Theory of Computation Notes

Contributor: Shakshi

[Mata Sundri (DU)]

Computer Science Notes

Page No.
Date

23/7/19.

Theory of Computation : Daniel A. Cohen

Chapter-2.

→ Alphabet - (Σ) - It is a finite set of symbols.

$\Sigma = \{a, b, c\}$

⇒ Language - Any set of strings over an alphabet Σ defined acc. to the conditions made by the rule. It can be finite as well as infinite.

⇒ Words - Set of symbols which is unique.

⇒ String - finite sequence of symbols from the alphabet.

If a string has no symbol then its length is 0, and it is known as empty string or null word. (Λ)
word having length 0.

NOTE - Λ cannot be a part of an alphabet.

EXAMPLE - $\Sigma = \{x\}$

$L_1 = \{ x^n \quad n = 0, 1, 2, \ldots \}$

that is $L_1 = \{\Lambda, x, xx, xxx, \ldots\}$
(no. of words is ∞. L_1 is infinite lang)

$L_2 = \{x, xxx, xxxxx, \ldots\}$

Page No.
Date

$L_3 = \{ xx, xxxx, xxxxxx, \ldots \}$

EXAMPLE $\Sigma = \{a, b\}$

$L_1 = \{aa, ab, ba, bb\}$ (length of strings is 2)
(finite language)

Eg. ⇒ if L can have strings of any length where each word has a finite length.

$L = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$
(infinite language)

⇒ Null Set (ϕ) - language having no word.

$L_1 = \{ \}$ or ϕ

Even it do not have null word that is Λ

$L_2 = \{ \Lambda \}$

$L_1 \neq L_2$

Ques. When will $L = L + \{\Lambda\}$?

It is only possible if $\{\Lambda\} \subseteq L$.

Ques. Is $L + \phi = L$? Yes. (Give example)

$\phi = \{ \}$

Let $L = \{a\}$

$\phi + L = \{ \} + \{a\} = \{a\} = L$

Page No.
Date

⇒ Closure (*) :-
↳ Kleene's Closure

We can apply closure on Σ and L.

Σ^* - result will be set of strings and it will have string of all possible length including ~~ϕ or ϵ~~. Λ.

EXAMPLE :- $\Sigma = \phi$ (Alphabet having no symbol)

Note → closure ~~of~~ ~~alphabet~~ is always ∞.

$\Sigma^* = \{\Lambda\}$ — 2 Exceptions

L or $S = \{\Lambda\}$
$S^* = \{\Lambda\}$

⇒ $L = \{\Lambda, a, b, aa, ab, ba, bb\}$

$L^* = \{\Lambda, a, b, aa, ab, ba, bb,$ ~~aa~~ $aaa, aab, aba, abb, baa, bab, bba, bbb, \ldots\}$ (infinite)

in ascending order

⇒ $S = \{a, b, ab\}$ or $\{a, b, bb\}$,

$S^* = \{\Lambda, a, b, ab, ba, aab, aba, baa, bab, bbb, \ldots\}$

Two different sets can have same Kleene closure.

Page No.
Date

⟹ S^+ = strings of all possible lengths >0.
(positive closure)

Q. Is it possible $L^+ = L^*$? Yes

if Language L contains Λ

$L = \{\Lambda, a\}$

$L^* = \{\Lambda, a, aa, aaa, \ldots\}$

$L^+ = \{\Lambda, a, aa, aaa, \ldots\}$

Q. Is it possible that $\Sigma^+ = \Sigma^*$ (NO)

Q. Prove:- $S^* = S^{**}$

Q. $S^+ = SS^*$ if S

Page No.
Date

24/July/2019

CH-4
Regular Expression

⇒ $\Sigma = \{a, b\}$

L = {aa, aba, abba, abbba, abbbba, ...}

Regular Expression = ab^*a

(b^* means 0 or more occurrences of b)

L = { aa, ab }

R.EX = a (a+b) or aa + ab

{ λ, a, b, aa, ab, bb, aaa, aab, abb, bbb, aaaa, ... }

R.EX = $a^* b^*$

Q. Is $a^* b^* = (ab)^*$ (NO)

$(ab)^*$ = { λ, ab, abab, ababab, ... }

$a^* b^*$ = { λ, a, b, aa, ab, bb, ... }

abab is not present in a^*b^* because a cannot occur after b.

aa is not present in $(ab)^*$ because a is always followed by b.

→ L = { a, c, ab, cb, abb, cbb, abbb, cbbb, abbbb, ... }

R.EX = $(a+c)\, b^*$

$b^+ = bb^*$

Page No.
Date

⇒ L = { aaa, aab, aba, abb, baa, bab, bba, bbb }

R.Ex = $(a+b)(a+b)(a+b)$

⇒ L = { λ, a, b, aa, ab, ba, bb, aaa, aab, a ... }

R.Ex = $(a+b)^*$ ↳ or

If λ not included

R.Ex = $(a+b)(a+b)^*$

atleast one a or b will be selected.

⇒ Language having atleast 2 a's

$\Sigma = \{a, b\}$

R.Ex = $(a+b)^* \, a \, (a+b)^* \, a \, (a+b)^*$

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

⇒ L - having exactly 2 a's - $b^* a b^* a b^*$

1. - having atleast 1 a & 1 b - $\cancel{(a+b)^*}$ $(a+b)^*$ ...

2. - Cannot have double 'a' or double 'b'

3. - Cannot have double 'a'.

Ans' 1. $(ab)^* \, a \, (a+b)^* \, b \, (a+b)^* + (b^* b a a^*)$

or $\left(a (a+b)^* b a^* \right) + \left(b (a+b)^* a b^* \right)$

Ans 3 $\cancel{b^* a b^*}$ $b^* (ab b^*)^* (\Lambda + a)$

Ans. 2 $(\Lambda + b)(ab)^*(\Lambda + a)$

25/July/2019.

Page No.
Date

⟹ $(a+b)^* ab (a+b)^* + b^* a^*$

$L \equiv (a+b)^*$

⟹ $(a+b)^* a (a+b)^* + b^*$

$L \equiv (a+b)^*$

⟹ $(a+b)^* a (a+b)^* b (a+b)^*$

L ≡ Set of all words where atleast one a proceeds atleast one b.

⟹ $L \neq \{\cancel{\lambda}, \cancel{a}, \cancel{bb}, \cancel{ab}, \cancel{aaa}, \cancel{bbbb}\}$

$L = \{abba, baaa, bbbb\}$

R.Ex = abba + baaa + bbbb

⟹ Regular Expression (R.Ex)

The set of regular expressions is defined by the following rules -

* Every letter of Σ can be made into a R.Ex. λ is also a REx

* If r_1 and r_2 are R.Ex and so are following (r_1), $r_1 r_2$, $r_1 + r_2$, r_1^*

⟹ $L = \{\Lambda, a, b, ab, bb, abb, bbb, abbb, bbbb, \ldots\}$

$ab^* + b^*$

26/July/2019

Page No.
Date

⟹ Even-even language - Even no. of a's & b's.

⟹ Non-regular languages - L for which regular expression cannot be formed

⟹ Even-even language.

R.Ex = $[aa + bb + (ab+ba)(aa+bb)^* (ab+ba)]^*$

⟹ $(a^*)^* = a^*$ ✓

$(a+b^*)^* = (a+b)^*$ ✓

$(aa+ab)^* = (aa+ab^*)^*$ ✗ (Eg. abb)

⟹ Regular languages -

The following rules define the L associated with any regular expression -

* The L associated with any regular exp. is just a single letter is that one letter word alone and the language associated with Λ is $\{\Lambda\}$ a one word language.

$(r_1^*) = L_1^*$ $(r_1 r_2) = L_1 L_2$

$(r_1 + r_2) = L_1 + L_2$

⟹ $L = \{\Lambda, x, xx, xxx, xxxx\}$

R.Ex = $\Lambda + x + xx + xxx + xxxx$

or $(\Lambda + x)^4$

or $(\Lambda+x)(\Lambda+x)(\Lambda+x)(\Lambda+x)$

26|July|2019

Page No.
Date

CHAPTER-5

Finite Automata / Acronym (FA)

It is a collection of following -

* finite set of states where one (1) is an initial state also called as start state and some or maybe none, of which are final state.

* An input alphabet Σ over which the FA is defined.

* A finite set of transitions that tell for each state and for each letter of the input alphabet which state to go to next.

→ Transitions (δ)

$\delta_0(S_1, 0) = S_1$

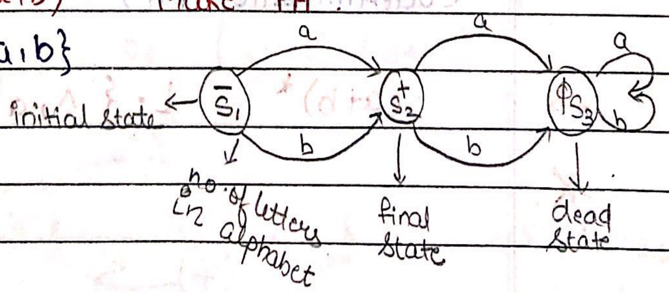
$\delta(S_1, 1) = S_2$

	0	1	
S_1	S_1	S_2	Initial ⊝ or ⊖
S_2	S_1	S_3	
S_3	S_1	S_4	
S_4	S_1	S_3	final ⓞ ⊕

Φ - dead state

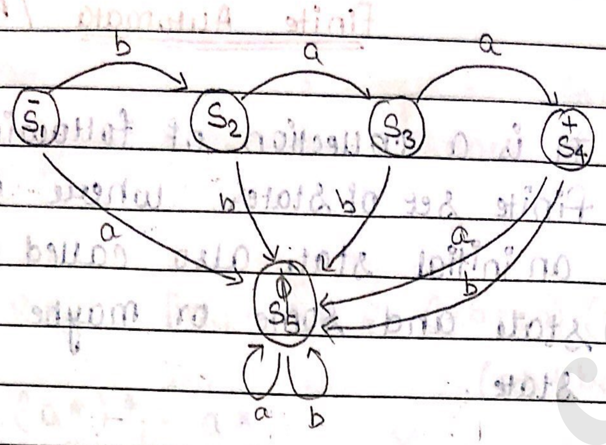
⇒ R.Ex. (a+b) (Make FA)

L = {a, b}



Page No.
Date

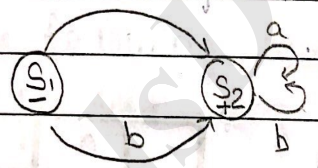
⇒ L = { baa }



⇒ R.Ex - a+b (a+b)*

atleast a 'a' or 'b'.

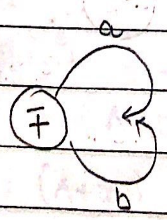
or, all strings of length > 0



FA is deterministic in nature that is an input will always go to next state or will remain in the same state. Thus FA is aka DFA (Deterministic FA).

→ (a+b)* L = { Λ, a, b, ab, aa, ... }

*

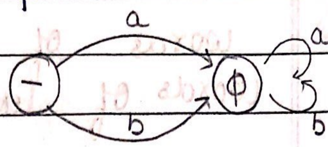


Page No.
Date

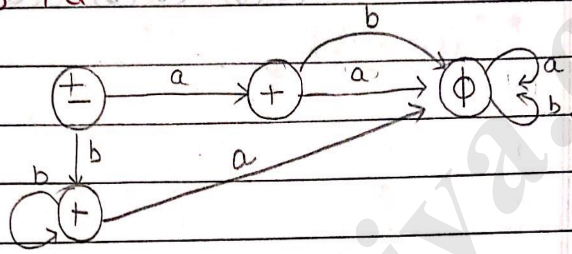
⇒ Null language does not have a final state.

$L = \{ \}$ or ϕ

F.A ⇒



⇒ $b^* + a$ - {Λ, a, b, bb, bbb, bbbb, ...}



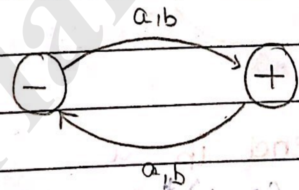
30/July/2019

⇒ finite automata which accepts odd length

{a, b, aaa, aab, aba, baa, bbb, aaaa, bbbb, ...}

R.Ex - $(a+b)\ (aa + ab + ba + bb)^*$

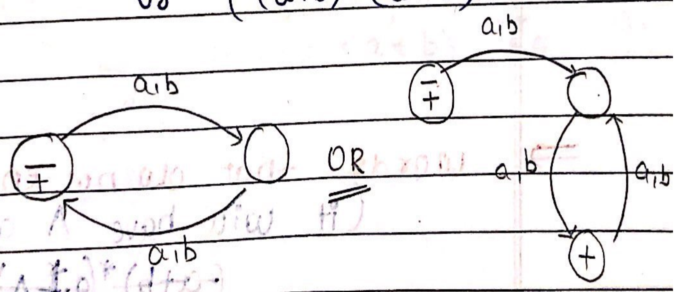
or $(a+b)\ ((a+b)\ (a+b))^*$



⇒ Even length words

R.Ex - $(aa + ab + ba + bb)^*$

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

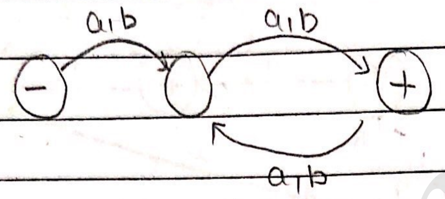


Page No.
Date

if $L_1 \subset L_2$ then automata of L_2 will also accepts words of L_1.

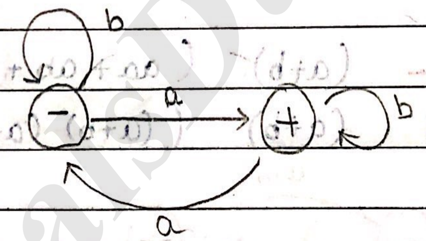
⇒ words of even length excluding words of length 0.

R.Ex - $(a+b)(a+b)((a+b)(a+b))^*$

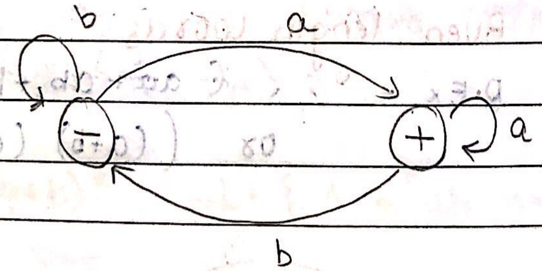


⇒ Odd no. of a's

R.Ex - $b^* a (b^* a b^* a)^* b^*$



⇒ L with end in a
R.Ex - $(a+b)^* a$

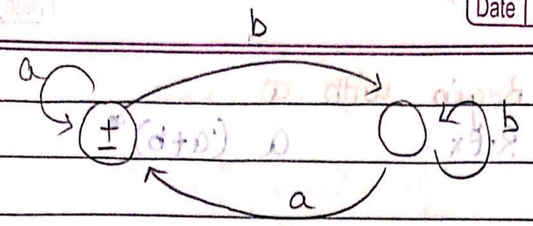


⇒ words that do not end in b (it will have Λ also)

~~$(a+b)^*(a+\Lambda)$~~

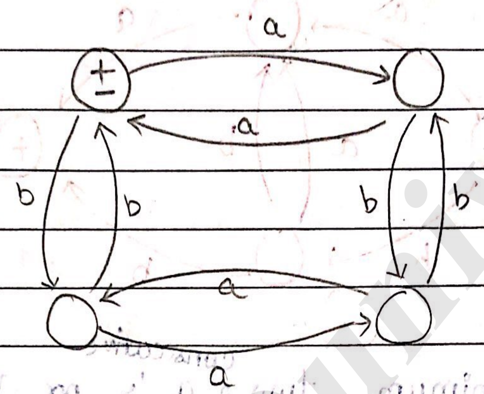
$((a+b)^* a)^*$

Page No.
Date



⇒ Even - even language

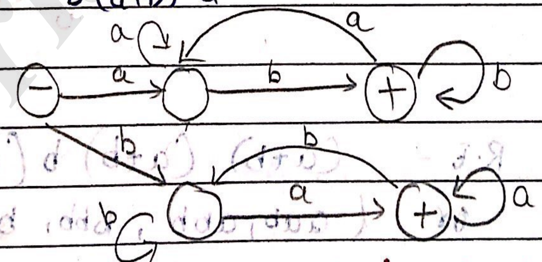
R.Ex - [aa+bb+ (ab+ba) (aa+bb)*
(ab+ba)]*



⇒ Words having different starting and ending state

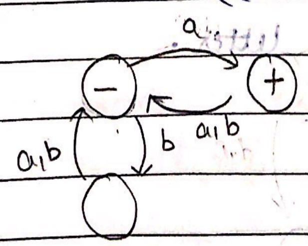
R.Ex = a(a+b)*b
+
b(a+b)*a

a —— b
or b —— a



⇒ Words having odd length that ends in a.

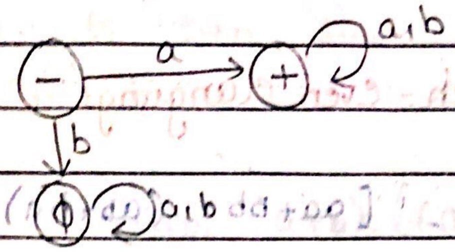
R.Ex - ((a+b).(a+b))*a

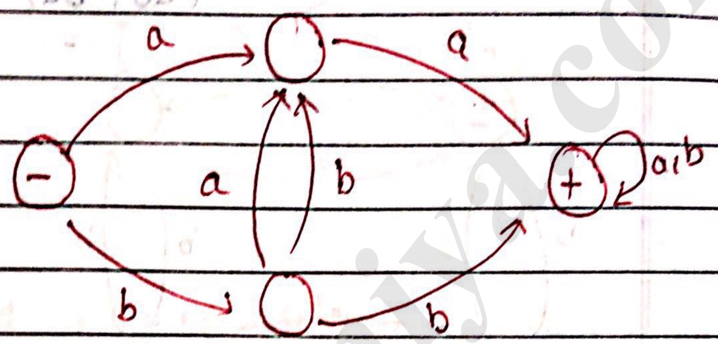


Date

→ Begin with a

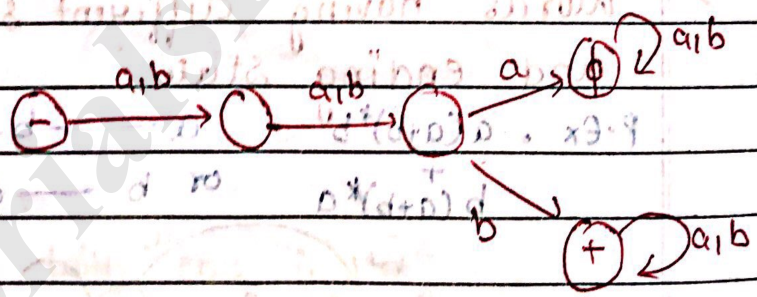
R.Ex - $a(a+b)^*$





Minimum two consecutive a's or b's.

R.Ex - $(a+b)^*(aa+bb)(a+b)^*$



R.E - $(a+b)(a+b)\,b\,(a+b)^*$

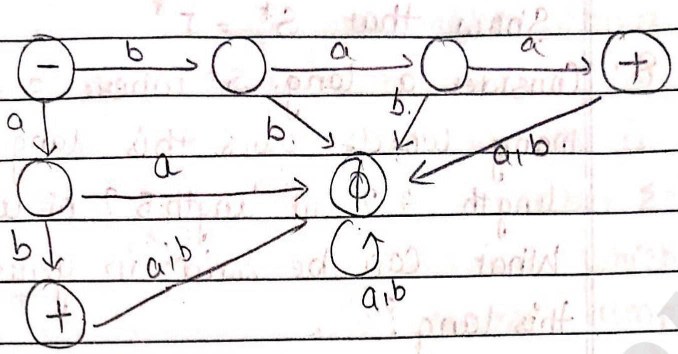
or $(aab, abb, bbb, bab)\,b\,(a+b)^*$

(All words) with 'b' as the third letter.

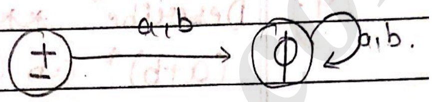
Page No.
Date

H.W.

⇒ L = { baa , ab }



⇒ L = {Λ}



(1) ⇒ Draw a machine for a language, ~~FA~~ accepts ~~lang~~. ~~with~~ all words with a 'aa'.

→ (*) Words that have even no. of letter in total. (even length word)

(2) → That accepts only those words that do not end in 'ba'. $(a+b)^*(aa+ab+bb)$

(3) Words that begin or end with a double letter.

(4) That have both the letters a & b in them.

(5) Language with only a's or b's.

3/7/19

(6) Words that do not have 'aa' as a substring (R.Ex)

Page No.
Date

7 Let $S = \{ab, bb\}$ & $T = \{ab, bb, bbbb\}$
Show that $S^* = T^*$

8. Consider a lang. S^* where $S = \{aa, b\}$. How many words does this lang have of length 4? of length 5? of length 6? What can be said in general about this lang?

9. Describe ** lang for R.Ex
$(a+b)^* \, ab \, (a+b)^*$
$((a+b)b)^*$

10. F.A that accepts fewer than 4 letters

11. R.Ex. for words that do not have a double letter.

12. $S = \{ab, bb\}$ $T = \{ab, bb, bbb\}$ Show that $S^* \neq T^*$ but $S^* \subset T^*$

13. R.Ex for lang of all the words that do not have a substring 'ab'.

14. (6 marks) lang. of all the strings of length 4 or more such that next to last second last letter is equal to 2nd letter of input string

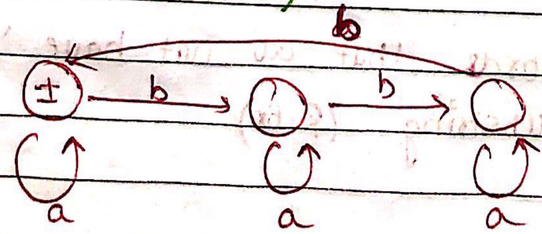
15. Explain with example: $(S^*)^+ = (S^+)^*$

16. (5 marks) R.Ex & FA lang of all words that have an even no. of all substrings 'ab' in them.

17. Is $(a^* b^*)^* = (a+b)^*$?

18. FA that accepts all strings that have an even length that is not divisible by 6. (3 marks)

19.



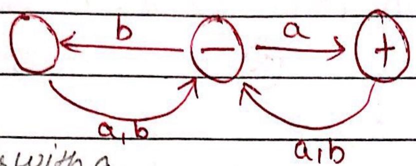
Page No.
Date

19+) FA for $(bb + bba)^* a$

20+) FA for all strings that either starts with ab or ends with ba.

21+) DFA for lang where every '00' is followed by 1 over the alphabet $\Sigma = \{0,1\}$

22+) R.Ex for lang having words in which a appears tripled i.e; in ~~external~~ lumps of 3, if at all.

24. 

Describe the language.

ends with a

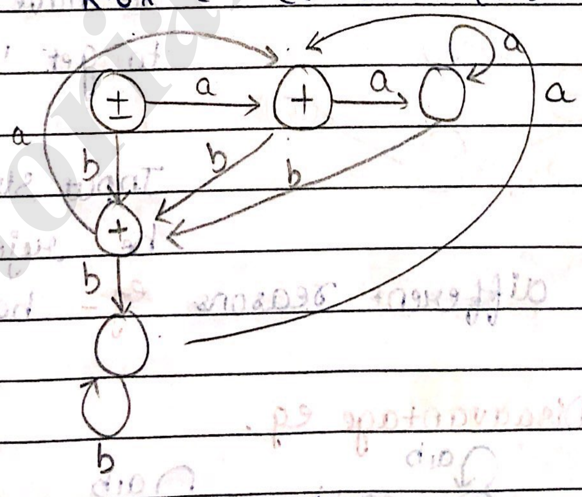
a, aaa, aba, ababa, abaaa, baa, bba, bbaba

25. $bba^*(a+b)$
$((a+b)a)^*$] describe the language.

Ch.5 Q. 18, 19, 20.

26. Words that donot end in double letter. (FA) (R.Ex)

Ans R.Ex = $(a+b)^*(ab + ba)$



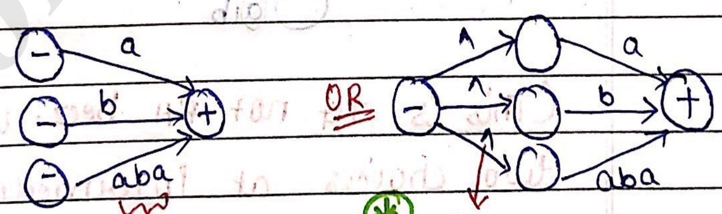
Page No.
Date

⇒ Advantage
* Easier to see which lang. the graph is using.

→ A transition Graph (TG) is a collection of the following:-
* a finite set of states of atleast one of which is designated as start state and some (may be none) of which are designated as final state.
* An Σ of possible i/p letters from which i/p strings are formed.
* A finite set of transitions that show how to go from one state to another based on reading some specified substrings of i/p letters. possibly even Λ.

EXAMPLE - (It can have more than one initial state)

$L = \{ a, b, aba \}$



Can be words also.

to reduce initial state (if you want to show transition without consuming any letters).

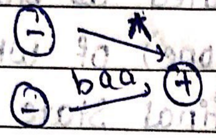
TG

Page No.
Date

* (−) this is a possible T.G it does not accept anything not even Λ.

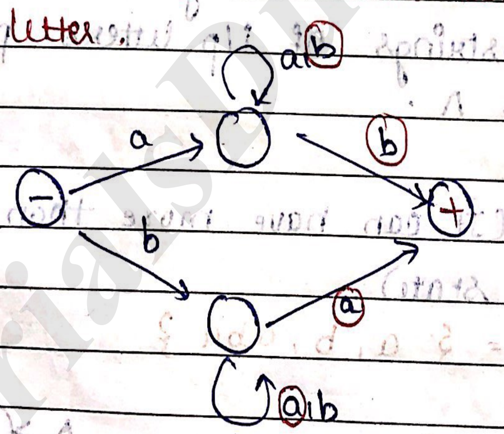
* (+) this T.G will accept only Λ.

* L = {Λ, baa}



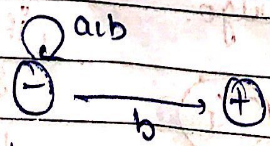
if i/p is baab the machine will crash becz your whole string is not consumed. ∴ the string will be rejected.

* word start & ends with different letter.



(This is TG not FA becz we have two choices at Intermediate state)

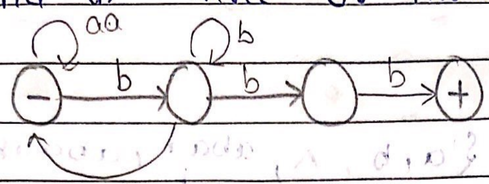
* L with word ending is 'b'.



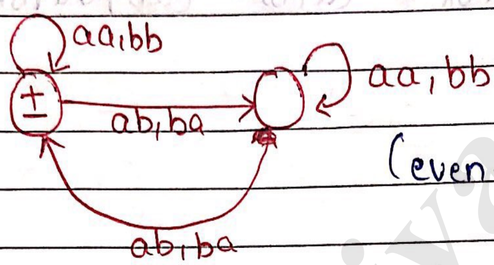
T.G has all possible, of i/p combination ∴ it is complex.

Page No.
Date

⟹ language of all words in which a's occur only in even clums and that end in three or more b's.



⟹



(even-even lang.)

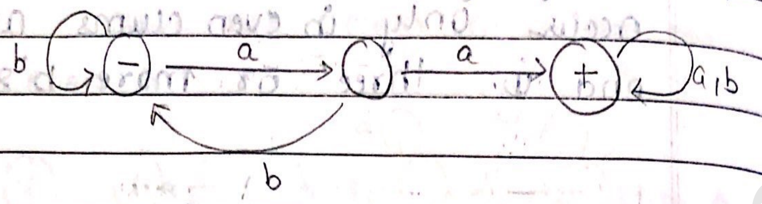
Pg. 94

ANSWERS

Finite Automata

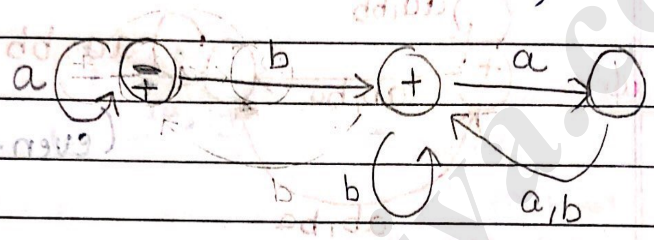
CHAPTER-5

(1) L = {aa, baa, abaa, baab, ...}

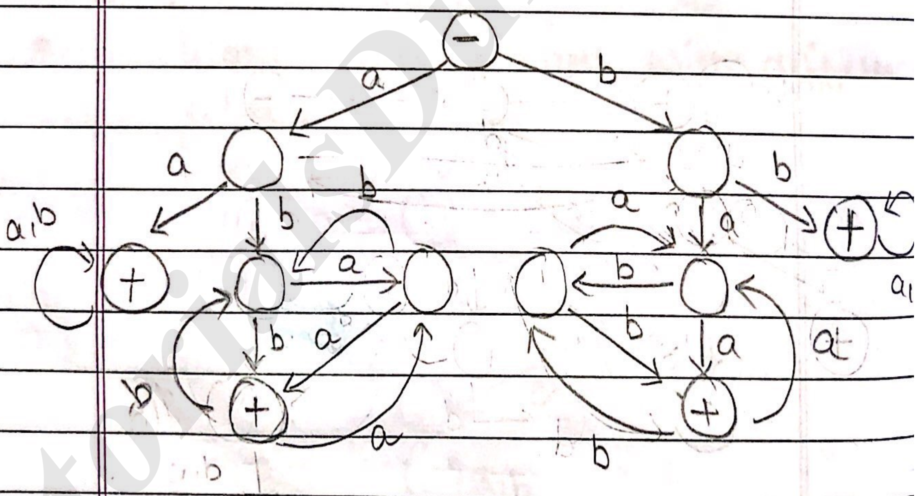


(2) L = {a, b, Λ, abaa, abaab, baabba, ...}

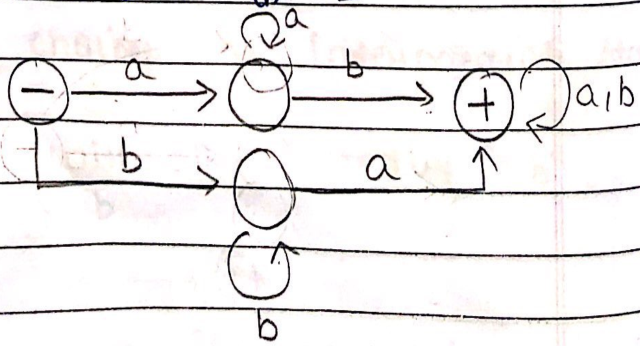
R.Ex = $(a+b)^*\,(\overset{a+b}{ab}+aa+bb)$



(3) begin or end with double letter.

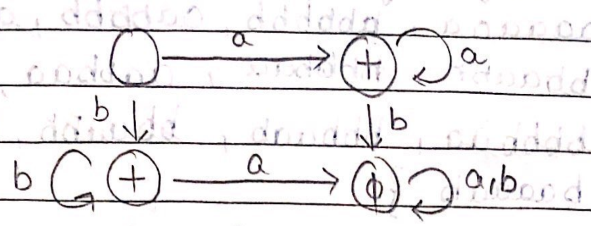


(4) both a & b in L.

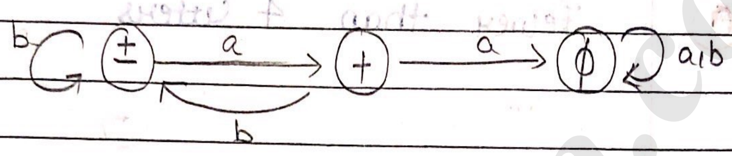


Page No.
Date

(5) L with only a or b



(6) L donot have 'aa'.



(7) S = {ab, bb} T = {ab, bb, bbbb}

S* = {Λ, ab, bb, abbb, bbab, abab, bbbb, ababab, ababbb, abbbab, bbabab, bbbbab, abbbbb, bbbbbb, ...}

T* = {ab, bb, bbbb, Λ, abbb, abbbbb, abab, bbab, bbbb, bbbbab, bbbbbb, ...}

Since S ⊆ T and all combinations bbbb can be obtained in S* using any number of bb in set S.

∴ S* = T*.

(8) S = {aa, b} S*

S* words of length 4
{aabb, aaaa, bbbb, baab, bbaa}

S* words of length 5
{aaaab, aabaa, baaaa, bbbaa, bbaab, baabb, aabbb, bbbbb}

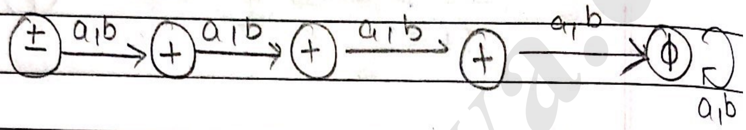
Page No. ____
Date

S^* of length 6

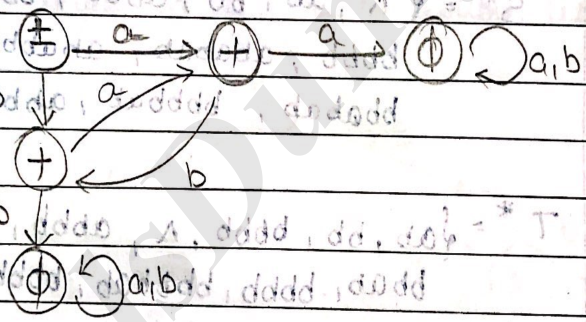
{ aaaaaa, bbbbbb, aabbbb, aaaabb, bbaabb, bbbbaa, aabbaa, bbaaaa, ~~bbbbaa~~, bbbaab, ~~bbaabb~~, baabbb}
baaaab }

Language - a will be occuring in even clum if it ever occuring.

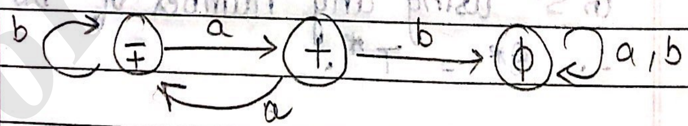
(10) fewer than 4 letters



(11) do not have double letter.



(13) donot have substring 'ab'

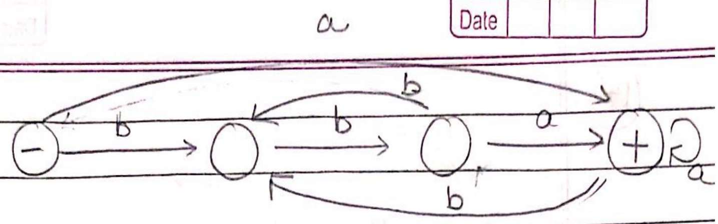


(19) Language having 0 or 3 b's.

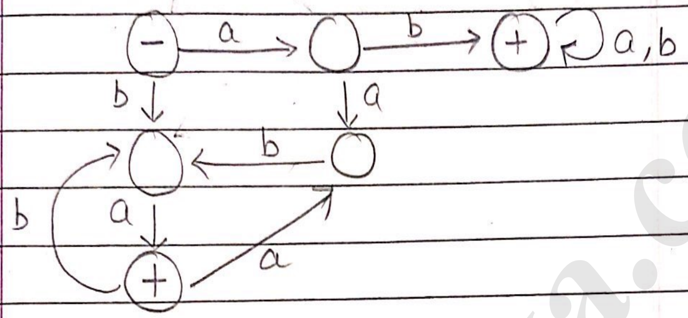
R.Ex - $(a^* b a^* b a^* b a^*)^*$

(20) $(bb + bba)^* a$

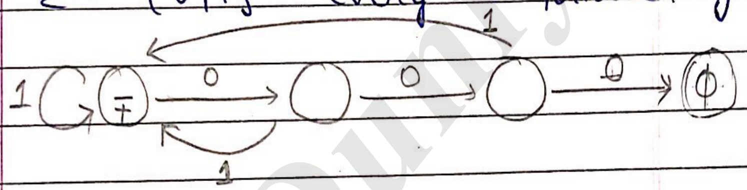
L ⇒ ends with a
even no. of b in with
b's occurs in pairs.



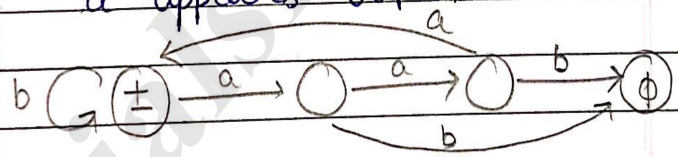
21 start with 'ab' or ends with 'ba'



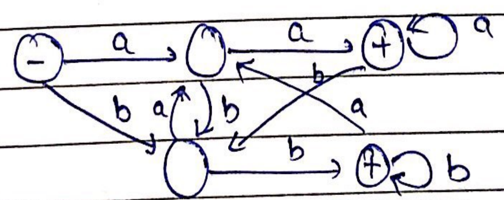
22 $\Sigma = \{0,1\}$ every 00 followed by 1.



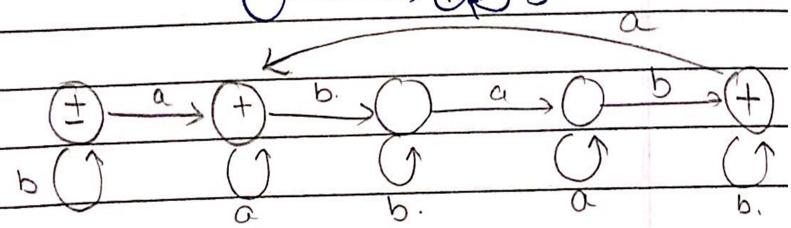
23 a appears tripled



~~24~~ Q. ends with double letter.

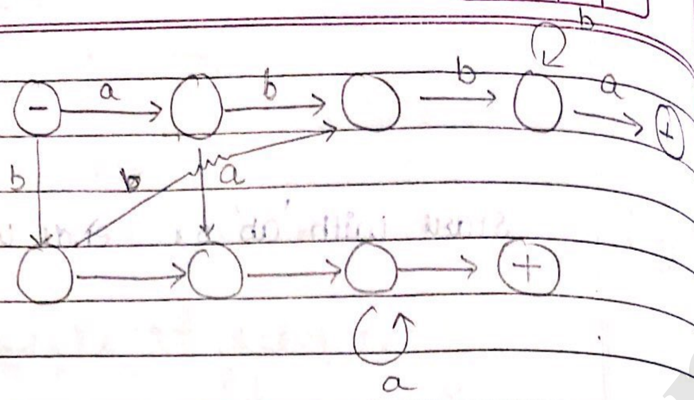


16



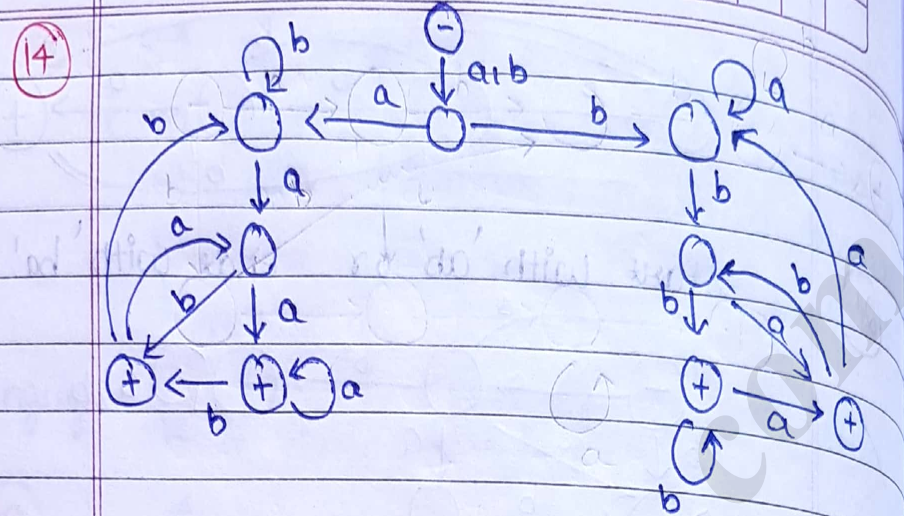
Page No.
Date

14

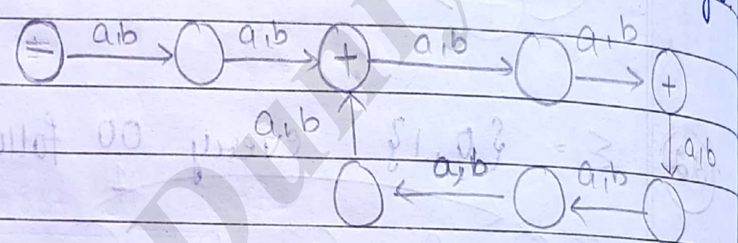


Page No.
Date

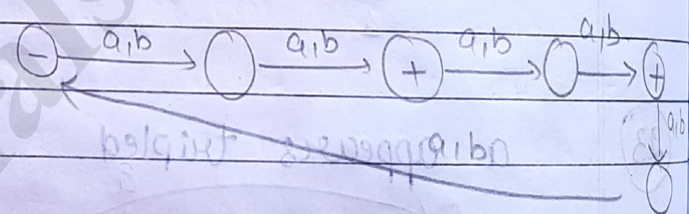
(14)



(18) even length that is not divisible by 6.



or



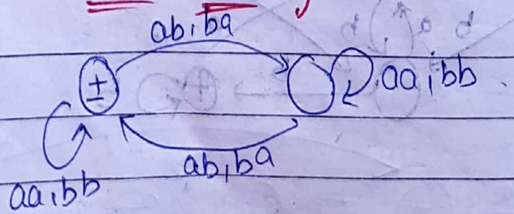
21/AUG/2019

CHAPTER-7

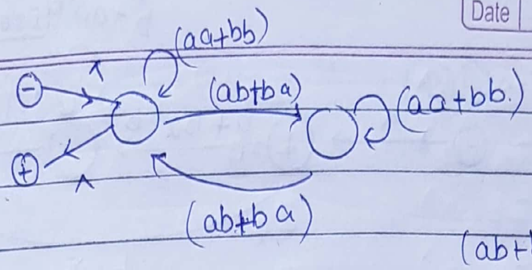
// Study from book.

→ State bypass theorem

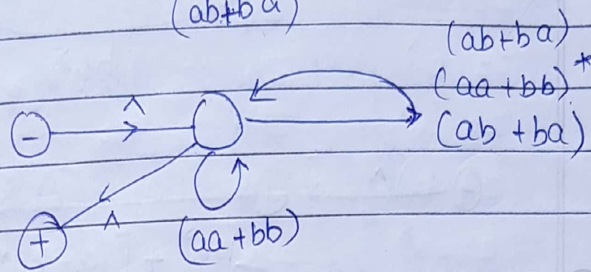
Q. Even - Even lang.



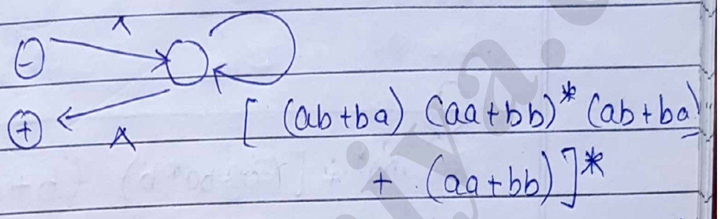
Step 1



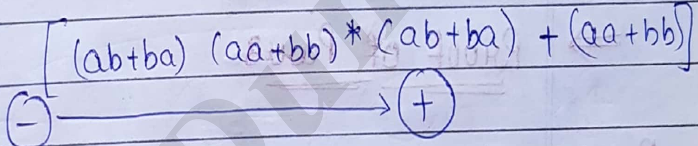
Step 2



Step 3

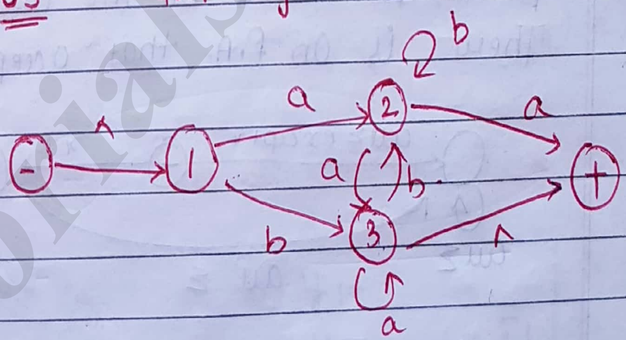


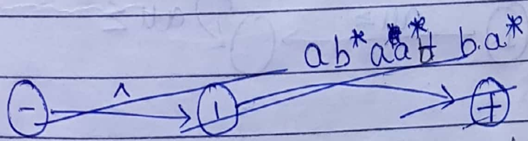
Step 4 -



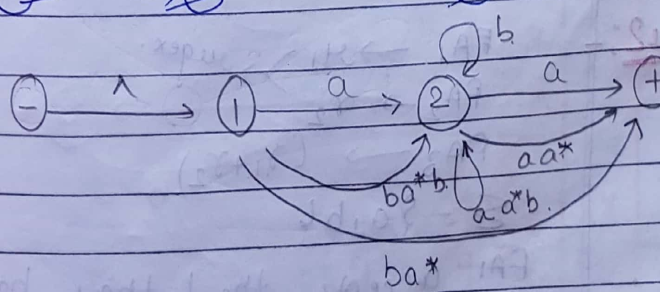
Pg. 105 – Graph Eg.

Q.





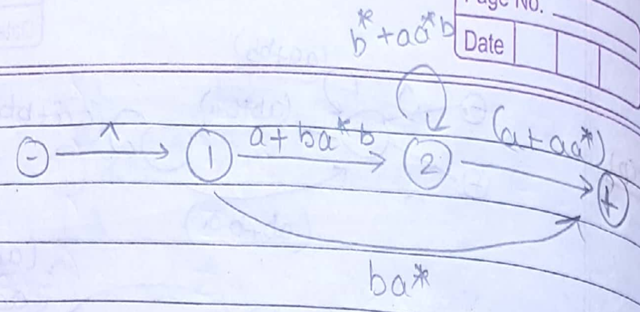
Step 1

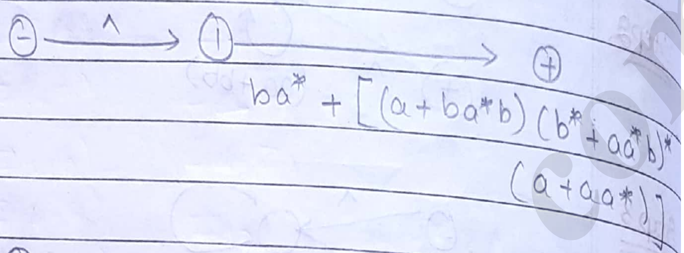


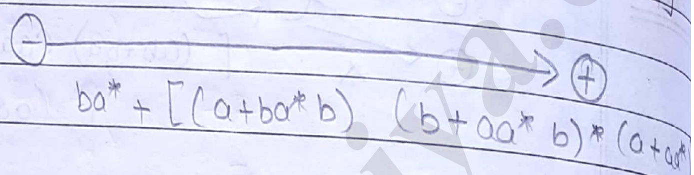
Scanned by CamScanner

Page No.
Date

Step 2 :

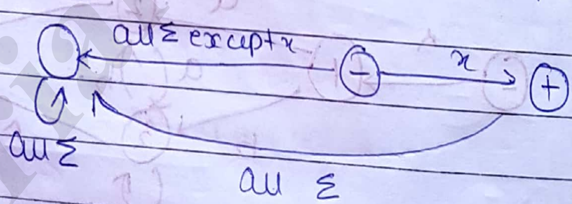


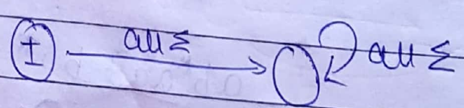




⇒ **Proof of Part 3**

Rule 1:- There is an F.A that accepts any particular letter of the alphabet.
There is an F.A that accepts only Λ.





Rule 2:-

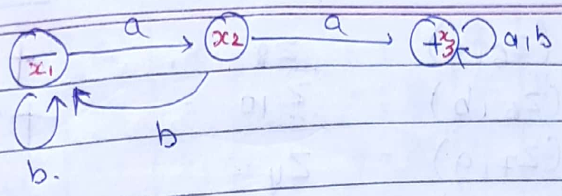
$FA_1 \rightarrow r_1$
$FA_2 \rightarrow r_2$ } regex.
$FA_3 \rightarrow (r_1 + r_2)$

$\Sigma = \{a, b\}$

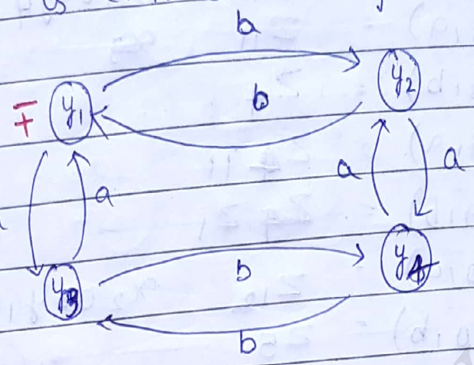
FA_1 accepts the L that have 'aa' somewhere in them.

Scanned by CamScanner

Page No. ____
Date



FA_2 is even even lang.



Transition table.

FA1	a	b
$-x_1$	x_2	x_1
x_2	x_3	x_1
$+x_3$	x_3	x_3

FA2	a	b
$\pm y_1$	y_3	y_2
y_2	y_4	y_1
y_3	y_1	y_4
y_4	y_2	y_3

Applying Rule to make $FA_3 \rightarrow (r_1 + r_2)$

In FA_3 $(-)$ $z_1 = x_1$ or y_1

(z_1, a) $z_2 = x_2$ or y_3

(z_1, b) $z_3 = x_1$ or y_2

$(z, \text{In } z_2) \Rightarrow$ $z_4 = x_3$ or y_1

going to a
(z_2, b) $z_5 = x_1$ or y_4

(z_3, a) $z_6 = x_2$ or y_4

$(z_3, b) \longrightarrow z_1$

(z_4, a) $z_7 = x_3$ or y_3

(z_4, b) $z_8 = x_3$ or y_2

(z_5, a) $z_9 = x_2$ or y_2

(z_5, b) $z_{10} = x_1$ or y_3

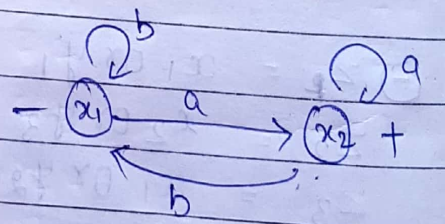
	a	b
$\pm z_1$	z_2	z_3
z_2	z_4	z_5
z_3	z_6	z_1
$+z_4$	z_7	z_8
z_5	z_9	z_{10}
z_6	z_8	z_{10}
$+z_7$	z_4	z_{11}
$+z_8$	z_{11}	z_4
$\pm z_9$	z_{11}	z_{11}
z_{10}	z_{12}	z_5

Page No.
Date

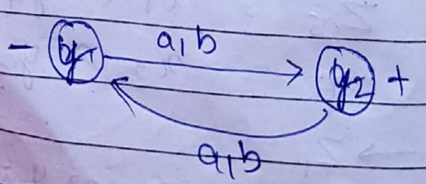
$(z_6, a) = z_8$ $+ z_{11}$ | z_8 | $z_...$
$(z_6, b) = z_{10}$ $+ z_{12}$ | z_4 | $z_...$
$(z_7, a) = z_4$
$(z_7, b) = z_{11} = x_3$ or y_4
$(z_8, a) = z_{11}$
$(z_8, b) = z_4$
$(z_9, a) = z_{11}$
$(z_9, b) = z_1$
$(z_{10}, a) = z_{12}$ x_2 or y_1
$(z_{10}, b) = z_5$
$(z_{11}, a) = z_8$
$(z_{11}, b) = z_7$
$(z_{12}, a) = z_7$
$(z_{12}, b) = z_3$

Q. FA_1 — all words having odd no. of letter
FA_2 — all words which end in 'a'.
FA_3 — ? either odd no. of letter or end in 'a'.

FA_2 ⇒



FA_1 ⇒



Scanned by CamScanner

Page No.
Date FA1

FA2

	a	b
$- x_1$	x_2	x_1
$+ x_2$	x_2	x_1

	a	b
$- y_1$	y_2	y_2
$+ y_2$	y_1	y_1

$z_1 = x_1$ or y_1

$(z_1, a) = x_2$ or $y_2 = z_2$

$(z_1, b) = x_1$ or $y_2 = z_3$

$(z_2, a) = x_2$ or $y_1 = z_4$

$(z_2, b) = z_1$

$(z_3, a) = z_4$

$(z_3, b) = z_1$

$(z_4, a) = z_2$

$(z_4, b) = z_3$

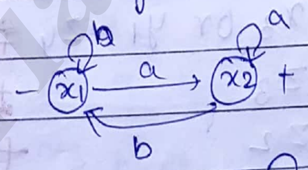
FA3

	a	b
$- z_1$	z_2	z_3
$+ z_2$	z_4	z_1
$+ z_3$	z_4	z_1
$+ z_4$	z_2	z_3

Q. FA_1 — words ending in a.
FA_2 — " " " b.
FA_3 — ending in a or b.

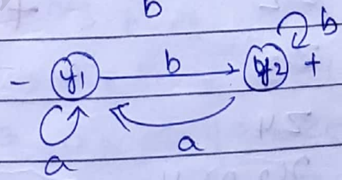
Ans:

$FA_1 \rightarrow$



	a	b
$- x_1$	x_2	x_1
$+ x_2$	x_2	x_1

$FA_2 \Rightarrow$



	a	b
$- y_1$	y_1	y_2
$+ y_2$	y_1	y_2

$z_1 = x_1$ or y_1

$(z_1, a) = x_2$ or $y_1 = z_2$

$(z_1, b) = x_1$ or $y_2 = z_3$

$(z_2, a) = z_2$

$(z_2, b) = z_3$

$(z_3, a) = z_2$

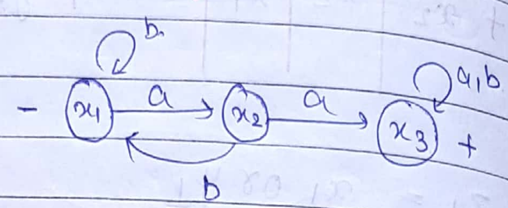
$(z_3, b) = z_3$

	a	b
z_1	z_2	z_3
$+ z_2$	z_2	z_3
$+ z_3$	z_2	z_3

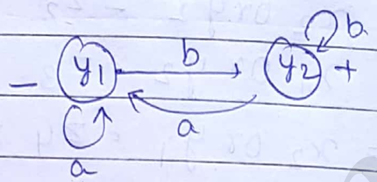
Page No.
Date

Q. FA_1 = double 'a' in them.
FA_2 = ending in 'b'.

FA_1 -



FA_2 =



	a	b
$-x_1$	x_2	x_1
x_2	x_3	x_1
$+x_3$	x_3	x_3

	a	b
$-y_1$	y_1	y_2
$+y_2$	y_1	y_2

$z_1 = x_1$ or y_1

$(z_1, a) = x_2$ or $y_1 = z_2$
$(z_1, b) = x_1$ or $y_2 = z_3$
$(z_2, a) = x_3$ or $y_1 = z_4$
$(z_2, b) = z_3$
$(z_3, a) = z_2$
$(z_3, b) = z_3$
$(z_4, a) = z_4$
$(z_4, b) = x_3$ or $y_2 = z_5$
$(z_5, a) = z_4$
$(z_5, b) = z_5$

	a	b
$-z_1$	z_2	z_3
$+z_2$	z_4	z_3
$+z_3$	z_2	z_3
$\pm z_4$	z_4	z_5
$+z_5$	z_4	z_5

Rule 3 - $FA_1 = r_1$ $FA_2 = r_2$ $FA_3 = r_1 . r_2$

Initial = initial (L_1)

final = final (L_2).

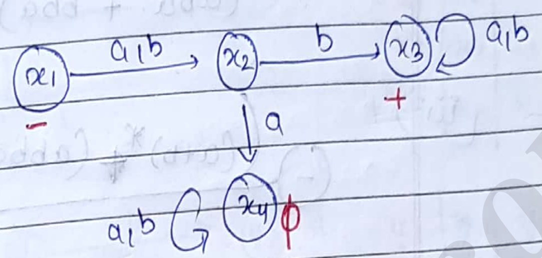
Scanned by CamScanner

Page No.
Date

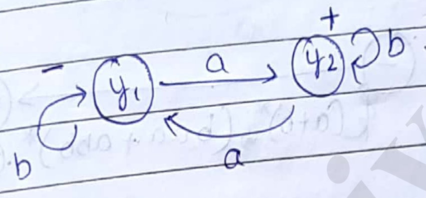
Q. L_1 be the lang. of all words with b as the second letter.
L_2 that have odd no. of a's.

$L_1 + L_2$

$FA_1 \Rightarrow$



$FA_2 \Rightarrow$



	a	b
$-x_1$	x_2	x_3
x_2	x_4	x_3
$+x_3$	x_3	x_3
x_4	x_4	x_4

	a	b
$-y_1$	y_2	y_1
$+y_2$	y_1	y_2

$z_1 = x_1$ or y_1

$(z_1, a) = x_2$ or $y_2 = z_2$

$(z_1, b) = x_3$ or $y_1 = z_3$

$(z_2, a) = x_4$ or $y_1 = z_4$

$(z_2, b) = x_3$ or $y_2 = z_5$

$(z_3, a) = x_3$ or $y_2 = z_5$

$(z_3, b) = x_3$ or $y_1 = z_3$

$(z_4, a) = x_4$ or $y_2 = z_6$

$(z_4, b) = x_4$ or $y_1 = z_4$

$(z_5, a) = x_3$ or $y_1 = z_3$

$(z_5, b) = x_3$ or $y_2 = z_5$

$(z_6, a) = x_4$ or $y_1 = z_4$

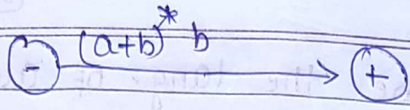
$(z_6, b) = x_4$ or $y_2 = z_6$.

	a	b
$-z_1$	z_2	z_3
$+z_2$	z_4	z_5
$+z_3$	z_5	z_3
z_4	z_6	z_4
$+z_5$	z_3	z_5
$+z_6$	z_4	z_6

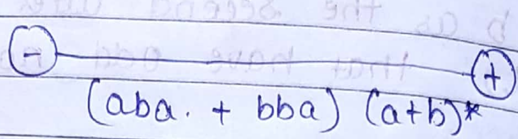
Scanned by CamScanner

EXERCISE

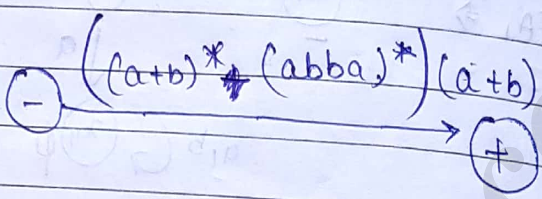
(1) (i)



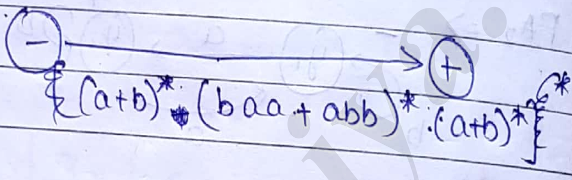
(ii)



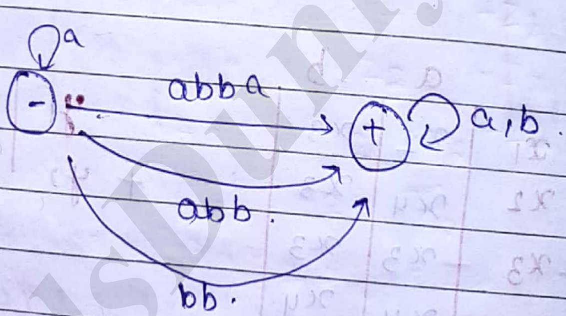
(iii)

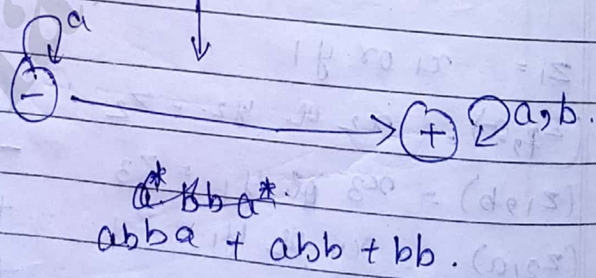


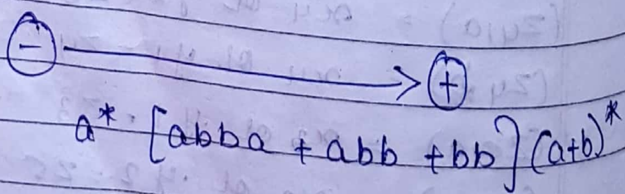
(iv)



(v)



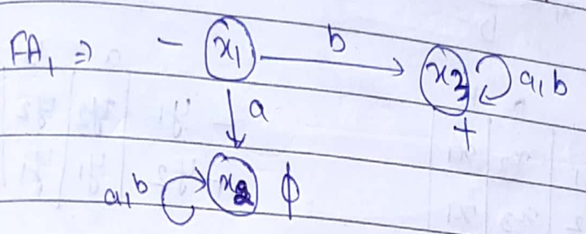




Page No.
Date

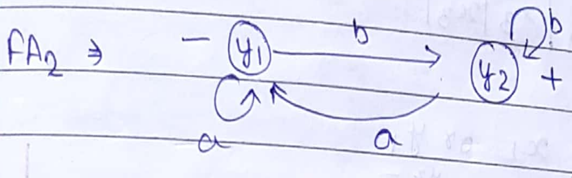
Q: L_1 having all words start with 'b'
L_2 ends with 'b'.

$FA_1 \Rightarrow$



	a	b
$-x_1$	x_2	x_3
x_2	x_2	x_3
$+x_3$	x_3	x_3

$FA_2 \Rightarrow$



	a	b
$-y_1$	y_1	y_2
$+y_2$	y_1	y_2

$z_1 = x_1 \quad (L_1)$

$(z_1, a) = x_2 = z_2$

$(z_1, b) = x_3$ (final) ~~start~~

$z_3 = x_3$ or y_1 (start FA_2)

	a	b
$-z_1$	z_2	z_3
z_2	z_2	z_2
z_3	z_3	z_4
$+z_4$	z_3	z_4

$(z_2, a) = z_2$

$(z_2, b) = z_2$

$(z_3, a) = z_3$

$(z_3, b) = x_3$ or y_2 or y_1 $= z_4$ ✱

$(z_4, a) = z_3$

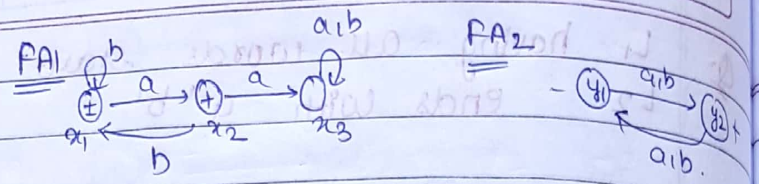
$(z_4, b) = z_4$

Q: L_1 - end with 'b' $\quad (L_1 \cdot L_2)$
L_2 - start with 'b'

Q: L_1 - do not contain substring 'aa'
L_2 - with odd no. of letters. $(L_1 \cdot L_2)$
$L_3 \Rightarrow$ it will not accept Λ only.

Scanned by CamScanner

22/AUG/2



T.T

	a	b
$\pm x_1$	x_2	x_1
$+ x_2$	x_3	x_1
x_3	x_3	x_3

	a	b
$- y_1$	y_2	y_2
$+ y_2$	y_1	y_1

$z_1 = x_1$ or y_1

$(z_1, a) = x_2, y_2 = z_2$ +

$(z_1, b) = x_1, y_2 = z_3$ +

$(z_2, a) = x_3, y_1, y_2 = z_4$ +

$(z_2, b) = x_1, y_1, y_2 = z_3$

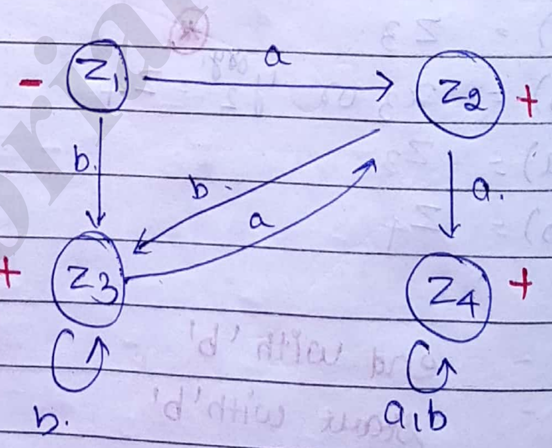
$(z_3, a) = x_2, y_1, y_2 = z_2$

$(z_3, b) = x_1, y_1, y_2 = z_3$

$(z_4, a) = x_3, y_2, y_1 = z_4$

$(z_4, b) = x_3, y_2, y_1 = z_4$

	a	b
$- z_1$	z_2	z_3
$+ z_2$	z_4	z_3
$+ z_3$	z_2	z_3
$+ z_4$	z_4	z_4

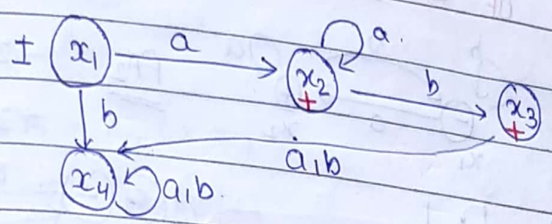


Rule 4 :- if r is regex of $L_1 \rightarrow FA_1$ then r^* is regex of $L_2 = L_1^* \rightarrow FA_2$

Scanned by CamScanner

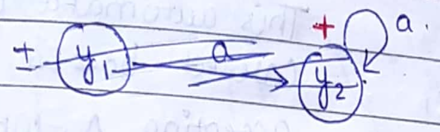
Page No.
Date

Q. $r = a^* + aa^*b$



	a	b
x_1	x_2	x_4
$+x_2$	x_2	x_3
$+x_3$	x_4	x_4
x_4	x_4	x_4

$r^* = [a^* + aa^*b]^*$



$z_1 = x_1$

$(z_1, a) = x_2, x_1 = z_2$

$(z_1, b) = x_4 = z_3$

$(z_2, a) = x_2, x_1 = z_2$

$(z_2, b) = x_3, x_1, x_4 = z_4$

$(z_3, a) = x_4 = z_3$

$(z_3, b) = x_4 = z_3$

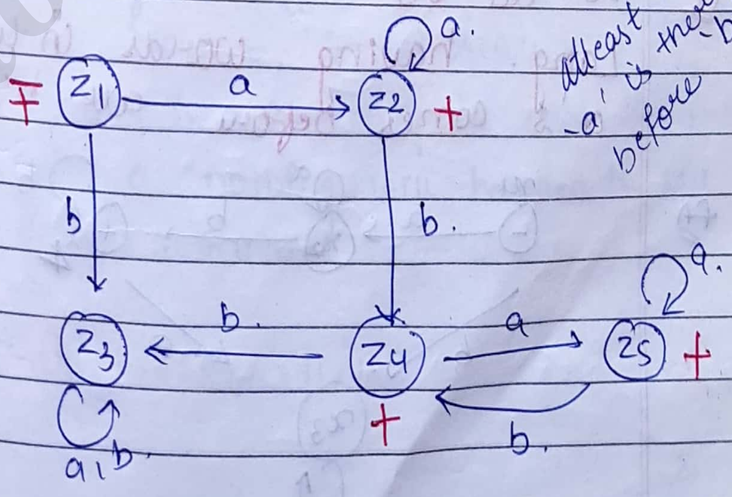
$(z_4, a) = x_2, x_1, x_4 = z_5$

$(z_4, b) = x_4 = z_3$

$(z_5, a) = x_2, x_1, x_4 = z_5$

$(z_5, b) = x_3, x_1, x_4 = z_4$

	a	b
$\pm z_1$	z_2	z_3
$+z_2$	z_2	z_4
z_3	z_3	z_3
$+z_4$	z_5	z_3
$+z_5$	z_5	z_4



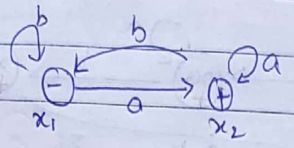
atleast one 'a' is there before 'b'.

Scanned by CamScanner

23/AUG/2019

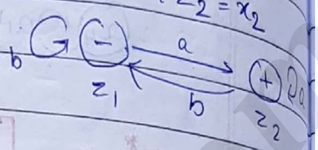
Eg. FA of all words ends in 'a'

FA_1 =>



FA_2 — $-z_1 = x_1$

$+z_2 = x_2$



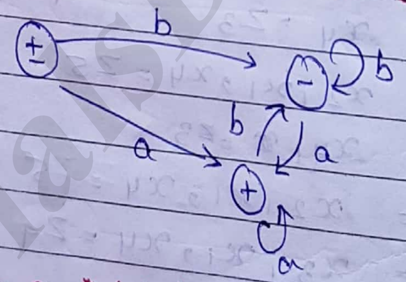
	a	b
x_1	x_2	x_1
x_2	x_2	x_1

z1	z2	z_1
z2	z_2	z_1

This automata is not correct becz it is not accepting Λ, while r^* always accepts Λ.

for making f.A.2 correct we will improve f.A.1

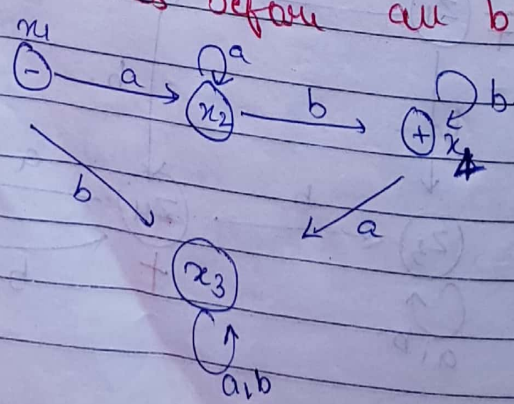
Note- final & Non-enterable * state (FA_1; Now, this will work with Rule 3)



Q. $r = aa^*bb^*$

Lang. having words in which all a's comes before all b's.

FA.



Scanned by CamScanner

Page No. ___
Date ___

$z_1 = x_1$

$z_2 = x_2$

$z_3 = x_3$

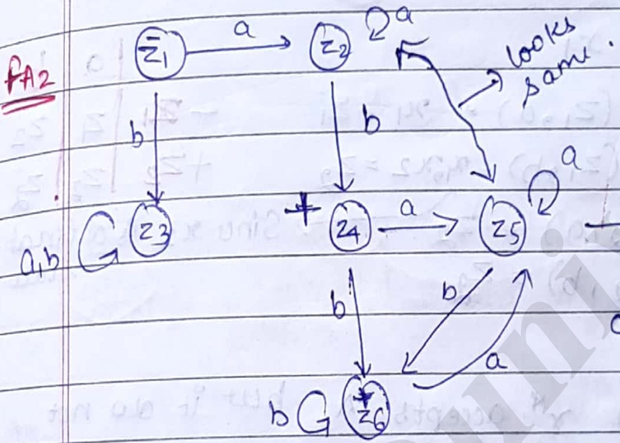
$+ z_4 = x_4$ or x_1

$z_5 = x_2$ or x_3

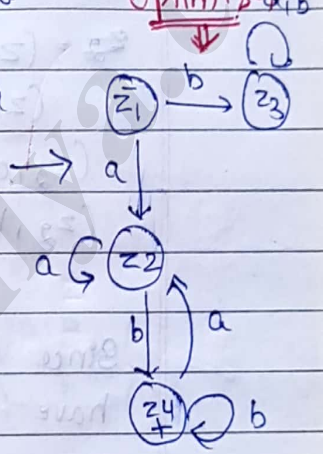
$+ z_6 = x_4$ or x_1 or x_3

	a	b
z_1	z_2	z_3
z_2	z_2	z_4
z_3	z_3	z_3
z_4	z_5	z_6
z_5	z_5	z_6
z_6	z_5	z_6

FA2



Optimised



This automata will accept the words which are starting with 'a' & ending with 'b'.

$$r^* = a(a+b)^* b + \Lambda$$

But this is still not accepting Λ.

So, we will change the state of z_1.

Initially z_1 was only an initial state. Now z_1 is $\pm$ state.

And this is done only to make the F.A, accept Λ nothing will happen to our final automata.

Hw.
Pg 143

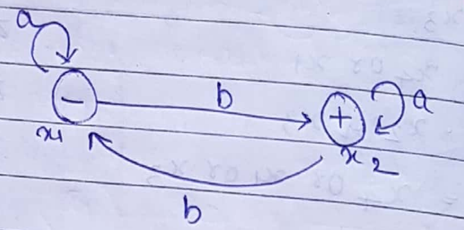
Q. 1, 3, 5, 6, 11, 12

Scanned by CamScanner

26/Aug/2015

Page No.
Date

Q. FA which accepts the lang of all words with odd no. of 'b'.



	a	b
$-x_1$	x_1	x_2
$+x_2$	x_2	x_1

$z_1 = x_1$

$z_2 = (z_1, a) = x_1 = z_1$

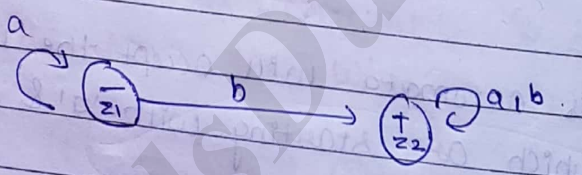
$(z_1, b) = x_2 x_2 = z_2$ → Since x_2 is a final state.

$(z_2, a) = z_2$

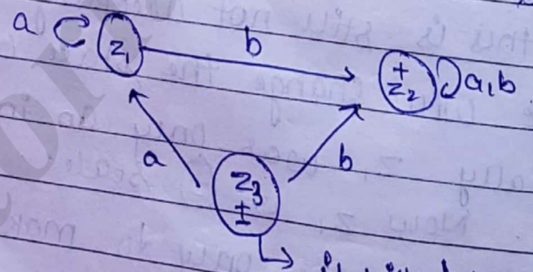
$(z_2, b) = z_2$

	a	b
$-z_1$	z_1	z_2
$+z_2$	z_2	z_2

Since r^* accepts Λ but it do not have Λ.



We will create a new state $\pm$



→ it will behave like same as z_1.

(atleast one 'b' or Λ)

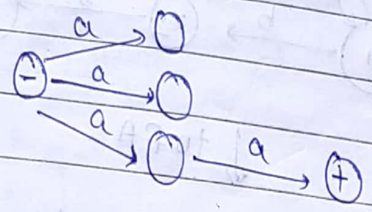
R.Ex = $\Lambda + (a+b)^* \, b \, (a+b)^*$

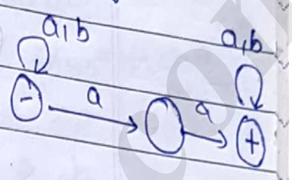
Page No.
Date

Non deterministic f.A.

An NF.A is a T.G with a unique start state with a property that each of its edge labels is a single alphabet letter.

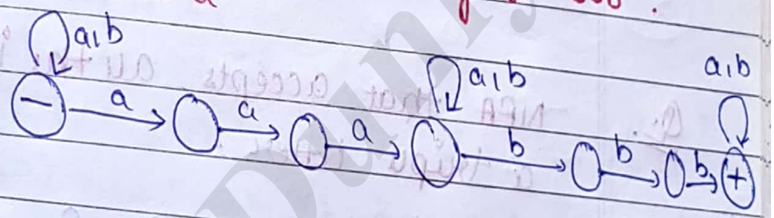
Eg.





Q. Design a machine for the lang for all strings with 'aaa' followed by a 'bbb'.

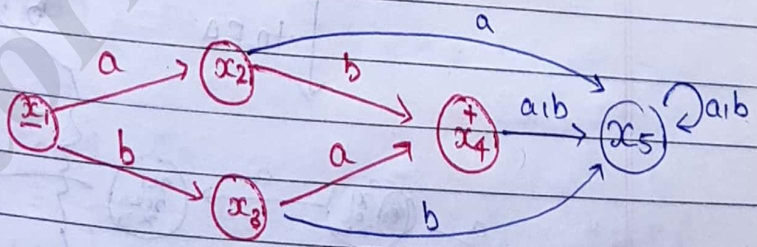
Ans



Theorem - For every NFA there is some FA that accepts exactly the same lang.

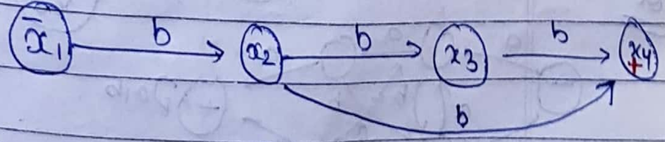
To prove this we can use Part-2&3 of Kleene's theorem.

Q.



NFA (Convert it to FA)

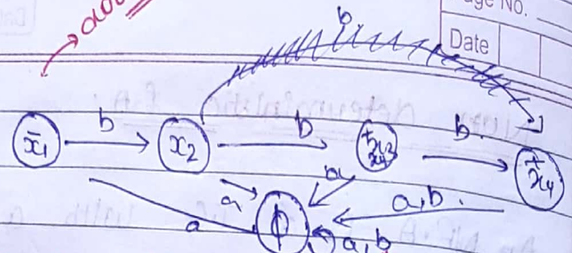
Q. NFA that accepts two words {bb, bbb}



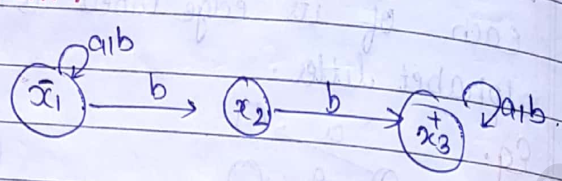
Convert it to FA.

Page No.
Date

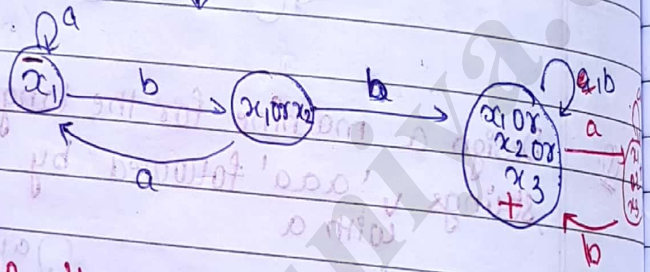
doubt



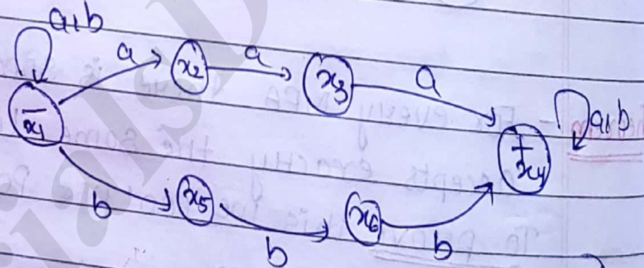
Q. NFA that accepts all i/p with a 'bb' in them.



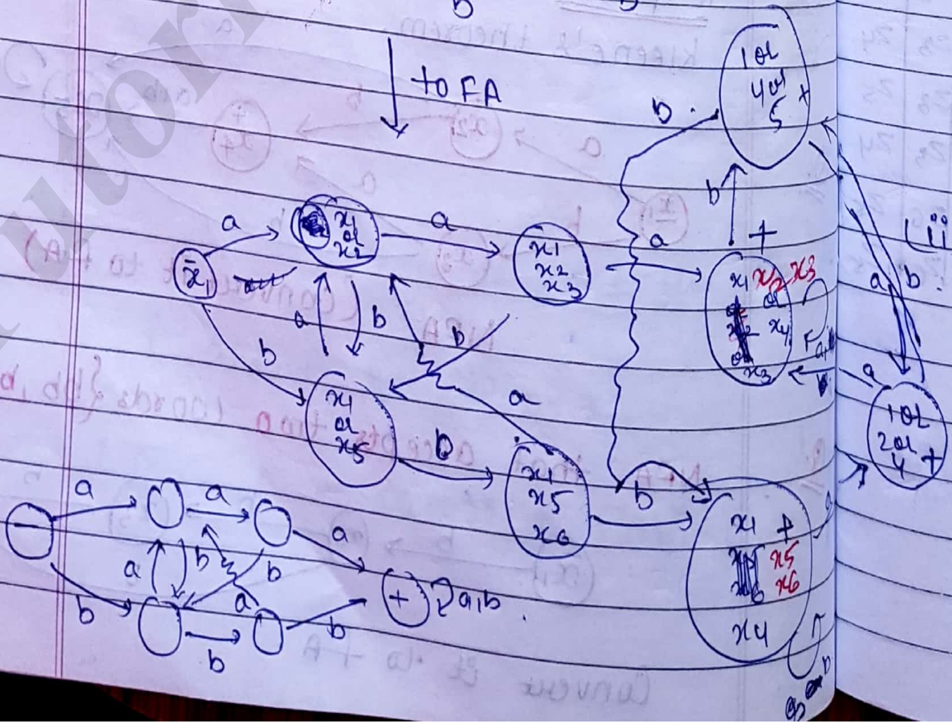
↓ to FA.



Q. NFA that accepts all the i/p with a triple letter.



↓ to FA

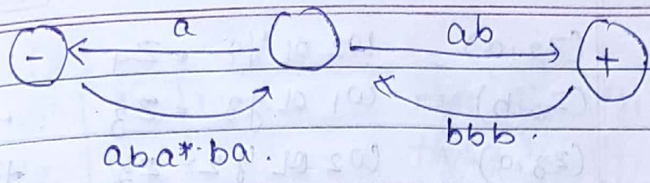


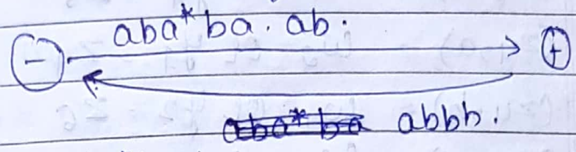
Q.

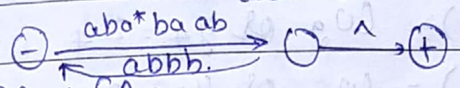
EXERCISE

Page No. ______
Date

(vi)







$(aba^*ba\,ab + abbb)^*$

(8) (i) $FA_1 + FA_2$.

	a	b
$-w_1$	w_2	w_1
$+w_2$	w_2	w_1

	a	b
$-x_1$	x_2	x_1
x_2	x_2	x_3
$+x_3$	x_3	x_3

$z_1 = w_1$ or x_1

$(z_1, a) = w_2$ or $x_2 = z_2$

$(z_1, b) = w_1$ or $x_1 = z_1$

$(z_2, a) = w_2$ or $x_2 = z_2$

$(z_2, b) = w_1$ or $x_3 = z_3$

$(z_3, a) = w_2$ or $x_3 = z_4$

$(z_3, b) = w_1$ or $x_3 = z_3$

$(z_4, a) = w_2$ or $x_3 = z_4$

$(z_4, b) = w_1$ or $x_3 = z_3$

	a	b
$-z_1$	z_2	z_1
$+z_2$	z_2	z_3
$+z_3$	z_4	z_3
$+z_4$	z_4	z_3

(ii) $FA_1 + FA_3$

	a	b
$-w_1$	w_2	w_1
$+w_2$	w_2	w_1

	a	b
$-y_1$	y_2	y_3
y_2	y_4	y_3
y_3	y_2	y_4
$+y_4$	y_4	y_2

$z_1 = w_1$ or y_1

$(z_1, a) = w_2$ or $y_2 = z_2$

$(z_1, b) = w_1$ or $y_3 = z_3$

Scanned by CamScanner

Date

$(z_2, a) = \omega_2$ or $y_4 = z_4$

$(z_2, b) = \omega_1$ or $y_3 = z_3$

$(z_3, a) = \omega_2$ or $y_2 = z_2$

$(z_3, b) = \omega_1$ or $y_4 = z_5$

$(z_4, a) = \omega_2$ or $y_4 = z_4$

$(z_4, b) = \omega_1$ or $y_2 = z_6$

$(z_5, a) = \omega_2$ or $y_4 = z_4$

$(z_5, b) = \omega_1$ or $y_2 = z_6$

$(z_6, a) = \omega_2$ or $y_4 = z_4$

$(z_6, b) = \omega_1$ or $y_3 = z_3$

	a	b
$-z_1$	z_2	[illegible]
$+z_2$	z_4	[illegible]
$+z_3$	z_2	[illegible]
$+z_4$	z_4	[illegible]
$+z_5$	z_4	[illegible]
z_6	z_4	[illegible]

5 (i) FA1 FA2

	a	b
$-\omega_1$	ω_2	ω_1
$+\omega_2$	ω_2	ω_1

	a	b
$-x_1$	x_2	x_1
x_2	x_2	x_3
$+x_3$	x_3	x_3

$z_1 = \omega_1$

$(z_1, a) = \omega_2$ or $x_1 = z_2$

$(z_1, b) = \omega_1 = z_1$

$(z_2, a) = \omega_2$ or x_1 or $x_2 = z_3$

$(z_2, b) = \omega_1$ or $x_1 = z_4$

$(z_3, a) = \omega_2$ or x_1 or $x_2 = z_3$

$(z_3, b) = \omega_1$ or x_1 or $x_3 = z_5$

$(z_4, a) = \omega_2$ or x_1 or $x_2 = z_3$

$(z_4, b) = \omega_1$ or $x_1 = z_4$

$(z_5, a) = \omega_2$ or x_1 or x_2 or $x_3 = z_6$

$(z_5, b) = \omega_1$ or x_1 or $x_3 = z_5$

$(z_6, a) = \omega_2$ or x_1 or x_2 or $x_3 = z_6$

$(z_6, b) = \omega_1$ or x_1 or x_3 ~~or x_1~~ $= z_5$

	a	b
$-z_1$	z_2	z_1
z_2	z_3	z_4
z_3	z_3	z_5
z_4	z_3	z_4
$+z_5$	z_6	z_5
$+z_6$	z_6	z_5

Page No.
Date

(ii) $FA_1 \quad FA_2$

$-w_1$	w_2	w_1
$+w_2$	w_2	w_1

$-y_1$	y_2	y_1
$+y_2$	y_2	y_1

$z_1 = w_1$

$(z_1, a) = w_2$ or $y_1 = z_2$.

$(z_1, b) = w_1 = z_1$

$(z_2, a) = w_2$ or $y_1^{\text{or } y_2} = z_2$

$(z_2, b) = w_1$ or $w_1 = z_1$

	a	b
z_1	z_2	z_1
z_2	z_2	z_1

(6) (i) $(FA_1)^*$

	a	b
$-w_1$	w_2	w_1
$+w_2$	w_2	w_1

$-z_1 = w_1$

$(z_1, a) = w_2 = z_2$

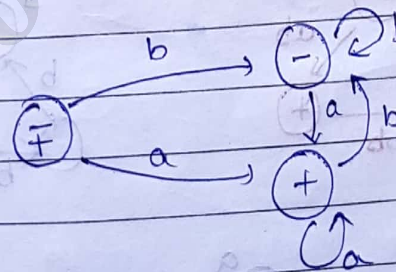
$(z_1, b) = z_1$

$(z_2, a) = z_2$

$(z_2, b) = z_1$

	a	b
$-z_1$	z_2	z_1
$+z_2$	z_2	z_1

This automata do not accept Λ but $(FA_1)^*$ have Λ.



(ii) $(FA_2)^*$

	a	b
$-x_1$	x_2	x_1
x_2	x_2	x_3
$+x_3$	x_3	x_3

Scanned by CamScanner

Page No.
Date

$z_1 = x_1$

$(z_1, a) = x_2 = z_2$

$(z_1, b) = x_1 = z_1$

$(z_2, a) = x_2 = z_2$

$(z_2, b) = x_3 \cancel{x_3}\ x_1 = z_3$

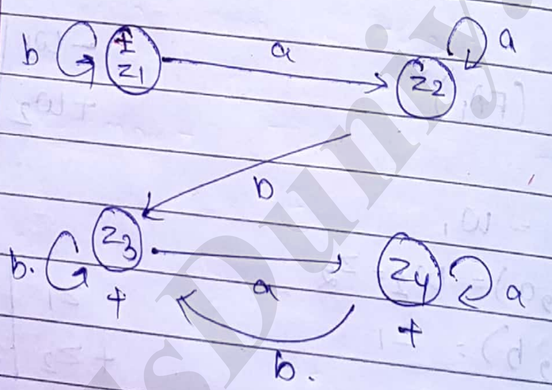
$(z_3, a) = \cancel{z_3}\ x_3 \text{ or } x_2 = z_4$

$(z_3, b) = z_3$

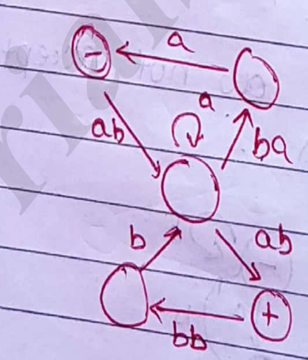
$(z_4, a) = x_3 \text{ or } x_2 = z_4$

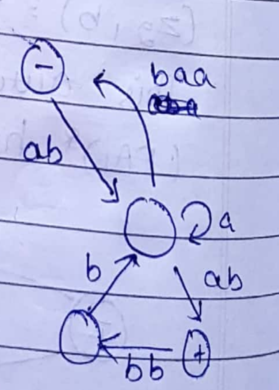
$(z_4, b) = x_3 \text{ or } x_1 = z_3$

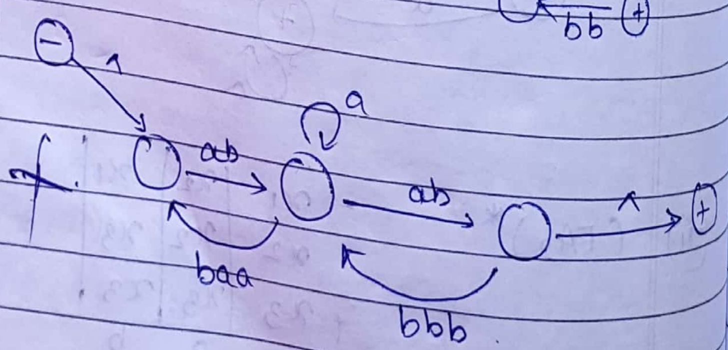
	a	b
$-z_1$	z_2	z_1
z_2	z_2	z_3
$+z_3$	z_4	z_3
$+z_4$	z_4	z_3

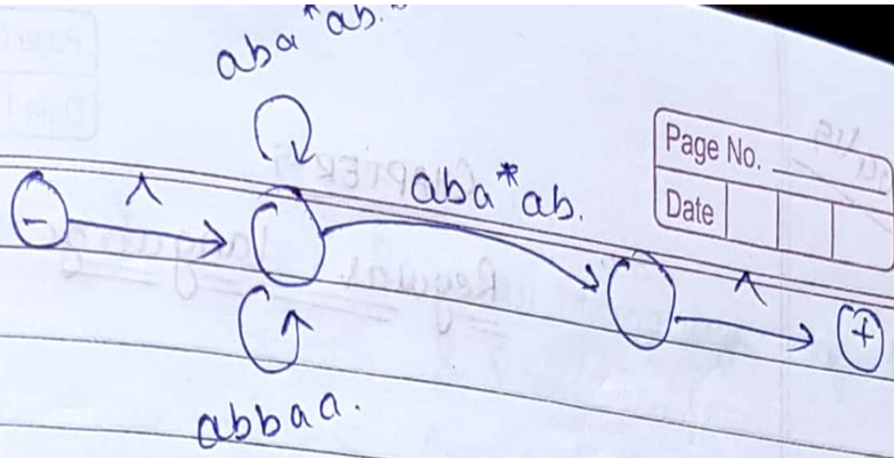


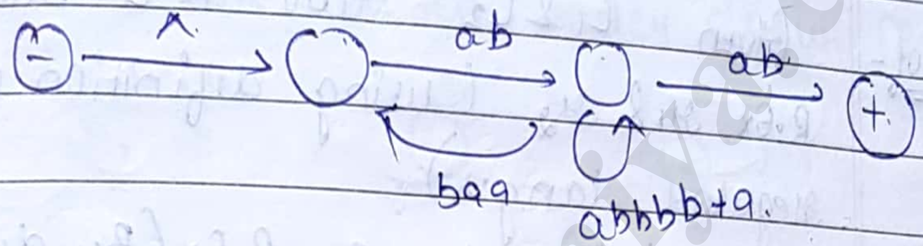
1 (vi)

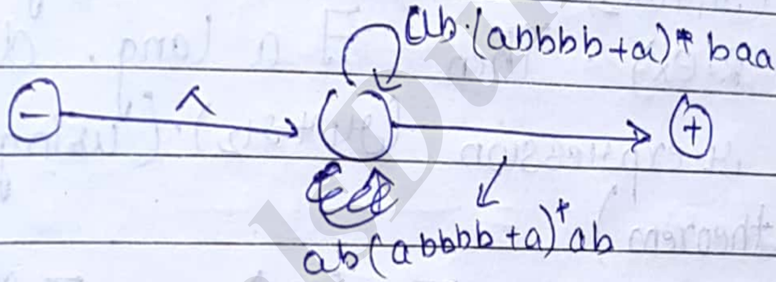


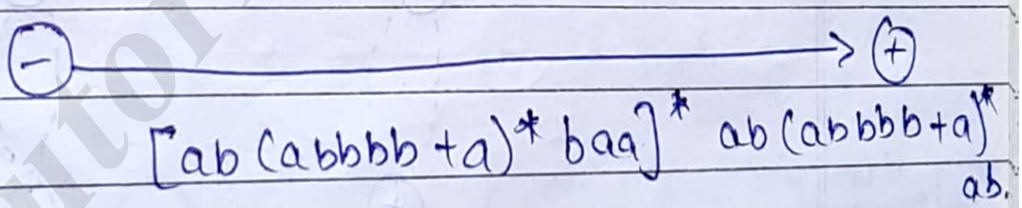












1 AUG 19

~~Convert NFA to FA.~~

28/Aug/19

Page No.
Date

CHAPTER-9

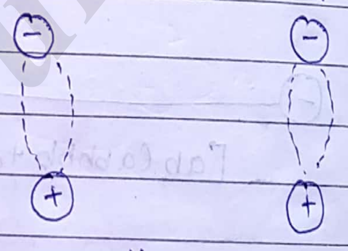
Regular language

→ Properties of R.L

* R.L are ~~the~~ closed under Union.

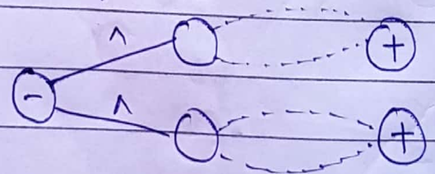
Proof- Given L_1 & L_2 two R.L having R.Ex r_1 & r_2 (using definition of regular lang's)
$(r_1 + r_2)$ is also a R.Ex (By definition R.Ex). Then ∃ a lang. defined over r.expression $(r_1 + r_2)$. (using Kleene's theorem)

Example - T.G1 (R.L1) T.G2 (R.L2)



⇓

(R.L Final)



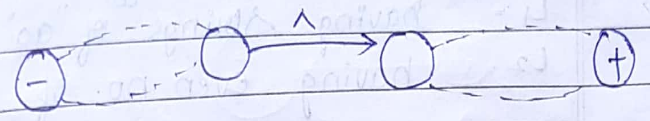
* R.L are closed under concatenation

Proof- Given L_1 & L_2 having R.Ex r_1 & r_2
$(r_1 \cdot r_2)$ is also a R.Ex (By defi of R.Ex). Then ∃ a lang. defined over R.Ex $(r_1 \cdot r_2)$ (using Kleene's theorem).

Scanned by CamScanner

Page No.
Date

Example Regular language of RE1 & RE2

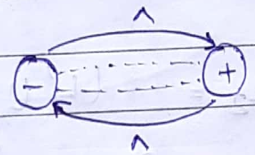


* R.L are closed under closure (Explaination above)

Example

$r_1 \Rightarrow$

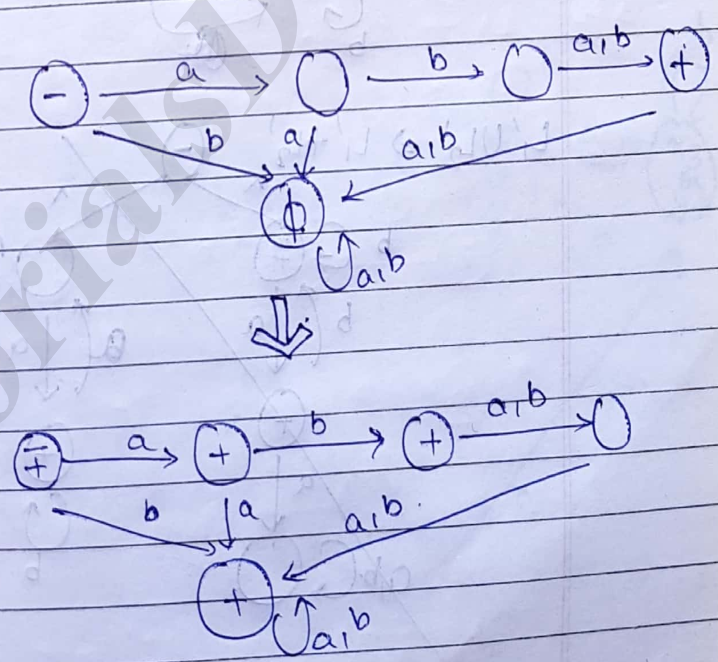
$r_1^* \Rightarrow$



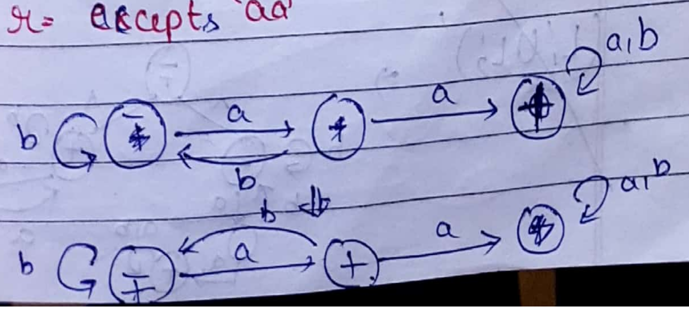
* R.L are closed under complement.

Example

$r = aba + abb$



r = accepts 'aa'



Scanned by CamScanner

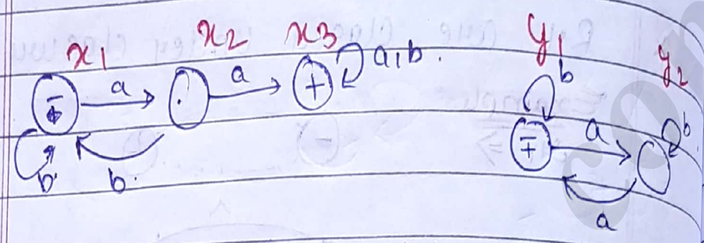
Page No.
Date

* R.L are closed under intersection.

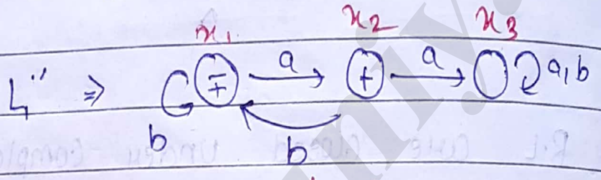
Example -

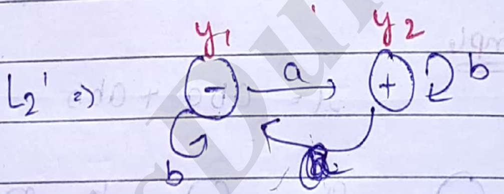
L_1 - having strings with "aa"

L_2 - having even no. of a's.

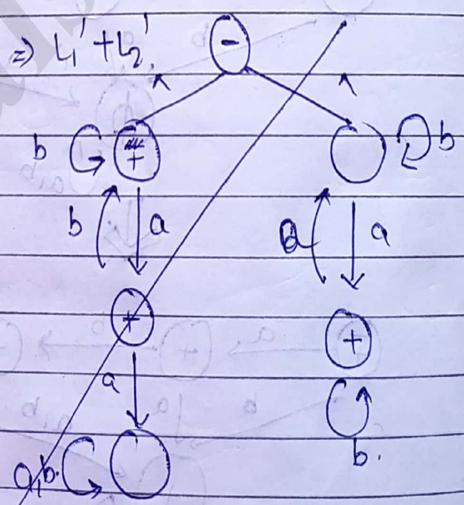


formula - $L_1 \cap L_2 = (L_1' \cup L_2')'$

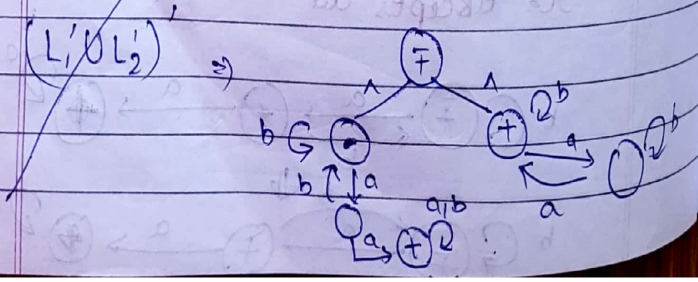




$L_1' \cup L_2' \Rightarrow L_1' + L_2'$



$(L_1' \cup L_2')' \Rightarrow$



Page No.
Date

	a	b
F x_1	x_2	x_1
+ x_2	x_3	x_1
x_3	x_3	x_3

	a	b
− y_1	y_2	y_1
+ y_2	y_1	y_2

$z_1 = x_1$ or y_1

$(z_1, a) = x_2$ or $y_2 = z_2$

$(z_1, b) = x_1$ or $y_1 = z_1$

$(z_2, a) = x_3$ or $y_1 = z_3$

$(z_2, b) = x_1$ or $y_2 = z_4$

$(z_3, a) = x_3$ or $y_2 = z_5$

$(z_3, b) = x_3$ or $y_1 = z_3$

$(z_4, a) = x_2$ or $y_1 = z_6$

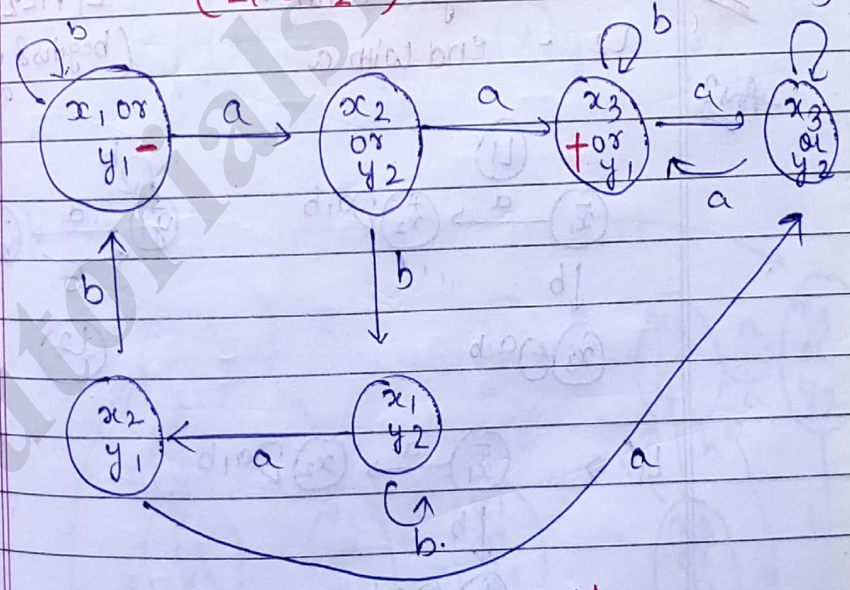
$(z_4, b) = x_1$ or $y_2 = z_4$

$(z_5, a) = x_3$ or

$(z_5, b) = x_3$ or

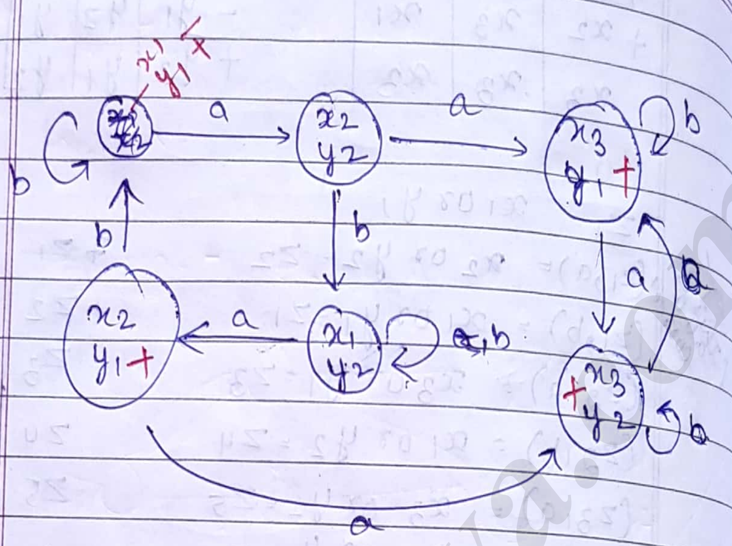
	a	b
F z_1	z_2	z_1
+ z_2	z_3	
z_3		
z_4		
z_5		

$(L_1' \cup L_2')'$



Ans

$L_1 \cup L_2 \Rightarrow$



Q1 ⟹ L_1 - accepts a 'aa'
12 states
L_2 - even even lang.

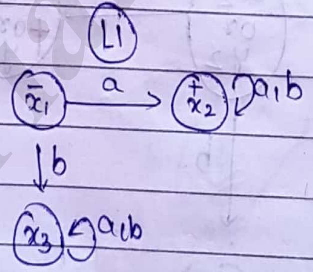
Make Union machine $L_1 + L_2$

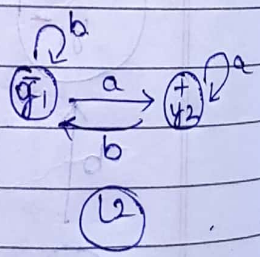
$L_1 \Rightarrow$

Q2 L_1 - begin with a
L_2 - end with a

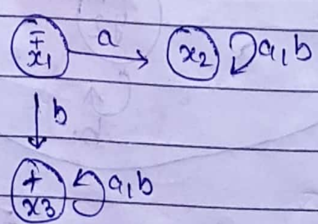
$L_1 \cap L_2$ (begins & end with a)

Ans2

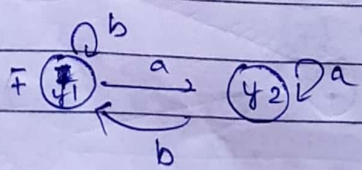




$L_1' \Rightarrow$

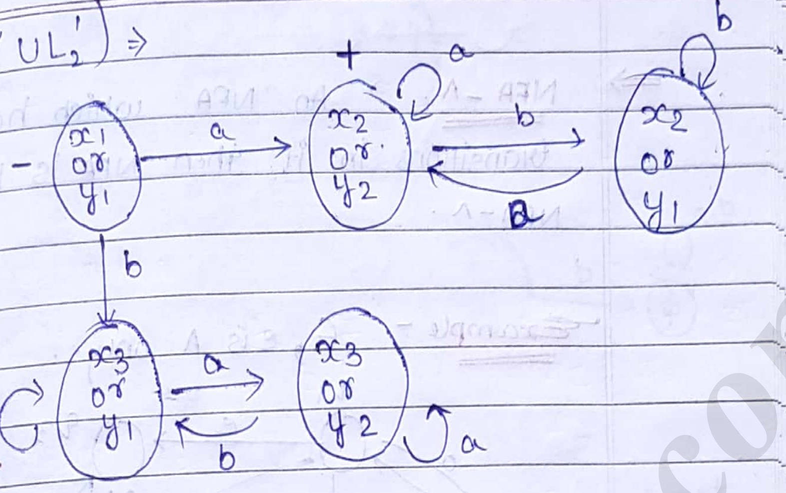


$L_2' \Rightarrow$



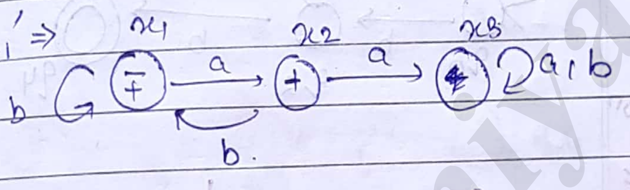
Page No.
Date

$(L_1' \cup L_2')' \Rightarrow$

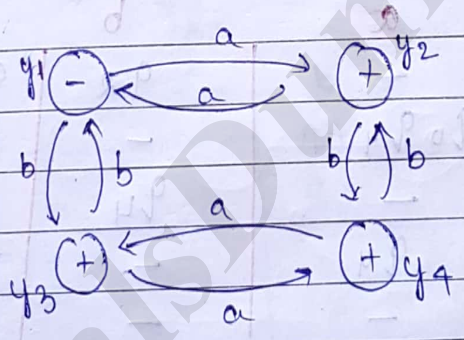


Ans.

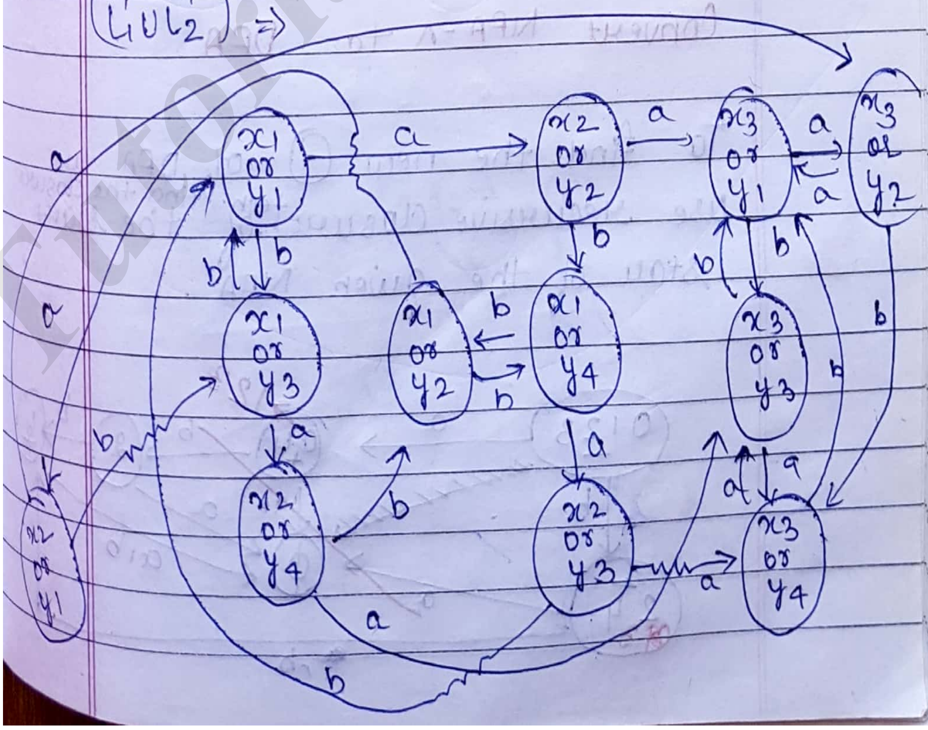
$L_1' \Rightarrow$



$L_2' \Rightarrow$



$(L_1' \cup L_2')' \Rightarrow$

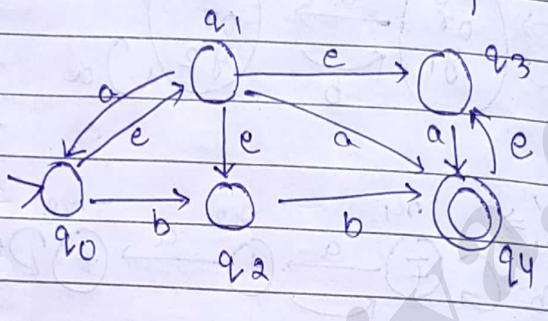


29/AUG/2019

Page No.
Date

⟹ NFA – Λ. → An NFA which has Λ transitions in it then NFA is known as NFA – Λ.

Example – Є, e is Λ only.



Transition table

	a	b	e
-q0	–	q2	q1
q1	q4	–	
q2			
q3			
q4			

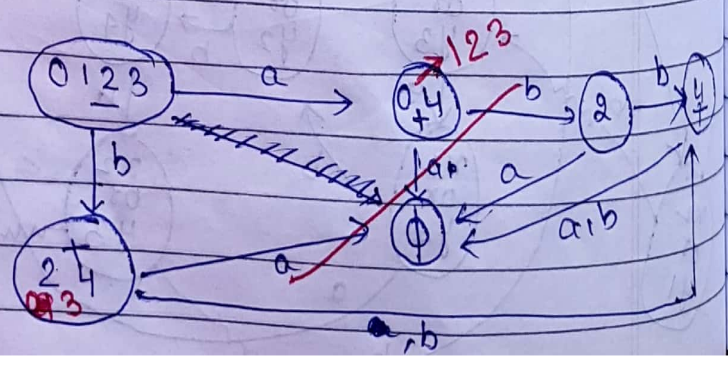
SAKSHI

Transition Table

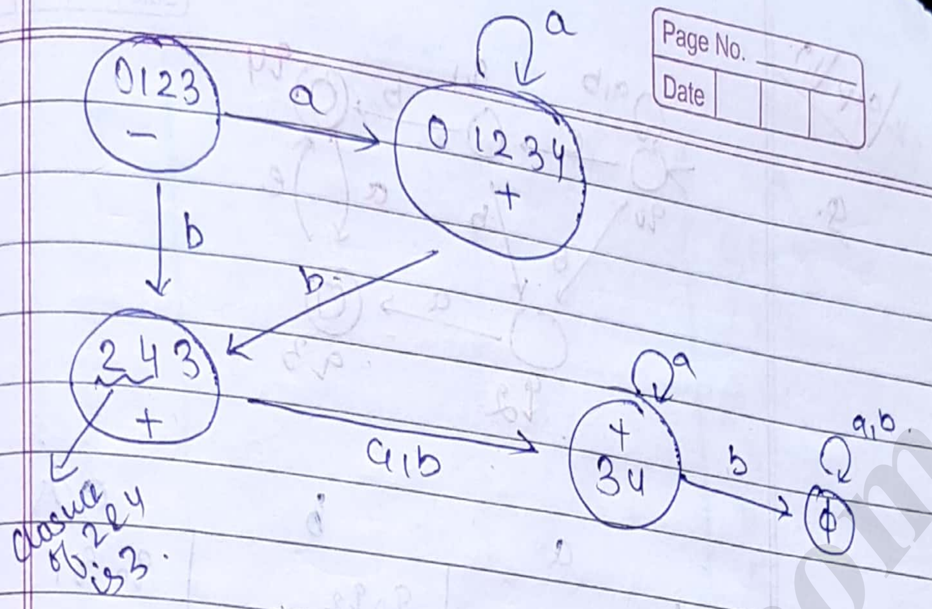
	a	b	e
-q_0	–	q_2	$q_0 q_1$
q_1	$q_0 q_4$	–	$q_1 q_2 q_3$
q_2	–	q_4	q_2
q_3	q_4	–	q_3
q_4	–	–	$q_3 q_4$

Convert NFA – Λ to DFA.

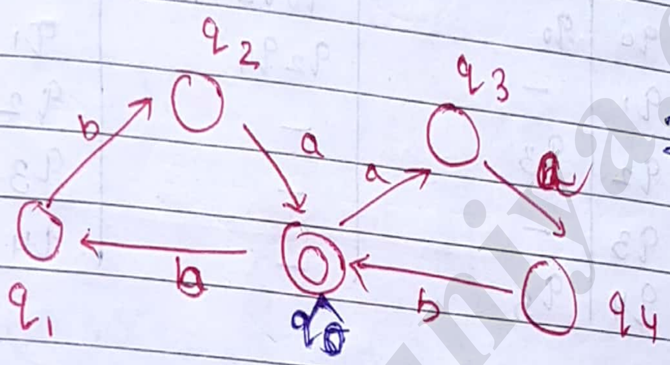
To find the new Θ of DFA we use recursive closure tech. find the closure of the start state of the given NFA.



Page No.
Date



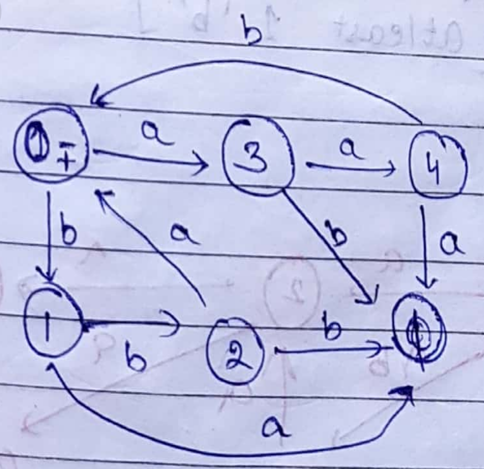
Q.



It is not NFA-Λ

Transition table

	a	b	e
q_0	q_3	q_1	q_0
q_1	—	q_2	q_1
q_2	q_0	—	q_2
q_3	q_4	—	q_3
q_4	—	q_0	q_4

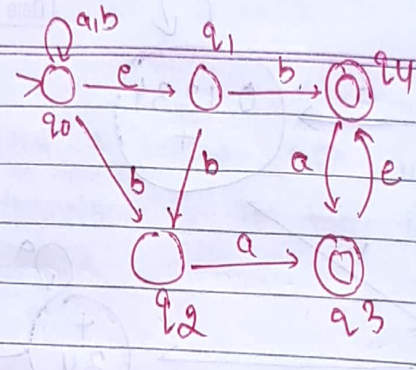


Scanned by CamScanner

Page No.
Date

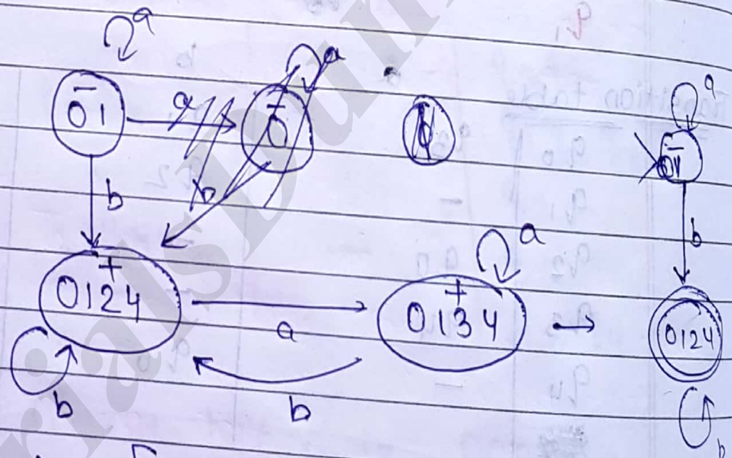
2/Sep/19

Q.



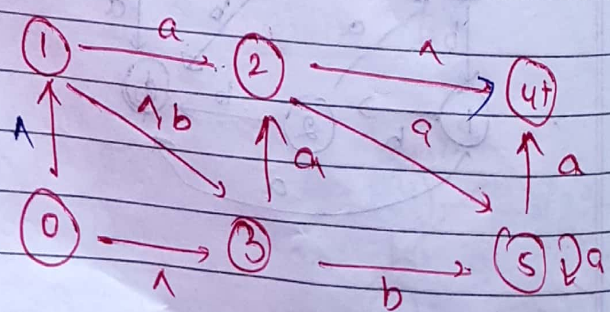
	a	b	e/Λ
q_0	q_0	$q_0 q_2$	$q_0 q_1$
q_1	—	$q_2 q_4$	q_1
q_2	q_3	—	q_2
q_3	—	—	$q_3 q_4$
q_4	q_3	—	q_4

SAKSHI

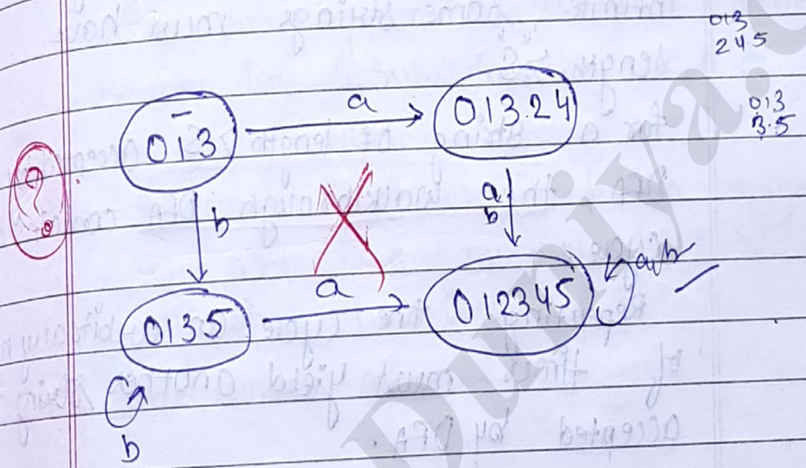


L = [atleast 1 'b']

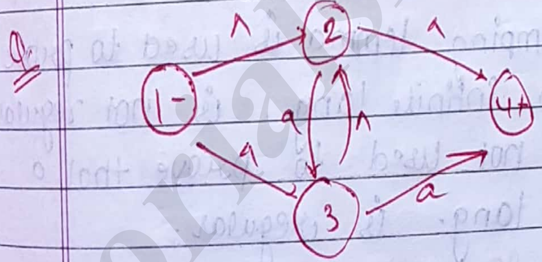
Q



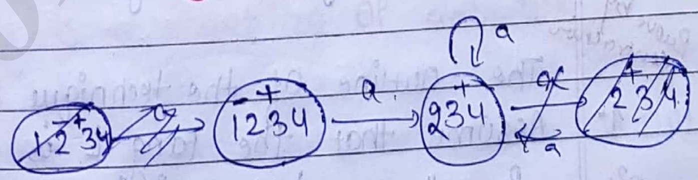
	a	b	∧
→0	–	–	0, 1, 3
1	2	3	1, 3
2	5	–	2, 4
3	2	5	3
+4	–	–	4
5	5, 4	–	5



Q.



	a	∧
1	–	1 2 3
2	3	2 4
3	4	2 3
+4	–	4



Lang. a^*

2|sep|19

CH-10

Pumping Theorem

⇒ **Pumping lemma** - If an infinite lang is regular it can be
(Purpose)
defined by a DFA. DFA has finite no of states say 'S' since the lang is infinite, some strings must have length > S.
for a string of length > S accepted by DFA, the walkthrough DFA contains a cycle.
Repeating the cycle an arbitrary no. of times must yield another string accepted by DFA.

The pumping lemma is used to prove that an infinite lang. is not regular
it is not used to prove that a given lang. is regular.

Prove by contradiction

The outline of the technique is as follows-
1. Assume that the lang L is regular.
2. By pigeon hole principle any sufficiently long string in 'L' must repeat some in DFA, Thus the walk contains a cycle.
3. Show that repeating the cycle some no. of times (pumping the cycle) yields a string that is not in 'L'.

Date

Conclude the assumption that L is regular is wrong & hence 'L' is not regular.

Eg. Show that $L = \{a^n b^n\}$ is not regular

Assume that the lang 'L' given by $L = \{a^n b^n\}$ be regular.

Choose 'S' to be a string $a^i b^i$.

By pumping lemma string 'S' can be broken into 3 substrings say x, y, z. where $y \neq$ NULL.

As xyz is a word in 'L' so xyyz will also be a word of lang 'L'.

We shall now choose 'y'.

Three cases arise :-

4/sep/19

Case 1 :- ~~b~~y is a substring of only a's

In this case xyyz will have more a's than b's & resulting word will not belong to 'L'.

Case 2 :- y is a substring of only b's

In this case xyyz is a word that has more b's than a's and hence is not word in 'L'.

Case 3 :- y is a substring 'ab'

In this case xyyz contains two instances of substring 'ab', which is not a word of 'L'.

Scanned by CamScanner

Page No.
Date

So, for no possible choice of y, xy^iz remains to be in 'L'. This implies that our assumption that 'L' is a reg. lang is wrong. This means the lang $L = \{a^n b^n\}$ is a non-regular lang.

Ques Show that the lang EQUAL a's & b's is not regular.

To Prove EQUAL is non-regular
We know,

$$\boxed{a^n b^n = a^* b^* \cap \text{EQUAL}} \quad \text{—①}$$

We also know from above proof that $a^n b^n$ is non-regular. Also regular langs are closed under intersection.

If EQUAL was a regular lang then from ① we infer that $a^n b^n$ should have been a regular lang which is not true. We also know that $a^* b^*$ is a regular lang. This implies that EQUAL must be non-regular for ① to be true.
So, we conclude that lang EQUAL is non-regular.

Ques Prove lang. PALINDROME is non-regular.

definition → Pumping lemma -
Let 'L' be a regular lang. there is an integer $n \geq 1$ such that any string

Scanned by CamScanner

Page No.
Date 1/9

'w' ∈ L with (length of $w \geq n$) or length $|w| \geq n$ can be rewritten as :-

$|w| \geq n$

$w = xyz$ such that y is $\neq e$. → y do not have Λ.

and length $(xy) \leq n$ such that xy^iz ∈ L for each 'i' ≥ 0 (y can be substring of a's or b's.)

x ———— x

Say pumping length is 'm' let the word in lang PALINDROME to be considered be:- $a^m b a^m$.

Acc. to pumping lemma ∃ a decomposition of this word as xyz where length$(xy) \leq m$

The only choice of 'y' is y to be a substring of a's as 'y' cannot be null (By definition of pumping lemma).

As 'y' is a substring of a's by pumping 'y' the words generated $xyyz$ does not belong to 'L' as it is not a palindrome.

So our assumption that lang PALINDROME is wrong. Hence we prove that lang PALINDROME is NON-Regular.

y^0 is not same as $y = \Lambda$ → y is empty.

↓

y is a non-empty string but we are not including it in xy^iz.

Eg $y = \{aa\}$

$y^0 = \{aa\}^0$

Scanned by CamScanner

4/Sep/19.

Page No.
Date

CHAPTER-12

languages

Context free Grammar (CFG).

L.H.S → R.H.S (Non-terminal /Terminal)
(Non-Terminal)

→ Compilation steps

① Lexical Analysis - Source code to tokens
② Syntax Analysis - Parsing - tokens → parse trees.

Context free grammar is built on set of productions which consists of N-T & T.
N.T → uppercase T → lowercase.

To start with the generation of words that are valid, there is 'S' i.e., '*'.

A CFG is a collection of following-

(i) An alphabet Σ of letters called T from which we are going to make strings that will be the words of a language.

(ii) A set of symbols called n-T one of which is a symbol S implying the *.

(iii) A finite set of productions of the form one NT → finite string(s) of T and/or n-T.

SAKSHI

Page No. ____
Date

Eg. of CFG

For a^* $S \to aS/\Lambda$ or $S \to Sa/\Lambda$

This is ~~Brak~~ Backus Naur form.

$S \to SS/a \equiv S \to Sa \mid S$ all possible a's

$S \to SS|a|\Lambda \equiv a^*$

$(a+b)(a+b)^*$
$S \to a$
$S \to b$
$S \to Sa$
$S \to Sb$

$(a+b)^*$
$S \to Sa$
$S \to Sb$
$S \to \Lambda$

$(a+b)^*$
$S \to X$ $Y \to bY$
$S \to Y$
$~~X~~ \to \Lambda$ $Y \to a$
$Y \to aY$ $Y \to b$

$a^n b^n$
$S \to aXb$
$X \to aXb|\Lambda$
or $S \to ab|aSb$

Palindrome
$S \to aSa$
$S \to bSb$
$S \to \Lambda$
$S \to a$
$S \to b$

Palindrome (even length)
$S \to aSa$
$S \to bSb$
$S \to \Lambda$

$L = \{a^n b a^n\}$
$S \to aSa$
$S \to aba$

Palindrome (odd length)
$S \to aSa$
$S \to bSb$
$S \to a$
$S \to b$

$(a+b)^* aa (a+b)^*$
$S \to XaaX$
$X \to aX|bX|X$

* if CFG is written as T, n-T, arrow, OR

$X \to aX \mid bX \mid \Lambda$

Such a form is known as Backus-Naur form (BNF).

Page No.
Date

(1) Write a CFG for lang having words with 'a' 'bbb' in them.
(2) $a(a+b)^*$
(3) Words having minimum 2 a's.

(1) R.Ex → $(a+b)^* bbb (a+b)^*$

$S \to XbX$

$X \to aX | bX | \Lambda$

(2) REx – $a(a+b)^*$

$S \to aX$

$X \to aX | bX | \Lambda$

(3) REx. $(a+b)^* a (a+b)^* a (a+b)^*$

$S \to XaXaX$

$X \to aX | bX | \Lambda$

* CFG for even even lang.

R.Ex ⇒ $[aa + bb + (ab+ba)(aa+bb)^*(ab+ba)]^*$

$S \to \Lambda$

$B \to aa | bb$

$U \to ab | ba$

$S \to USU$

$S \to SS$

$S \to BS | \cancel{SB}$

SS SS
SB SB
ΛΛ aaΛ
ab ab aa

⟹ [which do not have 'ab' in them.

Scanned by CamScanner

Page No.
Date

R.Ex - ~~(aa + bb + ba)~~* $b^* a^*$

~~S → A~~

B → Bb | ∧

C → Ca | ∧

S → BC

⇒ Words which have different first & last letter.

R.Ex - $a(a+b)^* b + b(a+b)^* a$

S → aXb | bXa

X → aX | bX | ∧

⇒ S → SS

S → XXX

~~X~~ → aX | Xa | b

Only b's
Only a's
No more than three b's.

{ bbb, abbb, babb, ~~aabb~~, bbab, bbba, aabbb, abab b, ab b ab, babab, bbbaa, abbba, ... }

Language [words having b's as multiple of 3.

9/Sep/19

Page No.
Date

Pumping lemma

Non - regular lang:

language of prime

Ques. Lang. Lpr consisting of all strings of 1's whose length is a prime no. Show that this lang is non-reg. lang.

Ans. Let language Lpr be regular. Let 1^p be a word of sufficient length such that it can be broken into three substrings xyz such that $y \neq \Lambda$. Here consider some prime 'p' $\geq n+2$. [there must be such a p since there are an infinite no. of ~~times~~ primes.]

~~By pumping lemma~~

Let $w = 1^p$ —— (1)

By pumping lemma we can break $w = xyz$ such that $y \neq e$ or ϵ and $|xy| \leq n$.

Let $|y| = m$ then, $|xz| = p - m$.

Now consider the string $xy^{p-m}z$, which must be in Lpr by the pumping lemma if Lpr is really regular.

However, $|xy^{p-m}z| = |xz| + (p-m)|y|$

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

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

$|xy^{p-m}z|$ has two factors (p-m) and (m+1) we must check that

Scanned by CamScanner

Page No.
Date

neither of these factors are 1, since then $(p-m)(m+1)$ might be a prime after all. But, $m+1 > 1$, since $y \neq \Lambda$ a ∈ tells us $m \geq 1$. Also, $p-m > 1$, since $p \geq n+2$ was choosen & $m \leq n$ since $m = |y| \leq |xy| \leq n$

Thus, $p-m \geq 2$ $[p-m > 1]$

or $p \geq n+2$

$|xy| \leq n$ & $|y| = m$ ∴ $m \leq n$

$p-m \geq \underbrace{n-m}_{\geq 0}+2$

Hence $|xy^{p-m}z|$ is not a prime no.
∴ lang Lpr is non-regular.

Context free language Grammar

Q. Words having atmost two a's.

$S \rightarrow B \mid BAB \mid BABAB$
$B \rightarrow \lambda \mid Bb$
$A \rightarrow a$

Q. Even no. of a's
R.Ex = $(b^* a b^* a b^*)^*$

$S \rightarrow SS \mid \Lambda$
$S \rightarrow BaBaB$
$B \rightarrow Bb \mid \Lambda$

Scanned by CamScanner

Page No. Date

Q. $S \to aSa \mid bSa \mid \Lambda$

$L = \{ w a^{length(w)} \}$

This lang is known as TRAILING COUNT.

Q.1) $S \to aB$
$S \to bA$
$A \to a$
$A \to aS$
$A \to bAA$
$B \to b$
$B \to bS$
$B \to aBB$

Q. even length string
$L = (aa + ab + ba + bb)^*$

$S \to AS \mid \Lambda$
$A \to aa \mid ab \mid ba \mid bb$

Q.2 $S \to aS \mid Sb \mid a \mid b$

a, b, ~~aa, ab, bb,~~ ~~aaa, aab, abb,~~ ...

~~all the a's are followed by b's if ever 'b' occurs.~~

a, b, aa, ab, bb, aaa, aab, abb, ~~aab~~ aa~~ab~~, ~~aaab~~, aabb,

Q.3 CFG of equal lang.
n no. of a's & n no. of b's.
~~equal lang~~ $\to a^n b^n$

$\cancel{a^i b^i} + b^i a^i$

2 $L = \{ a, b, aa, ab, bb, aaa, aab, abb, aaaa, aaab, aabb, \ldots \}$

there is not substring 'ba' in a string.

Scanned by CamScanner

Page No.
Date

all the a's always occur before all the b's or the word will contains only a's or only b's.

(1) L = { ab, ba, aabb, abab, abba, baab, baba, ~~abab~~, bbaa, aab.bab, aabbba, ~~bbaab~~, bbaaba ... }

equal no. of a's & b's.

(3) Q.1 CFG
Solve Exercise ques.

11/sep/19 .

Q. S → aSbS | bSaS | Λ ; ~~equal language~~ equal lang

* Every CFG has a CFL but it is not necessary every CFL has a CFG.

* CFL is a lang. for which Context free Grammar can be written.

* Difference b/w CFG & R.Ex.
R.Ex only contains non-terminals whereas a CFG can be combination of T & n-T.

⇒ Chomsky Normal form (CNF)
if only have rules as n-T → n-T₁ n-T₂
n-T → T
We need to remove all the rules having Λ production (something → Λ) where Λ production is unnecessary.

Scanned by CamScanner

Page No.
Date

Eg. $S \rightarrow aX$
$X \rightarrow \Lambda$ // this is not req.

Eg. $S \rightarrow aSa$
$S \rightarrow bSb$
$S \rightarrow \Lambda$
$\Rightarrow$
$S \rightarrow aSa$
$S \rightarrow bSa$
$S \rightarrow aa$
$S \rightarrow bb$

SAKSHI

Eg. $S \rightarrow a \mid Xb \mid aYa$
$X \rightarrow Y \mid \Lambda$
$Y \rightarrow b \mid X$

$S \rightarrow aYa \rightarrow aXa \rightarrow a\Lambda a$

$S \rightarrow a \mid b \mid Xb \mid aYa \mid aa$
$X \rightarrow Y$
$Y \rightarrow b \mid X$

* $S \rightarrow \Lambda$
$N \rightarrow S$ // N is also nullable

* **Unit production** :- Rules in which n-T is going to another n-T.

Theorem if L is a CFL, generated by a CFG that includes Λ productions then there is a different CFG that has no Λ-productions that generates either the whole lang L (if L does not include the word Λ) or else generates the lang. of all words in L that are not Λ.

Page No.
Date

Chomsky Normal form
$NT \to NT_1 NT_2$
$NT \to T$

Q. $S \to aSa \mid bSb \mid a \mid b \mid aa \mid bb$

$S \to a$
$S \to b$

$S \to S_1 A$
$S_1 \to AS$ } aSa

aa { $S \to AA$, $A \to a$

$S \to S_2 B$
$S_2 \to BS$ } bSb

bb { $S \to BB$, $B \to b$

Q. Convert CFG to CNF

$S \to bA \mid aB$
$A \to bAA \mid aS \mid a$.
$B \to aBB \mid bS \mid b$.

OR is not allowed in CNF.

a — $X \to a$
b — $Y \to b$
aS — $A \to XS$
bS — $B \to YS$

$S \to YA \mid XB$
$A \to YS_1$
$S_1 \to AA$
$B \to XS_2$

$S \to XB$
$A \to a$
$B \to b$
$S_2 \to BB$.

Q.1 Odd length Palindrome to CNF.
$S \to aaaS \mid aaaa$
~~S→A~~
to CNF

Q.2 $\{L = \{a^{4n} ; n \geq 1\}$

Ans. after 3 pages

* $S \to ABa$
$A \to aab$
$B \to aAc$

$S \to S_1 S_2$
$S_1 \to AB$
$S_2 \to a$
$A \to S_2 S_3$
$S_3 \to S_2 S_4$
$S_4 \to b$
$B \to S_2 S_5$
$S_5 \to c$

Scanned by CamScanner

Equal Lang - equal no. of a & b's.

Page No.
Date

→ CFL Properties

* $L_1 = a^n b^n$ $L_2 =$ even palindrome

$S_1 \rightarrow aS_1b \mid ab$ $S_2 \rightarrow aS_2a \mid bS_2b \mid aa \mid bb$

CFG for $L_1 + L_2$

$S \rightarrow S_1 \mid S_2$

$S_1 \rightarrow aS_1b \mid ab$

$S_2 \rightarrow aS_2a \mid bS_2b \mid aa \mid bb$

CFLs are closed under Union.

* $L_1 = aa^*$ $L_2 = a^n b^n$

$S_1 \rightarrow aS_1 \mid a$ $S_2 = aS_2b \mid ab$

$L_1 \cdot L_2$ $S \rightarrow S_1 S_2$

CFLs are closed under Concatenation.

* $L = (\text{Palindrome})^*$ ↳ simple

$S \rightarrow S_1S \mid \wedge$

$S_1 \rightarrow aS_1a \mid bS_1b \mid aa \mid bb \mid \wedge$

CFLs are closed under Kleene's closure

* $L = a^n b^n a^n$ // non Context free lang.

$a^n b^n c^n$ // "

Scanned by CamScanner

Page No.
Date

Q. Are CFLs closed under Intersection or not? NO

① $L_1 = a^n$ $L_2 =$ simple Palindrome

$L_1 \cap L_2 = L_1$ which is a CFL?

② $L_1 = a^m b^m a^n$ $L_2 = a^m b^n a^n$

CFG: $S \to XA$ CFG: $S \to AX$
$X \to aXb \mid ab$ $X \to bXa \mid ba$
$A \to aA \mid a$ $A \to aA \mid a$

$L_1 \cap L_2 \Rightarrow a^n b^n a^n$ which is not CFL

∴ we can say that CFLs are not closed under intersection becz. intersection of CFLs may or may not be equal.

Q. ③ $L_1 =$ palindrome (by default simple) & $L_2 = aa^* bb^* aa^*$
it is not regular but it is CFL. $S_2 = XYX$
$X \to aX$
$Y \to bX$
(reg & CF)

$S_1 : aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$

$L_1 \cap L_2$
↳ $a^n b^m a^n$ (it is CFL)

CFG → $S \to aSa$
$S \to aBa$
$B \to Bb \mid b$

Non reg but CFL.

Scanned by CamScanner

Page No.
Date

Q. (a) L_1 = EQUAL $\qquad$ $L_2 = a^n b^m a^n$

non-reg but CF $\qquad$ non-reg but CF

$S_1 \rightarrow aS_1bS_1 \mid bS_1aS_1 \mid \Lambda$ $\qquad$ $S_2 \rightarrow aS_2a$
$S_2 \rightarrow aBa$
$B \rightarrow Bb \mid b$

$L_1 \cap L_2 \Rightarrow a^n b^{2n} a^n$ $\qquad$ (Non-reg. Non CFL)

* Complement of a CFL may or may not be CFL.

Q. $M_{pq} = \{ a^p b^q a^r ;\ p > q$ & r is arbitrary$\}$

$\Downarrow$

$a^{p-q} \underbrace{a^q b^q}_{a^n b^n} a^r$

$S \rightarrow A S_1 A$
$A \rightarrow Aa \mid a$
$S_1 \rightarrow aS_1b \mid ab$

Q. $M_{qp} = \{ a^p b^q a^r ;\ q > p$ & r arbitrary$\}$

$a^p b^p b^{q-p} a^r$

$S \rightarrow XBA$
$X \rightarrow aXb \mid ab$
$B \rightarrow Bb \mid b$
$A \rightarrow Aa \mid a$

Scanned by CamScanner

Page No.
Date

Q. $M_{qr} = \{ a^p b^q a^r ;\ q > r \ \&\ p \text{ is cub.}$

$a^p \ b^{q-r} \ b^r a^r$

$S \rightarrow ABX$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$
$X \rightarrow bXa \mid ba$

Q1* $M_{rq} = \{ a^p b^q a^r ;\ r > q \}$
Q.2* $M_{pr} = \{ a^p b^q a^r ;\ p > r \}$
Q.3* $M_{rp} = \{ a^p b^q a^r ;\ r > p \}$

① $a^p b^q a^q a^{r-q}$
$S \rightarrow AXA$
$A \rightarrow aA \mid a$
$X \rightarrow bXa \mid ba$

② $a^{p-r} a^r b^q a^r$
$S \rightarrow AX$
$A \rightarrow aA \mid a$
$B \rightarrow bA \mid b$
$X \rightarrow \cancel{ABA} \mid \cancel{AA}$ $\cancel{AXBXA}$ $aBa \mid aXa$

③ $\underbrace{a^p b^q}\ a^p a^{r-p}$
$S \rightarrow XA$
$A \rightarrow aA \mid a$
$X \rightarrow aBa \mid aXa$

Q. $L = M_{pq} + M_{qp} + M_{qr} + M_{rq} + M_{pr} + M_{rp} + L_0'$

$L_0' = \{\Lambda, a, b, aa, ab, ba, bb, aaa, aab, abb, \ldots\}$

$L_0 = aa^* bb^* aa^*$
$= a^x b^y a^z \quad x, y, z \geq 1$

$L' = ?\ \ a^n b^n a^n$ (tell how)

if $L' = (a+b)^{'} \Leftrightarrow L' = a' \cap b'$

$L' = a^n b^n a^n \cap aa^* bb^*$
$= a^n b^n a^n$

Scanned by CamScanner

Page No.
Date

Q. $L_0 = aa^* bb^* aa^*$ — $L_0' = ?$

$L_0' = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, abb, \ldots\}$

L_0 is reg. reg. lang. are closed under compliment. ∴ L_0' is reg ∴ L_0' is CFL.

Q. Odd length palindrome to CNF

Imp

$S \to aSa \mid bSb \mid a \mid b$

$NT \to NT_1 NT_2$
$NT \to T$

$S \to a$ $S \to b$ $S \to XA$
$X \to AS$
$A \to a$

$S \to YB$
$Y \to bS$
$B \to b$

Q. $S \to aaaaS \mid aaaa$

Imp

$A \to aa$ ~~$A_2 \to a$~~ ~~$A_3 \to A$~~

aaaa $\begin{cases} S \to S_1 S_1 \\ S_1 \to AA \end{cases}$

~~$S_2 \to SS$~~

$S_1 \to AA$

aaaa $\begin{cases} A \to a \\ S_1 \to AA \\ S \to S_1 S_1 \end{cases}$

$S \to S_1 S_2$
$S_2 \to S_1 S$

12/Sep/19

⇒ In CNF

Live prod^n ⇒ $NT \to \underbrace{NT_1 NT_1}_{\text{LProd}^n}$

Dead prod^n ⇒ $NT \to \underbrace{T}_{\text{DProd}^n}$

SAKSHI

Scanned by CamScanner

Page No. ______
Date

Use of live production increase the no. of n-T by 1.
Use of dead production decreases the no. of n-T by 1.

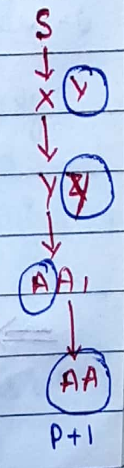
⇒ Observations w.r.t a CNF

Let G be a CFG in CNF if we restrict to use the live prod^n atmost one each we can generate only finitely many words.

* Let us suppose that a grammar G has exactly 'p' live prod^n & 'q' dead prod^n's.
becz. no. derivations ~~were~~ uses a live prod^n we can have atmost 'p' live prod^n's.
And we'll need atmost p+1 dead prod^n's.
Therefore, all words generated without repeating any live prod^n can have atmost p+1 letters in them.
Hence there can be only finitely many of them.

Eg.
1. $S \rightarrow XY$
2. $X \rightarrow YY$
3. $Y \rightarrow AA_1$
$Y \rightarrow b$
$A \rightarrow a$
4. $A_1 \rightarrow AA$

P=4 live pr^n.



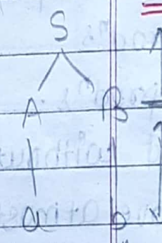
M-16 * Theorem 33

If G is CFG in CNF that has 'p' live prod^n & 'q' dead prod^n & if 'w' is a word generated by G that has more than 2^p letters in it.
Then somewhere in every derivation for 'w', there is some n-T being repeated twice. where this N-T is a tree descendent of the first.

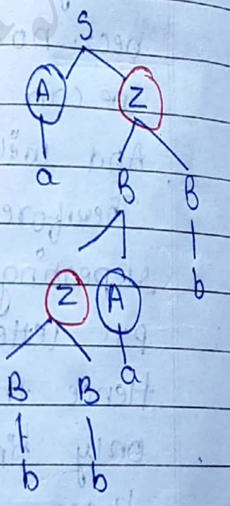
then there have been more than (P) prod^n in the ancestor path from a T symbol to S. But there are only p live prod^{n}s in G so if more than p have been used in the anchestor part then some live prod^n must have been used more than once.

In a derivation tree if bottom row has more than 2^p letters then tree must have more than p+1 rows. If any T from bottom row is considered & traced upto the tree top, we encounter one n-T after the another.
If there are more than p rows to re-traced (this will be the case if length(w) is more than 2^p), then there would be a repetition of live prod^n.

CFG -
S → AZ
Z → BB
B → ZA (p live prod^n)
A → a
B → b (1 dead prod^n)



1 - Live Prod^n
2 - dead prod^n



Z is true descendent of itself
Z → BB used twice
but A is not a true descendent of itsey.

⟹ **Self Embedded** - A NT is said to be self embedded if it occurs as a true descendent of itself.
Eg. Z in above tree.

13/sep/19

Q. ⟹ S → aS
S → Λ

make word - aaaa or a^4

symbol used is fixed

Page No. ____
Date ____

Derivation :-

$S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaS \Rightarrow aaaaS$

$L \Rightarrow a^*$ Working Strings / aaaa

* The sequence of app of rules that produces the finished string of T from the starting symbol is called a derivation or a generation of the word.

* Working strings : Unfinished stages in derivation.

Q. $S \to aS \quad S \to bS \quad S \to a \quad S \to b$

baab. $S \Rightarrow bS$ 2.
$\Rightarrow baS$ 1.
$\Rightarrow baaS$ 1.
$\Rightarrow baab$ 4.

L = all possible strings of a & b except Λ.

⇒ Left Most derivation : When in every working string if we expand only leftmost n-T then the derivation is known as L.M.D.

eg. $S \Rightarrow bSaS$
↳ if we expand this it is L.M.D.

Q. $S \to XaaX$ (atleast one double a)
$X \to aX \mid bX \mid \Lambda \to (a+b)^*$

baabaab

$S \Rightarrow XaaX \Rightarrow bXaaX \Rightarrow$ ~~baabX~~ baaX $\Rightarrow baabX$
$\Rightarrow baabaX \Rightarrow baabaaX \Rightarrow baabaabX$
$\Rightarrow baabaab.$

Scanned by CamScanner

Page No.
Date

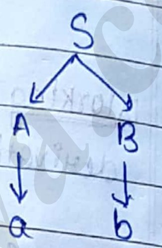
* Parse trees are also known as derivation tree, generation trees, syntax trees or production trees.

Q. $S \to AB \quad A \to a \quad B \to b$ ab

$S \Rightarrow AB \Rightarrow aB \Rightarrow ab$ (LMD)

or $\Rightarrow AB \Rightarrow Ab \Rightarrow ab$ (RMD)

* Parse tree of LMD & RMD is same, ∴ there is no ambiguity. Or the word is unambiguous.



Parse tree

Eg. $S \to S*S \mid S+S \mid$ number // ambiguous

Eg. $S \to (S*S) \mid (S+S) \mid$ number // unambiguous grammar

* A CFG is called ambiguous if for atleast one word in the lang that it generates there are two possible derivations of the word that correspond to different trees.

Eg. $S \to aS \mid Sa \mid a$ a^3

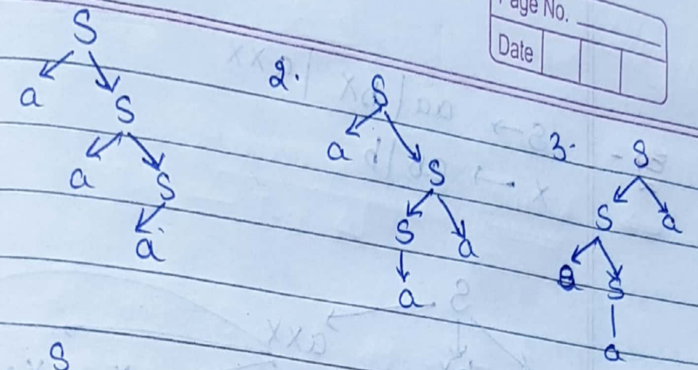
L = aa*

1. $S \Rightarrow aS \Rightarrow aaS \Rightarrow aaa$
3. $S \Rightarrow Sa \Rightarrow aSa \Rightarrow aaa$.
4. $S \Rightarrow Sa \Rightarrow Saa \Rightarrow aaa$.
2. $S \Rightarrow aS \Rightarrow aSa \Rightarrow aaa$.

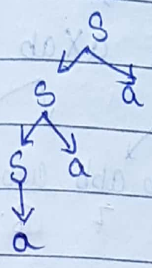
Scanned by CamScanner

Page No.
Date

1. 2. 3.



4.

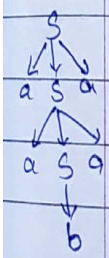


All the syntax trees are different. Therefore, this grammar is ambiguous.

Convert this lang into unambiguous grammar as :- $S \rightarrow aS \mid a$

Q. $S \rightarrow aSa \mid bSb \mid a \mid b \mid \Lambda$

aabaa



$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabaa$

$S \Rightarrow$ Only 1 possible derivative.

∴ this grammar is unambiguous.

⇒ Q. TOTAL LANGUAGE TREE

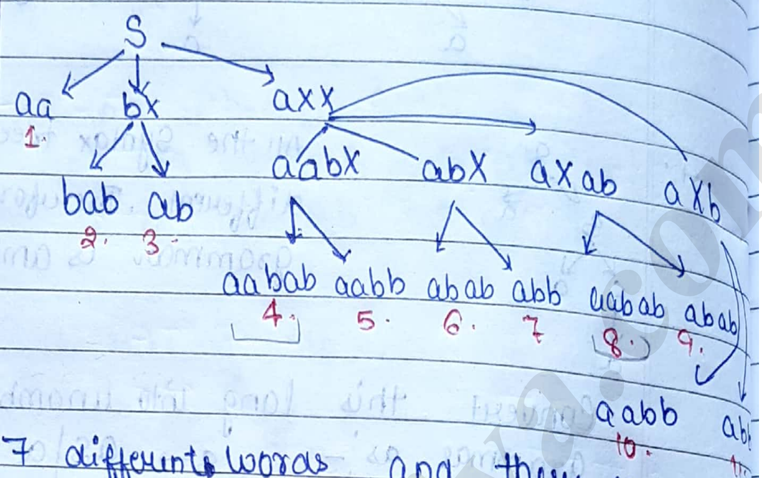
For a given CFG, we define a tree with the start symbol 'S' as its ~~rute~~ root & whose nodes are working strings of T & NTs. The descendants of each node are all the possible results of applying every applicable production to the working string, one at a time. A string of all Ts, is a T node in the tree.

The Resultant tree is called Total lang Tree of the CFG.

Scanned by CamScanner

Page No.
Date

Eg - $S \rightarrow aa \mid bX \mid aXX$
$X \rightarrow ab \mid b$



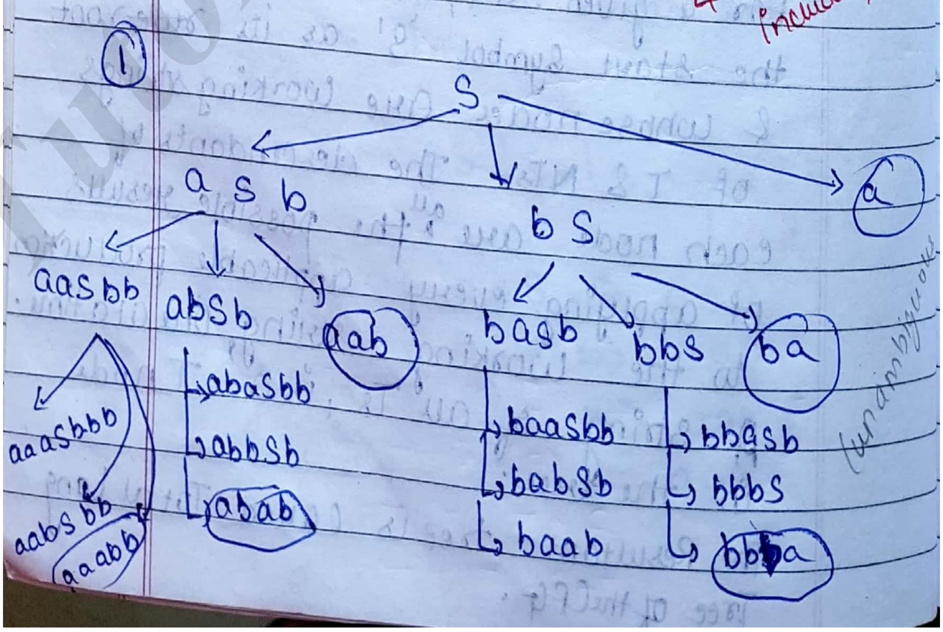
7 different words and there are 4 ~~dor~~ words that are repeated {abb, aabb, abab, aabab}

if for any repeated word their derivatives are same then the grammar is ambiguous.

Q.1 $S \rightarrow aSb \mid bS \mid a$
Q.2 $S \rightarrow X \mid b$
$X \rightarrow aX$

Make Total lang having 4 levels (including root)

(1)



Page No.
Date

(2) $S \to X \mid b$ $\quad X \to aX$

$$S \to Xb \to aXb \to aaXb \to aaaXb \to \vdots$$

$S \to b$ Terminal

Total lang tree is infinite becz CFG is incorrect but the lang is finite

aabab

$S \to aXX$, $X \to ab$, $X \to ab$

$S \to aXX$, $X \to ab$, $X \to ab$ (same)

aabb

$S \to aXX$, $X \to aab$, $X \to b$

$S \to aXX$, $X \to ab$, $X \to b$

check for all duplicates if their LMD & RMD are same then the lang is unambiguous.

Scanned by CamScanner

Section-7.2 from Book 2

18/sep/19

Referring Ch-13

Page No.
Date

→ Killing Unit Productions -
↳ n-T → n-T

EXAMPLE:-
S → A | bb
A → B | b
B → S | a

(3 Unit Productions are there we have to eliminate them)

Ans:
S → bb
S → b
S → a
A → a (A → B → a)
A → bb (A → B → S)
A → b (A → b)
B → a
B → bb (B → S)
B → b (B → S → A)

S → A
S → A → B
$S \overset{*}{\Rightarrow} B$
↳ S is going to B using 1 or more steps

or
S → a | b | bb
A → a | b | bb
B → a | b | bb

A & B production rule are redundant
S is a finite lang L = {a, b, bb}
becz. there is not nonterminal in any string produced from S.

EXAMPLE
I → a | b | Ia | Ib | I0 | I1
F → I | (E)
T → F | T * F
E → T | E + T

Scanned by CamScanner

Page No.
Date

Unit Prod^{n}s

$E \to T$

$E \overset{*}{\Rightarrow} F$

$E \overset{*}{\Rightarrow} I$

$T \Rightarrow F$

$T \overset{*}{\Rightarrow} I$

$F \to I$

After Elimination

$E \to T*F \mid E+T \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$T \to T*F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$F \to (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

(*) $I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

CNF do has $\wedge$ & unit prod^n.
It will also not have ~~useful~~ useless symbols. [those prod^n that are u

HW Q(1) Make CNF for the above (*) Grammar

Q(2) $S \to S \supset S \mid \sim S \mid (S) \mid p \mid q$

$\quad\quad\quad\ \ T \quad\quad\ T \quad\ T \quad\ T \ \ T$

Make $(p \supset (\sim p \supset q))$ find LMD

(1)

$E \to a$	$E \to ES_2$
$E \to b$	$S_2 \to PT$
$A \to a$	$Z \to 0$
$B \to B$	$O \to 1$
$E \to IA$	$E \to IZ$
$E \to IB$	$E \to IO$
$S \to *$	
$P \to +$	
$E \to TS_1$	
$S_1 \to SF$	

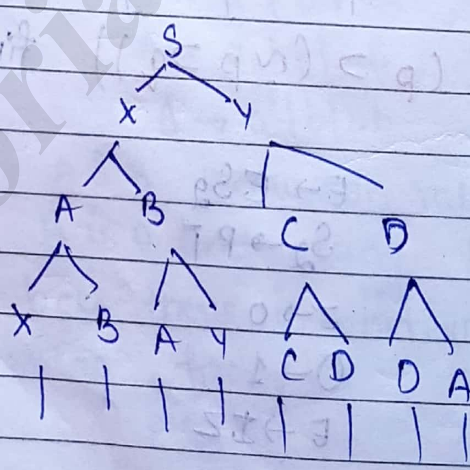
Scanned by CamScanner

(2) $S \Rightarrow (S) \Rightarrow (S \supset S) \Rightarrow (p \supset S) \Rightarrow (p \supset (S))$
$\Rightarrow (p \supset (S \supset S)) \Rightarrow (p \supset (\sim p \supset q))$

⟹ CHAPTER-16

live prod^n → NT → NT NT ....
dead prod^n → NT → T

Theorem 33



7 live prod.

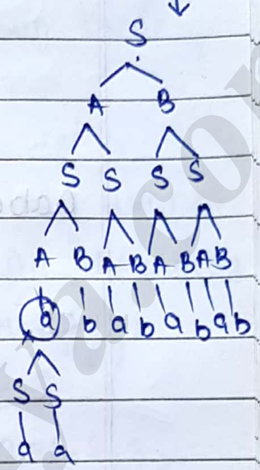
max. length of words that can be formed 7 production only once = 128.
if the word have length more than 128 then there will exist atleast

Page No.
Date

one live production that have been used more than one.

if bottom rows has 2^p letters.

then No. of row are more than ~~2p+1~~ p+1 rows

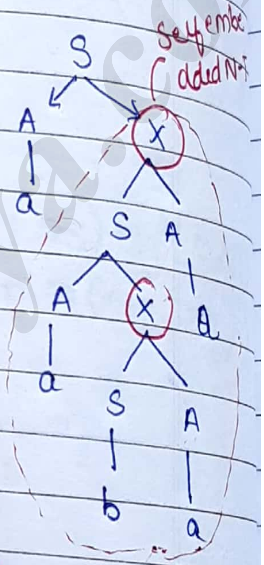


Hello.

→ Non-Null Palindrome (Odd Palindrome)

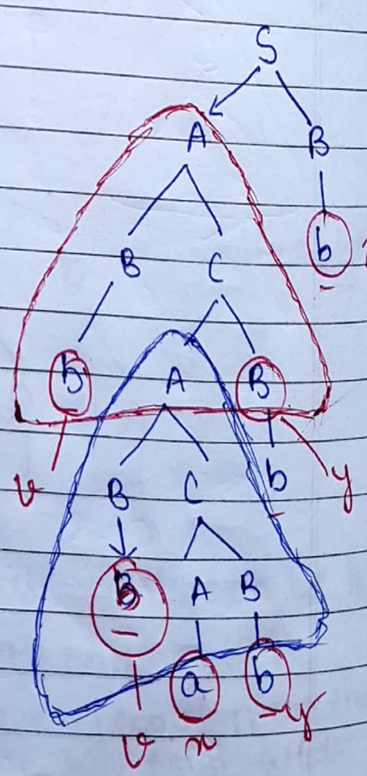
CFG :- $S \to AX$ $\quad S \to b$
$X \to SA$ $\quad S \to AA$
$S \to BY$ $\quad S \to BB$
$Y \to SB$ $\quad A \to a$
$S \to a$ $\quad B \to b$

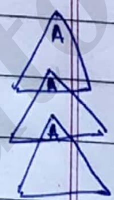
aabaa



⇒ $S \to AB$
$A \to BC$
$C \to AB$
$A \to a$
$B \to b$

bbabbb





$u = \lambda$
$z = b$
$v = b$
$y = b$
$x = a$

u

A is self embedded

u v v x y y z
| | | | | | |
A b b a b b b

Theorem 34

Page No.
Date

Pumping lemma for CFLs

If 'G' is any CFG in CNF with 'p' live prod^n s And 'w' is any word generated by G with $|w| > 2^p$ then we can breakup w into 5 substrings i.e.

$w = uvxyz$

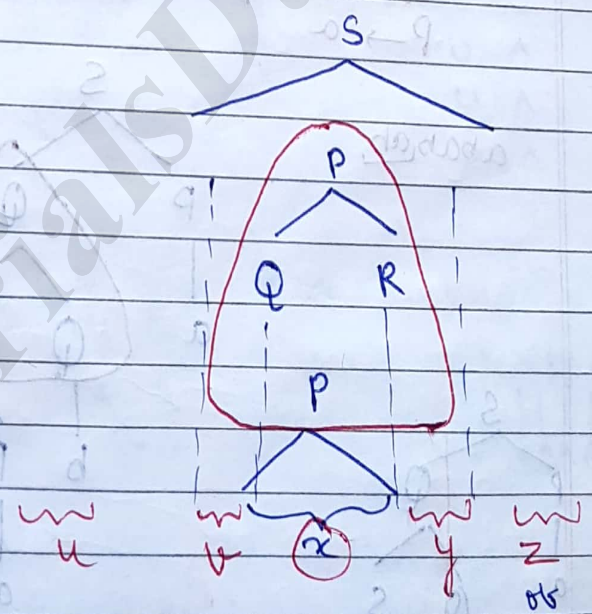
such that $x \neq \Lambda$ & v & y are not both Λ and such that all the words

$\underbrace{uvxyz, uvvxyyz, uvvvxyyyz}$

$uv^n xy^n z$ for $n = 1, 2, 3, \ldots$

can also be generated by 'G'.

Proof-



u = substring of all the letters 'w' generated to the left of triangle.

v = " " " " " " descendant from the first p but to the left of the letters by the second 'p' (this may be Λ).

Page No.
Date

x - substring of w descended from the lower 'p' (this may not be Λ becz. this N-T must turn into some Ts)

y - Substring of w generated by first p but to the right of the letters descendent from the second p. (this may be Λ but not if $v = \Lambda$)

z - substring of w generated ~~by first~~ p ~~but~~ to the right of Δ (may be Λ)

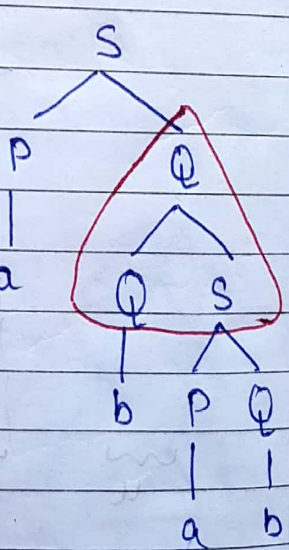
Example

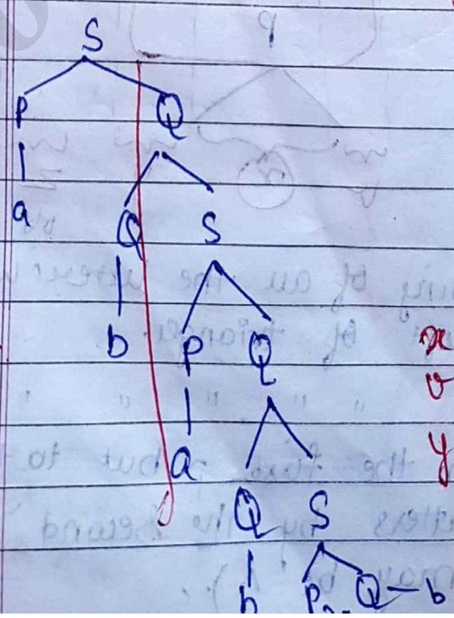
$S \rightarrow PQ$

$Q \rightarrow QS \mid b$

$P \rightarrow a$

ababab





$x = b$

$v = \Lambda$

$y = ab$

$u = a$

$z = \Lambda$

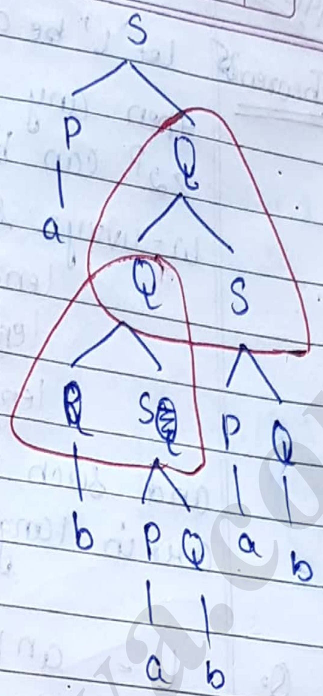
Inner Q, $x = b$

$v =$ null (Λ)

$u = a$

$y = ab$

$z = \Lambda$



pg 361 example see yrself.

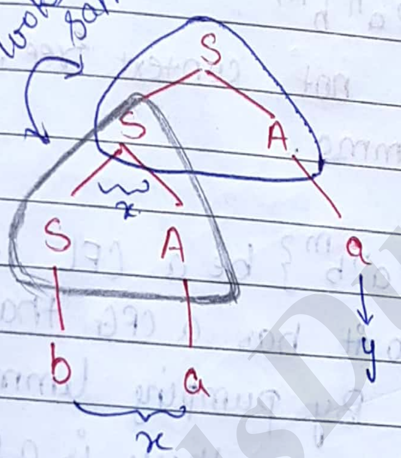
ababab

n is same.

$u\,v\,v\,x\,y\,y\,z$

$a\ \Lambda\ \Lambda\ b\ ab\ ab\ \Lambda$

Q. works same.



T1

$x = ba$

$y = a$

$v = \Lambda$

$u = \Lambda$

$z = \Lambda$

T2

$x = b$

$y = a$

$u = \Lambda$

$v = \Lambda$

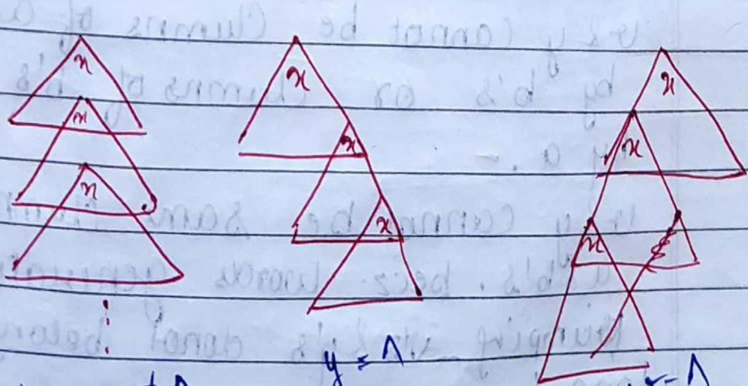
$z = \Lambda$

baa

$u\,v\,v\,x\,y\,y\,z$

$\Lambda\ \Lambda\ \Lambda\ b\ a\ a\ \Lambda$

Note



$v = y \neq \Lambda$ $\quad$ $y = \Lambda$, $v \neq \Lambda$ $\quad$ $v = \Lambda$, $y \neq \Lambda$

Scanned by CamScanner

Page No.
Date

19/Sep/19

Theorem 25 Let 'L' be a CFL in CNF with 'p' live prodn's, then any word 'w' in L with length $\geq 2^p$ can be broken into 5 parts:
$w = uvxyz$ such that

$$\text{length}(vxy) \leq 2^p$$
$$\text{length}(x) > 0$$
$$\text{length}(v) + \text{length}(y) > 0$$

and such that all words $uvvxyyz$, $uvvvxyyyz$, $uvvvvxyyyyz$ are in lang L.

Q. $L = a^n b^m a^n b^m$
Prove it is not context free, using pumping lemma.

Let 'L' $= \{a^n b^m a^n b^m\}$ be a CFL where $n, m \geq 1$ so it has a CFG that generates it. By pumping lemma of CFLs stated above there is a CFG that generates it and that CFG in CNF have 'p' live prod's.
Let the word be $a^{2^p} b^{2^p} a^{2^p} b^{2^p}$. As vxy has length $\leq 2^p$, v & y cannot be ab or ba.
v & y cannot be clumns of a's separated by b's or clumns of b's separated by a.
v & y cannot be same clumns of a & b's. becz. words generated by pumping v's & y's donot belong to lang L.
Hence, lang $L = a^n b^m a^n b^m$ is not a CFL

Scanned by CamScanner

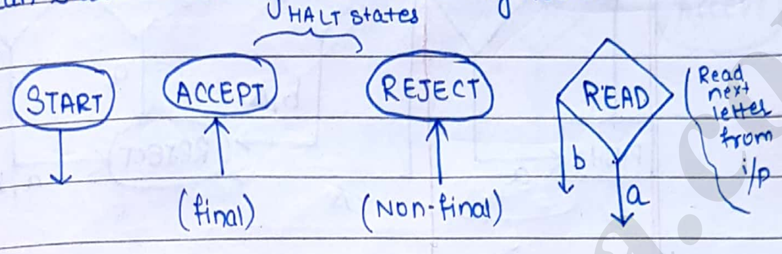
Page No.
Date

19/sep/15

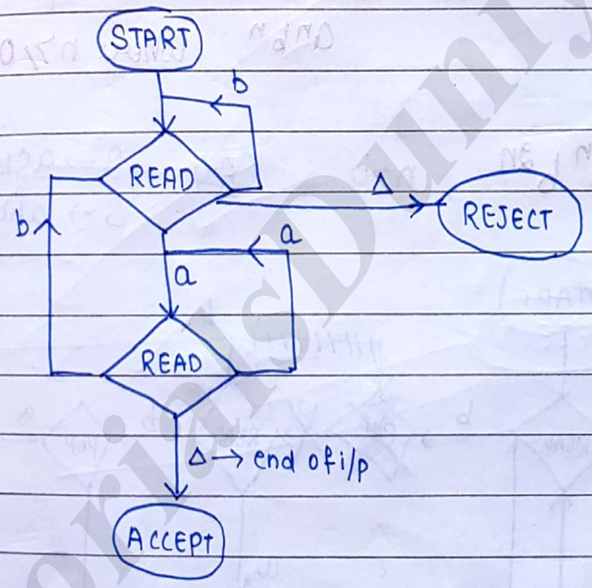
CH-14

⇒ PUSH DOWN AUTOMATA

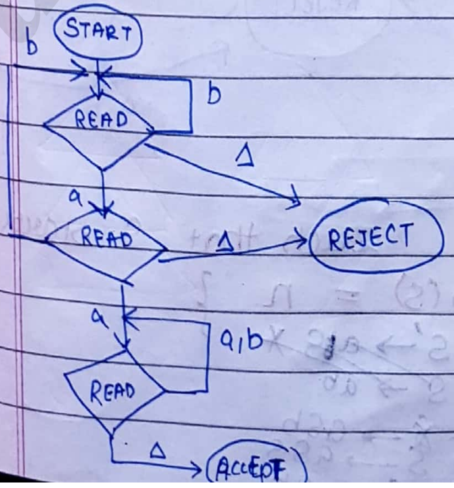
It is an NFA with ε-transitions permitted & one additional capability : a stack on which it can store a string of "stack symbols"

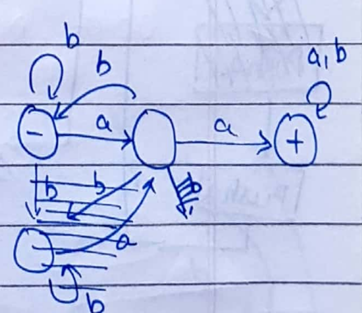


Q Ending in 'a'.



Q. Word having 'aa'.

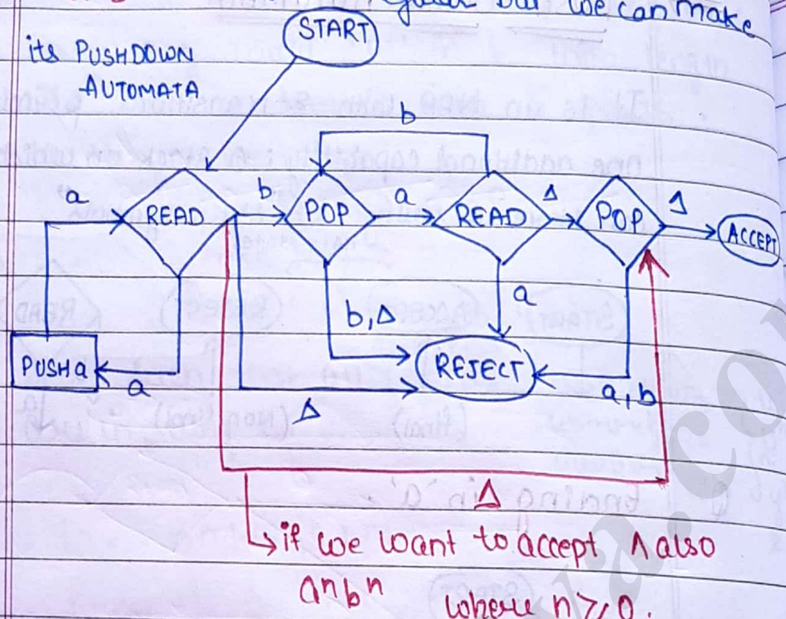




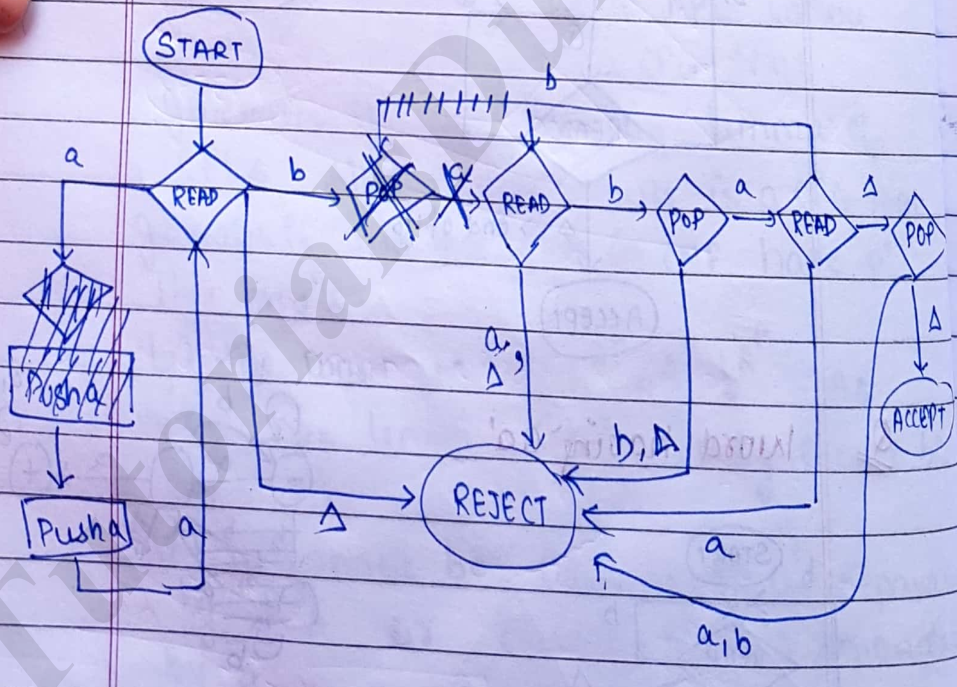
20/9/19

Page No.
Date

Q. $a^n b^n$ – it is not Regular but we can make its PUSHDOWN AUTOMATA



Q. $a^n b^{2n}$, $n > 0$ CFG $S \to aSbb$
$S \to abb$



$L = \{a^n S^b$ such that S starts with b & $length(S) = n\}$

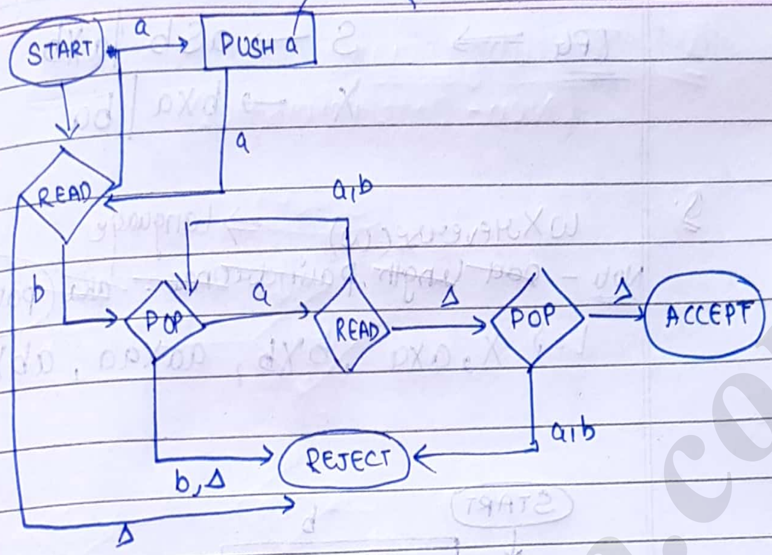
$S' \to aSS$ ✗
$S \to ab$

$S \to aSb$
$S \to aSa$
$S \to ab$

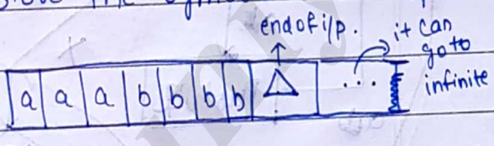
Scanned by CamScanner

Page No.
Date

it can be other letter also rather than letter from alphabet

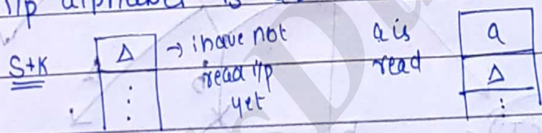


INPUT TAPE → store the symbols of the i/p string.



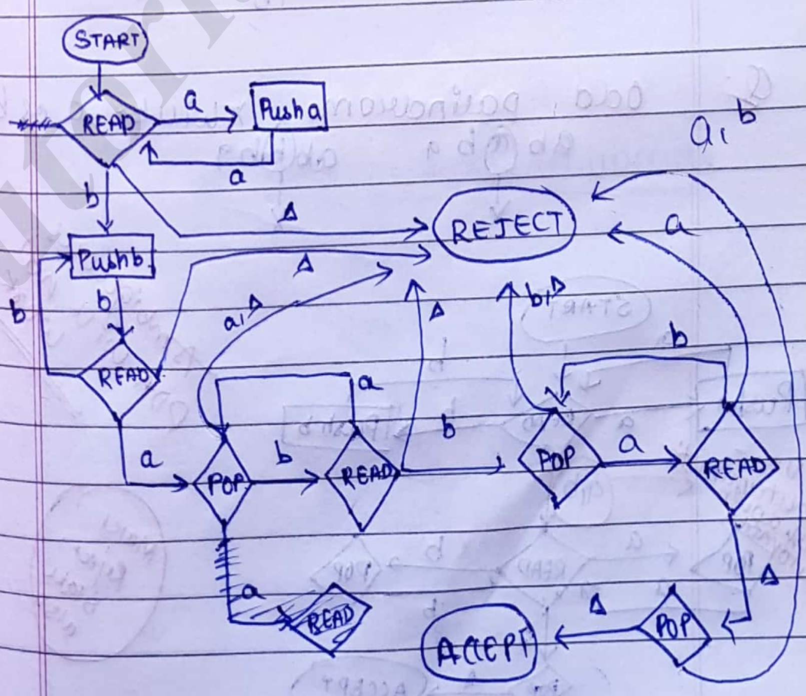
i/p alphabet is also known as tape alphabet.

S+K



23|Sep|19

Q. $a^n b^m a^m b^n$ — ~~NOT~~ Context free



Page No.
Date

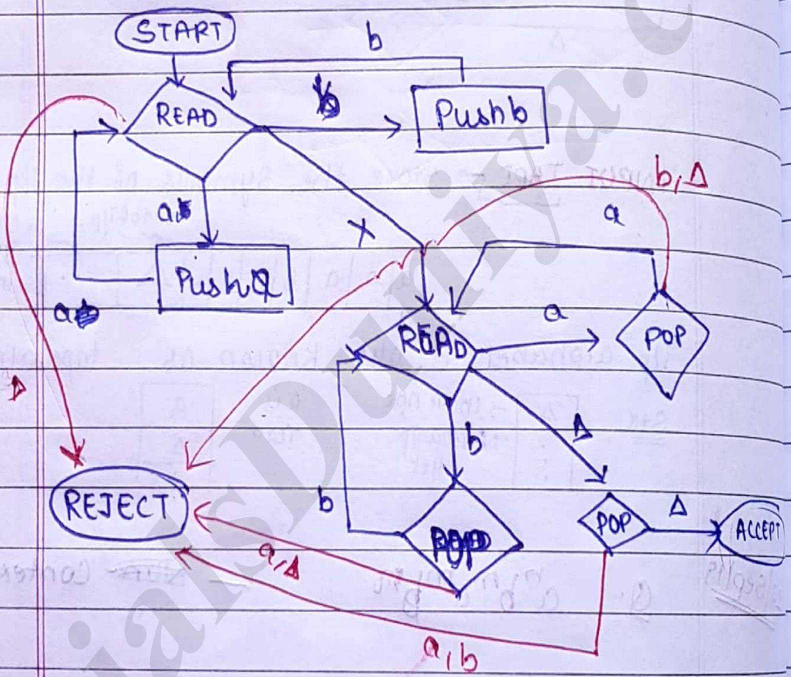
CFG ⟹ $S \rightarrow aSb \mid aXb$

$X \rightarrow bXa \mid ba$

Q. wXreverse(w) → language

Note – ~~odd length palindrome~~. ~~aka~~ palindromeX

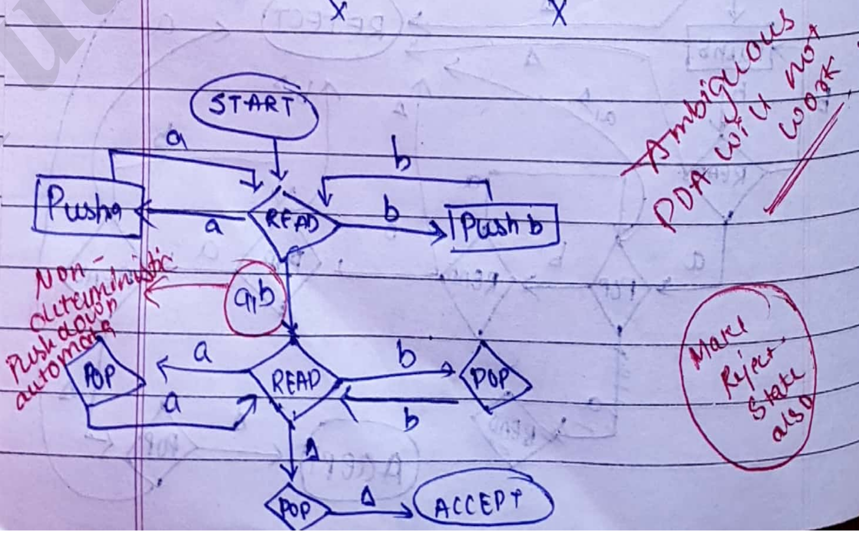
$L = \{ X, aXa, bXb, aaXaa, abXba, \ldots \}$



Q. odd palindrome X will be a or b

ab(a)ba   ab(b)ba

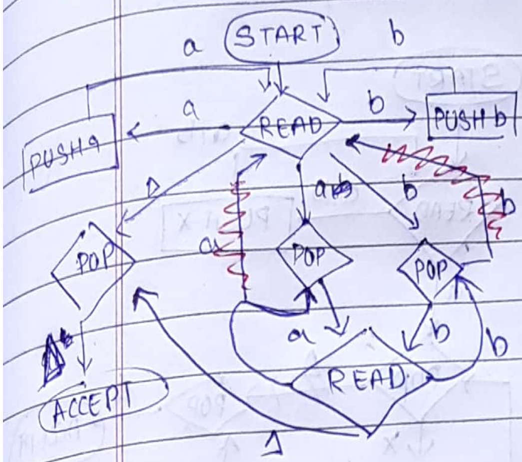
X   X



Scanned by CamScanner

25/sep/19

Q. even palindrome - Non-deterministic automata



Valid string will be accepted by atleast one paths of the possible.

H/W

Q.1 Equal no. of a & b's.

Q.2 $a^n b^{n+1}, n \geq 1$ — (4) marks.

Q.3 $a^n b^{2n}$ Show that lang. is non-reg using pumping lemma — (4)

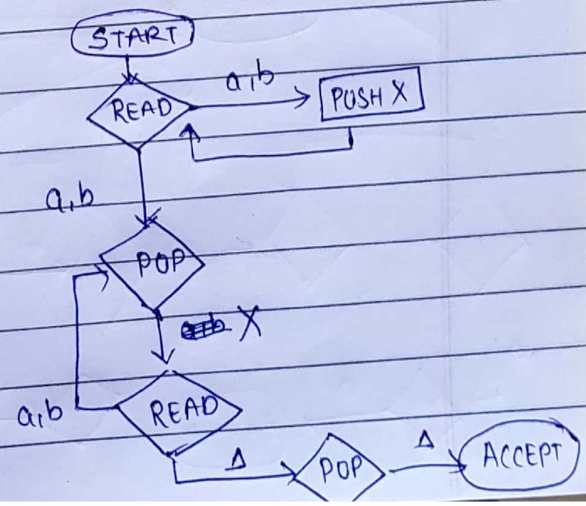
Q.4 $a^n b^n a^n$ is non-context free Proof.

Q. Even length words Pushdown Automata & CFG

$S \rightarrow aSa \mid aSb \mid bSa \mid bSb \mid \Lambda$

Q. ~~$a^n b^{n+1}$~~ — ~~4 marks~~.

Non-deterministic automata



Scanned by CamScanner

Page No.
Date

Q. Trailing Count = w a^{length(w)}
(Non-deterministics)