UNIT: 3

Dashrath Nandan

Date :

★ Context-Sensitive Grammer

A CSG is an unrestricted grammer in which all productions are of form-

$\alpha \longrightarrow \beta$, where $\alpha, \beta \in (V \cup T)^+$ [epsilon is Not allowed]

$|\alpha| \leq |\beta|$

- CSG are more powerful than CFG, because there are some language that can be described by CSG but not by CFG. and CSG is less powerful than Unrestricted Grammer.

Eg. $L = \{a^n b^n c^n \mid n \geq 1\}$ can generate CSL grammer.

* Context-Sensitive grammer has 4 tuples. $G = \{N, \Sigma, P, S\}$

N = set of non-terminals symbols

Σ = set of terminal symbols.

S = start symbol of production.

P = Finite set of productions.

All rules in P are of form $\alpha_1 A \alpha_2 \longrightarrow \alpha_1 \beta \alpha_2$

* Context-sensitive language: The language that can be defined by content-sensitive grammer is called CSL.

Properties of CSL are-

Union, Intersection and Concat. of two CSL is Context Sensitive.

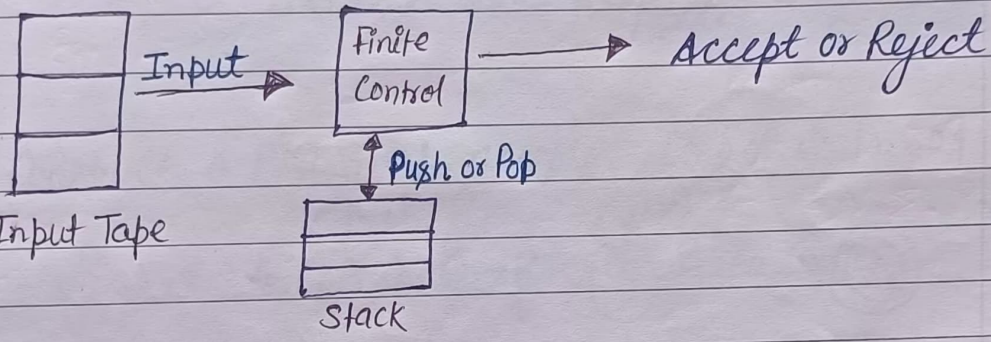
Complement of a context-SL is context-sensitive.

Date : _____

★ Pushdown Automata (PDA)

- PDA is a way to implement a CFG in the same way we design DFA for regular grammer.

- PDA is simply an NFA with an "external Stack Memory":
  - → PDA is more powerful than FSM.
  - → PDA can remember an infinite amount of info.
  - → PDA = FSM + A Stack.



Components of PDA:

i) **Input Tape**: Input Tape is divided in many cells or Symbol. The input head is read only, and move only from left to Right one symbol at a time.

ii) **Finite Control**: It has pointer which points the current symbol which is to be read.

iii) **Stack**: In PDA, it is used to store item temporarily.

★ **Formal Defination**: A PDA is defined by 7 tuples –

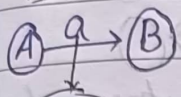
$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

Q = A finite set of states	q_0 = Start state.
Σ = Set of Input Symbols.	Z_0 = Start Stack Symbol
Γ = Finite Stack alphabet.	F = The set of Final state/ Accepting State.
δ = Transition Function.	

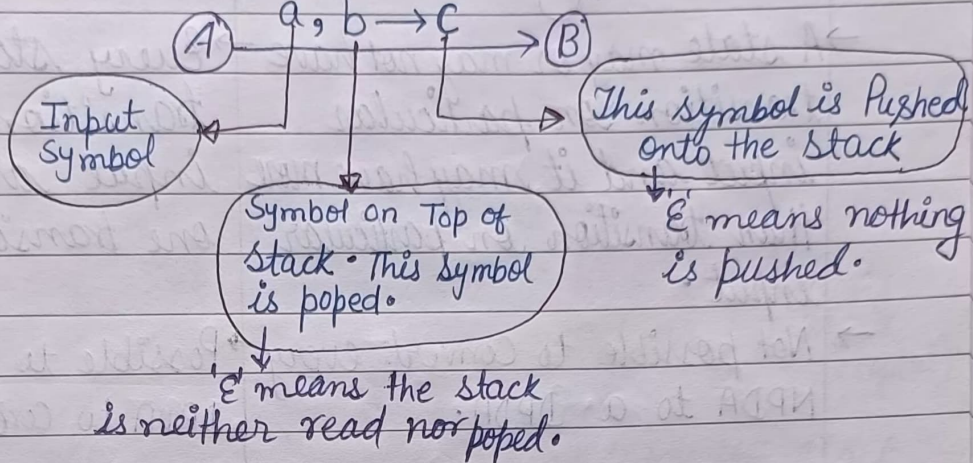
Date :

★ PDA Graphical Notations :

FSM

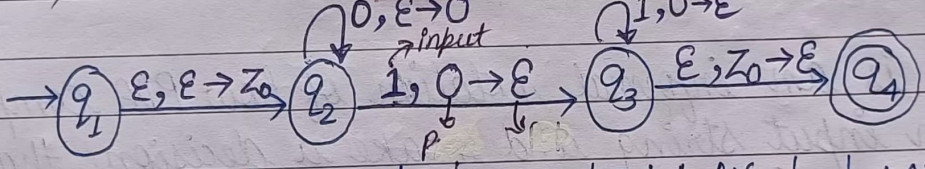


PDA

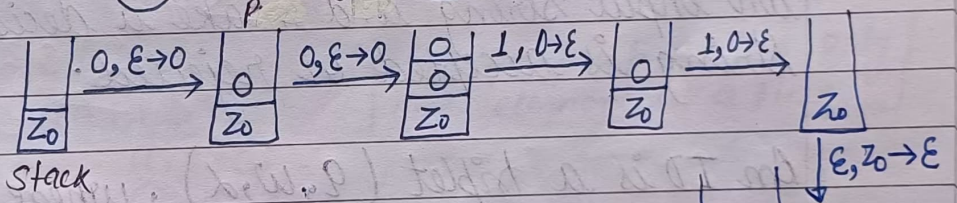


Ex: Construct a PDA that accepts $L = \{0^n 1^n \mid n \geq 0\}$

Sol:



Eg. 0011 ⇒ (Dry Run)

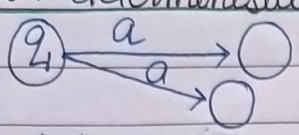
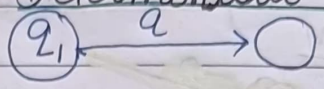


* PDA Acceptance

A language can be accepted by PDA using two approaches -

i) When we reach the final state.
ii) When we find stack is empty.

Date :

Non-deterministic PDA	Deterministic PDA
	
→ A state may or may not have transition on particular input and it may have more than transition on particular input.	→ Every state must have transition on every input and it also have one transition on each input.
→ Not possible to Convert every NPDA to a DPDA.	→ Possible to convert every DPDA to Corresp. NPDA.

★ Instantaneous Description (ID)

ID is an informal notation of how PDA computes an input string and make a decision that string is accepted or rejected.

An ID is a triplet (q, w, α), where -

q: describe the current state.
w: describe the remaining input.
α: describe the stack contents, top at the left.

★ Turnstile Notation :

- $\vdash$ sign describe turnstile notation and represent one move.
- $\vdash^*$ sign describe a sequence of moves.

Eg: $(p, b, T) \vdash (q, w, \alpha)$

While taking a transition from state p to q the input symbol 'b' is consumed, and top of the stack 'T' is represented by a new string α.

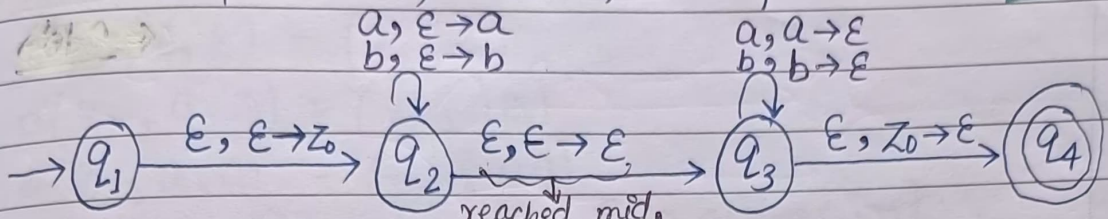
Date :

Nego Acad. L-89 Working

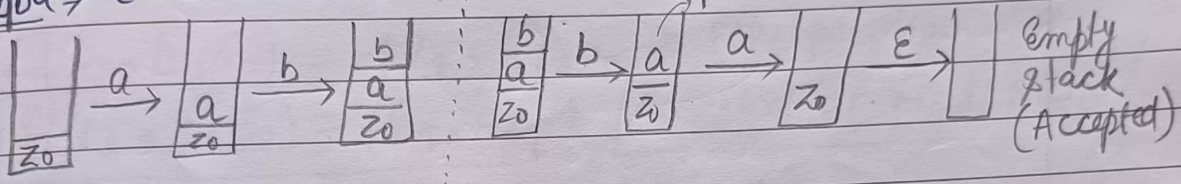
Ex: Construct a PDA that accepts $L = \{WW^R \mid W(a+b)^*\}$

$L = \{aabbaa, abba, \ldots\}$

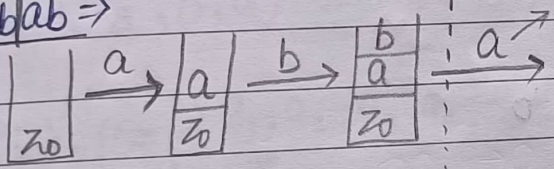
Sol:



abba ⇒ ε a ε b ε b ε a ε



abab ⇒

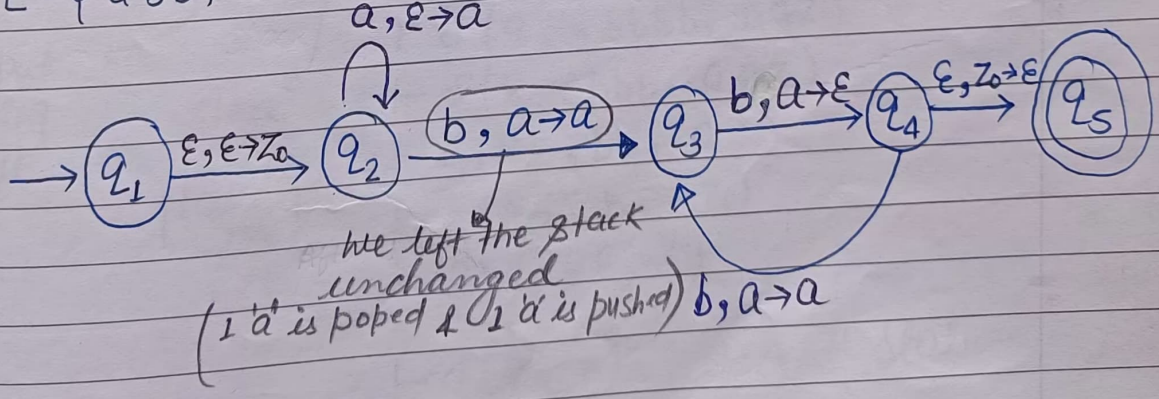


∴ Rejected

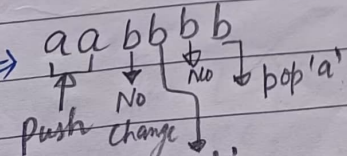
Ex: Construct a PDA for $L = \{a^n b^{2n} \mid n \geq 1\}$

[We can push or pop only one element at a time]

Sol:- $L = \{abb, aabbbb, \ldots\}$



dry run ⇒ aa bb bb



Date :

Transition function"

ID:

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

$\delta(q_2, a, Z_0) = (q_2, aZ_0)$

$\delta(q_2, a, a) = (q_2, aa)$

$\delta(q_2, b, a) = (q_3, a)$ [No operation]

$\delta(q_3, b, a) = (q_4, \varepsilon)$ → POP operation

$\delta(q_4, \varepsilon, Z_0) = (q_5, \varepsilon)$ → Accepted or Final state

$\delta(q_4, b, a) = (q_3, a)$ [No operation]

PDA, $P = (\{q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \{a, Z_0\}, \delta, q_0, Z_0, \{q_5\})$

Q Σ Γ δ q_0 Z_0 F

⇒ We will simulate PDA for aa bbbb

$\delta(q_1, aabbbb, \varepsilon) \vdash \delta(q_2, aabbbb, Z_0)$

$\vdash \delta(q_2, abbbb, aZ_0)$

$\vdash \delta(q_2, bbbb, aaZ_0)$

$\vdash \delta(q_3, bbb, aaZ_0)$

$\vdash \delta(q_4, bb, aZ_0)$

$\vdash \delta(q_3, b, aZ_0)$

$\vdash \delta(q_4, \varepsilon, Z_0)$

$\vdash \delta(q_5, \varepsilon)$ Accept state

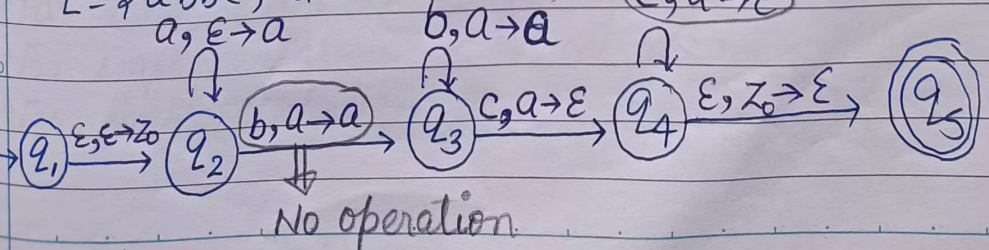
input string left

Stack Top (left is Top)

Ex: Construct PDA for $L = \{a^n b^m c^n \mid n \geq 1, m \geq 1\}$

$L = \{abbc, aabccc, \ldots\}$

Sol:-



Date : _____

how automata work for aabcc.

State	Input	δ used	Stack (Leftmost = Top)	State after move
q_1	ε	$\delta(q_1, \varepsilon, \varepsilon) = (q_2, Z_0)$	Z_0	q_2
q_2	aabcc	$\delta(q_2, a, Z_0) = (q_2, aZ_0)$	aZ_0	q_2
q_2	aabcc	$\delta(q_2, a, a) = (q_2, aa)$	aaZ_0	q_2
q_2	aabcc	$\delta(q_2, b, a) = (q_3, a)$	aaZ_0	q_3
q_3	aabcc	$\delta(q_3, c, a) = \delta(q_4, \epsilon)$	aZ_0	q_4
q_4	aabcc	$\delta(q_4, c, a) = \delta(q_4, \epsilon)$	Z_0	q_4
q_4	ε	$\delta(q_4, \epsilon, Z_0) = \delta(q_5, \varepsilon)$	ϵ	q_5 → final state

No operation.

* Application of PDA

i) For designing the parsing phase of a compiler ( ).
ii) For implementation of stack applications.
iii) For solving the Tower of Hanoi Problem.
iv) For evaluating the arithmetic expressions.

* Increasing sequence of expressive power of machines :-

FA < DPDA < PDA < LBA < TM.

* Application of LBA :

- For implementation of genetic programming.
- For constructing syntatic parse trees for semantic analysis.

* Application of TM :

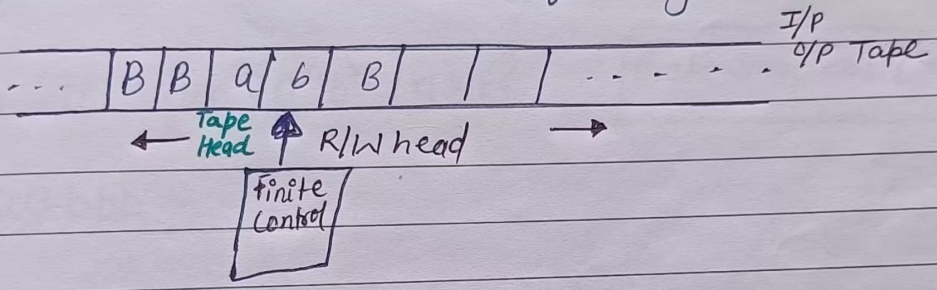
- For solving any recursively enumerable problem.
- For understanding complexity theory.
- For implementation of neural network.
- For implementation of AI.

Date :

Turing Machines (TM)

A Turing Machine is an accepting device which accepts the language (recursively enumerable set) generated by type 0 grammers.

- It was invented by Alan Turing in 1936.
- A TM consists of an infinite length tape and a R/W head, which can move left or Right.



* A TM can be formally described as 7 tuples –

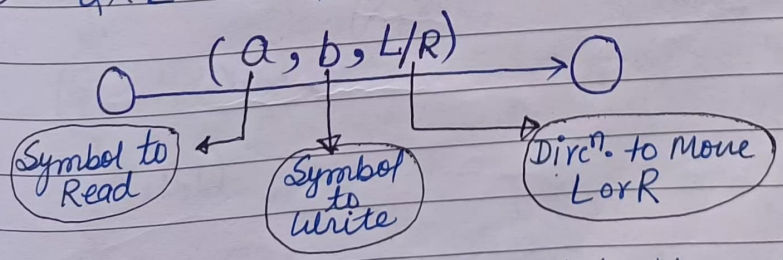
$(Q, \Sigma, \Gamma, \delta, q_0, B, F)$

Q: Set of states ; Σ: I/p alphabet

Γ: tape alphabet ($\Sigma \subset \Gamma$) ; q_0: Initial State

B: Blank Symbol ($B \in \Gamma$) ; F: Final State (Accept or Reject)

δ: $Q \times \Sigma \longrightarrow Q \times \Gamma \times (L/R)$



- If you don't want to update the cell, just write the same Symbol. Eg. (a, a, L/R)

* T.C. = $O(n \log n)$

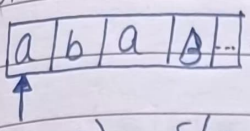
S.C. = $O(n)$

For 10 Marks: Show dry run of the Machine by an Example

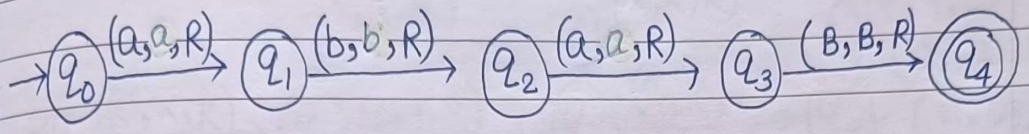
Date :

Ex: Construct a TM which accepts the $L = aba$ over $\Sigma = \{a, b\}$.

Sol:



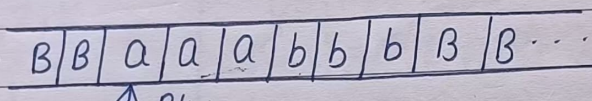
$\delta(q_0, a) = \delta(q_1, X, R)$, $\delta(q_1, b) = \delta(q_2, Y, R)$,
$\delta(q_2, a) = \delta(q_3, X, R)$, $\delta(q_3, B) = \delta(q_4, B, L/R)$.

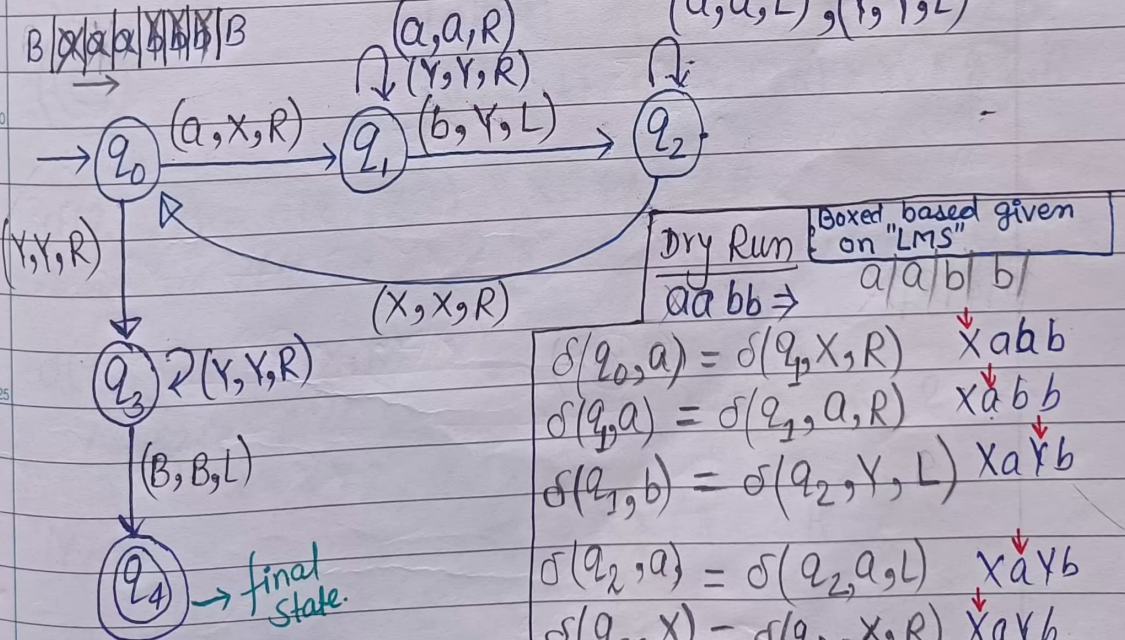


Ex: Design TM for $\{a^n b^n / n \geq 1\}$

Gate Smasher L-58

Sol: aaabbb





Dry Run (Boxed based given on "LMS")

aabb ⇒ a a b b

$\delta(q_0, a) = \delta(q_1, X, R)$ X a b b
$\delta(q_1, a) = \delta(q_1, a, R)$ X a b b
$\delta(q_1, b) = \delta(q_2, Y, L)$ X a Y b
$\delta(q_2, a) = \delta(q_2, a, L)$ X a Y b
$\delta(q_2, X) = \delta(q_0, X, R)$ X a Y b
$\delta(q_0, a) = \delta(q_1, X, R)$ X X Y b
⋮

Note: If TM is halt at any state other than q_4 ⇒ L is not accepted.

Date :

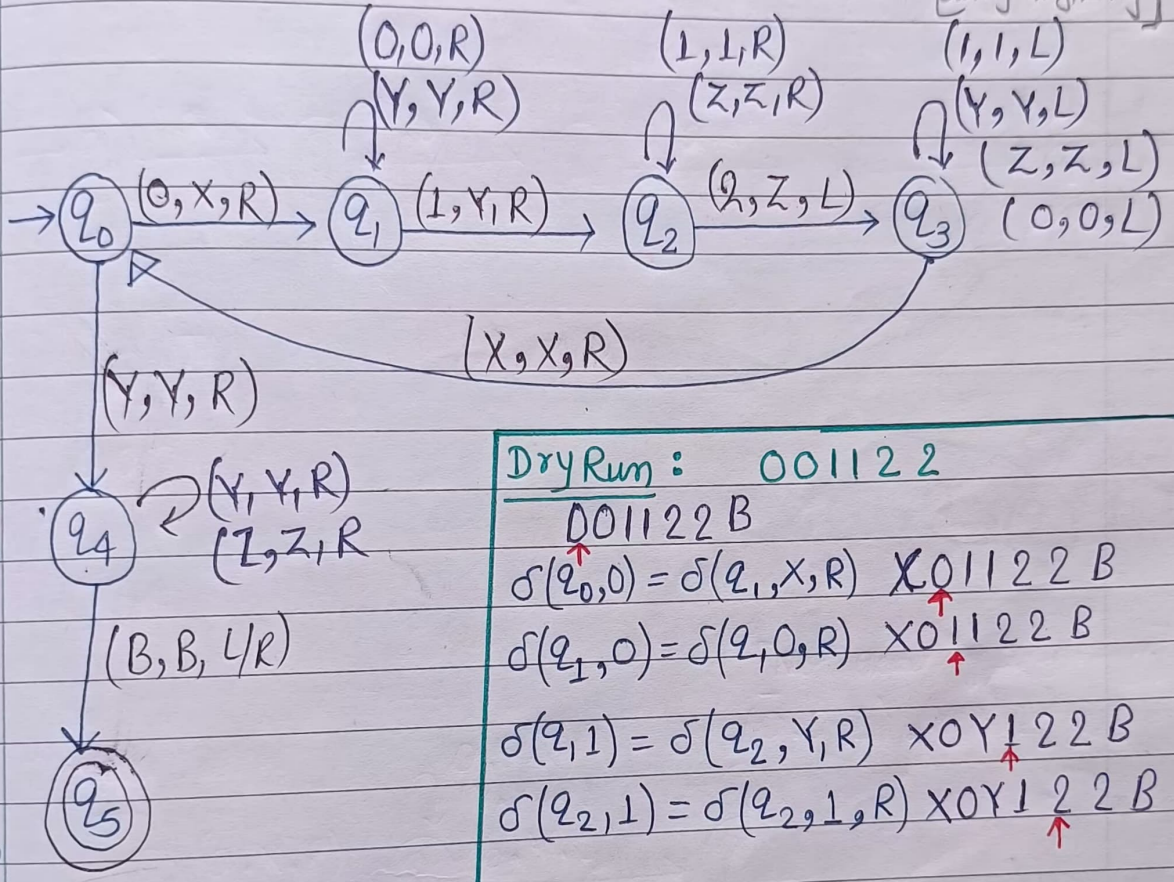
Ex:- Construct TM for language $L=\{0^n 1^n 2^m\}$

Gate Smasher
L - 59

Sol:

X	X	Y	Y	Z	Z	B
0	0	1	1	2	2	B

[Coming Back without doing anything]



Dry Run: 001122

001122B

$\delta(q_0,0) = \delta(q_1,X,R)$ X01122B

$\delta(q_1,0) = \delta(q_1,0,R)$ X01122B

$\delta(q_1,1) = \delta(q_2,Y,R)$ X0Y122B

$\delta(q_2,1) = \delta(q_2,1,R)$ X0Y122B

$\delta(q_2,2) = \delta(q_3,Z,L)$ X0Y1Z2B

come left till we find 'X' ignoring 1, Y and 0.

$\delta(q_3,X) = (q_0,X,R)$ X0Y1Z2B

Repeate till every $0 \to X$, $1 \to Y$ and $2 \to Z$.

XXYYZZB

$(q_3,X) = \delta(q_0,X,R) \Rightarrow$ XXYYZZB

$\delta(q_0,Y) = (q_4,Y,R)$

$\delta(q_4,Z) = (q_4,Z,R)$

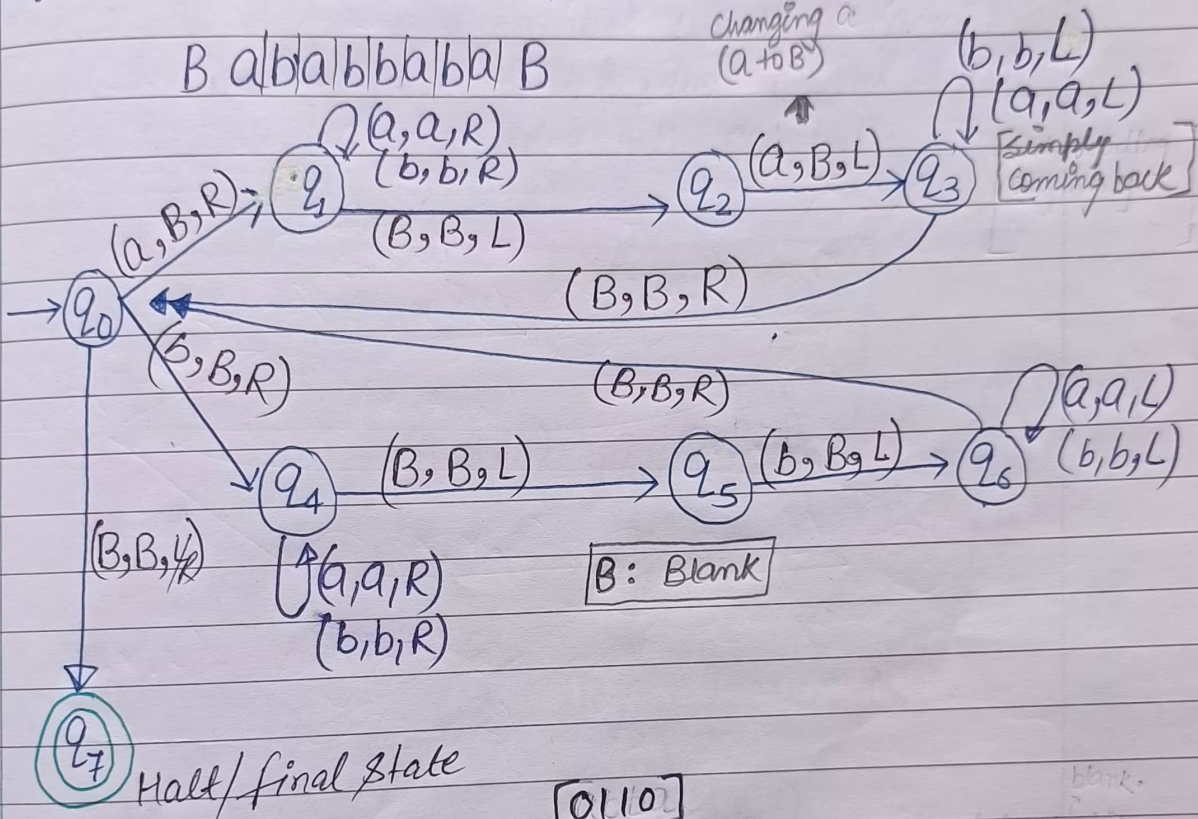
$\delta(q_4,B) = (q_5,B,L/R)$ = final state

Date :

Nesso Academy L-100

Eg Construct TM for Palindrom of even length.
abbabbaba

Sol:- Start reading the first symbol from left and compare it with the first symbol from right and if both are same, replace both with Blank.



[0110]

⇒ Read the first symbol, '1' move to (q_1) changing 1 to B(blank), we keep moving Right ignoring a/b until we find B(that means we reach the end of string) then turn left, $(q_1) \xrightarrow{(1,B,L)} (q_2)$, if the Element is 1, (i.e first and last same) replace it with 'B(blank)'. We keeps moving left till the start of string, when we read 'B' turn right and read next element.

⇒ Similarly for input B.

⇒ When we Read 'B' after turing Right [which mean we exhausted our string] and strings becomes "B B B B", then we reached the final state (q_7) and '1' is accepted.

Ex·3 of Ch: 3.3.1 Date :

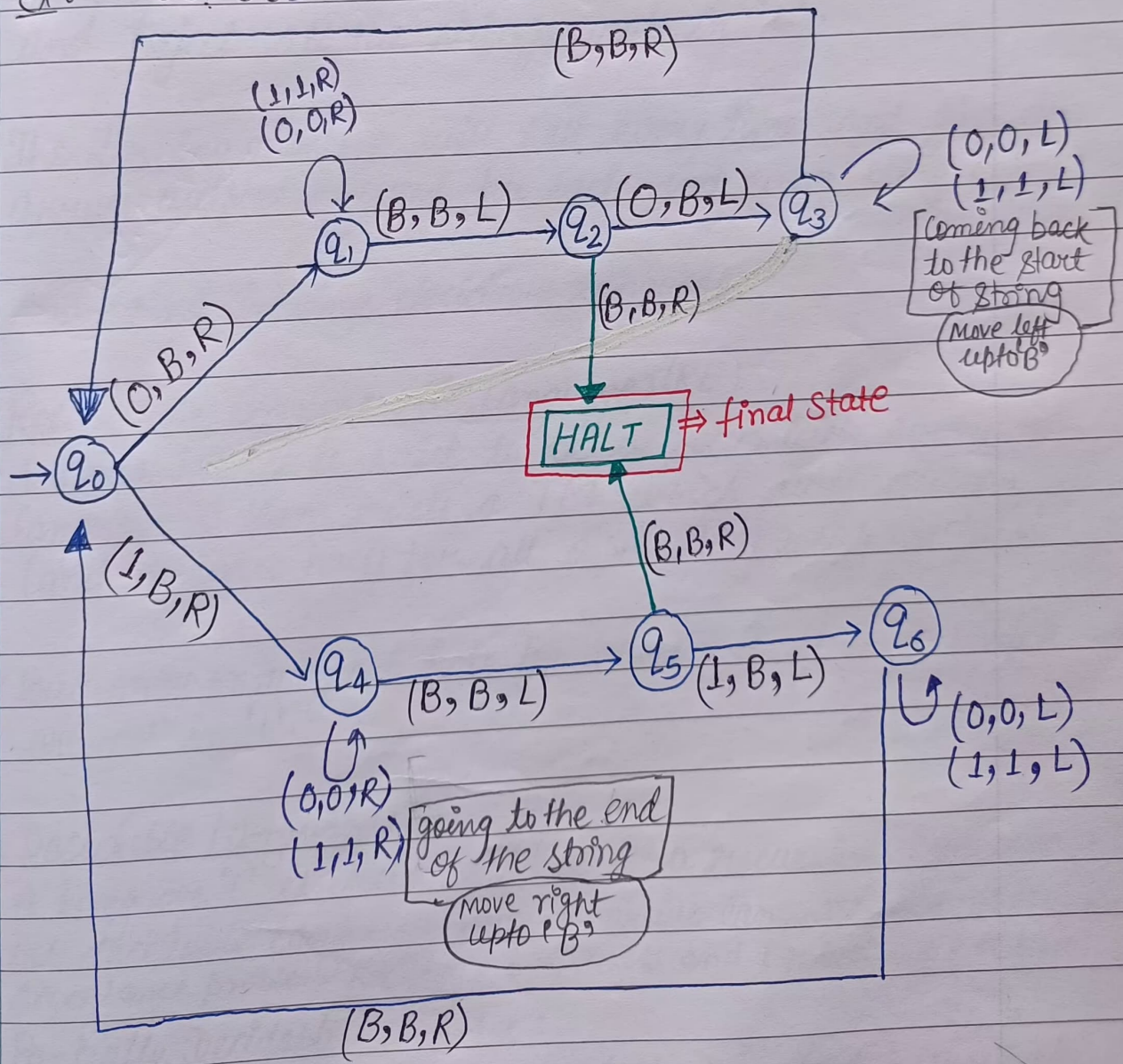
[Youtube or LMS]

Ques: Construct TM for checking palindrome of odd length.

[00100B]

Sol:- Firstly we read the first symbol from left and then we compare it with the first symbol from right to check whether it is same. Repeate this process until we found any symbol whole left and Right are 'B' (blank) Symbol B 0/1 B, that means it is an odd palindrome and string gets accepted.

Ex: 00100B or 11011B



Date :

★ Decidability and Undecidability

The class of problem which can be answered as 'Yes' are called solvable or decidable. Otherwise, the class of problems is said to be unsolvable or undecidable.

* Recursive Language : (REC)

- A language 'L' is said to be recursive if there exists a Turing Machine which will accept all the strings in 'L' and reject all the strings not in 'L'.

- The Turing machine will halt every time and give an answer accepted or rejected for each and every string input.

- Also called Turing decidable language.

* Recursively Enumerable Language : (RE)

- A language 'L' is said to be a recursively enumerable language if there exists a TM which will accept (and therefore halt) for all input string which are in 'L'.

- But may or maynot halt for all input strings which are not in 'L'.

* Decidable Language :

A Language 'L' is decidable if it is a recursive language.
All decidable languages are recursive language and vice-versa.
Ex: Acceptance problem for DFA, Emptiness and Equivalence problem for DFA.

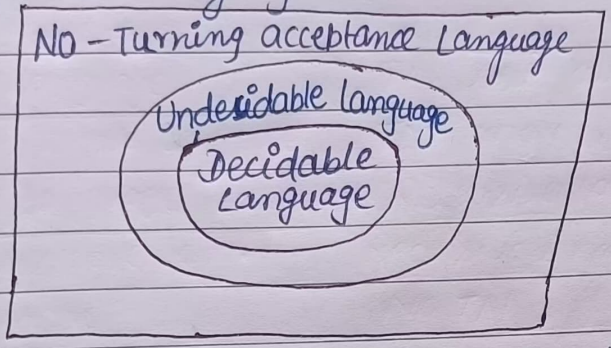
* Partially Decidable Language :

A language 'L' is partially decidable if 'L' is a recursively enumerable language.

Date :

* Undecidable Languages:

A language is undecidable if it is not decidable.
An undecidable language may sometimes be partially decidable but not decidable.
If a language is not even partially decidable, then there exists No TM for that language.



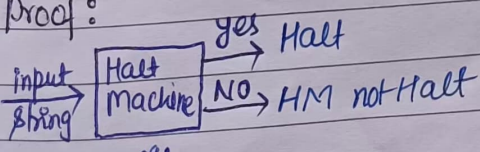
Ex: Halting problem of TM, The mortality problem, the mortal matrix problem, Post Correspondence problem, etc.

* TM Halting problem

Halting: It means that the program on certain input will accept it & halt or reject it & halt. It will never go into an infinite loop. Simply, halting means terminating.

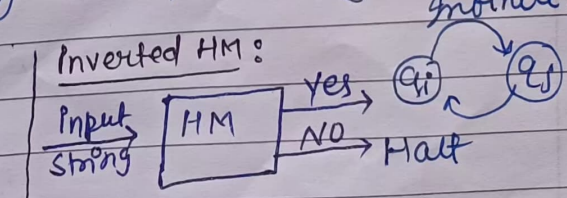
Input: A TM and an input string w.
Problem: Given a TM, will it halt when run on some particular given input string.

Proof:



H(P, I) → prog. → input
→ HALT
→ Not HALT

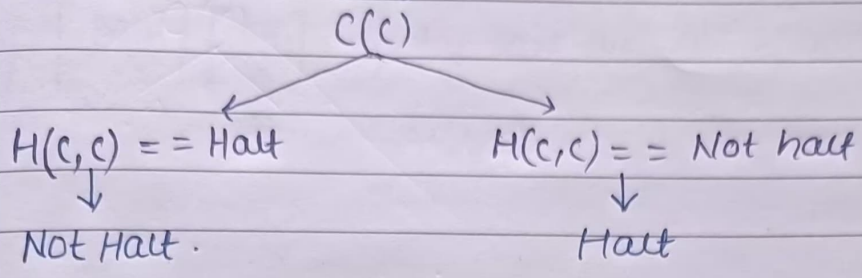
Inverted HM:




```
C(Y){
if(H(Y,Y)==Halt){ loop; }
else{
    return;}
```

Date :

if we run 'C' on itself.



∴ This is contradiction, Hence the halting problem is Undecidable.

★ Post Correspondence Problem:

↳ This problem is an undecidable decision problem.

↳ In this, we need to find a sequence of string such that the top & bottom strings are same.

↳ The PCP problem consists of two lists of strings of Equal length:- $A = w_1, w_2, w_3 \cdots w_n$, $B = x_1, x_2, x_3, \cdots x_n$

find a sequence such that $w_1 = x_1$ then we say that PCP has a solution.

i_1, i_2, i_3 such that $w_1 w_2 w_3 \cdots = x_1, x_2, x_3 \cdots$

Ex1: $M = (abb, aa, aaa)$ $N = (bba, aaa, aa)$

Sol:-

	X_1	X_2	X_3
M	abb	aa	aaa
N	bba	aaa	aa
	Y_1	Y_2	Y_3

$X_2 X_1 X_3 = aaabbaaa$

$\Rightarrow Y_2 Y_1 Y_3 = aaabbaaa$

We can see that $X_2 X_1 X_3 = Y_2 Y_1 Y_3$, Hence solution is $i=2, j=1, k=3$.

Ex2: $M = (ab, bab, bbaaa)$ and $N = (a, ba, bab)$

Sol:- $|X_2 X_1 X_3| \neq |Y_2 Y_1 Y_3|$ (length not same)

Also we see that

∴ PCP is undecidable.

Date :

[Neso Acad. L-112]

Ex: $M = (10, 011, 101)$, $N = (101, 11, 011)$

Sol:

D_1	D_2	D_3
10 / 101	011 / 11	101 / 011

⇒ Start match the numerator and denominator by making or reshuffling the order of domino (combination)

D_1 D_3 D_3

10 → 101 → 101 this will go on loop

101 011 011

∴ $D_1 D_3 D_3 \ldots\ldots \infty$ ⇒ This PCP is undecidable.

Ex: $M = (ab, bab, bbaaa)$, $N = (a, ba, bab)$

Sol:-

	x_1	x_2	x_3
M	ab	bab	bbaaa
N	a	ba	bab
	y_1	y_2	y_3

⇒

D_1	D_2	D_3
ab / a	bab / ba	bbaaa / bab

D_1 D_2

ab bab

a ba

We cannot make any combination of dominos which have same numerator and denominator.

∴ This PCP problem is Undecidable.

Ex: $A = (b, bab^3, ba)$ and $B = (b^3, ba, a)$. Input set $\Sigma = \{0, 1\}$

find solution:

Sol: 2, 1, 1, 3 $\quad \frac{ba\,b^3}{ba} \cdot \frac{b}{b^3} \cdot \frac{b}{b^3} \cdot \frac{ba}{a}$ ∴ A solution is 2, 1, 1, 3.

$\Rightarrow w_2 w_1 w_1 w_3 = x_2 x_1 x_1 x_3$

$(bab3a)$

Ex: two list, $X = (b, a, aba, bb)$, $Y = (ba, ba, ab, b)$ have a solution?

Sol: On try combination, we found a sequence 1, 2, 1, 3, 3, 4.

D_1 D_2 D_1 D_3 D_3 D_4

$\frac{b}{ba} \cdot \frac{a}{ba} \cdot \frac{b}{ba} \cdot \frac{aba}{ab} \cdot \frac{aba}{ab} \cdot \frac{bb}{b}$

Date :

Ex: $X = \{100, 0, 1\}$, $Y\{1, 100, 00\}$ · find sol. for PCP?

Sol:

$$\underset{\downarrow}{D_1}\ \underset{\downarrow}{D_3}\ \underset{\downarrow}{D_1}\ \underset{\downarrow}{D_1}\ \underset{\downarrow}{D_3}\ \underset{\downarrow}{D_2}\ \underset{\downarrow}{D_2}$$

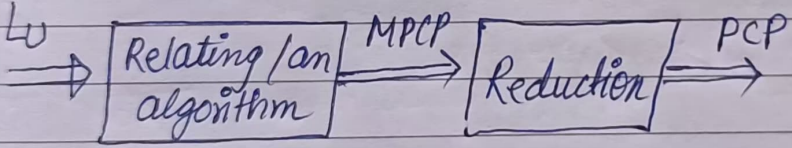
$$\frac{100}{1} \cdot \frac{1}{00} \cdot \frac{100}{1} \cdot \frac{100}{1} \cdot \frac{1}{00} \cdot \frac{0}{100} \cdot \frac{0}{100}$$

∴ The sequence is 1, 3, 1, 1, 3, 2, 2

★ Modified PCP [MPCP]

In MPCP, there is the additional requirements on a solution that the first pair on A and B lists must be the first pair in solution.

- We use MPCP as a medium for proving PCP is undecidable.
- First we reduce L_u (A_{TM}) to MPCP instance.
- We reduce MPCP instance to PCP instance.
- As from reduction from L_u to PCP as MPCP in middle it becomes easy to prove that PCP is undecidable.



Proof : [Neso Academy - (L 113)]

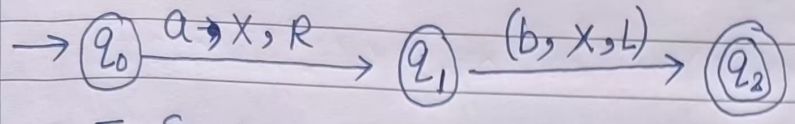
Approach: Take a problem that is already proven to be undecidable and try to convert it to PCP.
If we successfully convert it to an equivalent PCP then we prove that PCP is also undecidable.

Acceptance Problem of TM (This is undecidable Problem)

↓

Convert this to PCP (MPCP)

Date :



$\Sigma = \{a, b\}$ $\quad \Gamma = \{a, b, X, B\}$

i/p : $W = aba$.

Step 1: $\boxed{a|b|a}$ (with q_0 under the first cell) $\quad \frac{\#}{\#q_0 w} \Rightarrow \left[\frac{\#}{\#q_0 aba}\right]$

Step 2: $\delta(q_0, a) = (q_1, X, R)$

$\boxed{a}$ (with q_0) $= \boxed{X}$ (with q_1)

$(q_0 a) = X q_1 \Rightarrow \left[\frac{q_0 a}{X q_1}\right]$

Step 3: $\delta(q_1, b) = (q_2, X, L)$

$\frac{Y|b|}{q_1} = \frac{Y|X|}{q_2}$ $\quad Y \in \Gamma$, $Y \in \{a, b, X, B\}$

$Y q_1 b = q_2 Y X$

$Y = a$: $\left[\frac{a q_1 b}{q_2 a X}\right]$ $\quad Y = b$: $\frac{b q_1 b}{q_2 b X}$ $\quad Y = X$: $\frac{X q_1 b}{q_2 X X}$ $\quad Y = B$: $\frac{B q_1 b}{q_2 B X}$

Step 4: For all possible Tape Symbols.

$\left[\frac{a}{a}\right], \left[\frac{b}{b}\right], \left[\frac{X}{X}\right], \left[\frac{B}{B}\right]$

Step 5: For all possible tape Symbol, after reaching the accepting state accept - (q_2).

$\left[\frac{a q_2}{q_2}\right], \left[\frac{q_2 a}{q_2}\right], \left[\frac{b q_2}{q_2}\right], \left[\frac{q_2 b}{q_2}\right], \frac{X q_2}{q_2}, \frac{q_2 X}{q_2}, \frac{B q_2}{q_2}, \frac{q_2 B}{q_2}$

Step 6: $\left[\frac{\#}{\#}\right] \left[\frac{\#}{B\#}\right]$, domino for # and B Symbol

Step 7: $\left[\frac{q_2 \# \#}{\#}\right]$

Date :

Sol: from Step2 — Step4

$$\left[\frac{\#}{\#q_0aba\#}\right]\left[\frac{q_0a}{xq_1}\right]\left[\frac{b}{b}\right]\left[\frac{a}{a}\right]\left[\frac{\#}{\#}\right]$$

Step3 — Step4

$$\Rightarrow \left[\frac{\#}{\#xq_1ba\#}\right]\left[\frac{xq_1b}{q_2xx}\right]\left[\frac{a}{a}\right]\left[\frac{\#}{\#}\right]$$

⑤ ④ ④

$$\Rightarrow \left[\frac{\#}{\#q_2xxa\#}\right]\left[\frac{q_2x}{q_2}\right]\left[\frac{x}{x}\right]\left[\frac{a}{a}\right]\left[\frac{\#}{\#}\right]$$

⑤

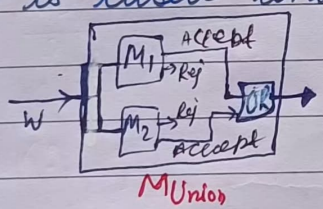
$$\Rightarrow \left[\frac{\#}{\#q_2xa\#}\right]\left[\frac{q_2x}{q_2}\right]\left[\frac{a}{a}\right]\left[\frac{\#}{\#}\right]$$

$$\Rightarrow \left[\frac{\#}{\#q_2a\#}\right]\left[\frac{q_2a}{q_2}\right]\left[\frac{\#}{\#}\right] \Rightarrow \left[\frac{\#}{\#q_2\#\#}\right]$$ Step7 $$\left[\frac{q_2\#\#}{\#}\right]$$

∴ We found a solution for this PCP.
⇒ PCP is undecidable

★ Closure properties of Recursive language (REC):

- REC. is closed under complementation
- REC. is closed under Union.



• If either M_1 OR M_2 accepts then M_U accepts.

- REC. is closed under Intersection [∵ if either M_1 AND M_2 accepts, then M_n accepts]
- REC are closed under–
  - Kleen closer
  - Concatenation

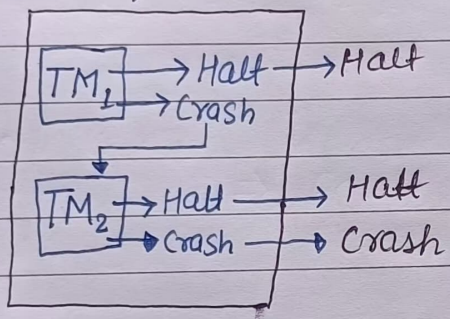
Date :

★ Properties of Recursively enumerable languages: (RE)

Some example of RE:
REC. Lang. , Regular language (RL) , CSL , CFL , etc.

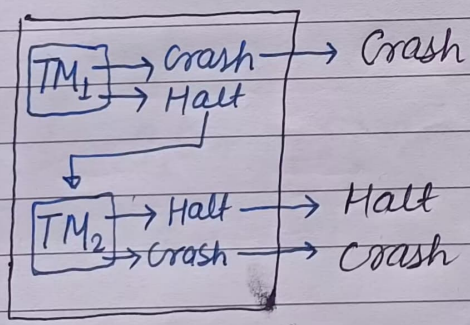
- RE is closed under –
  Union , intersection , Concatenation , Kleen closure
- RE is not closed under –
  - Complementation.

⇒ Union of RE.

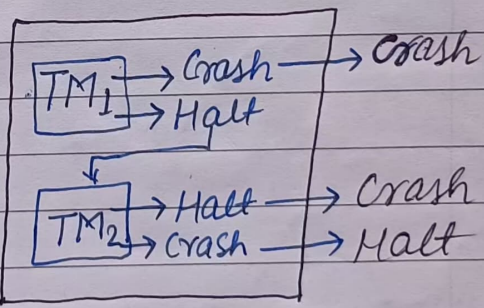


- If TM_1 halt ⇒ System halts.
- If TM_1 crash, System check TM_2 is ready to halt or not?
- If TM_1 not halt and TM_2 halt ⇒ System halt

⇒ Intersection of RE.

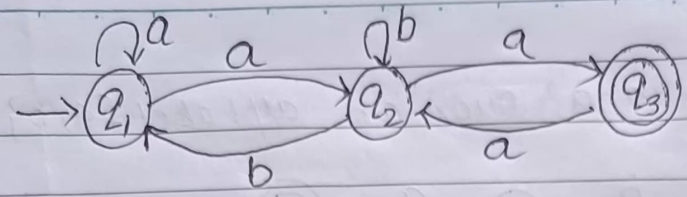


⇒ Complement of RE



Ardene's theorem Questions [unit-1]

Date :

Ques: 

, Find RE using ardene's theorem.

Sol:

$$q_1 = q_1a + q_2b + \varepsilon$$
$$q_2 = q_1a + q_2b + q_3a$$
$$q_3 = q_2a$$

Substituting q_3 in q_2:-

$$q_2 = q_1a + q_2b + q_2aa$$
$$= q_1a + q_2(b + aa) \quad [\Rightarrow R = QP^*]$$
$$R = Q + RP$$
$$q_2 = q_1a(b+aa)^*$$

Substituting q_2 in q_1:

$$q_1 = q_1a + q_1a(b+aa)^*b + \varepsilon$$
$$= q_1(a + a(b+aa)^*b) + \varepsilon$$
$$R \quad RP \quad + Q$$

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

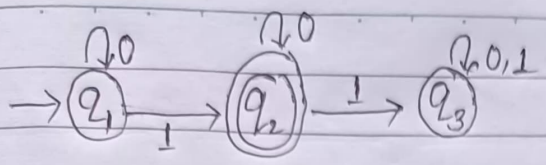
$\Rightarrow q_3$ is the final state.

$$\therefore q_3 = q_2a$$

$$\Rightarrow \boxed{R.E. = (a + a(b+aa)^*b)^*ba(b+aa)^*a}$$

Date :

Ex:



Sol:

$q_1 = q_1 0 + \varepsilon$

$q_2 = q_1 1 + q_2 0$

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

$\Rightarrow q_1 = q_1 0 + \varepsilon \quad (R = RP + Q)$

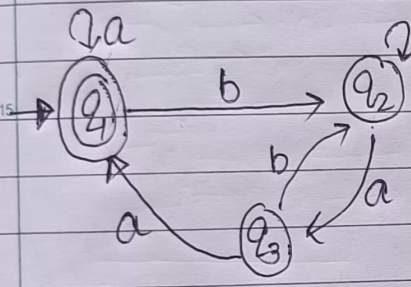
$\Rightarrow q_1 = \varepsilon 0^*$

$\Rightarrow$ Substituate q_1 in q_2 –

$q_2 = 0^* 1 + q_2 0 \quad [R = Q + RP]$

$q_2 = 0^* 1 0^*$, $\boxed{RE = 0^* 1 0^*}$

Ex



Sol: $q_1 = q_1 a + q_3 a + \varepsilon$

$q_2 = q_1 b + q_2 b + q_3 b$

$q_3 = q_2 a$

$\Rightarrow$ Substituating q_3 in q_2 –

$q_2 = q_1 b + q_2 b + q_2 a \cdot b$

$q_2 = q_1 b + q_2 (a + ab) \quad [R = Q + RP]$

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

Put q_2 in q_1 :

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

$q_1 = q_1 (a + b(a+ab)^* aa) + \varepsilon \quad [R = RP + Q]$

$\therefore \boxed{q_1 = (a + b(a+ab)^* aa)^*}$