a regular language, then its Complement($\bar{L}$) is also regular language.

③ If L_1 and L_2 are two regular languages, then $L_1 \cap L_2$ is also regular.

Proof Using De-Morgan's law, we know that,

$$L_1 \cap L_2 = \overline{(\bar{L_1} + \bar{L_2})} \quad \text{—①}$$

from R.H.S (since $A \cap B = \overline{(\bar{A} \cup \bar{B})} = \overline{(\bar{A} + \bar{B})}$)

Since L_1 is regular language, So its Complement $\bar{L_1}$ is also regular language by

Since L_2 is regular language, So its Complement $\bar{L_2}$ is also regular language

Since $\bar{L_1}$ and $\bar{L_2}$ are regular languages, So $\bar{L_1} + \bar{L_2}$ is also regular language.

as we know that the Complement of a regular language is also regular language, So $\overline{(\bar{L_1} + \bar{L_2})}$ is also regular

from eqn ① R.H.S = L.H.S

(i.e. $L_1 \cap L_2$ is also regular.

④ If L_1 and L_2 are regular languages then $L_1 - L_2$ is also regular.

Proof! from set theory, we know that

$A - B = A \cap \bar{B}$ So $L_1 - L_2 = L_1 \cap \bar{L_2}$

We also know that If L_2 is regular, then $\bar{L_2}$ is also regular.

since L_1 and $\bar{L_2}$ are both regular, So their Intersection $L_1 \cap \bar{L_2}$ is also regular language.

Homomorphism and inverse homomorphism in regular language!

Homomorphism(h): A homomorphism is a substitution in which a single letter is replaced with a string.

Let Σ and τ are two alphabets

$$h: \Sigma \longrightarrow \tau^*$$

Let $\Sigma = \{0,1\}$, $\tau = \{a,b\}$

$h(0) = aa$, $h(1) = bb$

If L is a regular language

$L = \{00, 101\}$

then $h(L) = \{h(00), h(101)\}$

$= \{aaaa, bbaabb\}$

If r is a regular expression

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

$h(r) = (aa+bb)^* (bb)^*$

Inverse homomorphism (h^{-1})

The inverse process of homomorphism is called inverse homomorphism. It is represented by h^{-1}.

Let $\Sigma = \{0,1,2\}$, $\tau = \{a,b\}$

$h(0) = a$, $h(1) = b$, $h(2) = ab$

Given regular language

$L = \{abab\}$

Case-1

$h^{-1}(L) = \{22\}$

Case-2

$L = \{abab\}$

$h^{-1}(L) = \{0101\}$

Case-3

$L = \{abab\}$

$h^{-1}(L) = \{201\}$

Case-4

$L = \{abab\}$

$h^{-1}(L) = \{012\}$

So $h^{-1}(L) = \{22, 0101, 201, 012\}$

Let $\Sigma = \{0,1\}$, $\tau = \{a,b\}$

$h(0) = aa$, $h(1) = bb$

and $h(r) = (aa+bb)^*(bb)^*$

then $h^{-1}(h(r)) = (0+1)^*1^*$

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

Describe the decision properties of regular languages:

There are three decision properties of regular languages:

1. Membership property
2. Emptiness property
3. Equivalence property
4. Finiteness Property

1. Membership property:

This property specify that whether a particular string w belongs to some language (Regular language)

Proof Let L is a regular language.
i.e. there exist a Finite Automata (DFA|NFA) for it.
If there exist a string w then if w reaches to an accept state (final state), then we can say that string w belongs to regular language.

2. Emptiness property:

Proof The emptiness property specify that whether a regular language is empty or not.

If L is a regular language then there exist a FA(DFA|NDFA) for it.
If there exist some path from start state to final state in FA, then the regular language is not empty.
However, if there is no path reaching to a final state from the start state, then the language is empty.

③ Equivalence property:

Equivalence property specify that whether two regular language are equivalent.

Proof Since we know that any regular language can be specified by FA(DFA | NDFA).
So If there are two regular language L_1 and L_2, and they can be represented by DFA_1 and DFA_2. Then by using Comparison method, we can say that whether DFA_1 and DFA_2 are equivalent or not. If $DFA_1 = DFA_2$ then, there regular language are also equivalent.

④ Finiteness Property:

Finiteness property specify that whether the strings generated by regular languages are finite or not.

Proof As we know that any regular language can by represented using DFA | NDFA.

(i) So for the DFA drawn from the regular language, check the states that can not be reached from the initial states & delete them. This is called Elimination of unreachable states.

(ii) Check the states from which we can not reach the final states & delete them. This is called elimination of dead states.

(iii) If the resulting DFA contains loops then the finite automaton accept infinite strings of a regular language

(iv) If the resulting DFA does not contain loops, then the finite Automaton accept finite strings of a language.

Some other closure properties of regular languages

If L is a regular language and h is a homomorphism on its alphabet, then h(L) is also a regular language.

Proof If L is a regular language then there exist a regular expression R for it. When we will apply homomorphism h on each symbol of regular expression R, then still we got another regular expression after homomorphism. i.e. the h(L) or homomorphism of a regular language is also a regular language.

If h(L) is a homomorphism and L is a regular language, ~~then~~ on its alphabet, then $h^{-1}(L)$ is also a regular language.

Proof If L is a regular language and h(L) is also regular language due to closure property, i.e. for h(L), there exist a ~~regular~~ regular expression h(r). If we take the inverse of regular expression $(h^{-1}(h(r))$, we still got regular expression r. So $h^{-1}(L)$ is also a regular language.