Lecture notes on Automata

(Used at Vidya College of Engineering, Meerut)

Instructor: Rajendra Kumar, Assistant Prof. CSE Department

kumar_rajen@yahoo.com,

rajendra04@gmail.com

Lecture-1

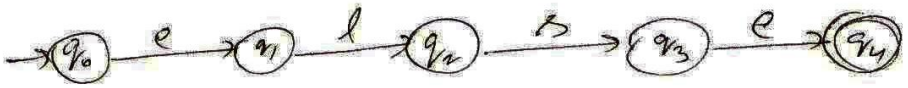
Need and scope of automata

- The revolution of automatic system is based on automata.
- The main application of automata is in compiler design.
- Pattern recognition is another important application of automata
- Every automatic system is expressed in terms of various stages (in automata we say states)
- The percentage amount of a process depends upon the state in which automaton currently is.

Automata in Compiler

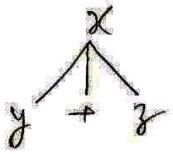
- In this subject we study four types of automata:
  (i) FA (ii) PDA (iii) LBA (iv) TM

- Finite automata ⇒ Lexical Analyser
  It is responsible for checking the validity of a string.



- Pushdown automata ⇒ Syntax Analyser
  It is responsible for checking the validity of a statement using derivation tree.

$x = y + z$



Lecture 2

Formal language – collection of strings or words

Words – collection of characters. The characters in terms of compiler are called tokens.

Generation of strings from tokens:

closure
- kleene closure
- positive closure

kleene closure-

A^* = closure of A

= set of all strings composed of $A's$ (zero occurance to ∞)

$= (A^0 \cup A^1 \cup A^2 \cup \dots A^\infty)$

$= (\wedge, A, A^2, A^3, \dots)$

$= \bigcup_{i=0}^{i=\infty} A^i$

A^+ = positive closure of A

= set of all strings composed of $A's$ (one occurrance to ∞)

$= (A^1 \cup A^2 \cup A^2 \cup \dots \cup A^\infty)$

$= (A, A^2, A^3, \dots, A^\infty)$

$= \bigcup_{i=1}^{i=\infty} A^i$

Lecture 3

Relation Between A^* & A^+

$$A^* = A^+ + \{\wedge\}$$

The closure operation is applicable to a character, a string and also a set.

$$a^* = (\wedge, a, a^2, a^3, \ldots)$$

$$(ab)^* = (\wedge, ab, (ab)^2, (ab)^3, \ldots)$$

$$(a+b)^* = (a \cup b)^* = (a,b)^*$$

$$= (\wedge, a, b, aa, ab, ba, bb, \ldots\ldots)$$

Important relations

1. $a^* = (a^*)^*$
2. $(a^+)^+ = a^+$
3. $(a^+)^* = a^*$
4. $(a^*)^+ = a^*$
5. $(\wedge + a)^+ = (\wedge, a, a^2, a^3, \ldots\ldots)$

$$= a^*$$

Example 1

find A^* and A^+ for $A = (a, ab, b, ba)$

Sol.

$A^* = (a, ab, b, ba)^*$

$= (\wedge, a, b, aa, ab, ba, bb, aaa, \ldots\ bbb, \ldots)$

$= (a, b)^*$

Lexicographic order

Generally, when we find out kleene closure and positive closure of some string or set, we write generated strings in increasing order of length called lexicographic order.

Example 2

find formal language of set of all strings over (a, b) starting with a or b

sol.

$(a(\wedge, a, b, aa, ab, ba, bb, \ldots\ldots))\cup(b(\wedge, a, b, aa, ab, ba, bb, \ldots\ldots))$

$= (a, aa, ab, aaa, aab, aba, abb, \ldots\ldots)\cup(b, ba, bb, baa, bab, bba, bbb, \ldots\ldots)$

$= (a, b)^*(a, b)$

$= (a, b)\ (a, b)^*$

$= (a, b)^+$

Lecture 4

operations on languages.

1. concatenation

   con (a, b) = ab ≠ ba

   con (ab, ∧, b) = abb

2. Length |aba| = 3

3. Union A ∪ B = (A, B)

4. Palindrome s = rev(s)

5. Reverse operation Rev (abb) = bba

6. Intersection operation

   Let A = (a, b, c)

   B = (b, c, d)

   A ∩ B = (b, c)

Example 1

find language over (0,1) of length at most 5.

Sol.

$$L = (\wedge, 0, 1, 00, 01, 10, 11, 000, \cdots, 111, 0000 \cdots 1111, 00000 \cdots 11111)$$

$$= \{s \in (0,1)^* \mid |s| \leq 5\}$$

Example 2

find language over (0,1) of length atleast 2.

Sol.

$$L = (00, 01, 10, 11, 000, \cdots 111, 0000, \cdots 1111, 0000 \cdots)$$

$$= \{s \in (0,1)^* \mid |s| \geq 2\}$$

$$= (00, 01, 10, 11)(\wedge, 0, 1, 00, 01, 10, 11, 000, \cdots)$$

$$= (00, 01, 10, 11)(0,1)^*$$

$$= (0,1)^*(00, 01, 10, 11)$$

Example 3

find $L = L_1 \cup L_2$ if $L_1 = a^*$, $L_2 = b^*$

$$L = a^* \cup b^*$$

$$= (\wedge, a, a^2, a^3, \cdots) \cup (\wedge, b, b^2, b^3, \cdots)$$

$$= (\wedge, a, b, aa, bb, aaa, bbb, \cdots)$$

$$= (\wedge, a, b, a^2, b^2, a^3, b^3, \cdots)$$

Lecture 5

Language of arithmatic expression (AE)

The arithmatic expressions can be expressed using recursive definition by formal language.

Def.

1. Any number or variable is an AE.
2. If A is any AE then
   (a) (A) is also AE
   (b) −A is also AE
3. If A and B are two different AEs then
   (a) A+B is also AE
   (b) A−B is also AE
   (c) A*B is also AE
   (d) A/B is also AE

By using above recursive definition we can find out the validity of an AE.

Example 1.

Check whether following AE is valid or not?

$$A = ((x+y)*3-2/y)-3$$

Sol. x and y are two diff. AEs. x+y is AE by 3(a)

(x+y) is AE by 2(a)

(x+y)*3 is AE by 3(c)

z and y are valid AEs.

z/y is AE by 3(d)

(x+y) * 3 − z/y is AE by 3(b)

((x+y) * 3 − z/y) is AE by 2(a)

as 3 is also valid AE therefore by rule 3(b)

((x+y) * 3 − z/y) − 3 is also valid AE.

Example 2

Check whether (a−b)+c) is valid or not?

Sol.

a, b are valid AE

a−b is AE by 3(b)

(a−b) is AE by 2(a)

(a−b)+c is AE by 3(a)

But (a−b)+c) has additional right bracket therefore it is not a valid AE.

Lect. 6

Finite automata (FA)

also - finite state automata
- finite state machine

FA $M = (Q, \Sigma, \delta, q_0, F)$

Q = finite non empty set of states

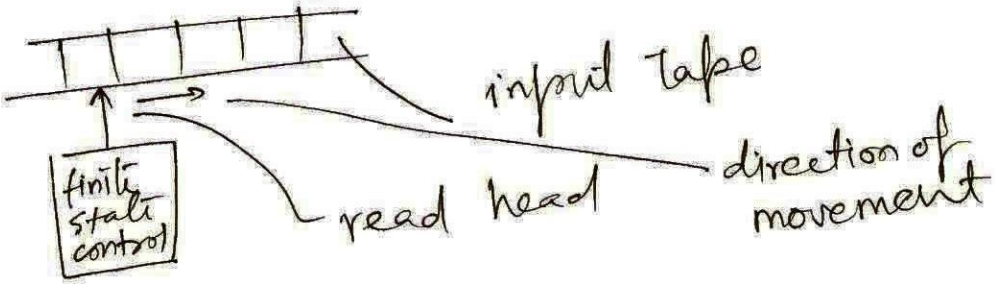
Σ = set of inputs

δ = transition function

$Q \times \Sigma \rightarrow Q$

q_0 = initial state, $q_0 \in Q$

F = set of final state, $F \subseteq Q$



Application

implementation of lexical analysis

Ex. 1

$M = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2\}$

$\Sigma = \{a, b\}$

δ is defined as

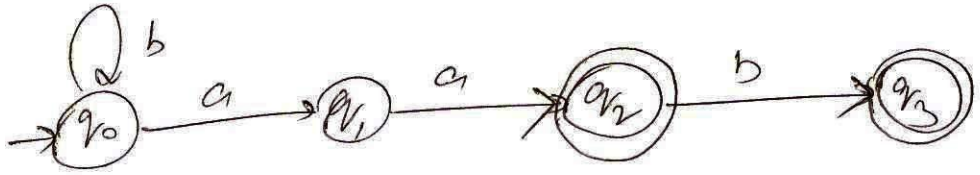
$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_0$

$\delta(q_1, a) = q_2,$

$\delta(q_2, b) = q_3$

$F = \{q_2, q_3\}$

The FA in diagram form is



the language of this FA is

$L = \boxed{b^n aaa}$

$L = \{b^n aaa \mid n \geq 0\}$

$= b^* aaa$

= set of all strings over $\{a, b\}$ such that

Lect. 7

transition diagram

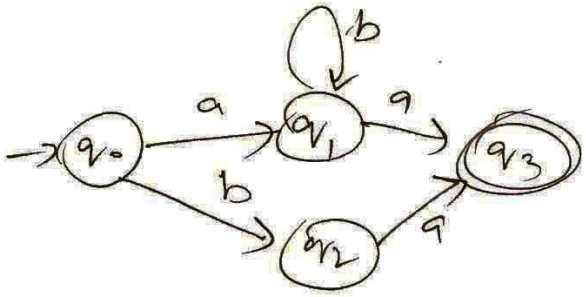
initial state →○

intermediate state ○

final state ◎

transition ⟶

Ex



Pr. State	Next state	
	a	b
$\rightarrow q_0$	q_1	q_2
q_1	q_3	q_1
q_2	q_3	—
(q_3)	—	—

Extended transition fun.

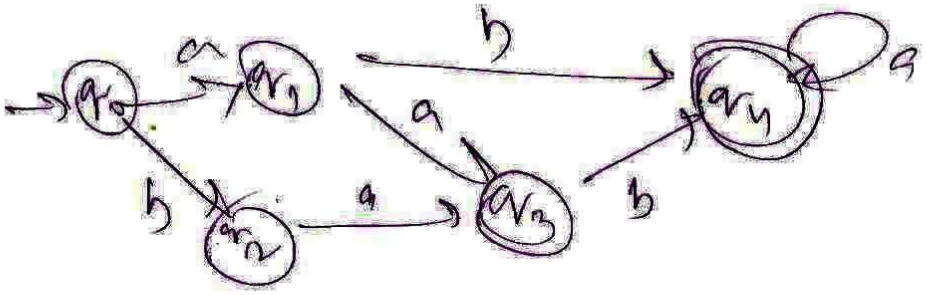
when input is a string on place of particular symbol.

$\delta^* : Q \times \Sigma^* \rightarrow Q$

$\delta^*(q_i, \wedge) = q_i$

Transition table

Ex Find transition table of following FA.



P. state	Next state a	Next state b
→ q_0	q_1	q_2
q_1	q_3	q_4
q_2	q_3	—
q_3	—	q_4
(q_4)	—	—

the language of this FA is

$$L = aba^* + aaba^* + baba^*$$

$$= (a + aa + ba)ba^*$$

Lect. 8

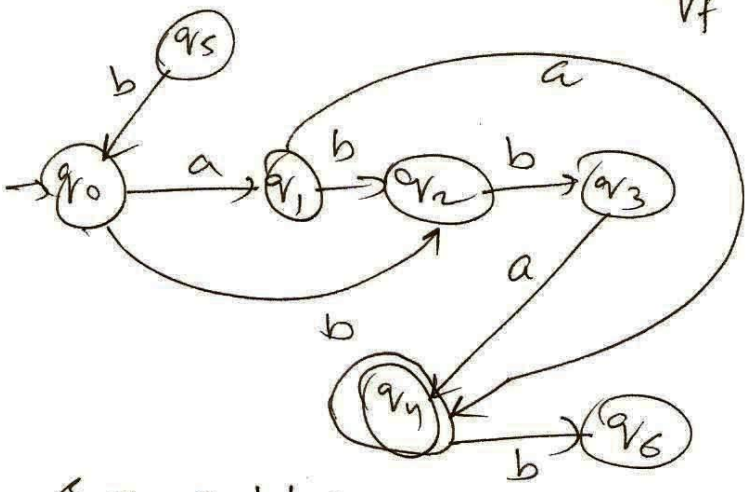
Acceptability by FA.

(i) whole input must be consumed

(ii) FA must reach in final state

$\delta^*(q_0, s) = (q_f, \Lambda)$ q_0 is initial state

$q_f \in F$



$s = a\ b\ b\ a$

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_3 \xrightarrow{a} q_4$

Non reachable states → q_5 in above diagram

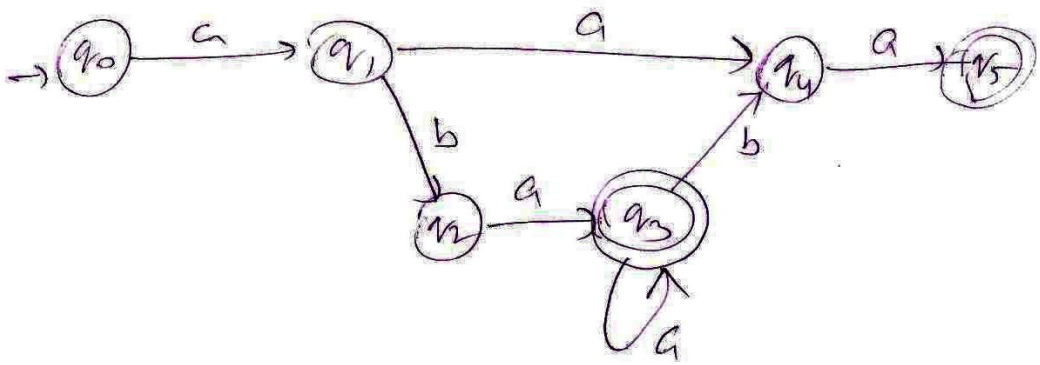
Dead State → q_6 in above diagram.

Ex find whether following strings are accepted by FA given below are accepted or not.

(i) ababa

(ii) babb

(iii) abaaa...aca



The strings (i) ababa and (iii) abaaa..a are acceptable as following

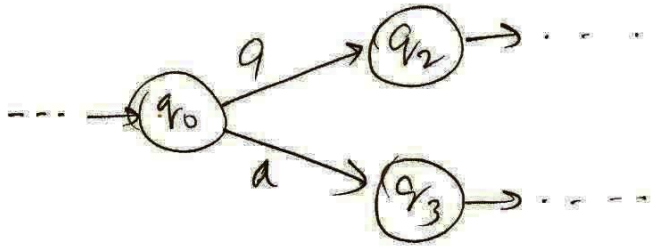
(i) $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \xrightarrow{b} q_4 \xrightarrow{a} q_5$

(ii) $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \xrightarrow{a} q_3 \ldots \xrightarrow{a} q_3$

Lect. 9

⇒ NDFA or NFA

more than one next states on particular input.



⇒ Equivalence of M_1 and M_2

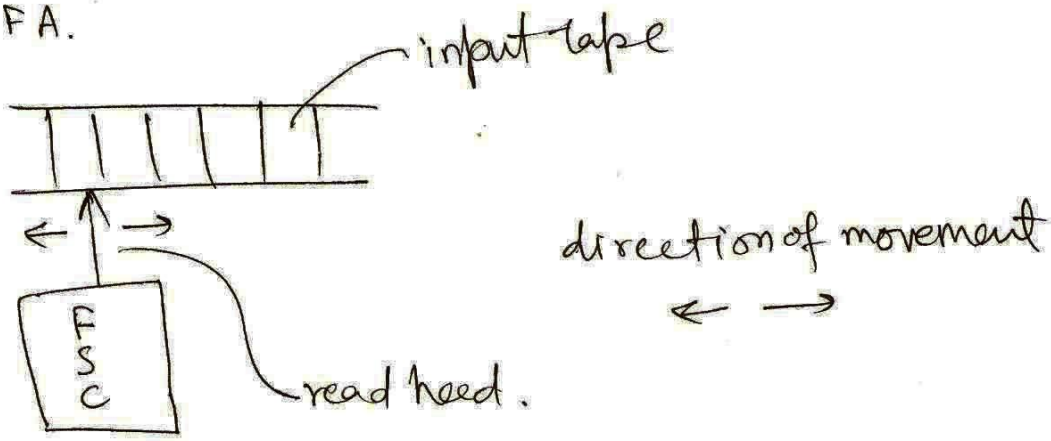
$L(M_1) = L(M_2)$

(i) language accepted by both m/c is same.

(ii) both m/c accept / recognize same set of strings.

⇒ NDFA to DFA.

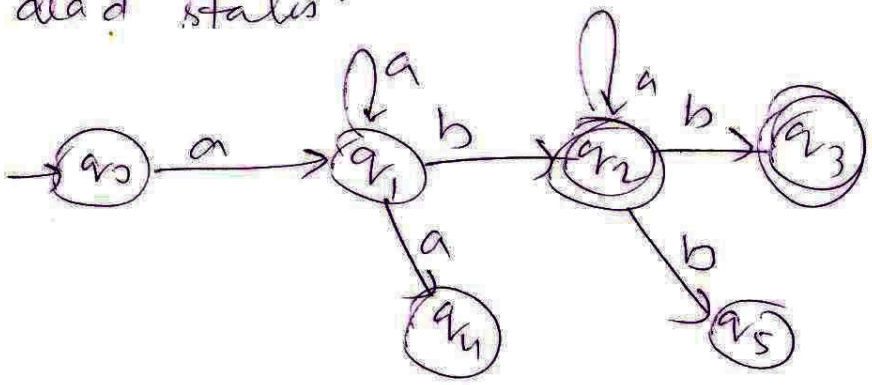
⇒ 2-way FA.



$\delta: Q \times \Sigma \rightarrow Q \times \{L, R\}$

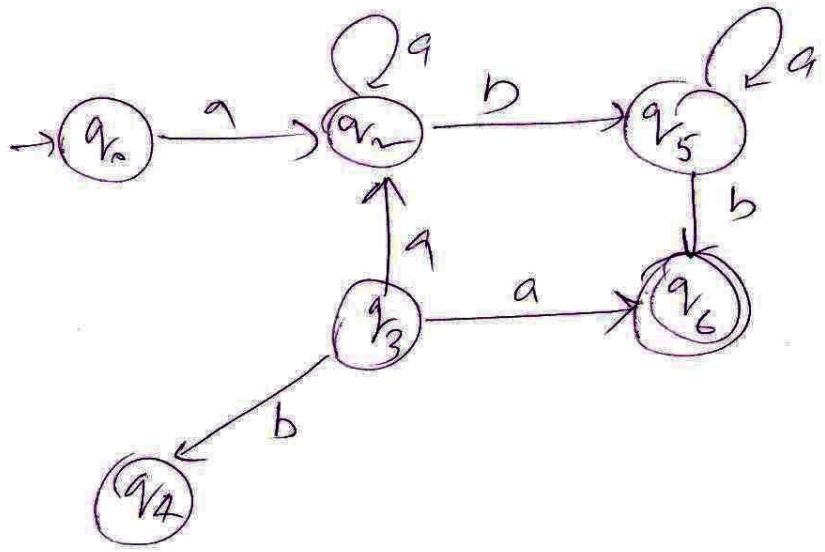
Dead States - which are accessible but useless states.

for example, in diagram below q_4, q_5 are dead states.

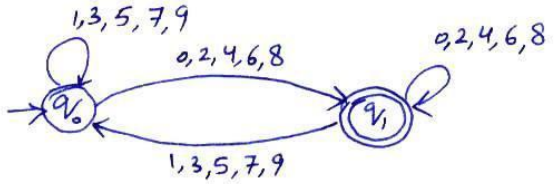


Non reachable states

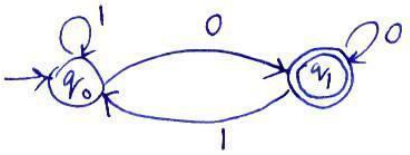
The states which are not reachable on any possible input. e.g. q_3, q_4 are non reachable states.



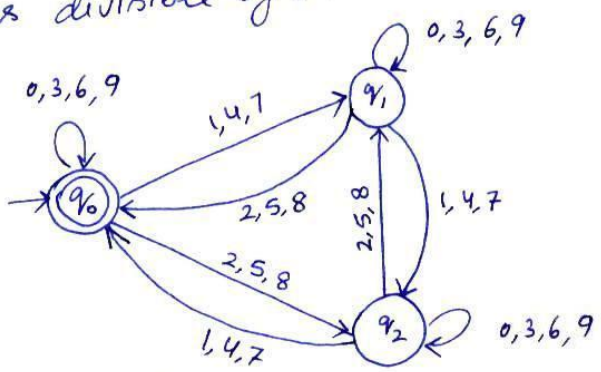
Following is the finite automaton to accept all even integer values:



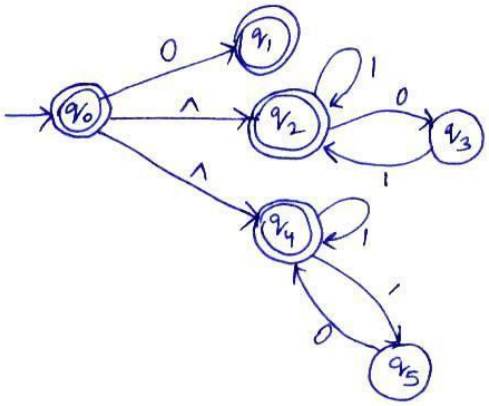
Following is the finite automaton to accept binary numbers whose equivalent decimal integer is even:



Following is the finite automaton to accept decimal numbers divisible by 3:



Following is the finite automaton such that 00 is not a substring:



Lect. 10

2-way FA

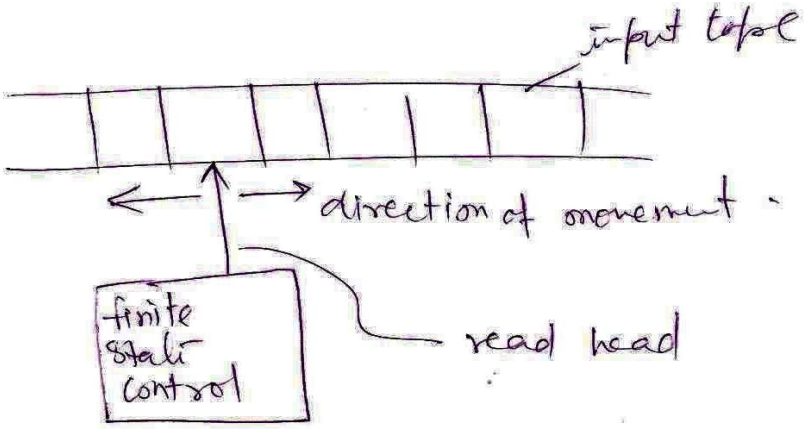
In the basic model of FA if we allow the read head to move in both directions then it becomes 2 way FA.

The transition function of 2-way FA is

$$Q \times \Sigma \longrightarrow Q \times \{L, R\}$$

- Q: fr. state
- Σ: fr. input
- Q: next state
- $\{L, R\}$: direction of movement of read head.

model



δ can be defined as:

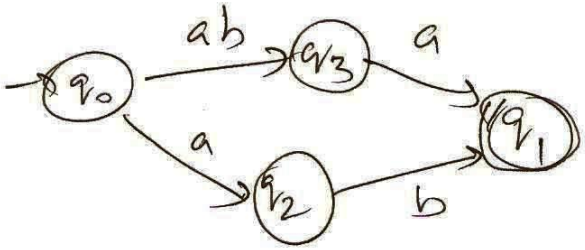
$$\delta(q_0, a) = (q_2, L)$$

$$\delta(q_0, b) = (q_1, R)$$

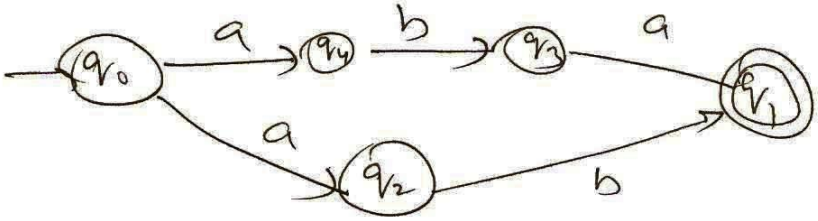
GTGs (Generalized transition Graphs)

When input is string atleat once in transition graph.

Eg

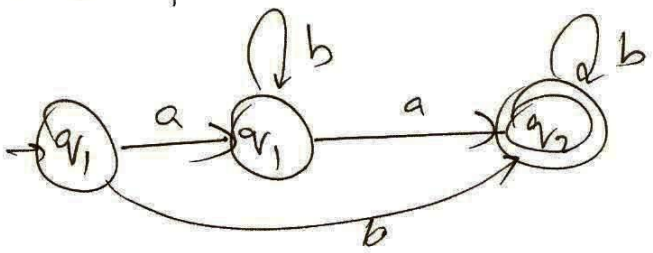


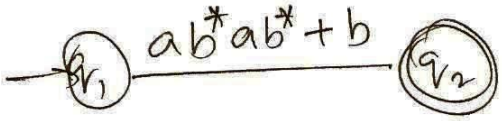
GTGs.



TGs.

find GTGs. for

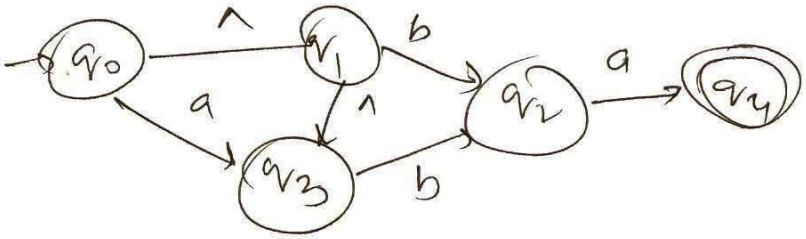




Lect. 11

finite automata with $\wedge$-moves

$\delta: Q \times \Sigma \cup \{\wedge\} \longrightarrow Q$



it accepts $s = bq$

$q_0 \xrightarrow{\wedge} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_4$

$\wedge ba = ba$

FA with output — Moore m/c, Mealy m/c

Moore m/c

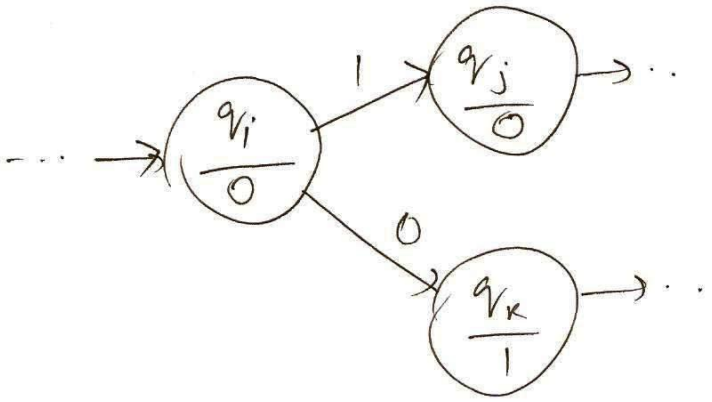
$\delta: Q \times \Sigma \to Q$

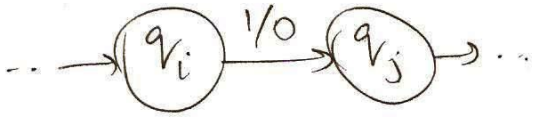
$\lambda: Q \to \Delta$

Mealy MC

$\delta: Q \times \Sigma \to Q$

$\lambda: Q \times \Sigma \to \Delta$





(i) Conversion from Moore m/c to Mealy m/c

(ii) Conversion from Mealy m/c to Moore m/c

(iii) Diff. bet. Mealy and Moore m/c

(iv) Equivalence of Mealy & Moore m/c

equivalence → both accept same langc.

not equivalem → both produce diff o/p.

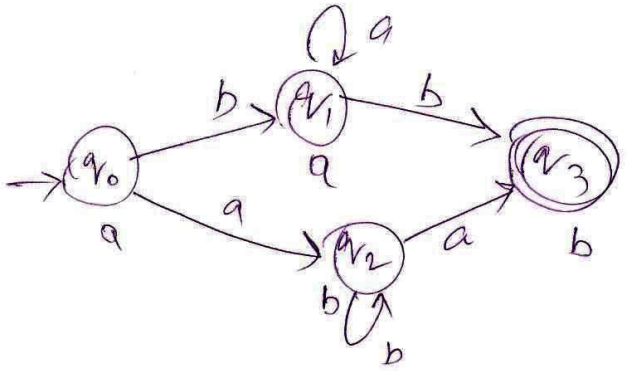
Moore m/c
input length = n
output length = n+1

Mealy m/c
input length = n
output length = n

Lect. 12

from Moore m/c to Mealy m/c.

Ex.



The transition table is

Pr. State	Next State		output
	a	b	
$\rightarrow q_0$	q_2	q_1	a
q_1	q_1	q_3	a
q_2	q_3	q_2	b
(q_3)	–	–	b

equivalent Mealy m/c

Pr. State	Next State			
	I/P a	o/p	I/P b	o/p
$\rightarrow q_0$	q_2	b	q_1	a
q_1	q_1	a	q_3	b
q_2	q_3	b	q_2	b
(q_3)				

Lect. 13

Minimization of FA.

1. Elimination of dead state
2. Elimination of non reachable state

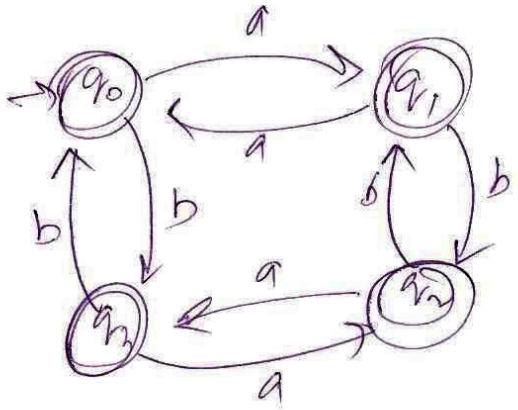
Apply partion rules until the groups are separated.

Use of transition property for separation.

initially two partions:

{set of final states}, {set of non final states}

Eg find minimum state FA for following.



The transition table of this FA is

Pr. State	Next State	
	a	b
→ q_0	q_1	q_3
q_1	q_0	q_2
q_2	q_3	q_1
q_3	q_2	q_0

set of final states = $\{ q_0, q_1, q_2, q_3 \}$

set of non final states = $\{ \phi \}$

$\delta(q_0, a) = q_1$

$\delta(q_1, a) = q_0$

$\delta(q_2, a) = q_3$

$\delta(q_3, a) = q_2$

as all the next state belongs to set of final states therefore no partition is possible

So the minimum FA is

Lect. 14

Introduction to Grammar

$G = (V_N, \Sigma, P, S)$

$L(G)$ = Language generated by grammar G.

Terminals and non terminals

Productions - definition.

Lang. from grammar

(i) $S \Rightarrow aS \mid \wedge$ and $S \rightarrow Sa \mid \wedge$

(ii) $S \rightarrow AB$

$A \rightarrow aA \mid \wedge$, $B \rightarrow bB \mid \wedge$

(iii) $S \rightarrow aSa \mid bS \mid b \mid a \mid b \mid \wedge$

Ex. find language of grammar $S \rightarrow aS \mid bS \mid \wedge$

it generalis all strings over (a,b)

therefore $L = \{s \in (a,b)^*\}$

Ex find language of grammar

$S \rightarrow aaS \mid abS \mid baS \mid bbS \mid \wedge$

It generates set of all strings over $\{a, b\}$ such that length of each strings is divisible by 2.

In the form of reg. expression it can be written as

$$R = (aa + ab + ba + bb)^*$$

It is regular language.

$$L = \{ s \in (a, b)^* \mid |s| \% 2 = 0 \}$$

$$= \{ s \in (aa, ab, ba, bb)^* \}$$

Ex find grammar for.

"Set of all strings over $\{0,1\}$ starting with 00 and ending with 11.

The reg. exp. for this lang. is

$$R = 00\,(0,1)^*\,11$$

The equivalent grammar is

$S \rightarrow ABC$

$A \rightarrow 00$

$B \rightarrow 0B \mid 1B \mid \wedge$

$C \rightarrow 11$

From language to grammar

we check whether lang. is finite or infinite. If the lang. is infinite then there is at least one prod. in which the variable in left side is also appearing in right side (direct loop) or there is an indirect loop.

for closure a^* or $a^n | n \geq 0$

$S \to aS | \Lambda \Rightarrow G = (\{S\}, \{a\}, \{S \to aS | \Lambda\}, S)$

for $(a, b)^*$ or $(a+b)^*$ or $(a \cup b)^*$ or $(a|b)^*$

$S \to aS | bS | \Lambda$, $G = (\{S\}, \{a, b\}, \{S \to aS | bS | \Lambda\}, S)$

Ex If $L = \{ab, ba, bb\}$ then following are its grammars:

(i) $S \to ab | ba | bb$

(ii) $S \to aB | Aa | Ab$, $A \to b$

Ex $L = \{$ set of all strings over (a, b) with even length $\}$

$$\left.\begin{array}{l} S \to AS | \Lambda \\ A \to aa | ab | ba | bb \end{array}\right\} P$$

$G = (\{S, A\}, \{a, b\}, P, S)$

Ex $L = \{a^n b^n | n \geq 0\}$,

$S \to \Lambda | aSb$

Ex $L = \{a^n b^{2n} | n \geq 2\}$

$S \to aSbb | aabbbb$

$a \underline{\geq} b$
$aa \underline{\geq} bb$
$aaa \underline{\geq} bbb$ ✓

$S \to abS$
$ababS$
$aababS$ ✗

Ex $L = \{a^n \mid n \geq 1\} \cup \{b^{2m} \mid m \geq 0\}$

$L = \{a, aa, aaa, \ldots\} \cup \{\wedge, bb, bbbb, \ldots\}$

$L = \{\wedge, a, aa, bb, aaa, bbbb, \ldots\}$

$S \to A \mid B$

A is representing $\{a^n \mid n \geq 1\}$ and B is representing $\{b^{2m} \mid m \geq 0\}$, the require grammar is

$S \to A \mid B$,

$A \to aA \mid a$

$B \to bbB \mid \wedge$

Ex $L = \{a^{n+1} b^{2n+3} \mid n \geq 0\}$

$S \to aSbb \mid abbb$

Ex $L = \{a^l b^m c^n d^k \mid l, m, n, k \geq 1\}$

$S \to ABCD$

$A \to aA \mid a$, $B \to bB \mid b$, $C \to cC \mid c$, $D \to dD \mid d$

Ex $L = \{a^m b^n \mid m > n \geq 0\}$

$S \to aSb \mid A$, $A \to aA \mid a$

Check whether $aaaabb \in L$ or not

$S \Rightarrow aSb$
$\Rightarrow aaSbb$
$\Rightarrow aaAbb$
$\Rightarrow aaaAbb$
$\Rightarrow aaaabb$

Ex $L = \{$ set of all palindromes over $(a, b)\}$

$= \{ s \in (a,b)^* \mid s = rev(s) \}$

$= \{ s, s^R \mid s \in (a,b)^* \}$

$a, b, \wedge$

$S \to aSa \mid bSb \mid a \mid b \mid \wedge$

Ex set of all strings over (a, b) where number of a's are equal to no. of b's.

$L = \{ s \in (a,b)^* \mid n_a(s) = n_b(s) \}$

$S \to abS \mid baS \mid aSb \mid bSa \mid Sab \mid Sba \mid \wedge$

Ex $L = \{ s \in (a,b)^* \mid n_a(s) > n_b(s) \}$

$S \to A \mid abS \mid baS \mid aSb \mid bSa \mid Sab \mid Sba$

$A \to aA \mid a$

Lecture 15

Find grammars of following languages

(i) $L = \{(ab)^n \mid n \geq 0\}$

(ii) Set of all palindromes over $\{0,1\}$ starting with 0.

(iii) $L = \{a^n b c^n \mid n \geq 0\}$

(iv) $L = \{a^n b^n c^n \mid n \geq 0\}$

(v) All strings over $\{0,1\}$ not ending with 0.

(i) $L = \{(ab)^n \mid n \geq 0\}$

$S \rightarrow abS \mid \wedge$

(ii) Set of all palindromes over $\{0,1\}$ starting with 0.

$S \rightarrow 0A0 \mid 0$
$A \rightarrow 0A0 \mid 1A1 \mid 0 \mid 1 \mid \wedge$

(iii) $L = \{a^n b c^n \mid n \geq 0\}$

$S \rightarrow aSc \mid b$

(iv) $L = \{a^n b^n c^n \mid n \geq 0\}$

$S \rightarrow aSBC \mid \wedge$
$B \rightarrow a$
$C \rightarrow c$
$CB \rightarrow BC$

(v) All strings over $\{0,1\}$ not ending with 0.

$S \rightarrow S_1 1$
$S_1 \rightarrow 0S_1 \mid 1S_1 \mid \wedge$

Ex2 find the language of following grammars:

(i) $S \to aS \mid \wedge$

(ii) $S \to aa$, $S \to bb$, $S \to \wedge$

(iii) $A \to aA \mid \wedge$, $S \to aA$

(iv) $S \to AB$, $A \to BB$, $B \to AA$

Sol. (i) $S \to aS \mid \wedge$ is equivalent to $S \to aS$, $S \to \wedge$

$S \Rightarrow \wedge$, $\wedge \in L$

$S \Rightarrow aS$

$\Rightarrow a\wedge = a$, $a \in L$

$S \Rightarrow aS$

$\Rightarrow aaS$ for $S \to \wedge$ $aa \in L$

$\Rightarrow aaaS$ for $S \to \wedge$ $aaa \in L$

$\vdots$

$= a^n S$

$L = \{a^n \mid n \geq 0\} = a^*$ (regular language)

(ii) $S \to aa$ $aa \in L$

$S \to bb$ $bb \in L$

$S \to \wedge$ $\wedge \in L$

$L = \{\wedge, aa, bb\} = \wedge + aa + bb$ (Regular language.

(iii) $A \to aA \mid \wedge$, $S \to aA$

$S \Rightarrow aA$ for $A \to \wedge$, $a \in L$

$\Rightarrow aaA$ for $A \to \wedge$, $aa \in L$

$\Rightarrow aaaA$ for $A \to \wedge$, $aaa \in L$

$\vdots$

$\Rightarrow a^n A$

$L = \{a^n \mid n \geq 1\} = a^+$ (regular language)

(iv) $S \to AB,\ A \to BB,\ B \to AA$

$L = \{\emptyset\}$ as there is no terminal in the grammar.

Ex. find language of following grammars

(i) $S \to aSb \mid ab$

(ii) $S \to A \mid B,\ A \to aA \mid \wedge,\ B \to bB \mid \wedge$

(iii) $S \to aSa \mid bSb \mid \wedge$

(iv) $S \to aSa \mid bSb \mid a \mid b$

(v) $S \to aSbb \mid bb$

Sol. (i) $S \Rightarrow ab, \quad ab \in L$

$S \Rightarrow aSb$

$\Rightarrow aaSbb$

$\Rightarrow aaaSbbb$

$\vdots$

$\Rightarrow a^{n-1}Sb^{n-1}$

$\Rightarrow a^{n-1}ab\,b^{n-1} = a^n b^n$

$L = \{a^n b^n \mid n \geq 1\}$ (it is not regular)

(ii) $S \to A \mid B,\ A \to aA \mid \wedge,\ B \to bB \mid \wedge$

the language will contain terminal strings generated by A and B.

$A \to aA \mid \wedge$ will generate $\{a^n \mid n \geq 0\}$

$A \overset{*}{\Rightarrow} a^n$

Similarly, B will generate $\{b^m \mid m \geq 0\}$

$L = \{a^n \mid n \geq 0\} \cup \{b^m \mid m \geq 0\}$

L will contain following strings.

$L = \{\wedge, a, b, aa, bb, aaa, bbb, \ldots\ldots\}$

(iii) $S \to aSa \mid bSb \mid \wedge$

It is representing all even length palindromes over $\{a, b\}$

$L = \{ s \in (a,b)^* \mid s = rev(s), |s| \%2 = 0 \}$

Let us show the steps to generate abbabbabba

$S \Rightarrow aSa$
$\Rightarrow abSba$ by $S \to bSb$
$\Rightarrow abbSbba$ by $S \to bSb$
$\Rightarrow abbaSabba$ by $S \to aSa$
$\Rightarrow abbabSbabba$ by $S \to bSb$
$= abbab \wedge babba = abbababba$
By $S \to \wedge$

aSa
abSba
abbSbba
abbaSabba
abbabaabba
abbaababba

(iv) $S \to aSa \mid bSb \mid a \mid b$ will generate all odd length palindromes over (a, b) with a or b as middle element.

$L = \{ s \in (a,b)^* \mid s = rev(s), |s| \%2 = 1 \}$

(v) $S \to aSbb \mid bb$

$S \Rightarrow aSbb$
$\Rightarrow aaSbbbb$
$\Rightarrow aaaSbbbbbb$
$\vdots$
$\Rightarrow a^n S b^{2n}$
$\Rightarrow a^n bb b^{2n}$. by $S \to bb$

$L = \{ a^n b^{2n+2} \mid n \geq 0 \}$

Q. find lang of following grammars

(i) $S \to aS \mid bS \mid \Lambda$

(ii) $S \to AS \mid \Lambda$, $A \to aa \mid bb$

(iii) $S \to ABCD$, $A \to aA \mid \Lambda$, $B \to bB \mid b$,
$C \to cC \mid cc$, $D \to dD \mid ddd$

Sol. (i) $S \to aS \mid bS \mid \Lambda$

Lang. will contain following strings

$L = \{\Lambda, a, b, aa, ab, ba, bb \ldots\ldots\}$
$= (a+b)^*$

$S \Rightarrow aS$ with $S \to \Lambda$, $S = a$
$= aaS$ with $S \to \Lambda$ $S = aa$
$\to aaaS$ with $S \to \Lambda$ $S = aaa$

$S \Rightarrow aS$ $\Rightarrow abS$ $\to ab\Lambda$	$S \Rightarrow bS$ $\Rightarrow baS$ $\Rightarrow ba\Lambda$	$S \Rightarrow bS$ $\Rightarrow bbS$ $\Rightarrow bb\Lambda$

(ii) $S \to AS \mid \Lambda$, $A \to aa \mid bb$

The prod. $S \to S \mid \Lambda$ will generate A^*
while A represents $(aa+bb)$

$L = (aa+bb)^*$

(iii) A will represent $\{a^n \mid n \geq 0\}$
B " " $\{b^n \mid n \geq 1\}$
C " " $\{c^l \mid l \geq 2\}$
D " " $\{d^k \mid k \geq 3\}$

The language of grammar is

$L = \{a^n b^m c^l a^k \mid n \geq 0, m \geq 1, l \geq 2, k \geq 3\}$

Lect. 16

Chomsky hierarchy

- Types of grammars
- Types of languages
- related Automata

Type 0 gram. = Unrestricted grammar

⇒ Type 0 Language

Type 1 gram. = context sensitive grammar

⇒ context sensitiv Language (CSL)

Type 2 gram. ⇒ Type 2 Lang. or CFL

Type 3 gram ⇒ Type 3 Lang. or R.L.

Chomsky Hierarchy

Four types of Grammars, four types of language and four types of machines

Grammar	Prod. Type	Language	Automata
1. Type-0 or unrestricted grammar	Any prod. is allowed	Type-0 lang. or unrestricted lang.	Turing m/c
2. Type-1 or context sensitive grammar (CSG)	$\alpha A \beta \rightarrow \gamma$ $\alpha, \beta \in V_N \cup \Sigma$ $A \in V_N$, $\gamma \in (V_N \cup \Sigma)^*$ α = left context β = right context Ex $BAb \rightarrow abc$	Type-1 language or context sensitive language (CSL)	LBA

3. Type-2 or context free grammar (CFG)	$A \to \alpha$ $A \in V_N$ $\alpha \in (V_N \cup \Sigma)^*$ Ex $S \to aS \mid \wedge$	Type-2 lang. or context free language (CFL)	PDA
4. Type-3 or Regular grammar (RG)	$A \to a \mid bB$ in right side small letter, or small letter followed by capital letter.	Type-3 language or regular lang.	FA

Ex the Prod. $S \to a \mid bS$ represent RG.

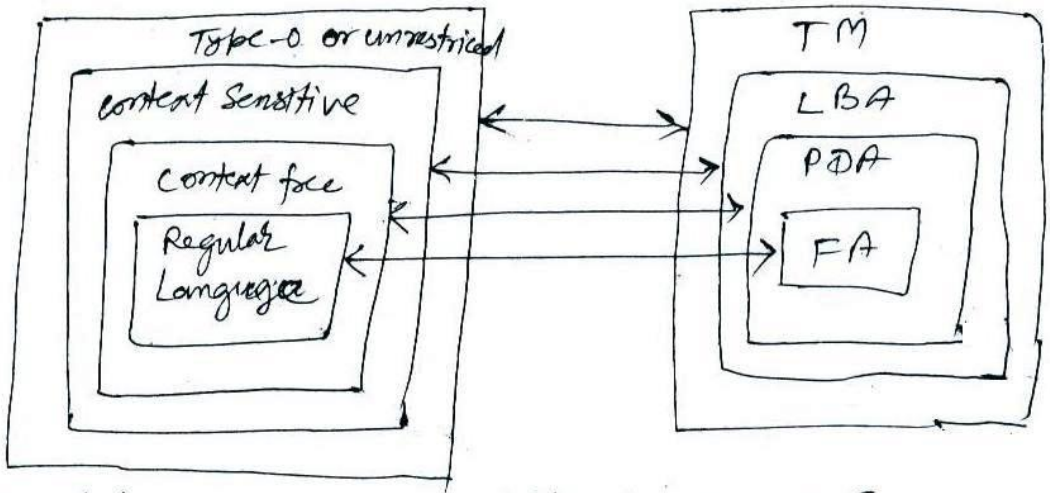
it also represents CFG

it also represents CSG as $\wedge S \wedge \to a \mid bS$

it also represents Type 0 grammar

$RL \subseteq CFL \subseteq CSL \subseteq T_0$

FA → PDA → LBA → TM



Ex what is the type of following grammar

$S \to aSBC$, $CB \to BC$, $B \to b$, $C \to c$

(2) (1) (3) (3)

The overall type of above grammar is type-1 or context sensitive. $L = \{a^n b^n c^n \mid n \geq 1\}$

Lect. 17

Relation among type of Automata & languages

FA	–	Regular Language (RL)	– Regular Grammar (RG)
PDA	–	Context free Language (CFL)	– Context free Grammar (CFG)
LBA	–	Context sensitive Lang. (CSL)	– Context sensitive Gram. (CSG)
TM	–	Unrestricted Lang. (Type 0 Lang.)	– Unrestricted Gram. (Type 0 grammar)

Relation among M/C.

$FA \subseteq PDA \subseteq LBA \subseteq TM$

Relation among Lang.

$RL \subseteq CFL \subseteq CSL \subseteq T_{0L}$

Relation among grammars

$RG \subseteq CFG \subseteq CSG \subseteq T_{0G}$

Lecture 18

1. Simplification of reg. exp.
   - using identities
   - using FA construction

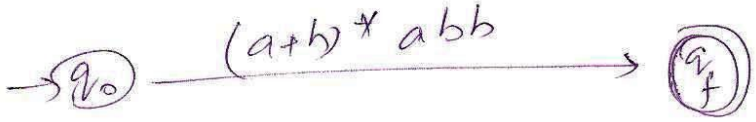
2. reg. exp. to FA
   (i) elimination of Union
   (ii) " " closure
   (iii) " " concatination.

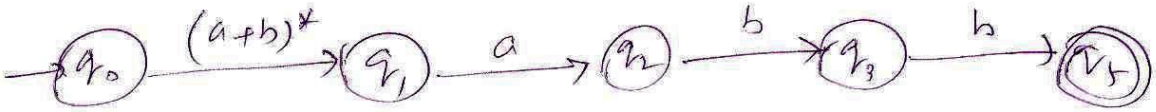
3. Elimination of $\wedge$ moves

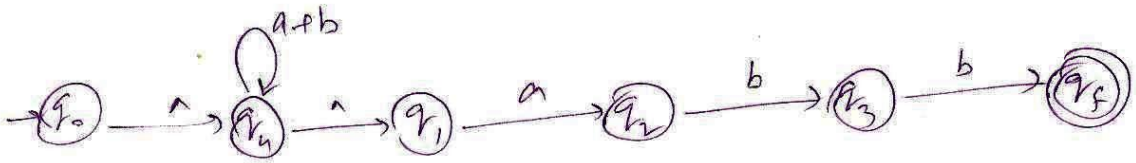
4. Arden's theorem
   If $R = Q + RP$ then $R = QP^*$

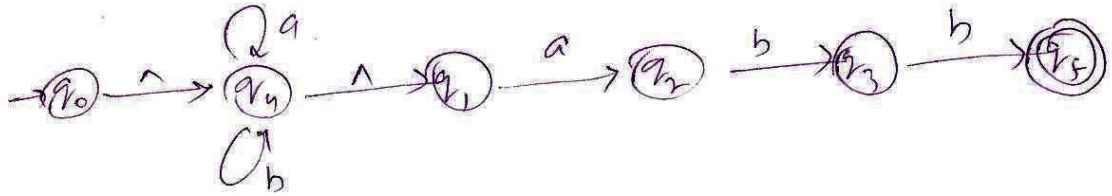
5. FA to reg. exp.

Ex. find FA for $(a+b)^* abb$.









Reg. grammar for this is

$$G = \{ \; S \to ABC,\ A \to aA \mid bA \mid \wedge \quad B \to a,\ C \to bD,\ D \to b. \; \}$$