symbol = a, b, c, 1, 2, 3

alphabet = {0, 1}, {a, b}

string = 01, 10, 110, 1230,

language = {01, 10, 110, 1110, 110}

Power of sigma = Σ^n = set of length n strings of Σ = {0, 1}

$\Sigma^0 = \{\epsilon\}$ set of strings length 0 denoted by ϵ epsilon

$\Sigma^1 = \{1, 2\}$, $\Sigma^2 = \{01, 10, 01, 11\}$

$\Sigma^3 = \{000, 101, 110, 011, 001, 101, 010, 001, 010, 100, 111\}$ = 9

cardinality = no of strings (or) elements per set

Cardinality
$\Sigma^n = 2^n$

~~PDFA~~ Σ^* → (kleen closore) → set of all strings of all possible length.

ϵ^+ → all possible strings except (ϵ)

$\Sigma^* = \epsilon^+ + \Sigma^0$
$\Sigma^+ = \Sigma^* - \Sigma^0$

DFA / FSM → $\{Q, \Sigma, \delta, q_0, F\}$ (states, inputs, transition, initial, final)

~~NDFA~~ → ~~only one path~~

DFA	NDFA
→ no empty string transition	→ can use empty set as transition
→ difficult to construct	→ easy to construct
→ All DFA are NDFA	→ No all NFA or NDFA
$Q \times \Sigma \to Q$ next possible state	$Q \times \Sigma \to 2^{\wedge Q}$ next possible states
→ only one next state → used in lexical analysis of compiler	→ many next states

Regular language :-

a language that is recognizes by FSM

① memory is very limited

② cannot store strings

③ FSM cannot count strings like (a^n, b^m)

~~not~~ Regular language :-

→ language which is not recognized FSM

→ language, which require memory

Operations of Regular or Regular

union (∪): $\{a, r\} \cup \{t, v\} = \{a, r, t, v\}$ → regular

intersection (∩): $\{a^m b^n, a b^m\} \cap \{a^m b^n, b^n a^m\} = \{a^m b^n\}$ (regular ∩ regular = regular)

concatenation (·): $\{P, Q\} \cdot \{R, S\} = \{PR, PS, QR, QS\}$ (regular · regular = regular)

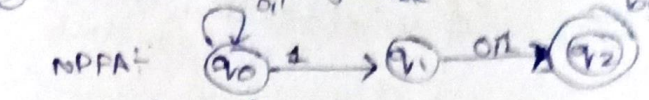
NDFA to DFA

① write transition table

② write final state later, for new states also write transition table by using union

③ combine state according to table

NDFA :-

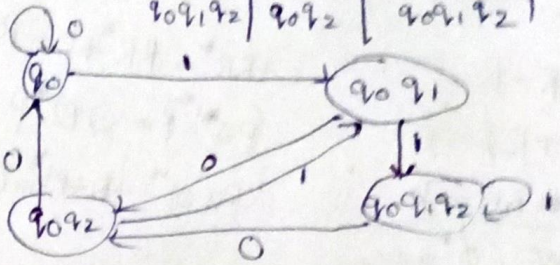


①

	0	1
q_0	q_0	$q_0 q_1$
q_1	q_2	q_2
q_2	ϕ	ϕ

②

	0	1
q_0	q_0	$q_0 q_1$
$q_0 q_1$	$q_0 q_1$	$q_0 q_2$
$q_0 q_2$	q_0	$q_0 q_1$
$q_0 q_1 q_2$	$q_0 q_1$	$q_0 q_1 q_2$



~~Minimization~~ ~~DFA to NDFA~~ : minimization of DFA

① Draw the transmission table

② check each group with other group, same means keep it same, different means separate

③ until $\pi_n = \pi_{n+1}$

given	a	b
q_0	q_1	q_2
q_1	q_3	q_4
q_2 (final state)	q_3	q_5
q_3 (final state)	q_3	q_1
q_4 (final state)	q_4	q_5
q_5 (final state)	q_5	q_4

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

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

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

$\lambda(t) = \lambda(q_t), x_t)$

moore machine :-

6 tuples (q_0/a) given n(inputs) = n+1 outputs

$m = \{Q, \Sigma, q_0, \Delta, \delta, \lambda\}$

input	0	1	output

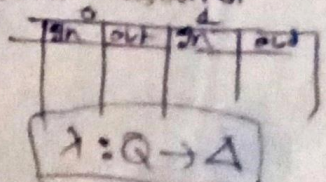
input initial state also considered as output

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

mealy machine

every arc (or) transition has output

n(input) = n(output)



Positive closure (Σ^+)

Kleen closure (Σ^*)

$\lambda(q_t) = \lambda(q_t)$

moore	mealy
output only depends upon present state	output depends upon both present state & present input
	value of output current state changes at clock edge
	less no of states (less hardware)
	input → direct output

mealy $\xrightarrow{\text{no of states}\uparrow}$ moore

limitation of FSM

① limited memory
② string without comparision
③ linear power

for NFA

$$Q \times \Sigma = 2^Q$$

for ϵ-NFA:

$$Q \times \Sigma \cup \Sigma = 2^Q$$

$\phi + R = R$

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

$\epsilon R = R\epsilon = R$

$\epsilon^* = \epsilon, \ \phi^* = \epsilon$

$R + R = R$

$R^* R^* = R^*$

$R R^* = R^* R$

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

$\epsilon + RR^* = RR^* + \epsilon = RR^*$

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

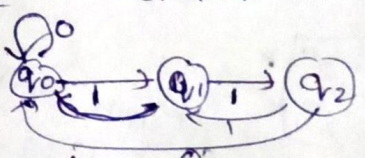
$(P+Q)^* = (Q+P)^* = (P^* Q^*)^*$

Conversion of moore to mealy:

① write states

Q B C

② draw transimition according to moore table & input



③ write the destination state output for current transition.

Conversion of mealy to moore

draw transition table of mealy

② check the input state & output if they are differ means seperate states

	a	b
q_0	$q_{3,0}$	$q_{1,0}$
q_1	$q_{0,1}$	$q_{3,0}$
q_2	$q_{2,1}$	$q_{2,0}$
q_2	$q_{1,0}$	$q_{0,1}$

here checking q_0

$(q_{0,0}, q_{0,1})$ same so don't differ

$(q_{1,0}, q_{1,1})$ differ so split

q_{10} q_{11}

replace q_{11} for entire q_1 and

q_{11} means q_1 $q_{21} \rightarrow q_2 \cup q_1$

$q_{11} \rightarrow$ makes output 1. q_{10} output 0

To find the equivalance

$\delta(A, B) = (q_i, q_j) = (q_1, q_1)$ (or)
FS FS

If L is content free lan, L' (Complement of L) is not content free

If L is regular language, then L' is also regular

L is recursive language, then L' is also regular

There is a unique minimal DFA. for every regular language

every Nfa converted into

ever content free is recursive