Page No.:
Date: / /

-: PUMPING LEMMA FOR REGULAR GRAMMAR

* Pumping Lemma is used to prove that a Language is NOT REGULAR.

* It cannot be used to prove that a Language is Regular.

If A is a Regular Language, then A has a Pumping Length 'P' such that any string 'S' where $|S| \geq P$ may be divided into 3 parts $S = xyz$ such that the following condition must be true:

(1) $xy^iz \in A$ for every $i \geq 0$
(2) $|Y| > 0$ OR $|Y| \geq 1$
(3) $|XY| \leq P$

Eg:- $L = \{a^n b^n \mid n \geq 1\}$
$L = \{ab, aabb, aaabbb \dots\}$

Statement:-

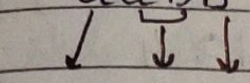
Let L be Regular Language then $\exists$ a constant n such that $\forall w \in L$ with $|w| \geq n$. we can write $w = xyz$ with $|xy| \leq n$ & $|y| \geq 1$ also $xy^iz \in L$ $\forall i \geq 0$

Here

n is ~~def~~ no of states in minimum DFA accepting L.

$\omega = aabb$ Length of $\omega = 4$
$|\omega| = 4$ $n = 4$

now we divide aabb into three parts



(i) $|xy| \leq n$
$|aab| \leq 4 \Rightarrow 3 \leq 4$ 1st condition satisfied

(ii) $|y| \geq 1$
$|ab| \geq 1 \Rightarrow 2 \geq 1$ 2nd condition satisfied

(iii) $xy^iz \in L \forall i \geq 0$
$a(ab)^ib$
put $i = 2$
$a(ab)^2b \Rightarrow aababb$

now we will check string aababb is present in our Language or not.
so $aababb \notin L$
[not present in Language]
Hence we proved a^nb^n not Regular Language

Q① Prove that $L = \{0^n 1^n \mid n \geq 0\}$ is not Regular.

Step 1:-
Assume L is regular ∃ {their exist} a minimum State finite automata M accepting a Language L.
let n be the no of states in M.

Page No.:
Date: / /

Step 2:

Consider a string $w = 0^n 1^n \in L$ $|w| \geq n$

i.e

$L = \{\epsilon, 01, 0011, 000111, 00001111 \ldots\}$

$w = 0011$ $\quad |w| \geq 4$

By pumping Lemma, we can write as

$w = xyz$

$w = 0\ 0\ 1\ 1$

$x \quad y \quad z$

now checks 3 conditions

(i) $|xy| \leq n \Rightarrow |001| \leq 4$

$3 \leq 4$ condition satisfied

(ii) $|y| \geq 1 \Rightarrow |01| \geq 1$

$2 \geq 1$ condition satisfied

Step 3 :-

(iii) $xy^iz \in L \quad \forall i \geq 0$

Let $i = 3$

$a(ab)^3b \Rightarrow a(ab)^3b$

$aababab \notin L$ given string not belongs to L.

So L is not satisfying pumping Lemma

* This contradicts our assumption that L is regular

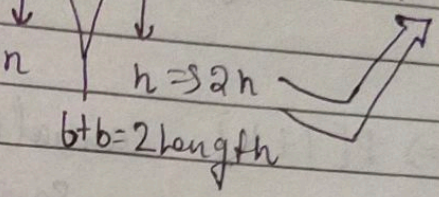
∴ L is not regular.

② Prove that the Language ww^R is not regular
$w \in [a,b]^*$

Step1:- Assume L is a regular Language ∃ a minimal state FA accepting L.

Step2:-

Consider $w = a^n bb a^n \in L$ $\quad |w| = 2n+2 \geq n$



By Pumping Lemma we can write as $w = xyz$ with $|xy| \leq n$ & $|y| \geq 1$ where y will be in the form of a^k $\quad y = a^k \quad k \geq 1$

Step3 :-

Choose $i = 0$

$xy^0z = a^{n-k} bb a^n \notin L$ as $k \geq 1$

Therefore L is not satisfying pumping Lemma
It is contradicting that L is RL.

∴

L is not a RL

③ Prove that $\{a^p / p$ is prime no$\}$ is not Regular

Step1:-

Assume L is a regular Language then there exist a minimal state FA accepting it Let n be the no of states in FA.

Step2:-

Consider $w = a^p$ with $p \geq n$ & $w \in L$
P is a prime no.

Page No.:
Date: / /

By pumping Lemma we can write. $w = xyz$
with $|xy| \leq n$ & $|y| \geq 1$ $y = a^k$ for $k \geq 1$

Step 3:-

Choose $i = p+1$

$$|xy^iz| = |xy^{p+1}z| = |xyz| + |y| * p = p + kp$$
$$\Rightarrow p[1+k] \quad k \geq 1$$

$\Rightarrow p[1+k]$ cannot be a prime no.

So $xy^iz \notin L$, for $i = p+1$

It is not satisfying pumping Lemma

Hence

L is not regular language

Page No.:
Date: / /

Regular Grammar

* Noam Chomsky gave a Mathematical model of Grammar which is effective for writing computer Language.

The four types of Grammar according to Noam Chomsky are :-

Grammar Type	Grammar Accepted	Language accepted	Automaton
TYPE-0	Unrestricted Grammar	Recursively Enumerable Language	Turing Machine
TYPE-1	Context Sensitive Grammar	Context Sensitive Language	Linear Bounded Automata
TYPE-2	Context Free Grammar	Context Free Language	Push down automata
TYPE-3	Regular Grammar	Regular Language	Finite State Machine

* GRAMMAR *

A Grammar 'G' can be formally described using 4 tuples as

$G = (V, T, S, P)$ where

V = Set of variables or Non-terminals
T = Set of terminals symbol
S = Start state
P = Production rules for Terminals & Non-termin

A production rules has the form $\alpha \to \beta$ where "α" & 'β' are strings on VUT and at least one symbol of α belongs to V.

Example:- $G = (\{S, A, B\}, \{a, b\}, S, \{S \to AB, A \to a, B \to b\})$

$V = \{S, A, B\}$

$T = \{a, b\}$

$S = S$

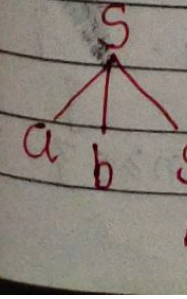
$P = S \to AB, A \to a, B \to b$

★ Regular Grammar can be divide into two types.

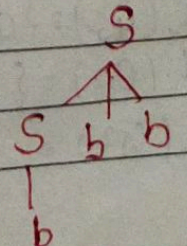
Right Linear Grammar	Left Linear Grammar
A grammar is said to be Right linear if all productions are of the form	A grammar is said to be Left linear Grammar if all production are of the form
$A \to xB$ $A \to x$	$A \to Bx$ $A \to x$
where $A, B \in V$ and $x \in T$	where $A B \in V$ and $x \in T$

eg:- $S \to abS \mid b$ Right linear Grammar

$S \to Sbb \mid b$ Left Linear Grammar



(RLG)



(LLG)

Page No.:
Date: / /

Derivations from a Grammar:-

The set of all strings that can be derived from a grammar is said to be the LANGUAGE generated from that Grammar.

Example:- Consider the Grammar
$G_1 = (\{S, A\}, \{a, b\}, S, \{S \to aAb, aA \to aaAb, A \to \epsilon\})$

Language Generated

$S \to aAb$ [By $S \to aAb$]
$S \to aaAbb$ [By $aA \to aaAb$]
$S \to aaaAbbb$ [by $aA \to aaAb$]
$L(G_1) \Rightarrow S \to aaabbb$ [by $A \to \epsilon$]

Example 2: $G_2 = (\{S, A, B\}, \{a, b\}, S, \{S \to AB, A \to a, B \to b\})$

$S \to AB$ {By $A \to a$, $B \to b$}
$S \to ab$

$L(G_2) = ab$ generated language

Examples:- $G_3 = (\{S, A, B\}, \{a, b\}, S, \{S \to AB, A \to aA \mid a, B \to bB \mid b\})$

$S \to AB$	$S \to AB$	$S \to AB$	$S \Rightarrow AB$
$S \to ab$	$S \to aAbB$	$S \to aAb$	$S \to abB$
	$S \to aabb$	$S \to aab$	$S \to abb$

$L(G_3) = \{ab, aabb, aab, abb \ldots\}$
$L(G_3) = \{a^1b^1, a^2b^2, a^2b, ab^2 \ldots\}$
$L(G_3) = \{a^m b^n \mid m \geq 0 \text{ and } n \geq 0\}$

CFG

Page No.
Date / /

* CFG:- 4 tuples [V, Σ, S, P]
V = Set of Variables or NonTerminal symbols
Σ = Set of Terminal symbols
S = Start Symbol
P = Production Rule

$A \rightarrow \alpha$

Left side only single nonterminal — Right combination of terminal & NT

Context Free Grammar has Production Rule of the form

$$A \rightarrow \alpha$$
where $\alpha = \{V \cup \Sigma\}^*$ and $A \in V$

eg: For generating a language that generated equal number of a's & b's in the form $a^n b^n$, the context Free Grammar will be defined as.

$G = \{(S, A), (a, b), (S \rightarrow aAb, A \rightarrow aAb | \epsilon)\}$

$S \rightarrow aAb$
$S \rightarrow aaAbb$ (by $A \rightarrow aAb$)
$S \rightarrow aaaAbbb$ (" ")
$S \rightarrow aaabbb$ (by $A \rightarrow \epsilon$)
$S \rightarrow a^3 b^3$
$\Rightarrow \boxed{a^n b^n}$

Page No.:
Date : / /

* Method to find whether a string belongs to a Grammar or not

① Start with the start symbol and choose the closest production that matches to the given string.

② Replace the variable with its most appropriate production. Repeat the process until the string is generated or until no other production are left.

Eg:- Verify whether the Grammar $S \to 0B \mid 1A$, $A \to 0 \mid 0S \mid 1AA \mid \wedge$, $B \to 1 \mid 1S \mid 0BB$ generate the string 00110101

$S \to 0B$ ($S \to 0B$)
$S \to 00BB$ ($B \to 0BB$)
$S \to 001B$ ($B \to 1$)
$S \to 0011S$ ($B \to 1S$)
$S \to 00110B$ ($S \to 0B$)
$S \to 001101S$ ($B \to 1S$)
$S \to 00110101B$ ($S \to 0B$)
$S \to 00110101$ ($B \to 1$)

Eg:- Verify whether the Grammar $S \to aAb$, $A \to aAb \mid \wedge$ generates the string aabbb

$S \to aAb$ ($S \to aAb$)
$S \to aaAbb$ ($A \to aAb$)
$S \to aabb$ ($A \to \wedge$)

if Replace $A \to aAb$ not generate given string

$S \to aaAbb$ $(A \to aAb)$

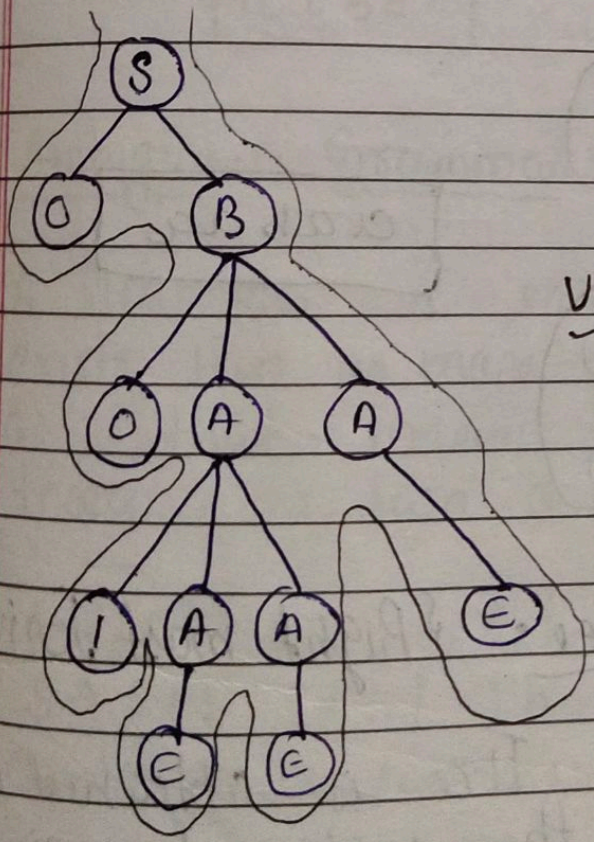
$S \to aaabbb$

not generate given string.

* Derivation tree :-

A Derivation tree or Parse tree is an ordered rooted tree that graphically represents the semantic information of string derived from a context free grammar.

Eg:- for the gramman $G = \{V, T, P, S\}$ where $S \to 0B$, $A \to 1AA/\epsilon$, $B \to 0AA$



Rootvertex :- Must be labelled by the start symbol.

Vertex :- Labelled by non-Terminal symbols.

Leaves :- Labelled by Terminal Symbols or ϵ

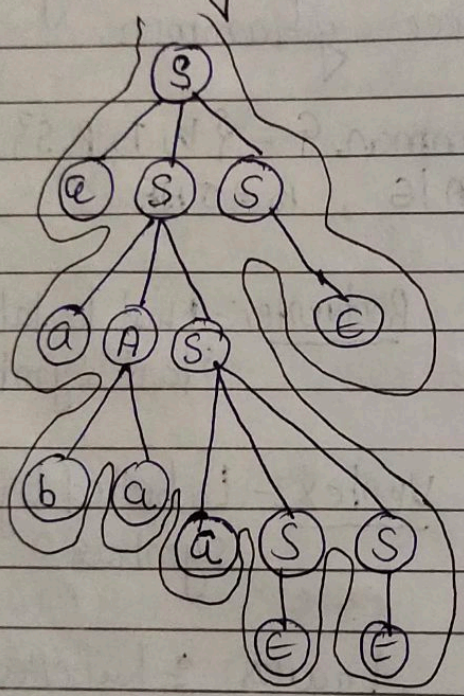
Generated lang :- 001

Page No.:
Date: / /

* **Left Derivation Tree:-** {Left most derivation}

A Left Derivation tree is obtained by applying production to the Leftmost variable in each step.

Eg: For generating the string aabaa from the grammar S → aAS | aSS | ε, A → SbA | ba

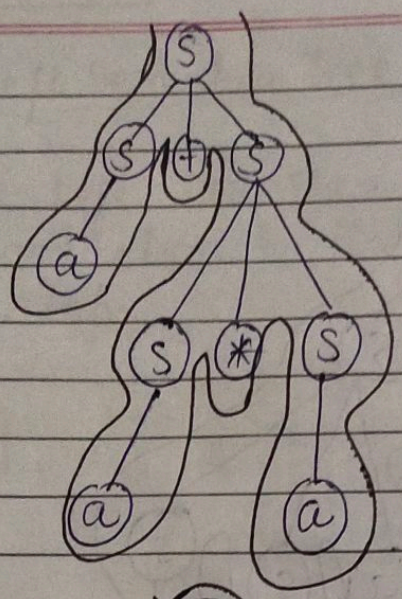


aabaa

* **Right Derivation tree:-** {Right most derivation}

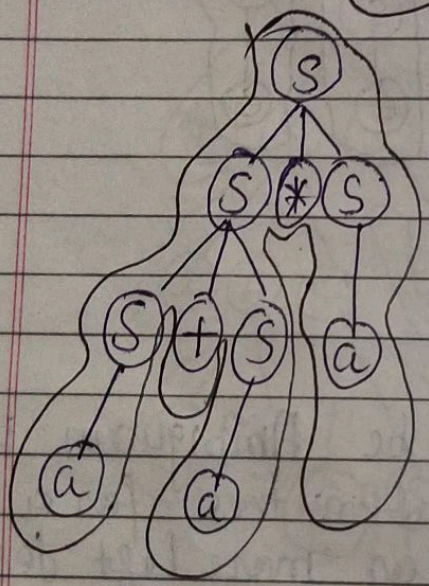
A Right Derivation Tree is obtained by applying production to the rightmost variable in each step.

Eg:- For generation the string aabbaa from the grammar S → aAS | aSS | ε, A → SbA | ba



LHD 1 Tree

a + a * a



LHD 2 Tree

a + a * a

(3) $S \rightarrow S+S$

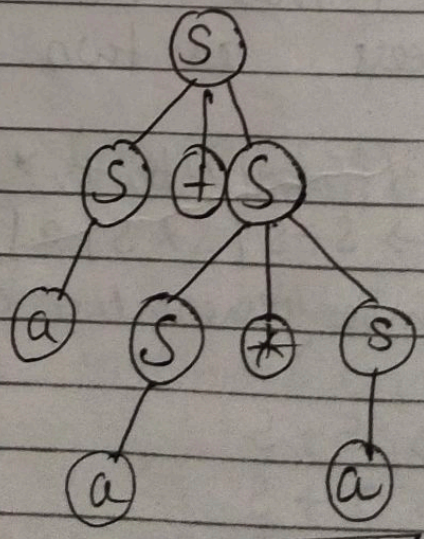
$S \rightarrow S+S*S$

$S \rightarrow S+S*a$

$S \rightarrow S+a*a$

$S \rightarrow a+a*a$

RMD1



a + a * a

RMD1 tree

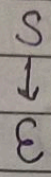
More than 1 RMD so Grammar is AMBIGUOUS

Page No.
Date / /

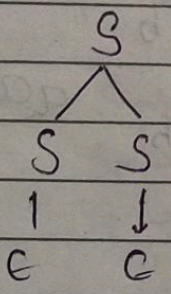
(4)

Que $S \to SS \mid (S) \mid \epsilon$ Prove Grammar is ambiguous.

$w = \epsilon$

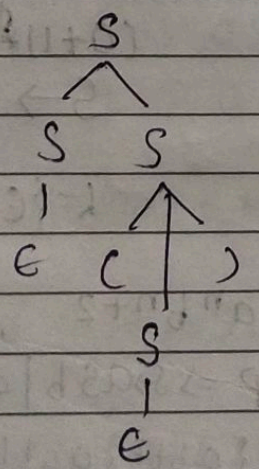


Tree 1

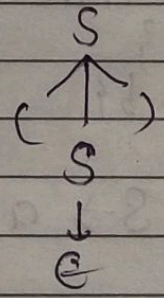


Tree 2

$w = ()$



Tree 1



Tree 2

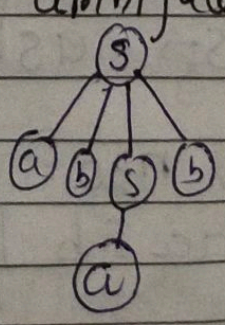
More than 1 tree so it is an Ambiguous.

Que Show that gramman is ambiguous or not.

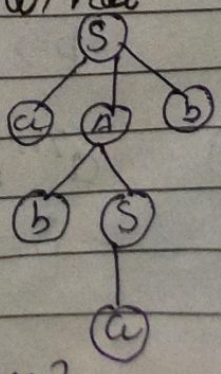
$S \to a \mid abSb \mid aAb$

$A \to bS \mid aAAb$

$w = abab$



Tree 1



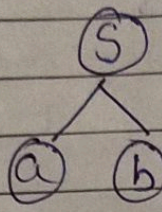
Tree 2

Date : / /

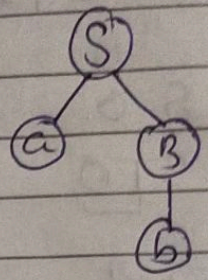
Ques Show that grammar is ambigious.

$S \to aB \mid ab$
$A \to aAB \mid a$
$B \to ABb \mid b$

w.r.t ab



Tree 1



Tree 2

Qu Consider the gramman

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

for string aaabbabbba

construct

(i) Left most derivation
(ii) Right Most derivation
(iii) Parse tree

is the ambigious or not.

LMD:-

$S \to aB$ ($S \to aB$)
$S \to aaBB$ ($B \to aBB$)
$S \to aaaBBB$ ($B \to aBB$)
$S \to aaabBB$ ($B \to b$)
$S \to aaabbB$ ($B \to b$)
$S \to aaabbaBB$ ($B \to aBB$)
$S \to aaabbabB$ ($B \to b$)
$S \to aaabbabbS$ ($B \to bS$)
$S \to aaabbabbbA$ ($S \to bA$)
$S \to aaabbabbba$ ($A \to a$)

Page No.
Date / /

RMD :-

$S \rightarrow aB$	[S → aB]
$S \rightarrow aaBB$	[B → aBB]
$S \rightarrow aaBbS$	[B → bS]
$S \rightarrow aaBbbbA$	[S → bA]
$S \rightarrow aaBbbba$	[A → a]
$S \rightarrow aaaBBbbba$	[B → aBB]
$S \rightarrow aaaBbbba$	[B → b]
$S \rightarrow aaabSbbba$	[B → bS]
$S \rightarrow aaabbAbbba$	[S → bA]
$S \rightarrow aaabbabba$	[A → a]

CFL to CFG

Page No.:
Date : / /

* Conversion CFL [context free Language] to CFG [context free Grammar]

① $a^n b^n$, $n \geq 0$ ★

$P \Rightarrow S \to aSb \mid \epsilon$

$G = [V, T, P, S]$
$V = \{S\}$
$T = \{a, b\}$

① S → ε

② S → aSb → ab

③ S → aSb → aaSbb → aabb

Q $a^n b^n \mid n \geq 3$
$S \to aSb \mid aaabbb$

② $a^n b^n \mid n \geq 1$ ★
$G = (V, T, P, S)$
$V = \{S\}$
$T = \{a, b\}$
$P \Rightarrow$
$S \to aSb \mid ab$

Q: Construct a CFG for the regular expression $(0+1)^*$
$S \to 0S \mid 1S \mid \epsilon$
$L = \{\epsilon, 0, 1, 01, 10, 00, 11 \ldots\}$

③ $a^n b^{n+2}$, $n \geq 0$
$P \Rightarrow S \to aSb \mid bb$

$a^n b^{n+2}$, $n \geq 1$
$P \Rightarrow S \to aSb \mid abbb$
$\{abbb, aabbbb, aaabbbbb \ldots\}$

④ $a^n b^{n+5}$ $n \geq 0$
$P \Rightarrow S \to aSb \mid bbbbb$

⑤ $a^{2n} b^n$, $n \geq 0$
$S \to aaSb \mid \epsilon$

Q:-

Page No.
Date / /

(6) $a^{2n} b^n \quad n \geq 1$

$S \rightarrow aaSb \mid aab$

(7) $a^{2n+3} b^n, \quad n \geq 0$

$S \rightarrow aaSb \mid aaa$

$a^{2n+3} b^n, \quad n \geq 1$

$S \rightarrow aaSb \mid aaaaab$

(8) $a^m b^n \mid m > n, \quad n \geq 0$

$S \rightarrow AS_1$
$S_1 \rightarrow aS_1b \mid \epsilon$
$A \rightarrow aA \mid a$

(9) $a^m b^n, \quad m \geq n, \quad n \geq 0$

$S \rightarrow AS_1$
$S_1 \rightarrow aS_1b \mid \epsilon$
$A \rightarrow aA \mid \epsilon$

Q:- $L = a^{4n} \mid n \geq 1$

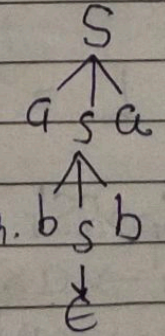
$S \rightarrow aaaaS \mid aaaa$

(10) $\{w \mid n_a(w) = n_b(w)\}$

$S \rightarrow aSb \mid bSa \mid SS \mid \lambda$

(11) $ww^R \cup w(a+b)w^R$

$S \rightarrow aSa \mid bSb \mid \epsilon$



abba

generate palindrome even length.

but $ww^n \cup w(a+b)w^R$

↳ It is odd length palindrome

So production will be

$S \rightarrow aSa \mid bSb \mid a \mid b \mid \epsilon$ → odd + even length

odd length palindrom

Page No.:
Date: / /

(12) $a^m b^m c^n \quad m, n \geq 0$

$S \rightarrow S_1 C$
$S_1 \rightarrow a S_1 b \mid \epsilon$
$C \rightarrow cC \mid \epsilon$

(13) $a^m b^n \mid m, n \geq 0$

$a^* b^*$

$S \rightarrow AB$ $\{a^* b^*\}$
$A \rightarrow aA \mid \epsilon$ $\{a^*\}$ ← generate
$B \rightarrow bB \mid \epsilon$ $\{b^*\}$ ← generate

Q14 Write a CFG, which generate Palindrome for binary no. even Length and odd Length.

$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \epsilon$

Q:15 $CFL = \{a^n b^m \quad n \neq m \{m, n \geq 0\}$

if $n \neq m$ then there are only two cases are possible.

Case 1:- if $n > m$

$L_1 = a^n b^m : n > m$

Grammar

Production ⇒ $S_1 \rightarrow AS_1'$
$S_1' \rightarrow aS_1'b \mid \epsilon$
$A \rightarrow aA \mid a$

$G_1 = \{V_1, T_1, P_1, S_1\}$
$V_1 = \{S_1, A, S_1'\}$
$T_1 = \{a, b\}$

Page No.
Date / /

Case 2:- if $n < m$

$L_2 = a^n b^m : n < m$

$G_2 = (V_2, T_2, P_2, S_2)$

~~$S_2 = S_2' B$~~

$V_2 = S_2, B, S_2'$

$T_2 = \{a, b\}$

Productions ⇒

$S_2 \Rightarrow S_2' B$

$S_2' \Rightarrow a S_2' b \mid G$

$B \Rightarrow bB \mid b$

Final grammar for Language L

$S \rightarrow S_1 \mid S_2$

Q:16 construct a CFG for set of all string of length 2. for $\Sigma = \{a, b\}$

$G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$S \rightarrow AA$

$A \rightarrow a \mid b$

Q17 $\{ ww^R \mid w \in (0,1)^* \}$;

$L = \{\epsilon, 00, 11, 0110, 1001 \ldots\}$

$w = 0$, $w^R = 0$

$w^R = 10$

$w^R = 01$

$S \rightarrow 0S0 \mid 1S1 \mid \epsilon$

Q18 $L = \{wcw^R \mid w \in (0,1)^*\}$ ★

$S \rightarrow 0S0 \mid 1S1 \mid c$

(19) $CFL = \{a^n b^m c^m d^n \mid m \geq 1, n \geq 1\}$

In this CFL 1st we will see same power symbol like a^n d^n and generate production but in B/W a^n d^n ($b^m c^m$) are also occurs, replace A, then product for A.

$S \rightarrow aSd \mid aAd$
$A \rightarrow bAc \mid bc$

(20) $\{CFL = a^n b^m c^n c^m \mid n \geq 1, m \geq 1\}$

$S \rightarrow aSc \mid aAC$
$A \rightarrow bAC \mid bC$

(21) $CFL = a^n b^n c^m d^m \mid n \geq 1, m \geq 1\}$

$S \rightarrow AB$
$A \rightarrow aAb \mid ab$
$B \rightarrow cBd \mid cd$

(22) $CFL = a^n b^n c^m d^m \mid n \geq 0, m \geq 0$

$S \rightarrow AB$
$A \rightarrow aAb \mid \epsilon$
$B \rightarrow cBd \mid \epsilon$

(23) $CFL = \{a^n b^n \mid n \geq 1\} \cup \{c^m d^m \mid m \geq 1\}$

$S \rightarrow A \mid B$
$A \rightarrow aAb \mid ab$
$B \rightarrow cBd \mid cd$

Date : / /

Que $a^n b^m \mid n = m-1 \;\&\; m \geq 1$

$L = \{ b, abb, aabbb \text{ ---} \}$

$a^n b^{n+1} \mid n \geq 0$

$S \rightarrow aSb \mid b$

Que $a^{3n} b^{5n} \mid n \geq 0$

$L = \{ \epsilon, aaabbbbb \}$

$S \rightarrow aaaSbbbbb \mid \epsilon$

Que $a^3 b^{5n} \mid n \geq 1$

$S \rightarrow aaaSbbbbb \mid aaabbbbb$

Que $a^{2n+3} b^{3n+2} \mid n \geq 0$

$S \rightarrow aaSbbb \mid aaabb$

Que $a^m b^n \mid m = n$

By default we consider $m, n = 0$

$\hookrightarrow a^m b^n \mid m, n \geq 0$

$\hookrightarrow a^n b^n \mid n \geq 0$

$S \rightarrow aSb \mid \epsilon$

Que $a^m b^n \mid m > n$

$L = \{ a, aa, aaa \text{ ---} \; aab, aaab, \text{ ---} \}$

$S \rightarrow aSb \mid aA$

$A \rightarrow aA \mid \epsilon$

OR

$S \rightarrow AS_1$

$S_1 \rightarrow aS_1b \mid \epsilon$

$A \rightarrow aA \mid a$

Que $a^m b^n \mid m < n$

$S \rightarrow BS1$

$S_1 \rightarrow aS_1b \mid \epsilon$

$B \rightarrow bB \mid b$

Page No.
Date / /

Que $a^m b^n \mid m \geq n$

$S \to aSb \mid \epsilon \mid aA$
$A \to aA \mid \epsilon$

Que $a^m b^n \mid m \leq n$

$S \to aSb \mid \epsilon \mid bB$
$B \to bB \mid \epsilon$

Que $a^m b^n \mid m \neq n$

Que $CFL = a^{n+m} b^n \mid m, n \geq 1$
$= a^n . a^m b^n \mid m, n \geq 1$

$S \to asb \mid as \mid a$

Que Design the CFG for the following language.

(i) $CFL = \{0^m 1^n \mid m \neq n \text{ & } m, n \geq 1\}$
if $m > n$ & $m < n$

$S \to S1 \mid S2$

$S_1 \to AS_1'$
$S_1' \to 0S_1'1 \mid \epsilon$
$A \to 0A \mid 0$

} $m > n$

$S_2 \to BS_2'$
$S_2' \to 0S_2'1 \mid \epsilon$
$B \to 1B \mid 1$

} $m < n$

final Grammar $S \to S1 \mid S2$

Q: Construct a CFG for the Language

$L = \{a^{2n} b^n \mid n >= 3\}$

$n = 3$

$L = a^{2\times3} b^3 = a^6 b^3 = aaaaaabbb$

$n = 4$

$L = a^{2\times4} b^4 = a^8 b^4 = aaaaaaaabbbb$

$S \rightarrow aaSb \mid aaaaaabbb$

Que Design the CFG for the following Language

$L_1 = \{a^l b^m c^n \mid l+m = n \text{ & } l, m \geq 0\}$

when

$l = 1 \quad m = 1 \quad n = l+m = 1+1 = 2$

$a^1 b^1 c^2 = abcc$

$a^1 b^2 c^3 = abbccc$

$a^2 b^1 c^3 = aabccc$

$a^2 b^2 c^4 = aabbcccc$

must add 'a' 'c' for each 'a' & 'b'

Production

$S \rightarrow aSc \mid \epsilon \mid S_1$

$S_1 \rightarrow bS_1c \mid \epsilon$

Que $L_2 = \{a^l b^m c^n \mid n+l = m\}$

$S \rightarrow S_1 S_2$

$S_1 \rightarrow aS_1b \mid \epsilon$

$S_2 \rightarrow bS_2c \mid \epsilon$

Must add a 'b' for each 'a' & 'c'

Page No.
Date: / /

Que $L_3 = \{a^n b^m \mid m \leq n \leq 2m$

must add 1 or 2 a's for each additional b.

$S \rightarrow aSb \mid aaSb \mid \epsilon$

Que $L_4 = \{w_1 c w_2 \mid w_1, w_2 \in \{a,b\}^*$, Length$(w_1)$ = Length$(w_2)\}$

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

Que $L_5 = \{w_1 c a^n b^m a^i b^j w_2 \mid w_1, w_2 \in \{a,b\}^*$
$\{$Length(w_1) = Length(w_2), $j = 2i$ $n \leq m\}$

$S \rightarrow aSb \mid aSa \mid bSa \mid bSb \mid S_1$
$S_1 \rightarrow cS_2S_3$
$S_2 \rightarrow aS_2b \mid S_2b \mid \epsilon$
$S_3 \rightarrow aS_3bb \mid \epsilon$

Que $L_6 = \{a^n b^m c^k \mid n \neq m$ or $m \neq k\}$
There are 4 cases are consider

① More a's than b's (w/ any number of c's)
② more b's than a's (w/ any number of c's)
③ More b's than c's (w/ any number of a's)
④ more c's than b's (w/ any number of a's)

$S \rightarrow S_1S_3 \mid S_2S_3 \mid S_4S_5 \mid S_4S_c$
Each of the 4 cases are accounted for [from left to right in the above production.]

Page No.:
Date: / /

$S_1 \rightarrow aS_1b \mid aS_1 \mid a$

$S_2 \rightarrow aS_2b \mid S_2b \mid b$

$S_3 \rightarrow S_3C \mid \epsilon$

$S_4 \rightarrow aS_4 \mid \epsilon$

$S_5 \rightarrow bS_5C \mid bS_6 \mid b$

$S_6 \rightarrow bS_6C \mid S_6C \mid C$

Page No.
Date / /

* Conversion CFG to CFL

(1) $S \to 0S0 \mid 1S1 \mid A$
$A \to 2B3$
$B \to 2B3 \mid 3$

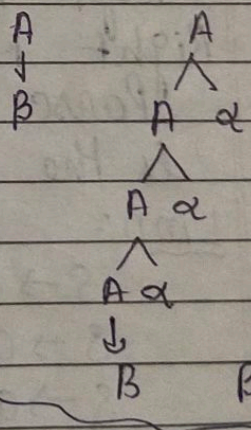
$S \to 0S0$
$\quad 0 \mid S \mid 0$

$\to wSw^R$ $\quad w, \in \{0,1\}^*$
$\to wAw^R$ $\quad \{\because S \to A\}$
$\to w2B3w^R$ $\quad A \to 2B3$
$\to w2^n3^{n+1}w^R$ $\quad n \geq 1$

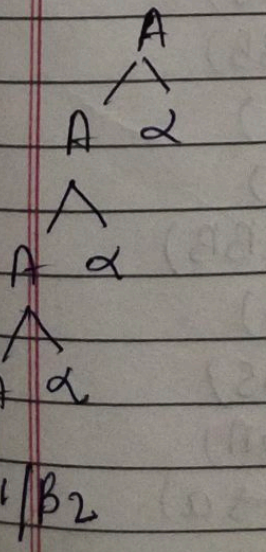
CFL = $w^{2^n}3^{n+1}w^R$ $\quad n \geq 1, w \in (0,1)^*$

(2) $A \to A\alpha \mid B$

CFL = $B\alpha^*$



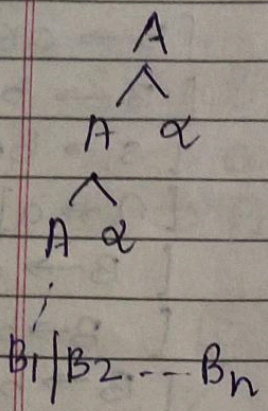
(3) $A \to A\alpha \mid B_1 \mid B_2$



CFL = $(\beta_1 + \beta_2)\alpha^*$

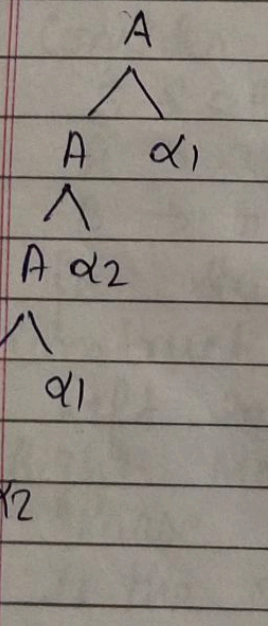
Page No.:
Date: / /

④ $A \to A\alpha \mid B_1 \mid B_2 \mid \dots \mid B_n$



$CFL = (B_1 + B_2 + \dots + B_n)\alpha^*$

⑤ $A \to A\alpha_1 \mid A\alpha_2 \mid \beta$



$CFL = \beta(\alpha_1 + \alpha_2)^*$

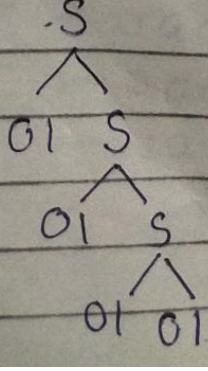
Q6:- $A \to A\alpha_1 \mid A\alpha_2 \mid \dots A\alpha_n \mid \beta$

$CFL = \beta(\alpha_1 + \alpha_2 + \dots + \alpha_n)^*$

Q7 $A \to A\alpha_1 \mid A\alpha_2 \mid \dots A\alpha_n \mid B_1 \mid B_2 \dots \mid B_m$

$CFL = (B_1 + B_2 + B_3 \dots B_m)(\alpha_1 + \alpha_2 + \dots + \alpha_n)^*$

Q8 $S \to 01S \mid 01$

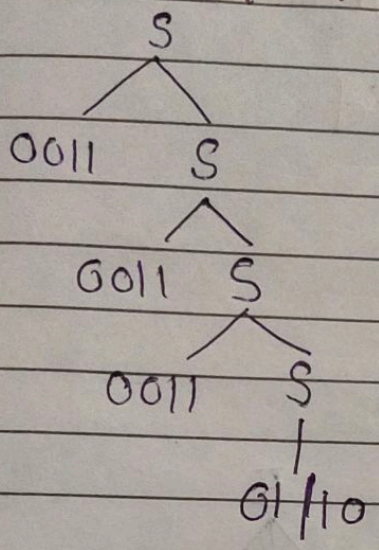


$CFL \Rightarrow (01)^* 01$

$CFL \Rightarrow (01)^+$

Page No.
Date / /

Q9 $S \to 0011S \mid 01 \mid 10$



$CFL = (0011)^*(01+10)$

Q10 $S \to 01A \mid B11$
$A \to 011A \mid 01$
$B \to 101B \mid 11$

$A \to 011A \mid 01 \Rightarrow (011)^*01$
$B \to 101B \mid 11 \Rightarrow (101)^*11$

$S \to 01(011)^*01 + (101)^*1111$

$CFL = 01(011)^*01 + (101)^*1111$

Q11 $S \to 011A \mid 101B$
$A \to 110A \mid 00 \to (110)^*00$
$B \to 11B \mid S$

$S \to 011(110)^*00 \mid 101B$
$B \to 11B \mid S \Rightarrow (11)^*S$

$S \to 011(110)^*00 \mid 101(11)^*S$

$[101(110)^*]^*011(110)^*00$

Page No.:
Date: / /

Conversion Grammar to Finite Automata

Eg: construct a FSM 'M' that recognises the Language generated by the following grammar G:

$S \rightarrow a \mid aA \mid bB \mid \epsilon$
$A \rightarrow aA \mid aS$
$B \rightarrow cS \mid \epsilon$

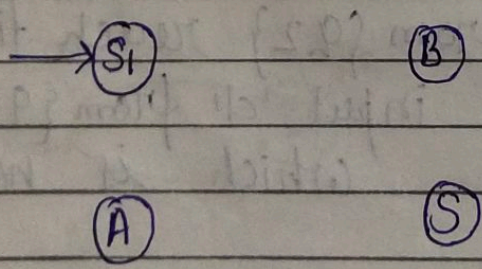
where 'S' is the start symbol.

Step 1:- Construct an equivalent ϵ-free grammar.

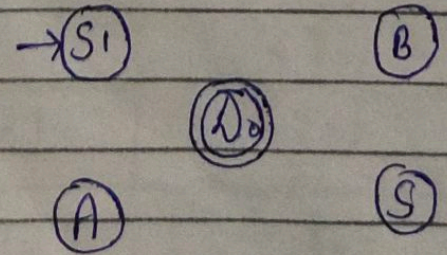
$S1 \rightarrow a \mid b \mid aA \mid bB \mid \epsilon$

where S1 is the start symbol.

Step 2:- Create a FSM 'M' with a state for every nonterminal in 'G' - {S1, S, A, B}, S1 being the start symbol is set to be the start state

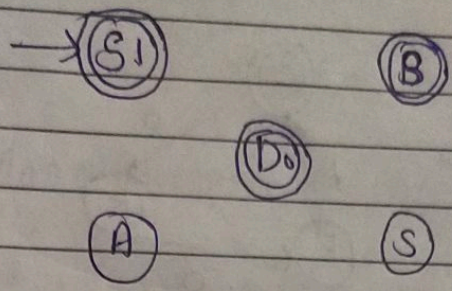


Step 3:- add another Non Terminal state Do & find state

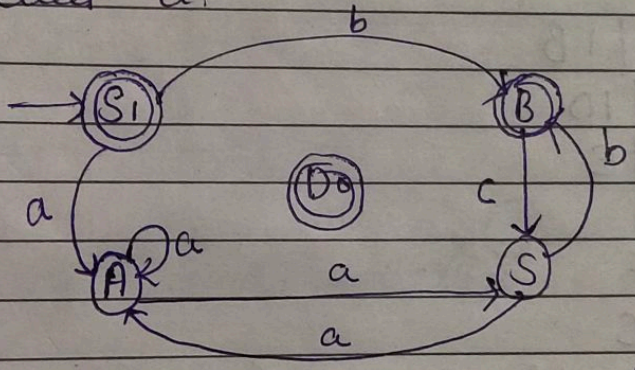


Page No.
Date: / /

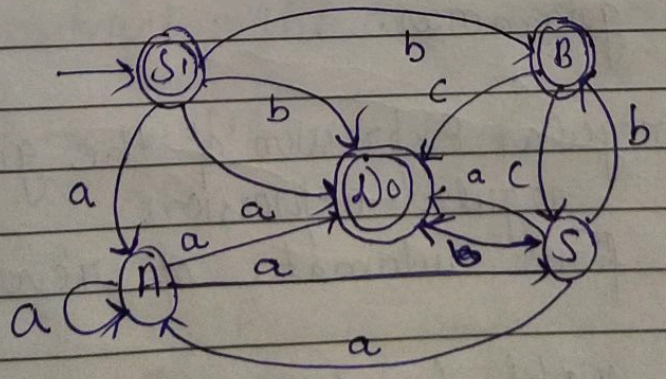
Step 4:- S1 gives ε {S1 → ε} was in the grammar so, state S1 is also set to be final



Step 5:- for every rule of the form A → aB, we add a transition from state A to state B labelled a:



Step 6:- for every rule of the form A → a we add a transition from state A to state D0 Labelled as:



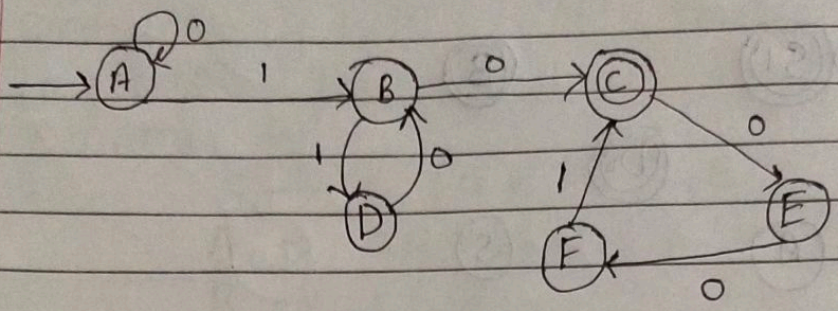
Qus G = ({a, b, c} {S, A, B, C} construct F.Automata

S → aA | bC
A → aA | bB | ε
B → bB | b | ε
C → cB | cC | ε

Page No.:
Date : / /

Conversion Finite Automata FA to Grammar

Q.



convert given FA into grammar.

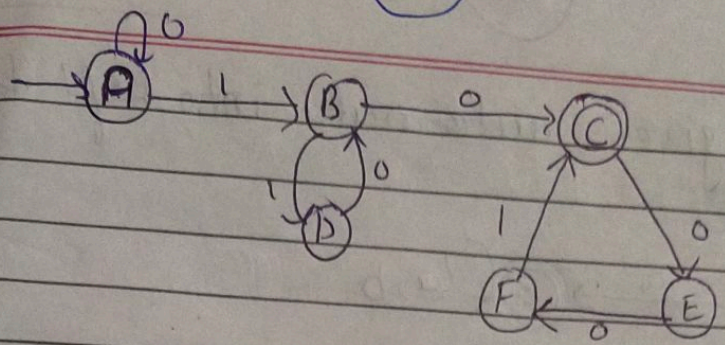
Sol^n :-

$A \rightarrow 0A \mid 1B$
$B \rightarrow 0C \mid 1D$
$C \rightarrow 0E \mid \epsilon$
$D \rightarrow 0B$
$E \rightarrow 0F$
$F \rightarrow 1C$

This is Right Regular Grammar

* To convert given ~~Gramm~~ Finite automata into Left Regular grammar.

(1) Obtain the regular Expression of the given FA
(2) Reverse the regular expression
(3) Construct the finite automata of reverse regular expression.
(4) Construct the right ~~hand~~ regular grammar.
(5) Reverse right hand side of every production



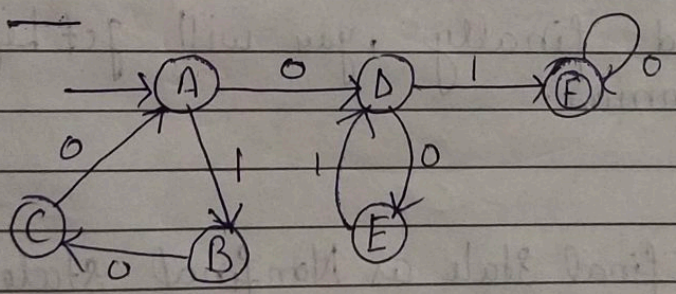
(1) Regular expression of given FA

$0^* 1 (10)^* 0 (001)^*$

(2) Reverse the R.E.

$(100)^* 0 (01)^* 10^*$

(3) Finite Automata



(4) Construct right regular Grammar

A → 0D | 1B
B → 0C
C → 0A
D → 0E | 1F
E → 1D
F → 0F / ∈

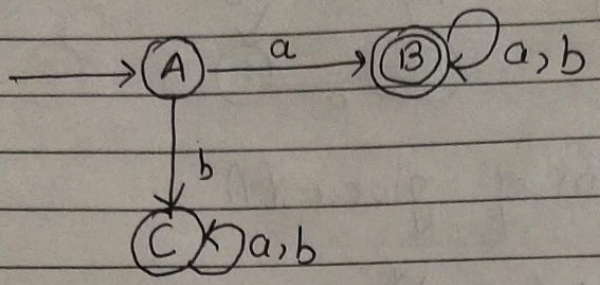
A → D0 | B1
B → C0
C → A0
D → E0 | F
E → D1
F → F0 | ∈

This is Left Regular Gramma

(5) Reverse right hand side of every production.

Page No.:
Date: / /

Qu Convert the given automata into Left linear grammar.



Steps for conversion

Step 1:- Take reverse of the finite automata.

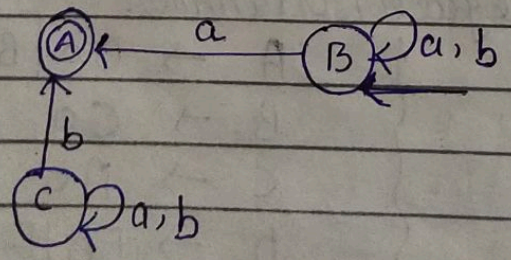
Step 2:- Write right linear grammar

Step 3:- Then take reverse of the right linear grammar.

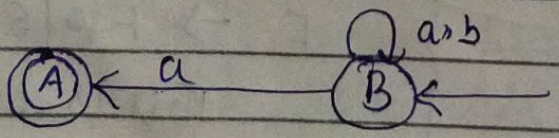
Step 4:- And finally, you will get Left linear grammar.

⇒ Step 1:-

Make final state as Non final state and initial state as final state.



Now remove the unreachable state.



After removing the unreachable states, the transition diagram is not in the DFA.

Page No.
Date / /

because state 'A' has no output symbol whereas state B is having two outputs on symbol 'a'
So the resultant diagram looks like the NFA.
First generate the Right linear grammar for the transition diagram.

$B \rightarrow aA \mid aB \mid bB$
$A \rightarrow \epsilon$

Now revers the Right linear grammar to generate left linear grammar.

$B \rightarrow Ba \mid Bb \mid Aa$
$A \rightarrow \epsilon$

Left Linear Grammar.

:- Simplification of CFG :-

means,

[Removal of unnecessary Symbol]

* Does not contain NULL production
  $\{A \to \epsilon\}$

* Does not contain UNIT production

  $\{A \to B\}$

means

[Nonterminal → Nonterminal]

* Does not contain USELESS Symbol

$$\left\{\begin{array}{l} S \to aA \\ A \to b \\ B \to a \end{array}\right\}$$

In this CFG 'B' is useless, Because from Start Symbol 'S', no use of 'B'.

$G = (V, T, P, S)$

$A \in V$ is said to be useful iff there is atleast one $w \in L(G)$ such that

$S \overset{*}{\Rightarrow} xAy \overset{*}{\Rightarrow} w$ $\quad x, y \in (V \cup T)^*$

Page No.
Date / /

Q:- Simplify the following CFG:

$S \rightarrow ABAC$
$A \rightarrow aA \mid \epsilon$
$B \rightarrow bB \mid C$
$C \rightarrow c$

Step 1:-

Eliminate of null production
Nullable variable is $\{A\}$
$\{A \rightarrow \epsilon\}$

*

The equivalent grammar G after eliminating null production is

Replace $S \rightarrow ABAC$ with $\{A \rightarrow \epsilon\}$
$S \rightarrow ABC \mid BAC \mid BC \mid ABAC$

Replace $A \rightarrow aA \Rightarrow A \rightarrow a\epsilon \Rightarrow A \rightarrow a$
$A \rightarrow a$ with
$A \rightarrow aA \mid a$

Resultant CFG

$S \rightarrow ABAC \mid ABC \mid BAC \mid BC$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid C$
$C \rightarrow c$

Step 2:-

Elimination of Unit Production

$B \rightarrow C$ is Unit production
replace by $B \rightarrow c$ {because NonTer → NonTer}

so

$S \to ABAC \mid BAC \mid ABC \mid BC$
$A \to aA \mid a$
$B \to bB \mid c$
$B \to c$

Steps:-

Elimination of useless symbol

(a)

Elimination of variables not deriving any terminal string.

$A \to a$
$B \to c$
$C \to c$

$w_1 = \{A, B, C\}$
$w_2 = w_1 \cup \{S\}$
$= \{A, B, C\} \cup \{S\} = \{A, B, C, S\}$

(b) Elimination of symbols not appearing in any sentential form.

$w_1 = \{S\}$
$w_2 = w_1 \cup \{A, B, C\} = \{S\} \cup \{A, B, C\} = \{S, A, B, C\}$
$w_3 = w_2 \cup \{a, b, c\}$
$= \{S, A, B, C, a, b, c\}$

Here

$V' = \{S, A, B, C\}$
$T' = \{a, b, c\}$

Final Grammar:-

$S \to ABAC \mid BAC \mid ABC \mid BC$
$A \to aA \mid a$
$B \to bB \mid c$
$C \to c$

Page No.
Date / /

Que Simplify the following CFG:-

$S \to AB$
$A \to a$
$B \to C \mid b$
$C \to D$
$D \to E$
$E \to a$
$F \to a$

Step1 Eliminate null production.
but in this CFG, there are no null production.

Step2:- Eliminate Unit production

$B \to C$
$C \to D$
$D \to E$ are the unit production

Replace

$D \to E$ with $D \to a$
$C \to D$ with $C \to a$
$B \to C$ with $B \to a$

Grammar becomes:-

$S \to AB$
$A \to a$
$B \to a \mid b$
$C \to a$
$D \to a$
$F \to a$

Step3:- Eliminate useless symbols.

(a) ~~There are no variable/Nonterminals who drive~~

Page No.:
Date: / /

$W_1 = \{A, B, C, D, E, F\}$
$W_2 = \{S\} \cup \{W_1\}$
$W_2 = \{S, A, B, C, D, E, F\}$

(b) Elimination of symbol not appearing in any sentetial form.

$W_1 = \{S\}$
$W_2 = W_1 \cup \{A, B\} = \{S, A, B\}$
$W_3 = W_2 \cup \{a, b\} = \{S, A, B, a, b\}$

$V' = \{S, A, B\}$
$T' = \{a, b\}$

Remove the production

$C \to a$ $\quad$ $E \to a$
$D \to a$ $\quad$ $F \to a$

Simplified grammar will be:-

$S \to AB$
$A \to a$
$B \to a/b$

Que Eliminate Null production from given Grammar.

$S \to aAB$
$A \to aAA/\epsilon$
$B \to bBB/\epsilon$

Soln:

identified Nullable variable } gives null production

Here

$A \to \epsilon$ $\quad$ A & B are Nullable
$B \to \epsilon$

Replace A & B by ∈

$S \to aAB \mid aA \mid aB \mid a$
$A \to aAA \mid aA \mid a$
$B \to bBB \mid bB \mid b$

Ques: Eliminate null production

$S \to AB$
$A \to aAA \mid \in$
$B \to bBB \mid \in$

$A \to \in$ } nullable variable
$B \to \in$ }

$S \to AB \mid B \mid A \mid \in$ ⟹ When start variable contain Null, no need to Eliminate it
$A \to aAA \mid aA \mid a$
$B \to bBB \mid bB \mid b$

Ques: Remove null production

$S \to AbaC$
$A \to BC$
$B \to b \mid \in$
$C \to D \mid \in$
$D \to d$

Nullable variable { $B \to \in$, $C \to \in$, $A \to \in$ }

$S \to AbaC \mid Aba \mid baC \mid ba$
$A \to BC \mid C \mid B \mid \in$

{ $A \to BC$, $A \to \in \in$, $A \to \in$ }

$A \to BC \mid C \mid B$
$B \to b$
$C \to D$
$D \to d$

↳ Here we can't put ∈, because A is not start Symbol, so remove

Final Grammar:-

$S \to AbaC \mid bac \mid Aba \mid ba$
$A \to BC \mid C \mid B$
$B \to b$
$C \to D$
$D \to d$

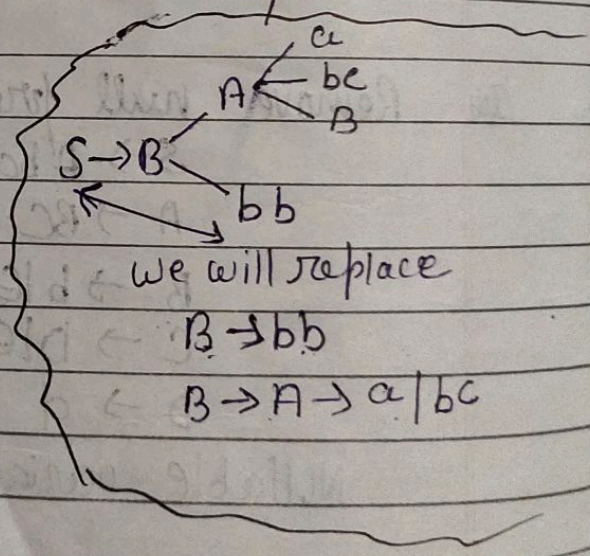
Q. Removal of Unit production

$S \to Aa \mid B$
$B \to A \mid bb$
$A \to a \mid bc \mid B$

Soln:-

$S \to B$
$B \to A$
$A \to B$

} all are unit production



$S \to Aa \mid bb \mid a \mid bc$
$B \to bb \mid a \mid bc$
$A \to a \mid bc \mid bb$
final grammar.

Q. Removal of unit Production.

$S \to AB$
$A \to a$
$B \to b \mid C$
$C \to D$
$D \to E$
$E \to a$

Page No.
Date: / /

Sol^n:-

$B \to C$
$C \to D$
$D \to E$

} unit production

$B \to C \to D \to E \to a$

$S \to AB$
$A \to a$
$B \to b \mid a$
$C \to a$
$D \to a$
$E \to a$

final grammar

NOTE

$C \to a$
$D \to a$
$E \to a$

These production all are useless. Because of all C, D, E can't reach from starting symbol

Removal of useless production:-

① Derivability
② Reachibility

$S \to AB \mid a$
$A \to BC \mid b$
$B \to aB \mid C$
$C \to aC \mid B$

Those symbol not participate to driving a storing known as useless symbol

Sol^n:-

find useful symbol:-
all terminal are {a, b, S, A}

$S \to AB \mid a$
$A \to BC \mid b$
$B \to aB \mid C$
$C \to aC \mid B$

$S \to AB$
$A \to BC$
} useless

both B & C are useless.
now we will check B & C right side of grammar if we got, then that production also useless.

$A \to BC$ useless production

$S \to a$

$A \to b$

Here we will check, from start symbol 'S' we reach on 'A', there is no way to reach 'A'
so $A \to b$ also useless.

Final CFG is $\boxed{S \to a}$

Ques Removal of useless productions:-

$S \to AB \mid AC$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$
$C \to abCA \mid aDb$
$D \to bD \mid aC$

$A \to a$
$A \to aAb$

Soln:-

useful symbol
$\{a, b, A, B, S\}$

useless
$C \to abCA \mid aDb$
$D \to bD \mid aC$
$S \to AC$

Final CFG:-

$S \to AB$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$

Page No.
Date / /

Removal of useless Production/Symbols:-

① Remove Non-Generating Symbols / Productions
② Remove Non-Reachable Symbols

Q Removal of useless production:

$S \to ABC \mid BaB$
$A \to aA \mid BaC \mid aaa$
$B \to bBb \mid a$
$C \to CA \mid AC$

Sol^n

* Remove Non-generating symbols

Useful symbol

$\{a, b\} \to \{a, b, A, B\} \to \{a, b, A, B, S\}$

$S \to ABC$
$A \to BaC$
$C \to CA \mid AC$

} These all are non generating

$S \to BaB$
$A \to aA \mid aaa$
$B \to bBb \mid a$

* Remove Non-reachable symbols

$A \to aA \mid aaa$
~~$B \to bBb \mid a$~~

~~'B'~~ 'A' non-reachable from start symbol so remove.

Final CFG:-

$S \to BaB$
$B \to bBb \mid a$

Page No.:
Date: / /

Chomsky Normal form & Convert CFG to CNF

$$\boxed{\begin{array}{l} A \to a \\ A \to BC \end{array}}$$

Procedure to find Equivalent Grammar in CNF:

① Eliminate the Null Production [ϵ]
Eliminate the unit production
Eliminate useless symbols to get Reduced grammar if any.

② Eliminate the terminals on RHS of the production having length 2 or more

Eg:-
$A \to aB$

A → gives single terminal & NT
CNF property fail.

So convert it

$$\boxed{\begin{array}{l} A \to C_1 B \\ C_1 \to a \end{array}}$$

* Here no, null, unit & useless production
* now check CNF property.
  $A \to a$
  $A \to BC$

Eg:-
$A \to aBb$
$A \to C_a B C_b$
$C_a \to a$
$C_b \to b$

Eg:-
$A \to aAbB$
$A \to C_a A C_b B$ ✗ not in CNF
$C_a \to a$
$C_b \to b$

* Restrict the number of variables on R.H.S of production

$A \rightarrow CaACbB$

$\Downarrow\Downarrow$

$A \rightarrow CaD_1$
$D_1 \rightarrow AD_2$
$D_2 \rightarrow CbB$

} Now this is in CNF

Advantage of CNF:-

(i) Length of each production is restricted
(ii) Derivation tree or parse tree obtained from CNF is always Binary tree.
(iii) The no of steps required to derive a string of length $|w|$ is $(2|w|-1)$

Q Convert CFG into CNF:-

$S \rightarrow ABa$
$A \rightarrow aab$
$B \rightarrow AC$

this grammar not follow the CNF property replace

$X \rightarrow a$
$Y \rightarrow b$
$Z \rightarrow C$

$S \rightarrow ABX$
$A \rightarrow XXY$
$B \rightarrow AZ$

} both production not follow the CNF

again replace

$T \rightarrow AB$
$W \rightarrow XX$

now resultant grammar

$S \rightarrow TX$
$A \rightarrow WY$
$B \rightarrow AZ$
$X \rightarrow a$
$Y \rightarrow b$
$Z \rightarrow c$
$T \rightarrow AB$
$W \rightarrow XX$

Que:- Convert the following CFG to CNF

$S \rightarrow ASA \mid aB$
$A \rightarrow B \mid S$
$B \rightarrow b \mid \epsilon$

① Since 'S' appears in RHS, we add a new State 'S' and $S' \rightarrow S$ is added to the production.

$S' \rightarrow S$
$S \rightarrow ASA \mid aB$
$A \rightarrow B \mid S$
$B \rightarrow b \mid \epsilon$

② Remove the NULL Production:

$B \rightarrow \epsilon$ & $A \rightarrow \epsilon$

After Removing $B \to \epsilon$

$S' \to S$

$S \to ASA \mid aB \mid a$

$A \to B \mid S \mid \epsilon$

$B \to b$

After Removing $A \to \epsilon$

$S' \to S$

$S \to ASA \mid aB \mid a \mid AS \mid SA \mid S$

$A \to B \mid S$

$B \to b$

(3) Remove Unit production

$S' \to S$

$S \to S$

$A \to B$

$A \to S$

After Removing $S \to S$

$S' \to ASA \mid aB \mid a \mid AS \mid SA$

$S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to B \mid S$

$B \to b$

State
oduction.

$S' \to S$

$S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to B \mid S$

$B \to b$

After removing $S' \to S$

$S' \to ASA \mid aB \mid a \mid AS \mid SA$

$S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to B \mid S$

$B \to b$

After removing $A \to B$

$S' \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to b \mid S$
$B \to b$

After removing $A \to S$

$S' \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to b \mid ASA \mid aB \mid a \mid AS \mid SA$
$B \to b$

④ Now find out the production that has more than two variables in RHS, dont follow CNF rule.

$S' \to ASA$
$S \to ASA$
$A \to ASA$

After removing these we get

$S' \to AX \mid aB \mid a \mid AS \mid SA$
$S \to AX \mid aB \mid a \mid AS \mid SA$
$A \to b \mid AX \mid aB \mid a \mid AS \mid SA$
$B \to b$
$X \to SA$

⑤ Now change the production who dont follow CNF property

$S' \to aB$
$S \to aB$
$A \to aB$
$Y \to a$

Finally we get

$S' \to AX \mid YB \mid a \mid AS \mid SA$
$S \to AX \mid YB \mid a \mid AS \mid SA$
$A \to b \mid AX \mid YB \mid a \mid AS \mid SA$
$B \to b$
$X \to SA$
$Y \to a$

Ex: Design a CFG for the Language $L = \{a^{4n} \mid n \geq 1\}$ & convert that CFG into CNF form.

Let CFG for the Language $L = \{0^{4n} \mid n \geq 1\}$

$S \to aaaaS \mid aaaa$

Now let us convert this CFG into CNF from replace each 'a' by 'A' then CFG

$S \to AAAAS \mid AAAA$
$A \to a$

the CFG form will be

$S \to AR_1$
$R_1 \to AR_2$
$R_2 \to AR_3$
$R_3 \to AS$
$S \to AR_4$
$R_4 \to AR_5$
$R_5 \to AA$
$A \to a$

Page No.:
Date: / /

Greibach Normal form

⇒ A CFG is in Greibach Normal form if the productions are in the following forms.

$A \to b$
$A \to bC_1C_2C_3 \text{ ---- } C_n$

where A, $C_1 \text{ ---} C_n$ are Non-Terminals and 'b' is a Terminal.

* Steps to convert a given CFG to GNF:-

Step1:-

Check if the given CFG has any Unit Production or NULL productions and Remove if there are any.

Step2:- Check whether the CFG is already in Chomsky Normal Form (CNF) and convert it to CNF if it is not.

Step3 Change the names of the Non-Terminals Symbols into some Ai in ascending order of i

Ex:-

$S \to CA \mid BB$
$B \to b \mid SB$
$C \to b$
$A \to a$

Page No.
Date / /

Step 1:- This grammar not contain NULL, unit & useless production.
Step 2:- already in CNF
Step 3:- Changes the names of the Nonterminal.

$S \to CA \mid BB$
$B \to b \mid SB$
$C \to b$
$A \to a$

Replace with

$S \to A_1$
$C \to A_2$
$A \to A_3$
$B \to A_4$

We get:

$A_1 \to A_2A_3 \mid A_4A_4$
$A_4 \to b \mid A_1A_4$
$A_2 \to b$
$A_3 \to a$

Step 4:- Alter the rules so that the Non-Terminals are in ascending order such that if the Production is of the form

$A_i \to A_jX$, then $i < j$ & should never be $i \geq j$

$i=1 \quad J=2, \; j=3$

$A_1 \to A_2A_3 \mid A_4A_4 \implies A_i \to A_jX \; [i<j]$ ✓
$A_4 \to b \mid A_1A_4$ ✗ $\quad A_i = A_4 \Rightarrow A_j = 1,4$ which is not allow
$A_2 \to b$
$A_3 \to a$

$A_4 \to b \mid A_1A_4$

So we replace $A_1 \to$ by A_2A_3 & A_4A_4

$A_4 \rightarrow b \mid A_1 A_4$
$A_4 \rightarrow b \mid A_2 A_3 A_4 \mid A_4 A_4 A_4$

Now il follow the rule step 4

$A_4 \rightarrow b \mid A_2 A_3 A_4 \mid A_4 A_4 A_4$

↓

not follow
$A_i < A_j$

so replace by

$A_2 \rightarrow b$

$A_4 \rightarrow b \mid b A_3 A_4 \mid A_4 A_4 A_4$ → Not follow GNF

↓

both follow GNF rules

$A_4 \rightarrow A_4 A_4 A_4$

Left recursion

Step 5:- Remove left recursion

Introduce a New variable to remove the Left Recursion

$A_4 \rightarrow b \mid b A_3 A_4 \mid A_4 A_4 A_4$

$Z \rightarrow A_4 A_4 Z \mid A_4 A_4$

$A_4 \rightarrow b \mid b A_3 A_4 \mid bZ \mid b A_3 A_4 Z$

This is GNF

Page No.
Date: / /

Now the Grammar is:-

$A_1 \rightarrow A_2 A_3 \mid A_4 A_4$
$A_4 \rightarrow b \mid bA_3A_4 \mid bZ \mid bA_3A_4Z$
$Z \rightarrow A_4A_4 \mid A_4A_4Z$
$A_2 \rightarrow b$
$A_3 \rightarrow a$

replace $A_2 \Rightarrow$ by $A_2 \rightarrow b$

$A_1 \rightarrow bA_3 \mid bA_4 \mid bA_3A_4A_4 \mid bZA_4 \mid bA_3A_4ZA_4$
$A_4 \rightarrow b \mid bA_3A_4 \mid bZ \mid bA_3A_4Z$
$Z \rightarrow bA_4 \mid bA_3A_4A_4 \mid bZA_4 \mid bA_3A_4ZA_4 \mid bA_4Z \mid bA_3A_4A_4Z \mid bZA_4Z \mid bA_3A_4ZA_4Z$
$A_2 \rightarrow b$
$A_3 \rightarrow a$

* FORMULA TO REMOVE LEFT RECURSION

$A \rightarrow A\alpha_1 \mid A\alpha_2 \ldots\ldots \mid \beta_1 \mid \beta_2$

A production
$A \rightarrow \beta_1 \mid \beta_2$
$A \rightarrow \beta_1 Z \mid \beta_2 Z$

Z production
$Z \rightarrow \alpha_1 \mid \alpha_2$
$Z \rightarrow \alpha_1 Z \mid \alpha_2 Z$

Eg:- $A_2 \rightarrow A_2A_2A_1 \mid \alpha A_1 \mid b$

(A_2: A; A_2A_1: $A\alpha$; αA_1: β_1; b: β_1)

A production
$A_2 \rightarrow \alpha A_1 \mid b$
$A_2 \rightarrow \alpha A_1 Z \mid bZ$

Z-production
$Z \rightarrow A_2A_1$
$Z \rightarrow A_2A_1Z$

Q... Convert the grammar

$$S \to AB$$
$$A \to BS \mid a$$
$$B \to SA \mid b$$

into GNF

Ans: The given grammar is in CNF, Let us change the name of variable as A_1, A_2, A_3 for S, A, B respectively, Now the given Productio are in the form (With A_1 as start Symbol)

$$A_1 \to A_2A_3$$
$$A_2 \to A_3A_1 \mid a$$
$$A_3 \to A_1A_2 \mid b$$

In the above grammar we need production must start with terminal or with higher no. variable.

Applying substitution rule for A_1 then we get

$$A_3 \to A_2A_3A_2 \mid b$$

Applying Substitution rule for A_2

$$A_3 \to A_3A_1A_3A_2 \mid aA_3A_2 \mid b$$

Eliminating the left recursion from A_3 Production, We get

$$A_3 \to A_3B_1 \mid aA_3A_2 \mid b$$

$$B_1 \to A_1A_3A_2B_1 \mid A_1A_3A_2$$

$$A_3 \to aA_3A_2B_1 \mid bB_1 \mid aA_3A_2 \mid b$$

B_1 is new variable introduced.

P.T.O

Now all the A_3 Production start with terminals. With the help of substitution rule we can replace A_3 in R.H.S of A_2 production.

$$A_2 \Rightarrow aA_3A_2B_1A_1 \mid bB_1A_1 \mid aA_3A_2A_1 \mid bA_1 \mid a$$

Now, all A_2 Productions start with terminal. Again we replace A_2 from $A_1 \rightarrow A_2A_3$

$$A_1 \rightarrow aA_3A_2B_1A_1A_3 \mid bB_1A_1A_3 \mid aA_3A_2A_1A_3 \mid bA_1A_3 \mid aA_3$$

one Production $B_1 \rightarrow A_1A_3A_2B_1 \mid A_1A_3A_2$ is not in GNF. Now applying substitution rule in Production.

$B_1 \rightarrow aA_3A_2B_1A_1A_3A_3A_2B_1 \mid bB_1A_1A_3A_2A_3A_2B_1 \mid aA_3A_2A_1A_3A_3A_2B_1 \mid bA_1A_3A_3A_2B_1 \mid aA_3A_3A_2B_1 \mid aA_3A_2B_1A_1A_3A_3A_2 \mid bB_1A_1A_3A_3A_2 \mid aA_3A_2A_1A_3A_3A_2 \mid bA_1A_3A_3A_2 \mid aA_3A_3A_2$

All Productions are.

$A_1 \rightarrow aA_3A_2B_1A_1A_3 \mid bB_1A_1A_3 \mid aA_3A_2A_1A_3 \mid bA_1A_3 \mid aA_3$

$A_2 \rightarrow aA_3A_2B_1A_1 \mid aB_1A_1 \mid aA_3A_2A_1 \mid bA_1 \mid a$

$A_3 \rightarrow aA_3A_2B_1 \mid bB_1 \mid aA_3A_2 \mid b$

$B_1 \rightarrow aA_3A_2B_1A_1A_3A_2A_3B_1 \mid bB_1A_1A_3A_2A_3B_1 \mid aA_3A_2A_1A_3A_3A_2B_1 \mid bA_1A_3A_2B_1 \mid aA_3A_3A_2B_1 \mid aA_3A_2B_1A_1A_3A_3A_2 \mid bB_1A_1A_3A_3A_2 \mid aA_3A_2A_1A_3A_3A_2 \mid bA_1A_3A_3A_2 \mid aA_3A_3A_2$

Ans

EQUIVALANCE of Two Finite Automata

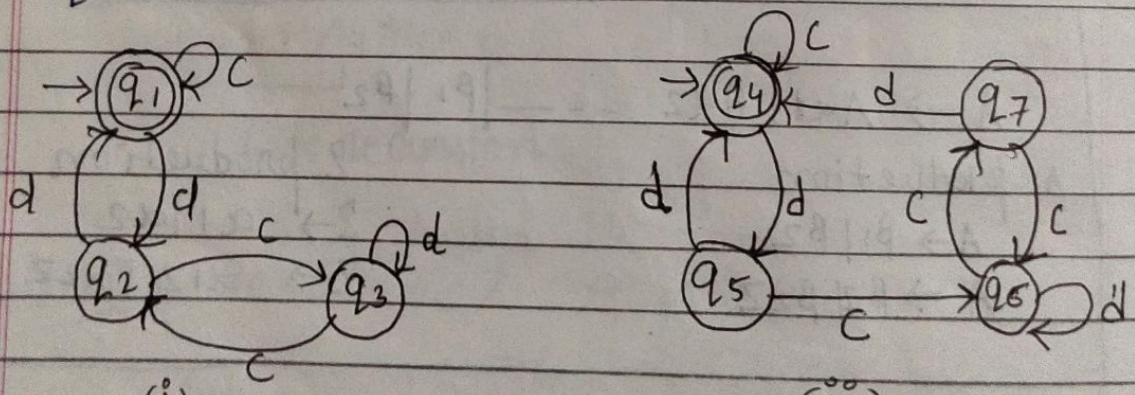
Steps to identify equivalence:-

① For any pair of states $\{q_i, q_j\}$ the transition for input $a \in \Sigma$ is defined by $\{q_a, q_b\}$ where $\delta\{q_i, a\} = q_a$ & $\delta\{q_j, a\} = q_b$

The two automata are not equivalent if for a pair $\{q_a, q_b\}$ one is INTERMEDIATE state and the other is final state.

② If initial state is final state of one automata then in second automaton also Initial State must be final state for them to be equivalent.

Eg:-



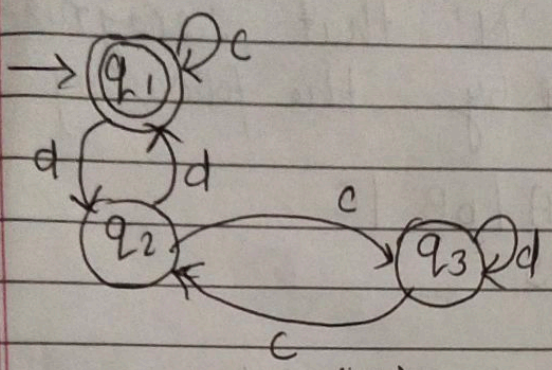
(i) (ii)

States	c	d
(q_1, q_4)	(q_1, q_4) FS FS	(q_2, q_5) IS IS
(q_2, q_5)	(q_3, q_6) IS IS	(q_1, q_4) FS FS
(q_3, q_6)	(q_2, q_7) IS IS	(q_3, q_6) IS IS
(q_2, q_7)	(q_3, q_6) IS IS	(q_1, q_4) FS FS

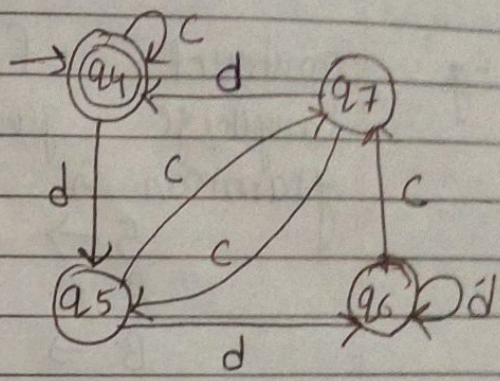
(i) & (ii) FA are Equivalent.

Page No.
Date / /

Q. Find out whether the following automata are equivalent or not.



(A)



(B)

State	c	d
{q1, q4}	{q1, q4} FS FS	{q2, q5} IS IS
{q2, q5}	{q3, q7} IS IS	{q1, q6} FS IS

{q1, q6} both states are different when we read input 'd' from {q2} reach final state {q1} & read input 'd' from {q5} reach initial state which is not same. Stop here.

(A) and (B) are not equivalent