UNIT-3 (REGULAR SETS & REGULAR GRAMMARS)

Regular Expression

The regular expressions are used for representing certain set of strings in an algebraic fashion.

APPLICATIONS -

1) Search commands - Systems use regular expression like notation for describing patterns that user wants to find in a file or web-browser. Different search systems convert the regular expression into DFA or NDFA, & simulate the automata on file being searched.

2) Lexical analyzer generator, a component of compiler accepts descriptions of the forms of tokens, which are essentially regular expressions, & produces a DFA that recognizes which token appears next on the input.

FORMAL RECURSIVE DEFINITION OF REGULAR EXPRESSION OVER Σ

1. Any terminal symbol (i.e. an element of Σ), ε and $\emptyset$ are regular expressions. When we view $a \in \Sigma$ as a regular expression, we denote it by **a** (BOLD a)

2. The union of 2 regular expressions R_1 & R_2, written as $R_1 + R_2$ is also a regular expression.

3. The concatenation of 2 r.e. R_1 & R_2, written as R_1R_2, is also a r.e.

4. The iteration (or closure) of a r.e. R, written as R^* is also a regular expression. (reflexive & transitive closure of concatenation) OR KLEENE-STAR

5. If R is a r.e., then (R) is also a r.e.

6. The r.e. over Σ are precisely those obtained recursively by application of the rules 1-5 once or several times.

Note that the parentheses used in def^n above help in the order of evaluation of a r.e.

Also, in the absence of parentheses, we have the Hierarchy of operations as follows:

Iteration (Closure)	>	Concatenation	>	Union

SIMILAR TO ARITHMETIC :- exp > mult. > add.

REGULAR SET

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

example:- $a, b \in \Sigma$, then

(i) a denotes $\{a\}$

(ii) $a+b$ denotes $\{a, b\}$

(iii) ab denotes $\{ab\}$

(iv) a^* denotes $\{\wedge, a, aa, aaa, \ldots\}$ and

(v) $(a+b)^*$ denotes $\{a, b\}^*$

The set represented by R is denoted by $L(R)$.

$L(R_1 + R_2) = L(R_1) \cup L(R_2)$; i.e. a string in $L(R_1+R_2)$ is a string from r.e. R_1 or r.e. R_2

$L(R_1 R_2) = L(R_1) L(R_2)$

$L(R^*) = (L(R))^* = \bigcup_{n=0}^{\infty} L(R)^n$

$L(\phi) = \phi$

$L(a) = \{a\}$

Remark: By def^n of r.e., the class of regular sets over Σ is closed under union, concatenation & closure (iteration).

Describe the followings sets by r.e.

(a) {101} 101

(b) {abba} abba

(c) {01, 10} 01 + 10

(d) {∧, ab} ∧ + ab

(e) {abb, a, b, bba} abb + a + b + bba

(f) {∧, 0, 00, 000, ...} 0^*

(g) {1, 11, 111, ...} ~~0^*~~ $1(1)^*$

(h) the set of all strings over {0, 1} ending in 00 $(0+1)^*00$

(i) the " " " " " " beginning with 0 & ending with 1. $0(0+1)^*1$

(j) {∧, 11, 1111, 111111, ...} $(11)^*$

IDENTITIES FOR REGULAR EXPRESSIONS

Two r.e. P & Q are Equivalent (we write P = Q) if P and Q represent the same set of strings.

Identities :-

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

where R means the set represented by r.e. R.

ARDEN'S THEOREM

Theorem:- Let P and Q be two regular expressions over Σ. If P does not contain Λ, then the following eqn. in R, namely

$$R = Q + RP \quad \text{——— (★)}$$

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

Proof:- First, we show that QP^* is a sol^n of (★),

$$\text{RHS} = Q + RP = Q + (QP^*)P$$
$$= Q + QP^*P = Q(\Lambda + P^*P)$$
$$= QP^*$$
$$= R = \text{LHS}.$$

$\therefore$ QP^* satisfies (★), & hence is a sol^n of (★).

Next, to prove UNIQUENESS, replacing R by $Q+RP$ on RHS of (★), we obtain

$$Q + RP = Q + (Q + RP)P$$
$$= Q + QP + RPP = Q + QP + RP^2$$
$$= Q + QP + (Q + RP)P^2$$
$$= Q + QP + QP^2 + RP^3$$
$$\vdots$$
$$= Q + QP + QP^2 + \ldots + QP^{i-1} + RP^{i+1}$$
$$= Q(\Lambda + P + P^2 + \ldots + P^i) + RP^{i+1}$$

$\therefore$ $R = Q(\Lambda + P + P^2 + \ldots + P^i) + RP^{i+1}$ for $i \geq 0$ ——— (★★)

We now show that any sol^n of (★) is equivalent to QP^*. Suppose R satisfies (★), then it satisfies (★★). Let w be a string of length i in the set R. Then, w belongs to the set $Q(\Lambda + P + P^2 + \ldots + P^i) + RP^{i+1}$. As P doesn't contain Λ, RP^{i+1} has no string of length $< (i+1)$ & so w is not in

the set RP^{i+1}. This means that w belongs to the set $Q(\Lambda + P + P^2 + \dots + P^i)$, & hence QP^*.

Consider, a string w in the set QP^*. Then, w is in the set QP^k for some $k \geq 0$, and hence in $Q(\Lambda + P + P^2 + \dots + P^k)$. So, w is on the RHS of (**). ∴ w is in R (LHS of (**)). Thus, R & QP^* represent the same set. This proves the uniqueness of the sol^n of (*).

Ques:- a) Give a r.e. for representing the set L of strings in which every 0 is immediately followed by at least two 1's.

b) Prove that the r.e. $R = \Lambda + 1^*(011)^*(1^*(011)^*)^*$ also describe the same set of strings.

Sol^n: a) A string $\omega \in L$ if w doesn't contain any 0 or it contain atleast 2 1's followed by 0.

∴ $\omega = \omega_1 \omega_2 \dots \omega_n$ where ω_i is either 1 or 011.

So, $L = (1 + 011)^*$ (Note $\omega = 111$
$\omega = 011$ or $011011\dots$
$\omega = 0111$
by combining 011 with 1 from 1^*)

b)

$R = \Lambda + 1^*(011)^*(1^*(011)^*)^*$

$= \Lambda + PP^*$ where $P = 1^*(011)^*$

$= P^*$ (using identity $\Lambda + RR^* = R^*$)

$= (1^*(011)^*)^*$

$= (1 + 011)^*$ (using $(P+Q)^* = (P^*Q^*)^*$)

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

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

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

$= (1+00^*1)(0+10^*1)^*$ (Using $\Lambda + R^*R = R^*$)

$= 1(\Lambda+00^*)1(0+10^*1)^*$ (Note comm from RHS)

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

$=$ RHS.

FINITE AUTOMATA & REGULAR EXPRESSIONS

Transition Systems containing Λ-moves

The transition systems can be generalized by permitting Λ-moves or Λ-transitions which are associated with a null symbol Λ. These transitions can occur when no input is applied.

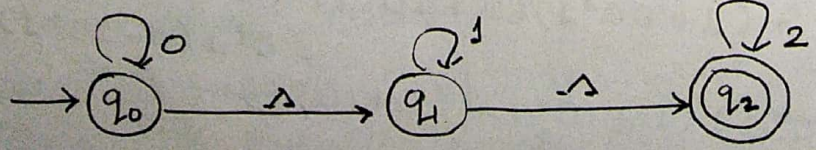
It is possible to convert a transition system with Λ-moves into an equivalent transition system without Λ-moves.

PROCEDURE

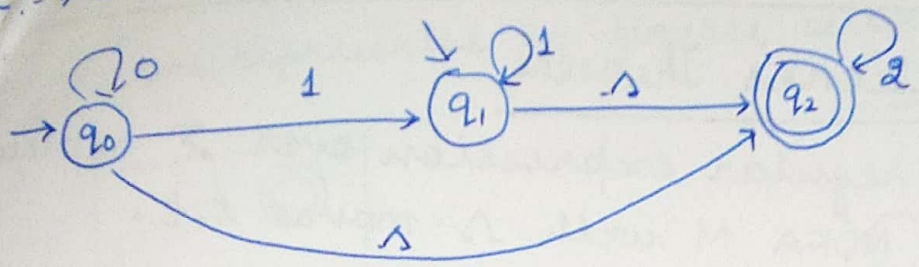
Suppose we want to replace a Λ-move from vertex v_1 to vertex v_2. Then, we proceed as follows:-

1. Find all the edges starting from v_2.
2. Duplicate all these edges now starting from v_1, without changing the edge labels.
3. If v_1 is an initial state, make v_2 also an initial state.
4. If v_2 is a final state, make v_1 also as the final state.

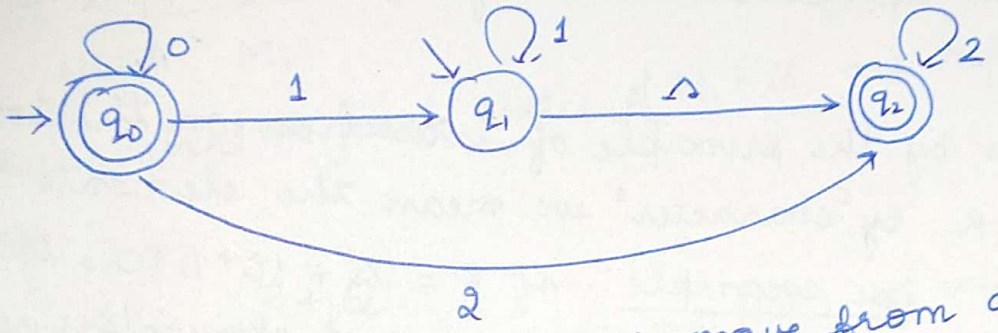
Ques (a) Convert given automata with Λ-moves into an equivalent automata without Λ-moves.



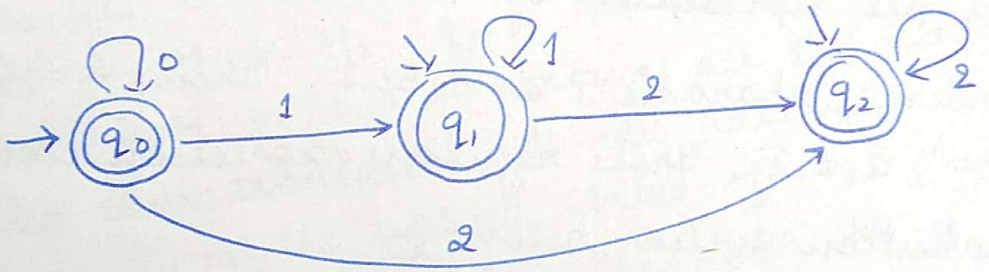
:- i) First eliminating $\wedge$-move from q_0 to q_1



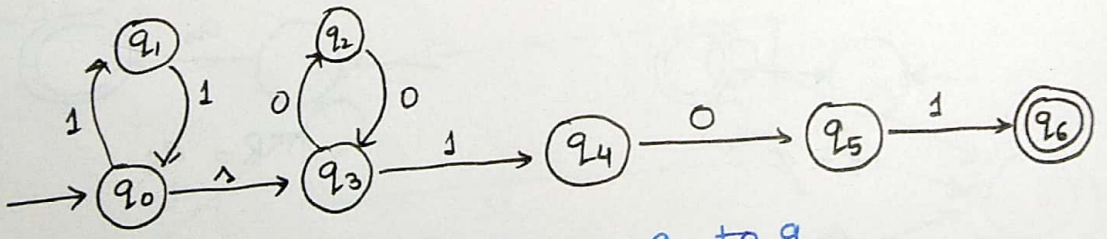
ii) Eliminate $\wedge$-move from q_0 to q_2.



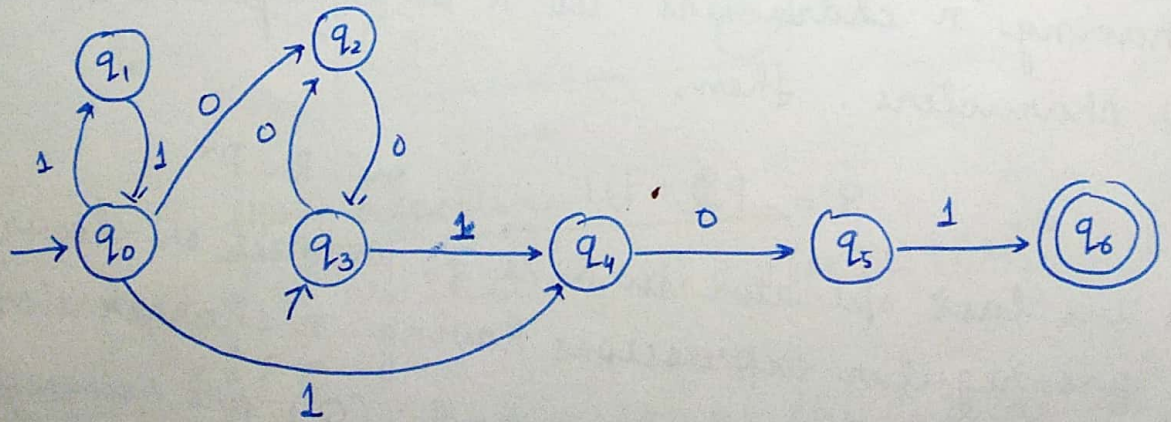
iii) Next, we eliminate $\wedge$-move from q_1 to q_2.



(b)



(i) Eliminate $\wedge$-move from q_0 to q_3



NDFA with Λ-moves & Regular Expressions

Kleene's Theorem

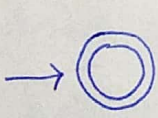
Theorem:- If R is a regular expression over Σ represents $L \subseteq \Sigma^*$, then $\exists$ an NDFA M with Λ-moves s.t.

$$L = T(M)$$

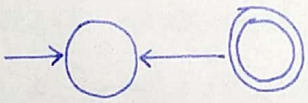
i.e. every regular expression is recognized by a NDFA with Λ-moves.

Proof:- the proof is by the principle of induction on the total # of characters in R. By 'character' we mean the elements of Σ, Λ, ϕ, * and +. for example:- if $R = \Lambda + 10^* 11^* 0$, the characters are Λ, +, 1, 0, *, 1, 1, *, 0 & the # of characters is 9. let L(R) denote the set represented by R.

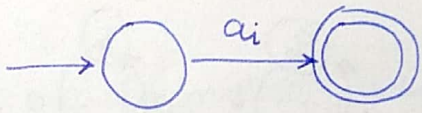
BASIS:- let the number of characters in R be 1. Then, $R = \Lambda$, or $R = \phi$, or $R = a_i$; $a_i \in \Sigma$. These regular expression can be recognized by transition system below:-



$R = \Lambda$



$R = \phi$



$R = a_i$

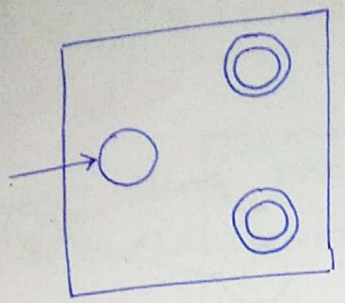
INDUCTION STEP:- Assume that the theorem is true for regular expressions having 'n' characters. let R be a regular expression having (n+1) characters. Then,

$R = P + Q$ or $R = PQ$ or $R = P^*$

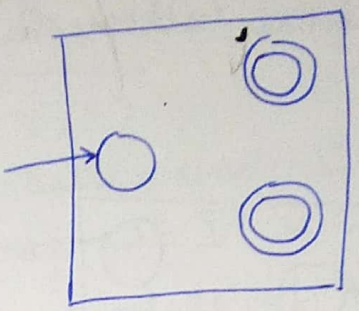
according as the last operator in R is +, product or closure. Also, P and Q are regular expressions having n characters or less. By induction hypothesis L(P) and L(Q) are recognized by M_1 and M_2 where M_1 & M_2 are NDFA's with Λ-moves, s.t.

$L(P) = T(M_1)$ and $L(Q) = T(M_2)$.

M_1 & M_2 are represented by figures below :-



NDFA M_1

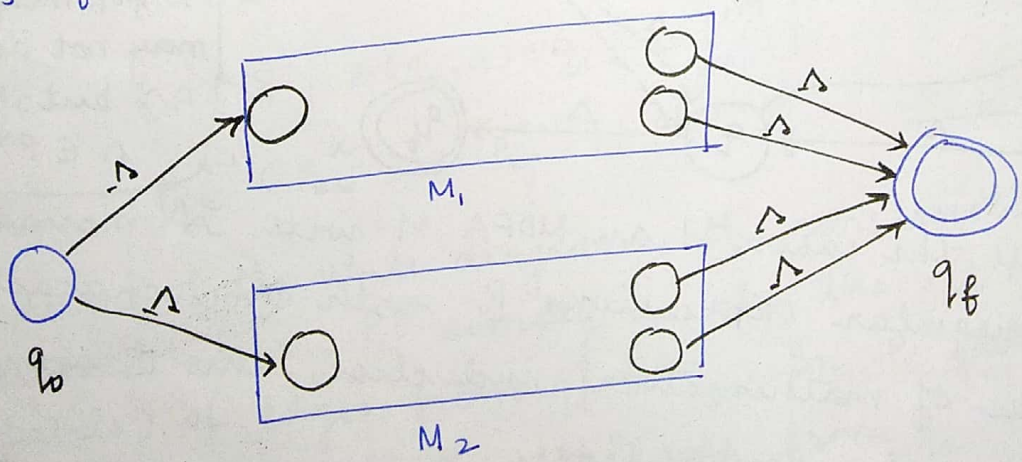


NDFA M_2

The initial & final states of M_1 & M_2 are represented in usual way.

Case 1 : $P + Q$

In this case we construct an NDFA M with Λ-moves that accepts $L(P+Q)$ as follows : q_0 is the initial state of M, q_0 not in M_1 or M_2. q_f is the final state of M; once again q_f not in M_1 or M_2. M contains all the states of M_1 and M_2 & also their transitions. We add additional Λ-transitions from q_0 to the initial states of M_1 & M_2 & from the final states of M_1 & M_2 to q_f. The NDFA M is as given below :-



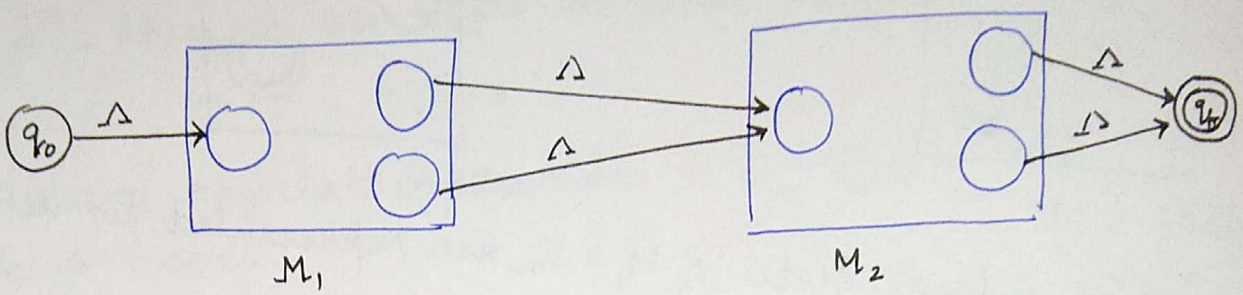
NDFA accepting $L(P+Q)$

It is easy to see that $T(M) = T(M_1) \cup T(M_2) = L(P+Q)$

Case 2 : $R = PQ$

In this case, we introduce q_0 as the initial state of M and q_f as the final state of M, both q_0 & q_f not in M_1 or M_2. New

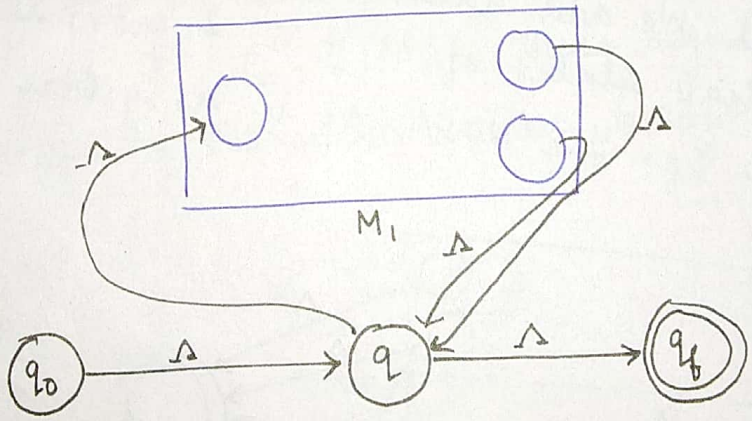
Λ-transitions are added b/w q_0 & the initial state of M_1 & between the final states of M_1 & the initial state of M_2, and between the final states of M_2 & the final state q_f of M.



NDFA accepting L(PQ)

Case 3: $R = (P)^*$

In this case, q_0, q and q_f are introduced. New Λ-transitions are introduced from q_0 to q, q to q_f, q to the initial state of M_1 & from the final state of M_1 to q.



Here, q is included to include Λ string in R ∵ M_1 may or may not accept Λ but M should ∵ $\Lambda \in P^* = R$

Thus, in all the cases, ∃ an NDFA M with Λ-moves, accepting the regular expressions R with (n+1) characters. By the principle of mathematical induction, this theorem is true for all regular expressions.

Remark:- By last theorem, every regular expression is accepted by an NDFA with empty moves.

⇓ converted to

NDFA without Λ-moves

converted to ⟶ A DFA

∴ for every regular expression R we can construct a DFA accepting L(R).

Theorem:- Any set L accepted by a finite automata M is represented by a regular expression.

(Converse of Last Theorem)

Proof:- Let $M = (\{q_1, \ldots, q_n\}, \Sigma, \delta, q_1, F)$

The construction that we give can be better understood in terms of the state diagram of M. If a string $\omega \in \Sigma^*$ is accepted by M, then $\exists$ a path from q_1 to some final state of M with path value ω. So, to each final state, say q_j, there corresponds a subset of Σ^* consisting of path values of paths from q_0 to q_j. As T(M) is the union of such subsets of Σ^*, it is enough to represent them by regular expressions. So, the main part of the proof lies in the construction of subsets of path values of paths from the state q_i to the state q_j.

Let P_{ij}^k denote the set of path values of paths from q_i to q_j whose intermediate vertices lie in $\{q_1, \ldots, q_k\}$. We construct P_{ij}^k for $k = 0, 1, \ldots, n$ recursively as follows:

$$P_{ij}^0 = \{a \in \Sigma \mid \delta(q_i, a) = q_j\} \quad \text{---(1)}$$

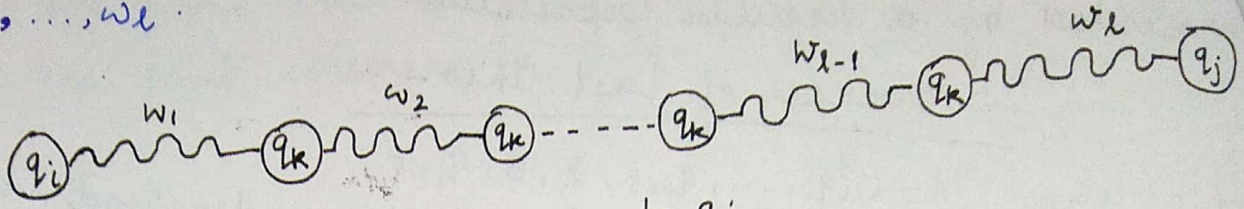
$$P_{ii}^0 = \{a \in \Sigma \mid \delta(q_i, a) = q_i\} \cup \{\Lambda\} \quad \text{---(2)}$$

$$P_{ij}^k = P_{ik}^{k-1} (P_{kk}^{k-1})^* P_{kj}^{k-1} \cup P_{ij}^{k-1} \quad \text{---(3)}$$

In terms of the state diagram, the construction can be understood better. P_{ij}^0 simply denotes the set of path values (i.e. labels) of edges from q_i to q_j. In P_{ii}^0 we include Λ in addition to labels of self-loops from q_i. This explains (1) & (2).

Consider a path from q_i to q_j whose intermediate vertices lie in $\{q_1, \ldots, q_k\}$. If the path does not pass through q_k, then its path value lies in P_{ij}^{k-1}. Otherwise, the path passes through q_k possibly more than once.

The path can be split into several paths with path values $w_1, w_2, \ldots, w_\ell$.



A path from q_i to q_j

$w = w_1 w_2 \ldots w_\ell$. w_1 is the path value of the path from q_i to q_k (without passing through q_k, i.e. q_k is not an intermediate vertex). $w_2, \ldots, w_{\ell-1}$ are the path values of paths from q_k to itself without passing through q_k. w_ℓ is the path value of the path from q_k to q_j without passing through q_k. So, w_1 is in P_{ik}^{k-1}, $w_2, \ldots, w_{\ell-1}$ are in (P_{kk}^{k-1}), & w_ℓ is in P_{kj}^{k-1}. This explain ③.

We prove that the sets introduced by ① – ③ are represented by regular expressions by induction on k (∀ i and j). P_{ij}^0 is a finite subset of Σ, say $\{a_1, \ldots, a_t\}$. Then, P_{ij}^0 is represented by $P_{ij}^0 = a_1 + a_2 + \ldots + a_t$ (UNION). Illy, we can construct P_{ii}^0 representing P_{ii}^0. Thus, there is a basis for induction.

Let us assume the result holds for (k-1), i.e. P_{ij}^{k-1} is represented by a regular expression P_{ij}^{k-1} ∀ i, j. From ③

$$P_{ij}^k = P_{ik}^{k-1} (P_{kk}^{k-1})^* P_{kj}^{k-1} \cup P_{ij}^{k-1}.$$

∴ the result is true for all k. By principle of induction the sets constructed by ① – ③ are represented by regular expressions.

As $Q = \{q_1, \ldots, q_m\}$, P_{1j}^m denotes the set of path values of all paths from q_1 to q_j. If $F = \{q_{f_1}, \ldots, q_{f_n}\}$, then

$T(M) = \bigcup_{j=1} P_{1f_j}^m$. So, $T(M)$ is represented by the r.e. $P_{1f_1}^m + \ldots + P_{1f_n}^m$. Thus, $L = T(M)$ is represented by a r.e.

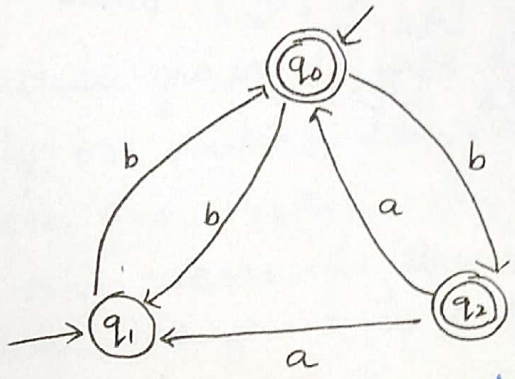
CONVERSION OF NONDETERMINISTIC SYSTEMS TO DETERMINISTIC SYSTEMS

↓

Transition systems i.e. # of initial states can be > 1.

The construction is similar to the construction of conversion of an NDFA to DFA except for the initial step. In the earlier method, we started with $[q_0]$. Here, we start with the set of all initial states. The other steps are similar.

Ques Obtain the deterministic graph (system) equivalent to the transition system below:

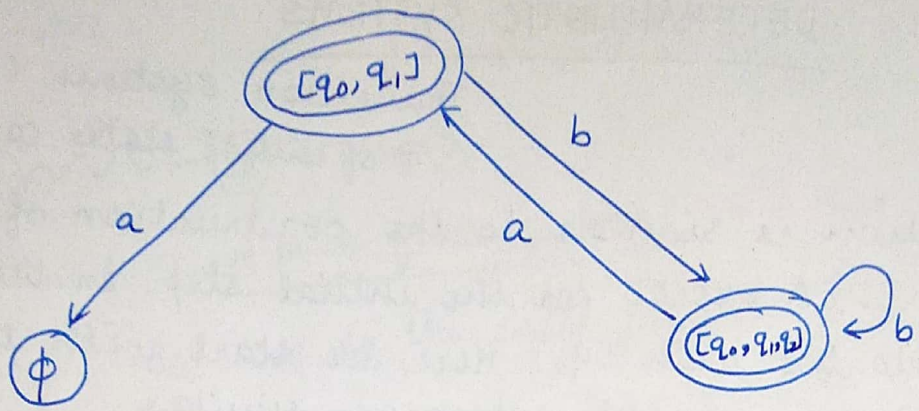


The transition table for the above automata :-

State/Σ	a	b
→ q_0		q_1, q_2
→ q_1	~~q_2~~	q_0
q_2	q_0, q_1	

We construct the table for corr. deterministic system by starting with $[q_0, q_1]$

State/Σ	a	b
$[q_0, q_1]$	$\emptyset$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2]$	$[q_0, q_1]$	$[q_0, q_1, q_2]$
$\emptyset$	$\emptyset$	$\emptyset$



As q_0, q_2 were final states; the new final states are the subsets containing either of q_0 or q_2.

$\therefore$ $[q_0, q_1]$ as well as $[q_0, q_1, q_2]$ both are final states. Initial state is $[q_0, q_1]$.

NOTE:- P_{ij}^{0} and P_{ii}^{0} are the subsets of $\Sigma \cup \{\Lambda\}$ & so they are finite sets. So, every P_{ij}^{k} is obtained by applying union, concatenation & closure to the set of all singletons in $\Sigma \cup \{\Lambda\}$.

ALGEBRAIC METHOD USING ARDEN'S THEOREM

($R = Q + RP \Rightarrow R = QP^*$)

The following method is an extension of Arden's theorem. This is used to find the r.e. recognized by a transition system.

The following assumptions are made regarding the transition system:

(i) The transition graph doesnot have empty moves.

(ii) It has only one initial state, say v_1.

(iii) Its vertices are $v_1, v_2, \ldots, v_n$.

(iv) V_i the r.e. represents the set of strings accepted by the system even though v_i is a final state.

(v) α_{ij} denotes the r.e. representing the set of labels of edges from v_i to v_j. When there is no edge, $\alpha_{ij} = \emptyset$. Consequently, we can get the following set of eq^{n}s of $V_1, \ldots, V_n$:

$$V_1 = V_1\alpha_{11} + V_2\alpha_{21} + \ldots + V_n\alpha_{n1} + \Lambda \quad \because q_1 \text{ is initial state.}$$

$$V_2 = V_1\alpha_{12} + V_2\alpha_{22} + \ldots + V_n\alpha_{n2}$$

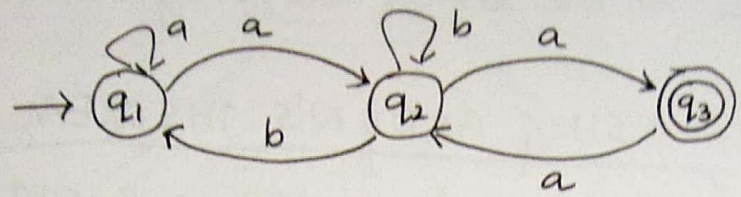
$$\vdots$$

$$V_n = V_1\alpha_{1n} + V_2\alpha_{2n} + \ldots + V_n\alpha_{nn}$$

By repeatedly applying substitutions and using Arden's theorem, we can express V_i's in terms of α_{ij}'s.

For getting the set of strings recognized by the transition system, we have to take the 'union' of all V_i's corresponding to final states.

Ques Consider the transition system given below. Prove that the set of strings recognized are $(a + a(b+aa)^*b)^* a(b+aa)^*$



Sol^n :-

Since the graph doesn't contain any Λ-moves & all other assumptions are satisfied, we can use Algebraic method using Arden's Theorem.

The 3 eq^{n}s for q_1, q_2 & q_3 can be written as :-

$$q_1 = q_1a + q_2b + \Lambda$$
$$q_2 = q_1a + q_2b + q_3a$$
$$q_3 = q_2a$$

We reduce # of unknowns by repeated substitution.

$$q_2 = q_1a + q_2b + (q_2a)a$$
$$= q_1a + q_2(b+aa)$$

$\Rightarrow q_2 = q_1a(b+aa)^*$ (Using Arden's Theorem)

$R = Q + RP \iff R = QP^*$

Now, $q_1 = q_1a + q_2b + \Lambda$

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

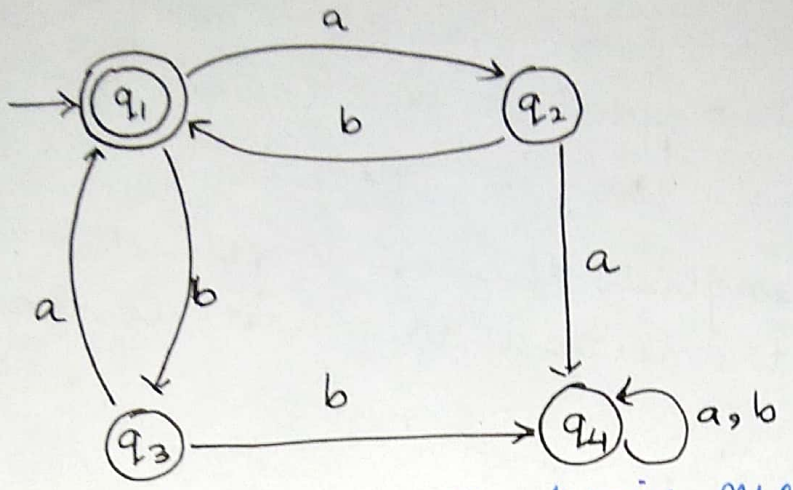
$\Rightarrow q_1 = \Lambda \cdot (a + a(b+aa)^*b)^*$ (Using Arden's Thm.)

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

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

$\because q_3$ is a final state, the set of strings recognized by the system is $\boxed{(a+a(b+aa)^*b)^* a(b+aa)^* a}$.

Prove that the transition system shown below (8)
accepts the set of all strings over {a, b} with an equal
# of a's & b's ; s.t. each prefix has atmost one more
a than the b's & at most one more b than a's.



∴ all assumptions for algebraic method holds, we get the following set of eq"s for q_1, q_2, q_3 & q_4.

$$q_1 = q_2 b + q_3 a + \Lambda$$

$$q_2 = q_1 a$$

$$q_3 = q_1 b$$

$$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$$

As q_1 is the final state & its eq" involves only q_2 & q_3 the q_4 eq" is redundant for our purposes. Substituting q_2 & q_3, we get :-

$$q_1 = q_1 ab + q_1 ba + \Lambda$$

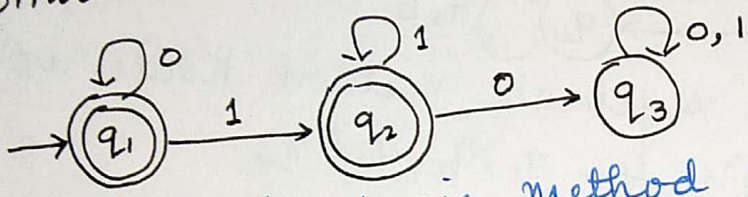
$$\Rightarrow q_1 = \Lambda (ab+ba)^* \quad \text{(Using Arden's Thm.)}$$

$$q_1 = (ab+ba)^*$$

∴ the strings accepted by the finite automata are the strings given by $(ab+ba)^*$. As any such string is a string of ab's & ba's, we get an equal # of a's & b's. If a prefix

x of a sentence accepted by the finite automaton has an even # of symbols, then it should have an equal # of a's & b's & since x is a substring formed by ab's & ba's. If prefix x has an odd number of symbols, then we can write x as ya or yb. As y has even # of symbols, y has an equal # of a's and b's. Thus, x has one more a than b or vice versa.

Ques :- Describe in english the set accepted by the finite automaton whose transition diagram as shown :-



Sol^n :- Using Algebraic method for Arden's Theorem :- all assumptions hold :- The eq^n's for q_1, q_2 & q_3 are as:

$$q_1 = q_1 0 + \Lambda$$

$$q_2 = q_1 1 + q_2 1$$

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

By using Arden's Theorem :-

$$q_1 = \Lambda 0^* = 0^*$$

$$q_2 = 0^* 1 + q_2 1 \Rightarrow q_2 = (0^* 1)1^*$$

As q_1 & q_2 are only final states; strings accepted by system is :-

$$q_1 + q_2 = 0^* + (0^* 1)1^*$$

$$= 0^*(\Lambda + 11^*) = 0^* 1^*$$ (Using $\Lambda + RR^* = R^*$)

∴ The string accepted by the finite automaton are the strings of any # of 0's (possibly Λ) followed by a string of any # of 1's (possibly Λ)