4th October, 2023

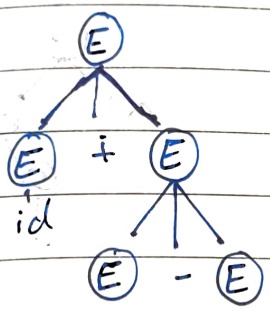
UNIT - 3

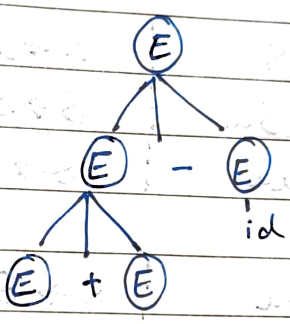
Parsing
↓
Derivation / Parse Tree

$E \rightarrow E + E \mid E - E \mid id$

id + id - id

Parse Tree

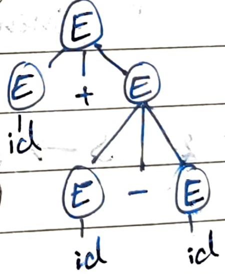




* if more than 1 parse tree for grammar
  ⇒ ambigous grammar.

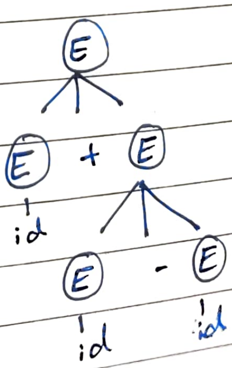
Derivation
LM

$E \Rightarrow E + E$
$\Rightarrow id + E$
$\Rightarrow id + E - E$
$\Rightarrow id + id - E$
} sentential form
$\Rightarrow id + id - id$ } sentence (no non terminal)



RM

$E \Rightarrow E + E$
$\Rightarrow E + E - E$
$\Rightarrow E + E - id$
$\Rightarrow E + id - id$
$\Rightarrow id + id - id$



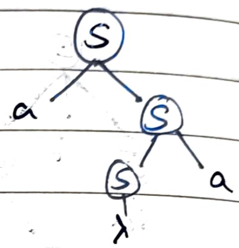
* if parse tree is same for LM and RM for grammar ⇒ non-ambiguous grammar

q. Find out whether grammar is ambiguous or not.

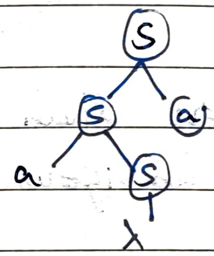
1. $S \to aS \mid Sa \mid \lambda$

i) choose min len string | what production

$S \Rightarrow aS$	$S \to aS$
$\Rightarrow aSa$	$S \to Sa$
$\Rightarrow aa$	$S \to \lambda$



$S \Rightarrow Sa$	$S \to Sa$
$\Rightarrow aSa$	$S \to aS$
$\Rightarrow aa$	$S \to \lambda$



Since two different parse trees for the same grammar for a given string ⇒ ambiguous grammar

2. $S \to aSbS \mid bSaS \mid \lambda$

$S \Rightarrow aSbS$
$\Rightarrow aaSbS\,b\,aSbS$
$\Rightarrow aabbab$

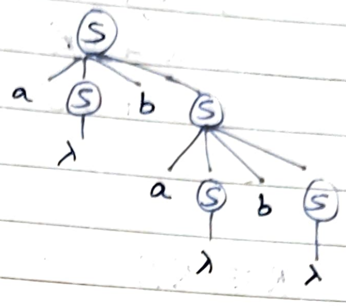
take first production same for both to identify.

$\times$

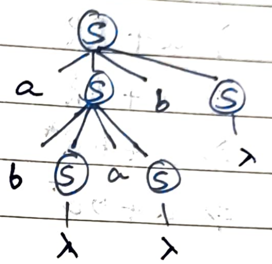
$S \Rightarrow bSaS$
$\Rightarrow b\,bSaS\,a\,bSaS$
$\Rightarrow bbaaba$

* consider 2 LM | 2 RM | 1 LM, 1 RM and check parse tree

$S \Rightarrow aSbS$
$\Rightarrow abS$
$\Rightarrow abaSbS$
$\Rightarrow ababS$
$\Rightarrow abab$



$S \Rightarrow aSbS$
$\Rightarrow abSaSbS$
$\Rightarrow ab\,aSbS$
$\Rightarrow ab\,abS$
$\Rightarrow abab$

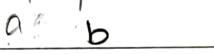


3. $R \to R+R \mid RR \mid R^* \mid a \mid b \mid c$

Two LMD
Two RMD
One LMD & one RMD

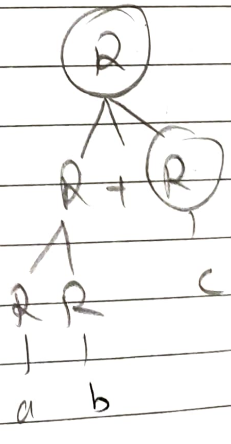
$ab+c$

$R \Rightarrow R+R$ ⇒ $R \to R+R$
$RR+R$ $R \to RR$
$aR+R$ $R \to a$
$ab+R$ $R \to b$
$ab+c$ $R \to c$

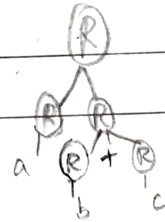


$R \Rightarrow R+R$
$RR+R$
$RR+c$
$Rb+c$
$ab+c$

X



$R \Rightarrow RR$
$\Rightarrow RR+R$
$\Rightarrow aR+R$
$\Rightarrow ab+R$
$\Rightarrow ab+c$



$R \Rightarrow R + R$
$\Rightarrow a + R$
$\Rightarrow a + RR$
$\Rightarrow a + bR$
$\Rightarrow a + bR^*$
$\Rightarrow a + bc^*$

$R \Rightarrow RR$
$\Rightarrow R + RR$
$\Rightarrow a + RR$
$\Rightarrow a + bR$
$\Rightarrow a + bR^*$
$\Rightarrow a + bc^*$

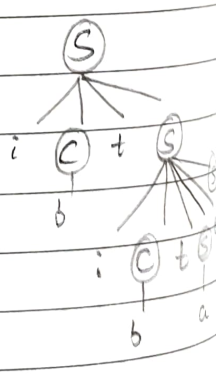
Q1. Derive a string ibtibtaea from the grammar and also draw parse tree

$S \rightarrow iCtS \mid iCtSeS$
$S \rightarrow a$
$C \rightarrow b$

Sol.

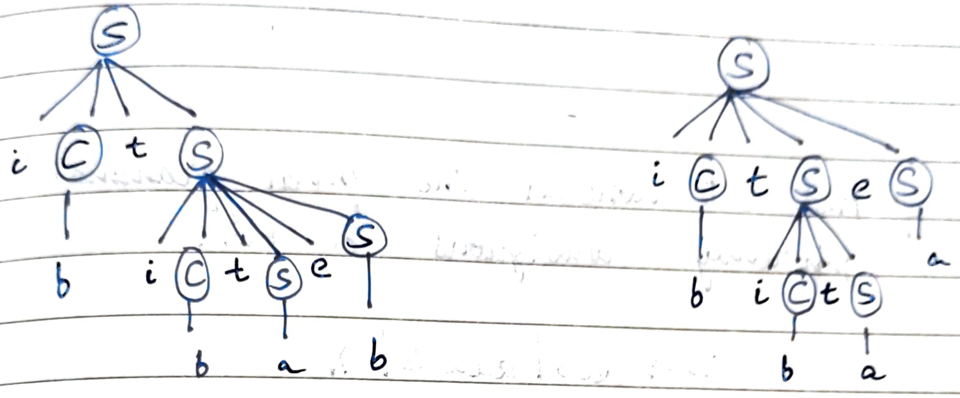
LMD		RMD	
$S \Rightarrow iCtS$	$S \rightarrow iCtS$	$S \Rightarrow iCtS$	$S \Rightarrow iCtS$
$ibtS$	$C \rightarrow b$	$\Rightarrow iCtiCtSeS$	$S \rightarrow iCtSeS$
$ibtiCtSeS$	$S \rightarrow iCtSeS$	$iCtiCtSea$	$S \rightarrow a$
$ibtibtSeS$	$C \rightarrow b$	$iCtiCtaea$	$S \rightarrow a$
$ibtibtaeS$	$S \rightarrow a$	$iCtibtaea$	$C \rightarrow b$
$ibtibtaea$	$S \rightarrow a$	$ibtibtaea$	$C \rightarrow b$

parse tree (any 1)



q2. Check whether the grammar is ambiguous
$S \rightarrow iCtS \mid iCtSeS \mid a$
$C \rightarrow b$

Sol. Parse tree 1 & Parse tree 2
(should be different)



→ inherently ~~in~~ambiguous

converting ambiguous → unambiguous

$E \rightarrow E+E \mid E-E \mid E*E \mid E/E \mid id \mid (E)$

ambiguous	unambiguous
$E \rightarrow E+E \mid E-E$	$E \rightarrow E+T \mid E-T \mid T$
$E \rightarrow E*E \mid E/E$	$T \rightarrow T*F \mid T/F \mid F$
$E \rightarrow id \mid (E)$	$F \rightarrow (E) \mid id$

q. Find out whether the given grammar is inherently ambiguous or not

$S \rightarrow SaS \mid b$

$S \rightarrow SaS \mid b$

regex: $b(ab)^+$

$S \rightarrow bA$
$A \rightarrow abA \mid ab \mid \lambda$

Not inherently ambiguous

every RG is CFG.

q. Find out whether the given grammar is inherently ambiguous or not.

$S \rightarrow aaS \mid aaaS \mid \lambda$

regex: $(aa + aaa)^*$

$S \rightarrow aaA \mid \lambda$
$A \rightarrow aA \mid \lambda$

q. $S \rightarrow AB \mid aaB$
$A \rightarrow a \mid Aa$
$B \rightarrow b$

regex: aa^*b

$S \rightarrow aAb$
$A \rightarrow aA \mid \lambda$

It is not inherently ambiguous

Normalization of grammar

1. CNF (Chomsky Normal Form)

$A \to BA$
$B \to c$

right side either 2 non-terminal or one terminal

$A \to B$ => unit production
$A \to \lambda$ => null
useless and unreachable symbols
Left recurssion $A \to AB$

not allowed → simplify

2. Greibach Normal Form

$V \to T V^*$ Terminal

$\alpha \in V^*$

Simplified form
λ not allowed

eg $A \to aBc$
$A \to a$
$A \to BC$ ×
$A \to aaB$ ×

→ CNF

q.1. Convert given CFG to CNF

1. $S \to aX \mid Yb$
$X \to S \mid \lambda$
$Y \to bY \mid b$

Step 1. Simplify Grammar
a. Eliminate λ-production - replace
b. Eliminate unit production - substitute
c. Eliminate useless & unreachable prods - remove

Step 2: Convert to CNF
a. Replace more than 1 T to N.T
b. Restrict no. of NT to 2

q. $S \to aX|Yb$
$X \to S|\lambda$
$Y \to bY|b$

Step 1 a. Eliminate λ.
$X \to \lambda$

$S \to aX|a|Yb$
$X \to S$
$Y \to bY|b$

b. Eliminate unit production

$X \to S$
substitute S with its production

$X \to aX|a|Yb$

grammar is : $S \to aX|a|Yb$
$X \to aX|a|Yb$
$Y \to bY|b$

for useless ⇒ check if

Step 2 Convert to CNF
$S \to aX|a|Yb$
$X \to aX|a|Yb$
$Y \to bY|b$

	adding new NT	
$S \to aX$	$A \to a$	$S \to AX$
$S \to Yb$	$B \to b$	$S \to YB$
$X \to aX$		$X \to AX$
$X \to Yb$		$X \to YB$
$Y \to bY$		$Y \to BY$

CNF : $S \to AX \mid YB \mid a$

$X \to AX \mid YB \mid a$

$Y \to BY \mid b$

$A \to a$

$B \to b$

* To remove left recursion

$$A \to A\alpha_i \mid \beta_i$$

eg: $A \to A\underset{\alpha}{a} \mid \underset{\beta}{b}$

$$A \to \beta_i A' \mid \beta_i$$
$$A' \to \alpha_i A' \mid \alpha_i$$

$A \to bA' \mid b$

$A' \to aA' \mid a$

eg: $A \to A\underset{\alpha_1}{a} \mid A\underset{\alpha_2}{b} \mid A\underset{\alpha_3}{c} \mid \underset{\beta_1}{a} \mid \underset{\beta_2}{b} \mid \underset{\beta_3}{c}$

$A \to aA' \mid bA' \mid cA' \mid a \mid b \mid c$

$A' \to aA' \mid bA' \mid cA' \mid a \mid b$

6th October, 2023

Q2. $S \to aSa \mid bSb \mid A \mid \lambda$
$A \to a \mid \lambda \mid b$

Step 1: Simplify grammar (no λ, no unit, no useless, no unreachable),

Eliminate λ prodⁿ $S \to aSa \mid aa \mid bSb \mid bb \mid A \mid \lambda$
$A \to a \mid b$

only for empty string

* only for start symbol we allow acceptance of empty strings

Eliminate unit prodⁿ (by substitution)

$S \to aSa \mid aa \mid bSb \mid a \mid b \mid \lambda$
$A \to a \mid b$

A is unreachable
⇒ Simplified grammar: $S \to aSa \mid aa \mid bSb \mid a \mid b \mid \lambda$

Step 2: CFG to CNF
→ $V \to VV$
$V \to T$

original	substitution	new production
$S \to aSa$	$A \to a$	$S \to ASA$
$S \to aa$	$B \to b$	✓$S \to AA$
$S \to bSb$		$S \to BSB$
$S \to bb$		✓$S \to BB$
already CNF $S \to a$		$S \to a$
$S \to b$		$S \to b$
$S \to \lambda$		$S \to \lambda$
		$A \to a$
		$A \to b$

$S \to ASA$
$S \to BSB$
$S \to AA$
$S \to BB$
$S \to a$
$S \to b$
$S \to \lambda$
$A \to a$
$A \to b$

$X \to AS$
$Y \to SB$

$S \to XA$
$S \to BY$
$S \to AA$
$S \to BB$
$S \to a$
$S \to b$
$S \to \lambda$
$A \to a$
$A \to b$
$X \to AS$
$Y \to SB$

q3. $S \to BAB$
$A \to Bc$
$B \to bba$

useless symbol $\Rightarrow$ non terminable grammar

grammar simplified

original
$S \to BAB$
$A \to Bc$
$B \to bba$

substitution
$L \to BA$
$F \to c$
$G \to b$

$S \to LB$
$A \to BF$
$B \to GGL$
$L \to BA$
$\cancel{S \to BAB}$
$F \to c$
$G \to b$

q4. $S \to Aa \mid aB \mid b \mid Ca$
$B \to aB \mid b$
$C \to Db \mid ab \mid d$
$D \to ab \mid d$
$E \to ab$

simplified grammar:
$S \to aB \mid b \mid Ca$
$B \to aB \mid b$
$C \to Db \mid ab \mid d$
$D \to ab \mid d$

$X \to a$
$Y \to b$

$S \to XB \mid b \mid CX$
$B \to XB \mid b$
$C \to DY \mid XY \mid d$
$D \to XY \mid d$
$X \to a$
$Y \to b$

q5. $S \to aAa \mid bBb \mid BB$
$A \to C$
$B \to S \mid A$
$C \to S \mid \lambda$

eliminate λ
SABC are nullable

$S \to aAa \mid aa \mid bBb \mid BB \mid bb \mid B \mid \lambda$
$A \to C$
$B \to S \mid A$
$C \to S$

removing unit productions

$S \to aAa \mid aa \mid bBb \mid bb \mid BB \mid \lambda$
$A \to aAa \mid aa \mid bBb \mid bb \mid BB$
$B \to aAa \mid aa \mid bBb \mid bb \mid BB$
$C \to aAa \mid aa \mid bBb \mid bb \mid BB$

A and B are reachable   C is unreachable.

$S \to aAa \mid aa \mid bBb \mid bb \mid BB \mid \lambda$
$A \to aAa \mid aa \mid bBb \mid bb \mid BB$
$B \to aAa \mid aa \mid bBb \mid bb \mid BB$

Simplified grammar

$X \to a$
$Y \to b$

$S \to XAX \mid XX \mid YBY \mid YY \mid BB \mid \lambda$
$A \to XAX \mid XX \mid YBY \mid YY \mid BB$
$B \to XAX \mid XX \mid YBY \mid YY \mid BB$
$X \to a$
$Y \to b$

$L \to XA$
$M \to YB$

$S \to LX \mid XX \mid MY \mid YY \mid BB \mid \lambda$
$A \to LX \mid XX \mid MY \mid YY \mid BB$
$B \to LX \mid XX \mid MY \mid YY \mid BB$
$X \to a$
$Y \to b$
$L \to XA$
$M \to YB$

final CNF

Q6. $E \to E + T \mid T$
$T \to T * F \mid F$
$F \to num \mid id$

q7. Simplify $S \to ABA$
$A \to aA \mid \lambda$
$B \to bB \mid \lambda$

$S \to ABA \mid AB \mid BA \mid AA \mid A \mid B \mid \lambda$
$A \to aA \mid a$
$B \to bB \mid b$

q6.

removing unit prodn.

$E \to E+T \mid T*F \mid num \mid id$
$T \to T*F \mid num \mid id$
$F \to num \mid id$

$X \to +$
$Y \to *$

$E \to EXT \mid TYF \mid num \mid id$
$T \to TYF \mid num \mid id$
$F \to num \mid id$

$Z \to EX$
$D \to TY$

$E \to ZT \mid DF \mid num \mid id$
$T \to DF \mid num \mid id$
$F \to num \mid id$

* this q can also be done using removing left recursion

$A \to A\alpha_i \mid \beta_i$

$\Rightarrow A \to \beta_i A' \mid \beta_i$
$A' \to \alpha_i A' \mid \alpha_i$

q. $E \to E+T \mid T$
$T \to T*F \mid T$
$F \to num \mid id$

$E \to \overbrace{E+T}^{\alpha} \mid \overbrace{T}^{\beta}$
$E \to TE' \mid T$
$E' \to +TE' \mid +T$

$T \to T * F \mid F$

$T \to F T' \mid F$

$T' \to * F T' \mid * T$

$F \to num \mid id$

$E \to T E' \mid F T' \mid num \mid id$

$E' \to D_1 E' \mid X T$

$T \to F T' \mid num \mid id$

$T' \to D_2 T' \mid Y F$

$F \to num \mid id$

$X \to +$

$Y \to *$

$D_1 \to X T$

$D_2 \to Y F$

→ Greibach Normal Form

$A \to a\alpha$

$\alpha \in V^*$ non- terminals

Algorithm:

1. Simplify Grammar

2. Convert to GNF
   i) replace every Terminal except the first one with NT
   ii) replace first NT to Terminal

Convert CNF to GNF

q1. $S \to abSb \mid aa$

Sol $A \to a$

$B \to b$

$S \to aBSB \mid aA$

10th October, 2023

q2. $S \to aSa \mid bSb \mid SS \mid \lambda$

Step 1. Simplify Grammar

(following stricter version no λ production)

$S \to aSa \mid aa \mid bSb \mid bb \mid SS$

$S \to aSa \mid aa \mid bSb \mid bb \mid aSaS \mid bSbS \mid aaS \mid bbS$

$A \to a$

$B \to b$

$S \to aSA \mid aA \mid bSB \mid bB \mid aSAS \mid bSBS \mid aAS \mid bBS$

q3. $S \to XY \mid Xn \mid p$

$X \to mX \mid m$

$Y \to Wn \mid o$

W is useless as it doesnt get terminated

$Y \to o$

$Z \to n$

$S \to XY \mid XZ \mid p$

$X \to mX \mid m$

$Y \to o$

$Z \to n$

$m X Y / m Y / m X Z / m Z / p$

$S \rightarrow (mX/m)Y \mid (mX/m)Z / p$

ans: $X \rightarrow mX / m$

$Y \rightarrow o$

$Z \rightarrow n$

q4. $S \rightarrow aA \mid bB$

$B \rightarrow bB \mid \lambda$

$A \rightarrow aA \mid \lambda$

remove λ

$S \rightarrow aA \mid bB \mid a \mid b$

$B \rightarrow bB \mid b$

$A \rightarrow aA \mid a$

→ Convert CFG to PDA

Step 1: Convert CFG to GNF

Step 2: Convert GNF to PDA (a, b & c same for every gram)

a) Push start symbol into stack without consuming i/p.

$\delta(q_0, \lambda, Z_0) = (q_1, SZ_0)$

b) Convert all productions of form $A \rightarrow a\alpha$ to transitions as

$\delta(q_1, a, A) = (q_1, \alpha)$

c) Acceptance

$\delta(q_1, \lambda, Z_0) = (q_f, Z_0)$

Convert CFG to PDA

Q1:

$S \to aABC$
$A \to aB \mid a$
$B \to bA \mid b$
$C \to a$

Step 1: Grammar is in GNF

Step 2: Convert to PDA

a) $\delta(q_0, \lambda, Z_0) = (q_1, SZ_0)$

b) $A \to a\alpha$	$\delta(q_1, a, A) = (q_1, \alpha)$
~~$S \to aABC$~~	~~$\delta(q_1, a, A) = (q_1, \alpha)$~~
$S \to aABC$	$\delta(q_1, a, S) = (q_1, ABC)$
$A \to aB$	$\delta(q_1, a, A) = (q_1, B)$
$A \to a$	$\delta(q_1, a, A) = (q_1, \lambda)$
$B \to bA$	$\delta(q_1, b, B) = (q_1, A)$
$B \to b$	$\delta(q_1, b, B) = (q_1, \lambda)$
$C \to a$	$\delta(q_1, a, C) = (q_1, \lambda)$

c) Acceptance

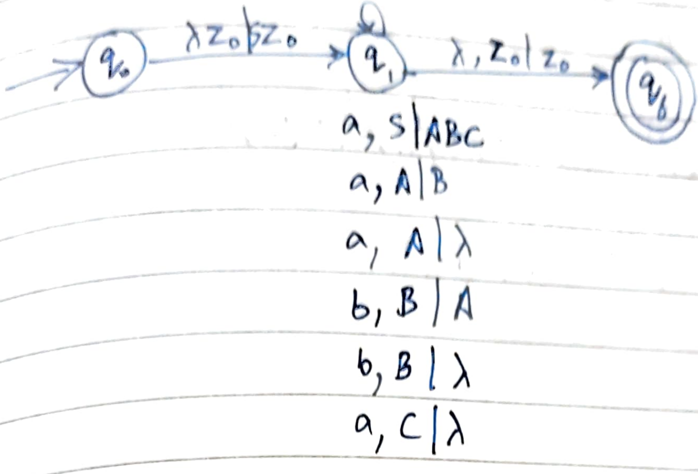
$\delta(q_1, \lambda, Z_0) = (q_f, Z_0)$

Instantaneous Description

$$(q_0, aaba, Z_0) \vdash (q_1, aaba, SZ_0)$$
$$\vdash (q_1, aba, ABCZ_0)$$
$$\vdash (q_1, ba, BCZ_0)$$
$$\vdash (q_1, a, CZ_0)$$
$$\vdash (q_1, \lambda, Z_0) \quad (q_f, Z_0)$$
$$\vdash (q_f, \lambda, Z_0) \text{ accepted}$$

q2. ~~S → aS+~~

PDA:



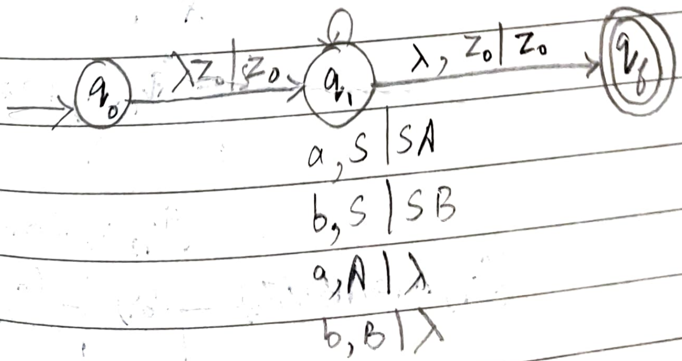
q2. $S \rightarrow aSA \mid bSB$

$A \rightarrow a$

$B \rightarrow b$

$A \rightarrow a\alpha$	$\delta(q_1, a, A) = (q_1, \alpha)$
$S \rightarrow aSA$	$\delta(q_1, a, S) = (q_1, SA)$
$S \rightarrow bSB$	$\delta(q_1, b, S) = (q_1, SB)$
$A \rightarrow a$	$\delta(q_1, a, A) = (q_1, \lambda)$
$B \rightarrow b$	$\delta(q_1, b, B) = (q_1, \lambda)$

$$\delta(q_1, \lambda, Z_0) = (q_f, Z_0)$$



11th October, 2023

q3 $S \to aSA \mid bSB \mid \lambda$

$A \to a$

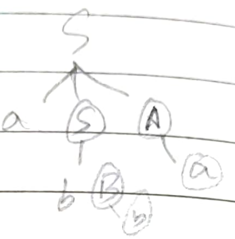
$B \to b$

$\cancel{S \to aSA \mid bSB \mid aA \mid aA \mid a \mid bB \mid b}$

$S \to aSA \mid aA \mid bSB \mid bB \mid \lambda$

$A \to a$

$B \to b$



$A \to a\alpha$ $\quad \delta(q, a, A) = (q, SZ_0)$

$\delta(q_0, \lambda, Z_0) = (q_1, \alpha)$

$S \to aSA$ $\quad \delta(q_1, a, S) = (q_1, SA)$

$S \to aA$ $\quad \delta(q_1, a, S) = (q_1, A)$

$S \to bSB$ $\quad \delta(q_1, b, S) = (q_1, SB)$

$S \to bB$ $\quad \delta(q_1, b, S) = (q_1, B)$

$A \to a$ $\quad \delta(q_1, a, A) = (q_1, \lambda)$

$B \to b$ $\quad \delta(q_1, \lambda, S) = (q_1, \lambda)$ $\quad \delta(q_1, b, B) = (q_1, \lambda)$

$S \to \lambda$ $\quad \delta(q_1, \lambda, Z_0) = (q_f, Z_0)$

→ check whether aabb is accepted / rejected.

$(q_0, aaba, Z_0) \vdash (q_1, aaba, SZ_0)$

$\vdash (q_1, aba, SAZ_0)$

$(q_1, a, S) = (q_1, SA)$ $\quad \vdash (q_1, ba, AAZ_0)$ X $\quad \vdash (q_1, ba, SAAZ_0)$

$(q_1, a, S) = (q_1, A)$

$\vdash (q_1, a, BAAZ_0)$ X

$(q_1, b, S) = (q_1, SB)$

$(q_1, b, S) = (q_1, B)$ $\quad (q_1, a, SBAAZ_0)$

$(q_1, \lambda, SABAAZ_0) \vdash (\delta) \vdash (q_1, \lambda, ABAA\ldots$

∴ Rejected

CYK algorithm

q.1. $S \to aSb \mid bSa \mid SS \mid \lambda$

Parse string abba using CYK algorithm

Step 1: Convert grammar to CNF

$S \to AB \mid AC \mid BA \mid BD \mid SS$

$C \to SB$

$D \to SA$

$A \to a$

$B \to b$

Step 2:

S			
C	ϕ		
S	ϕ	S	
A	B	B	A
a	b	b	a

a) Strings of length 1 are generated by $A \to a$, $B \to b$

b) Strings of length 2 $S \to AB$

$\phi \to BB$

$S \to BA$

c) Strings of length 3 by drawing one arrow from down to top and from top's adjacent to diagonal

the concatenated by taking union (of starting points of ending points)

i) $(A \cdot \phi) \cup (S \cdot B)$ $C \to S \cdot B$

ii) $(B \cdot S) \cup (\phi \cdot A)$ ϕ

d) Strings of length 4 we again draw arrows

$(A \cdot \phi) \cup SS \cup CA$

$\phi \cup S \cup \phi$

S

Step 3: Since top-most symbol is S

$\Rightarrow$ string belongs to grammar.

Q2. $S \to AB$

$A \to BB \mid a$

$B \to AB \mid b$

Parse string aabba using CYK

Sol. Grammar already in CNF

ϕ				
A	ϕ			
S,B	A	ϕ		
ϕ	S,B	A	ϕ	
A	A	B	B	A
a	a	b	b	a

$A \cdot (S \cup B) \cup \phi \cdot B$

$AS \cup AB \cup \phi$

$AA \cup (S, B) \cdot B$

$AA \cup SB \cup BB$

$B\phi \cup AA$

ϕ

$\therefore$ rejected

q3. $S \to AA \mid BC$
$A \to BA \mid a$
$B \to CC \mid b$
$C \to AB \mid a$

string : baaa

S, A			
S	C, A, S		
A, S	S, B	S, B	
B	A, C	A, C	A, C
b	a	a	a

$B \cdot (S \cup B) \cup (A \cup S) \cdot (A \cup C)$
$BS \cup BB \cup \underset{S}{AA} \cup AC \cup SA \cup SC$

$(A \cup C) \cdot (S \cup B) \cup (S \cup B) \cdot (A \cup C)$
$AS \cup \underset{C}{AB} \cup CS \cup CB \cup SA \cup SC \cup \underset{A}{BA} \cup \underset{S}{BC}$

$B \cdot (C \cup A \cup S) \cup S \cdot (A \cup C)$
$\underset{S}{BC} \cup \underset{A}{BA} \cup BS \cup SA \cup SC$

Since S is at top $\Rightarrow$ accepted

q4. $S \to aSa \mid bSb \mid A \mid \lambda$
$A \to a \mid b \mid \lambda$

string : aabaa

$S \to aSa \mid bSb \mid A \mid aa \mid bb$
$A \to a \mid b$

$C \to a \quad D \to b$
$S \to CSC \mid DSD \mid a \mid b \mid CC \mid DD$

$F \rightarrow CS$

$G \rightarrow DS$

$S \rightarrow FC \mid GD \mid a \mid b \mid CC \mid DD$

$C \rightarrow a$

$D \rightarrow b$

S				
F	ϕ			
ϕ	S	G		
F, S	F	G	F, S	
S, C	S, C	S, D	S, C	S, C
a	a	b	a	a

2q/p (SUC).F U (FUS)·(SUD)

SFUCF UFSUFDUSSUSD

(SUC)·G U F·(SUC)

SGUCG U FS UFC
S

(SUD)·(FUS) U G·(SUC)

SFUSSUDFUDS U GS U GC
G

3q/4 (SUC).S U ϕ U (FUS)·G

SSUCS UFGUSG
F

(SUC)·G U S·(SUC) U F·(FUS)

ϕ

∴ string accepted

Q. Write a grammar for the language equal no. of a's, equal no. of b's followed by any no. of c union

$\{a^n b^n c^m \mid n >= 0, m \geq 1\} \cup \{a^n b^m c^m \mid n >= 1, m \geq 0\}$

$S \to S_1 \mid S_2$

$S_1 \to Ac$ $\qquad$ $S_2 \to aB$

$A \to aAb \mid \lambda$ $\qquad$ $B \to bBc \mid \lambda$

13th October, 2023

Properties

* DCFL are not closed

super-set

CFL is closed under union → PDA | CFG

PDA | CFG

$L_1 \to$ CFL $\{a^n b^n \mid n \geq 0\}$

$L_2 \to$ CFL $\{c^m d^m \mid m \geq 0\}$

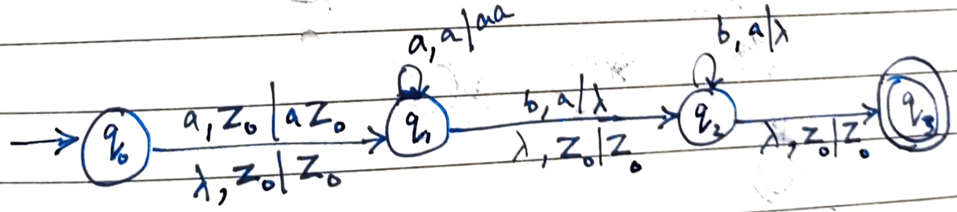
$S \to S_1 \mid S_2$

$S_1 \to a S_1 b \mid \lambda$

$S_2 \to c S_2 d \mid \lambda$

} here ambiguity from S to goto S_1 or S_2

* If λ is there in NDFA then it is Non-deterministic



DPDA

closed

closed

closed

* Concatenation $S \to S_1 S_2$
* Kleene's star $S \to S_1 S \mid \lambda$
* Reversal

* CFLs and DCFLs are not closed under intersection

$L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$ } $\cap$
$L_2 = \{a^n b^m c^m \mid n, m \geq 1\}$

$L = \{a^n b^n c^n \mid n \geq 1\}$

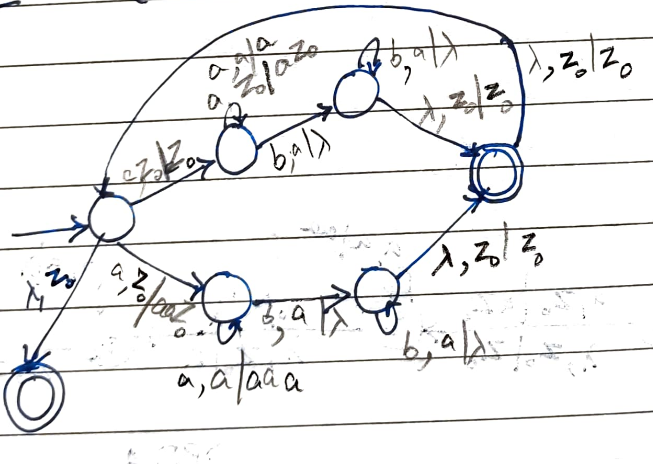
with 1 memory we cannot keep a track of the order.

* DCFLs are not closed under union

$L_1 = \{c\,a^n b^n \mid n \geq 0\}$
$L_2 = \{a^m b^{2m} \mid m \geq 1\}$

$L = L_1 \cup L_2$

$L^* = (c\,a^n b^n \cup a^m b^{2m})^*$



$L = w c w^R (a+b+c)^*$ $\quad w \in (a, b)^+$

aca aa | bb | cc | abc | bac | cab | λ
bcb
abcba
bacab

$L^R = (a+b+c)^* \, w c w^R$

aaa c a
ambiguous

q. $L = \{ w w^R , \; w \in (a,b)^* \}$

even length

L^c

odd length
Palindrome

PDA is not possible for this. We will need 2 stack

7th October, 2023

q1. Design a CNF for balanced parenthesis and parse string using CYK : (())

q2. $S \to aAa \mid bBb \mid BB$

$A \to C$

$C \to S \mid \lambda$

$B \to S \mid A$

parse string: aabbaa using CYK

q3. $S \to X \mid XYa \mid XbX$

$X \to Xa \mid \lambda$

$Y \to Yb \mid YZ$

$Z \to ZY \mid ZX \mid bY$

string : aabaaa : CYK

Sol. 1. Design CFG, convert CFG to CNF

λ

CFG : $S \to (S) \mid SS \mid \lambda$

()

(())

unit product

()()

CNF : $S \to (S) \mid () \mid SS \mid S \mid \lambda$

$S \to (S) \mid () \mid SS \mid \lambda$ Simplified grammar

$A \to ($

$B \to)$

$S \to ASB \mid AB \mid SS \mid \lambda$

$X \to AS$

CNF : $S \to XB \mid AB \mid SS \mid \lambda$

$X \to AS$

$A \to ($

$B \to)$

Parsing (( ))

4	S			
3	X	ϕ		
2	ϕ	S	ϕ	
1	A	A	B	B
	(	(	)	)

q2: $S \to aAa \mid bBb \mid BB$

$A \to C$

$C \to S \mid \lambda$

$B \to S \mid A$

all symbols are nullable

$S \to aAa$

simplified grammar : $S \to aAa \mid aa \mid bBb \mid bb \mid BB \mid \lambda$
$A \to aAa \mid aa \mid bBb \mid bb \mid BB \mid \lambda$
$B \to aAa \mid aa \mid bBb \mid bb \mid BB$

$X \to a$
$Y \to b$
$S \to XAX \mid XX \mid YBY \mid YY \mid BB \mid \lambda$
$A \to XAX \mid XX \mid YBY \mid YY \mid BB$
$B \to XAX \mid XX \mid YBY \mid YY \mid BB$

$L \to XA$
$M \to YB$
$X \to a$
$Y \to b$
$S \to LX \mid XX \mid MY \mid YY \mid BB \mid \lambda$
$A \to LX \mid XX \mid MY \mid YY \mid BB$
$B \to LX \mid XX \mid MY \mid YY \mid BB$

string : aabbaa

S,A,B					
φ	L				
A,S,B	S	A,S,B			
φ	L	φ	M		
SAB	φ	SAB	φ	SAB	
X	X	Y	Y	X	X
a	a	b	b	a	a

q3. $S \to X \mid XYa \mid XbX$

$X \to Xa \mid X$

$Y \to Yb \mid YZ$

$Z \to ZY \mid ZX \mid bY$

$S \to X \mid XYa \mid XbX$

$X \to Xa \mid a$

$Y \to Yb \mid Y_z$

$Z \to ZY \mid ZX \mid bY$

$S \to Xa \mid a \mid XYa \mid XbX$

$X \to Xa \mid a$

$Y \to Yb \mid Y_z$

$Z \to ZY \mid ZX \mid bY$

$\to$

$A \to a$

$B \to b$

$S \to XA \mid a \mid XYA \mid XBX$

$X \to XA \mid a$

$Y \to YB \mid YZ$

$Z \to ZY \mid ZX \mid BY$

$S \to XA \mid a \mid LA \mid MX$

$X \to XA \mid a$

$Y \to YB \mid YZ$

$Z \to ZY \mid ZX \mid BY$

$A \to a$

$B \to b$

$L \to XY$

$M \to XB$

string : aabaaa ✓

S					
∅	S				
S	S	∅			
M	S	∅	S, X		
SX	M	∅	SX	SX	
ASX	ASX	B	ASX	ASX	ASX
a	a	b	a	a	a

15th October, 2023

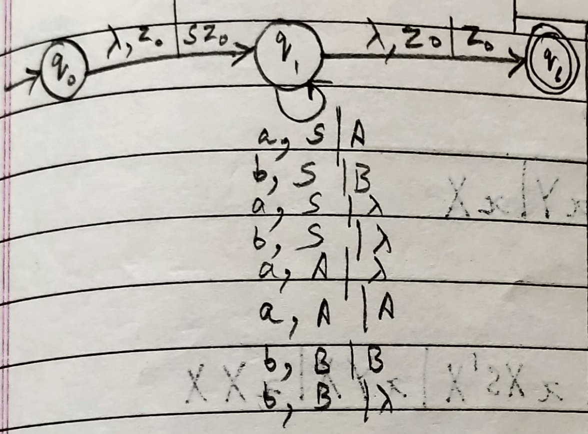
q. Convert to GNF, then draw PDA

i)
$S \to aA \mid bB$
$A \to aA \mid \lambda$
$B \to bB \mid \lambda$

ii)
$S \to XY1 \mid 0$
$X \to 00X \mid Y$
$Y \to 1X1 \mid \lambda$

sol i) remove λ

$S \to aA \mid bB \mid a \mid b$
$A \to aA \mid a$
$B \to bB \mid b$



ii) remove λ

$S \to XY1 \mid 0 \mid Y1 \mid X1 \mid 1$
$X \to 00X \mid Y \mid 00$
$Y \to 1X1 \mid 11$

substitute all terminals with non except first

$S \to XY1 \mid 0 \mid X1 \mid Y1 \mid 1$
$X \to 00X \mid 00 \mid 1X1 \mid 11$
$Y \to 1X1 \mid 11$

now $A \to 0$
$B \to 1$

$S \to XYB \mid 0 \mid XB \mid YB \mid 1$
$X \to 0AX \mid 0A \mid 1XB \mid 1B$
$Y \to 1XB \mid 1B$

$A \to 0 \quad B \to 1$
$S \to 0AXYB \mid 0AYB \mid 1XBYB \mid 1BYB \mid$
$0 \mid 0AXB \mid 0AB \mid 1XBB \mid 1BB \mid$
$1XBB \mid 1BB \mid 1$

remove duplicate

$X \to 0AX \mid 0A \mid 1XB \mid 1B$
$Y \to 1XB \mid 1B$

iii) $S \to AY \mid XX$
$X \to x \mid SX$
$Y \to y$
$A \to x$

$S \to \overset{\beta_1}{xY} \mid \overset{\beta_2}{xX} \mid S\overset{\alpha}{XX}$ (left recursion)
$X \to x \mid SX$
$Y \to y$
$A \to x$ (unreachable)

$A \to A\alpha \mid \beta$
$A \to \beta A' \mid \beta$
$A' \to \alpha A' \mid \alpha$

$S \to xYS' \mid xXS' \mid xY \mid xX$
$S' \to XXS' \mid XX$
$Y \to y$
$X \to x \mid SX$

$S \to xYS' \mid xXS' \mid xY \mid xX$
$S' \to XXS' \mid XX$
$Y \to y$
$X \to x \mid xYS'X \mid xXS'X \mid xYX \mid xXX$

* Homomorphism is closed under CFL

Pumping Lemma

RL	CFL $\Rightarrow$ CNF
$w = uv^iw$	$z = uv^iwx^iy \quad i \geq 0$
$S \to aA$, $A \to aA$ } only 1 loop	tree grows equally from both sides
	2 loops
$w \geq n$	$z \geq n \to$ no. of N.T
$\lvert uv \rvert \leq n$	$vwx \leq n$
$\lvert v \rvert \geq 1$	$\lvert vx \rvert \geq 1$

Q1. Prove $L = \{a^n b^n c^n \mid n \geq 0\}$ is not CFL

1) Assume lang is CFL

2) Assume #NT = N & pumping constant $n = 2^{N+1}$

3) Select string $|z| \geq n$

$$z = \underbrace{a \cdots a}_{n} \underbrace{b \cdots b}_{n} \underbrace{c \cdots c}_{n} = 3n$$

$z = uvwxy$ such that $|vwx| \leq n$ & $|vx| \geq 1$

case	location of vwx	i	n_a	n_b	n_c	Remark
1.	vwx within a^n	≥ 2	more	-	-	string $\notin L$
2.	vwx within b^n	≥ 2	-	more	-	"
3.	vwx within c^n	≥ 2	-	-	more	"
4.	vwx within $a^n b^n$	≥ 2	more	more	-	"
5.	vwx within $b^n c^n$ somepart of b^n, somepart of c^n	≥ 2	-	more	more	"

$\therefore$ string $\notin L$ $\therefore$ assumption wrong

2. Prove $L = \{ww \mid w \in (a,b)^*\}$ is not CFL

1)

2)

3) $z = a^n b^n a^n b^n = 4n \geq n$

4) $z = uvwxy$ such that $|vwx| \leq n$ & $|vx| \geq 1$

case	location of vwx	i	n_a	n_b	n_a	n_b
1.	vwx 1st a^n	≥ 2	more	–	–	–
2.	vwx 1st b^n	≥ 2	–	more	–	–
3.	vwx 2nd a^n	≥ 2	–	–	more	–
4.	vwx 2nd b^n	≥ 2	–	–	–	more
5.	vwx somepart $a^n b^n$	≥ 2	more	more	–	–
6.	vwx somepart $b^n a^n$	≥ 2	–	more	more	–
7.	vwx 2nd somepart $a^n b^n$	≥ 2	–	–	more	more

q3. $L = \{a^n b^m c^k \mid k = n \times m, \; n, m \geq 1\}$

$a^n b^m c^{n \times m}$ min: abc

let n = pumping constant

$a^n b^n c^{n \times n}$ $|z| \geq n$ $(n^2 + 2n)$

$z = uvxy$

$|vwx| < n$ & $|vx| \geq 1$

case	location	i	n_a	n_b	n_c	remark
1.	vwx within a^n	≥ 2	more	–	–	Not CFL
2.	" b^n	≥ 2	–	more	–	
3.	" $c^{n \times n}$	≥ 2	–	–	more	

≥ 2

}

19th October, 2023

Properties of CFL

Closure Properties

* CFLs are closed under: union, star, concatenation, reversal, homomorphism.

* CFLs are not closed under intersection, complement, difference

* DCFL is closed under complement

* DCFL is not closed under union, star, concatenation, reversal.

20th October, 2023

Q4. $a^{n!} \mid n \geq 0$

Assume L is context-free.
Let n be pumping constant
Let
$$n = 2^{N+1}$$
$$Z = a^{n!}$$
$$Z = uvwxy \quad , |vwx| \leq n \, , |vx| \geq 1$$

$Z = a^{n!-\beta} \cdot a^{\beta}$ where $v = a^{\beta}$, $w = \lambda$, $x = \lambda$
and $|a^{\beta}| \geq 1$ and $y = \lambda$

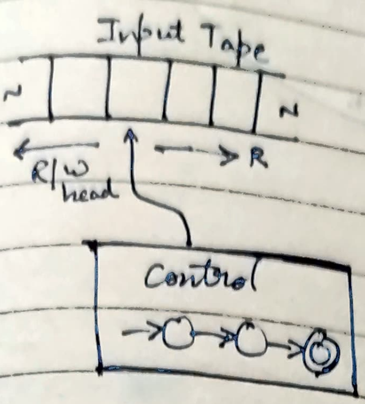
$$Z = uv^i w x^i y$$
$$Z = a^{n!-\beta} (a^{\beta})^i$$
$$= a^{n! - \beta + \beta i} \qquad \text{for } i = 2$$
$$= a^{n! + \beta}$$

$$n! \leq n! + \beta \qquad |\beta| \geq 1$$
$$(n! + \beta) \leq (n! + n)$$
$$n! < (n! + \beta) < (n! + n)$$
$$Z \notin L$$

Q5. $L = 0^p$, where p is prime

$$Z = 0^p$$
$$Z = uvwxy$$
$Z = 0^{p-\beta} \cdot 0^{\beta}$ $\quad w = \lambda \quad x = \lambda \quad y = \lambda$
$Z = uv^i w x^i y$ $\quad |0^{\beta}| \geq 1$
$$Z = 0^{p-\beta}(0^{\beta})^i$$
$$Z = 0^{p+\beta} \qquad i = 2$$
$$Z = 0^{p+2\beta} \qquad i = 3$$
$$p < p + \beta$$
$$p + \beta < p + n$$
$p < p+\beta < p + n$ [if p is prime no. $p + \beta$ may not be prime]
$$Z \notin L$$

Turing Machine



Transition function : $\delta(q_0, a) = (q_1, X, R)$

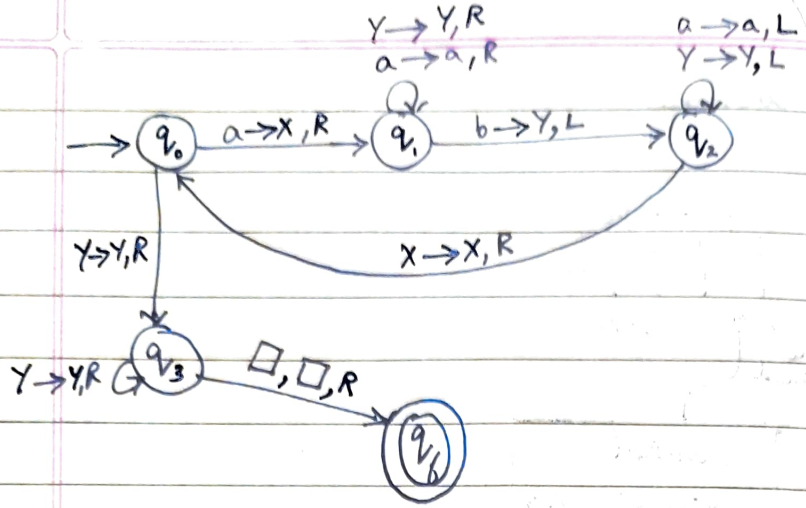
FA : $\delta(q_0, a) = q_1$

PDA : $\delta(q_0, a, Z_0) = (q_1, aZ_0)$

26 October, 2023

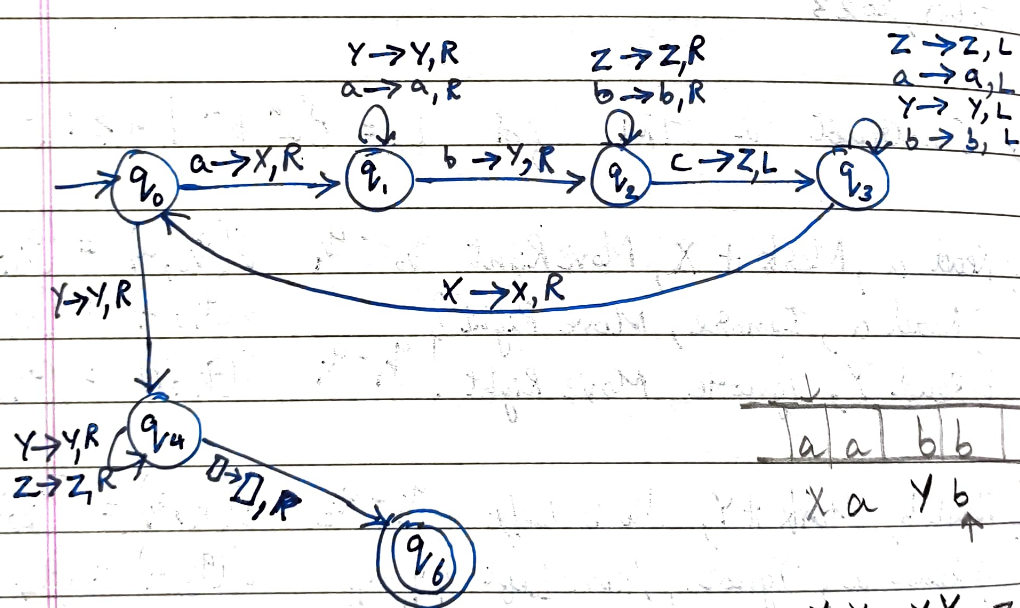
✓ Construct a TM for $L = \{a^n b^n \mid n \geq 1\}$

1) Read a, Mark it X, Move Right	$q_0 - q_1$	$\delta(q_0, a) = (q_1, X, R)$
a) Read a, Ignore, Move Right	$q_1 - q_1$	$\delta(q_1, a) = (q_1, a, R)$
b) Read Y, Ignore, Move Right		$\delta(q_1, Y) = (q_1, Y, R)$
2) Read b, Mark it Y, Move Left	$q_1 - q_2$	$\delta(q_1, b) = (q_2, Y, L)$
a) Read a, Ignore, Move Left	$q_2 - q_2$	$\delta(q_2, a) = (q_2, a, L)$
b) Read Y, Ignore, Move Left		$\delta(q_2, Y) = (q_2, Y, L)$
3) Read X, Ignore, Move Right	$q_2 - q_0$	$\delta(q_2, X) = (q_0, X, R)$
a) Read a => 1)	$q_0 - q_1$	
b) Read Y, Ignore, Move Right	$q_0 - q_3$	$\delta(q_0, Y) = (q_3, Y, R)$
Read Y, Ignore, Move Right	$q_3 - q_3$	$\delta(q_3, Y) = (q_3, Y, R)$
4) Read Blank, Ignore, Move Right & accept it	$q_0 - q_f$	$\delta(q_3, \square) = (q_f, B, R)$



Construct a TM for $L = \{a^n b^n c^n \mid n \geq 1\}$

aaabbbccc



a	a	b	b	c	c

X a Y b Z c

XX YY ZZ

XX YY ZZ

October, 2023

$L = \{ww \mid w \in (a,b)^*\}$

i) Identify string's mid point, even length

ii) Check first is equal to second half

i)

a	a	b	a	a	b	B

blank

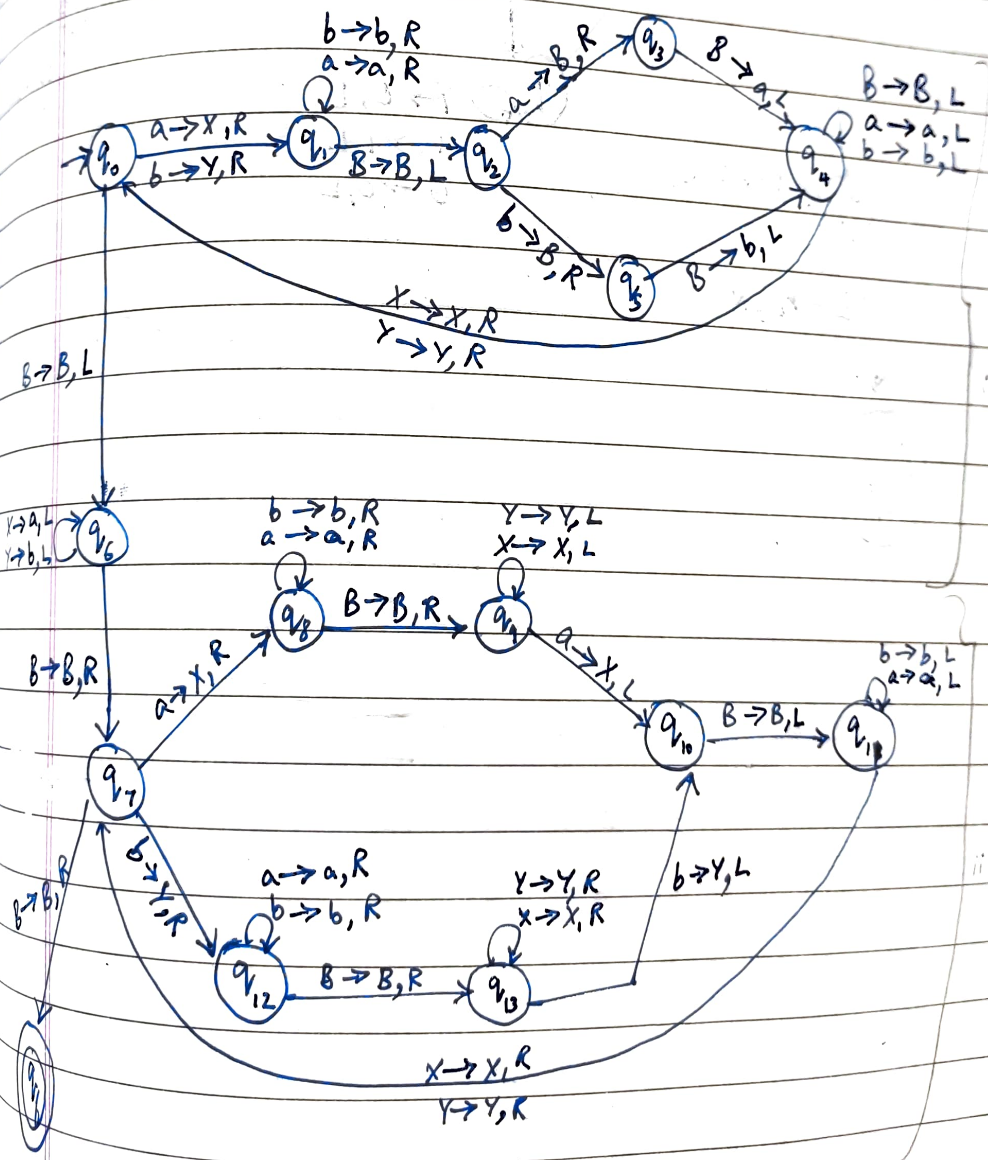
ii)

a	a	b	B	a	a	b

X	a	b	a	a	b	B

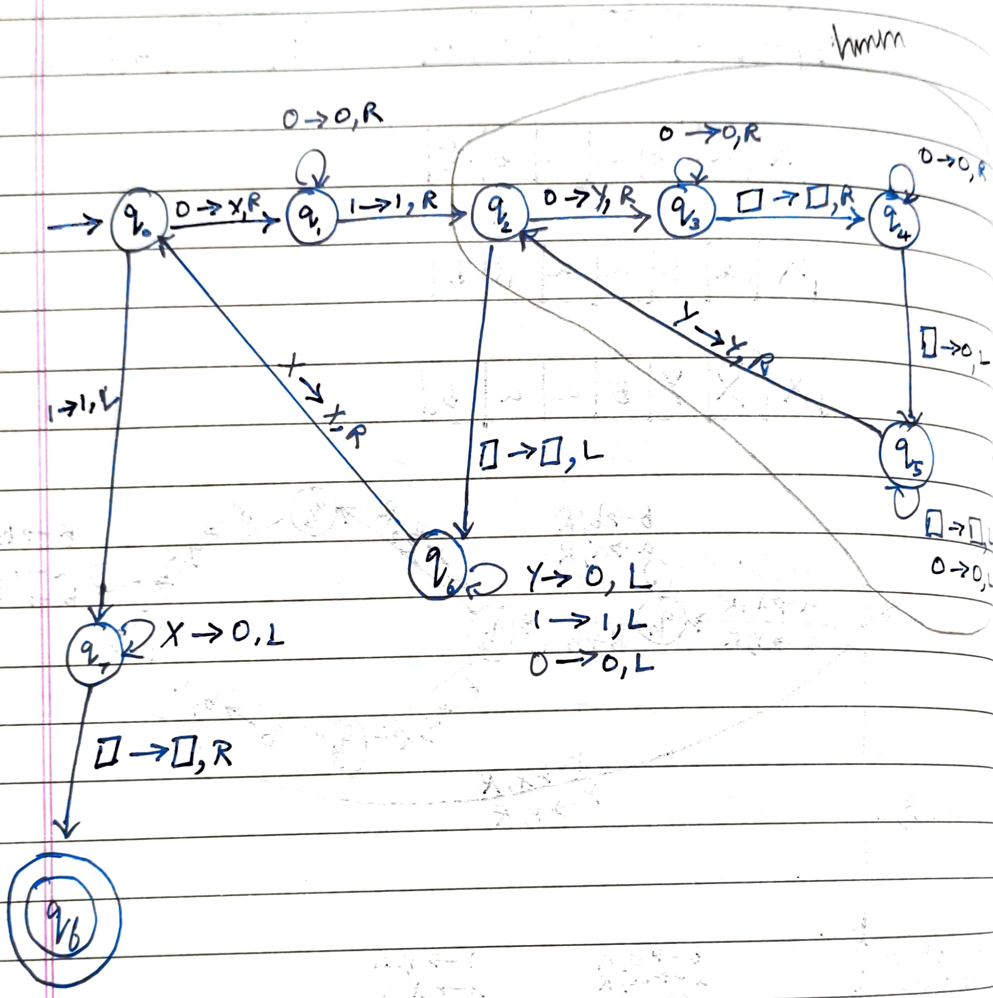
X	a	b	a	a	B	b

X	X	Y	B	a	a	b

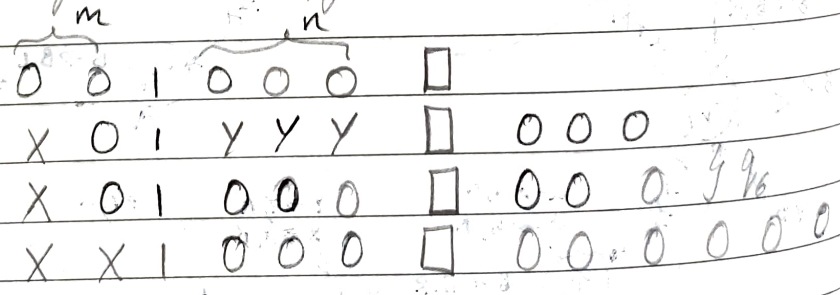


4. Language of Multiplication

input : $0^m 1 0^n$
output : $0^m 1 0^n \square 0^{m \times n}$



idea : for every m 0 copy n 0s to o/p



5. $L = \{a^n b^m c^{m+n} \mid n, m \geq 0\}$

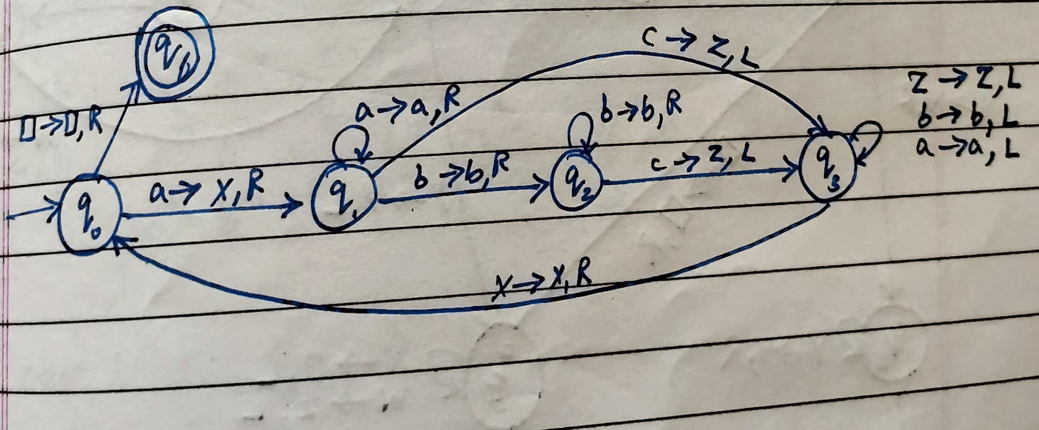
aabccc
abbccc
aacc
bbcc

after reading as change state as we have as followed by bs. All as are together

aab ccc
XXbZZc
XXYZZZ

001000 □
X01Y00 □ 0
X01YY0 □ 00
X01YYY □ 000 } hmm
X01000 □ 000 } q_6
XX1YYY □ 000000
XX1000 □ 000000
001000 □ 000000

$q_0 \to q_7$
$q_1 \to q_3$ ac



3rd November, 2023

6. $L = \{0^n 1^{n^2}, n \geq 1\}$

Z - iteration   W, Y - replacement

□ 0 0 1 1 1 1

□ Z 0 1 1 1 1   for every 0 mark 1 set of 1

W 0 Y 1 1 1

W X Y Y 1 1

W X Y Y 1 1   000 111 111 111

W X Y Y 1 1   Z

Z 0 Y Y 1 1   W 0 0 Y

Z Z Y Y 1 1   W X 0 Y Y

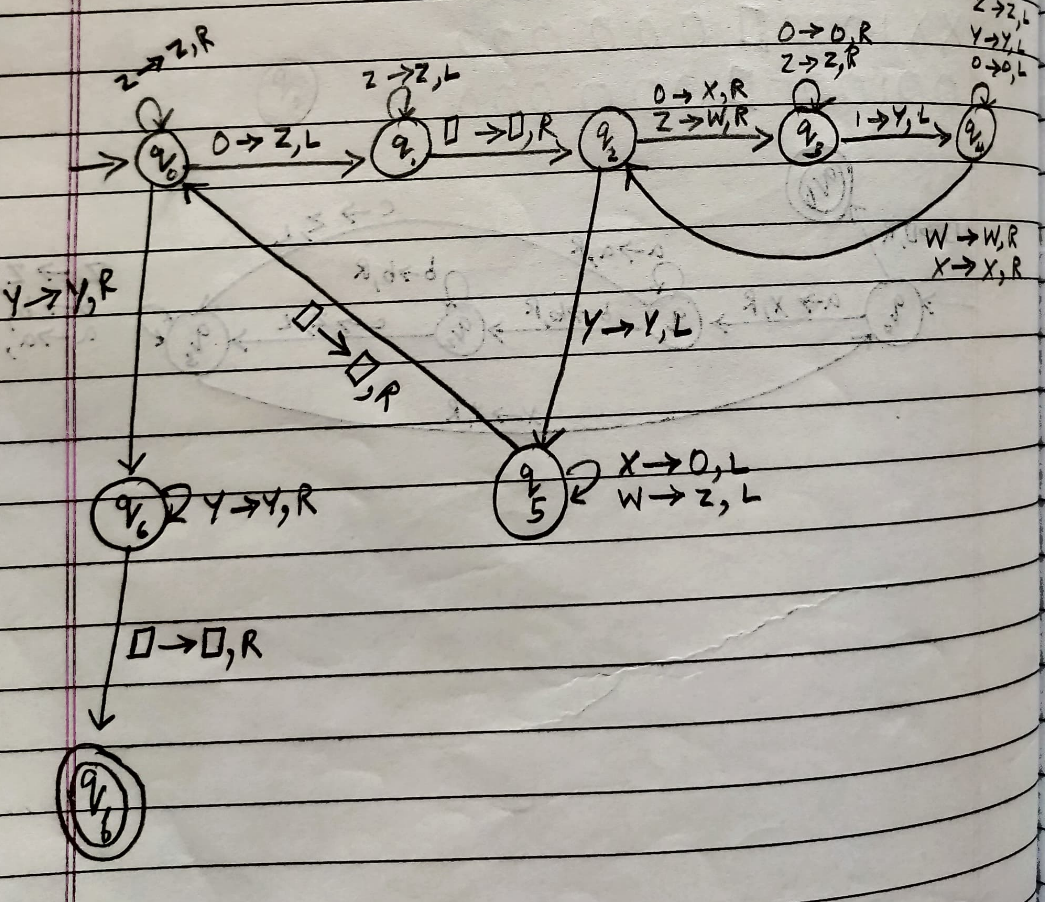
W Z Y Y Y 1   W X X Y Y Y

W W Y Y Y Y

Z 0 0 0

Z Z

$q_2 - q_5$: 1 iteration



$L = \{0^{2^n} ; n \geq 0\}$

$L = \{0, 00, 0000, 0000\ 0000, \ldots\}$

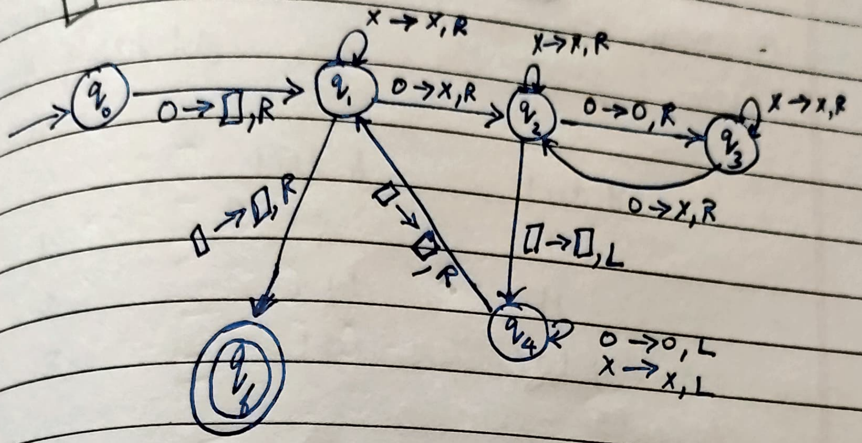
$\square \times 0 \times 0 \times 0 \times$

0 0 0 0

B X 0 X

$\square$ X X X

alternate 0s



Language of subtraction $= \{a^n b^m c^k, k = n-m, n,m \geq 1\}$

ip : $a^n b^m c^k$

op : $a^n b^m c^k \square c^k$ if k is +ve

$a^n b^m c^k \square -c^k$ if k is -ve

case 1 : as > bs

case 2 : as < bs

case 3 : as = bs

7th November, 2023

Post Correspondence Problem (PCP) - Undecidable

eg1.

	List A	List B
1.	10	10110
2.	111	01
3.	110110	1010

no possibility

eg2.

	List A	List B
1.	10	101
2.	011	11
3.	101	011

1 3 1

10 101

101 011

Machine will enter into infinite loop

101011

101011 101

7th November, 2023

UNIT - 4

Formal Languages and Logic

Programming Languages

Declarative Lang (Logic Programming)
eg: Prolog, SQL

Other Programming Paradigms
eg: C++, Java

*

* one parameter - function
* two parameters - relation

8th November, 2023

Propositional Logic

complex statement $(p \to q) \wedge (q \to p)$

i)

p	q	$p \to q$	$q \to p$	$(p \to q) \wedge (q \to p)$	$p \leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

$$(p \to q) \wedge (q \to p) \equiv p \leftrightarrow q$$

ii) $p \to q \equiv \neg p \vee q$

p	q	$\neg p$	$p \to q$	$\neg p \vee q$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

$\lor$ or

Q. Check if $(p \to q) \leftrightarrow (\neg p \lor q)$ is tautology

p	q	$\neg p$	$p \to q$	$\neg p \lor q$	$(p \to q) \leftrightarrow (\neg p \lor q)$
T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

→ Laws

Equivalence	Identity
$p \land T \equiv p$ $p \lor F \equiv p$	Identity Law
$p \land F \equiv F$ $p \lor T \equiv T$	Domination Law
$p \land p \equiv p$ $p \lor p \equiv p$	Idempotent Law
$\neg(\neg p) \equiv p$	Double Negation Law
$p \land q = q \land p$ $p \lor q = q \lor p$	Commutative Law
$(p \land q) \land r \equiv p \land (q \land r)$ $(p \lor q) \lor r \equiv p \lor (q \lor r)$	Associative Law
$p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$ $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$	Distributive Law

$\neg(p \wedge q) \equiv \neg p \vee \neg q$ De Morgan's Laws
$\neg(p \vee q) \equiv \neg p \wedge \neg q$

$p \wedge (p \vee q) \equiv p$ Absorption Laws
$p \vee (p \wedge q) \equiv p$

$p \wedge \neg p \equiv F$ Negation Laws
$p \vee \neg p \equiv T$

q. 1. PT $\neg(p \rightarrow q) \equiv p \wedge \neg q$

LHS $\equiv \neg(p \rightarrow q)$ $\because p \rightarrow q \equiv \neg p \vee q$
$\equiv \neg(\neg p \vee q)$ De Morgan's Law
$\equiv \neg(\neg p) \wedge \neg q$ Double Negation Law
$\equiv p \wedge \neg q$
$\equiv$ RHS

2. PT $p \wedge (p \rightarrow q) \equiv p \wedge q$

LHS $\equiv p \wedge (\neg p \vee q)$
$\equiv (p \wedge \neg p) \vee (p \wedge q)$
$\equiv F \vee (p \wedge q)$
$\equiv (p \wedge q)$
$\equiv$ RHS

th November, 2023

q. PT the foll expressions are tautologies using laws of logic

1. $(p \wedge q) \longrightarrow (p \vee q)$

write the laws used

$\neg(p \wedge q) \vee (p \vee q)$
$(\neg p \vee \neg q) \vee (p \vee q)$ Demorgan's Law
$\neg p \vee \neg q \vee p \vee q$
T

2. $(p \wedge q) \rightarrow (p \rightarrow q)$

$\neg(p \wedge q) \vee (p \rightarrow q)$

$(\neg p \vee \neg q) \vee (\neg p \vee q)$

$\neg p \vee \neg q \vee \neg p \vee q$

T

* left hand side = premise
* right hand side = conclusion

q. Use Resolution Refutation

1. **Premises**

1) $P \vee Q$
2) $P \rightarrow R$
3) $Q \rightarrow R$

Conclusion

R

implication $((P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R)) \rightarrow R$

2. $P \rightarrow R \equiv \neg P \vee R$

3. $Q \rightarrow R \equiv \neg Q \vee R$

Premises	
1) $P \vee Q$	Use 1&2 to resolve P
2) $\neg P \vee R$	$\cancel{P} \vee Q$
3) $\neg Q \vee R$	$\cancel{\neg P} \vee R$
4) $\neg R$	$Q \vee R$

Premises	
3) $\neg Q \vee R$	Use 3&5 to resolve Q
4) $\neg R$	$\neg \cancel{Q} \vee R$
5) $Q \vee R$	$\cancel{Q} \vee R$
	R

and of or.

Premises

4> $\neg R$

6> R

Resolve 4 & 6

$R \land \neg R$

$= \text{False}$

2. Premises

$P \to Q$

$R \to S$

Conclusion

$(P \lor R) \to (Q \lor S)$

Implication: $((P \to Q) \land (R \to S)) \to ((P \lor R) \to (Q \lor S))$

Sol. Premises

1> $P \to Q \equiv \neg P \lor Q$

2> $R \to S \equiv \neg R \lor S$

3> $\neg((P \lor R) \to (Q \lor S)) \equiv \neg(\neg(P \lor R) \lor (Q \lor S))$

$\equiv \neg(\neg P \land \neg R \lor Q \lor S)$

$\equiv P \lor R \land \neg Q \land \neg S$

$\equiv (P \lor R) \land \neg Q \land \neg S$

4> $(P \lor R)$

5) $\neg Q$

6) $\neg S$

Maam's soln: Premises:

1> $\neg P \lor Q$

2> $\neg R \lor S$

3> $P \lor R$

4> $\neg Q$

5> $\neg S$

1> Eq 3> $\Rightarrow$ 2> $\neg R \vee S$

3> $\neg Q$

5> $\neg S$

6> $Q \vee R$

2> Eq 5> $\Rightarrow$ 4> $\neg Q$

5> $Q \vee R$

7> $\neg R$

4> Eq 6> $\Rightarrow$ 7> $\neg R$

8> R

False

3. Premises Conclusion

P Z

$\neg P$

False True

$F \rightarrow T \equiv T$

Sol. $(P \wedge \neg P) = $ False

False $\rightarrow Z$ is True

4. $(A \vee \neg B)$ D $(A \vee \neg B) \wedge (\neg C \vee B) \wedge (\neg A) \wedge (C \vee B \vee D)$

$\neg C \vee B$ $\rightarrow D$

$\neg A$

$C \vee B \vee D$

Premises :

1. $A \vee \neg B$
2. $\neg C \vee B$
3. $\neg A$
4. $C \vee B \vee D$
5. $\neg D$

1 Eq 3 : $\neg B$

2 Eq 4 : $B \vee D$

6. D

6 Eq 5 : F

5. Premises

$(P \to Q) \to Q$

$(P \to P) \to R$

$(R \to S) \to \neg(S \to Q)$

Conclusion

R

$\neg(P \to Q) \vee Q$

$\neg(\neg P \vee Q) \vee Q$

$P \wedge \neg Q \vee Q$

1) P

$(P \to P) \to R$

$\neg(P \to P) \vee R$

$\neg(\neg P \vee P) \vee R$

$F \vee R$

2) R

$(R \to S) \to \neg(S \to Q)$

$\neg(R \to S) \vee \neg(S \to Q)$

$\neg(\neg R \vee S) \vee \neg(\neg S \vee Q)$

$R \wedge \neg S \vee S \wedge \neg Q$

$R \wedge \neg Q$

3) R

4) $\neg Q$

Premises

1) P

2) R

3) R $\Big\}$ R

4) $\neg Q$

5) $\neg R$

R

$\neg R$

False

November 2023 Premises

Conclusion

6. $B_{11} \leftrightarrow (P_{12} \vee P_{21})$ $\neg P_{12}$

$\neg B_{11}$

$B_{11} \rightarrow (P_{12} \vee P_{21}) \quad \wedge \quad (P_{12} \vee P_{21}) \rightarrow B_{11}$

CNF: $\neg B_{11} \vee (P_{12} \vee P_{21}) \wedge \neg(P_{12} \vee P_{21}) \vee B_{11}$

$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12}) \wedge (\neg P_{21} \vee B_{11})$

Premises:
1. $\neg B_{11} \vee P_{12} \vee P_{21}$
2. $\neg P_{12}$
3. $\neg P_{21} \vee B_{11}$
4. P_{12}

2 & 4 $\Rightarrow$ F

First Order Predicate Logic

q1. Convert to FOPL: i) Each animal is an organism
ii) All animals are organisms
iii) If it is an animal then it is an organism

Sol i) $\forall x (Animal(x) \rightarrow Organism(x))$

q2. Not all birds can fly

Sol. $\neg(\forall x\, Bird(x) \rightarrow Fly(x))$

or $\exists x\, Bird(x) \wedge \neg Fly(x)$

as not is mentioned in q

q3. All birds cannot fly

Sol. $\forall x\, Bird(x) \rightarrow \neg Fly(x)$

or $\neg(\exists x\, Bird(x) \wedge Fly(x))$

q4. If anyone can solve the problem then Hilary can

$\times(\exists x\, Anyone(x) \rightarrow Solve(x)) \rightarrow Hilary(x)$

cannot use anyone as obj

$(\exists x\ solves(x, problem)) \rightarrow solves(Hilary, problem)$

q5. John hates all people who do not hate themselves

$\forall x\ Person(x) \land \neg Hates(x, x) \rightarrow Hates(John, x)$

q6. Everyone is loyal to someone

$\forall x\ \exists y\ LoyalTo(x, y)$

21st November, 2023

q: $C(x)$ = "x is a used-car dealer"
$H(x)$ = "x is honest"

i) $(\exists x)\, C(x)$
ii) $(\exists x)\, H(x)$
iii) $(\forall x)\, C(x) \rightarrow \sim H(x)$
iv) $(\exists x)(C(x) \land H(x))$

Sol: i)

December, 2023

$S \to AP \mid AB$
$E \to AP \mid EB \mid b$
$P \to EB$
$A \to a$
$B \to b$

S E				
ϕ	S, E, P			
ϕ	S, E	E, P		
ϕ	S	P, E	P, E	
A	A	B, E	B, E	B, E
a	a	b	b	b

2. AB or AE | BB or BE or EB or EE
S | P E ~~P~~

3. AS | AP or AE or SB or SE | BP or BE or EP or EE or
ϕ | S, E | PB or PE or EB or EE
E P

4. AS or AE | AE or AP or SB or SE or EB or EE or SP or SE
ϕ | S E | E P

5. AS AE AP

Properties Of Regular Languages

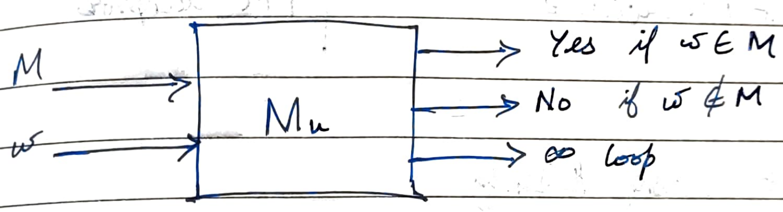
1. Concatenation
2. Union
3. Closure
4. Reversal
5. Complement
6. Intersection
7. Set Difference

→ Decidable properties of RL

1. Testing emptiness
2. Membership
3. Finite | Infinite
4. $L_1 = L_2$
5. $L = \Sigma^*$

* CNF : 2cap | 1 smol or 1 smol
* GNF : 1 smol. (many cap) or 1 smol

UTM (M_u)



* 3-tape

→ Recursively enumerable lang : lang accepted by TM

i) if ∃ algo/procedure that can enumerate all members of set

ii) make sense – i^{th} element of set

iii) can generate an element of set in finite time

iv) if set can be mapped with set of natural nos N in 1-1 correspondence

v) finite no. of elements b/w 2 elements in a set

→ Non-Recursively enumerable

i) cantor-pair elements of diff sets

ii) diagonalisation

Recursive	Recursively Enumerable
* recursive $\subset$ RE	
* enumeration procedure (TM)	
* Membership func.	No membership func. (can enter ∞ loop)
* TM deciders	TM acceptors

→ Chomsky Heirarchy