16-07-2021

CLOSURE PROPERTIES OF CFL and DCFL

Subset -

CFL

$$\underset{CSL}{\{a^n b^n c^n\}} \subseteq \underset{CFL}{\{a^n b^n c^m \mid n, m \geq 0\}}$$

May or may not be CFL

Not closed

DCFL

$$\underset{CSL}{\{a^n b^n c^n\}} \subseteq \underset{DCFL}{\{a^n b^n c^m\}}$$

May or may not be DCFL

Not closed

Union -

CFL

$L_1 = CFL = S_1$

$S \rightarrow S_1 / S_2$

$L_2 = CFL = S_2$

$L_1 \cup L_2$

Closed

DCFL

$$L = \underset{DCFL}{\{a^n b^{2n}\}} \cup \underset{DCFL}{\{a^n b^{3n}\}} = \text{CFL but not DCFL}$$

Not closed

Intersection -

CFL

$$\underset{CFL}{\{a^n b^n c^m\}} \cap \underset{CFL}{\{a^n b^m c^m\}}$$

$$= \underset{CSL}{\{a^n b^n c^n\}}$$

Not closed

DCFL

$$\underset{CFL}{\{a^n b^n c^m\}} \cap \underset{CFL}{\{a^n b^m c^m\}}$$

$$= \underset{CSL}{\{a^n b^n c^n\}}$$

Not closed.

Intersection with Regular -

CFL $\cap$ Regular = Regular = DCFL $\cap$ Regular

$$\{a^n b^n\} \cap \{a+b\}^* = \{a^n b^n\}$$

Always CFL; but may or maynot be Regular

Smaller set Bigger set

$$\{a^n b^n\} \cap \{a^* b^*\} = \{a^n b^n\} \text{ CFL}$$

$$\{a^n b^n\} \cap \{ab\} = \{ab\} \text{ Regular}$$

Complement -

You cannot complement NFA

CFL $\leftrightarrow$ NPDA

$$L = \underbrace{\Sigma^* - \{\overbrace{ww \mid w \in \{a,b\}^*}^{\text{NON-CFL}}\}}_{\text{CFL}}$$

NOT CLOSED

Complement of $\{ww\}$ is CFL

DCFL $\leftrightarrow$ DPDA

$\hookrightarrow$ can be complemented

CLOSED

Difference -

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

CFL

NOT CLOSED

DCFL

NOT CLOSED

Q. If L_1 and L_2 are context free languages and R is a regular set, one of the languages below is not necessarily a context free languages. Which one?

a) $L_1 L_2$

b) $L_1 \cap L_2$

c) $L_1 \cap R$

d) $L_1 \cup L_2$

closed or not closed

	OPERATION	CFL	DCFL	
1.	Union	✓	✗	→ closed → Not closed
2.	Concatenation	✓	✗	
3.	Kleene Closure	✓	✗	
4.	Subset	✗	✗	
5.	Complement	✗	✓	
6.	Intersection	✗	✗	
7.	Intersection with Regular Language	✓	✓	
8.	Regular − L	✗	✓	Reg $\cap L^c$ → may or may not be CFL
9.	Difference $(L_1 - L_2)$	✗	✗	
10.	L − Regular	✓	✓	$L \cap \bar{R} = L$
11.	Reversal	✓	✗	
12.	Substitution	✓	✗	
13.	Homomorphism	✓	✗	
14.	Inverse Homo.	✓	✓	
15.	L_1/L_2	✗	✗	
16.	L/Regular	✓	✓	
17.	L ∪ Regular	✓	✓	
18.	Prefix	✓	✓	
19.	Suffix	✓	✓	

$L_1 = \{ ww \mid w \in \{a,b\}^* \}$ ⇒ NOT-CFL (CSL)

$L_2 = \Sigma^* - L_1$ ⇒ CFL

Q. Context free languages are -

✓ a) closed under union
b) closed under complementation
c) closed under intersection
✓ d) closed under Kleene closure

Q. Which of the following is false?

$L_1, L_2 \Rightarrow DCFL$ $\quad$ $L_5 \Rightarrow$ Regular
$L_3, L_4 \Rightarrow CFL$

a) $(L_3 \cup L_4)^c$ need not be CFL
b) $L_1^c - L_5 \rightarrow DCFL$
c) $(L_3^R \cup L_4) \cap L_5 \rightarrow CFL$
✓ d) $L_3^c \cap L_4^c \rightarrow CFL$

$L_3 \cup L_4 \Rightarrow (CFL \cup CFL)^c \Rightarrow$ may or may not be CFL ✓

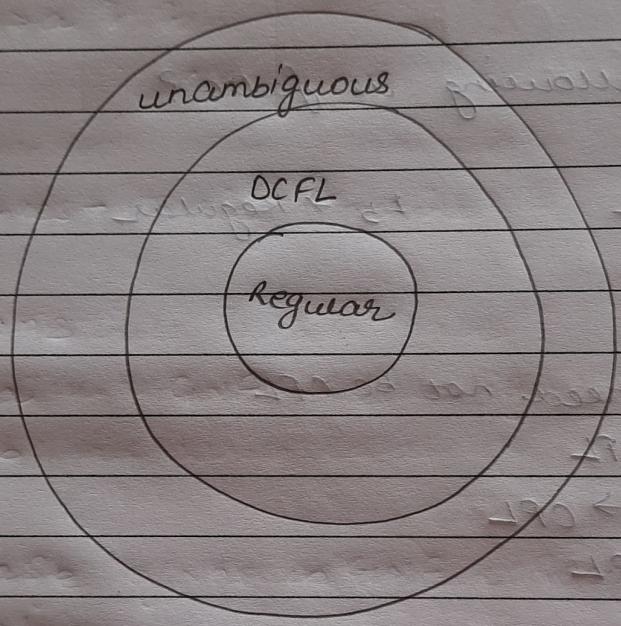
$L_1^c \Rightarrow DCFL$ and $L_5 = Reg$
$L_1^c \cap L_5^c \Rightarrow DCFL \cap Reg = DCFL$ ✓

$L_3^R \Rightarrow (CFL \cup CFL)$
$CFL \cap Reg \Rightarrow CFL$ ✓

$L_3^c \Rightarrow$ may or may not be CFL $\cap L_4^c$??

NOTE:-

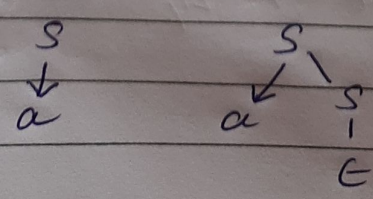
Language generated by LR(1) grammar and LL(1) grammar is DCFL



** Q. Which of the following is false?

a) Every DCFL is unambiguous
b) Every regular language is unambiguous
✓c) Every regular grammar is unambiguous
d) None

E.g $S \to aS \mid a \mid \epsilon$



Regular grammar can be ambiguous.

NOTE :-

NPDA

Empty Stack = Final State

DPDA

Empty Stack < Final state

Prefix Property -

If any language have prefix property then only it is accepted by Empty stack method of DPDA (Final state also)

If any language does not have prefix property then it is accepted by Final state method but not by empty stack method.

Example -

DPDA → E.S ✓
→ F.S ✓

$L_1 = \{a^n b^n \mid n \geq 1\}$ → Prefix Property

$= \{ab, aabb, \quad\}$

Proper Prefixes does not belong to lang.

ϵ, a | ϵ, aa, aab

$L_2 = \{a^n b^n \mid n \geq 0\}$ → Not having Prefix Property

$= \{\epsilon, ab, aabb, \quad\}$

DPDA → E.S ✗
→ F.S ✓

ϵ

Proper prefix belong to the language

Normal Forms of CFG

Chomsky Normal Form

$A \to BC$

$A \to a$

Greibach Normal Form

$A \to aV^*$

$a \in T$

Examples -

1) $S \to aAB$

$A \to a$ GNF

$B \to b$.

2) $S \to aSb$ Not in GNF

$S \to ab$

Q. Convert CFG into GNF grammar

$S \to aSb / ab$

1) Convert total RHS into non-terminals

$S \to ASB / AB$

$A \to a$

$B \to b$

2) Replace first non-terminal of RHS.

$S \to aSB / aB$
$A \to a$
$B \to b$

Q. Convert the following to GNF.

* Left Recursion must not be there

$S \to Sa / b$

If left recursion present in the grammar, then it should be eliminated to convert to GNF grammar.

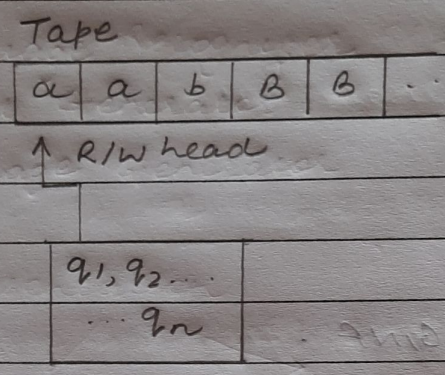
$S \to bS'$
$S' \to aS' / \epsilon$

$S \to bS' / b$ GNF
$S' \to aS' / a$

L-50

7-07-21

TURING MACHINE



Turing Machine

1) Infinite length tape
2) Turnaround capability (Moves in both Left and Right Side)
3) Read/Write capability

Language of TM is called Recursive Enumerable languages.

A TM can accept
a) Recursive Enumerable language
b) Context Sensitive language
c) Context Free language
d) Regular language

Turing machine is a mathematical model used to study capability of computer.

Turing Machine ≃ Computer

Decidable problems and undecidable problems

Undecidable problems - Not solvable by computers.

Decidable problems - Solvable by computers.

Formal Definition of Turing Machine -

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

- Q : Finite No. of states
- Σ : Input alphabet
- q_0 : Initial state
- F : Set of Final states
- B : Blank symbol
- Γ : Tape Alphabet E.g $\{a, b, X, Y, B\}$
- δ : Transition function.

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

Example :-

input Symbol Replaced alphabet

$\delta(q_1, a) = (q_2, X, L)$

Initial state Next State Direction

$\delta(q_2, b) = (q_3, Y, R)$

$\delta(q_3, X) = (q_4, Z, L)$

Types of Turing Machine

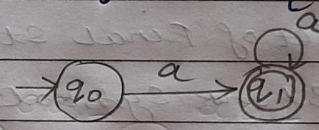
1) Language Recognizer
2) Output Generator
3) Language Generator

* Representation
  - Transition diagram
  - Transition table

* LANGUAGE RECOGNIZER

Q. Construct Turing machine for $L = \{a^n \mid n \geq 1\}$

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

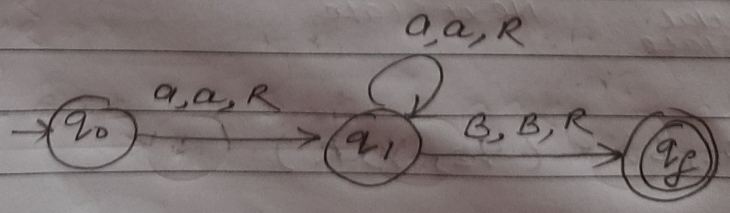


DFA

	a	B
→ q_0	(q_1, a, R)	
q_1	(q_1, a, R)	(q_f, B, R)
q_f	Halting state	

a	a	a	B

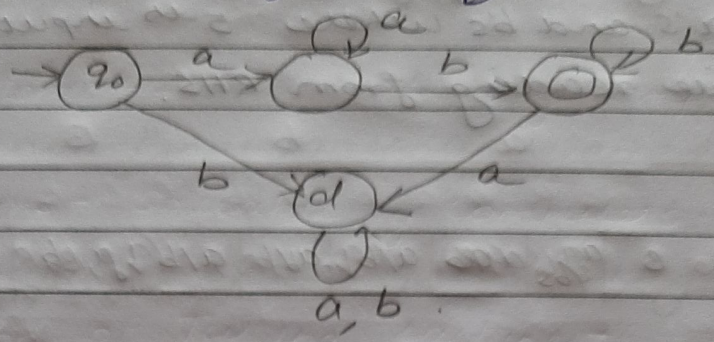
Transition table



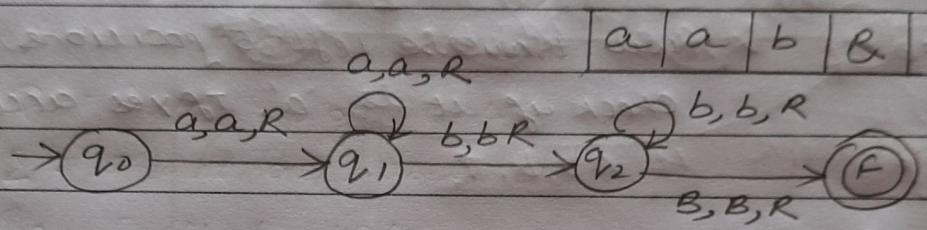
Transition diagram

Q. Construct Turing machine for $L = \{a^n b^m \mid n, m \geq 1\}$

Finite Automata

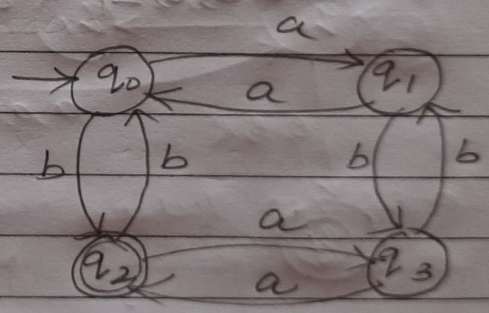


Turing Machine

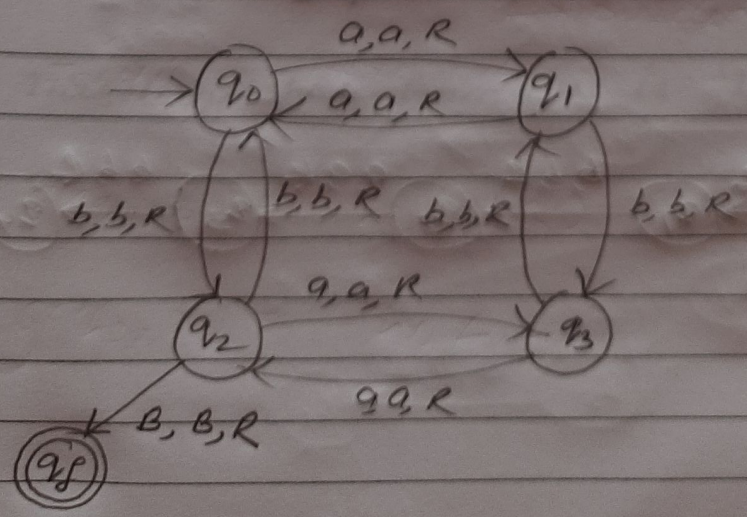


Q. Construct Turing Machine that accepts even as and odd b's.

Finite Automata



Turing Machine

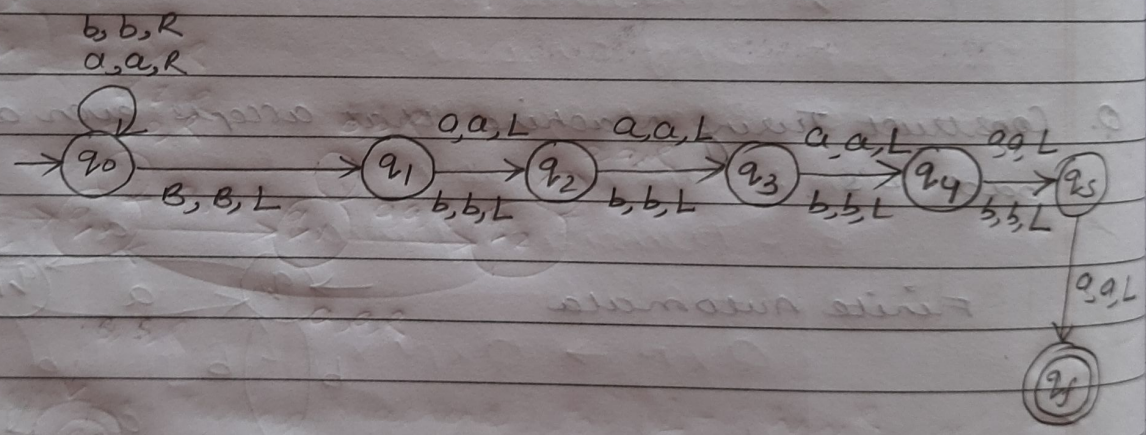


NOTE :- FA ⇒ 2 TYPES
PDA ⇒ 1 TYPE
TM ⇒ 3 TYPES

Q. Construct Turing machine that accepts all strings of a's and b's where 5th input symbol is 'a' while reading from RHS.

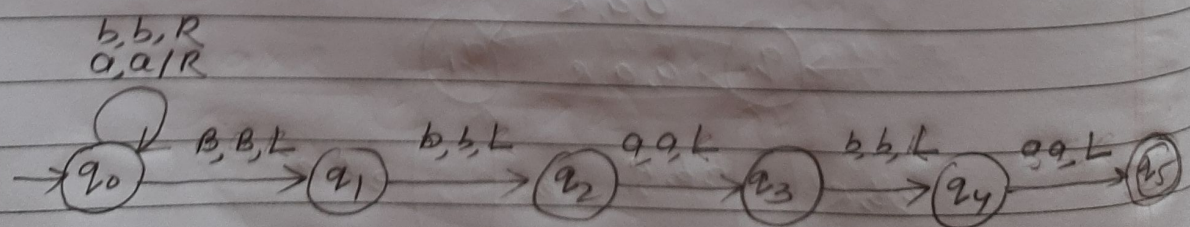
B	as & bs	a	a/b	a/b	a/b	a/b	B

Logic : Replace all symbols as it is upto first blank symbol.
Now traverse left (5 positions) to check if it is 'a'. If yes accept



Q. Construct Turing machine for L = (a+b)*abab

B	(a+b)*	a	b	a	b	B

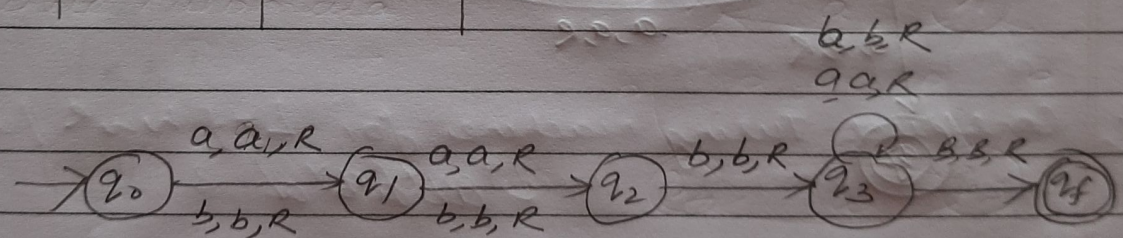


* Acceptance Methods -

- TM Halts in final state is accepted
- TM Halts in non-final state is rejected.

Q. Identify the language of the given Turing machine

	a	b	B
$\rightarrow q_0$	(q_1, a, R)	(q_1, b, R)	
q_1	(q_2, a, R)	(q_2, b, R)	
q_2	Halt	(q_3, b, R)	
q_3	(q_3, a, R)	(q_3, b, R)	(q_f, B, R)

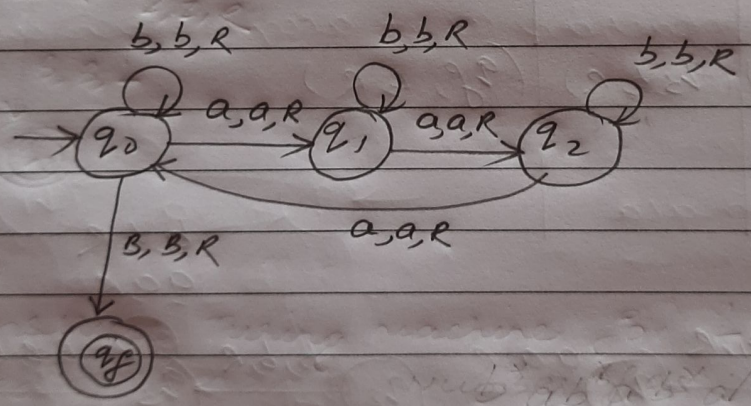


$(a+b)(a+b)\,b\,(a+b)^*$

Third symbol from LHS is 'b'

Q. Identify the language of the following Turing Machine

	a	b	B
$\rightarrow q_0$	(q_1, a, R)	(q_0, b, R)	(q_f, B, R)
q_1	(q_2, a, R)	(q_1, b, R)	Halt
q_2	(q_0, a, R)	(q_2, b, R)	Halt
q_f	HALT		



Number of as divisible by 3

Q. Construct Turing machine for $L = \{a^n b^n / n \geq 1\}$

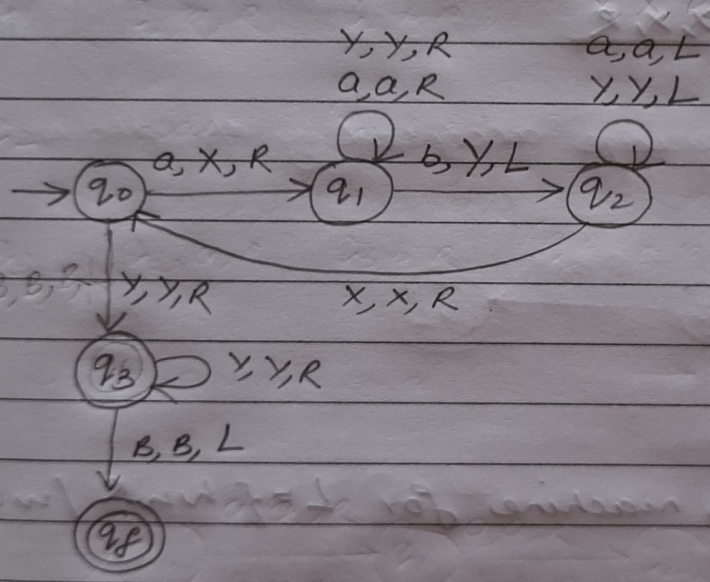
B	a	a	a	b	b	b	B

X → Y

Logic :

X X → Y Y

X X X Y Y Y

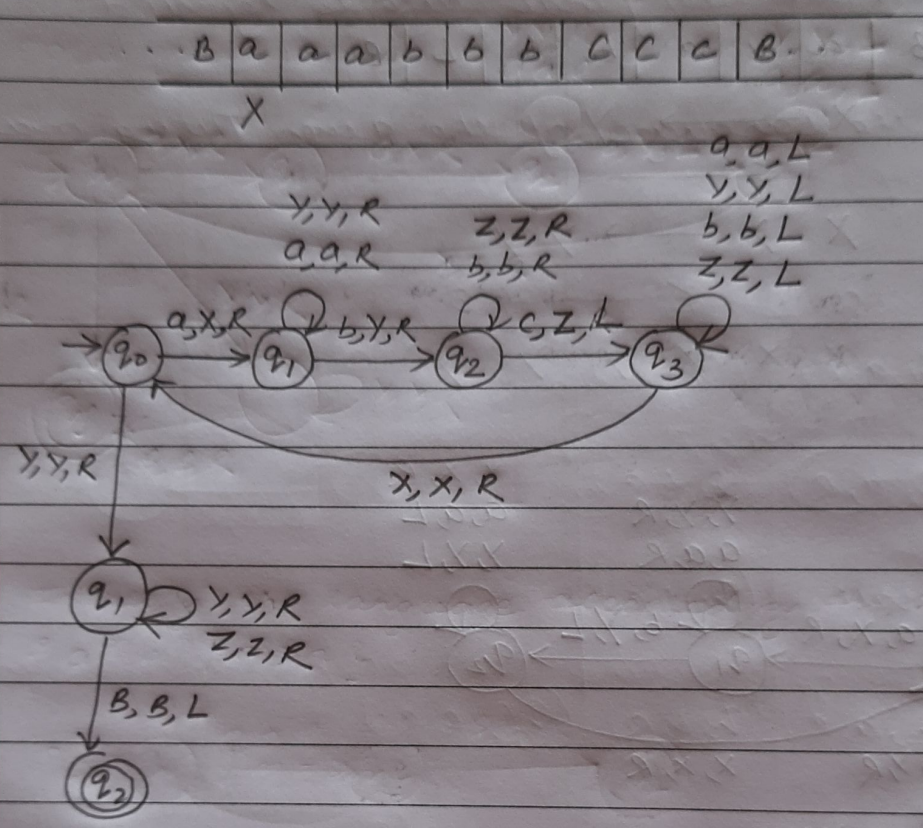


What are the halting states for -

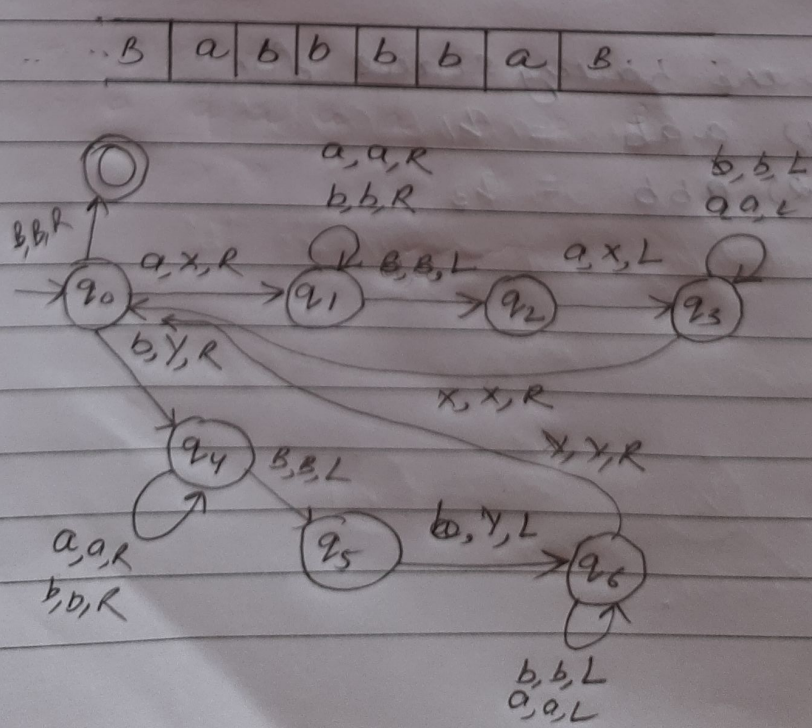
1) aab = q_1

2) abb = q_3

Q. Construct Turing machine for $L = \{a^n b^n c^n / n \geq 1\}$



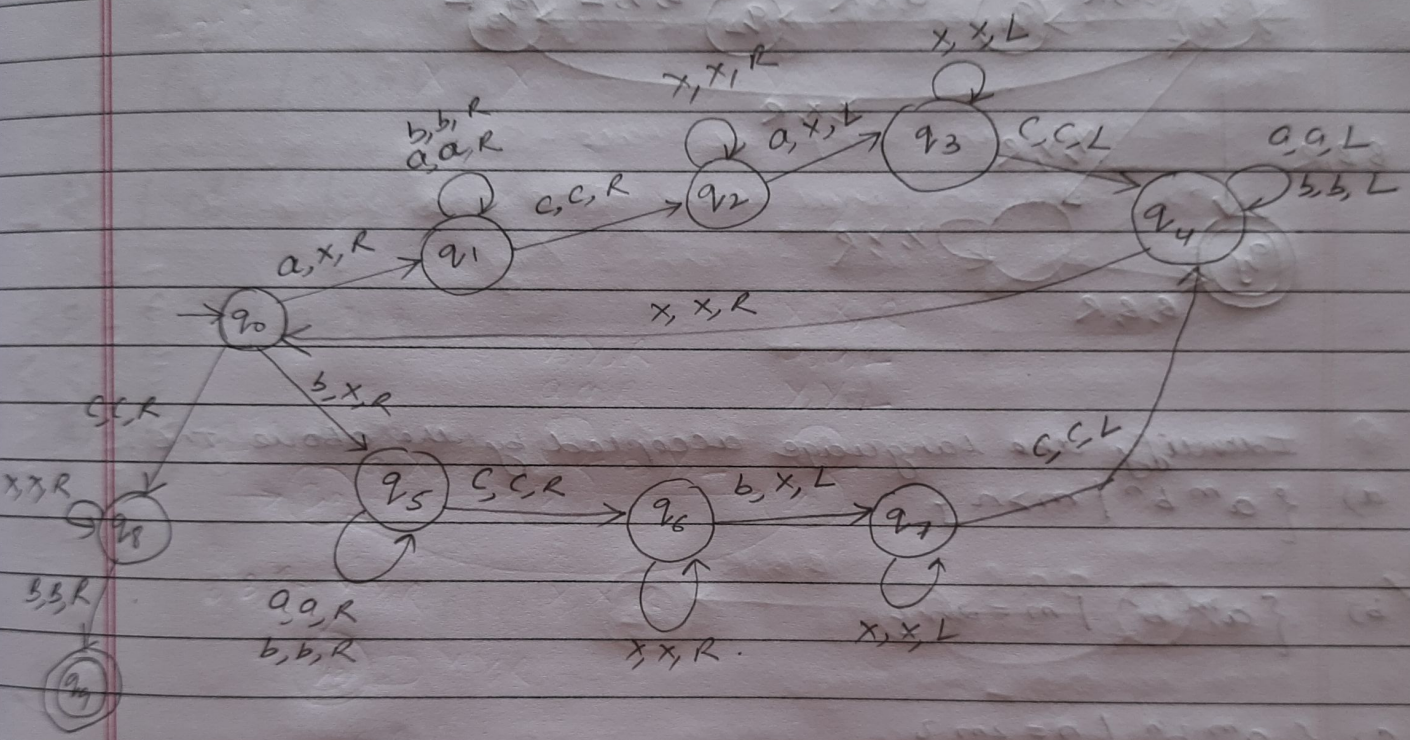
Q. Construct Turing machine for $L = \{ww^R / w \in \{a,b\}^*\}$



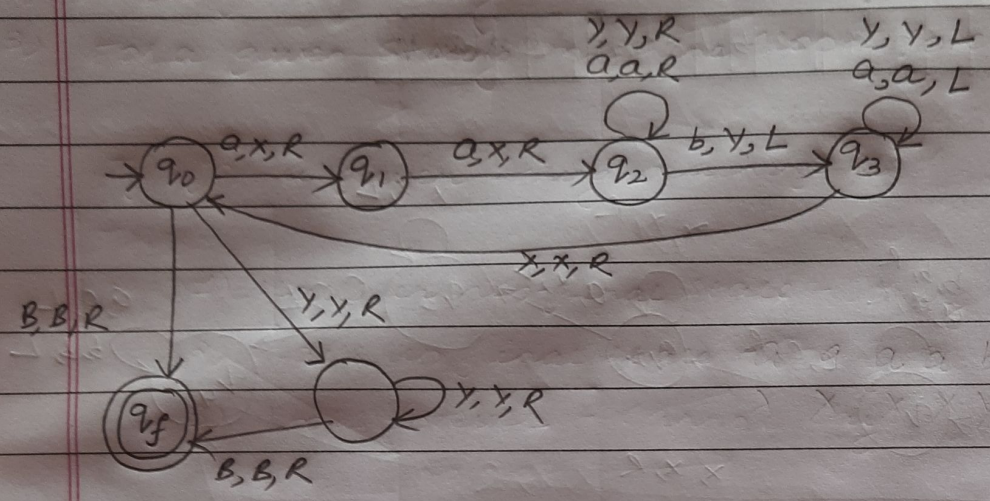
18-07-21

Q. Construct TM for $L = \{ wcw \mid w \in \{a, b\}^* \}$

-	a	b	a	b	c	a	b	a	b	...
	X					X				



Q. Consider the following Turing machine.



Identify the language accepted by the above TM:

a) $\{a^m b^n \mid m > n\}$

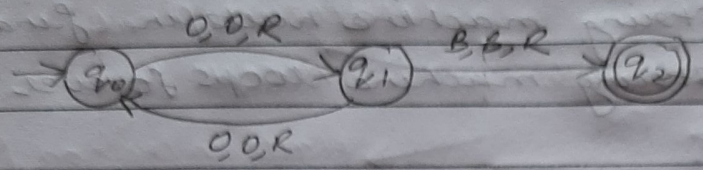
✓ b) $\{a^m b^n \mid m = 2n\}$

c) $\{a^m b^n \mid n = 2m\}$

d) $\{a^m b^n \mid m \geq n\}$

... B	a	a	a	a	b	b	B ...
	X	X	X	X	Y	Y	

Q. Identify the language of the following Turing machine.



a) $\{0^{2n} \mid n \geq 0\}$

✓ b) $\{0^{2n+1} \mid n \geq 0\}$

c) $\{0^n \mid n \geq 0\}$

d) $\{0^n \geq 1\}$

B	0	0	0				B

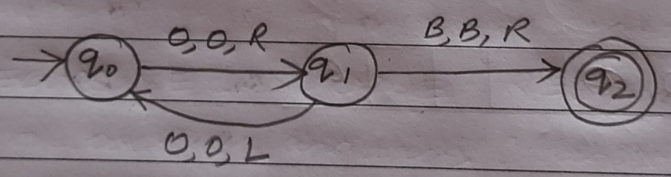
Possibilities for a Turing Machine

1) For a given string machine halts in non-final state.
2) For a given string machine halts in final state.
3) For a given string machine loops forever.

If for a given string machine

1) Halts in non-final state - String is rejected
2) Halts in final state - String is accepted
3) Loops forever - cannot say

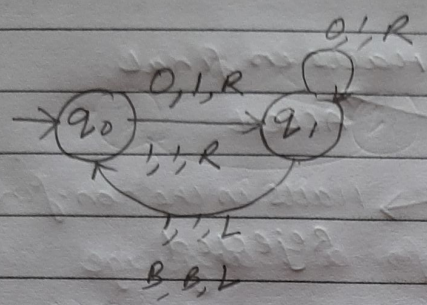
Q. Identify the language accepted by the following TM.



B	0	B
	X	

a) $L = \{00\}$
✓ b) $L = \{0\}$
c) $L = \{0^{2n} \mid n \geq 1\}$
d) $L = \{0^n \mid n \geq 0\}$

GATE 2003



✓a) M does not halt on any string in $(0+1)^+$

b) M does not halt on any string in $(00+1)^*$

c) M halts on all string ending with 0

d) M halts on all string ending in a 1

B	0	B		B	1	B

$1 \quad q_0 \to q_1$

$q_0 \to q_1$

Not halting

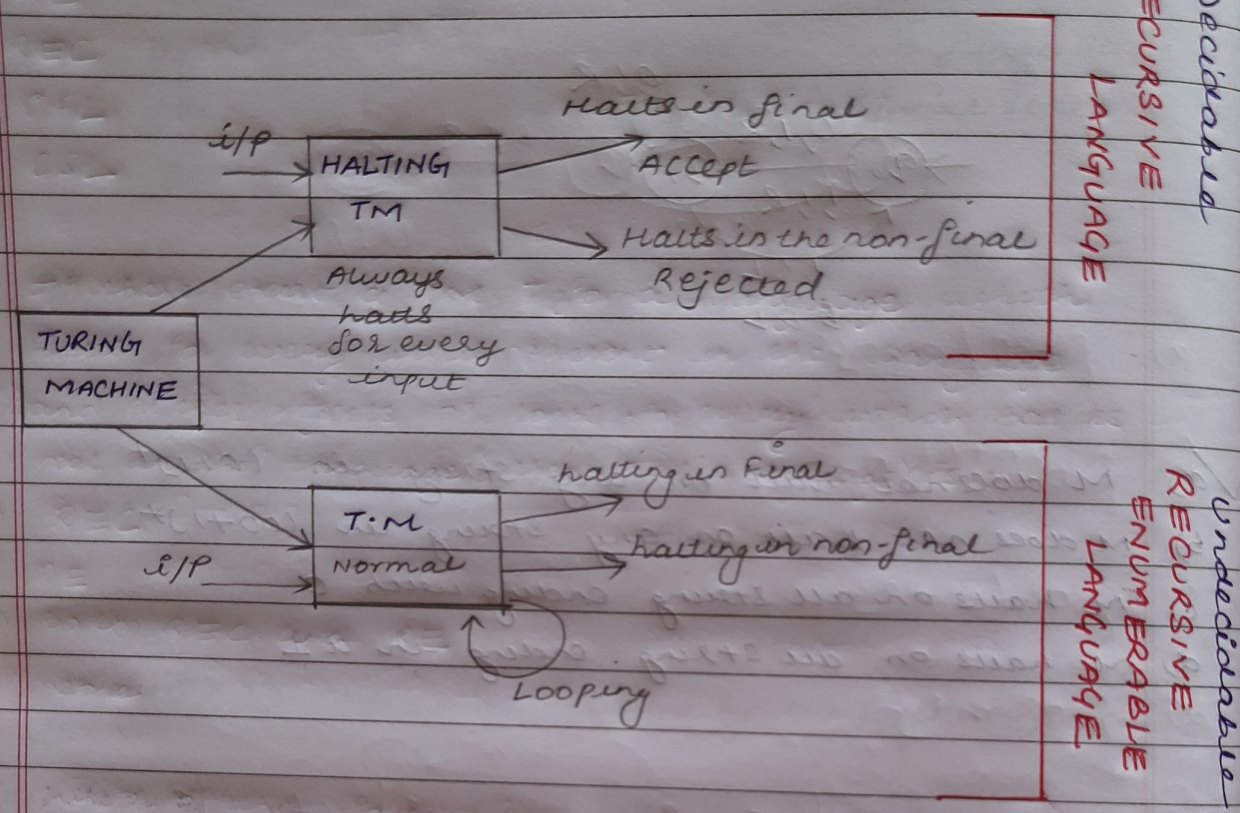
$q_0 \to q_1$

Not halting

Nature of this machine is that it halts only on ϵ.

ϵ means | B | B | B |

RECURSIVE & RECURSIVE ENUMERABLE



RECURSIVE LANGUAGE (REC)

A recursive language (subset of RE) can be decided by Turing machine which means it will enter into final state for the strings of language and rejecting state for the strings which are not part of the language.

Example -

$L = \{a^n b^n c^n \mid n \geq 1\}$ is Recursive we can construct a turing machine which will move to final state if the string is of the form $a^n b^n c^n$ else move to non-final state.

So the TM will always halt in this case, REC languages are also called as Turing decidable languages.

RECURSIVE ENUMERABLE (RE) or TYPE-0 Languages

Recursive Enumerable languages or type-0 languages are generated by type-0 grammars.

A RE language can be accepted or recognised by Turing machine which means it will enter into final state for the strings of language and may or maynot enter into rejecting state for the strings which are not part of the language.

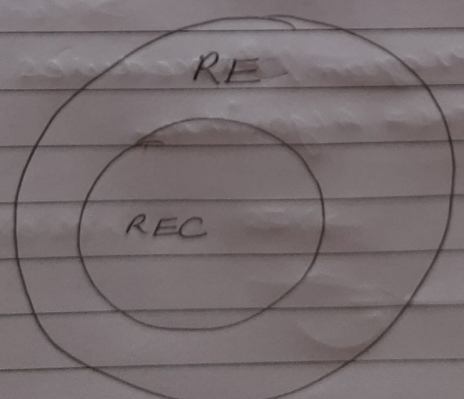
It means TM can loop forever for the strings which are not part of the language. RE languages are also called Turing recognizable languages.

For the strings ∈ Language halts in final state

For the strings ∉ Language

May halt in non-final state

Loop forever



Every REC language is RE but every RE need not be REC

Q. The language accepted by halting turing machine is:

a) RE

✓b) REC

c) CFL

d) CSL

Q. If there exist a TM T for a language which halts for every word present in the language and either rejects or loops for every word that is not in the language, then that language is:

a) REC

✓b) RE

c) Both REC and RE

d) None

** Q. Which of the following statements is true regarding finite length C programs & Turing machines

✓a) Each one can simulate the other

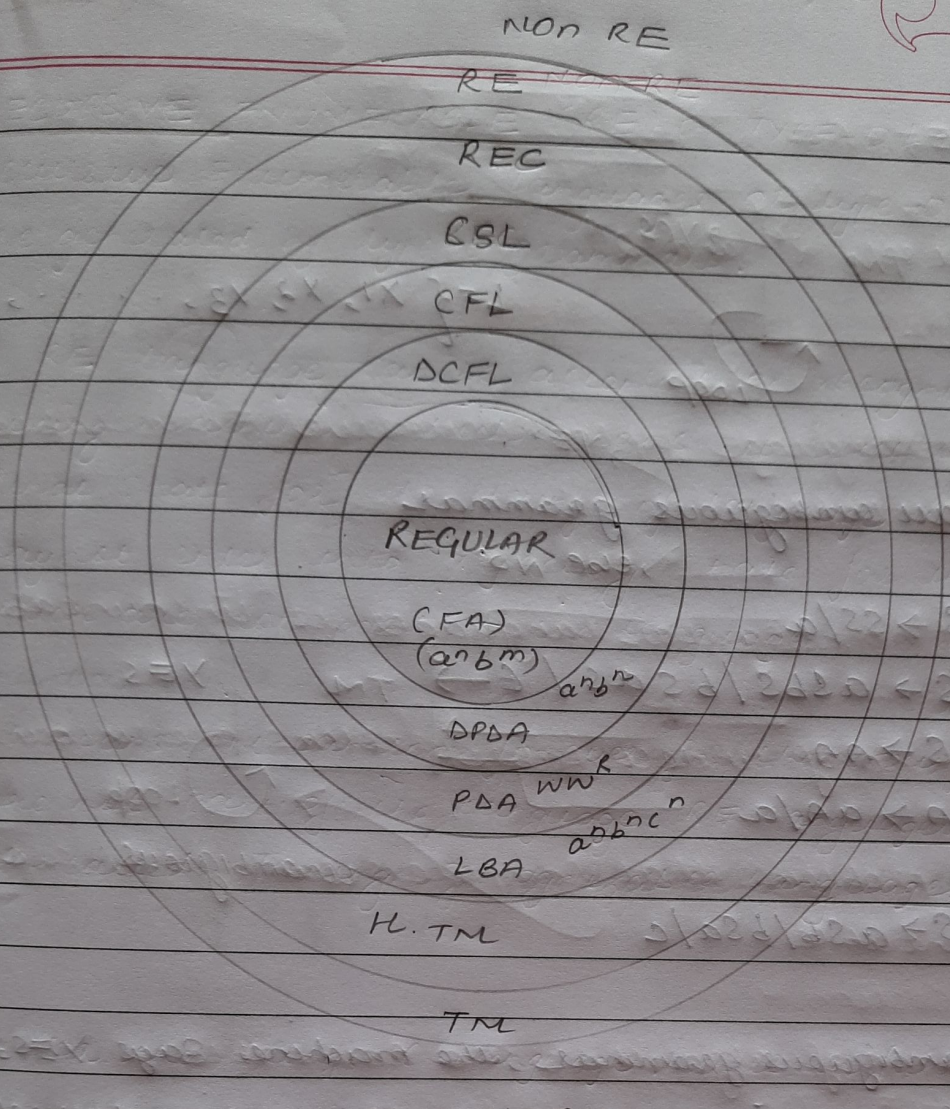
b) The turing machines always halts which represents all C programs.

c) The C programs that always halt can simulate all turing machines

d) All of the above.

TM = Program / Procedures

Halting TM = Algorithm



CHOMSKY Heirarchy

Examples-

1) REC -

$L = \{x_1, x_2, x_3 \ldots\} = TM \rightarrow$ YES / NO

Empty language.

X_1 a) $S \rightarrow SS / aS$

X_2 b) $S \rightarrow AB$

X_3 c) $S \rightarrow aSb / Sb$

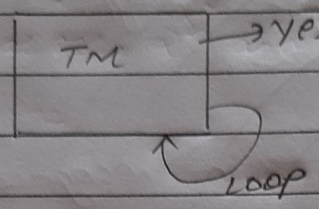
X_4 d) $S \rightarrow aSb / a$

$\left.\begin{array}{l} L \\ \{ \\ \end{array}\right\} \rightarrow$ TM $\rightarrow$ YES / NO } Recursive

Set of all Empty grammars/is Recursive

Finite grammars.

2) RE -



$L = \{ x_1, x_2, x_3, \ldots, x_n \}$

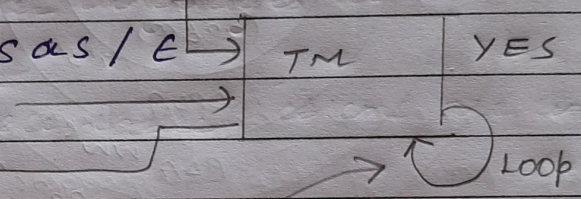
Set of all ambiguous grammar → YES
→ else NO

X_1 1) $S \to SS / a$

X_2 2) $S \to aSbS / bSaS / \epsilon$

X_3 3) $S \to AA$

$A \to aA / a$



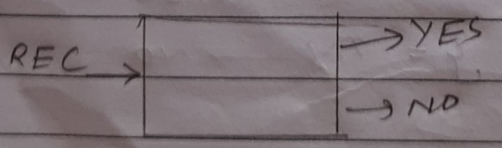
unambiguous

X_4 4) $S \to aSb / bSa / c$

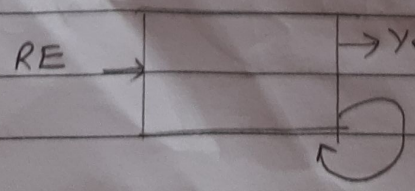
For ambiguous grammar, the machine says YES, but for unambiguous grammar, the machine does not answer anything.

undecidable

⇒ set of all ambiguous grammar



Decidable



undecidable.

thee2aNgauceu3u

Set

- Finite Set
  - $\{ab, ba\}$
- Infinite Set
  - TM Possible → Countably Infinite
  - NO TM Possible → Uncountable
    - set of real numbers

$$\Sigma^* = \{\epsilon, a, b \ldots\}$$

* Countable set -

$N = \{1, 2, 3, 4 \ldots\}$ = Countable

One to one correspondance with natural set

$\{S\} \leq N$

* Uncountable set -

set size more than natural set.

E.g → set of all real numbers.

NOTE:-

$\Sigma^* \rightarrow$ Countable

Suppose:-

$S = \{a, b\}$

Powerset $= \{\{\phi\}, \{a\}, \{b\}, \{ab\}\}$

Powaset > S

uncountable

$\Sigma^* = N \rightarrow$ Countable

2^{Σ^*} = Uncountable

3) Non RE -

Real set, 2^{Σ^*} and 2^N

Every language upto Recursive Enumerable is Countable.

* Set of non-Regular languages is uncountable

* Total no. of undecidable problems are uncountable

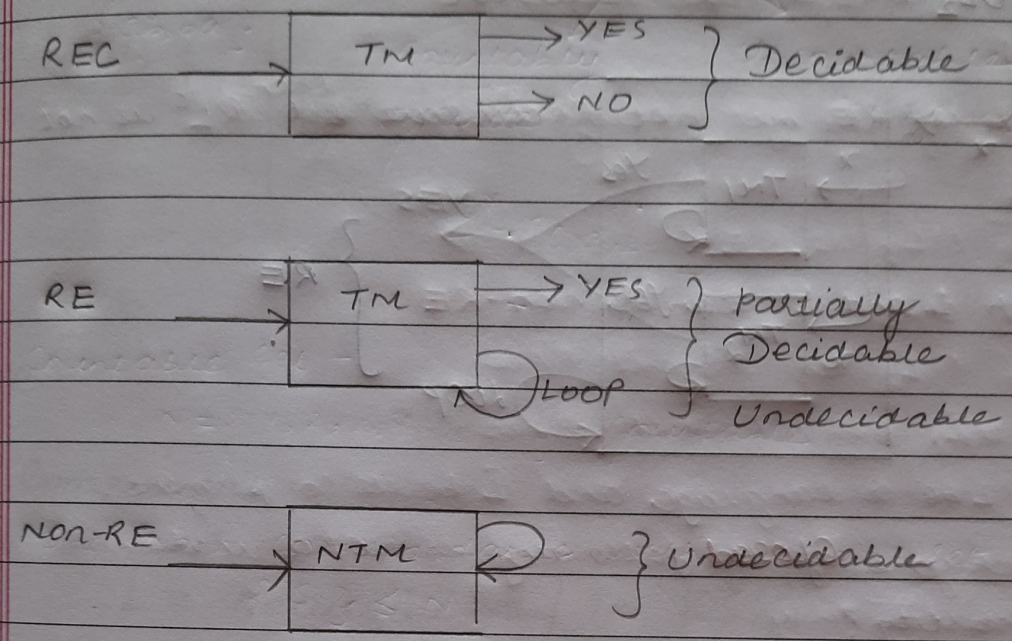
* Set of problems that Computers can't solve are uncountable.

* Total no. of algorithms are Countable.

L - 54

19-07-21

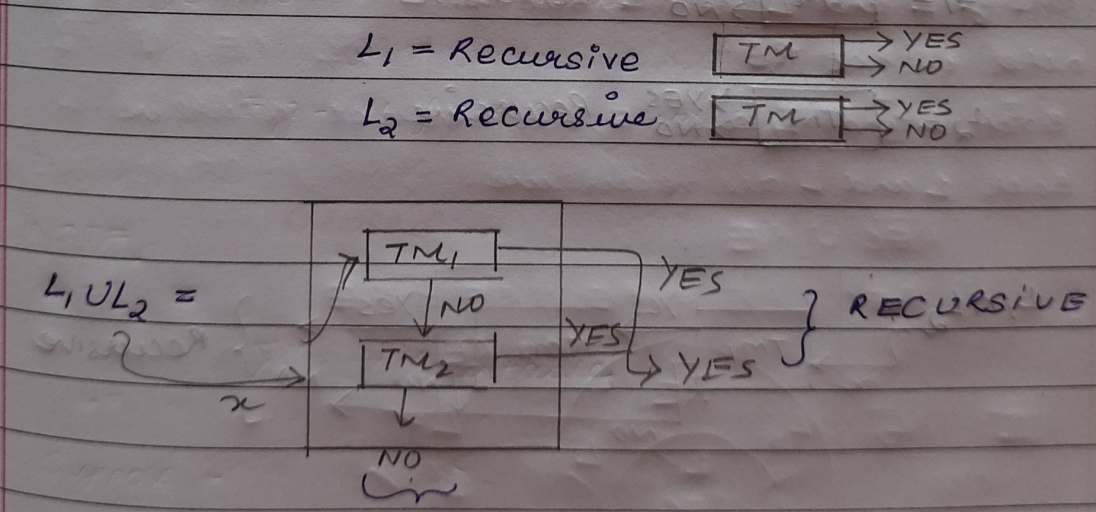
Closure Properties of Recursive & Recursive Enumerables language.



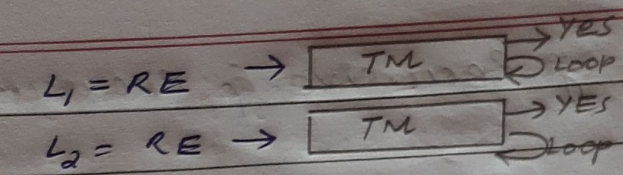
Union -

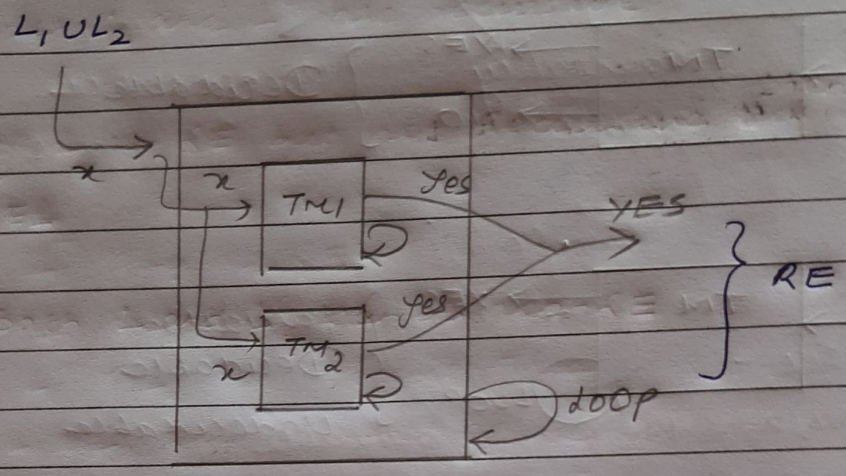
L_1 = Recursive

L_2 = Recursive



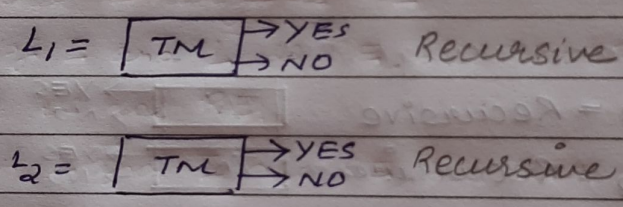
Closed

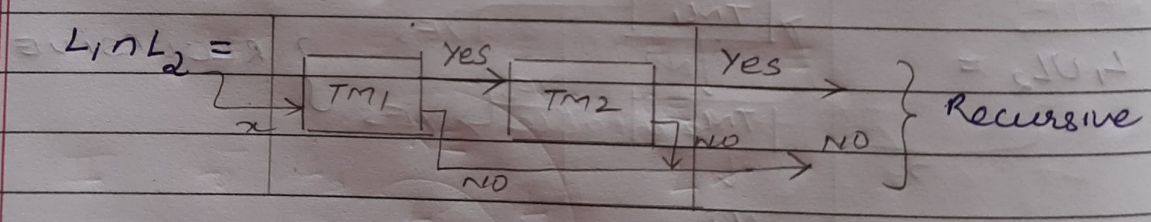




Closed

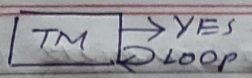
Intersection -

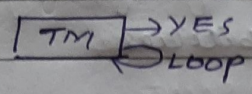


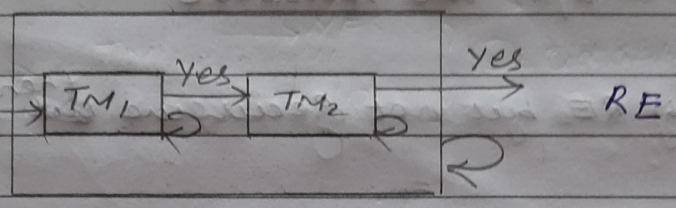


Closed

$\underbrace{RE = REL}$

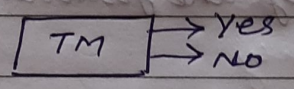
$L_1 = REL$ 

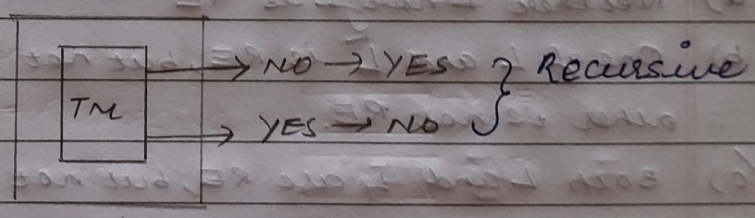
$L_2 = REL$ 

$L_1 \cap L_2 =$ 

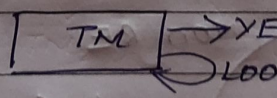
Closed

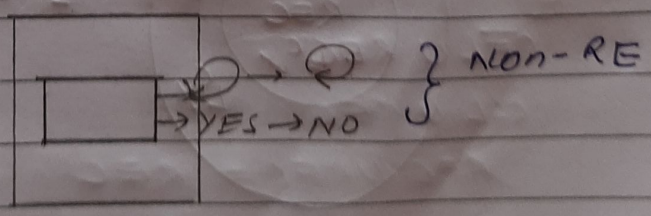
Complement -

$L =$ Recursive $=$ 

$\Sigma^* - L =$ 

Closed

$L =$ RE $=$ 

$\Sigma^* - L =$ 

Not-closed

Complement of RE is non-RE

* For a given language L and its complement L^c; the following are possibilities.

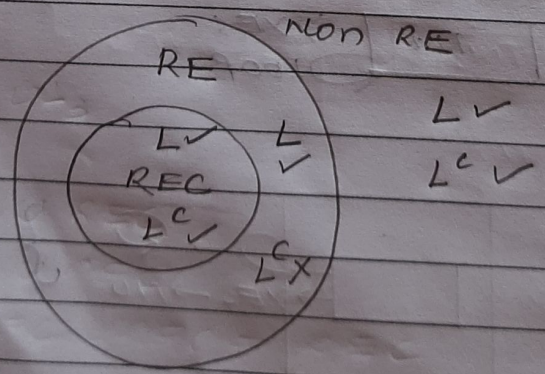
1) Both L and L^c are recursive

2) If L is RE but not Recursive then L^c is not RE

3) Both L and L^c can be non-RE

* Q. Let L be a language and $\bar{L}$ be its complement, which one of the following is NOT a viable possibility?

a) Neither L nor $\bar{L}$ is RE
b) One of L and $\bar{L}$ is RE, but not recursive; the other is not RE.
✓ c) Both L and $\bar{L}$ are RE, but not not recursive
d) Both L and $\bar{L}$ are recursive



Q. L_1 = Recursive Language
L_2 and L_3 = RE but not REC

which of the following is not necessarily true ?

a) $L_2 - L_1$ is RE
b) $L_1 - L_3$ is RE
c) $L_2 \cap L_3$ is RE
d) $L_2 \cup L_3$ is RE

a). $L_2 - L_1 \Rightarrow L_2 \cap \overline{L_1}$

RE ∩ REC ⇒ RE
RE ∩ RE

b) $L_1 - L_3 \Rightarrow L_1 \cap \overline{L_3} \Rightarrow$ REC ∩ NON RE ⇒ RE NON RE

c) $L_2 \cap L_3 \Rightarrow L_2 \cap L_3 \Rightarrow$ RE ∩ RE ⇒ RE

d) $L_2 \cup L_3 \Rightarrow$ RE

$CFL^c = RECURSIVE$

Q. Consider the following types of languages L_1 : Regular
L_2 : Context Free
L_3 : Recursive
L_4 : RE

Which of the following is/are True?

I $\overline{L_3} \cup L_4$ is RE ✓
II $\overline{L_2} \cup L_3$ is REC ✓
III $L_1^* \cap L_2$ is Context Free
IV $L_1 \cup \overline{L_2}$ is Context Free

a) I only
b) I and III only
c) I and IV only
✓ d) I, II & III only

I) $\overline{L_3} \cup L_4$

$\overline{REC} \cup RE$

$REC \cup RE = RE \cup RE = RE$

II $\overline{L_2} \cup L_3$

$\overline{CFL} \cup REC$

$CSL \cup REC$

$REC \cup REC = REC$

III $REG^* \cap CFL$

$REG \cap CFL$

CFL

IV $REG \cup \overline{CFL}$

$REG \cup CSL$

$REG \cup REC = REC$

Undecidability

Recursive language = Decidable
Recursive Enumerable language = Undecidable (partially decidable)
Non-RE = Undecidable

→ Decidable problem -
The problem (or) language for which halting Turing machine exists is known as decidable problem.
(or)
The problem for which algorithm exists is known as decidable problem

→ Undecidable problem -
The problem (or) language for which no halting Turing machine exists is known as undecidable problem
(or)
The problem for which no algorithm exists is known as undecidable problem.

Reduction

Problem A is reducible to Problem B means B is atleast as hard as problem A.

$$A \leq B$$

E.g

$$A \leq B \leq C \leq D$$

If UD → UD → UD → UD

Example:-

Small Big

* $A \leq B$

$UD \rightarrow UD$

* $A \leq B$

$? \leftarrow UD$

$A \leq B$

$UD \not\leftarrow UD$

Maths $\leq$ Engg.

Don't know $\rightarrow$ Don't know

May or may not understand maths $\leftarrow$ Don't know

* $A \leq B$

$D \leftarrow D$

* $A \leq B$

$D \not\rightarrow D$

Only this is possible

$\therefore \left\{ \begin{array}{l} UD \rightarrow UD \\ D \leftarrow D \end{array} \right\}$

NOTE:-

* $A \leq B$

$UD \rightarrow UD$

$D \leftarrow D$ ✓

$REC \leftarrow REC$ ✓

$RE \leftarrow RE$ ✓

$NonRE \rightarrow NonRE$

$NOT\ REC \rightarrow NOT\ REC$

Q. Let $A \leq_m B$ denotes that language A is mapping reducible (also known as many-to-one reducible) to language B. Which of the following is FALSE?

(a) If $A \leq_m B$ and B is recursive then A is recursive.
(b) If $A \leq_m B$ and A is undecidable then B is undecidable.
(c) If $A \leq_m B$ and B is RE then A is RE.
✓(d) If $A \leq_m B$ and B is not RE then A is Not RE.

NOTE:-
Do not confuse Undecidable with RE.
Undecidable = Non-RE

Q. Consider three decision problems P_1, P_2, P_3. It is known that P_1 is decidable and P_2 is undecidable. Which of the following is TRUE?

a) P_3 is decidable if P_1 is reducible to P_3.
b) P_3 is undecidable if P_3 is reducible to P_2
✓c) P_3 is undecidable if P_2 is reducible to P_3.
d) P_3 is decidable if P_3 is reducible to P_2's complement.

a) $P_1 \leq P_3$
Decidable → X

$P_3 \leq P_2$
X ← undecidable

$P_2 \leq P_3$
undecidable → Undecidable

$P_3 \leq \overline{P_2}$

20-07-21

Decision Properties of formal languages

1. Emptyness Problem -

a) Regular Languages → Decidable
b) CFL = CFG → Decidable

2. Intersection Empty Problem -

Is $L_1 \cap L_2 = \phi$?
UD ⟷ UD For RE

For REC = UD
CFL = Undecidable (CFL is not closed under ∩)
Intersection
To satisfy this, {∩, ϕ} Both should satisfy
Regular = Decidable

3. Completeness Problem -
Checking whether given language is complete or not? (Complement it & check for Emptiness.)

Regular : Decidable (Complement is closed and Emptiness is decidable)
CFL : Undecidable
DCFL : Decidable
RE : Undecidable

4. Subset Problem -
is $L_1 \subseteq L_2$?
L_1's subset of L_2 or not?

Regular ⇒ Decidable
DCFL ⇒ Undecidable
CFL ⇒ Undecidable
RE ⇒ Undecidable

$L_1 \cap L_2 = L_1$
Intersection & Equivalence

Important

** Decidability and Undecidability table
(D) (UD)

Problems	RL	DCFL	CFL	CSL	REC	RE
* Does 'w' belong to language L? (Membership Problem)	D	D	D	D	D	UD (Loop case)
* Is L = null? (Emptyness Prob.)	D	D	D	UD	UD	UD
* Is L = Σ* ? (Completeness Problem)	D	D	UD	UD	UD	UD
* Is L1 = L2 ? (Equality Prob.)	D	D	UD	UD	UD	UD
Is L1 subset of L2? (Subset Problem)	D	UD	UD	UD	UD	UD
Is L1 intersection L2 = null?	D	UD	UD	UD	UD	UD
* Is 'L' finite or not? (Finiteness prob)	D	D	D	UD	UD	UD
Is complement of 'L' a language of same type or not?	D	D	UD	D	D	UD

closed or not?

Problems	RL	DCFL	CFL	CSL	REC	RE
Is intersection of two languages of same type or not?	D	UD	UD	D	D	D
Is 'L' regular language or not? (L' is any language)	D	D	UD	UD	UD	UD

Halting Problem of Turing machine :

By reading string x Turing machine Halts or not ?

This problem is undecidable for RE.

This problem is decidable for REC.

Q. Which of the following problems are undecidable ?

a) Membership problem in context free languages.

✓ b) Whether a given context-free language is regular

c) Whether a finite state automation halts on all inputs.

✓ d) Membership problem for type 0 languages.

Q. Which of the following statements is false ?

a) The halting problem for Turing machine is undecidable.

b) Determining whether a CFG is ambiguous in undecidable.

c) Given two arbitrary CFGs G_1 & G_2, it is undecidable whether $L(G_1) = L(G_2)$

✓ d) Given two Regular grammars G_1 and G_2 it is undecidable whether $L(G_1) = L(G_2)$

GATE 2 MARKS

Q. Consider the following problems $L(G)$ denotes the language generated by a grammar G. $L(M)$ denotes the language accepted by a machine M.

I For an unrestricted grammar G and a string w, whether $w \in L(G)$.

II Given a Turing Machine M, whether $L(M)$ is regular.

III Given two grammars G_1 and G_2, whether $L(G_1) = L(G_2)$

IV Given an NFA N, whether there is a deterministic PDA P such that N and P accept the same language.

Which one of the following statements is correct?

a) Only I and II are undecidable

b) Only III is undecidable

c) Only II and IV are undecidable

✓ d) Only I, II and III are undecidable

Unrestricted grammar ⇒ RE

If not mentioned; default grammars are RE

Default TM = RE

Regular Exp. = R.G = FA = RL

CFL = CFG = PDA

Q. Checking whether a Turing machine having 100 states or not?

[STATE COUNTING PROBLEMS]

Decidable

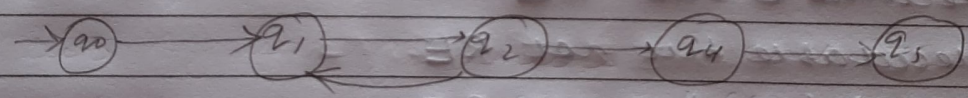
Q. Checking whether ~~by~~ reading string x Turing Machine halts in a particular state or not?

[STATE ENTRY PROBLEM]

Undecidable

(There might be some loop)

Q. By ~~st~~ reading string x whether TM halts in particular state Q (or) not after 100 steps?



Decidable because number of steps are fixed

Q. Whether Turing machine accepts given string x or not?

Undecidable (Could fall into loop)

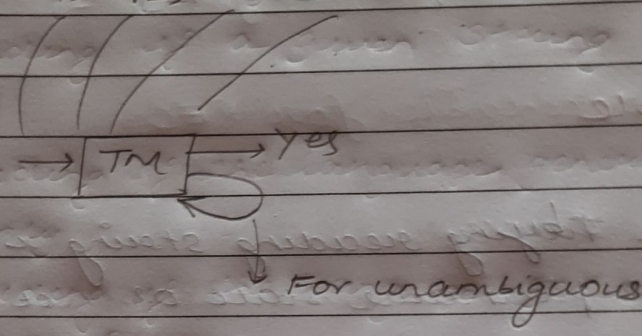
Q. Whether TM accepts given string x or not within 100 steps?

Decidable because finite no. of steps.

Infinite grammars

Q. $L = \{ \langle G \rangle \mid G \text{ is ambiguous grammar} \}$

$\langle G_1, G_2, G_3 \ldots \rangle$



Language = RE

GATE 2014 Q. Let $\langle M \rangle$ be the encoding of a Turing machine as a string over $\Sigma = \{0, 1\}$. Let $L = \{ \langle M \rangle \mid M \text{ is a Turing machine that accepts a string of length } 2014 \}$. Then, L is

a) decidable and RE

✓ b) undecidable but RE

c) Undecidable but not RE

d) decidable but not RE

decidable $\Rightarrow$ REC $\Rightarrow$ RE

$\langle M1, M2, M3, M4, \ldots \rangle$

For string of length 2014, it accepts but for more than 2014 ?

Looping is a possibility

$\therefore$ Undecidable but RE

Q. Consider the following languages :

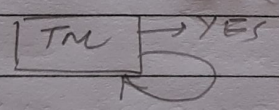
$L_1 = \{ \langle M \rangle \mid M$ takes at least 2016 steps on some input $\}$

$L_2 = \{ \langle M \rangle \mid M$ takes at least 2016 steps on all inputs $\}$ and

$L_3 = \{ \langle M \rangle \mid M$ accepts $\epsilon \}$

where for each turing machine M, $\langle M \rangle$ denotes specific encoding for M. Which of the the following is TRUE ?

a) L_1 is recursive and L_2, L_3 are not recursive
b) L_2 is recursive and L_1, L_3 are not recursive
✓ c) L_1, L_2 are recursive and L_3 is not recursive
d) L_1, L_2, L_3 are recursive

$L_3 =$ 

Q. Which of the following problems is undecidable?

✓ a) Deciding if a given CFG is ambiguous
b) Deciding if a given string is generated by a CFG
c) Deciding if the language generated by a given CFG is empty.
d) Deciding if the language generated by a given CFG is finite.

Membership Problem
Emptiness Problem
Finiteness Problem
} Decidable for CFGs

NOTE :-

Three Problems ⇒ P1, P2, P3
P1 ⇒ Decidable
P2 ⇒ Undecidable

P3 is undecidable if P2 is reducible to P3

$P_2 \leq P_3$
$UD \rightarrow UD$

ॐ