Examples for Identify Type of language :-

- $L = \{a^n b^n c^p \mid n > p\} \longrightarrow$ CSL (2 cmp b/w n & p and $|a| = |b|$ & $|c| > |b|$)

$L = \{a^n b^n c^p \mid n, p \geq 0\} \longrightarrow$ CFL (1 cmp, $|a| = |b|$)

- $L = \{a^m b^n c^{m+n} \mid m, n \geq 0\} \longrightarrow$ CFL ($|a| + |b| = |c|$)

- $L = \{a^m b^n c^p \mid m = n \text{ or } n = p\} \longrightarrow$ CFL

Due to OR it has 1 CMP at a time.

$L = \{a^m b^n c^p \mid m = n \ \& \ n = p\} \longrightarrow$ CSL

String matching :-

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

CFL CSL

OR OR

$L = \{w_1 w_2 \mid w_2 = w_1^R, w \in \{a, b\}^*\}$

$L = \{w_1 w_2 \mid w_1 = w_2, w \in \{0,1\}^*\}$

- $L = \{ w c w^R w^R \mid w \in \{0,1\}^* \}$ — CSL

- $L = \{ w_1 w_2 w_3 \mid w_2 = w_1^R$ or $w_3 = w^R \}$ — CFL

1 CMP at a time

- $L = \{ w_1 w_2 w_3 \mid w_2 = w_1^R$ & $w_3 = w_1^R \}$ — CSL

2 CMP at a time

Pumping Lemma :-

L is Regular $\Rightarrow$ L satisfy the P. Lemma for Reg. language

so, if,

L doesn't satisfy P. Lemma $\Rightarrow$ L is Non-Regular

- we can't say that if L satisfy pumping lemma then L is Regular.

i.e Some Non-Regular Lang. also satisfy Pumping Lemma.

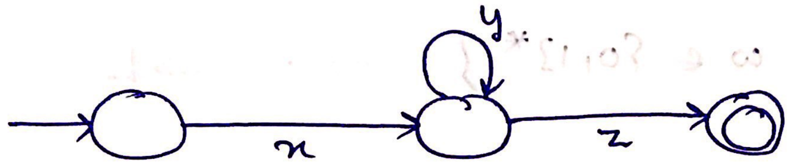
Defination :- $\exists$ FA with N-states which accept L, if

M accepts $|w| \geq N \Rightarrow w = xyz \in L(M)$

where $|y| \geq 1$

then $xy^iz \in L(M)$

or $xy^*z \in L(M)$



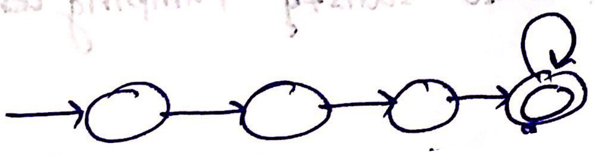
NOTE:

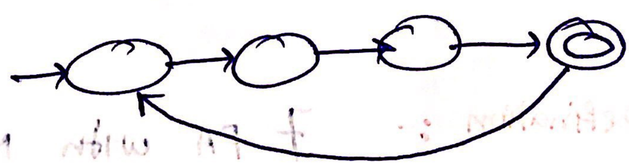
5 state FA

- accept string of $|w| = 5$ — [Infinite]
- accept string of $|w| = 4$ — [Infinite or finite]
- accept only string of $|w| = 4$ — [Finite]

NOTE:- for an infinite language, with N state FA accepting that language.

then $\exists w$ such that $(N \leq |w| \leq 2N-1)$ which definitely accept by the FA.





$|w| = N$
$= 4$

$|w| = 2N-1$
$= 8-1 = 7$

- Minimum pumping length for a language is always unique.

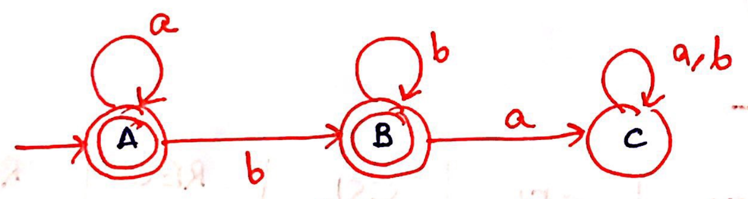
Myhill - nerode Theorem :-

(L is Regular $\longleftrightarrow$ no. of M-N Eq classes is Finite)

$\Rightarrow$ L is non-regular then no. of M-N Eq classes is Infinite.

(No. of M-N Eq classes = no. of states in minimal DFA)

(M-N Eq = RE of individual states)



r_A — a^*

r_B — a^*bb^*

r_C — $a^*bb^*a(a+b)^*$

"L is regular if and only union of sum of equivalent classes of right invariance of myhill nerode index $x R y$ iff $\delta^*(q,x) = \delta^*(q,y)$"

⇒ Properties :-

- $E_1, E_2, E_3 \ldots E_n$ are equivalence classes

(i) $|E_i| \geq 1$ (Equivalence class can't empty i.e ϕ)

(ii) $i \neq j \Rightarrow E_i \cap E_j = \phi$

(iii) $\bigcup_{i=1}^{n} E_i = \Sigma^*$ & $\bigcap_{i=1}^{n} E_i = \phi$

NOTE :-

$x \equiv y$ iff $\delta^*(q_0, x) = \delta^*(q_0, y)$

$(x \equiv y \rightarrow xz \equiv yz)$

⇒ Closure Property :-

opn	REG	DCFL	CFL	CSL	REC	RE
Union $\cup$	✓	✗	✓	✓	✓	✓
Intersection $\cap$	✓	✗	✗	✓	✓	✓
Complement $\bar{L}$	✓	✓	✗	✓	✓	✗
concat $\cdot$	✓	✗	✓	✓	✓	✓
Kleen closure $*$	✓	✗	✓	✓	✓	✓

op^n	REG	DCFL	CFL	CSL	REC	RE
Transpose or reversal L^R	✓	✗	✓			
substitution or Homomorphism $h(L)$	✓	✗	✓			
inverse substitution $h^{-1}(L)$	✓	✓	✓			
$L \cup$ Regular	✓	✓	✓	✓	✓	✓
$L \cap$ Regular	✓	✓	✓	✓	✓	✓
$L -$ Regular	✓	✓	✓	✓	✓	✓

$L_1 - L_2 = L_1 \cap \overline{L_2}$

$L_1 \oplus L_2 = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$

$L_1 \downarrow L_2 = \overline{L_1} \cap \overline{L_2} = \overline{L_1 \cup L_2}$
NOR

$L_1 \uparrow L_2 = \overline{L_1 \cap L_2} = \overline{L_1} \cup \overline{L_2}$
NAND

$L_1 \rightarrow L_2 = \overline{L_1} \cup L_2$

$L_1 \leftrightarrow L_2 = (L_1 \cap L_2) \cup (\overline{L_1} \cap \overline{L_2})$

Operators Precendence

$( ) > L^c > \cap > \cup > \rightarrow > \leftrightarrow$

Examples:-

1. L_1 is DCFL & L_2 is DCFL

$$L_1 \cup L_2 = DCFL \cup DCFL \quad \text{(Push-up DCFL)}$$

$$= CFL \cup CFL = CFL$$

So, $(L_1 \cup L_2 = CFL)$

DCFL $\subseteq$ CFL

2. DCFL $\cup$ Regular

(DCFL $\cup$ Regular = DCFL)

(No Push-up required)

3. RE − REC

$= RE \cap \overline{REC}$

$\doteq (RE \cap REC\uparrow$

$= RE \cap RE$

$= (RE)$

$= RE - REC\uparrow$

$= RE - RE$

$= RE \cap \overline{RE}$

$= \phi$

FALSE method

Product automata :-

DFA1 $M_1(Q_1, \Sigma_1, \delta_1, q_{0_1}, F_1)$

DFA2 $M_2(Q_2, \Sigma_2, \delta_2, q_{0_2}, F_2)$

$$M_1 \times M_2 = (Q_3, \Sigma_3, \delta_3, q_{0_3}, F_3)$$

$Q_3 = Q_1 \times Q_2$ $\qquad$ $q_0 = (q_{0_1}, q_{0_2})$

$\Sigma_3 = \Sigma_1 = \Sigma_2$ $\qquad$ $\delta_3((A, D), a) = (\delta_1(A, a), \delta_2(D, a))$

eg:-

Q_1	a	b
A	B	A
B	B	B

Q_2	a	b
C	C	D
D	D	C

Q_3	a	b
AC	BC	AD
AD	BD	AC
BC	BC	BD
BD	BD	BC

- **Substitution** : $L = \{aa, bab\}$, $h(a) = 00$, $h(b) = 10$

$h(L) = \{0000, 100010\}$

so, L is finite , h(L) is also finite so,

h(L) is also regular.

Similarly h(L) is regular for infinite lang also.

> **NOTE :-** $h(\phi) = \phi$
> $h(\epsilon) = \epsilon$

- **Inverse** : $L = \{1010, 101\}$, $h(a) = 0$ $h(b) = 1$

$h(c) = 01$

$h^{-1}(L) = \{1010, 101\} = \{baba, bab\}$

$= \{1010, 101\} = \{bca, bc\}$

and so on,

so, $h^{-1}(L) = \{baba, bab, bca, bc, \ldots\}$

Thus, $|h^{-1}(L)| > |L|$; $|h^{-1}(L)| = |L|$; $|h^{-1}(L)| < |L|$

so, $|h^{-1}(L)|$ can be anything we can't predict

- CFL $\cap$ CFL = CSL

eg:- $\{a^n b^n c^m\} \cap \{a^n b^m c^m\} = \{a^n b^n c^n\}$

CFL CFL CSL

- $\overline{\text{CFL}} \neq$ CFL

eg: $L = \{a^n b^m c^p \mid m = n \text{ or } n = p\}$ CFL

$\Big\downarrow$ compliment

$\overline{L} = \{a^n b^m c^p \mid m \neq n \ \& \ n \neq p\}$ CSL

2 cmp

- CFL $L_1 \xrightarrow{\text{is}} R$ and $L_2 \xrightarrow{\text{is}} R$ ~~and~~ so, $L_1 \cup L_2$ will regular

L is R & L_2 is R $\longrightarrow$ $L_1 \cup L_2$ is R

eg: $\overset{R}{\Sigma^*} \cup \overset{\text{CFL}}{a^n b^n} = \underset{R}{\Sigma^*}$

$L_1 \cup L_2$ is R $\longrightarrow$ L is R & L_2 is R ✗

so, if $(L_1 \cup L_2)$ is Regular then;

- L_1 is Reg, L_2 is Reg
- L_1 is Reg, L_2 is non-Reg
- etc

- L_1 is REG & L_2 is REG $\rightarrow$ $L_1 \cap L_2$ is Reg

- L_1 is REG & L_2 is REG $\rightarrow$ $L_1 \cdot L_2$ is Reg

so, if $L_1 \cdot L_2$ is regular then atleast (L_1 or L_2) one should be regular.

Q.

(i) $L_1 \cdot L_2$ Reg & L_1 is Reg then L_2 is <u>Can't say</u>

(ii) $L_1 \cdot L_2$ Reg & L_1 is non-Reg then L_2 is <u>Reg</u>

- L is Reg $\rightarrow$ L^* is reg

L^* reg $\rightarrow$ L is Reg ✗

eg:- $\underbrace{(\{a^{n^2}\})}_{\text{Non Reg}}{}^{*} = (\epsilon + a + a^4 + \dots)^* = \underbrace{a^*}_{\text{Reg}}$

⇒ Decidability :-

	REG	CFL	CSL	REC	RE
Membership	✓	✓	✓	✓	✗
Emptiness	✓	✓	✗	✗	✗
Finiteness	✓	✓	✗	✗	✗
Equivalence	✓	✗	✗	✗	✗
Regularity	✓	✗	✗	✗	✗
Ambiguity	✓	✗	✗	✗	✗
Completeness	✓	✗	✗	✗	✗
Disjointedness	✓	✗	✗	✗	✗

* Decidability means , whether a algorithm exist for finding solution for a problem.

Membership problem :- Given L, $w \in \Sigma^*$, check whether w belongs to L or not.

Emptiness problem :- $L = \phi$ or not

Finiteness problem :- Given L, checks whether L is finite or not.

Equivalence problem :- Given L_1, L_2, checks $(L_1 = L_2)$

Regularity problem :- Given L, is regular lang or not

Ambiguous problem :- L is ambiguous or not

Completeness problem :- $L = \Sigma^*$ or not

Disjointedness problem :- Given L_1 and L_2

$L_1 \cap L_2 = \phi$ or not

NOTE:-

(Problem is decidable $\longleftrightarrow$ L(Problem) is REC)

- Recursive language is <u>decidable lang</u>

- Recursive Enumerable is <u>semi-decidable lang</u>

Q. which of the following is REC lang?

(i) $\{ (w, M) \mid M \text{ is FA}, \ w \in L(M) \}$

(ii) $\{ (G_1, G_2) \mid G_1 \text{ and } G_2 \text{ are type-2 Grammer}, \ L(G_1) = L(G_2) \}$

(iii) $\{ r \mid r \text{ is regular expression } \& \ L(r) \text{ is ambiguous} \}$

(iv) $\{ (M_1, M_2) \mid M_1 \text{ and } M_2 \text{ are PDA}, \ L(M_1) \cap L(M_2) = \emptyset \}$

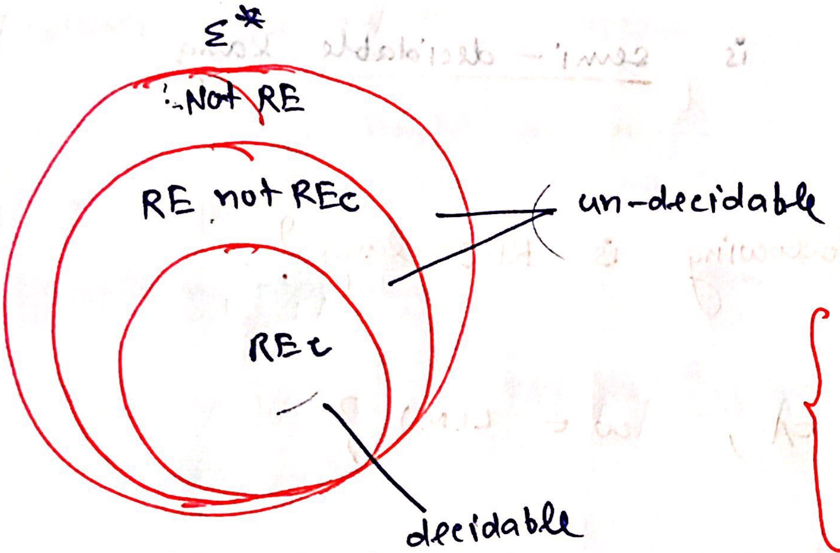
* (i) and (iii) problem are decidable so they are REC

M is FA so L(M) is regular and membership problem in REG is decidable

NOTE:- The problem which are decidable, converse of those problem is also decidable.

if P = (decidable)

then $\overline{P}$ = decidable



$$\left\{ \begin{array}{l} \overline{REC} = REC \\ \overline{RE \text{ not } REC} = \text{Not } RE \end{array} \right\}$$

Q. $L = \{ (G_1, G_2) \mid G_1$ and G_2 are type 2 grammar, $L(G_1) = L(G_2) \}$

(a) Not RE

(b) REC

(c) ✓ RE not REC

(d) REG

Q. $L = \{ (G_1, G_2) \mid G_1$ and G_2 and type-2 grammar $L(G_1) \neq L(G_2) \}$

Predict Language

(a) Not RE ✓

(b) REC

(c) RE not RE

(d) REG

Rice Theorem:-

"It states that all non-trivial; semantic properties of programs are undecidable". A property is non-trival if it is neither true for every computable function, nor false for every computable function.

$\{ T \mid T$ is Turing m/c. $L(T)$ is RE $\}$ — Not REC (undecidable)

Application of FA:-

(i) Lexical analysis (Pattern matching) :- compiler application

(ii) Text editor app ⇒ Search, replace, spell checker, UNIX GREP (General RE processor)

(iii) Sequential ckt ⇒ Melliy or moore machine

⇒ Variants of FA :-

1. Multi - tape FA

2. Multi- head FA

3. 2-way Automata (2-DFA / 2-NFA) (FA + Right/Left)
(FA + Read/write)

4. 1-way TM

5. FA + 1 counter (Counter Automata) > FA eg:- $(a^n b^n)$
TM ≡ FA + 2 counter > 1CA > FA

6. FA + 1 stack (PDA) > FA + 1C > FA eg:- (ww^R)
FA + 2C ≡ TM ≡ FA + 2 stack (2-PDA) > CA > FA

7. FA + Queue (Queue Automata)

Pumping Lemma (For Regular Language) :-

If A is a Regular language, then A has a Pumping length 'P' such that any string 's' where $|s| \geq P$ may be divided into 3 parts $s = xyz$ such that the following condition must be true :

1. $xy^iz \in A \quad \forall \ i \geq 0$
2. $|y| > 0$
3. $|xy| \leq P$

Steps to prove L to be Non regular (contradiction) :-

- Assume that A is Regular
- It has to have a pumping length (say P)
- All strings longer than P can be pumped $|s| \geq P$
- Now find a string 's' in A such that $|s| \geq P$
- Divide s into xyz
- Show that $xy^iz \notin A$ for some i
- Then consider all ways that s can be divided into xyz
- Show that none of these can satisfy 3 pumping condition at the same time
- Hence S cannot be Pumped, so our assumption is wrong.