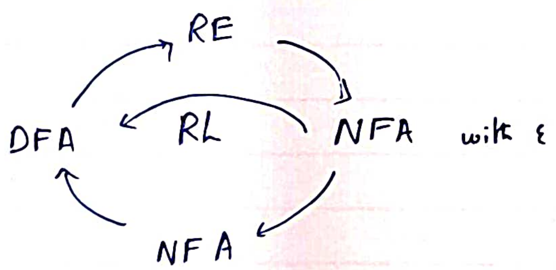
Language contains strings in dictionary

strings → english → alphabet

Regular Language – pattern

- Regular Expression
- Finite Automata (graph)
  - Non-Deterministic
    - with ε
      - string of length 0
    - without ε
  - Deterministic



Minimization Algorithm → to reduce parameters from conversion.

Moore | Mealy , pumping Lemma.

→ Proof → 1. Deduction $H \rightarrow C$
2. Contradiction $\sim H \rightarrow \sim C$
3. Contrapositive $\sim C \rightarrow \sim H$
4. Induction

now, $S \rightarrow \{\ \} \rightarrow \infty$

cardinality, $|S| = \infty$

$T \cup \overline{T} = S$

$|T| + |\overline{T}| = |S|$

$n + |\overline{T}| = \infty \rightarrow \infty$ } deduction.

→ Structural Induction.

Exp: (E)

or E op G

Base: a, (a), (E)

Ind: E or $G \rightarrow E_1$

(E_1)

→ Regular Expression

or – operator $+$, $|$ Union | dot, → $\cdot$ concatenation | $*$ → kleen closure; $+$ → +ve closure. as superscript

$a, 0, 1, b$
↘ just 1 string

L_1 : $a \Rightarrow \{a\}$

L_2 : $a+b \Rightarrow \{a, b\}$

$a+a \Rightarrow \{a\}$

$b+a \Rightarrow \{b, a\}$

L_3 : $a.b \Rightarrow \{ab\}$

$a.b \Rightarrow \{ba\}$

$a.a \Rightarrow \{aa\}$

} strings of length 2

L_4 : $a+b\cdot a \Rightarrow \{a, ba\}$

$a\cdot b+a \not\Rightarrow \{ab, a\}$

$(a+b)\cdot a \Rightarrow \{aa, ba\}$

$a\cdot(b+a) \Rightarrow \{ab, aa\}$

L_5 : $a\cdot b + b\cdot a \Rightarrow \{ab, ba\}$

$(a\cdot b)+(b\cdot a)$

$L^* = \bigcup_{i=0}^{\infty} L^i$ (L repeated i times)

$a^* = a^0, a^1, a^2, \dots$

$= \epsilon, a, aa, aaa, \dots$

Precendence

$(\,)\,, */+\,, \cdot\,, +$

$L = (a+b)^*$

$= (a+b)^0 = \epsilon$

$= (a+b)^1 = a+b$

$= (a+b)^2 = ab + ba + aa + bb$

$L_6 = \cancel{a+b^*}\,a$

$a+b^*\cdot a$

$= \{a, ba, bba, \dots\}$

$= (\epsilon + b^*)a$

$= b^*a$

[A-Z] → range

([A-Z] | [a-z]) ([A-Z] | [a-z] | [0-9] | -)*

Q. write RE starts with a, 4^{th} is b, ends with c over a, b, c

* smallest possible len is 5 here a _ _ b c

$$a \cdot (a+b+c)(a+b+c) \cdot b \, (a+b+c)^* \cdot c$$

→ RE for gmail

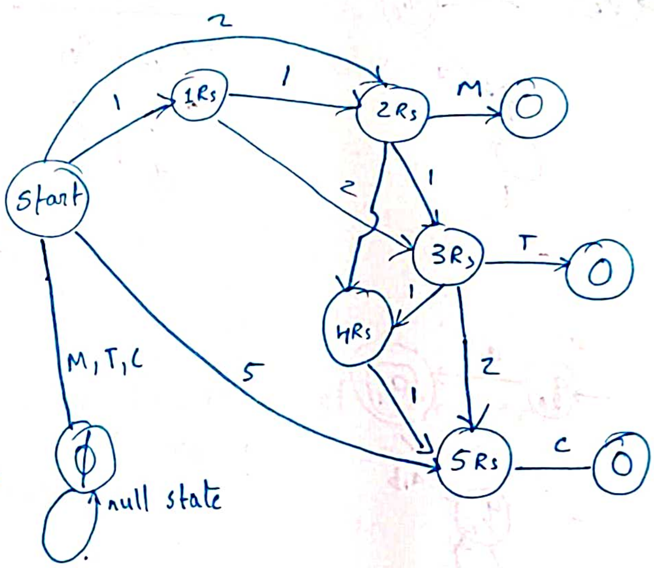
([A-Z] | [a-z] | [0-9]) ([A-Z] | [a-z] | [0-9] | .)* @gmail

→ Vending M/C

Milk – 2
Tea – 3
Coffee – 5

11 M
2 M
1 2 T
1 1 1 T
2 1 T
2 1 2 C
⋮
1 1 1 1 C

} strings



$\{1, 2, 5, C, M, T\}$ → alphabets

Finite Automata

↳ due to finite no. of states (not finite no. of strings)

- Can contain more than one final state
- From every state, on every input symbol, there is exactly 1 transition → DFA.
  ↳ lang acceptor.

Automata

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

Σ → alphabets

q_0 → start state

F → set of final states $(F \subseteq Q)$ $(q_0 \subseteq Q)$

Q → finite set of states

$$\boxed{\delta : Q \times \Sigma \text{ to } Q}$$

string is valid if it takes from start to end state.

$$L_{DFA} = \{ w \mid \delta^{\wedge}(q_0, w) = r \subseteq F \}$$

L_{DFA} ↙ under assumption that DFA cannot change state on ϵ → $\boxed{\delta(q, \epsilon) = q}$

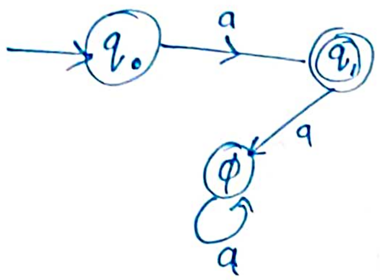
$\delta^{\wedge}(q_0, w)$ ↓ exhausted every input.

$\underbrace{u, v, w, x, y, z}_{\text{strings}}$

$\underbrace{a, b, c, d, 0, 1}_{I/P}$

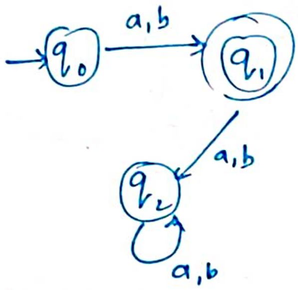
$\underbrace{p, q, r, A, B, C, D}_{\text{state}}$.

Eg: accept only 1 a



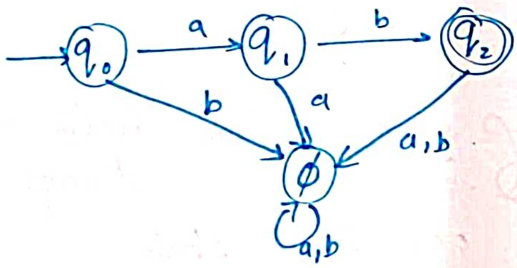
δ	a
$\rightarrow q_0$	q_1
$* q_1$	ϕ
ϕ	ϕ

Eg:



a or b

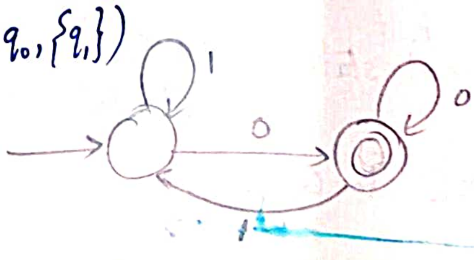
Eg:



δ	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_3	q_2
$* q_2$	q_3	q_3
q_3	q_3	q_3

Q. Design DFA that when interpreted in binary is a multiple of 2.

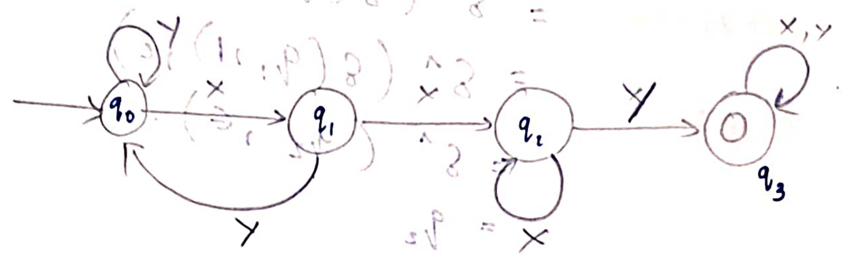
$(\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$



δ	0	1
q_0	q_1	q_0
q_1	q_1	q_0

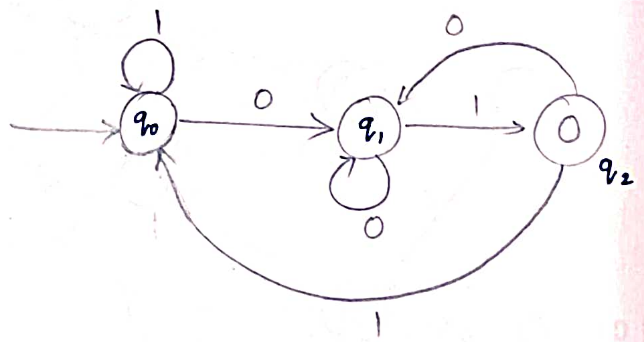
- Language of the Automata - explained here

Q. set of strings over X, Y that contains XXY



δ	X	Y
q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_2	q_3
q_3	q_3	q_3

Q. set of strings over $(0, 1)^*$ st it ends with 01



$$\hat{\delta}(q_0, w) = \hat{\delta}(\delta(q_0, a_1), a_2, \dots, a_n)$$

$$= \hat{\delta}(p_0, a_2, \dots, a_n)$$

$$= \hat{\delta}(\delta(p_0, a_2), \dots a_3, \dots, a_n)$$

$$= \vdots$$

$$= \hat{\delta}(p_n, a_n)$$

$$= \hat{\delta}(\delta(p_{n-1}, a_n)) = \hat{\delta}(p_n, \varepsilon)$$

now, $\delta^\wedge(q_0, 0101) = \delta^\wedge(\delta(q_0, 0), 101)$
$= \delta^\wedge(q_1, 101)$
$= \delta^\wedge(\delta(q_1, 1), 01)$
$= \delta^\wedge(\delta(q_2, 0), 1)$
$= \delta^\wedge(\delta(q_1, 1), \varepsilon)$
$= \delta^\wedge(q_2, \epsilon)$
$= q_2$

Automata → Language Acceptor.

Q. Design Automa

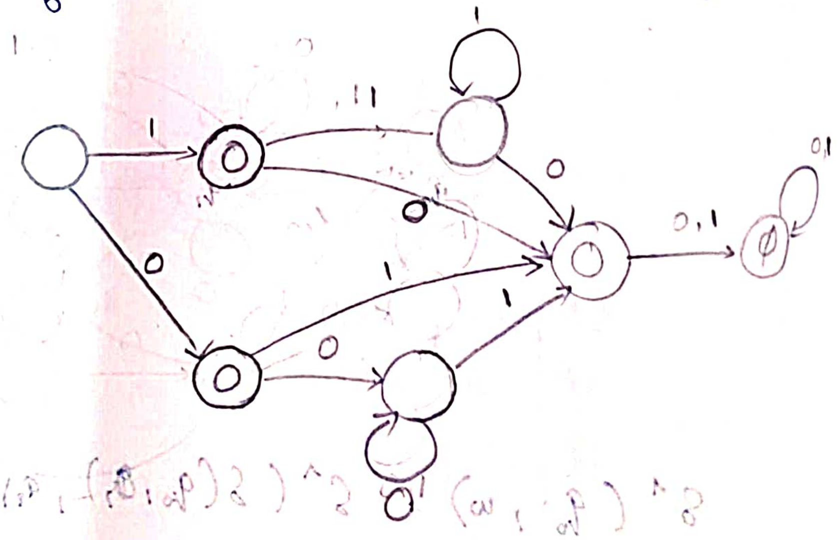
(i) set of strings over $\{0,1\}^*$ st. when interpreted in binary is multiple of 3

(ii) set $\{0,1\}^*$ st. string is either multiple of 3 or ends with 0.

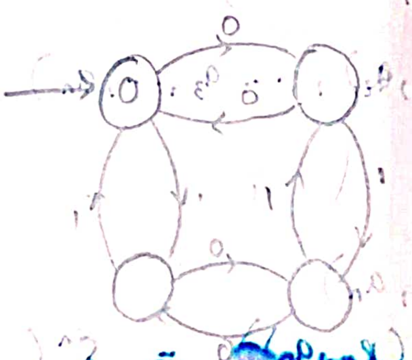
(iii) RE: $0^*1 + 1^*0$

(iv) even no. of zeroes and even no. of 1s

(iii)



(iv)

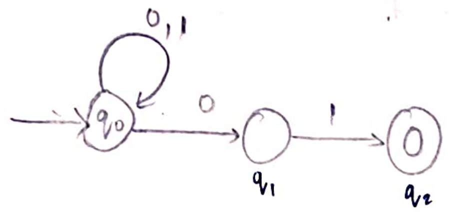


Non deterministic Finite Automata (NFA without ε)

→ On every state, on every input symbol, There are zero or more transitions.

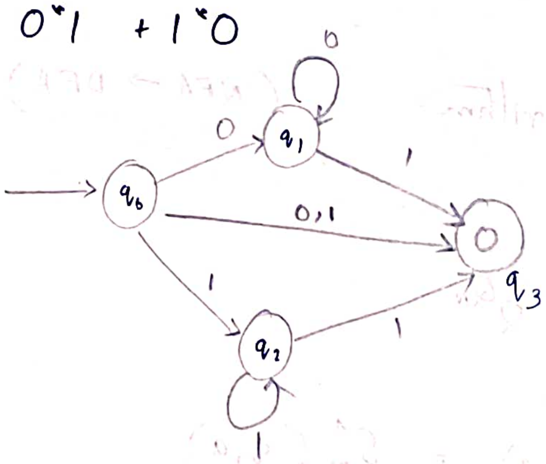
$(Q, \Sigma, \delta, q_0, F)$ $\qquad \delta : Q \times \Sigma$ to $\underbrace{2^Q}_{\text{subset of } Q}$

Eg: $(0+1)^* 01$



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

Eg: $0^*1 + 1^*0$



$\delta^\wedge(q_0, 0101) = \delta^\wedge(\delta(q_0, 0), 101)$

$= \delta^\wedge(q_0, 101) \cup \delta^\wedge(q_1, 101)$

$= \delta^\wedge(\delta(q_0, 1), 01) \cup \delta^\wedge(\delta(q_1, 1), 01)$

$= \delta^\wedge(q_0, 01) \cup \delta^\wedge(q_2, 01)$

$= \delta^\wedge(\{q_0, q_1\}, 1) \cup \cancel{\delta^\wedge(\phi, 1)}$

$= \delta^\wedge(q_0, 1) \cup \delta^\wedge(q_1, 1)$

$= q_0 \cup q_2$

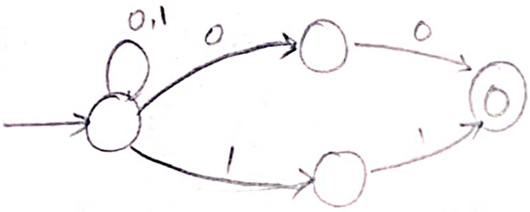
one state is final

- Computational capabilities in question, both NFA, DFA are the same, i.e; all done with same NFA, DFA can be done with DFA, NFA.

0 $$L_{NFA} = \{ w \mid \hat{\delta}(q_0, w) = R,\ R \cap F \neq \phi \}$$

$$\mid \exists\, r_i \in R,\ r_i \in F \}$$

Q. $\{0,1\}^*$, end with 00 or 11



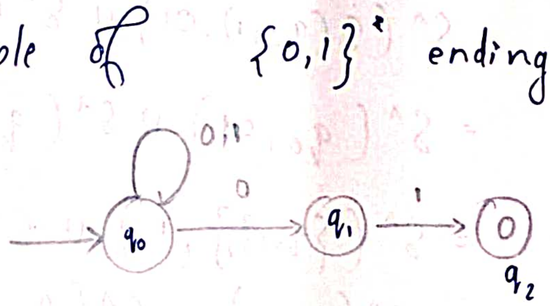
Q. $\{0,1\}^*$, even zeroes and even ones.

same as DFA

→ Subset construction Algorithm (NFA → DFA)

M_N	M_D
Q_N	$Q_D \leq 2^{Q_N}$
Σ	Σ
δ_N	$\delta_D(q, a) = \hat{\delta}_N(q, a)$
q_{0N}	$q_{0D} = [\{q_{0N}\}]$
F_N	F_D = states containing F_n

Take example of $\{0,1\}^*$ ending with 01



$\delta_D([q_0], 1) = \hat{\delta_N}(\{q_0\}, 1)$

$= [q_0]$

$\delta_D([q_0], 0) = \hat{\delta_N}(\{q_0, q_1\}, 0)$

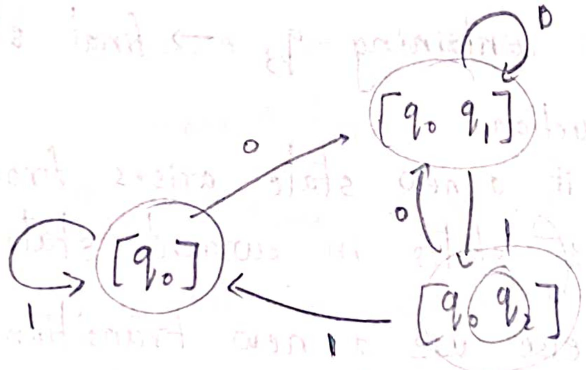
$= \cancel{\hat{\delta_N}(\{q_0, 0) \cup \hat{\delta_N}(q_1, 0)}$

$= \cancel{q_0 \cup \phi}$ $[q_0 q_1]$

$\delta_D([q_1], 0) = \phi$

$\delta_D([q_1], 1) = q_2$

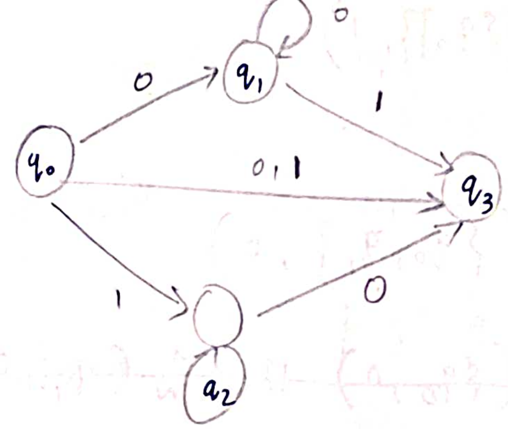
⋮



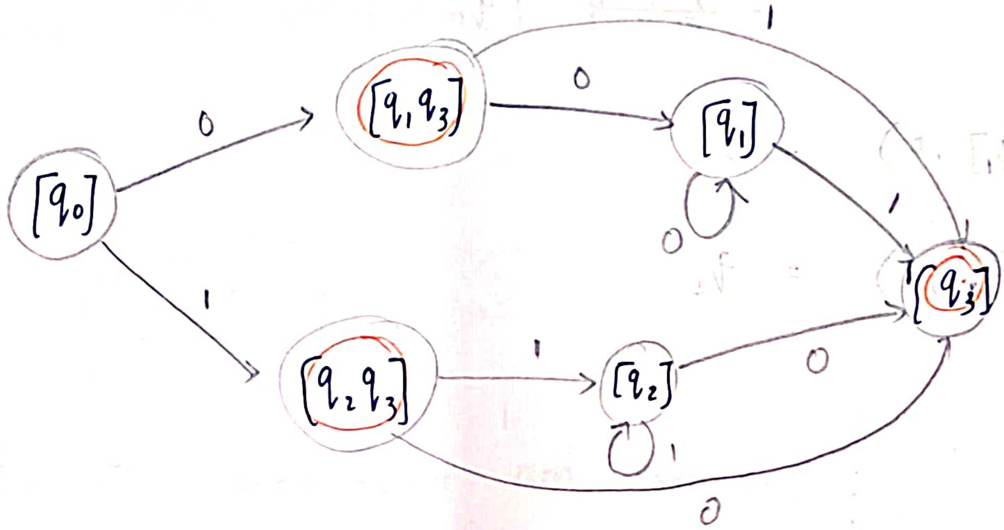
final

Automata

Eg 2:

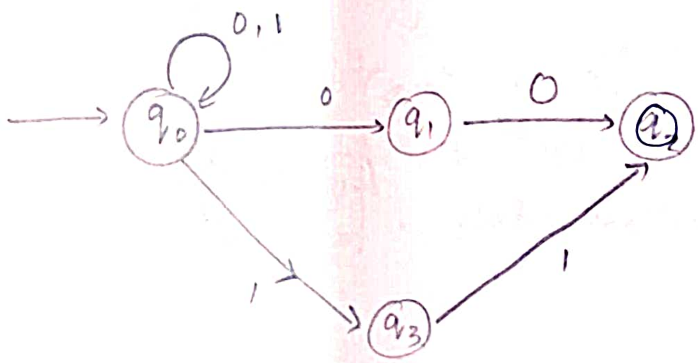


δ	0	1
q_0	$\{q_1, q_3\}$	$\{q_2, q_3\}$
q_1	$\{q_1\}$	$\{q_3\}$
q_2	$\{q_3\}$	$\{q_2\}$
q_3	ϕ	ϕ

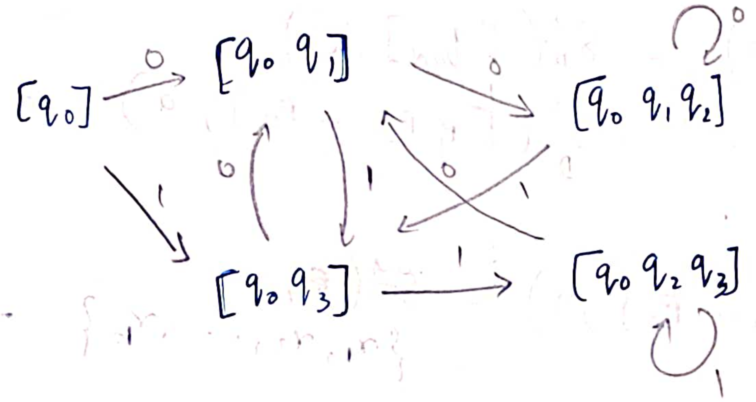


- States containing q_3 $\rightarrow$ final state
- Construction
  - (i) if a new state arises from union of states in current state, add. else use a new transition on existing state.

Eg 3:



δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	$\{q_2\}$	ϕ
q_2	ϕ	ϕ
q_3	ϕ	$\{q_2\}$



Thm 1: Language L is accepted by an NFA iff there exists a DFA

i.e.

if $L \in NFA \Rightarrow L \in DFA$ → subset construction

and

if $L \in DFA \Rightarrow L \in NFA$
→ every DFA is NFA

Proof: if $L \in NFA \Rightarrow L \in DFA$

$L_{NFA} \equiv L_{DFA}$

if $w \in L_{NFA} \Rightarrow w \in L_{DFA}$ write algorithm too

Basis : $|w| = 0 \Rightarrow w = \epsilon$

if $w \in L_{NFA} \Rightarrow q_{0N} \in F$ (start state = final state)

in the DFA,

$[q_{0N}]$ is start state

$\Rightarrow F_D$ will contain $[q_{0N}] \Rightarrow \epsilon \in L_{DFA}$.

Induction: $w = xa$, $|w| = n+1$

$$\hat{\delta}_N(q_{0N}, w) = \hat{\delta}_N(q_{0N}, xa)$$

$$= \hat{\delta}_N(\delta(q_{0N}, x_1), x_2 \dots x_n a)$$

$$\vdots$$

$$= \hat{\delta}_N(\{p_1, p_2, \dots, p_l\}, a) \quad \text{——} \; (1)$$

$\hat{\delta}_D([q_{0N}], w) = \hat{\delta}_D([q_{0N}], xa)$

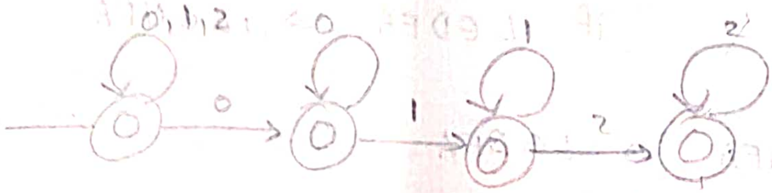
$\hat{\delta}_D([p_1\, p_2 \cdots p_n], a)$ —— (2)

$\hat{\delta}(\{p_1, p_2, \ldots p_l\}, a) = \cancel{\hat{\delta}(\epsilon}$

$= \{r_1, r_2, \ldots, r_k\}$ —— (3)

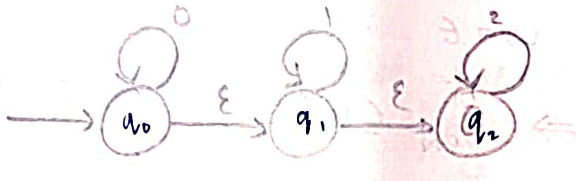
(2) $\Rightarrow \hat{\delta}([p_1, p_2, \ldots, p_n], a) = [r_1, r_2, \ldots, r_k]$

Q. Construct an NFA for ~~A~~ $0^* 1^* 2^*$



or

→ NFA with ε



ε - closure
states including itself that can reach other states. (not a null set)

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

ε - closure $(q_1) = \{q_1, q_2\}$

$(q_2) = \{q_2\}$

ε- NFA → $\hat{\delta} \neq \delta$

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

input symbols

does not have ε

$\delta : Q \times (\Sigma \cup \{\varepsilon\})$ to 2^Q

$$L_{\text{NFA with } \varepsilon} = \{ w \mid \delta^\wedge(q_0, w) = R,\ \varepsilon\text{-closure}(R) \cap F \neq \phi \}$$

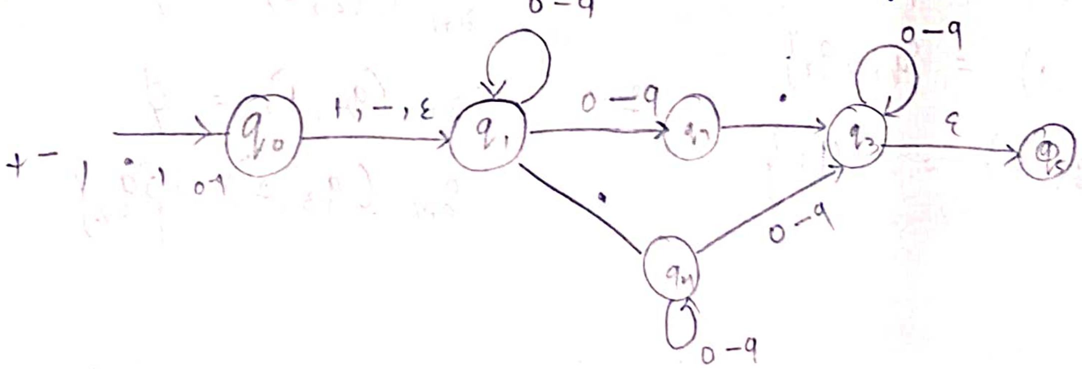
* here $\delta^\wedge$ and δ are different.

for $\rightarrow$

δ	0	1	2	ε
q_0	$\{q_0\}$	ϕ	ϕ	$\{q_1\}$
q_1	ϕ	$\{q_1\}$	ϕ	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$	ϕ

$$\begin{aligned}
\delta^\wedge(q_0, 01) &= \delta^\wedge(\delta(q_0, 0), 1) \\
&= \delta^\wedge(\epsilon\text{-closure}(q_0), 1) \\
&= \delta^\wedge(\{q_0, q_1, q_2\}, 1) \\
&= \delta^\wedge(q_0, 1) \cup \delta^\wedge(q_1, 1) \cup \delta^\wedge(q_2, 1) \\
&= \delta^\wedge(\delta(q_0, 1), \varepsilon) \cup \delta^\wedge(\delta(q_1, 1), \varepsilon) \cup \delta^\wedge(\delta(q_2, 1), \varepsilon) \\
&= \delta^\wedge(\phi, \varepsilon) \cup \delta^\wedge(\{q_1, q_2\}, \varepsilon) \cup \delta^\wedge(\phi, \varepsilon) \\
&= \delta^\wedge(\{q_1, q_2\}, \varepsilon) = \delta^\wedge(\epsilon\text{ closure}(q_2)) \\
&= \{q_2, q_1\}
\end{aligned}$$

Q. Construct ε-NFA for representing a decimal number



ε — NFA	NFA
Q_E	$Q_N = Q_E$
Σ	Σ
δ_E	$\delta_N(q,a) = \hat{\delta}_E(q,a)$
q_{0E}	$q_{0N} = q_{0E}$
F_E	$F_N = \begin{cases} F_E \cup \{q_{0N}\} & \text{if } \epsilon\text{-closure}(q_0) \text{ contains } F \\ F_E & \text{otherwise.} \end{cases}$

For same $q \longrightarrow$ $\delta_N(q_0, 0) = \hat{\delta}_E(q_0, 0)$

$$= \hat{\delta}_E(\delta_E(q_0, 0), \epsilon)$$

$$= \cancel{\hat{\delta}_E} \{ \{q_0, q_1, q_2\} \cancel{\epsilon} \}$$

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

$$\delta_N(q_0, 1) = \hat{\delta}_E(q_0, 1)$$

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

$$\delta_N(q_0, 2) = \{q_2\}$$

$\delta_N(q_1, 0) = \phi$

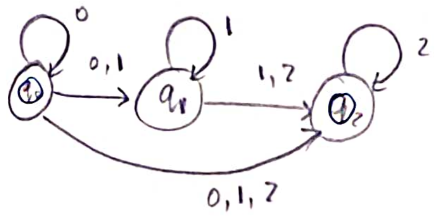
$\delta_N(q_1, 1) = \{q_1, q_2\}$

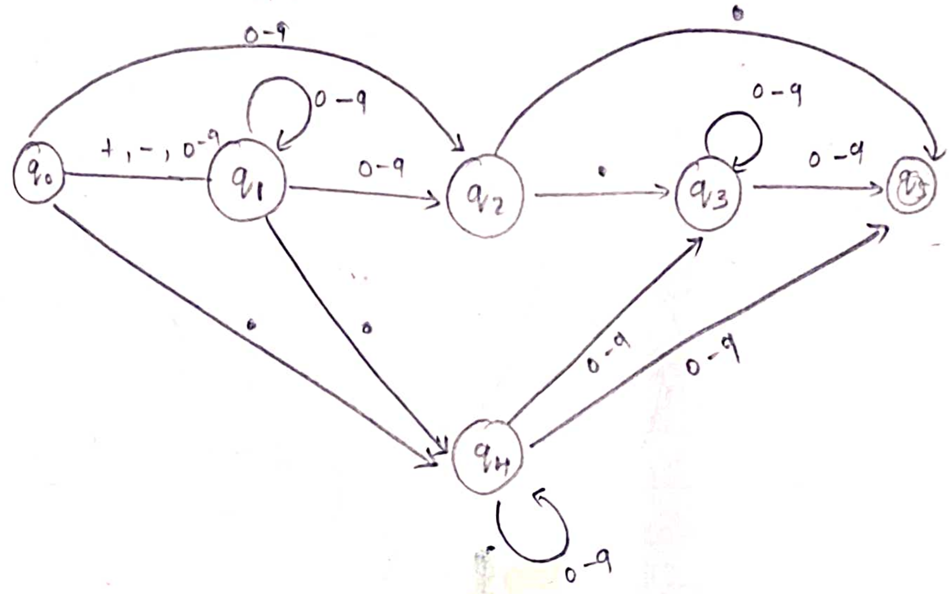
$\delta_N(q_1, 2) = \{q_2\}$

$\delta_N(q_2, 0) = \phi$

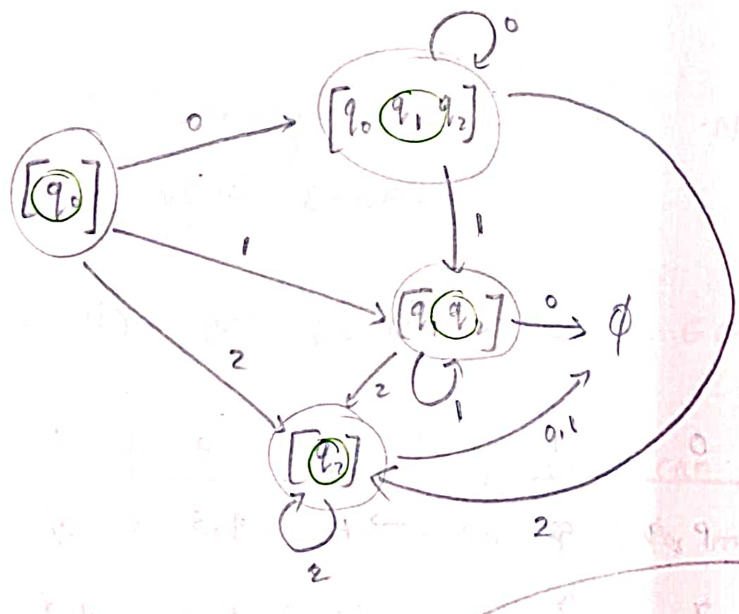
$\delta_N(q_2, 1) = \phi$

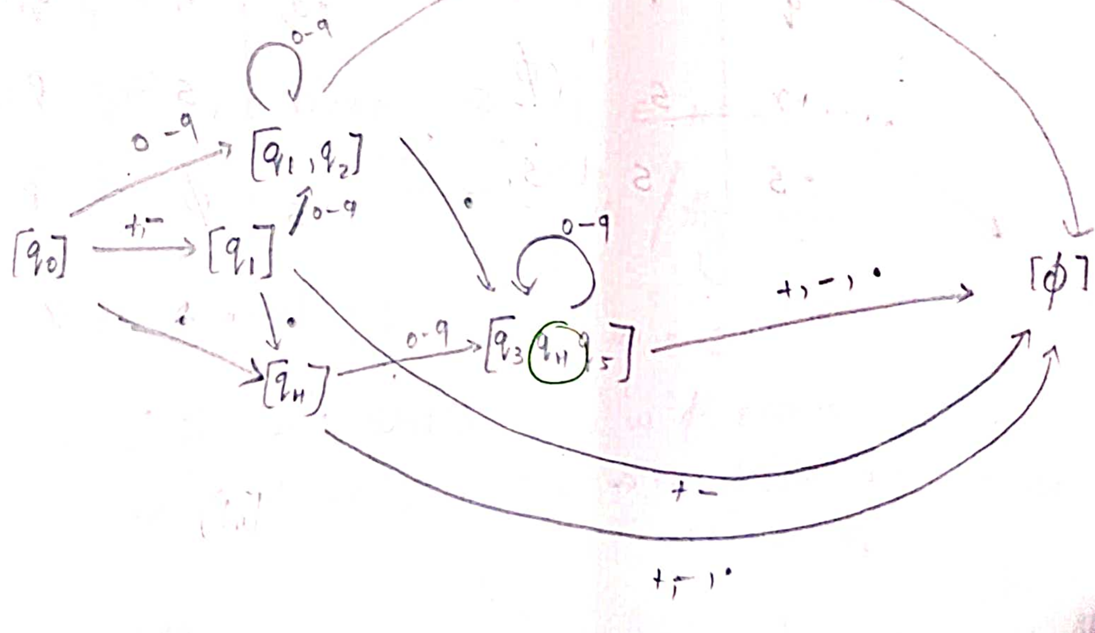
$\delta_N(q_3, 2) = \{q_2\}$


0
0,1
1
1,2
2
0,1,2


0-9
0-9
+,-,0-9
0-9
.
0-9
0-9
.
.
0-9
0-9
0-9

DFA


0
0
1
1
0
ϕ
2
2
1
0,1
2
2


0-9
0-9
$[q_1, q_2]$
+,-
.
0-9
$[q_0]$
+,-
$[q_1]$
0-9
.
+,-,.
$[\phi]$
0-9
$[q_4]$
+-
+,-,.

ε - NFA to DFA

not necessary

Q.

S	0	1
→ p	p, q	φ
q	r	r
r	s	φ
* s	s	s

S	0	1
→ p	q, s	q
* q	r	q, r
r	s	p
* s	φ	p

$$\begin{tikzcd} & {[pq]} \arrow[r, "0"] & {[p, q, r]} \\ P \arrow[ur, "0"] \arrow[dr, "1"'] & & \\ & {[P]} & \end{tikzcd}$$

Do As

Assigment

Theorem : A Language has a valid NFA iff it has a valid ε-NFA.

Prove : (i) If $L \in NFA \Rightarrow L \in \varepsilon\text{-}NFA$

on every state, we can explicitly mention the self loop on ε to make it look like ε-NFA.

(ii) If $L \in \varepsilon NFA \Rightarrow L \in NFA$

write table / Algorithm.

Basis : $|w| = 0 \Rightarrow w = \varepsilon$

if $w \in \varepsilon NFA$	$w \notin \varepsilon NFA$
$\Rightarrow$ ε-closure (q_0) contains F_E	$\Rightarrow$ ε-closure (q_0) does not contain F_E
According to conversion Algo, F_N contains q_0	According to conversion Al[go], F_N does not contain q_0

$\Rightarrow$ NFA accepts ϵ | $\Rightarrow$ NFA does not contain ϵ

$\Rightarrow$ ϵ NFA

Induction : $w = xa$

$\Rightarrow |w| = |xa| = n+1$ if $|x| = n$

$$\hat{\delta_E}(q_{0_E}, w) = \hat{\delta_E}(\delta_E(q_{0_E}, x), a)$$

$$= \hat{\delta_E}(R, a) \quad \text{--- (1)}$$

$$= \hat{\delta_E}(\delta_E(R,a))$$

$$= \hat{\delta_E}(P, \epsilon)$$

$$= \epsilon\text{-closure}(P) = P' \quad \text{--- (2)}$$

From (1)

$$\hat{\delta_N}(q_{0E}, w) = \hat{\delta_E}(q_{0_E}, xa)$$

$$= \hat{\delta_E}(R, a) \quad \text{--- (3)}$$

$$\hat{\delta_N}(R, a) = \hat{\delta_E}(R, a)$$

$$= \hat{\delta_E}(P, \epsilon)$$

$$= \epsilon\text{-closure}(P)$$

$$= P'$$

$\rightarrow w \in \epsilon$-NFA This contains start state

if $P' \cap F_E \neq \phi$

NFA $\rightarrow F_N = F_E$ or $F_E \cup \{q_{0E}\}$

$\rightarrow w \notin \epsilon NFA$

if $P' \cap F_E = \phi$

w.r.t $F_N = F_E$ or

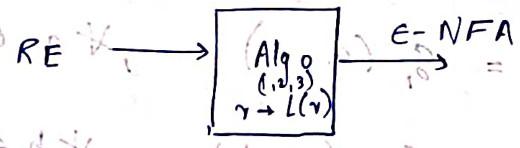
$F_E \cup \{q_{0E}\}$

$\Rightarrow P' \cap F_N = \phi$.

{ How can we prove $q_{0E} \notin P'$ }

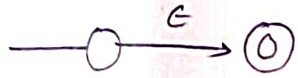
Thm : If $L(r)$ is a language of RE r, then there exists ϵ-NFA M accepting $L(r)$

1. No arc should leave final state
2. No arc should entering start state
3. Only one final state

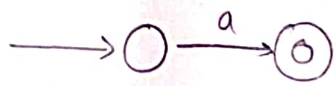


Basis : $r \longrightarrow L(r)$

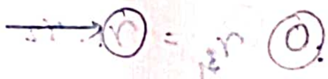
$r_1 = \epsilon \longrightarrow \{\epsilon\}$



$r_2 = a \longrightarrow \{a\}$



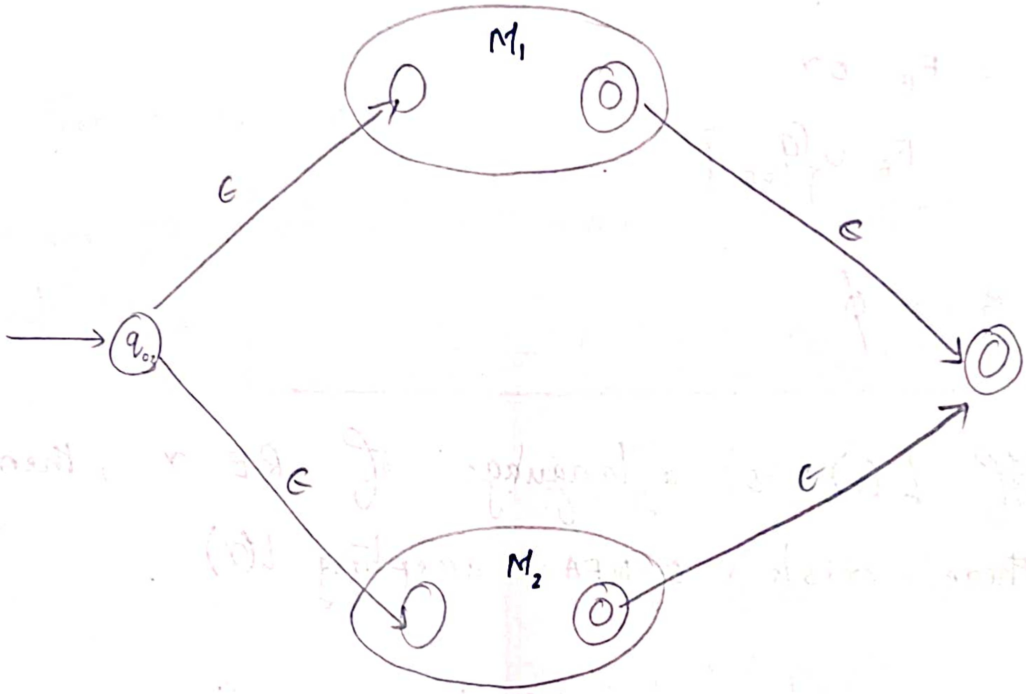
$r_3 = \phi \rightarrow \{\}$



Induction : $r_1 \rightarrow M_1 - (Q_1, \Sigma_1, \delta_1, q_{01}, F_1)$ → 1 state

$r_2 \rightarrow M_2 - (Q_2, \Sigma_2, \delta_2, q_{02}, F_2)$

Case 1 : $r_3 = r_1 + r_2$



$$\delta_3 (q_{03}, \epsilon) = \{q_{01}, q_{02}\}$$

$$\delta_3 (\{F_1, F_2\}, \epsilon) = \{q_{F3}\}$$

$$\delta_3 (q, a) = \delta_1 (q, a) \quad , \forall a \in \Sigma_1 , \forall q \in Q_1$$

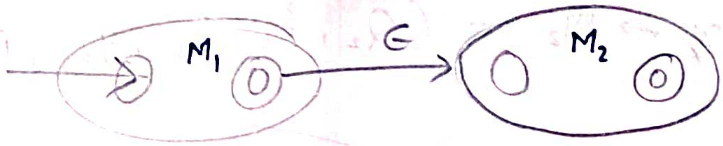
$$\delta_3 (q, b) \quad \delta_2 (p, b) \quad , \forall b \in \Sigma_2 , \forall p \in Q_2$$

?

$$r_3 = \left(Q_1 \cup Q_2 \cup \{q_{03}, q_{F3}\}, \Sigma_1 \cup \Sigma_2, \delta_3, q_{03}, \{q_{F3}\} \right)$$

Case 2 : $r_4 = r1 . r2$

$$M_4 = (Q_1 \cup Q_2, \Sigma_1 \cup \Sigma_2, \delta_4, q_{01}, F_2)$$



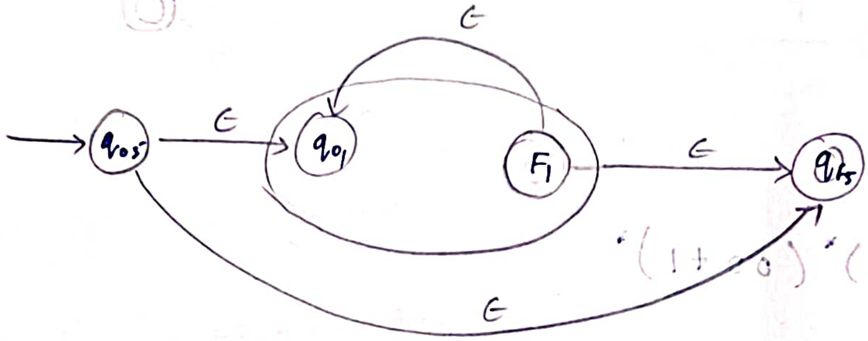
$\delta_4 (F_1, \epsilon) = \{q_{02}\}$

$\delta_4 (q, a) = \delta_1 (q, a), \forall q \in Q_1, a \in \Sigma_1$

$\delta_4 (p, b) = \delta_2 (p, b), \forall p \in Q_2, b \in \Sigma_2$

Case 3 : $r_5 = r_1^*$

$M_5 = (Q_1 \cup \{q_{05}, q_{f5}\}, \Sigma_1, \delta_5, q_{05}, \{q_{f5}\})$



$\delta_5 (q_{05}, \epsilon) = \{q_{01}, q_{f5}\}$

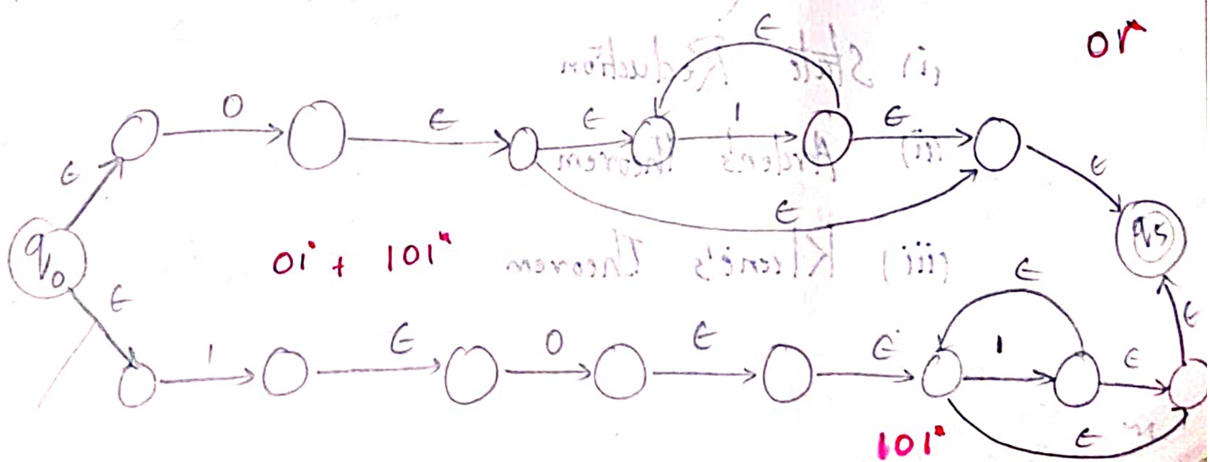
$\delta_5 (F_1, \epsilon) = \{q_{01}, q_{f5}\}$

$\delta_5 (q, a) = \delta_1 (q, a) \quad \forall q \in Q_1$
$\forall a \in \Sigma_1$

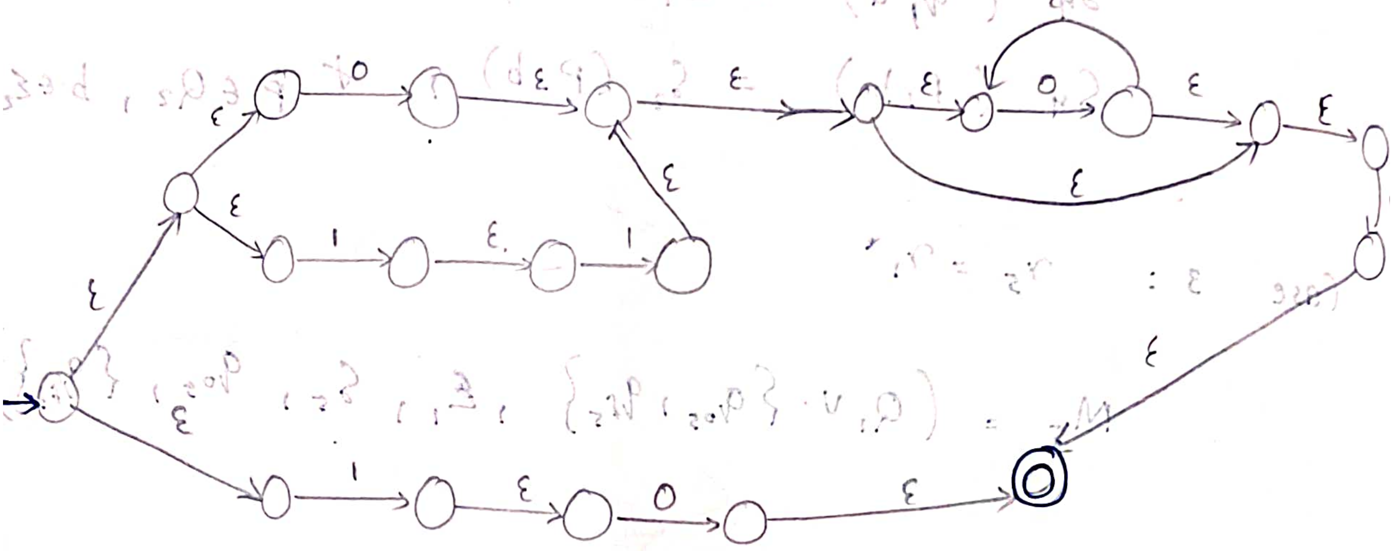
Q. $\underbrace{\underbrace{0 \, 1^*}_{r_8}}_{r_1 \; r_2 \, r_3} + \underbrace{1 \, 0 \, 1^*}_{r_4 \, r_5 \; r_6 \; r_7}$

r_9, r_{10}, r_{11}

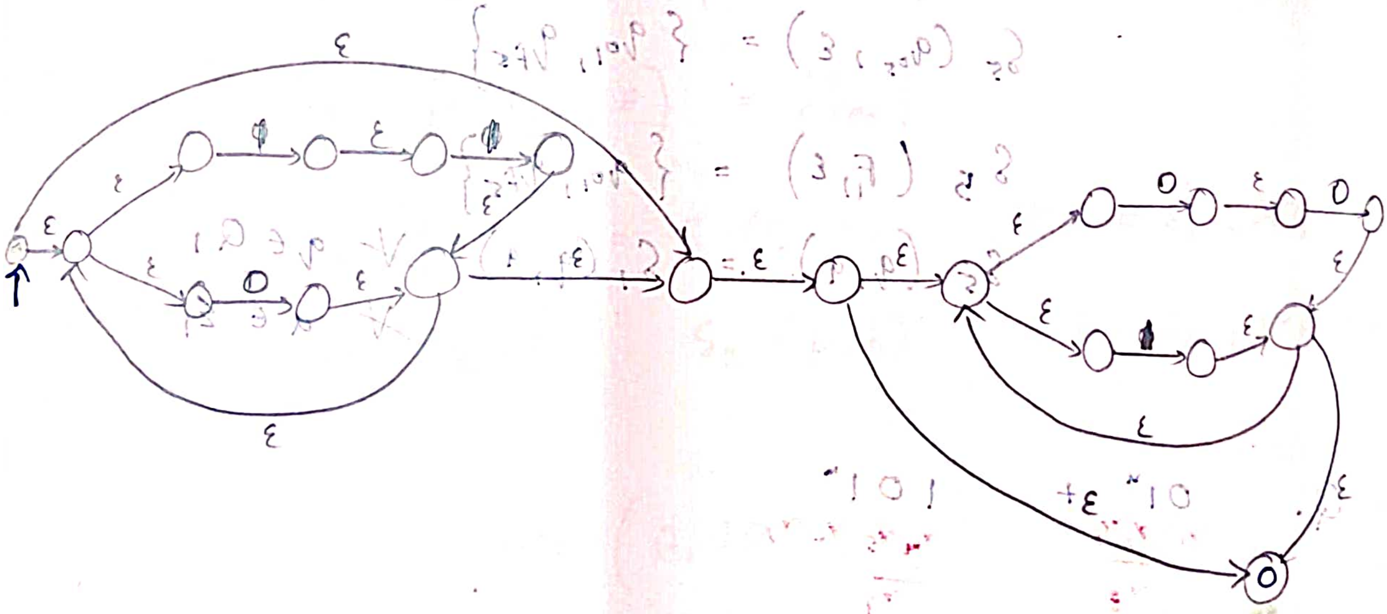
or



Q. $10 + (0+11)0^*1$



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



→ DFA to RE

(i) State Reduction

doesn't work for all DFA (ii) Arden's theorem

(iii) Kleene's Theorem → Dynamic Programming

(III) Kleen's Algo

$R^{k}_{ij} \rightarrow$ RE from i to j through k

$$R^{k}_{ij} = R^{k-1}_{ij} + R^{k-1}_{ik} \left(R^{k-1}_{kk}\right)^{*} R^{k-1}_{kj}$$

$\forall\ i = 1, 2, \dots, n$
$j = 1, 2, \dots, n$
$k = 0, \dots, n$

compute R^{0}_{ij} separately

Base:

$k=0$	$i \neq j$	$i = j$
no-edge	ϕ	ϵ
one edge a	a	$a+\epsilon$
multiple edges $a_1, a_2, \dots, a_n$	$a_1 + a_2 + \dots + a_n$	$a_1 + a_2 + \dots + a_n + \varepsilon$

★ $\phi^{*} = \varepsilon$ → according to automata for r^{*}

→ $r\phi = \phi$

→ $\phi r = \phi$

→ $r + \phi = r$

→ $\varepsilon\phi = \phi$

→ $(\varepsilon + r)^{*} = r^{*}$

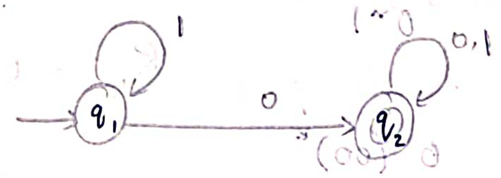
→ $(\varepsilon + r)^{*}(\varepsilon + r) = r^{*}$

$(r^{*})^{*} = r^{*}$

$\bigcup_{j_p \in F} R^{n}_{1 j_p} \rightarrow$ RE

union

Q.



R_{ij}	$k=0$	$k=1$
R_{11}	$1+\epsilon$	1^{*}
R_{12}	0	$1^{*}0$
R_{21}	ϕ	ϕ
R_{22}	$0+1+\epsilon$	$\varepsilon+0+1$

$R^{1}_{11} = R^{0}_{11} + R^{0}_{11}(R^{0}_{11})^{*}R^{0}_{11}$

$= (1+\epsilon) + (1+\epsilon)(1+\epsilon)^{*}(1+\epsilon)$

$= (1+\epsilon) + 1^{*}(1+\epsilon)$

$= 1+\epsilon + 1^{*}$

$= 1^{*}$

$R^{1}_{12} = R^{0}_{12} + R^{0}_{11}(R^{0}_{11})^{*}R^{0}_{12}$

$= 0 + (1+\epsilon)(1+\epsilon)^{*}0$

$= 0 + 1^{*}0$

$= (1^{*}+\epsilon)0$

$= 1^{*}0$

now, $R_{12}^{2} = R_{12}^{1} + R_{12}^{1}(R_{22}^{1})^{*} R_{22}^{1}$

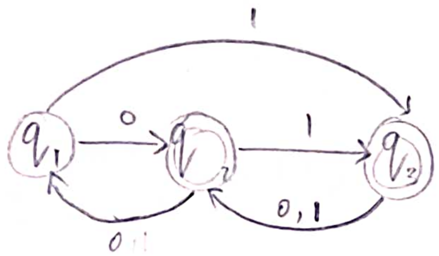
$$= 1^*0 + 1^*0\,(\varepsilon+\underbrace{0+1}_{r})^{*}\,(\varepsilon+\underbrace{0+1}_{r})$$

$$= 1^*0 + 1^*0\,(\varepsilon+(0+1))^{*}$$

$$= 1^*0\,(\varepsilon + (\varepsilon+(0+1))^{*})$$

$$= 1^*0\,((\varepsilon+(0+1))^{*})$$

$$= 1^*0\,(0+1)^{*}$$

Q. 

	K = 0	K = 1	K = 2
R_{11}	ε	ε	$(00)^*$
R_{12}	0	0	$0(00)^*$
R_{13}	1	1	0^*1
R_{21}	0	0	$0(00)^*$
R_{22}	ε	$\varepsilon+00$	$(00)^*$
R_{23}	1	$1+01$	
R_{31}	ϕ	ϕ	
R_{32}	$0+1$	$0+1$	$(0+1)(00)^*$
R_{33}	ε	ε	$\varepsilon+10^*1+0^*1$

final RE

$R_{12}^{3} + R_{13}^{3}$

$R_{12}^{3} = R_{12}^{2} + R_{13}^{2}(R_{33}^{2})^{*} R_{32}^{2}$

$R_{13}^{3} = R_{13}^{2} + R_{13}^{2}(R_{33}^{2})^{*} R_{33}^{2}$

$$R_{11}^{1} = R_{11}^{0} + R_{11}^{0} (R_{11}^{0})^{*} R_{11}^{0}$$
$$= \varepsilon + \varepsilon (\varepsilon^{*}) \varepsilon$$
$$= \varepsilon$$

$$R_{12}^{1} = R_{12}^{0} + R_{11}^{0} (R_{11}^{0})^{*} R_{12}^{0}$$
$$= 0 + \varepsilon (\varepsilon)^{*} 0$$
$$= 0$$

$$R_{13}^{1} = R_{13}^{0} + R_{11}^{0} (R_{11}^{0})^{*} R_{13}^{0}$$
$$= 1 + \varepsilon \varepsilon^{*} 1$$
$$= 1$$

$$R_{21}^{1} = R_{21}^{0} + R_{21}^{0} (R_{11}^{0})^{*} R_{11}^{0}$$
$$= 0 + 0 \varepsilon^{*} \varepsilon$$
$$= 0$$

$$R_{22}^{1} = R_{22}^{0} + R_{21}^{0} (R_{11}^{0})^{*} R_{12}^{0}$$
$$= \varepsilon + 0 \varepsilon^{*} 0$$
$$= \varepsilon + 00$$

$$R_{23}^{1} = R_{23}^{0} + R_{21}^{0} (R_{11}^{0})^{*} R_{13}^{0}$$
$$= 1 + 0 \varepsilon^{*} 1$$
$$= 1 + 0 1$$

$$R_{31}^{1} = R_{31}^{0} + R_{31}^{0} (R_{11}^{0})^{*} R_{11}^{0}$$
$$= \phi + \phi \cdots$$
$$= \phi$$

$$R_{32}^1 = R_{32}^0 + R_{31}^1 (R_{11}^1)^* R_{12}^1$$
$$= 0+1 + \phi(\ldots)$$
$$= 0+1$$

$$R_{33}^1 = R_{33}^0 + R_{31}^1 \ldots$$
$$= \varepsilon$$

$$R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1$$
$$= 0 + 0(\varepsilon+00)^*(\varepsilon+00)$$
$$= 0 + 0(00)^*$$
$$= 0(00)^*$$

$$R_{13}^2 = R_{13}^1 + R_{12}^1 (R_{22}^1)^* R_{23}^1$$
$$= 1 + 0(\varepsilon+00)^*(1+01)$$
$$= 1 + 0(00)^*(1+01)$$
$$= 1 + 0(00)^*1 + 0(00)^*01$$ expand and check

only ε is missing

$$= 1 + 0^*1 + \varepsilon$$
$$= 0^*1 + \varepsilon$$

$$R_{11}^2 = R_{11}^1 + R_{12}^1 (R_{22}^1)^* R_{21}^1$$
$$= \varepsilon + 0(\varepsilon+00)^* 0$$
$$= \varepsilon + 0(00)^*0$$ → even 0's except ε
$$= \varepsilon + (00)^*$$
$$= (00)^*$$

$$R_{21}^2 = R_{21}^1 + R_{22}^1 (R_{22}^1)^* R_{21}^1$$

$$= 0 + (\varepsilon + 00)(\varepsilon + 00)^* 0$$

$$= 0 + (00)^* 0$$

$$= 0 + (00)^* 0$$

$$= (00)^* 0$$

$$R_{32}^2 = R_{32}^1 + R_{32}^1 (R_{22}^1)^* R_{22}^1$$

$$= 0+1 + (0+1)(\varepsilon + 00)^* (\varepsilon + 00)$$

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

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

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

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

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

$$R_{33}^2 = R_{33}^1 + R_{32}^1 (R_{22}^1)^* R_{23}^1$$

$$= \varepsilon + (0+1)(\varepsilon + 00)^* (1+01)$$

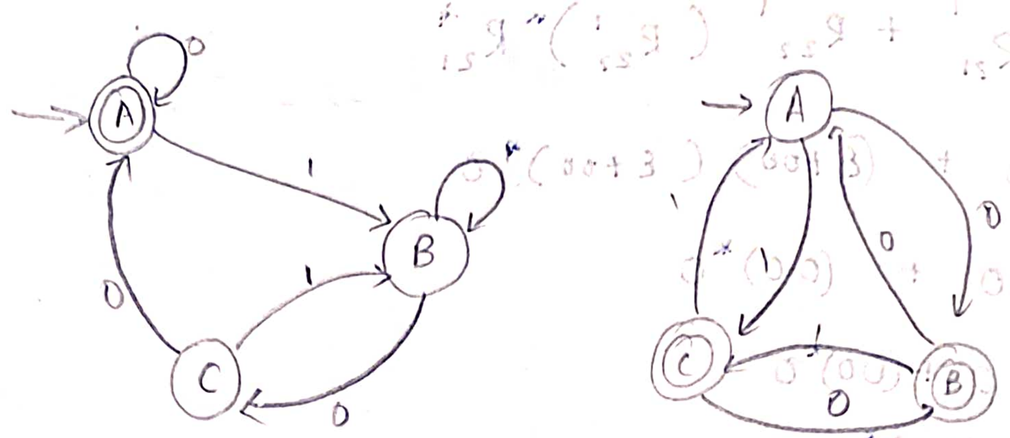
$$= \varepsilon + (0+1)(00)^* (1+01)$$

$$= \varepsilon + (0+1)((00)^* 1 + (00)^* 01)$$

$$= \varepsilon + 0(00)^* 1 + 0(00)^* 01 + 1(00)^* 1 + 1(00)^* 01$$

? $$= \varepsilon + (0^+ 1) + 10^+ 1$$

Q.



Thm: 4

If L is accepted by a DFA, L is denoted by a RE.

Basis : $|w| = 0 \Rightarrow w = \varepsilon$

if $w \in$ DFA

$\Rightarrow q_1 \subseteq F$

w.k.t $R_{11}^{0} = \varepsilon + r$

$R_{11}^{n} \in RE$

ε belongs in R_{11}^{0} (as ε gets accumulate

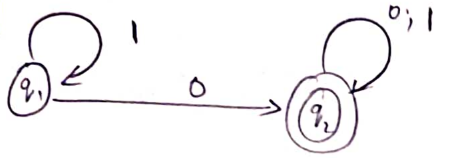
Inductive : show that $= \cup R_{1 i_p}^{n}$ contains all possible paths from start state to final state using formula

$$R_{ij}^{k} = R_{ij}^{k-1} + R_{ik}^{k-1} \left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1}$$

in an accumulative fashion.

→ Ardeens Theorem

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

→ 

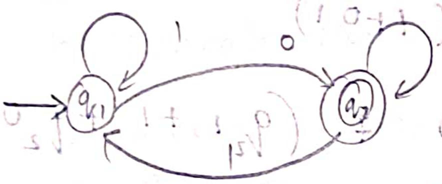
$$q_1 = \varepsilon + q_1 \cdot 1$$

$$\Rightarrow q_1 = \varepsilon 1^* = 1^*$$

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

$$= \overset{Q}{1^*0} + \overset{R}{q_2}\overset{P}{(0+1)}$$

$$\Rightarrow \boxed{q_2 = 1^*0(0+1)^*}$$ → multiple final states ↙ take union



$$q_1 = \overset{Q}{(\varepsilon + q_2 \cdot 1)} + \overset{R}{q_1} \cdot \overset{P}{1}$$

$$= (\varepsilon + q_2 \cdot 1)\, 1^*$$

$$= 1^* + q_2 1 1^*$$

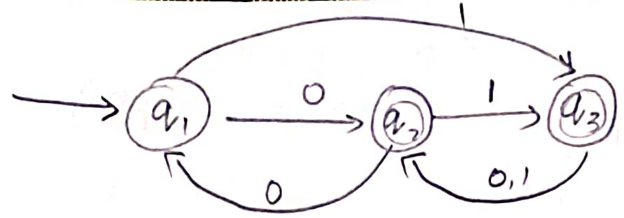
$$q_2 = q_1 0 + q_2 0$$

$$= (1^* + q_2 11^*)\, 0 + q_2 0$$

$$= 1^*0 + q_2 11^*0 + q_2 0$$

$$= 1^*0 + q_2(11^*0 + 0)$$

$$= 1^*0\,(11^*0 + 0)^*$$

Q. 

$$q_1 = \varepsilon + q_2 0 \quad \text{——— (1)}$$

$$q_2 = q_1 0 + q_3 (0+1) \quad \text{——— (2)}$$

$$q_3 = q_2 1 + q_1 1 \quad \text{——— (3)}$$

(1) in (3) $\Rightarrow$ $q_3 = q_2 1 + (\varepsilon + q_2 0) 1$

$= q_2 1 + 1 + q_2 01$

$= 1 + q_2 (1 + 01)$ ——— (4)

(4) in (2) $\Rightarrow$ $q_2 = (\varepsilon + q_2 0) 0 + (q_2 1 + 1 + q_2 01)(0+1)$

$= \cancel{0} + q_2 00 + q_2 10 + \cancel{10} + q_2 010 +$
$q_2 11 + 11 + q_2 011$

$= (0 + 10 + 11) + q_2 (00 + 10 + 010$
$+ 11 + 011)$

$= (0 + 10 + \cancel{0}11)(00 + 10 + 010 + 11 + 011)^*$ ——— (5)

(5) in (1)

$q_1 = (0 + 10 + 11)(00 + 10 + 010 + 11 + 011)^* 0$

(5) in (4) $q_3 = 1 + (0 + 10 + 11)(00 + 10 + 010 + 11 + 011)^* 0(1 + 01)$

$00 + (1 + 01)(0 + 1)$

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

$$q_2 = (\varepsilon + q_2 0) 0 + (1 + q_2(1+01))(0+1)$$

$$= 0 + q_2 00 + (10 + 11) + q_2(1+01)(0+1)$$

$$= (0 + 10 + 11) + q_2(00 + (1+01)(0+1))$$

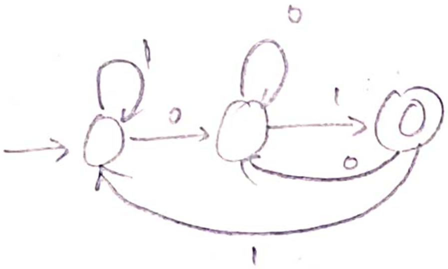
$$= (0 + 10 + 11)(00 + (1+01)(0+1))^*$$

→ Properties of Regular languages.

1. Union — $L_1 \cup L_2$ $(r_1 + r_2 = r_3)$
2. Intersection —
3. Complement —
4. Concatenation — $r_1 \cdot r_2$
5. Kleene closure $r_1^* \;?$
6. Reversal
7. Difference —

→ complement → $L(Q, \Sigma, \delta^{\wedge}, q_0, F)$

$\bar{L}(Q, \Sigma, \delta^{\wedge}, q_0, Q-F)$



→ switch final states.

→ Intersection:

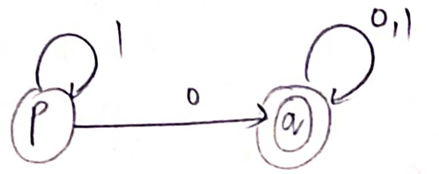
$L_1 = M_1(Q_1, \Sigma, \delta_1, q_1, F_1)$

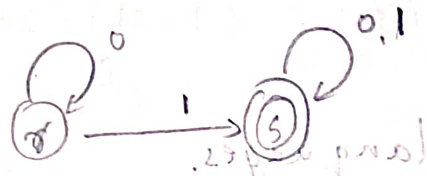
$L_2 = M_2(Q_2, \Sigma, \delta_2, q_2, F_2)$

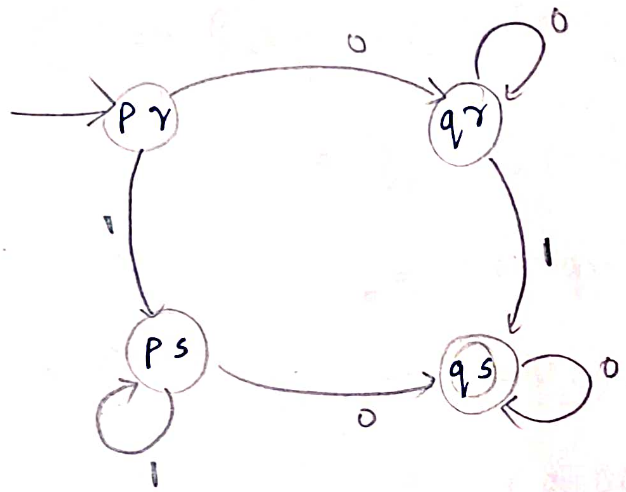
$L_1 \cap L_2 \rightarrow M_3 (Q_1 \times Q_2 , \Sigma , \delta_3 , [q_1, q_2] , F_1 \times F_2)$

$\delta_3 ([p,q], a) = (\delta_1 [p,a] , \delta_2 [q,a])$

Q.







→ Difference → $L_1 - L_2 \Rightarrow L_1 \cap \overline{L_2}$

→ Reversal

$L \rightarrow$ Regular

$L^R \rightarrow$ ..

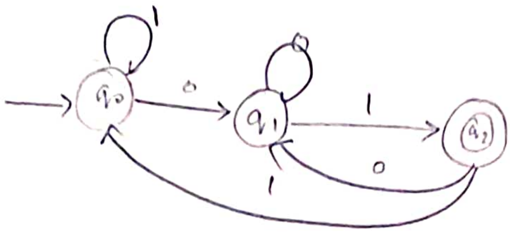
→ $(Q, \Sigma, \delta, q_0, F)$

→ $(Q \cup \{p_0\}, \Sigma, \delta', p_0, \{q_0\}, \cancel{F})$

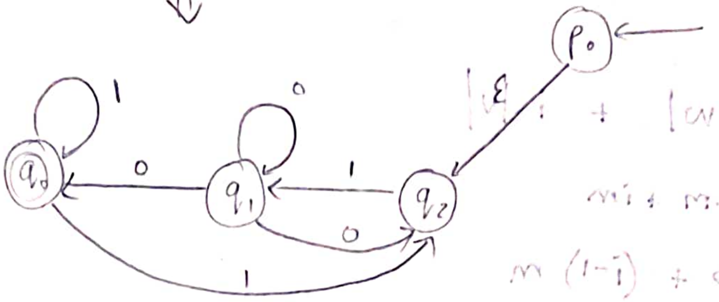
→ reverse edges

→ $\delta'(q, a) = p$ if $\delta(p, a) = q$

$\delta'(p_0, \epsilon) = F \quad \forall F \subseteq M$



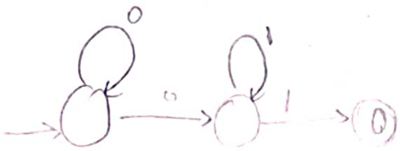
$\Downarrow$



→ Pumping Lemma

- To prove a language to be non-regular.

Q. $0^m 1^n$



Q. $(0^p$, p is prime$)$

Thm : given a sufficiently long string $|z| \geq n$

$z = uvw$

(1) $|v| \geq 1$

(2) $|uv| \leq n$ if $\exists\, i \geq 0$, $\cancel{uvw}$ $uv^iw \notin L$

then L is not R.

$\rightarrow \{ 0^p \mid p \text{ is prime} \}$

$|z| = p$

$z = uvw$

$|v| = m$

$|uw| = p - m$

Consider

$$|uv^im| = |uw| + i|v|$$

$$= p - m + im$$

$$= p + (i-1)m$$

$m = 0$

not possible as $|v| \neq 0$

$i = 1$

not possible $i \in I^+$

$\rightarrow \{ 0^i \mid i = n^2 \}$

$|z| = n^2$

$z = uvw$

$|v| = m$

$\Rightarrow |uw| = n^2 - m$

Consider

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

$$= 2m + n^2 - m$$

$$= n^2 + m$$

Q. $0^n 1^n \mid n \geq 0$ $\longrightarrow$ pattern

Case 1 : $v = 0^k$

$$\underset{u}{0^{n-k}} \mid \underset{v}{0^k} \mid \underset{w}{1^n}$$

remove $\longrightarrow$ does not have equal 0's, 1's

Case 2: $v = 1^k$

$$\underset{u}{0^n} \mid \underset{v}{1^k} \mid \underset{w}{1^{n-k}}$$

remove $\longrightarrow$ does not have equal 0's, 1's

Case 3: $v = 0^k 1^k$

$$\underset{u}{0^{n-k}} \mid \underset{v}{0^k 1^k} \mid \underset{w}{1^{n-k}}$$

now $uv^2w \rightarrow 0^{n-k}\, 0^k 1^k 0^k 1^k\, 1^{n-k}$

$= 0^n 1^k 0^k 1^n$

language itself has changed after pumping in more v.

Q. $\{0^i \mid i \rightarrow \text{perfect cube}\}$

Q. $\{0^m 1^n \mid \gcd(m,n) = 1\}$

→ Minimization of DFA

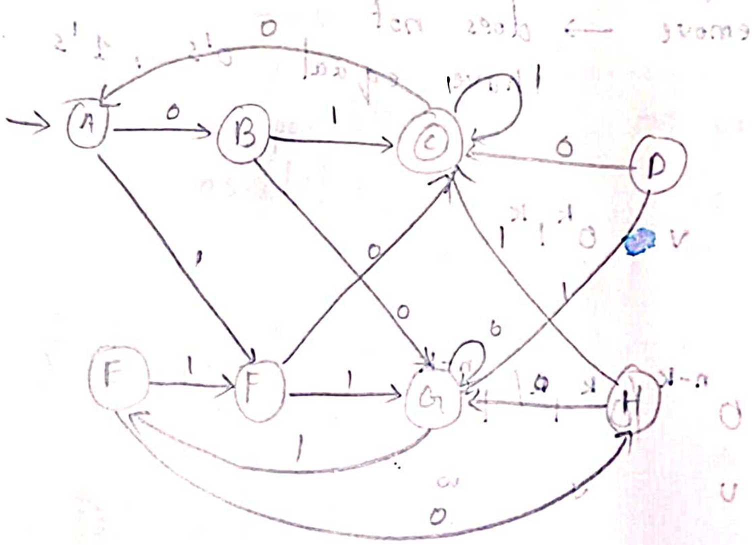
non distinguishable distinguishable

1. Equivalent / Non-equivalent ....
   Table filling

2. From table construct a new DFA.

→ DFAs equivalent?

*→ Two states are distinguishable if one is accepting state
* and other is non accepting state.



	A	B	C	D	E	F	G
B	X						
C	X	X					
D	X	X	X				
E		X	X	X			
F	X	X	X		X		
G	X	X	X	X	X	X	
H	X		X	X	X	X	X

$\delta(B,1) = C$

$\delta(A,1) = F$

$\delta(D,0) = C$

$\delta(A,0) = B$

$\delta(A, 0) = B$
$\delta(E, 0) = H$

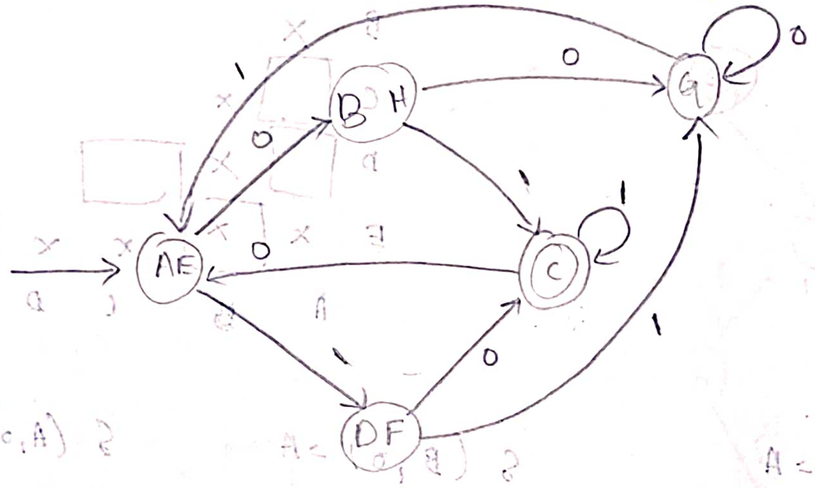
$\delta(A, 1) = F$
$\delta(E, 1) = F$ } on 1 same path

$\Rightarrow$

$\delta(B, 0) = C$
$\delta(H, 0) = C$

$\delta(B, 1) = G$
$\delta(H, 1) = G$ } equivalent

start and final state → mandatory



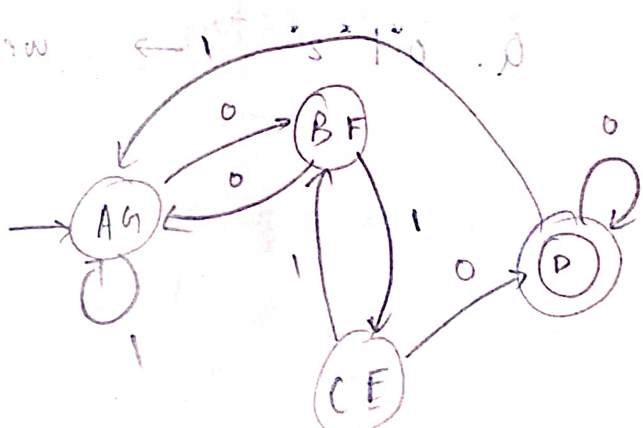
all merged states

Q.

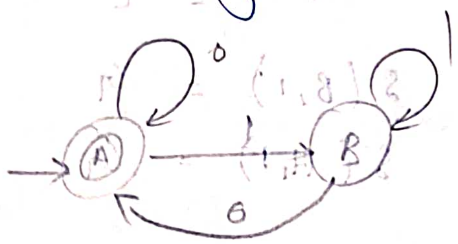
δ	0	1
→A	B	A
B	A	C
C	D	B
*D	D	A
E	D	F
F	G	E
G	F	G
H	G	D

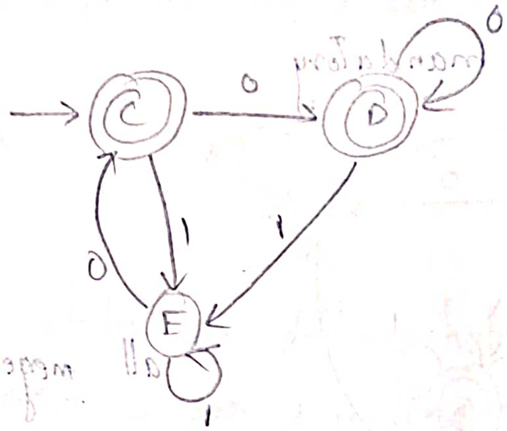
	A	B	C	D	E	F	G
B	x						
C	x	x					
D	x	x	x				
E	x	x		x			
F	x		x	x	x		
G		x	x	x	x	x	
H	x	x	x	x	x	x	x

if cycle → all pairs are equivalent



→ Equivalence of Automatas.





B	X			
C		x		
D		x		
E	x		x	x
	A	B	C	D

$\delta(A,0) = A$

$\delta(A,1) = B$

$\delta(D,0) = D$

$\delta(D,1) = E$

$\delta(B,0) = A$

$\delta(B,1) = B$

$\delta(E,0) = C$

$\delta(E,1) = E$

$\delta(A,0) = A$

$\delta(A,1) = B$

$\delta(C,0) = D$

$\delta(C,1) = E$

if BE is eq. ⇒ AD, AC ⇒ eq

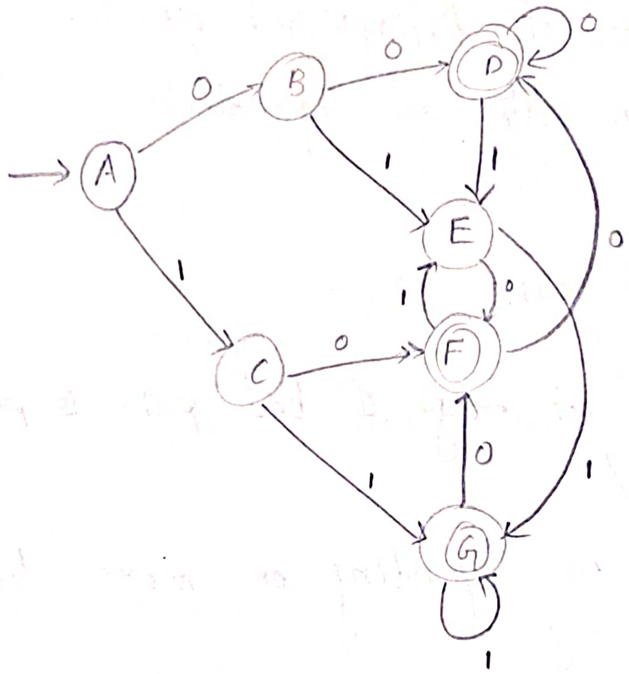
Q. $0^*1^*2^*$ → we have 4, 5 state Automata

↙

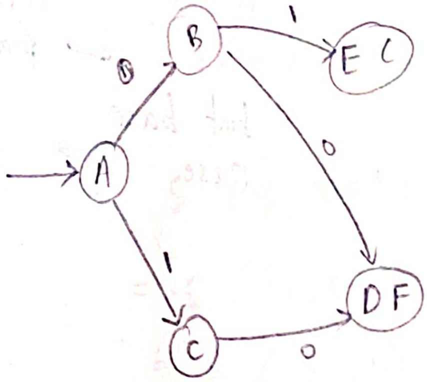
compare and combine.

Carmen → Algo

Q.



	A	B	C	D	E	F
B	x					
C	x	x				
D	x	x	x			
E	x	x	$\boxed{\ }$	x		
F	x	x	x	$\boxed{\ }$	x	
G	x	x	x	x	x	x



Thm 1: (p,q) & (q,r) are equivalent $\Rightarrow$ (p,r) is also equivalent

Contradiction : Assume (p,r) is not equivalent

$\Rightarrow$ p is accepting and r is not accepting

given → (p,q) is equivalent

p is accepting $\Rightarrow$ q is accepting.

also (q,r) is equivalent

q is accepting $\Rightarrow$ r is accepting

But r is not accepting → contradiction $\Rightarrow$ p, r → equivalent

Thm 2: If two states are not distinguished by table filling algorithm, then states are equivalent.

(p, q) → bad pair

↓ not only 1 bad pair is possible

↓ keeps on depending on more bad pairs

↓ filling the entire table with bad pair

but base case → filling final / non final state as distinguishable

↓ base case violated.

⇐ Thm proved.

→ Moore / Mealey machine

(Finite state Transducer)

$$\circ \xrightarrow{a/a} \circ \xrightarrow{b/b} \circ \xrightarrow{\epsilon/c} \circ$$

if output is on edge → mealey

" " " in node → moore.

→ Moore machine.

$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

Σ → input symbols

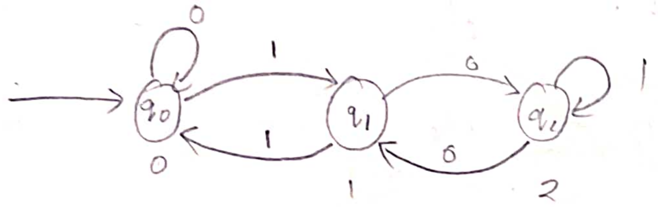
$\delta : Q \times \Sigma$ to Q

Q → Finite set of state

Δ → Finite set of output symbols

$\lambda : Q$ to Δ

q_0 → start state



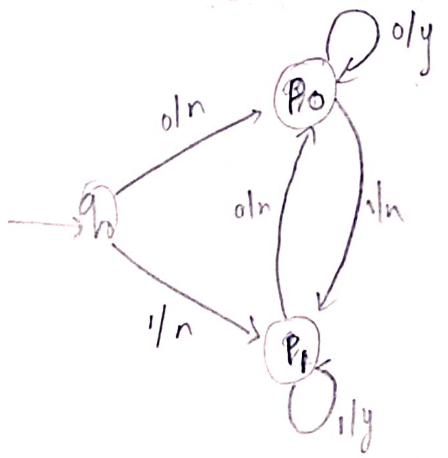
$\lambda(q_0) = 0$
$\lambda(q_1) = 1$
$\lambda(q_2) = 2$

$1001 \rightarrow \delta(q_0, 1001) = \delta(q_1, 001)$
$= \delta(q_2, 01)$
$= \delta(q_1, 1)$
$= \delta(q_0, \varepsilon)$
$= q_0$

output → $\lambda(q_1)\, \lambda(q_2)\, \lambda(q_1)\, \lambda(q_0)$
$= 1210$

→ Mealey machine

$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$ $\qquad \lambda : Q \times \Sigma$ to Δ



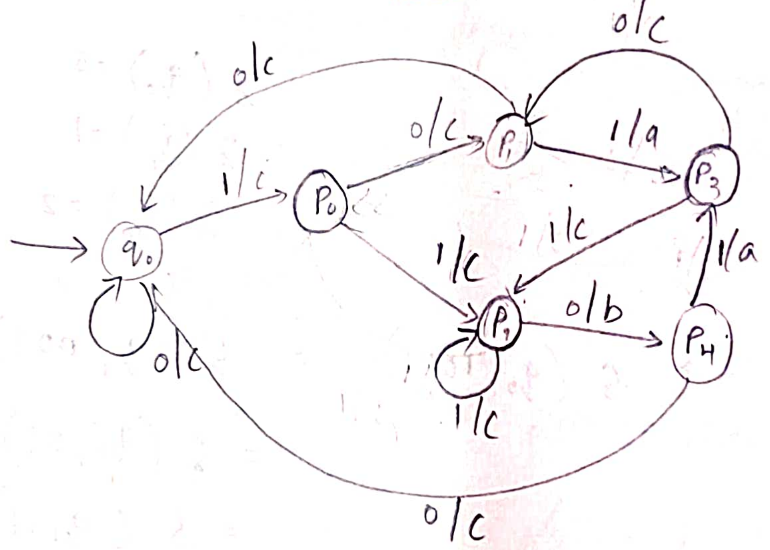
Q. Give mealey, moore machine for following

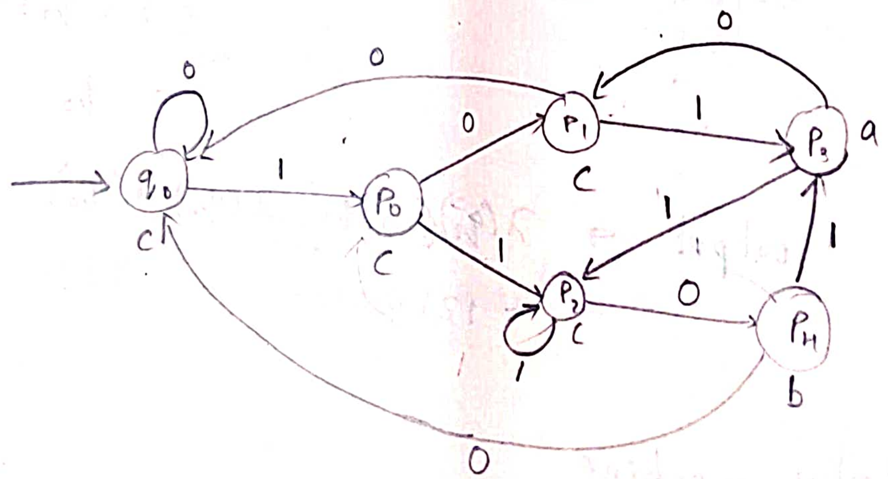
(i) for input 101 → out a

110 → out b

otherwise c.

(ii) out 1 if you have one's complement of a number or zero other wise.





Q. Adder (serial) → if carry out.

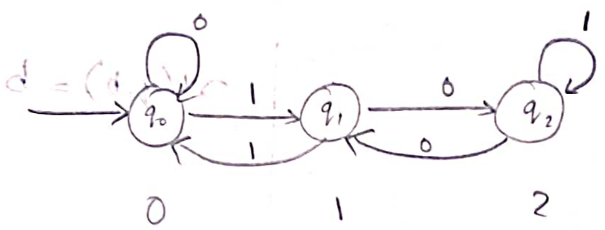
→ Moore to Mealy

Thm : If $M_1 (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ then there exists M_2 equivalent to M_1 where $M_2 (Q, \Sigma, \Delta, \delta, \lambda', q_0)$

→ $\lambda : Q$ to Δ

$$\lambda'(q', a) = \lambda(\delta(q, a))$$

Q.



δ	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

λ	
q_0	0
q_1	1
q_2	2

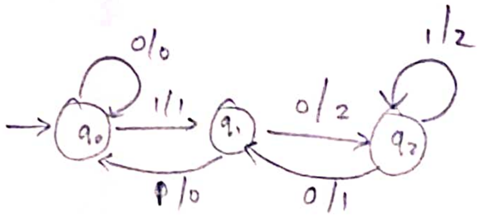
$\lambda'(q_0, 0) = \lambda(\delta(q_0, 0)) = \lambda(q_0) = 0$

$\lambda'(q_0, 1) = \lambda(\delta(q_0, 1)) = \lambda(q_1) = 1$

⋮

λ'	0	1
q_0	0	1
q_1	2	0
q_2	1	2

δ → stays same.



→ Mealey to Moore

$M_1 (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

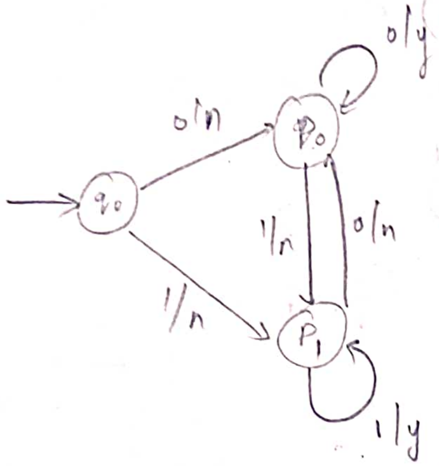
$M_2 (Q \times \Delta, \Sigma, \Delta, \delta', \lambda', [q_0, b_0])$

a_i b_i

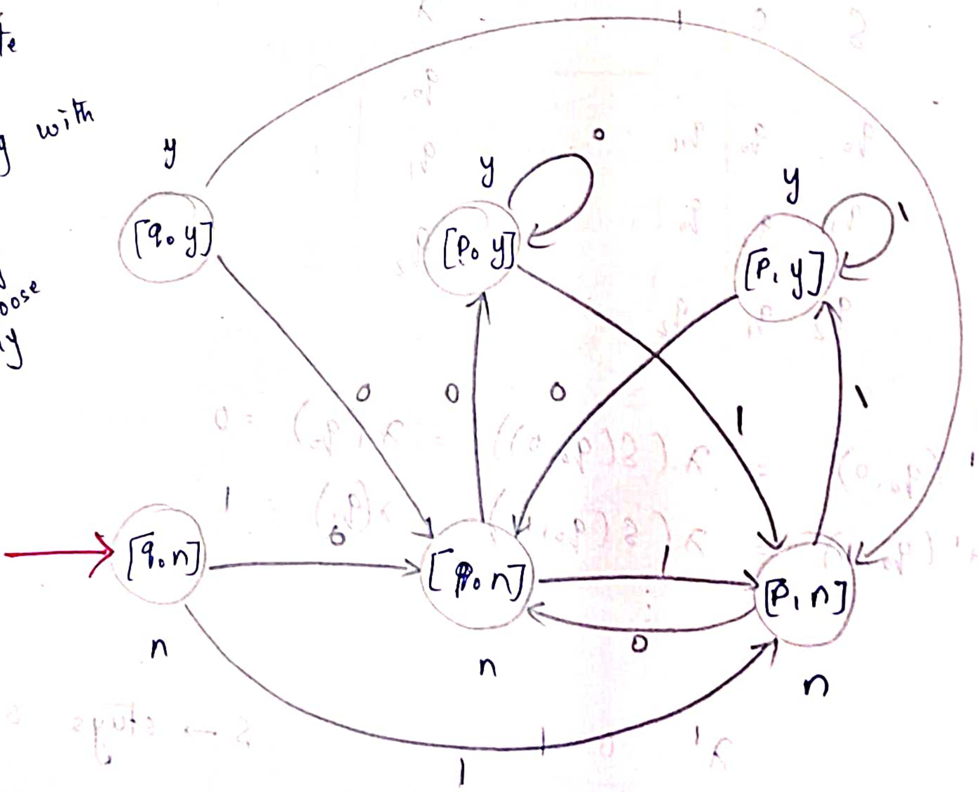
$\delta'([q, b], a)$
$= [\delta(q, a), \lambda(q, a)$

$\lambda'(q, b) = b$

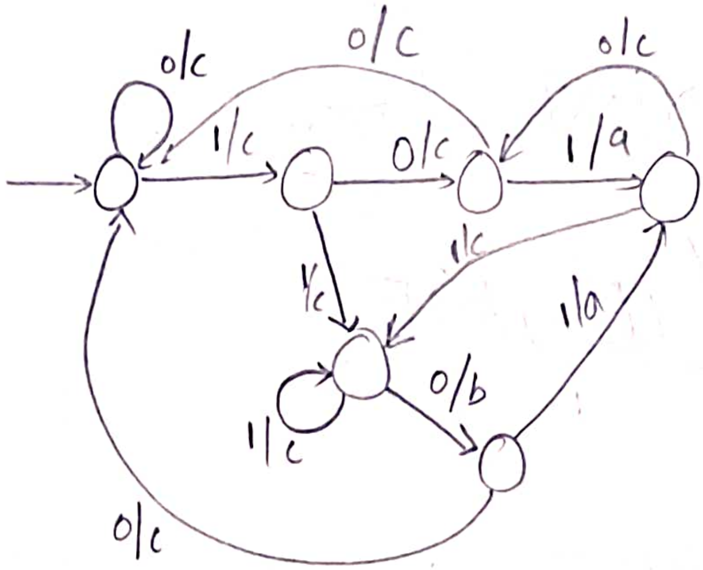
Q.

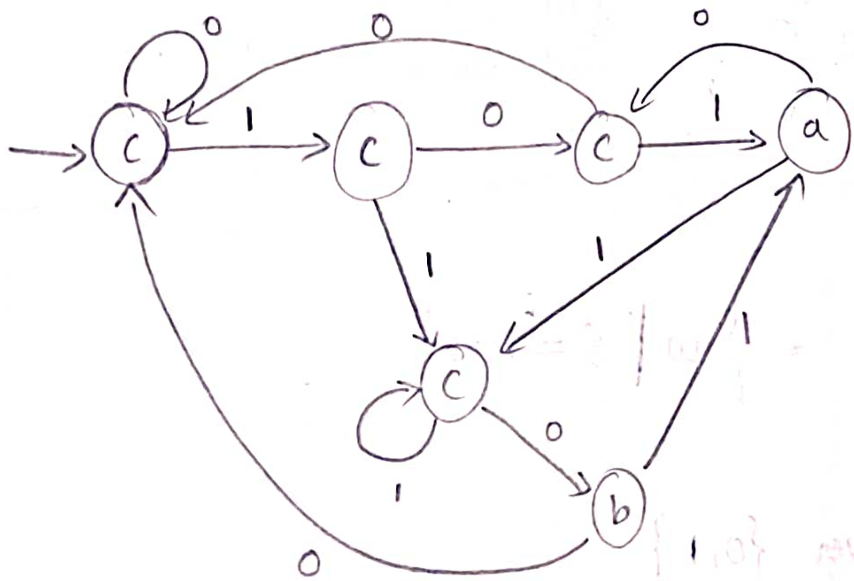


start state
↓
anything with q_0
↓
but try and choose logically



Q.





CONTEXT FREE LANGUAGE

LHS - (single non terminal)
RHS (mix b/w terminal & non-terminal)

CFG (type 2 grammar) $\rightarrow (V, T, P, S)$

terminals → T

Production → P $(V \rightarrow \alpha)$, $\alpha \in (V \cup T)^*$

set of variables (non-terminal) → V

start symbols → $S \in V$

CFG → language acceptor

$$L_{CFG} = \{ w \mid S \overset{*}{\Rightarrow} w \}$$

$\rightarrow$ (means produces or gives)

$\Rightarrow$ (derives)

Q. Palindromes over $\{0,1\}^*$

$= \{ \varepsilon, 0, 1, 00, 11, 010, 000, 111, 101 \}$

$P \rightarrow 0 \mid 1 \mid \varepsilon$

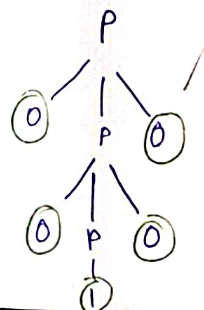
$P \rightarrow 0P0 \mid 1P1$

$L = (\{P\}\ \{0,1\}, \text{Prod.}, P)$ → does not include

Process of checking if strings belongs to CFG is called derivation / parsing.

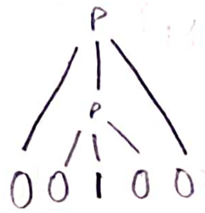
Top down / Bottom up.

Eg: 00100



→ yield of derivation tree → w (sequence of nodes visited from left to right) → belongs to CFG if root is start state

top down derivation tree.



bottom up

Leftmost derivation → leftmost non-terminal is replaced first.

Q. $L = \{a^n b^n \mid n \geq 0\}$ (in programming context, we can detect missing parenthesis)

$= \{\varepsilon, ab, aabb, aaabbb$

$P \to \varepsilon \mid ab$ $\Longrightarrow$ $S \to \varepsilon \mid aSb$

$P \to aPb$

Q. $L = \{a^n b^{2n} \mid n > 0\}$

$S \to abb \mid aSbb$

Q. $L = \{a^i b^j \mid i \leq j \leq 2i, \; i, j \geq 1$

$= \{ab, abb, aabb, aabbb, aaabbb, aabbbb\}$

$S \to ab \mid abb \mid aSb \mid aSbb$

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

$B \to b \mid bB$

$S \to aBc \mid aSc$

$B \to bB \mid \epsilon$

$S \to B \mid aSc$

$(\{S, B\}, \{a, b, c\}, P, S)$

Q. $L = \{a^n b^m c^m d^n, \; m, n \geq 1\}$

$S \to aSd \mid aBd$

$B \to bBc \mid bc$

$L = \{a^i b^j c^k \mid j > i+k,\ i,j,k \geq 1\}$

$= \{abbbbc,\ aabbbbc,\ abbbbcc \ldots$ for $j = i+k+1$

or $L = \{a^i b^i b^l b^k c^k \ldots\}$

$S \to ABC$

$A \to ab \mid aAb$

$B \to bB \mid b$

$C \to bc \mid bCc$

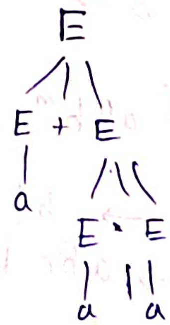
→ Expression Grammar.

$E \to a \mid (E) \mid E+E \mid E*E$

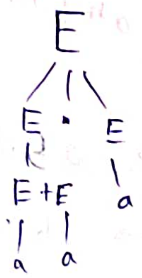
$(\{E\}, \{a, (,), +, *\}, P, E)$

for $a + a * a$

$E \Rightarrow E+E$
$\Rightarrow a + E$
$\Rightarrow a + E * E$
$\Rightarrow a + a * E$
$\Rightarrow a + a * a$



$E \Rightarrow E * E$
$\Rightarrow E * a$
$\Rightarrow E+E * a$
$\Rightarrow E + a * a$
$\Rightarrow a + a * a$



resolving ambiguity

$\rightarrow$ $E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow a \mid (E)$

$S \rightarrow iCtS \mid iCtSeS \mid a$

$C \rightarrow b$

Thm :

$G = (V, T, P, S)$ be a context free grammar. $S \overset{*}{\Rightarrow} w$ iff there is a derivation tree in grammar with yeild w.

(i) if $S \overset{*}{\Rightarrow}$ then derivation tree with yeild w

(ii) if derivation tree with yeild w, $S \overset{*}{\Rightarrow} w$

Basis :

$|w|$ = smallest possible string

$S \overset{0}{\Rightarrow} w$

$\rightarrow S \rightarrow w \in G \Rightarrow$ S w

and S w (smallest possible tree) $\rightarrow$ S | w $\Rightarrow$ $S \rightarrow w$

$S \Rightarrow w$

→ Simplification of CFG.

Order v. imp

1. ε - production elimination. ($G \rightarrow G'$)

2. Unit " " ($G' \rightarrow G''$)

3. Useless symbol

(i) → removing non-terminal not yeilding a terminal string. G'''

(ii) → if nonterminal is non reachable G''''

G^i ← yes.

⇓

convert G^i → chomsky normal form.

⇓

$A \rightarrow BC \mid a$

→ Greibach normal form

↓

$A \rightarrow a\alpha \rightarrow C^k$

Q.

$S \rightarrow ABC \mid ABD$

$A \rightarrow aA \mid a$

$B \rightarrow cD \mid c$

$C \rightarrow CC$

$D \rightarrow E$

$F \rightarrow fF \mid f$

For ϵ- elimination

(i) $A \rightarrow \epsilon$, A is said to be nullable

(iii) Identify all nullable symbols

begin

if $A \rightarrow \epsilon$, $\{ old V = \{A\}\}$

new $V = \phi$

while (old $v \neq$ new v)

new V = old V

old $v = \{A \mid A \Rightarrow \epsilon$ & $A \rightarrow \alpha$, $\alpha \in oldv^*\}$

$\cup$ old V

return old v

(iii) Replace nullables symbols with ϵ & generate new P, retaining original production except ϵ.

Eg:

$S \rightarrow ABC$

$A \rightarrow aA \mid aAA \mid \epsilon$

$B \rightarrow bB \mid bBB \mid \epsilon$

$C \rightarrow cC \mid cCC \mid \epsilon$

$\rightarrow$ old $v = \{A, B, C\}$ = new V

now $\rightarrow$ old $v = \{A, B, C, S\}$ $\rightarrow$ this becomes new V

stop.

$\rightarrow$ $S \rightarrow ABC \mid AB \mid BC \mid AC \mid A \mid B \mid C$

$A \rightarrow aA \mid aAA \mid a$

$B \rightarrow bB \mid bBB \mid b$

$C \rightarrow cC \mid cCC \mid c$

Q. $S \rightarrow AB$

$A \rightarrow aS \mid Sb \mid \epsilon$

$B \rightarrow bS \mid Sa \mid \epsilon$

$$\Downarrow$$

$S \rightarrow AB \mid A \mid B$

$A \rightarrow aS \mid Sb \mid a \mid b$

$B \rightarrow bS \mid Sa \mid a \mid b$

→ Unit production elimination

(i) Identify unit pairs

begin

old $V = \{ A \mid A \rightarrow B \in P \}$

end

(ii) Formulate unit pairs recursively / using transitive.

$A \rightarrow B \quad (A, B)$

$B \rightarrow C \quad (B, C)$

$\Rightarrow (A, C)$ is also a unit pair.

(iii) replace RHS non-terminal with its non-~~terminal~~ unit production.

$S \rightarrow AB \mid aS \mid Sb \mid a \mid b \mid bS \mid Sa$

$A \rightarrow aS \mid Sb \mid a \mid b$

$B \rightarrow bS \mid Sa \mid a \mid b$

(S, A) (S, B)
unit pairs
(not transitivity)

Previous problem

$\Rightarrow \quad S \rightarrow ABC \mid AB \mid BC \mid AC \mid aA \mid aAA \mid a \mid bB \mid bBB \mid b \mid cC \mid cCC \mid c$

$\left.\begin{array}{l} A \rightarrow \ldots \\ B \rightarrow \ldots \\ C \rightarrow \ldots \end{array}\right\}$ (same)

$G(V, T, P, S)$
$\downarrow$ ε-elimination
$G'(V, T, P', S)$
$\downarrow$ unit-production elimination

→ 3 (I)

(i) Identify NT yeilding TS

begin
  oldV = $\{A \mid A \rightarrow w\}$
  new V = ϕ
  while (oldV ≠ new V)
    new V = old V
    old V = old V $\cup \{A \mid A \rightarrow w,\ A \rightarrow \alpha,\ \alpha \in (\text{old } V \cup T)^*\}$

(ii) V - old V → usless = U

(iii) remove productions involving U.

Q. $S \rightarrow ABC \mid \cancel{ADD}$

$A \rightarrow aA \mid a$

$B \rightarrow \cancel{cD} \mid c$

$C \rightarrow CC \mid b$

$\cancel{D \rightarrow Ef}$

$\cancel{F \rightarrow fF \mid f}$

old V = $\{A, B, C, F\}$

new V = ϕ

new V = $\{A, B, F, C\}$

old V = $\{A\ B\ F\ C\} \cup \{S\}$

V - old V = $\{D, E\}$

remove production

$\rightarrow$ 3 $\rightarrow$ (II)

(ii) identify 77 reachable from S

old V = $S \cup \{A \mid S \rightarrow \alpha,\ A \in \alpha\}$

new V = ϕ

while (old V $\neq$ new V)

new V = old V

old V = old V $\cup \{A \mid B \rightarrow \alpha,\ B \in$ old V, $A \in \alpha\}$

return old V

(ii) V - old V ⇒ useless = U
(iii) remove production utilizing U

Q. $S \rightarrow ABC$
$A \rightarrow BC \mid a$
$B \rightarrow bAC \mid \epsilon$
$C \rightarrow cAB \mid \epsilon$

(i) Null → remove

old V = {B, C} → new V ⇒ old V = {A, B, C}
↙ next step
{A, B, C, S}

substitute nullable symbols with ε.

$S \rightarrow ABC \mid AB \mid AC \mid BC \mid A \mid B \mid C$
$A \rightarrow a \mid B \mid C \mid BC$
$B \rightarrow bAC \mid bA \mid bC \mid b$
$C \rightarrow cAB \mid cA \mid cB \mid c$

(ii) Unit Pairs

(S, A) (S, B), (S, C), (A, B) (A, C)

$S \rightarrow ABC \mid AB \mid AC \mid BC \mid \underbrace{a}_{a} \mid \underbrace{bAC \mid bA \mid bC \mid b}_{b} \mid \underbrace{cAB \mid cA \mid cB \mid c}_{c}$

$A \rightarrow a \mid BC \mid bAC \mid bA \mid bC \mid b \mid cAB \mid cA \mid cB \mid c$

$B \rightarrow bAC \mid bA \mid bc \mid b$

$C \rightarrow cAB \mid cA \mid cB \mid c$

(i) →

$S \rightarrow AB \mid a$

$A \rightarrow a$

Both productions have non-terminal, but B is useless
⇒ eliminate productions related to B.

(ii) →

$S \rightarrow a$

$A \rightarrow a$

A not reachable, eliminate

⇒ $S \rightarrow a$

Priority → (i) then (ii)

Q. $S \rightarrow AB \mid CA$

$A \rightarrow a$

$B \rightarrow BC \mid AB$

$C \rightarrow aB \mid b$

⇒ no ε-productions
" unit "

Use non-terminals ⇒ A, C, S → indirectly yielding non-terminals

⇒ eliminate B.

⇒ $S \rightarrow CA$

$A \rightarrow a$

$C \rightarrow b$

→ Chomsky Naur Form (CNF)

(i) iff of form $A \rightarrow BC \mid b$

Conversion → (i) DND : Do not disturb productions already in CNF

(ii) $D_1 \rightarrow a$ and ~~and~~ make appropriate substitutions

(iii) Group NT's in 2's

$D_1 \to D_2 D_3$ $\qquad$ $G_1' = (V'', T, P'', S)$

Q. Prev problem.

$S \to ABC \mid AB \mid BC \mid AC \mid a \mid bAC \mid bA \mid bC \mid b \mid c \mid cB \mid cA \mid cAB$

$A \to BC \mid a \mid bAC \mid bA \mid bc \mid b \mid c \mid cB \mid cA \mid cAB$

$B \to b \mid bAC \mid bA \mid bC$

$C \to c \mid cAB \mid cA \mid cB.$

Let $D_1 \to b$

$D_2 \to c$

$\Rightarrow$ $S \to ABC \mid AB \mid BC \mid AC \mid a \mid D_1AC \mid D_1A \mid D_1C \mid b \mid c \mid D_2B \mid D_2A \mid D_2AB$

$A \to BC \mid a \mid b \mid c \mid D_1AC \mid D_1C \mid D_2B \mid D_2A \mid D_2AB$

$B \to b \mid D_1AC \mid D_1A \mid D_1C$

$C \to c \mid D_2AB \mid D_2A \mid D_2B$

Let $D_3 \to AB$

$D_4 \to AC$

$D_1 \to b$, $D_2 \to C$

$\Rightarrow$ $S \to D_3C \mid AB \mid BC \mid AC \mid a \mid b \mid c \mid D_1D_4 \mid D_1A \mid D_1C \mid D_2B \mid D_2A \mid D_2D_3$

$A \to BC \mid a \mid b \mid c \mid D_1D_4 \mid D_1C \mid D_2B \mid D_2A \mid D_2D_3$

$B \to b \mid D_1D_4 \mid D_1A \mid D_1C$

$C \to c \mid D_2D_3 \mid D_2A \mid D_2B$

$\rightarrow$ $S \rightarrow ABC \mid BaB$

$A \rightarrow aA \mid BaC \mid aaa$

$B \rightarrow bBb \mid a \mid D$ $\quad \Rightarrow$ to CNF

$C \rightarrow CA \mid AC$

$D \rightarrow \epsilon$

Nullable $\rightarrow$ $(\cancel{D}, B)$

$S \rightarrow ABC \mid BaB \mid AC \mid a \mid Ba \mid aB$

$A \rightarrow aA \mid BaC \mid aaa \mid aC$

$B \rightarrow bBb \mid a \mid bb \mid \cancel{D}$

$C \rightarrow CA \mid AC$

write already existing, replace nullable with ϵ, if result is ϵ, skip else add to prod

unit $(B, D) \rightarrow$ unit production

(~~A D too?~~)

$(S, A, B) \longrightarrow$ useful

$C \rightarrow$ remove

$\Rightarrow$ $S \rightarrow BaB \mid a \mid Ba \mid aB$

$A \rightarrow aA \mid aaa$

$B \rightarrow bBb \mid a \mid bb$

$\hookrightarrow$ A is non reachable

$\Rightarrow$ $S \rightarrow BaB \mid a \mid Ba \mid aB$

$B \rightarrow bBb \mid a \mid bb$

$D_1 \rightarrow a$

$D_2 \rightarrow b$

$\Rightarrow \quad S \rightarrow BD_2B \mid a \mid BD_1 \mid D_1B$

$B \rightarrow D_2B\,D_2 \mid a \mid D_2D_2$

$\rightarrow \quad D_3 \rightarrow D_1B$

$D_4 \rightarrow D_2B$

$\Rightarrow$

$S \rightarrow BD_3 \mid a \mid BD_1 \mid D_1B$

$B \rightarrow D_4D_2 \mid a \mid D_2D_2$

Q. $S \rightarrow AAA \mid B$

$A \rightarrow aA \mid B$

$B \rightarrow \epsilon$

$S, B, A \rightarrow \epsilon$ - production.

$S \rightarrow AAA \mid B \mid AA \mid A$

$A \rightarrow aA \mid B \mid a$

$(S, A), (S, B), (A, B) \rightarrow$ unit Pairs

$\Rightarrow \quad S \rightarrow AAA \mid AA \mid aA \mid a$

$A \rightarrow aA \mid a$

$D_1 \rightarrow a$

$D_2 \rightarrow AA$

$\Rightarrow \quad S \rightarrow AAA \mid AA \mid D_1A \mid a$

$A \rightarrow D_1A \mid a$

$\Rightarrow S \rightarrow D_2A \mid AA \mid D_1A \mid a$

$A \rightarrow D_1A \mid a$

: If $L = L(G)$ for some grammar defined by $G = (V, T, P, S)$ then $L(G') = L(G) - \{\epsilon\}$, G' is a grammar without ϵ production.

Basis : $|w| = 1 \Rightarrow$ if $w \in L(G)$

$$S \overset{*}{\Rightarrow} w$$

hence $S \Rightarrow w$ in G'

$w \in L(G')$

Induction : $S \Rightarrow X_1 X_2 \ldots X_n$

$\Rightarrow w_1 w_2 \ldots w_n$

$\Rightarrow w$

⋮

Thm 2 : If $L = L(G)$ with ϵ - productions, $L = L(G')$ where G' will not have any unit production

then $L = L(G)$ $G = (V, T, P, S)$

$L(G') \rightarrow G'$ has no unit prod.

Basis $|w| = 1$ iff $w \in L(G)$

$|w| = 1$

$S \rightarrow w$

$S \rightarrow A$

$A \rightarrow w$ $\Rightarrow$ $S \rightarrow w$

$S \overset{*}{\Rightarrow} w$ $L(G')$

Induction : $w = xy$

$S \Rightarrow X_1 X_2 \ldots X_n$

$\Rightarrow Y_1 Y_2 \ldots Y_n$

$\Rightarrow w_1 w_2 \ldots w_n$

X_i derives w_i $\left.\begin{matrix} X_i = Y_i = w_i \\ X_i = Y_i \end{matrix}\right\}$

Thm: 3 If $L = L(G)$ with no unit or ε production, then $L(G') = L(G)'$ is without useless symbols

Basis

$|w| = 1$

if $w \in L(G)$ | $S \to w \in G'$

$S \overset{0}{\Rightarrow} w$

$\Rightarrow S \to w$

Induction

$|w| = n+1$

$S \Rightarrow X_1 X_2 \dots X_n$

$\Rightarrow Y_1 Y_2 \dots Y_n$

$\Rightarrow w_1 w_2 \dots Z_1 Z_2 \dots Z_k$

→ GNF (Greibach Naur Form)

Based on (i) Substitution

(ii) Left recursion elemination.

eg: $A \to \alpha_1 B \alpha_2$

$B \to \beta_1 | \beta_2 | \dots | \beta_K$ → (B productions)

LHS non terminal and RHS first non terminal are same

$B\gamma_1 | B\gamma_2 \dots$

left recursive

* Corollary of CNF

sequence of non-terminals | single terminal

→ substitution

$\Rightarrow A \to \alpha_1 \beta_1 \alpha_2 | \alpha_1 \beta_2 \alpha_2 | \dots | \alpha_1 \beta_K \alpha_2$

remains same $B \to \beta_1 | \beta_2 | \dots | \beta_K$

Left heavy to right heavy grammar

(i) introduce 1 new non-terminal for every A production (a letter)

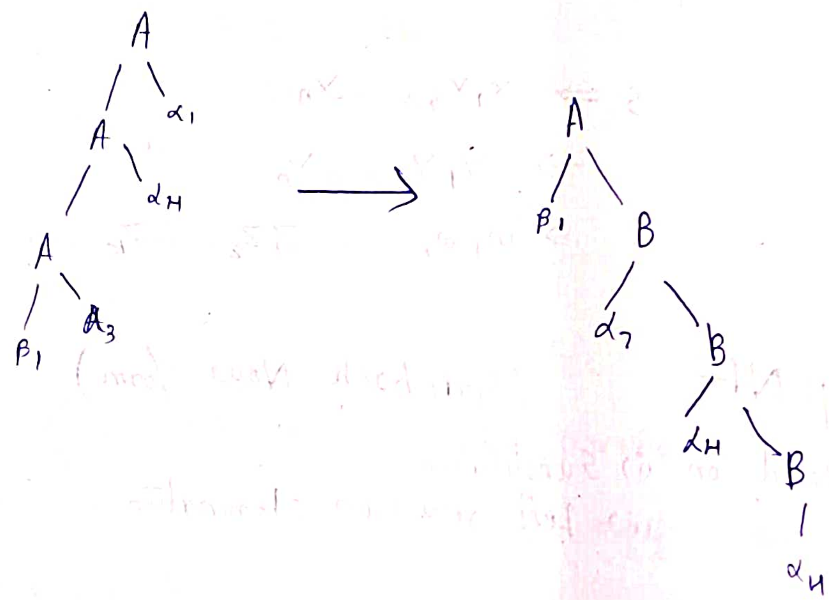
$A \to \beta_1 \mid \beta_2 \dots \beta_k \mid \beta_1 B \mid \beta_2 B \dots \mid \beta_k B$

$B \to \alpha_1 \mid \alpha_2 \mid \dots \alpha_n \mid \alpha_1 B \mid \alpha_2 B \mid \dots$

from

$A \to A\alpha_1 \mid A\alpha_2 \mid \dots \mid A\alpha_n$

$A \to \beta_1 \mid \beta_2 \mid \dots \mid \beta_n$



input for conversion to GNF → CNF or corollary of CNF

step 1 : rename all variables $A_1 \dots A_n$
↓
S

begin
for $k = 1$ to n
for $j = 1$ to $k-1$
$\forall A_k \to A_j \alpha$
$\forall A_j \to \beta$
Remove $A_k \to A_j \alpha$
Add $A_k \to \beta\alpha$

for $\forall \; A_K \rightarrow A_K \alpha$

$\forall \; A_K \rightarrow \beta_l$

Remove $A_K \rightarrow A_K \alpha$ → for all β

Add $A_K \rightarrow \beta_l B_K$

$B_K \rightarrow \alpha_l B_K \mid \alpha_l$

three cases (i) $k > j$ → only use substitution to convert to (ii) (iii)

(ii) $k < j$

(iii) $k = j$

Q. $A_1 \longrightarrow A_2 A_3$

$A_2 \longrightarrow A_3 A_1 \mid b$

$A_3 \longrightarrow A_1 A_2 \mid a$

for $k = 1$ → nothing

for $k = 2$ → $j = 1$ → $A_k \longrightarrow A_j \alpha$ (no form like this exists)

for $k = 3$, $j = 1$

$A_3 \rightarrow A_1 A_2 \mid a$

k j α

$A_j \rightarrow \beta$

⇓

$A_3 \rightarrow a \mid A_2 A_3 A_2$

j α

$j = 2$

⇓

use second part for loop

$A_3 \rightarrow a \mid b A_3 A_2 \mid A_3 A_1 A_3 A_2$

β_1 β_2 $j=3$ α_1

$\beta_1 B_3$ $\beta_2 B_3$

$A_3 \rightarrow a \mid b A_3 A_2 \mid a B_3 \mid b A_3 A_2 B_3$

$B_3 \rightarrow A_1 A_3 A_2 \mid A_1 A_3 A_2 B_3$

Sub in A_2, A_1

now, $A_2 \to \cancel{A_3} A_1 \mid b$

$\Rightarrow A_2 \to b \mid aA_1 \mid bA_3A_2A_1 \mid aB_3A_1 \mid bA_3A_2B_2A_1$

now, $A_1 \to \cancel{A_2} A_3$

$\Rightarrow A_1 \to bA_3 \mid aA_1A_3 \mid bA_3A_2A_1A_3 \mid aB_3A_1A_3 \mid bA_3A_2B_2A_1$

now, $B_3 \to A_1A_3A_2 \mid A_1A_3A_2B_3$

substitute $\Rightarrow$

$A_1 \searrow \quad 5 \times 2 \to 10$ terms.

Q. $S \to AA \mid 0$
$A \to SS \mid 1$

$\downarrow$ rename.

$A_1 \to A_2A_2 \mid 0$
$A_2 \to A_1A_1 \mid 1$

for $k = 1 \to$ nothing.
$k = 2 \to j = 1 \to \underset{k}{A_2} \to \underset{j}{A_1} \underset{\alpha}{A_1} \mid 1$

$\downarrow$

$A_2 \to \underbrace{A_2A_2A_1}_{\alpha} \mid \underset{\beta_2}{0A_1} \mid \underset{\beta_1}{1}$

$\longleftarrow$

$A_2 \to 0A_1 \mid 1 \mid \cancel{A} 1 B_2 \mid \beta_2 B_2$
$B_2 \to A_2A_1 \mid A_2A_1B_2$

$B_2 \rightarrow 0A_1A_1 \mid 0A_1A_1B_2 \mid 1A_1 \mid 1A_1B_2 \mid$

$0A_1B_2A_1 \mid 0A_1B_2A_1B_2 \mid 1B_2A_1 \mid 1B_2A_1B_2$

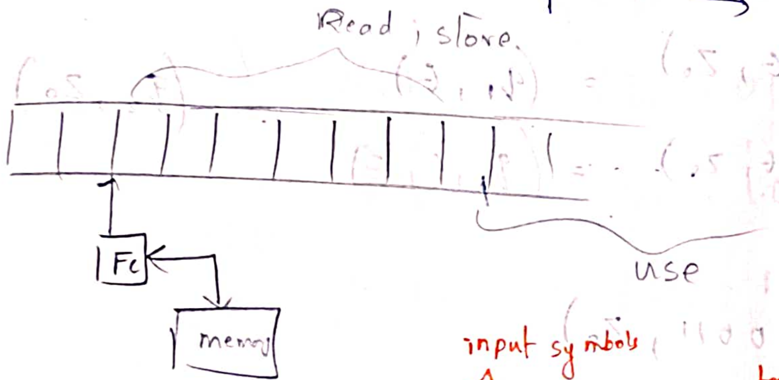
$A_1 \rightarrow 0 \mid 0A_1A_2 \mid 1A_2 \mid 0A_1B_2A_2 \mid 1B_2A_2$

PDA (Push Down Automata)

→ default → non deterministic

Deterministic → inferior

→ for recognizing context-free Language.



Definition $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, \phi)$

input symbols; transition; initial stack symbol; or F

set of state; stack symbols; start state; final state does not matter

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow Q \times \Gamma^*$$

Lang of PDA

(i) Acceptance by Final state $\{ w \mid (q_0, w, Z_0) \vdash^* (p, \epsilon, \gamma) \}$ PCF

(ii) " " Empty stack $\{ w \mid (q_0, w, Z_0) \vdash (p, \epsilon, \epsilon) \}$

input, stack — both are empty

→ Instantaneous Description ID

$$(q_0, w, Z_0) \vdash^* (p, x, \eta)$$

step

Q. $0^n 1^n$

$$\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$$

$$\delta(q_0, 0, 0) = (q_0, 0, 0)$$

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

$$\delta(q_1, 1, 0) = (q_1, \epsilon)$$

empty string $\delta(q_1, \epsilon, Z_0) = (q_1, \epsilon)$ / (q_2, Z_0)

only 1. empty string $\delta(q_0, \epsilon, Z_0) = (q_f, \epsilon)$

$$(q_0, 0011, Z_0)$$

$$\vdash (q_0, 011, 0Z_0)$$

$$\vdash (q_0, 11, 00Z_0)$$

$$\vdash (q_1, 1, 0Z_0)$$

$$\vdash (q_1, \epsilon, Z_0)$$

$$\vdash (q_2, \epsilon, Z_0)$$

$$(\{q_0, q_1\}, \{0, 1\}, \{0, Z_0\}, \delta, q_0, Z_0, \phi)$$

Q. Odd length palindrome

$wcw^R \mid w \in \{0+1\}^*$

Even length $\rightarrow ww^R$

$\delta(q_0, 0, Z_0) = \{(q_0, 0Z_0)\}$

$\delta(q_0, 1, Z_0) = \{(q_0, 1Z_0)\}$

$\delta(q_0, 0, 0) = \{(q_0, 00), (q_1, \epsilon)\}$

$\delta(q_0, 0, 1) = \{(q_0, 01)\}$

$\delta(q_0, 1, 0) = \{(q_0, 10)\}$

$\delta(q_0, 1, 1) = \{(q_0, 11), (q_1, \epsilon)\}$

$\delta(q_0, c, 0) = (q_0, 0)$

$\delta(q_0, c, 1) = (q_0, 1)$

$\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$

$\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, Z_0) = \{(q_1, \epsilon)\}$ → Accepting by empty stack)

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

For deterministic PDA → 1. $\delta(q, a, z) = (p, \gamma)$

$\forall p, q \in Q, \forall a \in \Sigma$

2. if $\delta(q, a, z)$

$\cancel{\delta(q, \epsilon, z)}$

$\forall z \in \Gamma$

Q. $a^n b^{2n}$, $n \geq 1$

$\delta(q_0, a, Z_0) = (q_0, aaZ_0)$

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

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

$\delta(q_1, b, a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)$

Q. $a^{2n} b^n$, $n \geq 1$

$\delta(q_0, a, Z_0) = (q_1, Z_0)$

$\delta(q_1, a, Z_0) = (q_0, aZ_0)$

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

$\delta(q_1, a, a) = (q_0, aa)$

$\delta(q_0, b, a) = (q_2, \epsilon)$

$\delta(q_2, b, a) = (q_2, \epsilon)$

$\delta(q_2, \epsilon, Z_0) = (q_3, Z_0)$

Thm 1 : If L is $N(M_1)$ for some PDA M_1 then $L = L(M_2)$ for some PDA.

I/P : $M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_{01}, Z_0, \phi)$

O/P : $M_2 = (Q_1 \cup \{q_{02}, q_F\}, \Sigma, \Gamma_1 \cup X_0, \delta_2, q_{02}, X_0, \{q_F\})$

$$\delta_2(q_{02}, \epsilon, X_0) = (q_0, Z_0 X_0)$$

$$\delta_2(q, a, A) = \delta_1(q, a, A)$$

$\forall q \in Q,$
$\forall a \in \Sigma$
$\forall A \in \Gamma_1$

$$\delta_2(q, \epsilon, X_0) = (q_F, X_0)$$

some q

now, the other way.

I/P : $M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_{02}, Z_0, F)$

O/P : $M_1 = (Q_2 \cup \{q_{01}, q_e\}, \Sigma, \Gamma_2 \cup X_0, \delta_1, q_{01}, X_0,$

$\phi)$ $\phi)$

$$\delta_1(q_{01}, \epsilon, X_0) = (q_{02}, Z_0 X_0)$$

$$\delta_1(q, a, A) = \delta_2(q, a, A)$$

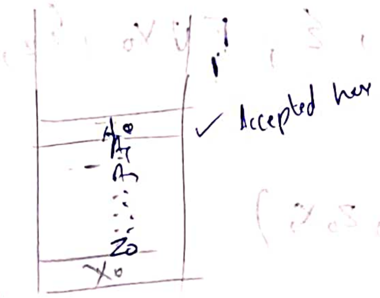
$\forall q \in Q_2 - F$
$\forall a \in \Sigma, \forall A \in \Gamma_2$

$$\delta_1(p, \epsilon, A) = (q_e, \epsilon)$$
$$\forall p \in F$$

$$\delta_1(q_e, \epsilon, A) = (q_e, \epsilon) \quad \forall A \in \Gamma_2$$

emptying the stack

$$\delta_1(q_e, \epsilon, X_0) = (q_e, \epsilon)$$



accept by empty stack

Thm: If L is context free, then there exists a PDA $\rightarrow M$ st: $L = N(M)$

$G \rightarrow (V, T, P, S)$

PDA $- (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \phi)$

for 1 single state,

PDA $- (\{q\}, T, \{V \cup T\}, \delta, q, S, \phi)$

δ:

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, \epsilon, A) = \{(q, \alpha), (q, \beta), \dots (q, \eta)\}$

if $A \rightarrow \alpha \mid \beta \mid \dots \eta$

Q. $S \rightarrow aSb \mid ab$

$(\{q\}, \{S\}, \{a, b\}, \{a, b, S\}, \delta, q, S, \phi)$

δ :

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, b, b) = (q, \epsilon)$

$\delta(q, \epsilon, S) = \{(q, aSb), (q, ab)\}$

$(q, aabb, S)$

$\vdash (q, aabb, aSb)$ $\quad$ $\vdash (q, aabb, ab)$

$\vdash (q, abb, Sb)$ $\quad$ $\vdash (q, abb, b)$ ✗

$\vdash (q, abb, aSbb)$ $\quad$ (q, abb, abb)

after 3 steps

(q, ϵ, ϵ)

$\vdash (q, bb, Sbb)$

$\vdash (q, bb, abbb)$ ✗ $\quad$ $\vdash (q, bb, aSbbb)$ ✗

Q. $P \rightarrow 0P0 \mid 1P1 \mid 0 \mid 1$

$(\{q\}, \{P\}, \{0,1\}, \{0,1,P\}, \delta, q, P, \phi)$

$\underline{\delta:}$

$\delta(q, 0, 0) = (q, \epsilon)$
$\delta(q, 1, 1) = (q, \epsilon)$
$\delta(q, \epsilon, P) = \{(q, 0P0), (q, 0), (q, 1P1), ($

Thm: If $L = N(M)$ for some pda M, then L is context free.

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

$\downarrow$

(V, T, P, S)

$\delta(q, a, A) = (q_i, X_1 X_2 \ldots X_m)$

$[q A q_m] = a [q_i X_i q_1][q_1 X_2 q_2] \ldots [q_{m-1} X_m q_m]$

$m = 0 \Rightarrow$

$\delta(q, a, A) = (q_i, \epsilon)$

$[q, A, q_i] \rightarrow a$

Variables

$\forall\ p, q \in Q$

$\forall\ A \in \Gamma$

$[p A q]$

$[p A p]$

$[q A q]$

$[q A p]$

$*[q_0 Z_0 q]$ qualifies for start symbol

Q. Consider PDA

$$\delta(\{q_0, q_1\}, \underbrace{\{X, Z_0\}}_{\text{stack symbols}}, \underbrace{\cancel{\{0,\$,1\}}\ \{0,1\}}_{\text{alphabets}}, \delta, q_0, Z_0, \phi)$$

~~Variables~~

→ Transition function, δ.

$\delta(q_0, 0, Z_0) = (q_0, XZ_0)$

$\delta(q_0, 0, X) = (q_0, XX)$

$\delta(q_0, 1, X) = (q_1, \epsilon)$

$\delta(q_1, 1, X) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, X) = (q_1, \epsilon)$, $\delta(q_1, \epsilon, Z_0) = (q_1, \epsilon)$

→ Variables →

$[q_0\ X\ q_0]$	$\boxed{[q_0\ Z_0\ q_0]}$
$[q_0\ X\ q_1]$	$\boxed{[q_0\ Z_0\ q_1]}$
$[q_1\ X\ q_0]$	$[q_1\ Z_0\ q_0]$
$[q_1\ X\ q_1]$	$[q_1\ Z_0\ q_1]$

start symbol ↓ qualifies

→ Productions

(i) $\delta(q_0, 0, Z_0) = (q_0, XZ_0)$

$[q_0\ Z_0\ q_1] = 0\ [q_0\ X\ q_0]\ [q_0\ Z_0\ q_1]$ (permute q_0, q_1 in empty space)

$[q_0\ Z_0\ q_0] = 0\ [q_0\ X\ q_0]\ [q_0\ Z_0\ q_0]$

$[q_0\ Z_0\ q_0] = 0\ [q_0\ X\ q_1]\ [q_1\ Z_0\ q_0]$

$\boxed{[q_0\ Z_0\ q_1] \Rightarrow 0\ [q_0\ X\ q_1]\ [q_1\ Z_0\ q_1]}$

S A C

(ii) $\delta(q_0, 0, x) = (q_0, xx)$

$[q_0\, x\, q_0] \longrightarrow 0\, [q_0\, x\, q_0]\, [q_0\, x\, q_0]$

$[q_0\, x\, q_0] \longrightarrow [q_0\, x\, q_1]\, [q_1\, x\, q_0]$

$[q_0\, x\, q_1 \longrightarrow [q_0\, x\, q_0]\, [q_0\, x\, q_1]$

$\boxed{[q_0\, x\, q_1] \longrightarrow 0\, [q_0\, x\, q_1]\, [q_1\, x\, q_1]}$

$\quad A \qquad\qquad\qquad A \qquad\qquad B$

(iii) $\delta(q_0, 1, x) = (q_1, \epsilon)$

$[q_0\, x\, q_1] \longrightarrow 1$

A

(iv) $\delta(q_1, 1, x) = (q_1, \epsilon)$

• → name state

$[q_1\, x\, q_1] \longrightarrow 1$

B

(v) $\delta(q_1, \epsilon, x) = (q_1, \epsilon)$

$[q_1\, x\, q_1] \longrightarrow \epsilon$

B

comes out to

$0^m 1^n \quad n < m$

(vi) $\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$

$[q_1\, z_0\, q_1] \longrightarrow \epsilon$

C

$S \to 0AC$

$A \to 1 \mid 0AB$

$B \to 1 \mid \epsilon$

$C \to \epsilon$

Q. Same as last

(i) $\delta(q_0, 1, Z_0) = (q_0, XZ_0)$

(ii) $\delta(q_0, 1, X) = (q_0, XX)$

(iii) $\delta(q_0, 0, X) = (q_1, X)$

(iv) $\delta(q_0, \epsilon, Z_0) = (q_0, \epsilon)$

(v) $\delta(q_1, 1, X) = (q_1, \epsilon)$

(vi) $\delta(q_1, 0, Z_0) = (q_0, Z_0)$

→ Productions

(iv) $\delta(q_0, \epsilon, Z_0) = (q_0, \epsilon)$

$[q_0 Z_0 q_0] \longrightarrow \epsilon$

S

(v) $\delta(q_1, 1, X) = (q_1, \epsilon)$

$[q_1 X q_1] \longrightarrow 1$

B

(iii) $\delta(q_0, 0, X) = (q_1, X)$

$[q_0 X q_0] \longrightarrow 0 [q_1 X q_0]$

$\boxed{[q_0 X q_1] \longrightarrow 0 [q_1 X q_1]}$

A B

(vi) $\delta(q_1, 0, Z_0) = (q_0, Z_0)$

$\boxed{[q_1 Z_0 q_0] \longrightarrow 0 [q_0 Z_0 q_0]}$

C

$[q_1 Z_0 q_1] \longrightarrow 0 [q_0 Z_0 q_1]$

(i) $\delta(q_0, 1, z_0) = (q_0, x z_0)$

$[q_0\ z_0\ q_0] \rightarrow 1\ [q_0\ x\ q_0]\ [q_0\ z_0\ q_0]$

$[q_0\ \underset{S}{z_0}\ q_0] \rightarrow 1\ [q_0\ \underset{A}{x}\ q_1]\ [q_1\ \underset{C}{z_0}\ q_0]$

$[q_0\ z_0\ q_1] \rightarrow 1\ [q_0\ x\ q_0]\ [q_0\ z_0\ q_1]$

$[q_0\ z_0\ q_1] \rightarrow 1\ [q_0\ x\ q_1]\ [q_1\ z_0\ q_1]$

(ii) $\delta(q_0, 1, x) = (q_0, xx)$

$[q_0\ x\ q_0] \rightarrow 1\ [q_0\ x\ q_0]\ [q_0\ x\ q_0]$

$[q_0\ x\ q_0] \rightarrow 1\ [q_0\ x\ q_1]\ [q_1\ x\ q_0]$

$[q_0\ x\ q_1] \rightarrow 1\ [q_0\ x\ q_0]\ [q_0\ x\ q_1]$

$[q_0\ \underset{A}{x}\ q_1] \rightarrow 1\ [q_0\ \underset{A}{x}\ q_1]\ [q_1\ \underset{B}{x}\ q_1]$

$S \rightarrow \epsilon \mid 1AC$

$B \rightarrow 1$

$A \rightarrow 0B \mid 1AB$

$C \rightarrow 0S$

Proof

Thm: $w \in L(G)$, then $(q, w, S) \vdash^* (q, \epsilon, \epsilon)$

Basis: $|w| = 1$

if $w \in L(G)$

$\Rightarrow S \Rightarrow w$

$\Rightarrow S \rightarrow w$

$\Rightarrow \delta(q, \epsilon, S) = (q, w)$

$\delta(q, w, w) = (q, \epsilon)$

Induction: $w = xa$

$S \Rightarrow xa$

$\Rightarrow w$

$(q, w, S) \vdash (q, xa, S)$

$\vdash (q, a, \eta)$

$\vdash (q, a, a)$

$\vdash (q, \epsilon, \epsilon)$

Thm: input $\rightarrow$ PDA (Empty stack)

output $\rightarrow$ GNF

$\delta(q_i, a, A) = (q_{i+1}, B_1 B_2 \cdots B_m)$

$[q_i\ A\ q_m] \longrightarrow a\, [q_{i+1}\ B_1\ q_1][q_1\ B_2\ q_2] \cdots [q_{m-1}\ B_m\ q_m]$

$\delta(q_{i+1}, a_1, B_1) = (q_k, \epsilon)$ → popping S_1 from stack

$\delta(q_k, a_2, B_2) = (q_2, \epsilon)$

Properties of CFL

(i) Union

(ii) Concatenation

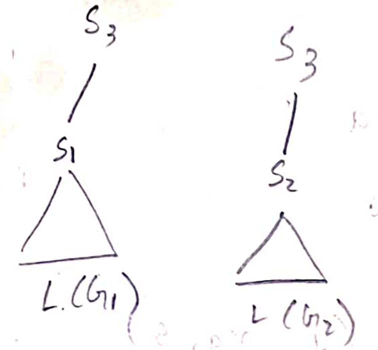
(iii) Kleen closure

If $L_1 \to G_1 = (V_1, T_1, P_1, S_1)$

$L_2 \to G_2 = (V_2, T_2, P_2, S_2)$

(i) $(V_1 \cup V_2 \cup S_3, T_1 \cup T_2, P_1 \cup P_2 \cup S_3 \to S_1 | S_2, S_3)$

$L_1 \cup L_2$

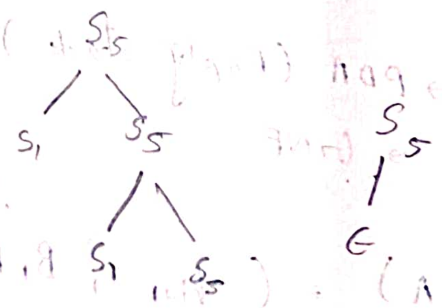


(ii) $(V_1 \cup V_2 \cup S_4, T_1 \cup T_2, P_1 \cup P_2 \cup S_4 \to S_1 S_2, S_4)$

$L_1 L_2$

(iii) $G_5 = (V_1 \cup S_5, T_1, P_1 \cup S_5 \to S_1 S_5 | \epsilon, S_5)$

L_1^*



Not closed under → complement, intersection, difference

Pumping Lemma.

$z = uvwxy$, $|z| = n$

$|vx| \geq 1$

$|uvw| \leq n$

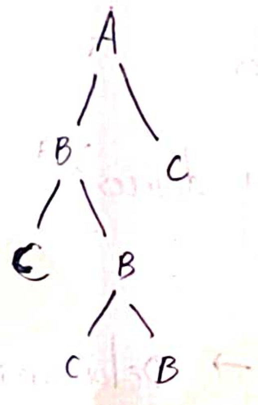
$\exists i \geq 0$

$uv^iwx^iy \notin L$

(only for CNF)

↓

so → convert to CNF



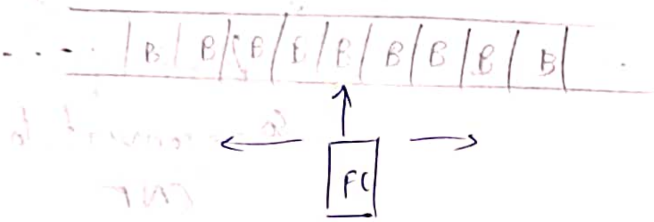
→ for a very long string, since we have only two non terminals,

NT bound to repeat itself

TURING MACHINE

mathematical model of a computer
↓
I/O
memory
IF goto

Read/write tape (two way infinite tape)



(B → Blank)
↓
encounter blank
↓
stop

(i) Acceptor

(ii) Generator

(iii) Computing devices.

- By default → Deterministic
  ↓
  both deterministic and non-deterministic have same power.

Γ → tape symbols → things we write on tape (includes Σ)

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

$$Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

$$L = \{ w \mid \delta(q_0, w) = r \in F \}$$

$$(q_0, w) \vdash$$

	0	1	X	Y	B
q_0	(q_1, X, R)			(q_3, Y, R)	(q_H, B, R)
q_1	$(q_1, 0, R)$	(q_2, Y, L)		(q_1, Y, R)	
q_2	$(q_2, 0, L)$		(q_0, X, R)	(q_2, Y, L)	
q_3				(q_3, Y, R)	(q_H, B, R)
q_H	-	-	-	-	-

2^{nd} approach

B

$\cancel{0}$ 0 0 0 1 1 1 $\cancel{1}$ B

make blank $\longrightarrow$

$\longleftarrow$ move both ways

δ	0	1	B
q_0	(q_1, B, R)		(q_H, B, R)
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$	(q_2, B, L)
q_2		(q_3, B, L)	
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_0, B, R)
q_H	—	—	—

Q. $a^n b^n c^n$

$\cancel{a}\,\cancel{a}\,\cancel{a}\,\cancel{a}\,\cancel{b}\,b\,\cancel{b}\,b\,\cancel{c}\,\cancel{c}\,\cancel{c}\,c$

X X X Y Y Y Z Z Z

δ	a	b	c	X	Y	Z	B
q_0	(q_1, X, R)						$(q_5, B,$
q_1	(q_1, a, R)	(q_2, Y, R)			(q_1, Y, R)		
q_2		(q_2, b, R)	(q_3, Z, L)			(q_2, Z, R)	
q_3	(q_4, a, L)	(q_3, b, L)		(q_3, X, R)	(q_3, Y, L)	(q_3, Z, L)	$(q_5, B,$
q_4	(q_4, a, L)			(q_0, X, R)			
q_5	–	–	–	–		–	

Q. $w \in \{0,1\}^*$ → palindrome. (can start all over ↙ recursively enumerable)

/ 0 0100 / B

δ	0	1	B
q_0	(q_1, B, R)	(q_2, B, R)	(q_6, B, R)
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$	(q_3, B, L)
q_2	$(q_2, 0, R)$	$(q_2, 1, R)$	(q_4, B, L)
q_3	(q_5, B, L)		
q_4		(q_5, B, L)	
q_5	$(q_5, 0, L)$	$(q_5, 1, L)$	(q_0, B, R)
q_6	–	–	–

Q. $\{w \mid n_0(w) = n_1(w)\}$, $w \in \{0,1\}^*$

δ	0	1	X	B
q_0	(q_1, B, R)	(q_2, B, R)	(q_0, B, R)	(q_H, B, R)
q_1	$(q_1, 0, R)$	(q_3, X, L)	(q_1, X, R)	
q_2	(q_3, X, L)	$(q_2, 1, R)$	(q_2, X, R)	
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_3, X, L)	(q_0, B, R)
q_4	-	-	-	-

$\cancel{0}$ 1 1 1 0 1 0

B X

0 0 0 1 1 1 0 1

q_0 sees X
↓
all left symbols have been used.

Q. ww , $w \in \{0,1\}^*$ → preprocessing

q_{10} → final state

0 1 1 0 0 1 1 0

X Y Y X C D D C

0 1 1 1 1 0 1 1 0

δ	0	1	X	Y	C	D	B
q_0	(q_1, X, R)	(q_1, Y, R)			(q_4, C, R)	(q_4, D, R)	
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$			(q_2, C, L)	(q_2, D, L)	(q_2, B, L)
q_2	(q_3, C, L)	(q_3, D, L)					
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_0, X, R)	(q_0, Y, R)			
q_4					(q_4, C, R)	(q_4, D, R)	(q_5, B, L)
q_5	—				$(q_6, 0, L)$	$(q_7, 1, L)$	
q_6	$(q_6, 0, L)$	$(q_6, 1, L)$	$(q_8, 0, R)$		(q_6, C, L)	(q_6, D, L)	
q_7	$(q_7, 0, L)$	$(q_7, 1, L)$		$(q_8, 1, R)$	(q_7, C, L)	(q_6, D, L)	
q_8	$(q_8, 0, R)$	$(q_8, 1, R)$			(q_9, C, R)	(q_9, D, R)	(q_{10}, B, L)
q_9	$(q_5, 0, L)$	$(q_5, 1, L)$			(q_9, C, R)	(q_9, D, R)	

→ Computation :

(i) Addition (m, n)

000100

⇓

0000

	0	1	B
q_0	q_1, B, R	(q_3, B, R)	
q_1	$(q_1, 0, R)$	$(q_2, 0, R)$	
q_2	$(q_2, 0, R)$		(q_3, B, R)
q_3	–	–	–

to handle

method 2 : copy ~~000~~ ~~1~~00$\underline{0}$ $\underline{0}$ $\underline{0}$

	0	1	B
q_0	q_1, B, R	q_3, B, R	
q_1	$q_1, 0, R$	$q_1, 1, R$	q_2, B, L
q_2	$q_2, 0, L$	$q_2, 0, L$	q_0, B, R
q_3	–	–	–

(ii) $n \% 2$

	0	B
q_0	(q_1, B, R)	(q_2, B, R)
q_1	(q_0, B, R)	$(q_2, 0, R)$
q_2	–	–

(iii) $n \% 3$

	0	B
q_0	(q_1, B, R)	(q_3, B, R)
q_1	(q_2, B, R)	$(q_3, 0, R)$
q_2	(q_0, B, R)	$(q_1, 0, R)$
q_3	–	–

(iv) $m \dot{-} n = \begin{cases} m-n & \text{if } m > n \\ 0 & \text{else} \end{cases}$

	0	1	B
q_0	(q_1, B, R)	(q_6, B, R)	
q_1	$(q_1, 0, R)$	$(q_2, 1, R)$	
q_2	$(q_2, 0, R)$		(q_3, B, L)
q_3	(q_4, B, L)	$(q_5, 0, R)$	
q_4	$(q_4, 0, L)$	$(q_4, 1, L)$	(q_0, B, R)
q_5	–	–	–
q_6	(q_6, B, R)		q_5, B, R

(v) Increment in binary

δ	0	1	B
q_0	$(q_2, 1, R)$	$(q_1, 0, L)$	
q_1	$(q_2, 1, R)$	$(q_1, 0, L)$	$(q_2, 1, R)$
q_2	{ –	–	–

try decrement

Binary bitwise AND, OR, XOR, NAND, NOR.

convert binary to unary and vice versa.

string length.

$\log_2 n$ and 2^n

(vi) Decrement

δ	0	1	B
q_0	$(q_1, 1, L)$	$(q_2, 0, R)$	
q_1	$(q_1, 1, L)$	$(q_2, 0, R)$	
q_2	–		

(vii) Binary to unary

δ	0	1	#	B
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$		$(q_1, \#, L)$
q_1	$(q_2, 1, L)$	$(q_3, 0, R)$		
q_2	$(q_2, 1, L)$	$(q_3, 0, R)$		(q_6, B, R)
q_3	$(q_3, 0, R)$	$(q_3, 1, R)$	$(q_4, \#, R)$	
q_4		$(q_4, 1, R)$		$(q_5, 1, L)$
q_5		$(q_5, 1, L)$	$(q_1, \#, L)$	
q_6		(q_6, B, R)	(q_6, B, R)	

(viii) Unary to Binary

q	0	1	#	B
q_0		$(q_0, 1, R)$		$(q_7, \#, L)$
q_1		$(q_1, 1, L)$		(q_2, B, R)
q_2		(q_3, B, R)		
q_3		$(q_3, 1, R)$	$(q_4, \#, R)$	
q_4	$(q_6, 1, L)$	$(q_5, 0, R)$		
q_5	$(q_6, 1, L)$	$(q_5, 0, R)$		$(q_6, 1, L)$
q_6	$(q_6, 0, L)$	$(q_6, 1, L)$	$(q_1, \#, L)$	
q_7		$(q_1, \cancel{\#}1, L)$		(q_8, B, R)
q_8			(q_f, B, R)	
q_f	–	–	–	–

(ix) $\log_2 n$

→ convert n to largest power of 2 less than number

↓

by converting every other '1' to 0 except the highest set bit.

↓

then just remove the highest bit → left over zero

↓

$\log_2 n$ in unary

↓

if needed convert to binary

(x) 2^n → convert binary to unary and ~~convert to~~ and add 1 to the first Blank → ans.

(xi) $m \times n$

$\cancel{0}0001\cancel{00}100$

1

00 ←

→ Increasing power of turing machine

1. Storage in finite control
2. Multiple Tracks
3. Checking off symbols
4. Subroutines
5. Shifting over

(I) (i) Ensure first char is unique.

the alphabet set is $\{0,1\}$.

* Try this with ww^R

	0	1	B
$[q_0, B]$	$([q_0, 0], 0, R)$	$([q_0, 1], 1, R)$	$([q_1, B], B, R)$
$[q_0, 0]$	–	$([q_0, 0], 1, R)$	$([q_1, B], B, R)$
$[q_0, 1]$	$([q_0, 1], 0, R)$	–	$([q_1, B], B, R)$
$[q_1, B]$	–	–	–

(II) (i)

	0		
	1		
	1		

$\delta(q_0, [0\ 1\ 1]) = (q_1, [0\ 0\ 1], R)$

↳ reads whole column.

(ii) used for and, or operations and stuff

→ Keep first two tracks with the two numbers

→ Keep third track for answers.

(III) (i) To check palindrome

a	b	b	a	b	b	a	B
✓						✓	B

$\delta([q_0, B], [a, B]) = ([q_0, a], [a, ✓], R)$

$\delta([q_0, B], [b, B]) = ([q_0, b], (b, ✓), R)$

$\delta([q_0, a], [a/b, B]) = ([q_0, a], [a/b, B], R)$

$\delta([q_0, b], [a/b, B]) = ([q_0, b], [a/b, B], R)$

$\delta([q_0, a], [B, B]) = ([q_1, a], [B, B], L)$

$\delta([q_0, b], [B, B]) = ([q_1, a]\ [B, B], L)$

$\delta([q_1, q_9], [a, B]) = ([q_2, B], [a, \checkmark], L)$

$\delta([q_1, b], [b, B]) = ([q_2, B], [b, \checkmark], L)$

$\delta([q_2, B], [a/b, B]) = ([q_2, B], [a/b, B], L)$

$\delta([q_2, B], [a/b, \checkmark]) = ([q_0, B], [a/b, \checkmark], R)$

$\delta([q_0, b], [a/b, \checkmark]) = ([q_1, b], [a/b, \checkmark], i)$

no need to go to end ! See a tick and reset.

$\delta([q_0, a], [a/b, \checkmark]) = ([q_1, a], [a/b, \checkmark], L)$

$\delta([q_0, B], [a/b, \checkmark]) = ([q_3, B], [a/b, B], R)$

(V) (i) left shift by 2

BB abba

$\delta([q_0, B, B], A_1) = ([q_0, B, A_1], B, R)$

$\delta([q_0, B, A_1], A_2) = ([q_0, A_1, A_2], B, R)$

$\delta([q_0, A_1, A_2], A_3) = ([q_0, A_2, A_3], A_1, R)$

→ Variations of TM

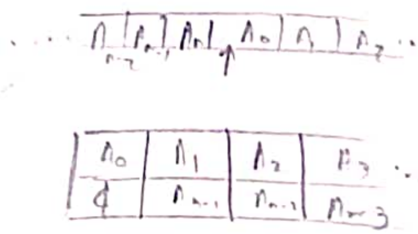
1. One way infinite tape
2. Non-deterministic
3. Multi tape.

Thm : If a Language L is recognized by turing machine, with a two-way infinite tape, then L is recognized by TM with a one-way infinite tape.

$M_2 \rightarrow$ two-way

$(Q_2, \Sigma_2, \Gamma_2, \delta_2, q_2, B, F_2)$

$M_1 \rightarrow (Q_1, \Sigma_1, \Gamma_1, \delta_1, q_1, B, F_1)$



$\forall a \in \Sigma_2 \cup \{B\}$

① $\delta_1(q_1, [a, B]) = ([q, U], x, ¢, R)$

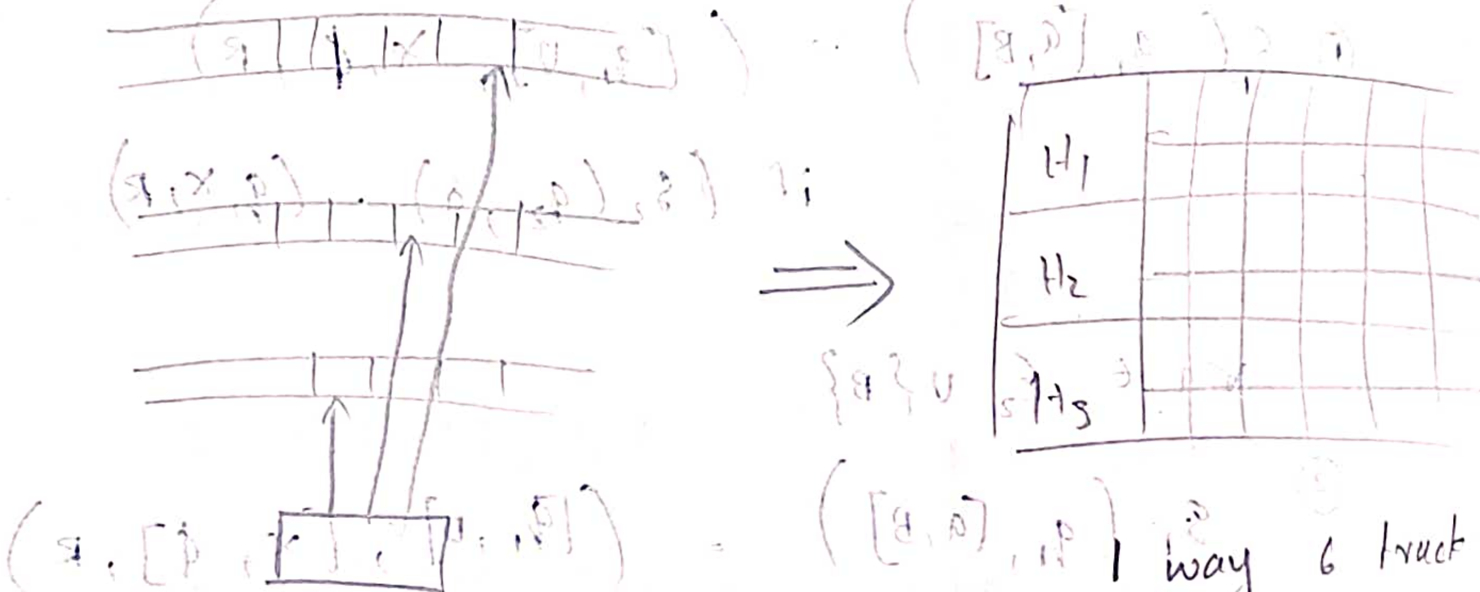
if $\delta_2(q_2, a) = (q, x, R)$

$\forall a \in \Sigma_2 \cup \{B\}$

② $\delta_1(q_1, [a, B]) = ([q, D], [x, ¢], R)$

$\delta_2(q_2, a) = (q, x, L)$

Multi - tape



1 way 6 track

all controls

move independent

2 way

3 track

same power

Undecidibility

Solvable | Unsolvable → Infinity

Infinity → countable | uncountable

Gregor cantor thm.

countable → map something to class of integers

Solvable → P (polynomial time) | NP (non-deterministic polynomial)

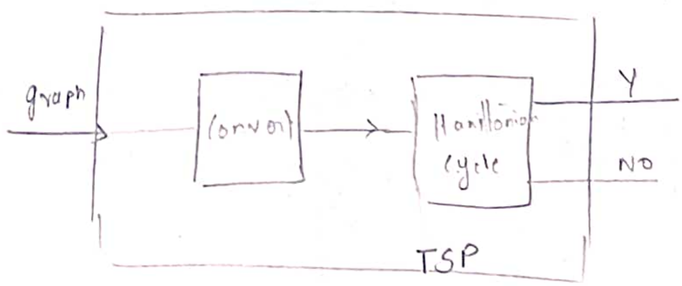
NP → Complete | Hard

$L \leq_p L'$ → NP hard

Verifiable in P

* some problems
↓
no turing machine possible
↓
undecidable or unsolvable.

Eg: Reducibility



by using a convert function (to complete graph)
↓
we can solve TSP by using algo for Hamiltonian cycle

* Decision problem
⇒ (yes or no) answer
↓
Polynominal time.

$\{ x\#y \mid A(x,y) = 1$, $x \in$ Problem , y instance

question

Problem converted to language.

Algorithm that gives yes

any problem can be converted to graph → can be converted to string

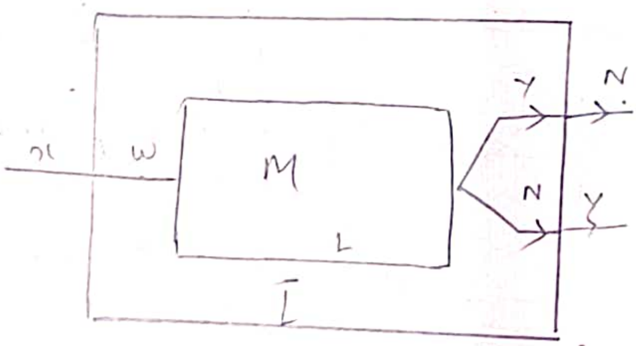
TM

Recursive Enumerable → subset → Recursive

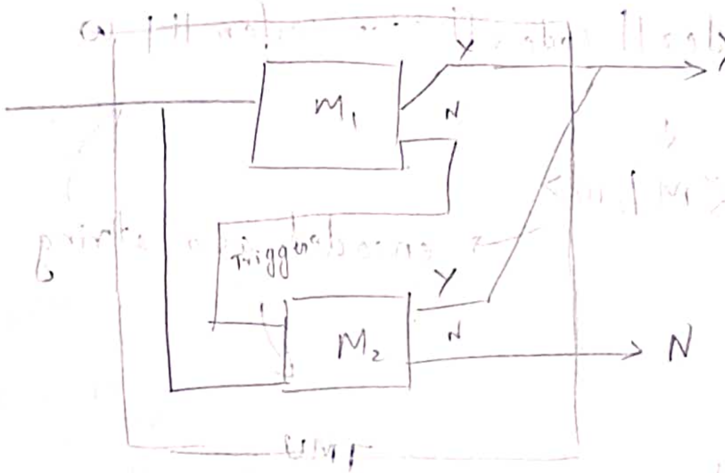
Recursive Enumerable: only yes

Recursive: yes, no

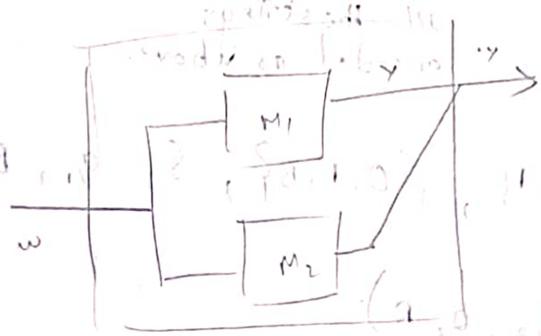
Thm: A language and its complement are recursive.



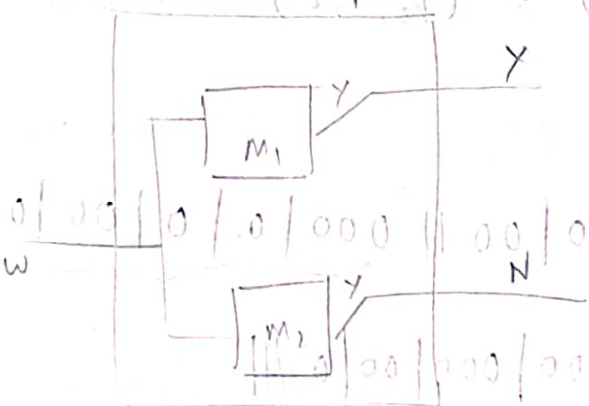
Thm 2: Union of two Languages → recursive is recursive, recursively enumerable is same.



$L_1 \cup L_2$



Thm: If a language and its complement are recursively enumerable, they are recursive.



→ TM code

$$\delta(q_i, X_j) = (q_k, X_l, D_m)$$

$$0^i 1 0^j 1 0^k 1 0^l 1 0^m$$

$\Gamma = \{0, 1, B\}$ — 0, 00, 000

$L - 0$

$R - 00$

111 code 1 11 code 2 11 code 3 11 .... code n 111 w

$\langle M, w \rangle$ → encoded as a string

L_U → recursively enumerable

TMU → Universal Turing machine → accepts all the strings encoded as above.

Q. $(\{q_1, q_2, q_3\}, \{0,1\}, \{0,1,B\}, \delta, q_1, B, \{q_2\})$

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

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

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

$\delta(q_3, B) = (q_3, 1, L)$

111 0 1 00 1 000 1 0 1 00 11 000 1 0 1 0 1 00 1 00 11 000 1 00 1 00 1 0 1 00 11 000 1 000 1 000 1 00 1 0 111

→ Consider a 3-tape Turing machine

<M;w>

copy

w

$w \in L(M)$ and w is accepted by M.

I/P

$0^i 1 0^j 1 0^k 1 0^l 1 0^m$

$\delta(q_i, X_j) = (q_k, X_l, D_m)$

$1 \le j \le 3$

$1 \le l \le 3$

$1 \le m \le 2$

$\Big\}$ first step, validate

Second step → copy w to second tape.

third step → copy the first transition function

now, we need to find 0 1 matches first char of w.

q_1 → start state

if not remove that transition and add in the next until you find one.

Then cut off $\cancel{0^i 1 0^j 1} 0^k \cancel{1 0^l 1 0^m}$

then skip 0^k ⇒ 0^l in second tape is written

and acc^n to 0^m, move is made.

left over is the state you need to find in the first step for which δ is defined.

if nothing is defined

* final state

→ Diagonal Language.

L_d j → string

i – TM

	1	2	3	4	5	6
1	1	0	1	1	0	0
2	0	1	0	1	1	0
3	1	0	1	1	0	0
4	0	0	0	0	1	1
5	1	0	1	0	1	1
6						

$L_d = \{ w_i \mid (i, i) = 0 \}$ → i^{th} string doesn't belong to i^{th} TM.

L_d is not RE → prove by contradiction.

$L_u \rightarrow R$

$L_d \rightarrow$ not RE

$\Rightarrow$ not R

$\Rightarrow \bar{L_d}$ not R

~~$\bar{L_d}$ not RE~~.

$\bar{L_d}$?

$L \to RE$
$\to R$

$\bar{L} \to R$

$\bar{L} \to RE?$

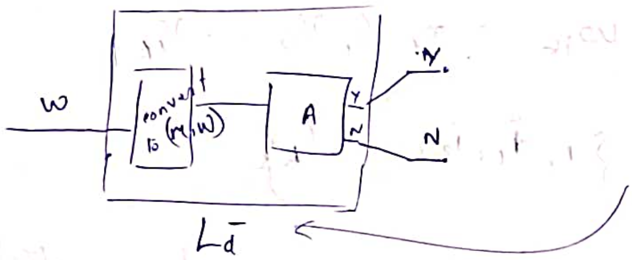
$L \to RE$, not R

$\bar{L} \to$ not R

$\bar{L} \to RE?$

not RE

Claim : L_u is not R



$L_d = \{ w \mid (i,i) = 0 \}$
$L(M_i) \neq w_i$

$L_{\bar{d}} = \{ w \mid (i,i) = 1 \}$

$L_u = \{ \langle M, w \rangle \mid w \in L(M) \}$

we can reduce $L_{\bar{d}}$ to L_u

$>$ and $L_{\bar{d}}$ is not R

$\Downarrow$

L_u is not R $\Leftarrow$ This algorithm is hypothetical

→ Post Correspondence problem.

PCP is undecidable

$\Rightarrow$ PCP is not R

$A_{(w_i)}$	$B_{(x_i)}$
1	111
10111	10
10	0

$w_{i_1} w_{i_2} w_{i_3} \cdots w_{i_k}$

$= x_{i_1} x_{i_2} \cdots x_{i_k}$

$\{ i_1, i_2, \ldots, i_k \}$
$< n$

A 10 | 111 | 11 | 10

B 10 | 111 | 111 | 0

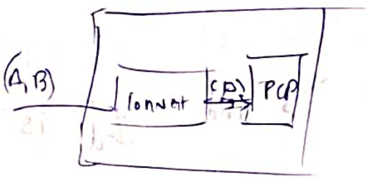
$\{2, 1, 1, 3\}$

Modified PCP

$w_1 w_{i_1} w_{i_2} \cdots w_{i_k} = x_1 x_{i_1} x_{i_2} \cdots x_{i_k}$

$\{1, i_1, i_2, \ldots, i_k\}$

means you always starts with the first string.



~~This~~

	A	B
	w_i	x_i
1		
2		
3		
⋮		
n		

	C	D
	y_i	z_i
0		
1		
2		
3		
4		
⋮		
n		
n+1		

y_i is constructed from w_i by adding ¢ ~~before~~ After every string (char)

z_i ¢ → ¢ before every char

Eg:

A	B
1	111
10111	10
10	0

C	D
¢1¢	¢1¢1¢1
1¢	¢1¢1¢1
1¢0¢1¢1¢1¢	¢1¢0
1¢0¢	¢0
$	¢$

$y_0 = ¢y_1 \quad z_0 = z_1$

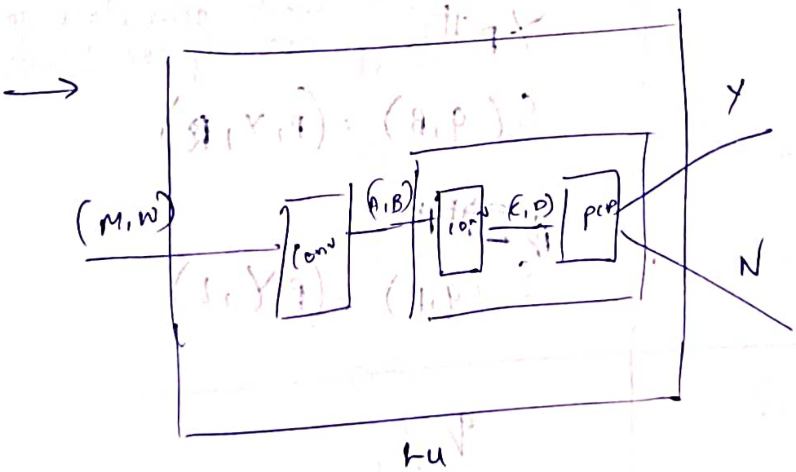
$y_{n+1} = \$ \quad z_{n+1} = ¢\$.$

consider eg:

	A	B
1	10111	10
2	1	111
3	10	0

$\Rightarrow$

	C	D
0	¢1¢0¢1¢1¢1¢	¢1¢0
1	1¢0¢1¢1¢1¢	¢1¢0
2	1¢	¢1¢1¢1
3	1¢0¢	¢0
4	$	¢$



if we can reduce Lu to MPCP to PCP and we know that Lu is not R

so PCP's algorithm is hypothetical.

	A	B
I	$\#$	$\# q_1 w \#$
II	X	X
	$\#$	$\#$
		$X \in \Gamma$
		→ in this context $\#$ is B.
III	qX	Yp
		$\delta(q, X) = (p, Y, R)$
$Z \in \Gamma - \{B\}$	ZqX	pZY
		$\delta(q, X) = (p, Y, L)$
		convert X to Y on your right → move left by skipping the prev char → in state p → ready to read the char on right.
	$q\#$	$Yp\#$ → shows after blank you can only see blank
		$\delta(q, B) = (p, Y, R)$
	$Zq\#$	$pZY\#$
		$\delta(q, B) = (p, Y, L)$
IV	XqY	q
$X, Y \in \Gamma$	Xq	q
	qX	q
		$q \in F$
V	$q\#\#$	$\#$
$q \in F$		

δ	0	1	B
q_1	$(q_2, 1, R)$ 1	$(q_2, 0, L)$ 4	$(q_2, 1, L)$ 6
q_2	$(q_3, 0, L)$ 5	$(q_1, 0, R)$ 2	$(q_2, 0, R)$ 3
q_3	—	—	—

$w = 01$

	A	B	
I	#	$\# q_1 01 \#$	
II	X	X	$X \in \Gamma$
	#	#	
	$q_1 0$	$1 q_2$	1
	$q_2 1$	$0 q_1$	2
	$q_2 \#$	$0 q_2 \#$	3
	$0 q_1 1$	$q_2 0 0$	4
	$1 q_1 1$	$q_2 1 0$	4
	$0 q_2 0$	$q_3 0 0$	5
	$1 q_2 0$	$q_3 1 0$	5
	$0 q_1 \#$	$q_1 0 1 \#$	6
	$1 q_1 \#$	$q_1 1 1 \#$	6

$0\,q_3\,0$	q_3
$0\,q_3\,1$	q_3
$1\,q_3\,0$	q_3
$1\,q_3\,1$	q_3
$0\,q_3$	q_3
$q_3\,0$	q_3
$1\,q_3$	q_3
$q_3\,1$	q_3
q_3 ###	#

\# $q_1 0$ 1 # 1 q_2 1 # 1 0 q_1 #

\# q_1 0 1 # 1 q_2 1 # 1 0 q_1 # 1

Q. $\delta(q_1, 1) = (q_3, 0, R)$ ①

$\delta(q_3, 0) = (q_1, 1, R)$ ②

$\delta(q_3, 1) = (q_2, 0, R)$ ③

$\delta(q_3, B) = (q_3, 1, L)$ ④

A	B	
#	$\# q_1 1011 \#$	
$\times$	$\times$	$\times \in \Gamma$
#	#	
$q_1 1$	$0 q_3$	1
$q_3 0$	$1 q_1$	2
$q_3 1$	$0 q_2$	3
$0 q_3 \#$	$q_3 01 \#$	4
$1 q_3 \#$	$q_3 11 \#$	
$0 q_2 0$	q_2	
$0 q_2 1$	q_2	
$1 q_2 0$	q_2	
$1 q_2 1$	q_2	
$0 q_2$	q_2	
$q_2 0$	q_2	
$1 q_2$	q_2	
$q_2 1$	q_2	
$q_2 \#\#$	$q_2 \#$	

$\# q_1 1 \mid 011\#0 \mid q_3 011\#01 \mid q_1 \mid 11\#010 \mid q_3 1 \mid 1\#010 \mid 0 q_2 1 \mid$

$\# q_1 1 \mid 011\# 0 \mid q_3 011\#01 \mid q_1 \mid 11\#010 \mid q_3 1 \mid 1\#010 \mid 0 q_2 1 \mid \#010 q_2$