Theory of Computation

Prequisite -> Symbol - an entity / individual object ($\alpha, \beta, \gamma, A, B$)etc
alphabet - finite set of symbols $\Sigma = \{0, 1\}$
string - finite collection of symbols alphabet $\Sigma = \{a, b\}$

L_1 = { Set of all string of length 2 }
L_2 = { Set of all string ended with symbol b }
ω = { a, b, aa, aaaa, ab, ba, aba }

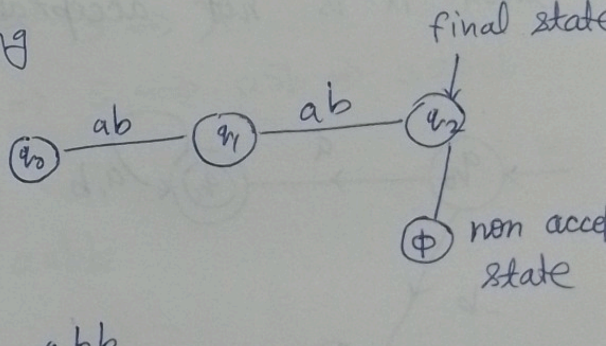
Language - collection of all strings
L_3 = { set of all string start with a }
L_4 = { set of all string in which ab appears as substring }

finite Language - { aa, bb, ab, ba }

$\left\{ \begin{array}{l} b, bb, bbb \ldots \quad \infty \\ a, ab, abb \ldots \ldots ab \end{array} \right\}$ — infinite set of string
= infinite language

{ a, aa, aabb, ... }
$\uparrow$
First symbol should always be a of any string.

ω < Σ (Name of the occurance of symbols from alphabet)
$|\omega|$ ← length of string



ω = aabaa
$|\omega| = 5$

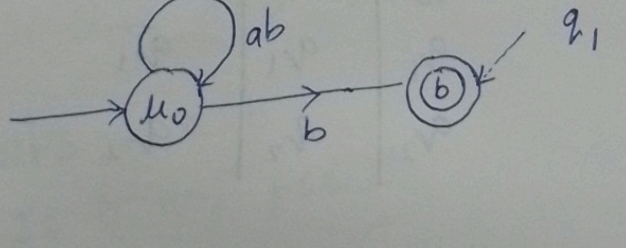


abb
abbb

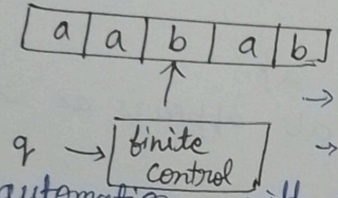
Finite Automata –

→ Used to recognize pattern on string of languages

→ it take inputs (string of symbols) & change its state accordingly.

→ During Transition, finite automata can be either moved to next state (s) or stay at same state.

I/P Symbols – finite symbol
finite control



→ either moved to next state $Q \times \Sigma \Rightarrow Q$

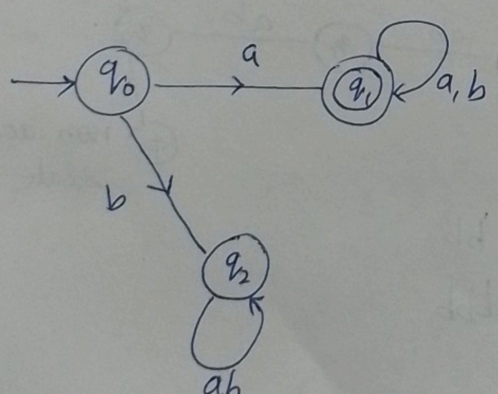
→ stay at same state

An automation with a finite number of states is called a FA. or Finite State Machine (FSM)

5 tupple $(Q, \Sigma, \delta, q_0, F)$

- Q – finite set of states
- Σ – finite set of alphabet
- δ – transition state $\delta : Q \times \Sigma \rightarrow Q$ (can be any state)
- q_0 – initial state $q_0 \in Q$
- F – final state; accepting state

Set of all string starting from a (if not start then it is not acceptable).



Transition Table

δ	I/P a	b
q_0	q_1	q_2
q_1	q_1	q_1
q_2	q_2	q_2

deterministic & non deterministic

Theory of computer Science, Automata language & computation
↳ KLP Mishra

An Introduction to formal language & Automata
↳ Peter Lenz

Language & grammar – A Phrase structure grammar or grammar is (N, T, P, S) where

Mathematical study of Language.

$G(\cancel{N}\, V, T, S, P)$

Non empty set of non-terminal

Non empty set of terminal

Start Symbol (non terminal)

Set of production Rule. $(\alpha \to \beta)$

$\to \cancel{N} \cap T = \phi$

$$\left.\begin{array}{l} S \to aSb \\ S \to \lambda \end{array}\right\} \text{Production rule}$$

Grammar – $G\{\{S\}, \{a, b\}, S, P\}$

Single Step derivation

Multi step derivation

$S \to aSb \Rightarrow ab$

start symbol

$S \to aSb \to aaSbb \to aaaSbbb$

$\Rightarrow \cdots \Rightarrow a^n S b^n \Rightarrow a^n b^n$

$\Rightarrow S \overset{*}{\Rightarrow} a^n b^n$

↳ Multistep derivation.

$\Rightarrow S \Rightarrow \lambda$

$\Rightarrow S \Rightarrow aSb \Rightarrow aaSbb$

$$L(G) = \{ w \in T^* \mid S \overset{*}{\Rightarrow} w \}$$

$S \to aSb \Rightarrow aaSbb \Rightarrow aabb$

$abb \Rightarrow aSb$

It is not possible

Sentence – A group of words that contains subject, verbs etc. & express statement or questions etc.

Syntax – Set of rules needed to ensure sentense is grammatically correct.

Sentential Form –

$A \rightarrow \alpha$

if $\alpha \in (V \cup T)^*$

then it is known as sentential form.

Semantics – If the rule has some meaning

eg. → A brown fox jumped quickly over the lazy dog.

↳ Sentactically & symantically correct

→ Cows flows spremely.

↳ semantically not correct.

identifiers in PL –

int abc ; ✓

int 8abc ; ×

int #abc ; ×

int _abc ; ✓

<Letter> <rest> / <underscore> <rest>

<rest> → <Letter> <Digits> / <Digit> <Letter> / <underscore> <rest>

<Letter> – a | b | - - - - | z | A | B | - - - Z |

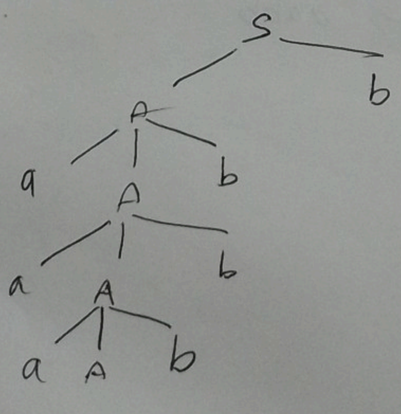
<Digit> – 0 | 1 . . . | 9

Terminals – $\{a, z\}$, $\{0, 9\}$

Suppose we define Grammar –

$$P \begin{cases} S \rightarrow Ab \\ A \rightarrow aAb \mid \lambda \end{cases}$$

$S\{\{S, A\}, \{a, b\}, S, P\}$



$L(G) = \{a^n b^{n+1} \mid n \geq 0\}$

Generated sentences –

b, abb, aabbb,

aaaabbbbb

aaabbb ×

→ $S \to SS \mid \lambda$

$S \to aSb \mid bSa$

$S \to aSb$

→ ~~ab~~ $absab$ ×

$L(G)\ \{a^n S b^{n+1} \mid n \geq 0\}$

$S \to SS$

$\Rightarrow aSbS$

$\Rightarrow abS \Rightarrow abbSa$

$\Rightarrow abba$

$\boxed{S - aabbab}$

$S \to SS$

$= aSbS$

$= aaSbbS$

$= aabbS$

$= aabbaSb$

$= aabbab$

$S \to SS$

$\Rightarrow aSbS$

$\Rightarrow abS$

$\Rightarrow abSS$

$\Rightarrow abaSbS$

$\Rightarrow ababS$

(Thus we can't generate abaaba Because no. of b is not equal to a).

Chomsky Classification of Grammar/Language (VTSP)

In General it is a type i grammar where $i = 0, 1, 2, 3$.

1. type 0 grammar (unrestricted grammar / phrase structured grammar)

Production rule $\alpha \to \beta$ where, $\alpha \in (V \cup T)^* V (V \cup T)^*$, $\beta = (V \cup T)^*$

→ It is also known as recursive enumerable language.

→ This language is accepted by Turing Machine.

eg. $bAa \to aa$, $S \to \epsilon$

2. type 1 grammar (Linear Bounded Automata) - Context sensitive grammar

Production rule $\psi_1 A \psi_2 \to \psi_1 \sigma \psi_2$

$\psi_1, \psi_2, \sigma \in (V \cup T)^*$ & $A \in V$ or, $\alpha \to \beta$

but here σ is not empty. s.t. $|\alpha| \leq |\beta|$ & $\alpha \in (V \cup T)^* V (V \cup T)^*$

$\beta \in (V \cup T)^*$

β is ϵ iff there is no start symbol(s) appears RHS in Production rule

→ The rule of the form $A \to \epsilon$ is not allowed unless A is a start symbol. It does not ~~exceed the~~ occur on the right hand side.

eg. $\underset{\alpha}{Sb} \to \underset{\beta}{b}$ } — type 0 grammar. not type 1 grammar.

$|\alpha| = 2, \ |\beta| = 1$

If g_3 is a grammar

$g_3 \subseteq g_2 \subseteq g_1 \subseteq g_0$

$L_3 \subseteq L_2 \subseteq L_1 \subseteq L_0$

$\underset{FA}{L_{RL}} \subseteq \underset{PDA}{L_{CFL}} \subseteq \underset{LBA}{L_{CSL}} \subseteq \underset{TM}{L_{REL}}$

REL - Recursive enumerable Language

CFL - Context free Language

CSL - Context Sensitive Language

RL - Regular Language

→ Acceptor of RL is Finite Automata. (FA)

→ " CFL is ~~Posh~~ Push down Automata.

→ " CSL is Linear Bounded Automata (LBA)

→ REL is Turing Machine

$L = \{a^n b^n c^n \mid n \geq 0\}$ it is only accepted by TM only.

→ Type 2 Language - Also known as context free grammar, when production rule is written as.

$A \rightarrow \alpha$

$A \rightarrow aB \mid a$

$A \rightarrow Ba \mid a$

R.H.S epsilons ϵ iff ~~if~~ there is no S present in the ~~right~~ right hand side of terminal

→ The Language generated by the grammar is recognized by a Pushdown automata. In Type 2:

→ First of all it should be Type 1.

→ The left hand side production can have only one variable & there is no restriction on β.

$\alpha \longrightarrow \beta$

$|\alpha| = 1$

e.g. $S \longrightarrow AB$

$A \longrightarrow a$

→ <u>type 3 grammar</u> – (Also known as Regular Grammar)

If the production rule $A \to \alpha$ satisfies either $A \to aB \mid a$ or, → ~~Right~~ Right -Linear Grammar

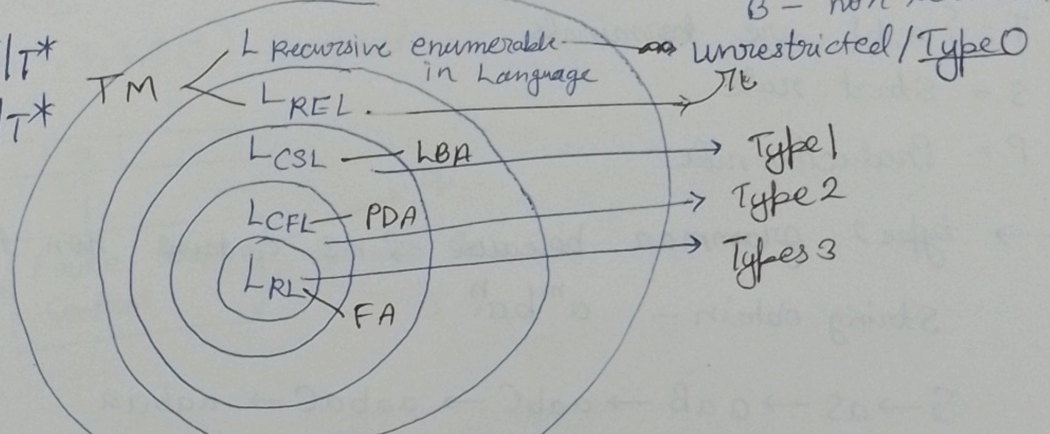
$A \to Ba \mid a$ → Left Linear Grammar.

B – non terminal

$V \to VT^* \mid T^*$

$V \to T^*V \mid T^*$

↳ extended right regular grammar



write a grammar for this Language

Suppose $\{a^m b a^n \mid m, n > 0\}$

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

can be written as $a^+ b a^+$

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

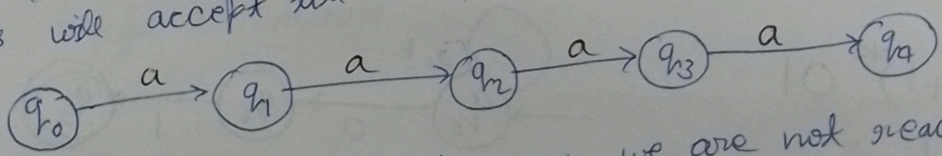
$\{ a^+ =$ positive closure
= one or more occurance of a.
= a, aa, aaa, ----

$a^* =$ zero or more occurance of a.
= ε, a, aa, aaa, ----
= Klean closure

without any string or any alphabet we reach to the final state.

eg. $L = \{a^n \mid n = 4\}$

this will accept till $L = \{a, aaa\}$

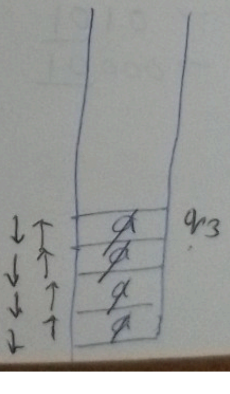


aaa is not acceptable because we are not reaching to the final state.

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

Let the string is $\{a^4 b a^4\} = \{aaaabaaaa\}$

we reach the final state



The set of Production rule -

$S \rightarrow aS$ $\quad C \rightarrow aC$

e.g. Also. $S \rightarrow aB$ $\quad C \rightarrow a$

then $B \rightarrow bC$

$V = \{S, B, C\}$ are non terminals

$T = \{a, b\}$ are terminals

S = Start rule

P = Production rule

$\rightarrow$ type 2 grammar because L.H.S contains non terminal only.

String obtain - $a^m b a^n$

$S \rightarrow aS \rightarrow aaB \rightarrow aabC \rightarrow aabaC \rightarrow aabaa$

Finite Automata

Deterministic FA

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

- Q – Non empty set of internal state
- Σ – Set of Input Alphabet
- δ – transition Function $\delta: Q \times \Sigma \rightarrow Q$
- q_0 – Initial Point
- F – Accepting state

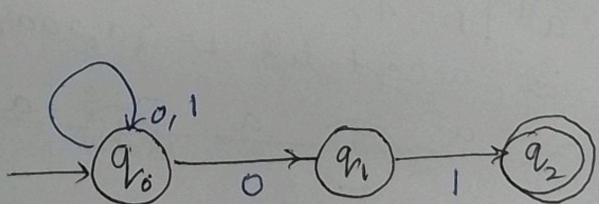
Non Deterministic FA

$(Q, \Sigma, \delta, q_0, F)$ (NDFA)

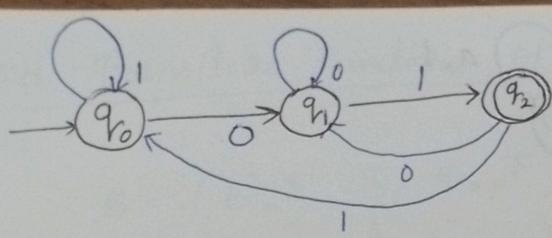
$\delta: Q \times \Sigma \rightarrow 2^Q$

e.g. $(0+1)^* 01$

$\rightarrow$ 01
$\rightarrow$ 001
$\rightarrow$ 101
$\rightarrow$ 0101
$\rightarrow$ 00001

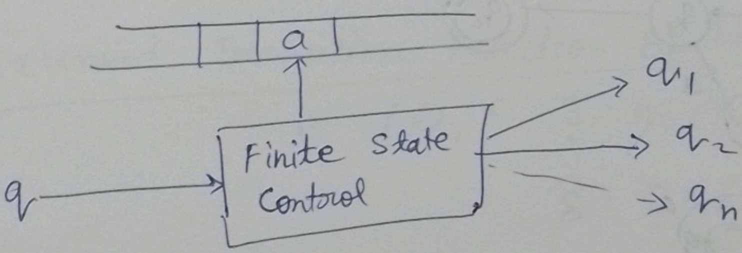


δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
q_2	ϕ	ϕ



δ	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

111010001
$q_0 q_0 q_0 q_1 q_2 q_1 q_1 q_1 q_2$ Accepted



<u>type1 Automata</u> - Any string start from some specific symbol and end by some specific symbol.
then $(n+1)$ state, n = no. of specific symbol at the end of string
eg.- $(a+b)^* abb$
state = 4

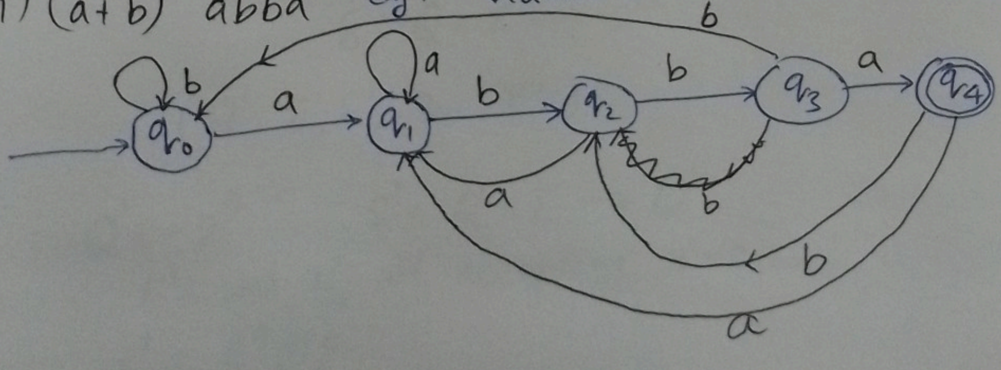
<u>type2 Automata</u> -
Any string start from some specific symbol and end with some symbol. e.g. $abb(a+b)^*$
state = 5
then $(n+2)$ states are required.

e.g. (i) $(a+b)^* abba$ ~~After that~~ (ii) $a(a+b)^*$

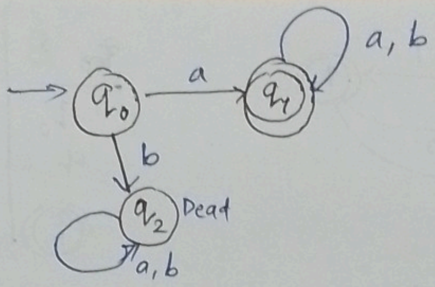
(i) $n = 4$ = no. of specific symbol. - Automata - type1

(ii) $n = 1$, - type 2. Automata
$(2+1) = 3$ state

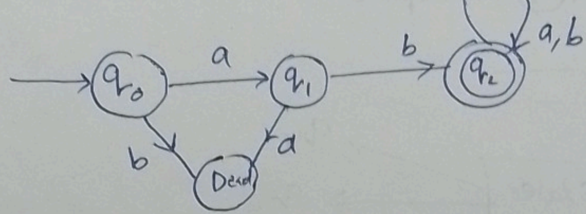
(i) $(a+b)^* abba$ eg. aabbbabba $\rightarrow q_0 q_1 q_2 q_3 q_0 q_0 q_1 q_2 q_3 q_4$



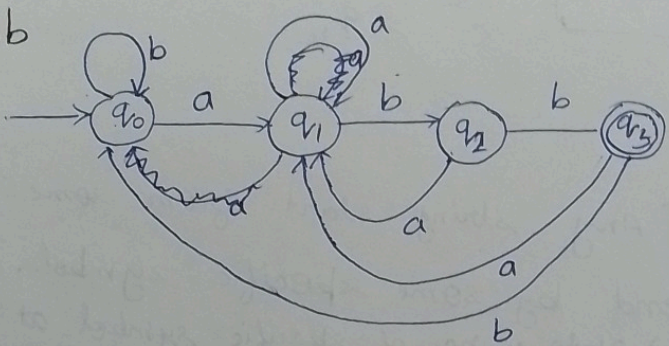
(ii) $a(a+b)^*$



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

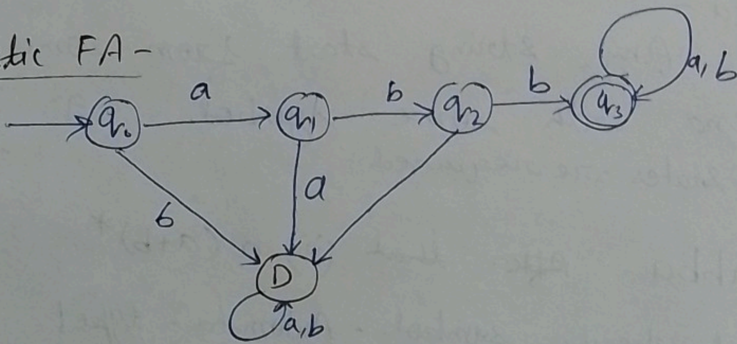


(iv) $(a+b)^* abb$

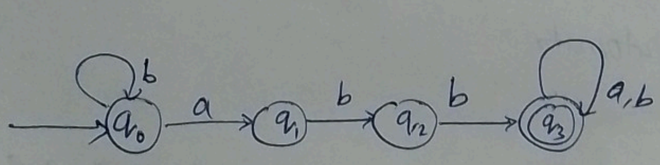


babb
aabb
ababb
abbaabb
abbbabb

Deterministic FA -



NDFA -



Extended Transition Function of DFA $-(\hat{\delta})$

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

$$\hat{\delta} : Q \times \Sigma^* \longrightarrow Q.$$

i) $\hat{\delta}(q, \varepsilon) = q$

ii) $\hat{\delta}(q, ya) = \delta(\hat{\delta}(q, y), a)$

Extended Transition Function of NDFA $-(\hat{\delta})$

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

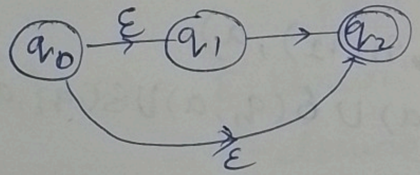
$$\hat{\delta} : Q \times \Sigma^* \longrightarrow 2^Q$$

$q \in Q$, $y \in \Sigma^*$, $a \in \Sigma$ then,

i) $\hat{\delta}(q, \varepsilon) = \{q'\}$ $\{\varepsilon$-closure of $\{q_0\}$ $\Rightarrow$ state that are at distance zero from q_0

ii) $\hat{\delta}(q, ya) = \bigcup_{r \in \hat{\delta}(q, y)} \delta(r, a)$

$\rightarrow$ ε transition not allowed in DFA allowed in NDFA.



ε closure of $q_0 = \{q_0, q_1, q_2\}$

$q_1 = \{q_2\}$

$q_2 = \{q_2\}$

$\rightarrow$ computing power of DFA is equal to NDFA. There is no difference in computing power.

Conversion of NDFA into equivalent DFA $-$

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

$$\delta : Q \times \Sigma \longrightarrow 2^Q$$

$$(Q', \Sigma, \delta', q_0, F')$$

$Q' \sim 2^Q$ $\quad \delta' : Q' \times \Sigma \rightarrow Q'$

F' = set of all accepting state

$F' = Q' \cap F$

NFA with ε -

$$\delta' : Q \times \underset{\{\Sigma \cup \varepsilon\}}{\underset{\Downarrow}{\Sigma^*}} \longrightarrow 2^Q$$

NFA without ε

$\Downarrow$

Equivalent DFA

ε-closure - $\quad q \xrightarrow{\varepsilon} p$

$\Rightarrow$ Every state is at distance zero from itself.

$\Rightarrow$ All the states that are in zero from q belongs to ε-closure.

$$\hat{\delta}(q_0, \varepsilon) = \varepsilon\text{-closure}\ \{q_0\}$$

$$\delta'(q, a) = \varepsilon\text{-closure}\,(\delta(\hat{\delta}(q, \varepsilon), a))$$

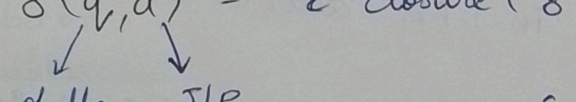
q → any of the state NFA; a → I/P alphabet

$\hat{\delta}(q, \varepsilon) \Downarrow$ ε-closure(q)

$\delta((q_0, q_1, q_2), a)$

$\Rightarrow \delta(q, a) \cup \delta(q_1, a) \cup \delta(q_2, a)$

e.g.



Starting state of NFA without ε

If there is a transition between two state q_1 & q_2 on ε.

$$q_1 \xrightarrow{\varepsilon} q_2$$

If q_1 is the starting state $\Rightarrow$ q_2 is also start.

Accepting / NFA without ε / Final state

If in any state of NFA without ε Final state at NFA with ε present then this state becomes final state.

ε-closure $\{q_0\} = \{q_0, q_1, q_2\}$

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

$\{q_2\} = \{q_2\}$

Now, $\delta'(q_0, 0) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_0, \varepsilon), 0))$

$= \varepsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 0))$

$= \varepsilon\text{-closure}(\underbrace{\delta(q_0, 0)}_{q_0} \cup \underbrace{\delta(q_1, 0)}_{\phi} \cup \underbrace{\delta(q_2, 0)}_{\phi})$

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

$\delta'(q_0, 1) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_0, \varepsilon), 1))$

$= \varepsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 1))$

$= \varepsilon\text{-closure}(\underset{\phi}{\delta(q_0, 1)} \cup \underset{q_1}{\delta(q_1, 1)} \cup \underset{\phi}{\delta(q_2, 1)})$

$= \{q_1, q_2\}$

$\delta'(q_0, 2) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_0, \varepsilon), 2))$

$= \varepsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 2))$

$= \varepsilon\text{-closure}(\underset{\phi}{\delta(q_0, 2)} \cup \underset{\phi}{\delta(q_1, 2)} \cup \underset{q_2}{\delta(q_2, 2)})$

$= \{q_2\}$

$\delta'(q_1, 0) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, \varepsilon), 0))$

$= \varepsilon\text{-closure}(\delta(\{q_1, q_2\}, 0))$

$= \varepsilon\text{-closure}(\underset{\phi}{\delta(q_1, 0)} \cup \underset{\phi}{\delta(q_2, 0)})$

$= \phi$

$\delta'(q_1, 1) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, \varepsilon), 1))$

$= \varepsilon\text{-closure}(\delta(\{q_1, q_2\}, 1))$

$= \varepsilon\text{-closure}(\underset{q_1}{\delta(q_1, 1)} \cup \underset{\phi}{\delta(q_2, 1)})$

$= \{q_1, q_2\}$

$\delta'(q_1, 2) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, \varepsilon), 2))$

$= \varepsilon\text{-closure}(\delta(\{q_1, q_2\}, 2))$

$= \varepsilon\text{-closure}(\underset{\phi}{\delta(q_1, 2)} \cup \underset{q_2}{\delta(q_2, 2)})$

$= \{q_2\}$

$\delta'(q_2, 0) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_2, \varepsilon), 0))$

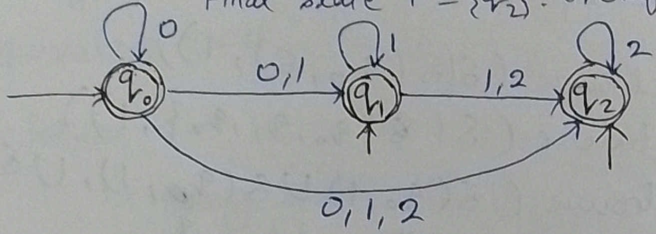
$= \varepsilon\text{-closure}(\delta(q_2, 0)) = \phi$

$\delta'(q_2, 1) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_2, \varepsilon), 1)) = \delta(q_2, 1) = \phi$

$\delta'(q_2, 2) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_2, \varepsilon), 2))$
$= \varepsilon\text{-closure}(\delta(q_2, 2))$
$= \{q_2\}$

ε-closure $\{q_0\} = \{q_0, q_1, q_2\}$ ⇒ q_0 is also final state
ε-closure $\{q_1\} = \{q_1, q_2\}$ ⇒ q_1 is also final state
ε-closure $\{q_2\} = \{q_2\}$ ⇒ q_2 is also final stat.

Final state $F = \{q_2\}$. Therefore final state is $F' = F \cup \{q_0\}$
$= q_1 q_2 \cup q_0$
$= \{q_0 q_1 q_2\}$



← NFA without ε

NFA without ε move to DFA

Minimization of DFA → Equivalence Theorem, Myphel Maroto Theorem

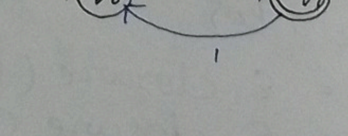
<u>Conversion of NFA into equivalent DFA –</u>

$N = (Q, \Sigma, \delta, q_0, F)$
- F → set of accepting state
- q_0 → Initial state
- δ → Transition
- Σ → Set of input alphabet
- Q → Non empty set of internal state of NFA.

$D = (Q', \Sigma, \delta', \{q_0\}, F')$
- F' → suppose S be state in Q' then if $S \cap F \neq \phi$ then S is accepting state.
- δ' → $\delta' : Q' \times \Sigma \rightarrow Q'$
- Q' → If n states are in NFA then maximum 2^n state are in DFA

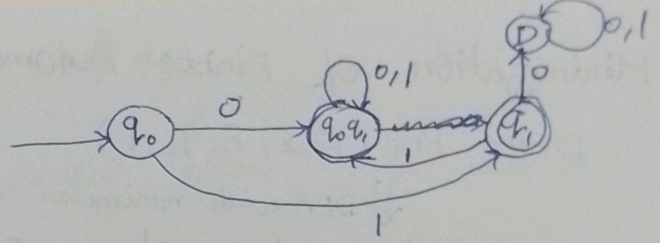
→ We start from initial state(s) of given NFA then find transition table T'.

δ	0	1
→ q_0	$\{q_0, q_1\}$	$\{q_1\}$
q_1	ϕ	$\{q_0, q_1\}$



T'

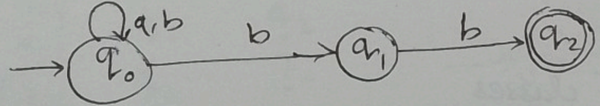
δ'	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$
$\{q_1\}$	$\{\phi\}$	$\{q_0, q_1\}$
$\{\phi\}$	$\{\phi\}$	$\{\phi\}$



$$\delta'((q_0, q_1), 0) = \bigcup_{p \in S} \delta(p, 0)$$
$$= \delta(q_0, 0) \cup \delta(q_1, 0)$$
$$= \{q_0, q_1\} \cup \phi$$
$$= \{q_0, q_1\}$$

$$\delta'((q_0, q_1), 1) = \bigcup_{p \in S} \delta(p, 1)$$
$$= \delta(q_0, 1) \cup \delta(q_1, 1)$$
$$= \{q_1\} \cup \{q_0, q_1\}$$
$$= \{q_0, q_1\}$$

$$\delta'(q_1, 0) = \delta(q_1, 0) = \phi$$

Q-→ 

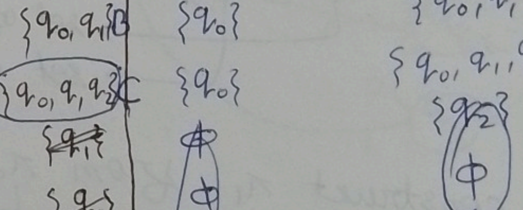
construct equivalent DFA —

δ	a	b
$\rightarrow q_0$	$\{q_0\}$	$\{q_0, q_1\}$
q_1	ϕ	$\{q_2\}$
q_2	ϕ	ϕ

NFA

δ'	a	b
$\rightarrow \{q_0\}$ A	$\{q_0\}$	$\{q_0, q_1\}$
$\{q_0, q_1\}$ B	$\{q_0\}$	$\{q_0, q_1, q_2\}$
$\{q_0, q_1, q_2\}$ C	$\{q_0\}$	$\{q_0, q_1, q_2\}$
$\cancel{\{q_1\}}$	ϕ	$\{q_2\}$
$\cancel{\{q_2\}}$	ϕ	ϕ
$\cancel{\phi}$	ϕ	ϕ

$$\delta'(\{q_0\}, a) = \delta(q_0, a) = q_0$$
$$\delta'(\{q_0\}, b) = \delta(q_0, b) = q_0 q_1$$
$$\delta'(\{q_0, q_1\}, a) = \delta(q_0, a) \cup \delta(q_1, a) = \{q_0\}$$
$$\delta'(\{q_0, q_1\}, b) = \delta(q_0, b) \cup \delta(q_1, b) = \{q_0, q_1, q_2\}$$
$$\delta(\{q_0, q_1, q_2\}, a) = \delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a) = \{q_0\}$$
$$\delta(\{q_0, q_1, q_2\}, b) = \delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_2, b) = \{q_0, q_1, q_2\}$$
$$\delta(\{q_1\}, a) = \delta(q_1, a) = \phi$$
$$\delta(\{q_2\}, a) = \delta(q_2, a) = \phi$$
$$\delta(\{q_1\}, b) = \delta(q_1, b) = \{q_2\}$$
$$\delta(\{q_2\}, b) = \delta(q_2, b) = \phi$$



$(a+b)^* bb$

Minimization of Finite Automata

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

$\Downarrow$ DFA with minimum no. of states

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

(I) Equivalence Method / Partitioning Method –

→ Two states q_1 & q_2 are equivalent

$$q_1 \equiv q_2$$

$\left.\begin{matrix}\delta(q_1, x)\\ \delta(q_2, x)\end{matrix}\right\}$ both belong to final states of DFA
or
both belong to non final states of DFA.

→ Any two final states or any two non final states are zero equivalent.

π_0 = zero equivalence classes

$= (Q^F, Q^{NF})$

Q^F → set of final states

Q^{NF} → set of non-final states

→ construct π_1 from π_0

π_0 → zero equivalence classes

π_1 → one equivalence classes.

e.g. $Q_F = \{q_0, q_3\}$

$Q_{NF} = \{q_1, q_2, q_4, q_5, q_6, q_7\}$

$\pi_0 = \{\{q_0, q_3\}, \{q_1, q_2, q_4, q_5, q_6, q_7\}\}$

π_1 from π_0 – $q_0 \overset{1}{\equiv} q_3$ one equivalence

iff $\left.\begin{matrix}\delta(q_0, x)\\ \delta(q_3, x)\end{matrix}\right\}$ both belong to Q_F
or
both belong to Q_{NF}

→ π_K (K-equivalence classes) –

two states q_1, q_2 are k equivalent $(k \geq 0)$ if both $\delta(q_1, x)$ & $\delta(q_2, x)$ are in Q_i^{K-1} where Q_i^{K-1} subset in $\pi_{\ldots}$

If $\left.\begin{array}{l}\delta(q_1, x) \in Q_i^{k-1} \\ \delta(q_2, x) \in Q_j^{k-1}\end{array}\right\}$ $q_1 \not\equiv^{k} q_2$

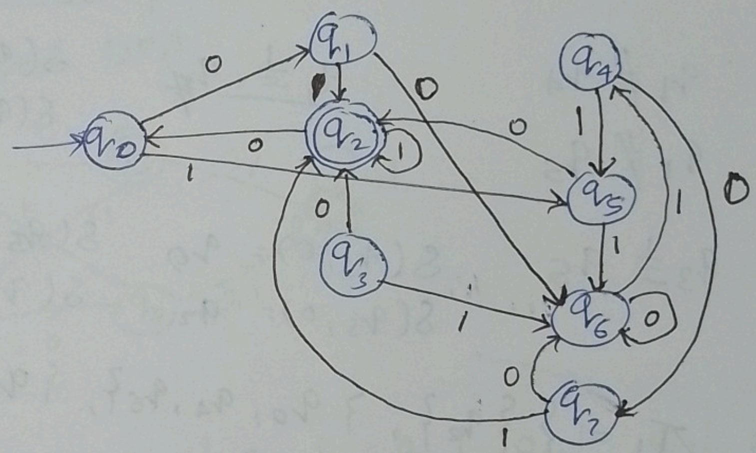
→ If q_1 & q_2 are $(k+1)$ equivalent then it is also k-equivalent.

→ If any subset of π_n contains initial states then this is also a initial states.

→ If any subset of π_n contains final states then this is also a final states.

e.g.- Suppose transition table of DFA is given.

δ	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
q_2	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2



$\pi_0 = \{ Q^F, Q^{NF} \}$

$= \{\{q_2\}, \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}\}$

π_1 from π_0 –

$\underline{q_0 \overset{1}{\equiv} q_1}$ $\quad \delta(q_0, 0) = q_1 \quad \delta(q_0, 1) = q_5$
$\quad \delta(q_1, 0) = q_6 \quad \delta(q_1, 1) = q_2$ } belong to different set

Thus $q_0 \not\equiv q_1$

$\underline{q_0 \overset{1}{\equiv} q_3}$ $\quad \delta(q_0, 0) = q_1 \quad \delta(q_0, 1) = q_5$
$\quad \delta(q_3, 0) = q_2 \quad \delta(q_3, 1) = q_6$

Thus $\therefore q_0 \not\equiv q_3$

$\boxed{q_0 \overset{1}{\equiv} q_4}$ $\quad \delta(q_0, 0) = q_1 \quad \delta(q_4, 1) = q_5$
$\quad \delta(q_4, 0) = q_7 \quad \delta(q_4, 1) = q_5$

Thus, $q_0 \overset{1}{\equiv} q_4$

$\underline{q_0 \overset{1}{\equiv} q_5}$ $\quad \delta(q_0, 0) = q_1 \quad \delta(q_0, 1)\ q_5$
$\quad \delta(q_5, 0) = q_2 \quad \delta(q_5, 0) = q_6$

$q_0 \not\equiv q_5$

$q_0 \overset{1}{\equiv} q_6$ $\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_5$

$\delta(q_6, 0) = q_6$ $\delta(q_6, 1) = q_4$

Thus, $\boxed{q_0 \overset{1}{\equiv} q_6}$

$q_0 \overset{1}{\equiv} q_7$ $\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_5$

$\delta(q_7, 1) = q_6$ $\delta(q_7, 1) = q_2$

Thus $q_0 \overset{1}{\not\equiv} q_7$

$q_0 \overset{1}{\equiv} q_4 \overset{1}{\equiv} q_6$, $q_1 \overset{1}{\equiv} q_7$, $q_3 \overset{1}{\equiv} q_5$

$q_1 \overset{1}{\not\equiv} q_3$ $\delta(q_1, 0) = q_6$ $\delta(q_1, 1) = q_2$

$\delta(q_3, 0) = q_2$ $\delta(q_3, 0) = q_6$

$q_1 \overset{1}{\not\equiv} q_4$ $q_1 \overset{1}{\equiv} q_7$ $\delta(q_1, 0) = q_6$ $\delta(q_1, 1) = q_2$

$\delta(q_7, 0) = q_6$ $\delta(q_7, 1) = q_2$

$q_1 \overset{1}{\not\equiv} q_5$

$q_3 \overset{1}{\equiv} q_5$ $\delta(q_3, 0) = q_2$ $\delta(q_3, 1) = q_6$

$\delta(q_5, 0) = q_2$ $\delta(q_5, 1) = q_6$

$\pi_1 = \{\{q_2\}, \{q_0, q_4, q_6\}, \{q_1, q_7\}, \{q_3, q_5\}\}$

π_2 from π_1 $q_0 \overset{2}{\not\equiv} q_4$ ✓ $\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_5$

$\delta(q_4, 0) = q_7$ $\delta(q_4, 1) = q_5$

$q_0 \overset{2}{\not\equiv} q_6$

$q_1 \overset{2}{\equiv} q_7$

$q_3 \overset{2}{\equiv} q_5$

$\pi_2 = \{\{q_2\}, \{q_0, q_4\}, \{q_6\}, \{q_1\}, \{q_7\}, \{q_3\}, \{q_5\}\}$

$q_0 \overset{3}{\not\equiv} q_4$

$\therefore \pi_3 = \pi_2$

q_0 is initial state so $\{q_0, q_4\}$ is initial state.

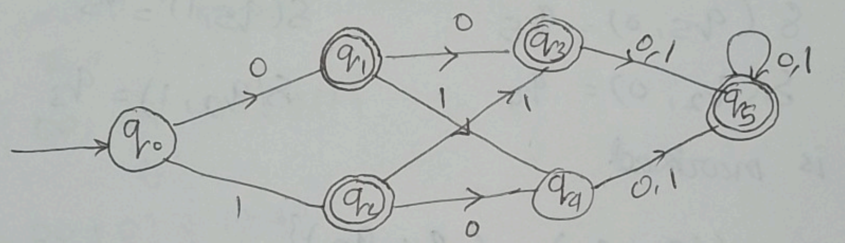
Myhill Nerode Theorem (Table Filling Method) -

Steps – Create a pairs of all state involved in DFA then simply mark all the pairs (Q_a, Q_b) s.t. $Q_a \in F$, $Q_b \in NF$.
↳ non final state.

→ If any unmaked pairs (Q_a, Q_b) s.t. $\delta(Q_a, x)$ & $\delta(Q_b, x)$ are marked then marked (Q_a, Q_b) where x is an input symbol. And repeat this step until no more marking can be made.

→ Combine all the unmarked pairs & make them a single state in the minimized DFA.

e.g. -



δ	0	1
→ q_0	q_1	q_2
q_1	q_3	q_4
q_2	q_4	q_3
q_3	q_5	q_5
q_4	q_5	q_5
q_5	q_5	q_5

	q_0	q_1	q_2	q_3	q_4	q_5
q_0						
q_1	✓					
q_2	✓					
q_3	✓✓	✓	✓			
q_4	✓✓	✓	✓			
q_5	✓	✓✓	✓✓	✓	✓	

$F = \{q_1, q_2, q_5\}$
$NF = \{q_0, q_3, q_4\}$

First unmarked pair is $\{q_2, q_1\}$ we check $\delta(q_2, 0) = q_4$
$\delta(q_1, 0) = q_3$, $\delta(q_2, 1) = q_3$ $\delta(q_1, 1) = q_4$
Now, $\{q_2, q_1\}$ is unmarked,
check $\{q_3, q_0\}$ pair - $\delta(q_3, 0) = q_5$ $\delta(q_3, 1) = q_5$
$\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_2$
(q_5, q_1) & (q_5, q_2) is unmarked so no change in table.

For, (q_4, q_0) - $\delta(q_4, 0) = q_5$ $\delta(q_4, 1) = q_5$

$\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_2$

(q_4, q_0) is unmarked, no change in table.

For, (q_4, q_3) - $\delta(q_4, 0) = q_5$ $\delta(q_4, 1) = q_5$

$\delta(q_3, 0) = q_5$ $\delta(q_3, 1) = q_5$

(q_4, q_3) is unmarked.

For (q_5, q_1) - $\delta(q_5, 0) = q_5$ $\delta(q_5, 1) = q_5$

$\delta(q_1, 0) = q_3$ $\delta(q_1, 1) = q_4$

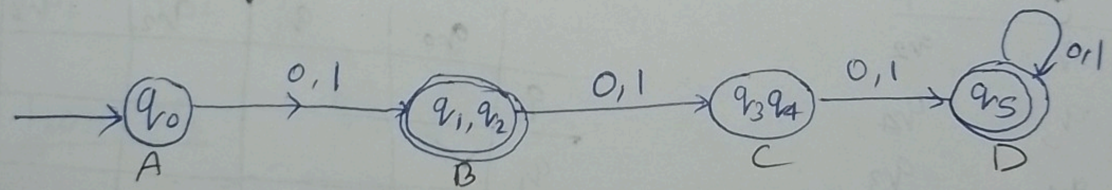
(q_5, q_3) & (q_5, q_4) both are marked. So marked the pair (q_5, q_1).

For (q_5, q_2) = $\delta(q_5, 0) = q_5$ $\delta(q_5, 1) = q_5$

$\delta(q_2, 0) = q_4$ $\delta(q_2, 1) = q_3$

$\therefore$ (q_5, q_2) is marked

Unmarked pairs = $\{(q_2, q_1), (q_4, q_3)\}$



	0	1
q_0	$\{q_1, q_2\}$	$\{q_1, q_2\}$
$q_1 q_2$	$\{q_3, q_4\}$	$\{q_3 q_4\}$
$q_3 q_4$	$\{q_5\}$	$\{q_5\}$
q_5	$\{q_5\}$	$\{q_5\}$

From equivalence -

$\pi_0 = [\{q_1, q_2, q_5\}, \{q_0, q_3, q_4\}]$

$\pi_1 =$ $q_1 \overset{1}{\equiv} q_2$, $q_1 \overset{1}{\not\equiv} q_5$, $\{\{q_1, q_2\}, \{q_5\}, \{q_0, q_3, q_4\}$

$q_0 \overset{1}{\equiv} q_3$ $q_0 \overset{1}{\equiv} q_4$

$q_0 \equiv q_3 \equiv q_4$

$\pi_2 = \quad q_1 \overset{2}{\equiv} q_2 \quad q_0 \overset{2}{\not\equiv} q_3 \quad q_0 \overset{2}{\not\equiv} q_4$

$q_3 \overset{2}{\equiv} q_4 \quad \{ (q_1, q_2), \{q_5\}, \{q_0\}, \{q_3, q_4\} \}$

$\pi_3 = \{ \{q_1, q_2\}, \{q_5\}, \{q_0\}, \{q_3\ q_4\} \}$

Regular Expression

→ This is used to represent certain set of strings in the form of algebraic equation.

→ Any set that represents the value of regular ~~automata~~ expression is called Regular set.

R.E
↓
NFA with ε
↓
NFA without ε
↓
DFA
↓
minimized DFA.

If $\Sigma = \{0, 1\}$

$L = \{00, 01, 10, 11\}$

$\Downarrow$

$= 00 + 01 + 10 + 11$

$= 0(0+1) + 1(0+1)$

$\boxed{R = (0+1)(0+1)}$ → Regular expression

Rule for R.E. –

i) ϕ is a regular expression denoted by set $\{\phi\}$

ii) ε is a regular expression denoted by set $\{\varepsilon\}$

iii) for each $a \in \Sigma$, a is a regular expression denoted by set $\{a\}$

Σ ↑ set of alphabet

iv) If r & s are regular expression denoted by set R & S then, $r+s$ is r.e. denoted by set $R \cup S$

concatenation → $r \cdot s$ is r.e. denoted by set ~~RES~~ R.S

$R \cup S = \{x \mid x \in R \text{ or } x \in S\}$

$R \cdot S = \{xy \mid \in R \ \& \ y \in S\}$

v) if r is r.e. denoted by set R then r^* is also a r.e. denoted by set $\{R^+\}$

vi) Only those expression consisting of symbols, Parenthesis & operator(+,.) are r.e. which are ~~ob~~tained by finite many applications of above rule.

$\{1, 11, 111, 1111, \ldots\}$ → regular set

regular exp. = 1^* → value of regular exp.

$\{1, 11, 111, 1111, \ldots\}$

r.e. = 1^+ (closure when empty symbol is not included

→ Set of all strings over $\Sigma = \{0, 1\}$ ending with 01.

= $(0+1)^* 01$ → r.e.

= $\{01, 001, 101, 1101, 10001 \ldots\}$ → regular set

→ Set of all strings begin from 0 & end with 1.

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

→ Regular set $L = \{abb, b, a, bba\}$

$R = abb + b + a + bba$

$\{ ab \neq ba$
$a+b = b+a$

→ Find R set for the given set of input $L = \{1, 111, 11111, 1111111\}$.

= $1 + 111 + 11111 + 1111111 + \ldots = 1[\lambda + 11 + 1111 + 111111 + \ldots]$

= $1(11)^*$

→ $L = \{0, 1, 2\}$

RE = $0 + 1 + 2$

Identities for Regular Expression -

Let ϕ is the set & R is RE. then,

$\phi + R = R$

$\phi R + R\phi = \phi$

$\phi R = R\phi = \phi$

$\lambda R = R\lambda = R$

ϕ is the identity for union/addition (+)

λ is an identity for concatenation (·)

$\lambda^* = \lambda$

$\phi^* = \lambda$

$R + R = R$

$R^* \cdot R^* = R^*$

$R^* R = R R^*$

$(R^*)^* = R^*$

$\lambda + R^* R = \lambda + R R^* = R^*$

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

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

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

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

Ardens Theorem for Regular Expression –

If P & Q be a r.e. over set of Σ & P does not contain λ. Then following equation in R like

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

if $\Rightarrow QP^*$ is a sol^n

$\Rightarrow$ Unique sol

Proof –

$$R = Q + RP \quad \text{—— (1)}$$

replace R.H.S. R by QP^* in eq^n (1)

$$R = Q + QP^* P$$
$$= Q(\lambda + P^* P) \qquad \{ \text{by identity } \lambda + R^* R = \lambda + R R^* = R^*$$
$$\Rightarrow R = QP^*$$

This proves that $R = QR^*$ is a solution of eq^n (1)

Uniqueness – $\quad R = Q + RP$

R in right side of eq^n is replaced by $R = Q + RP$

$$R = Q + (Q + RP)P$$
$$= Q + QP + RP^2$$
$$= Q + QP + (Q+RP)P^2$$
$$= Q + QP + QP^2 + RP^3$$
$$= Q + QP + QP^2 + QP^3 + RP^4$$
$$\vdots$$

$$R = Q + QP + QP^2 + QP^3 + \cdots + QP^i + RP^{i+1}$$
$$= Q[\lambda + P + P^2 + P^3 + \cdots + P^i] + RP^{i+1} \quad \text{—②}$$

If R satisfy eqn ① then this also satisfy eqn ②

Let w be string of length i in R.

⇒ w also belong to eqn ②

⇒ w belong to the set $Q(\lambda + P + P^2 + \cdots + P^i) + RP^{i+1}$

⇒ if P does not contains λ, RP^{i+1} has no string of length less than (i+1).

⇒ $w \notin RP^{i+1}$

⇒ $w \in Q(\lambda + P + P^2 + \cdots P^i)$

⇒ $w \in QP^*$

Let us assume that w is in the set QP^*

⇒ w is in the set QP^k set $k \geq 0$

⇒ w is in the set $Q(\lambda + P + P^2 + \cdots + P^k)$

⇒ w is in R.H.S. of eqn ②

⇒ w is in R.

⇒ This represents R & QP^* have same set.

1→ Prove that RE representing the set L of the string in which each zero is immidiately followed by atleast two one's is same as r.e.,

$$R = a + 1^*(011)^*(1^*(011)^*)^*$$

→ $R = (1+011)^*$ ~~eg~~ set = {1, 011, 1011011, ...}

$R = \lambda + \underline{1^*(011)^*}(1^*(011)^*)^*$

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

$= P^*$ (by identity $\lambda + RR^* = R^*$

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

$= (r^*s^*)^*$ (where $r = 1$, $s = 011$

$= (r + s)^*$ $\{\because (r^*s^*)^* = (r + s)^*$

$= (1 + 011)^*$

2) Show that RE $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$
$= 0^*1(0+10^*1)^*$

L.H.S,

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

$= (1+00^*1)[\lambda + (0+10^*1)^*(0+10^*1)]$

$= (1+00^*1)(0+10^*1)^*$ $\quad$ { By R.E. Identity $\lambda + PP^* = P^*$
$\Rightarrow \lambda + (0+10^*1)(0+10^*1)^* = (0+10^*1)^*$

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

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

RE into NFA to DFA — DFA to regular

Transition system with λ-moves to transition System without λ-moves -

Suppose,

$$q_1 \xrightarrow{\lambda} q_2$$

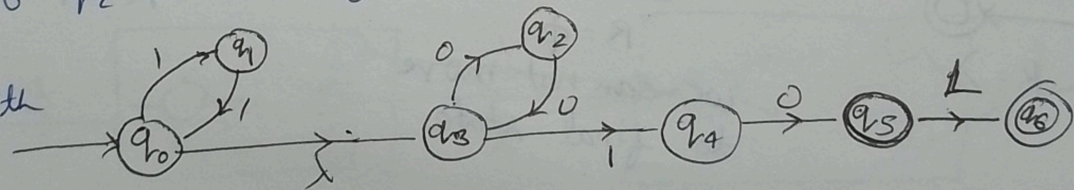
(I) Find All the edges starting from q_2

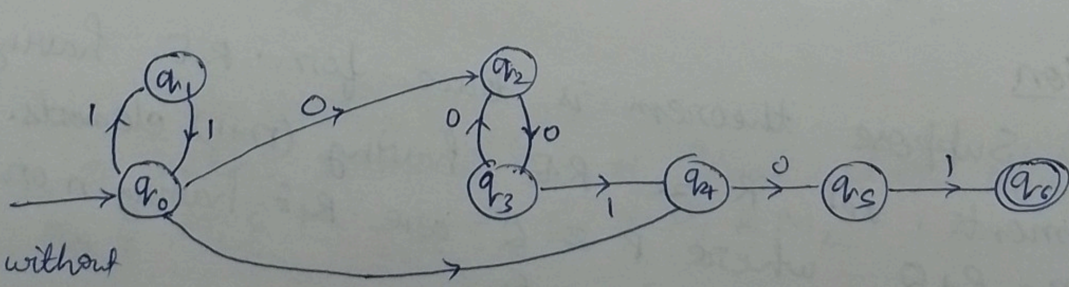
(II) Duplicate all these edges starting from q_1 without changing the edge level.

(III) If q_1 is the ~~final~~ initial state, make q_2 as initial state

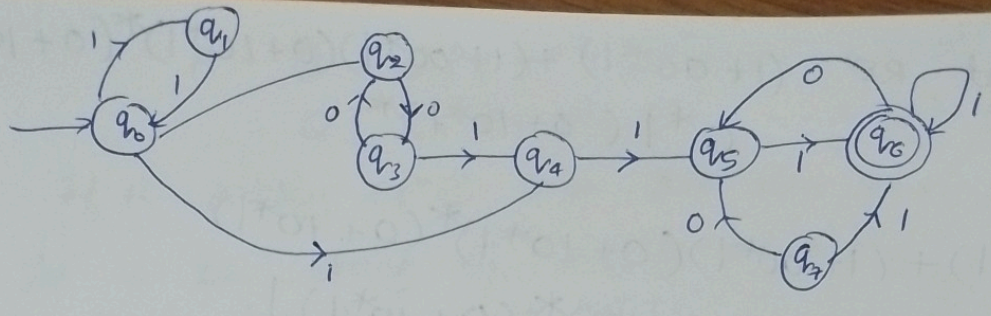
(IV) If q_2 is the final state, make q_2 as final state.

NFA with ε-moves





NFA without ε-moves



Every R.E. is recognized by a NDFA with λ moves s.t.

$$L = T(M)$$

Any R.E. consist of following $\Sigma, \lambda, \phi, *, \cdot, +$

(Product: $\cdot$; closure: $*$; add: $+$)

$R = \lambda + 01^*(1+10)^*$

$\Sigma = \{0,1\}$

λ = Null string

$*$ - closure

~~Let~~ $L(R)$ denotes the set representing R.

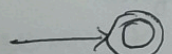
1→ If R has only single element

$R = \lambda$ or $R = \phi$ or $R = a_i$, where $a_i \in \Sigma$

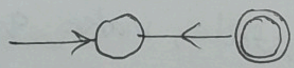
⇓

equivalent transition system.

~~Then~~,

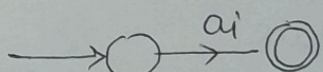


$R = \lambda$



$R = \phi$

we can not move to final state



$R = a_i$

Induction

Suppose theorem is true for R.E. having n elements. Let R be R.E. having (n+1) elements.

1- $R = P + Q$ where P & Q are R.E. having n or less elements.

2- $R = P \cdot Q$

3- $R = P^*$

$L(P) = T(M_1)$ (M_1 = NDFA with ε moves

$L(Q) = T(M_2)$ (M_2 = NDFA of R.E. Q.

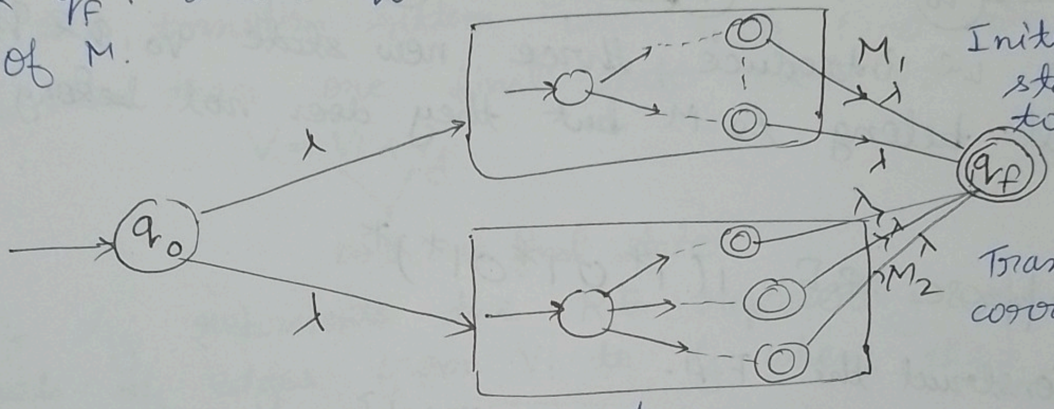
case I :- $R = P + Q$ —

Let, $L(P) = T(M_1)$ (M_1 transition state corresponding to P)

$L(Q) = T(M_2)$

$L(R) = T(M)$

Here, we introduce two new state in M. i.e., q_0 & q_F. where q_0 is initial state of M & q_F final state of M.

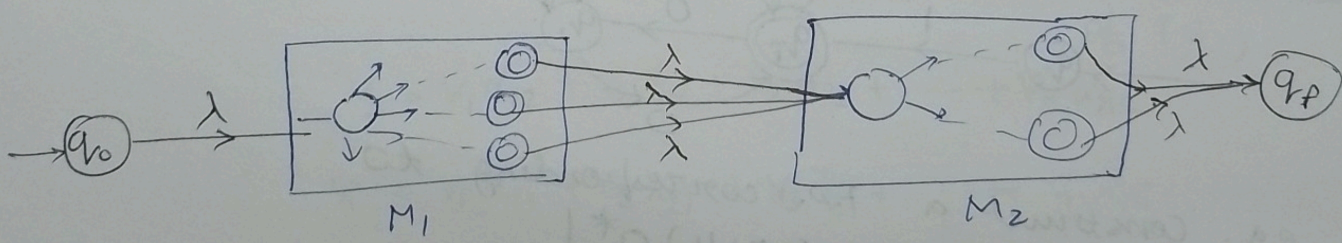


Initial transition state corresponding to M_1

Transition state corresponding to M_2

$q_0 \notin M_1$ and M_2 & $q_F \notin M_1$ & M_2

case II $R = P.Q$ —

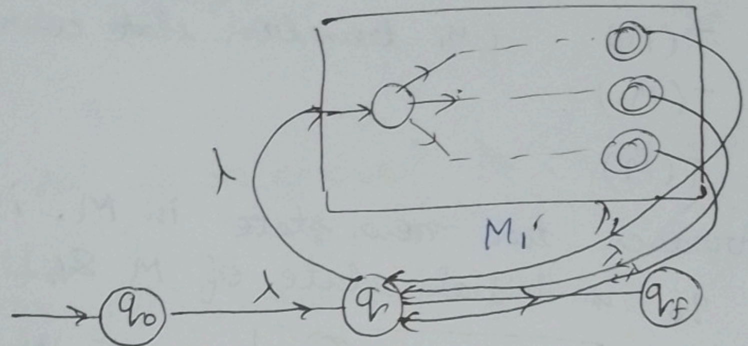


q_0 is initial state & q_f is final state of M & $q_0 \notin M_1$ & M_2 and $q_f \notin M_1$ & M_2

Case III :- $R = P^*$ -

where P is r.E.

$L(P) = T(M_1)$ & $L(R) = T(M)$

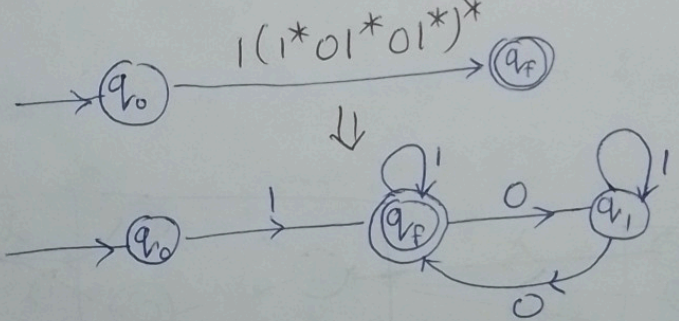


Here we introduce three new state q_0, q & q_f and all belong to M but they does not belong to M_1

Ex- Suppose R.E. $1(1^*01^*01^*)^*$

Construct the F.A.

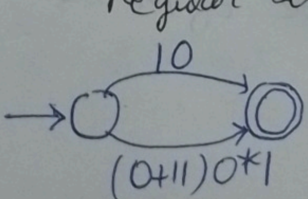
Regular set - $\{1, 100, 11001, 110101\}$

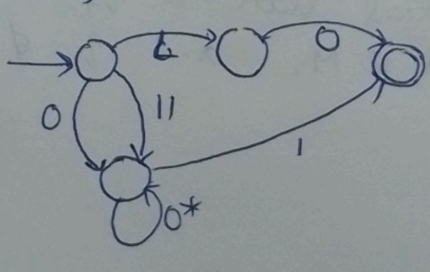


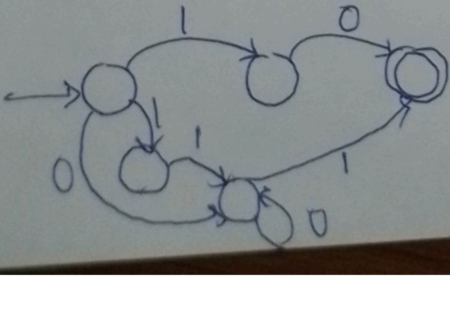
eg. Construct a F.A. corresponding to

R.E. $10 + (0+11)0^*1$

Regular Set - $\{10, 01, 111, 001, 1101 \ldots$







Regular Expression from Transition System [Arden's Theorem & R.E. identities] – Algebraic Method –

Assumptions –

1- The transition System (T.S.) does not contain λ moves.

2- The transition system has only one initial states

3- No. of Vertices/states $V_1, V_2 \ldots V_i, \ldots V_n$

initial state (Only one) — V_1

Final state (Can be more than one) — V_i

4- V_i is the R.E. representing the set of Accepted string by the transition system when V_i is Final state.

if more than one final state are in T.S.

$$V = V_i + V_j$$

both are final state.

5- If α_{ij} represents the R.E. representing the set of labels of edges from V_i to V_j when there is no such edge than $\boxed{\alpha_{ij} = \phi}$

T.S. is converted to Algebraic eqn which is of the form –

$$V_1 = V_1\alpha_{11} + V_2\alpha_{21} + V_3\alpha_{31} + \cdots + V_n\alpha_{n1} + \lambda$$

(only when V_1 is the initial state of T.S.)

α_{11} is the transition from V_1 to V_1

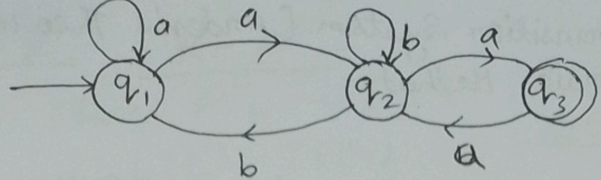
$$V_2 = V_1\alpha_{12} + V_2\alpha_{22} + V_3\alpha_{32} + \cdots + V_n\alpha_{n2}$$

$$\vdots$$

$$V_i = V_1\alpha_{1i} + V_2\alpha_{2i} + V_3\alpha_{3i} + \cdots + V_n\alpha_{ni}$$

$$\vdots$$

$$V_n = V_1\alpha_{1n} + V_2\alpha_{2n} + V_3\alpha_{3n} + \cdots + V_n\alpha_{nn}$$

1→ 

$$q_1 = \lambda + q_1 a + q_2 b \quad \text{— ①}$$

$$q_2 = q_1 a + q_2 b + q_3 a \quad \text{— ②}$$

$$q_3 = q_2 a \quad \text{— ③}$$

replace the value of q_3 from eqⁿ ② to eqⁿ ①

$$q_2 = q_1 a + q_2 b + q_2 aa$$

$$\underbrace{q_2}_{R} = \underbrace{q_1 a}_{Q} + \underbrace{q_2}_{R} \underbrace{(b+aa)}_{P}$$

$$q_2 = q_1 a (b+aa)^*$$

(By Arden's theorem, if $R = Q + RP$ $\Rightarrow R = QP^*$

Now, replace the value of q_2 from eqⁿ ①.

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

$$\Rightarrow \underbrace{q_1}_{R} = \underbrace{\lambda}_{Q} + \underbrace{q_1}_{R} \underbrace{[a + a(b+aa)^* b]}_{P}$$

$$\Rightarrow q_1 = \lambda [a + a(b+aa)^* b]^* \quad \text{(By Arden's theorem)}$$

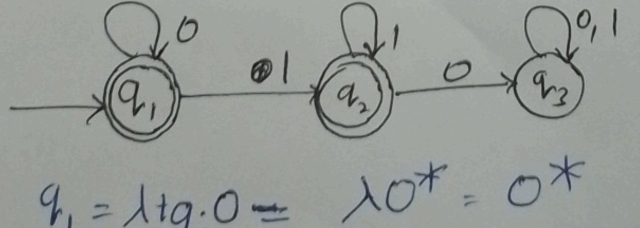
$$= [a + a(b+aa)^* b]^*$$

∴ The Final state

$$q_3 = q_2 a$$

$$= q_1 a (b+aa)^* a$$

$$= [a + a(b+aa)^*]^* a (b+aa)^* a$$

2→ 

$$q_1 = \lambda + q_1 0 = \lambda 0^* = 0^* \quad \text{(By Arden's law)}$$

$$q_2 = q_1 1 + q_2 1$$

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

Our objective is to find R.E. corresponding to q_1 & q_2 As both are final state.

$$\underset{R}{q_2} = \underset{Q}{\underline{0^*1}} + \underset{R}{q_2}\underset{P}{1}$$

$$= 0^*11^*$$

The final state $q = q_1 + q_2$

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

$$= 0^*(\lambda + 11^*)$$

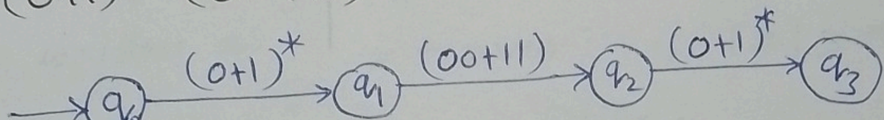
$$= \cancel{0^*(11^*)^*} \qquad (\because \lambda + RR^* = R^*)$$

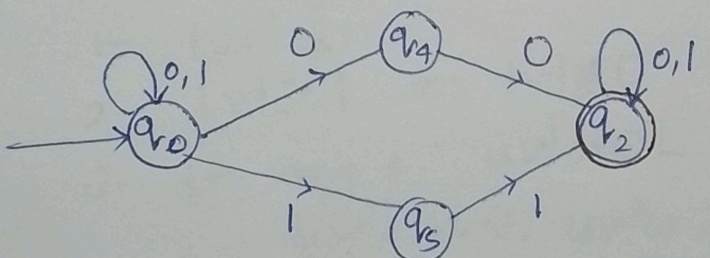
$$= 0^*1^*$$

If RE is given then how to find F.A.

(Finite Automata from R.E.)

1→ $(0+1)^*(00+11)(0+1)^*$





NFA without-ε.

$$q_0 = q_0(0+1) + \lambda$$

$$q_4 = q_0 0$$

$$q_5 = q_0 1$$

$$q_2 = q_4 0 + q_5 1 + q_2(0+1)$$

δ	0	1
$\rightarrow q_0$	$q_0 q_4$	$q_0 q_5$
q_4	q_2	—
q_5	—	q_2
q_2	q_2	q_2

δ	0	1
A $\{q_0\}$	$\{q_0 q_4\}$ B	$\{q_0, q_5\}$ C
$\{q_0, q_4\}$ B	$\{q_0, q_2, q_4\}$ D	$\{q_0, q_5\}$ C
C $\{q_0, q_5\}$	$\{q_0, q_4\}$ B	$\{q_0, q_2, q_5\}$ E
D $\{q_0, q_2, q_4\}$	$\{q_0, q_2, q_4\}$ D	$\{q_0, q_2, q_5\}$ E
E $\{q_0, q_2, q_5\}$	$\{q_0, q_2, q_4\}$ D	$\{q_0 q_2 q_5\}$ E

$\pi_0 = \{\{A, B, C\}, \{D, E\}\}$

π_1, $A \overset{1}{\not\equiv} B$

$A \overset{1}{\not\equiv} C$

$B \overset{1}{\not\equiv} C$

$\pi_1 = \{\{A\}, \{B\}, \{C\}, \{D, E\}\}$

δ	0	1
$\rightarrow A$	B	C
B	{DE}	C
C	B	{DE}
{DE}	{DE}	{DE}

Pumping Lemma for regular Language –

→ The non regularity of certain language.

Language – Finite (regular)

Language – Infinite $L = \{a^p \mid p \text{ is prime no}\}$

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

$L = \{0^n 1^n \mid n \geq 0\}$

$L = \{ww^R \mid w \in 0,1\}$

Pumping Lemma is used to determine that these infinite language are ~~not~~ not regular.

→ If L is regular set then $\exists$ a particular integer $n \geq 1$ (pumping length) such that every string x of L is written as $x = uvw$, $|x| \geq n$

This satisfy the condition –

1 – $|uv| \leq n$

2 – $|v| \geq 1$ or $|v| > 0$

3 – for any i, $uv^i w \in L$, $i = 0, 1, 2, \ldots$

$uw, uvw, uv^2w, uv^3w \ldots uv^kw \ldots \in L$

eg – Let us assume that $L = \{a^p \mid p \text{ is Prime}\}$ be a regular language then $\exists$ a positive integer $n > 0$ s.t.

$|x| = P \geq n$

Let, $x = uvw$ s.t. $|uv| \leq n$, $|v| \geq 1$, $uv^i w \in L$

Let $x = aaa$ ($u = a$, $v = a$, $w = a$) $|uv| = |aa| = 2 \leq 3$

$|v| = |1| = 1 \geq 1$

Also, if $i = 0$ then $uv^0w = uw = aa \in L$

if $i = 1$ then $uvw = aaa \in L$

if $i = 2$ then $uv^2w = aaaa \notin L$ which is contradiction of the theorem means it does not follow for any i, $uv^i w \in L$

Q. Let us assume that $L = \{a^n b^{2n} \mid n \geq 0\}$

$|x| = |n+2n| = 3n \geq n$

Let, $x = uvw$, s.t. $|uv| \leq n$, $|v| \geq 1$, $uv^iw \in L, \forall i$

Let $x = \underline{a}\,\underline{a}\,\underline{bbbb}$ $(n=2)$

$u \quad v \quad w$

$|uv| = |aa| = 2 \leq 2$

$|v| = |a| = 1 \geq 1$

if $i=0$, $\quad uv^0w = abbbb \notin L$

$\therefore$ The given set is not regular language.

Q. Let us take $L = \{a^{n^2} \mid n \geq 0\}$ a regular set.

$|x| = |n^2| = n^2 \geq n$

Then there exists a positive integer n s.t. the string $|x| \geq n$.

s.t. $x = uvw$.

$|v| \geq 1$

$|uv| \leq n$

$uv^iw \in L$

if $n = 2$

then $x = \underline{a}\,\underline{a}\,\underline{aa}$

$u \quad v \quad w$

$|v| = |a| = 1 \geq 1$

& $|uv| = |aa| = 2 \leq 2$

& if $i=0$ $\quad uv^0w = aa \notin L$

Or, $\quad |uvw| = n^2$

if $i=2$

$|uv^2w| = |uvw| + |v|$

$= n^2 + m$

$\leq n^2 + n \quad$ (when $m \leq n$)

$< n^2 + n + n + 1$

$\cancel{\text{b}}\ |uvw| < |uv^2w| < (n+1)^2$

$n^2 < |uv^2w| < (n+1)^2$

<u>Proof</u> Let L be a regular set then there exists a finite automata $m = (Q, \Sigma, \delta, q_0, F)$ s.t. $L(M) = L$.

Put $n = |Q|$

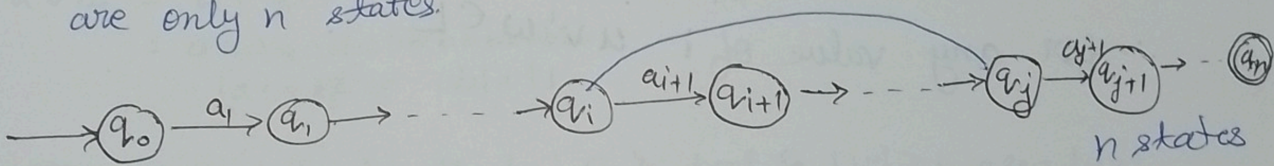
Let x be string $x = a_1 a_2 \cdots a_m$ where $a_i \in \Sigma$

$|x| = |m| \geq n$

~~where~~ Let $\delta(q_0, a_1), q_1$, $\delta(q_1, a_2) = q_2 \cdots \delta(q_{n-1}, a_n) = q_n$

$\delta^*(q_0, a_1, a_2 \cdots a_i) = q_i$

consider the state $(q_0, q_1, \cdots q_n)$
there are $(n+1)$ in no. but there are only n states.



$q_i = q_j$

Put $a_1, a_2 \cdots a_j = u$

$a_{i+1} \cdots a_j = v$

$a_{j+1} \cdots a_m = w$

$\Rightarrow uvw = a_1 a_2 \cdots a_i a_{i+1} \cdots a_j a_{j+1} \cdots a_m$

$|uv| \leq n$

$|v| \geq 1 \qquad j-i$, s.t. $j > i$

for any i, $uv^iw \in L$

Let us assume that $i = 0$

$uv^0w = uw$

$(q_0, uv^0w) = (q_0, uw)$

$= (q_i, w)$

$= (q_j, w)$

$= \cancel{(q_n, w)}\ q_n \in F$

if $i=1$ $\quad uvw \in L$

if $i \geq 2$

$$
\begin{aligned}
(q_0, uv^2w) &= (q_i, v^i w) \\
&= (q_i, v.v^{i-1}w) \\
&= (q_j, v^{i-1}w) \\
&= (q_j, v.v^{i-2}w) \\
&= (q_i, v.v^{i-2}w) \\
&= (q_j, v^{i-2}w) \\
&\vdots \\
&= (q_j, v^0 w) \\
&= (q_j, w) \\
&= q_n \in F
\end{aligned}
$$

$\therefore$ for any value of i $\; uv^iw \in L$

Using Pumping Lemma Prove that the Language

$A = \{a^n b^n \mid n \geq 0\}$ is not regular.

~~Proof~~ - Assume that A is regular. And Pumping length = n

$S = a^n b^n \Rightarrow$ if $n = 7$ then, $S = aaaaaaabbbbbbb$

$\underset{uvw}{S} = a^n b^n$

Case I - If v is in 'a' part & $|v| \geq 1$. Also $|uv| = 6 \leq n = 7$

$\underbrace{a}_{u}\underbrace{aaaaa}_{v}\underbrace{abbbbbbb}_{w}$

$uv^iw = uv^2w$ (Let $i = 2$)

$= aaaaaa\,aaaaa\,bbbbbbb$

$|a| = 11 \neq$ & $|b| = 7$

Case II - If v is in 'b' part & $|v| \geq 1$ Also $|uv| = 13 \nleq n = 7$

$\underbrace{aaaaaaab}_{u}\underbrace{bbbb}_{v}\underbrace{b}_{w}$

$uv^iw = uv^2w$

$= aaaaaaabbbbbbbbbbb$

$|a| = 7 \neq |b| = 11$

Case III - If v is in the 'a' & 'b' part & $|v| \geq 1$ Also $|uv| = 9 \nleq n = 7$

$\underbrace{aaaaaaa}_{u}\underbrace{abb}_{v}\underbrace{bbbbb}_{w}$

$uv^iw = uv^2w$

$= aaaaaaaabbaabbbbbbb$ not following $a^n b^n$ pattern

It is not following $a^n b^n$ pattern

It is not following a^nb^n pattern $uv^iw \notin A$

Thus we have shown that

Using Pumping Lemma prove that $A = \{yy \mid y \in \{0,1\}^*\}$

is not regular -

Assume that A is regular Language. Then it must have pumping Length n.

$S = 0^p 1 0^p 1$ if $P = 7 \Rightarrow S = \underbrace{00}_{u}\underbrace{0000}_{v}\underbrace{0100000001}_{w}$

$\underset{u\ v\ w}{S}$

Now, $|v| = 4 \geq 0$

& $|uv| = 6 \leq 7$

$$\& \quad uv^iw = \cancel{aa}\, uv^2w$$

$$= \underbrace{00000000000}_{11}1\underbrace{0000000}_{7}01 \notin A$$

$$11 \neq 7$$

Hence it is contradict to our assumption.

Push Down Automata – (PDA) –

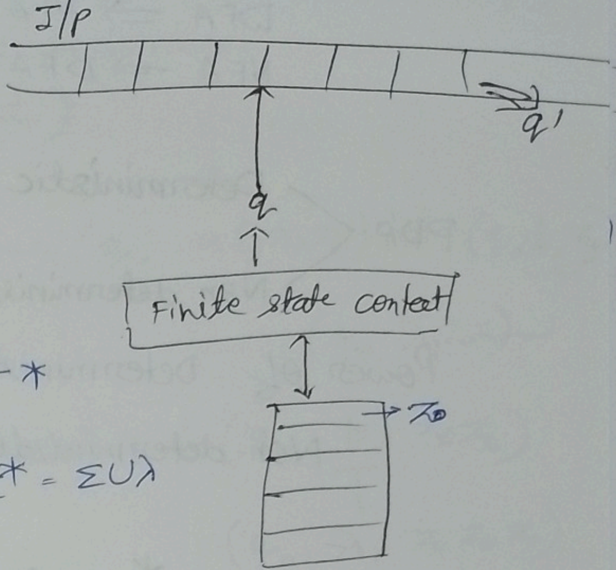
→ Context free grammar.

→ δ is defined as –

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

↳ Transition of NFA

Σ^* is set of I/P alphabet including null string (ϵ)



→ Here δ' is defined as –

$$\delta' : Q \times (\Sigma \cup \lambda) \times \Gamma \longrightarrow Q' \times \Gamma^*$$

Q = set of Internal state & $\Sigma^* = \Sigma \cup \lambda$

Γ = set of symbol on stack

$\Gamma^* = \{\Gamma \cup \lambda\}$

Σ = is set of I/P alphabet including null string (ϵ).

→ Mathematically defined by 7 tuples.

$$(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

Q – Non empty set of internal states

Σ – set of I/P Alphabet

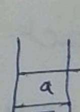
Γ – set of symbols on K₀ stack

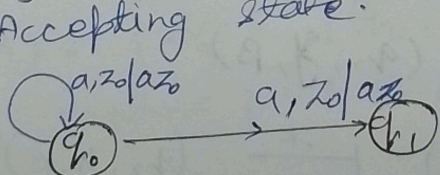
δ – transition function.

q_0 – Initial state.

Z_0 – Special symbol, initial ~~top~~ most symbol of the state

F – Set of Accepting state.





Instantaneous description (ID's) can be defined as triplets

$$(q_0, \omega, \alpha)$$

q_0 = current states

ω = remaining symbols of string

α = stack contain in which top symbol is at left.

$(q_0, a, Z_0) \longrightarrow \{(q_0, aZ_0), (q_1, aZ_0)\}$ (new state are more than one state)

Non deterministic PDA

Power of DFA = Power of NFA } FA
DFA ⇒ NFA
NFA ⇒ DFA

PDA < Deterministic PDA / Non deterministic PDA

Power of Deterministic ≠ Non-deterministic PDA

Non-deterministic power > Deterministic PDA

$(q, w, \alpha) \vdash^{*} (q', \lambda, \gamma)$ → stack content

turn style notation — Null string

$(q_0\ a\ Z_0) \vdash (q_0, aZ_0)$

↑ single move

<u>Properties for instantaneous Description</u> -

1) If $(q_1, x, \alpha) \vdash^{*} (q_2, \lambda, \beta)$

then for every $y \in \Sigma^*$, x can be $a_1, a_2, \ldots a_n$

α = stack content can be aabbz₀

$$(q_1\ xy, \alpha) \vdash^{*} (q_2, y, \beta)$$

Conversely if $(q_1, xy, \alpha) \vdash^{*} (q_2, y, \beta)$

then for some $y \in \Sigma^*$

$$(q_1, x, \alpha) \vdash^{*} (q_2, \lambda, \beta)$$

2) If $(q, x, \alpha) \vdash^{*} (q', \lambda, \gamma)$

then for every $\beta \in \Gamma^*$

$$(q, x, \alpha\beta) \vdash^{*} (q', \lambda, \gamma\beta)$$

Converse part of this is not true means.

if $(q, x, \alpha\beta \vdash^{*} (q', \lambda, \gamma\beta)$ then $(q, x, \alpha) \vdash^{*} (q', \lambda, \gamma)$ is not true always

Q. PDA $M\{\{q_0\}, \{a, b\}, \{Z_0\}, \delta, q_0, Z_0, \phi\}$
where δ is defined as $\delta(q_0\ a\ Z_0) = \{(q_0\ \lambda)\}$
λ - mean pop operations are performed.
$\delta(q_0, b, Z_0) = (q_0, Z_0 Z_0)$
$(q_0, aab, Z_0 Z_0 Z_0 Z_0)$

Sol -
$\alpha = Z_0 Z_0 Z_0$
$\beta = Z_0$
$\gamma =$ ~~aab~~ $Z_0 Z_0$

(q_0, aab, α)
$(q_0, aab, Z_0 Z_0 Z_0) \vdash (q_0, ab, Z_0 Z_0) \vdash (q_0, b, Z_0)$
$\vdash (q_0, \lambda, Z_0 Z_0)$

$(q_0, aab, Z_0 Z_0 Z_0 Z_0) \vdash (q_0, ab, Z_0 Z_0 Z_0) \vdash (q_0, b, Z_0 Z_0)$
$\vdash (q_0, \lambda, Z_0 Z_0 Z_0)$
$(q_0, aab, \alpha\beta)$

Language Acceptance By PDA - Three Types of PDA -
→ 1. Language Acceptance by Final state
→ 2. Language Acceptance by Null stack
→ 3. Language Acceptance by Final state & Null ~~stack~~

Suppose given PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0\ F)$
$\delta: Q \times \{\Sigma \cup \lambda\} \times \Gamma \longrightarrow Q \times \Gamma^*$
Language Accepted by final state is defined as -

$$L(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (p, \lambda, w),$$
where $p \in F$ & $w \in \Gamma^*$

λ → Nothing in string
w → stack

Language Accepted by Null stack is defined as -

$$N(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (p, \lambda, \lambda)$$
where $p \in Q$

→ Nothing is remaining on stack.

Language Accepted by Final & Null stack is

$$T(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (p, \lambda, \lambda)$$

where $p \in F$

e.g. $M = \{(q_0, q_1, q_f), (a, b, c), (A, B, Z_0), \delta, q_0, Z_0, q_f\}$

where δ is defined as - $\delta(q_0, a, Z_0) = \{(q_0, AZ_0)\}$

$\delta(q_0, a, A) = \{(q_0, AA)\}$

$\delta(q_0, b, Z_0) = \{(q_0, BZ_0)\}$

$\delta(q_0, b, B) = \{(q_0, BB)\}$

$\delta(q_0, a, B) = \{(q_0, AB)\}$

$\delta(q_0, b, A) = \{(q_0, BA)\}$

$\delta(q_0, c, A) = \{(q_1, A)\}$

$\delta(q_0, c, B) = \{(q_1, B)\}$

$\delta(q_1, a, A) = \{(q_1, \lambda)\}$

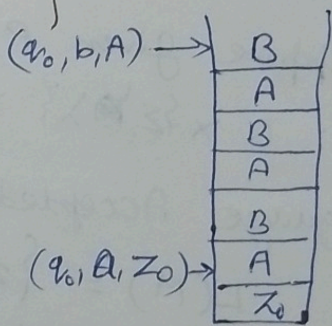
$\delta(q_1, b, B) = \{(q_1, \lambda)\}$

$\delta(q_1, \lambda, Z_0) = \{(q_f, Z_0)\}$

$\delta(q_0, c, Z_0) = \{(q_1, Z_0)\}$

$$\left.\begin{array}{l} \delta(q_1, a, B) = \phi \\ \delta(q_1, b, A) = \phi \\ \delta(q_1, c, Z_0) = \phi \\ \delta(q_1, a, Z_0) = \phi \\ \delta(q_1, b, Z_0) = \phi \end{array}\right\} \text{These are not defined.}$$

Let $L = \{WCW^R \mid W \in (a, b)^*\}$



Let $L =$ ababab$C$bababa

If $(q_0, c, B) \longrightarrow (q_1, B)$

~~q_1, b, P~~

$(q_0, cbababa, B) \vdash (q_1, babbaba, B) \vdash (q_1, abab a, ABABAZ_0)$

~~q_1, baba, BABAZ0~~ $\vdash$ ~~(q_1, ababa, ABABZ0)~~ $\vdash (q_1, baba, BABAZ_0)$

$(q_1, a, AZ_0) \vdash (q_1, ba, BAZ_0) \vdash (q_1, aba, ABAZ_0)$

$(q_1, \lambda, Z_0) \vdash (q_f, Z_0)$

→ The given PDA is deterministic.

Deterministic PDA -

Any PDA $M = \{Q, \Sigma, \sqrt{}, \delta, q_0, z_0, F\}$ is said to be Deterministic if for each $a \in \{\Sigma \cup \lambda\}$, $q \in Q$ & $z \in \sqrt{}^*$

1- $\delta(q, a, Z)$ is either empty or singleton set.

2- if $\delta(q, \lambda, Z) \neq \phi \Rightarrow \delta(q, a, Z) = \phi$ for each $a \in \Sigma$.

→ Acceptance by Final state

→ Acceptance by Null state

→ Acceptance by Final state & Null state

$M = (Q, \Sigma, \delta, \sqrt{}, q_0, Z_0, F)$

Z_0 → Top most symbol of stack

F → set of accepting state

δ: $Q \times \Sigma^* \times \sqrt{} \to Q \times \sqrt{}$

$L(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (q_F, \lambda, w)\}$

w → w is any string present on the stack.

→ Acceptance by Final state.

$N(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (p, \lambda, \lambda)\}$

where p is any internal state.

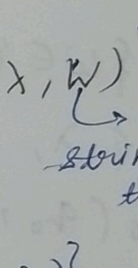
→ Acceptance by Null state

$T(M) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (p, \lambda, \lambda)\}$

$p \in F$

e.g. $L = \{w c w^T \mid w \in (a, b)^*\}$

let L = ababcbaba



$\delta(q_1, a, a) = \{(q_1, \lambda)\}$

→ If top most is a

$\delta(q_1, b, b) = \{(q_1, \lambda)\}$

$\delta(q_1, \lambda, z) = (q_f, z)$

$\delta(q_0, c, z) = (q_1, z)$

$\delta(q_0, a, Z_0) = \{(q_0, aZ_0)\}$

$\delta(q_0, b, Z_0) = \{(q_0, bZ_0)\}$

$\delta(q_0, a, a) = \{(q_0, aa)\}$

$\delta(q_0, a, b) = \{(q_0, ab)\}$

$\delta(q_0, b, a) = \{(q_0, ba)\}$

$\delta(q_0, b, b) = \{(q_0, bb)\}$

$\delta(q_0, c, a) = \{(q_1, a)\}$

$\delta(q_0, c, b) = \{(q_1, b)\}$

$\delta(q_1, a, a) = \{(q_1, \lambda)\}$

$\delta(q_1, b, b) = \{(q_1, \lambda)\}$

$\delta(q_1, \lambda, Z_0) = \{(q_f, Z_0)\}$

If $\delta(q_1, a, b)$ then this string is not accepted

Q $L = \{w \in \{a, b\}^* \mid n_a(w) = n_b(w)\}$ constract PDA

$L = \{\lambda, ab, aabb, abab, ababab, aaabbb \dots\}$

$\delta(q_0, \lambda, Z_0) = \{(q_f, Z_0)\}$

$\delta(q_0, a, Z_0) = \{(q_0, aZ_0)\}$

$\delta(q_0, b, Z_0) = \{(q_0, bZ_0)\}$

$\delta(q_0, \underline{a}bbaab, Z_0) \vdash (q_0, bbaab, aZ_0)^{(Push)}$

$\vdash (q_0, baab, Z_0)^{(Pop)}$

$\vdash (q_0, aab, bZ_0)$ (push

$\vdash (q_0, ab, Z_0)^{pop}$

$\vdash (q_0, b, aZ_0)^{push}$

$\vdash (q_0, \lambda, Z_0)^{pop} \rightarrow$ Final

$\delta(q_0, a, a) = \delta(q_0, aa)$

$\delta(q_0, b, a) = \delta(q_0, \lambda)$

$\delta(q_0, a, b) = \delta(q_0, \lambda)$

$\delta(q_0, b, b) = \delta(q_0, bb)$

$M = \{Q, \Sigma, \delta, \Gamma, q_0, Z_0, f\}$

$= \{(q_0, q_f), \{a, b\}, \{a, b\}, \delta(q_0), Z_0, \{q_f\}\}$