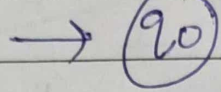
Page :
Date : / /

Unit 2

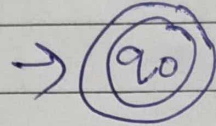
Regular Expression

Conversion of regular expression to finite automata :—

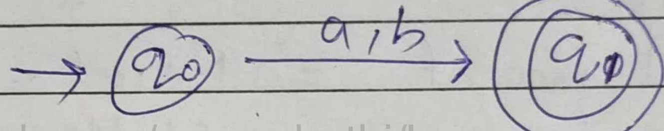
(i) ϕ



(ii) ϵ / $\wedge$

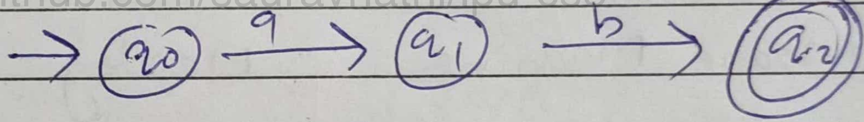


(iii) a+b

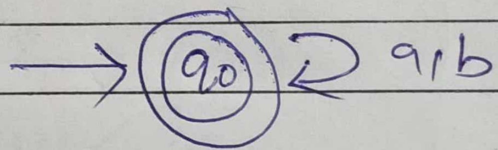


https://github.com/sauravhathi/lpu-cse

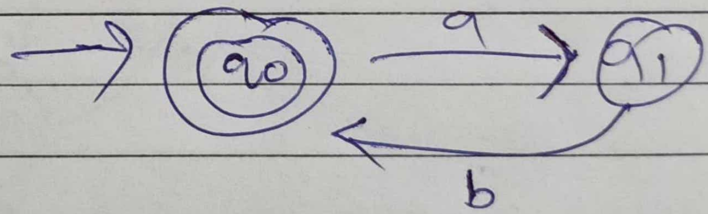
(iv) ab



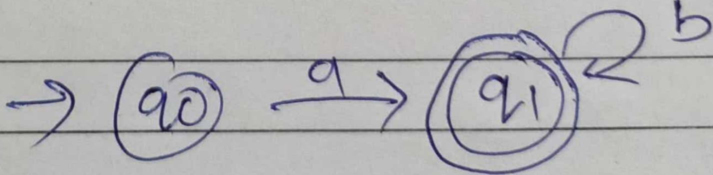
(v) $(a+b)^*$



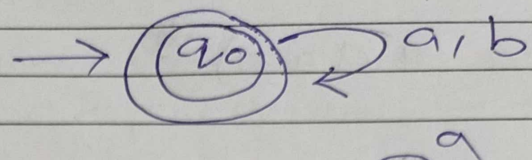
(vi) $(ab)^*$

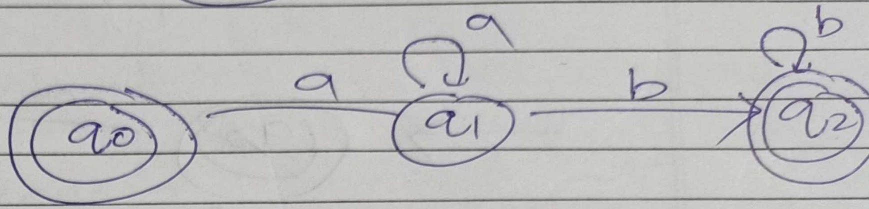


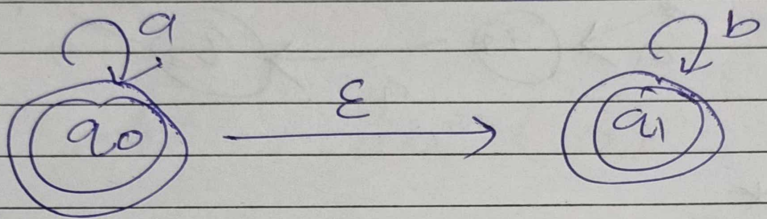
(vii) ab^*



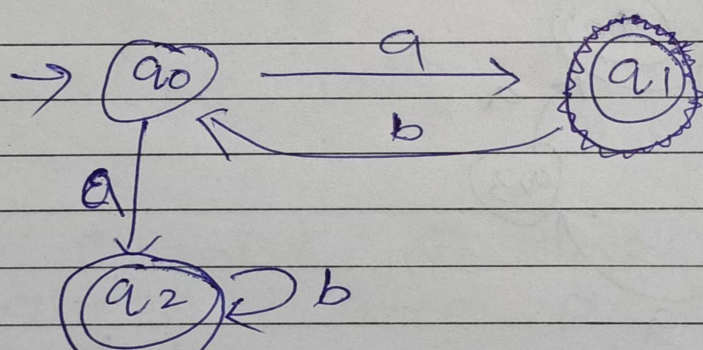
(8) $a^* . b^*$







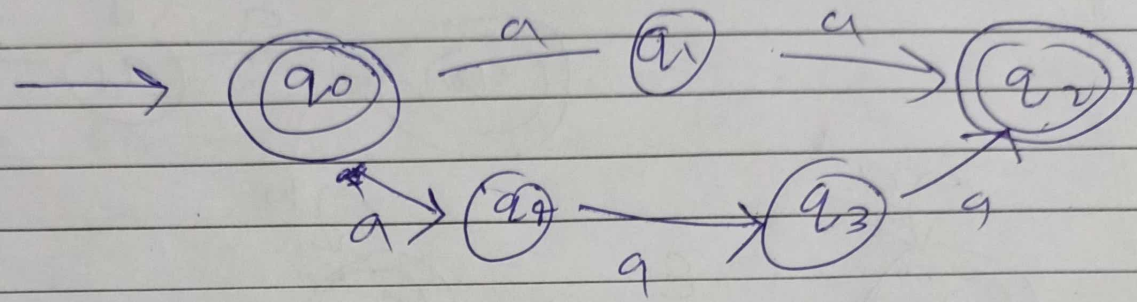
(9) $(ab)^* \; a \; b^*$



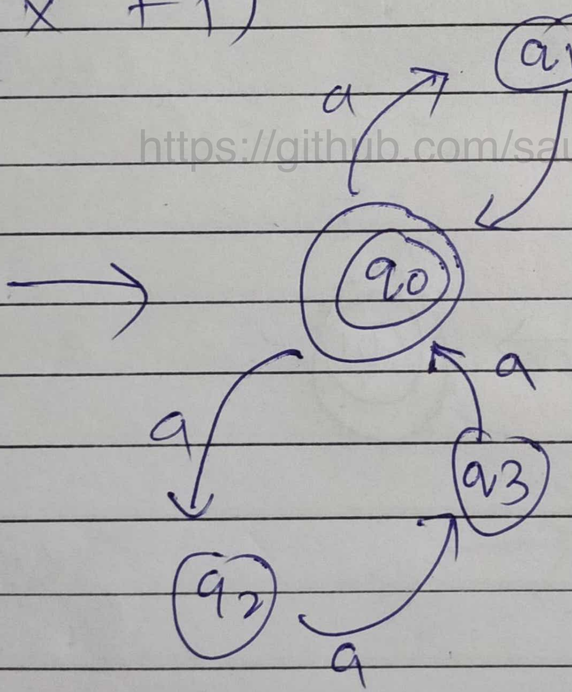
$\phi . 0 = 0$ $\qquad$ $\phi . 1 = 0$

$\varepsilon . 1 = 1$ $\qquad$ $\varepsilon . 0 = 0$ $\qquad$ $\varepsilon . (0+1) = 0+1$

10) $(aa + aaa)^*$



$(x + y)^*$



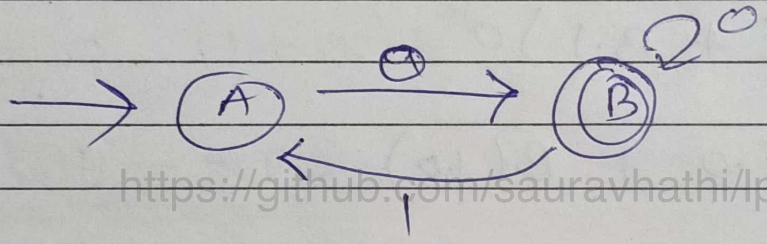
Conversion from finite Automata to Regular expression:-

(i) Direct Method

(ii) Arden's Theorem

(i)

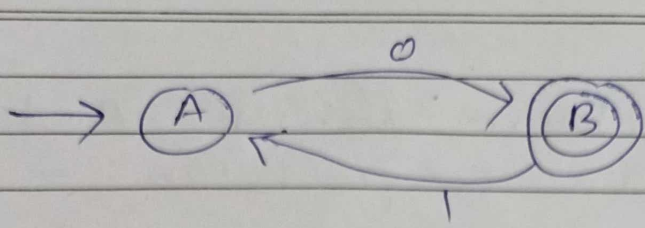
$00^*(10)^*$



$0(0+10)^*$

(ii) If P and Q are 2 regular expressions and P does not contain null then the equation $R = Q + RP$ has a unique sol^n $R = QP^*$

R is state



$A = \varepsilon + B.1$ — (I)

$B = A.0$ — (II)

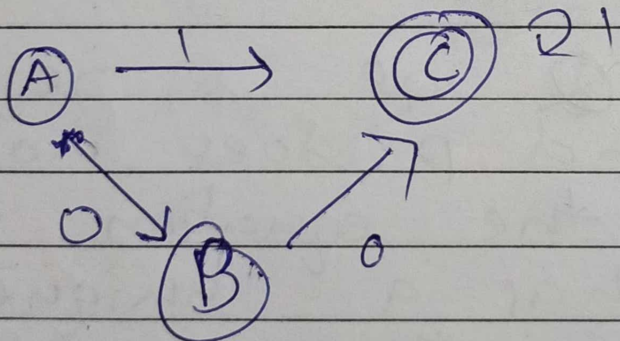
(I) in (II)

$B = (\varepsilon + B.1)0$

$B = \varepsilon.0 + B(1.0)$

$B = 0 + B(1.0)$

$B = 0(10)^*$

Q) 

$\cancel{+(00+1)}^*$ $11^* +$
001^*

$(1+00)1^*$

$A = \cancel{B \cdot 0} \quad \varepsilon$

$B = A \cdot 0$

$C = A \cdot 1 + B \cdot 0 + C \cdot 1$

$B = \varepsilon \cdot 0 \quad \Rightarrow \quad B = 0$

$C = \varepsilon \cdot 1 + 0 \cdot 0 + C \cdot 1$

$C = 1 + 00 + C \cdot 1$

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

Null - NFA

$\delta : Q \times \Sigma \rightarrow 2^Q$ NFA

$\delta : Q \times (\Sigma \cup \Lambda) \rightarrow 2^Q$ ε-NFA

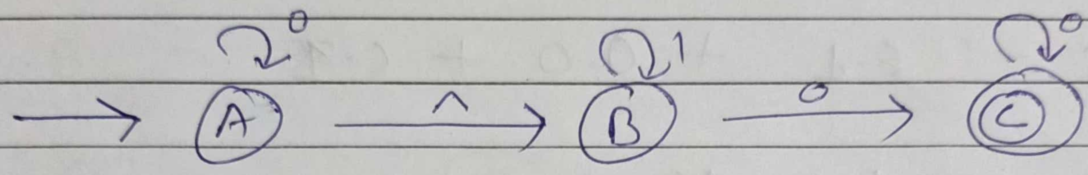
Removal of null :-

(i) If there exist null between A and B states then copy the transitions of B to A and remove the null

Page :
Date : / /

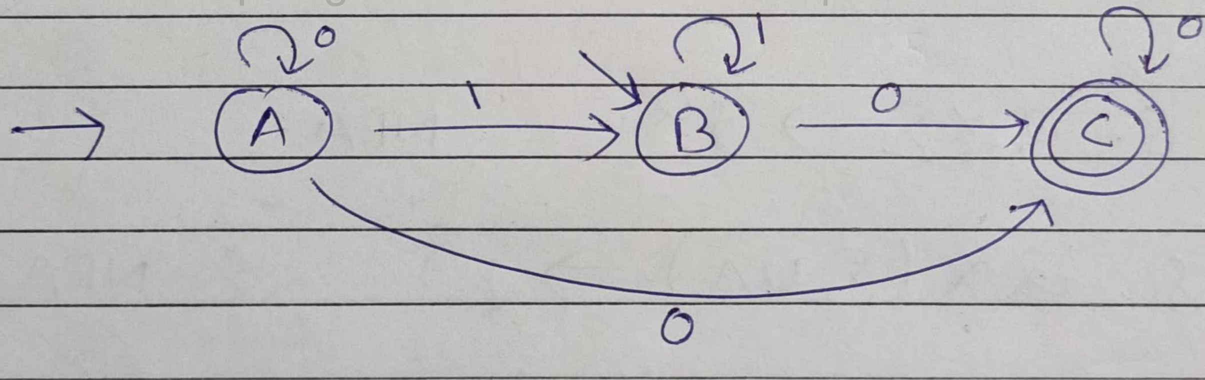
(ii) If A is initial state make B also as initial state.

(iii) If B is the final state make A also as the final state



$\delta(B, 0) \rightarrow C$ $\delta(B, 1) \rightarrow B$

$\delta(A, 0) \rightarrow C$ $\delta(A, 1) \rightarrow B$



Pumping Lemma :-

$L = a^n . b^n$; $n \geq 0$ Not Regular

$L = a^n . b^m$; $n > m$ Not regular

$L = a^n . b^m$; $n, m \geq 0$ Regular

There is no memory in finite automata therefor in a language there is a requirement of memory, then the language is not regular.

It is a negation taste to test to check whether the langaag is regular or not.

Statement → If L is a regualar language with any integer n such that $W \in L$ where $|W| \geq n$ and $W = XYZ$ then it must follow

(i) $|XY| \leq n$ (ii) $|Y| \geq 1$

(iii) For all $i \geq 0$; $xy^i z \in L$

$L = a^n b^n$; $n \geq 0$

$w = aaabbb$ $\quad\quad n = 3$

$|w| = 6$

$$w = \underbrace{a\ a}_{x}\ \underbrace{a}_{y}\ \underbrace{b\ b\ b}_{z}$$

$$\underbrace{aa}_{x}\ \underbrace{a^i}_{y}\ \underbrace{bbb}_{z}$$

$L \xrightarrow{\text{PL satisfies}}$ Not decidable (50 % chance)

$L \xrightarrow{\text{PL Dissatisfies}}$ Decidable (Always not regular)

* Properties of Regular Expression :-

i) Associativity

$$R_1 + (R_2 + R_3) = (R_1 + R_2) + R_3$$

Page :
Date : / /

$R_1 \cdot (R_2 \cdot R_3) = (R_1 \cdot R_2) \cdot R_3$

associative property holds for both union and concatenation

(ii) Commutative

$R_1 + R_2 = R_2 + R_1$

$R_1 \cdot R_2 \neq R_2 \cdot R_1$

$a \cdot b \neq b \cdot a$

Commutative property holds only for union.

(iii) Idempotent

$R_1 + R_1 = R_1$

$R_1 \cdot R_1 \neq R_1$

Idempotent property holds only for union.

Closure Properties

(i) Union (ii) Concatenation

(iii) Kleene

$a^* = \{\varepsilon, a, aa, aaa, ---\}$

$a^+ = \{a, aa, aaa, ----\}$