CKB

17/2/2020

III - Sessional

* Turing Machine :-

=> The most general model of computation
=> Developed by Turing in 1936.

× $L_1 = \{ a^N b^N c^N \mid a, b, c \in \Sigma \}$
↳ It cannot be accepted by PDA.

$L_2 = \{ SS \mid S \in \{a, b\}^* \}$

abb abb ← It cannot be accepted by Push down Automata.

=> The action of human computer comprises of examining the current symbol, ~~ears~~ erasing it and looking at some other part of the paper.

=> The corresponding machine actions are:
(i) Changing the current symbol &
(ii) Moving ahead to the left or right
(iii) Changing the state.

"Church Turing Thesis"

Definition :-
A turing machine T.M is a five tuple $T = \langle Q\ \Sigma\ \Gamma\ q_0\ \delta \rangle$
↳ Marker Symbols

=> There is only one halting state

CKB

17/2/2020

III - Sessional

* Turing Machine :-

⇒ The most general model of computation
⇒ Developed by Turing in 1936.

× $L_1 = \{ a^N b^N c^N \mid a, b, c \in \Sigma \}$
↳ It cannot be accepted by PDA.

$L_2 = \{ SS \mid S \in \{a, b\}^* \}$

abb abb ← It cannot be accepted by Push down Automata.

⇒ The action of human computer comprises of examining the current symbol, ~~ears~~ erasing it and looking at some other part of the paper.

⇒ The corresponding machine actions are:
(i) Changing the current symbol &
(ii) Moving ahead to the left or right
(iii) Changing the state.

"Church Turing Thesis"

Definition :-
A turing machine T.M is a five tuple $T = \langle Q \ \Sigma \ \Gamma \ q_0 \ \delta \rangle$
↳ Marker Symbols

⇒ There is only one halting state

Q → finite set of states.

Σ → Input alphabet

Γ → The tape alphabel, doesn't contain Δ (blank)

q_0 → Initial state

δ → The state transition function

$$\delta : Q \times (\Gamma \cup \{\Delta\}) \rightarrow (Q \cup \{\overset{\text{halt}}{h_a}, \overset{\text{rejection}}{h_r}\}) \times (\Gamma \cup \{\Delta\}) \times \{L, R, S\}$$

Move to left, move to right, either remain stationary

(tape)

Γ → is semi-infinite in length.

It has the ~~left or hand~~ end/left end but doesn't have right ~~hand~~ and marked off into squares.

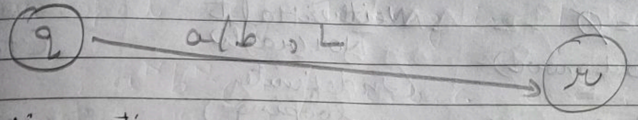
Squares on the tape are numbered as 0, 1, 2, --- from left. This is for our reference only.

A TM doesn't recognize the cells by this no.

A TM has a head which move to the left, right or remains stationary

Example of a move in TM :-

$\delta(q, a) = (r, b, L)$



Configuration of a TM:

It indicates the current status of a TM and is a pair $(q, x\underline{a}y)$

where $q \in Q$,

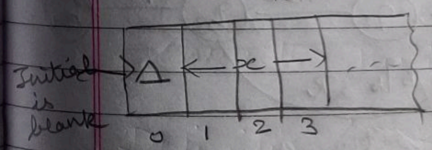
$x, y \in (\Gamma \cup \{\Delta\})^*$

$a \in (\Gamma \cup \{\Delta\})$

Acceptance of a String by Turing Machine (T):

A string $x \in \Sigma^*$ is accepted by T with the following condition:

$$(q_0, \underbrace{\Delta}_{\text{blank}} x) \vdash_T^* (h_a, y\underline{a}z)$$



$y, z \in (\Gamma \cup \{\Delta\})^*$

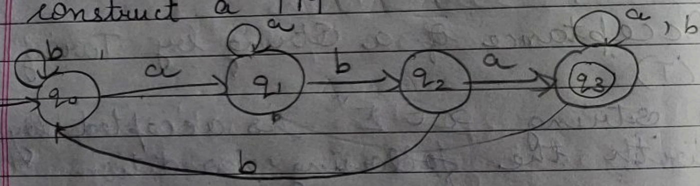
$a \in \Gamma \cup \{\Delta\}$

$L(T) = \{x \in \Sigma^* / x \text{ is accepted by } T\}$

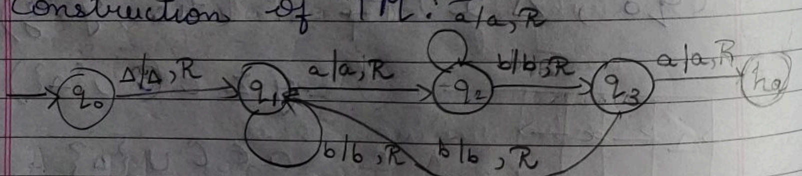
A TM May:

crash	Neither Halt Nor crash	Halt
(i) $\delta(q, a) = \phi$ (unspecified move)	(Indefinite looping)	(Halting state is reached)
(ii) Invalid Operation ex. Moving head to the left of the 0th cell.	$\Rightarrow$ String neither accepted nor rejected	$\Rightarrow$ String accepted
$\Rightarrow$ String rejected		

Ex. for $L = \{ x \in \{a, b\}^* / x$ contains "aba" $\}$ construct a TM



Construction of TM:

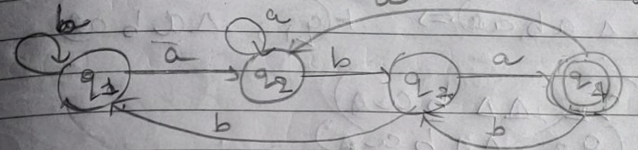


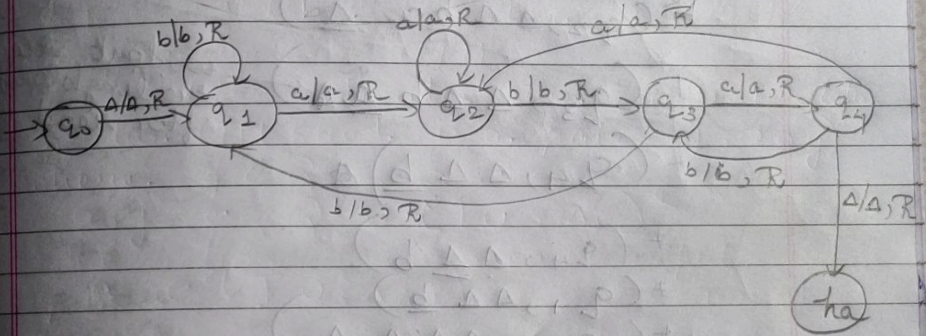
$\Delta/\Delta \rightarrow \Delta$ is replaced with Δ.

$\Rightarrow \delta(q, \Delta) \rightarrow$ if found then, rejection state.

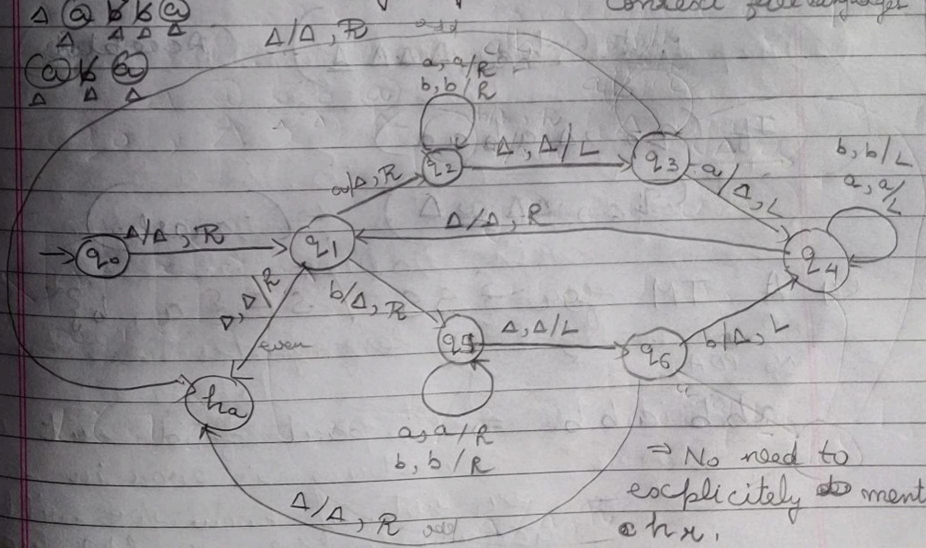
Δ bbbb $\Delta\Delta$

2. $L = \{ x \in \{a, b\}^* / x$ ends with "aba" $\}$





19/2 3. Construct a Turing Machine for accepting the language of PALINDROME
context free languages



$\Rightarrow$ No need to explicitely mention oths.

Execution Trace :-

$(q_0, \Delta aba\Delta) \Rightarrow \vdash (q_1, \Delta \underline{a} ba\Delta)$

$\vdash (q_2, \Delta\Delta \underline{b} a\Delta)$

$\vdash (q_2, \Delta\Delta b \underline{a} \Delta)$

$\vdash^* (q_2, \Delta\Delta ba \underline{\Delta})$

$\vdash (q_3, \Delta\Delta b \underline{\overset{a}{\Delta}} \Delta)$ ← don't show trailing blank after read write head

$\vdash (q_4, \Delta\Delta \underline{b}) \Delta$

$\vdash (q_4, \Delta \underline{\Delta} b)$

$\vdash (q_1, \Delta\Delta \underline{b})$

$\vdash (q_5, \Delta\Delta\Delta \underline{\Delta})$

$\vdash (q_6, \Delta\Delta \underline{\Delta})$

$\vdash (h_a, \Delta\Delta\Delta \underline{\Delta})$ (Accepted)

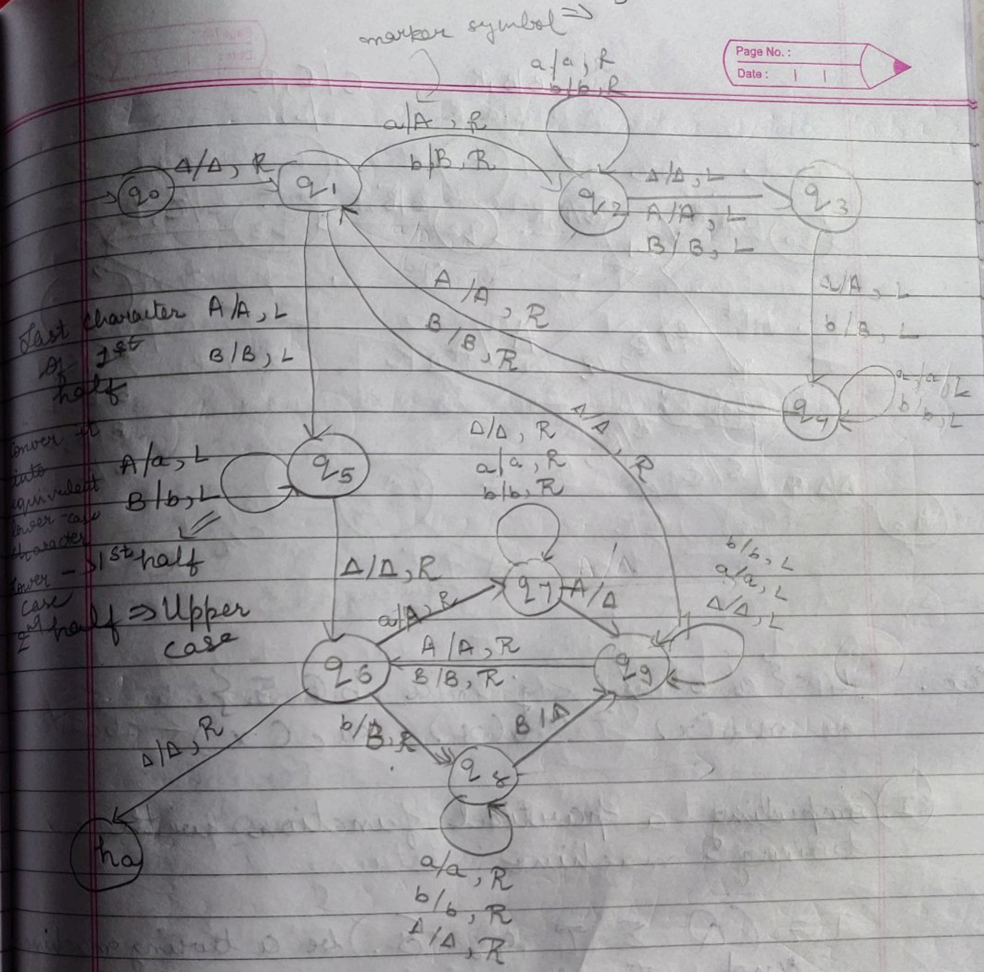
If $(q_0, \Delta abaa)$

$\vdash (q_6, \Delta\Delta \underline{a} \Delta)$ ← (crash)

*4 A TM for $\{ SS \mid S \in \{a, b\}^* \}$

No context free language is also there
No PDA

abb abb ⇒ 1st part find out whether string is even in length or not. If even we will accordingly place the R/W head

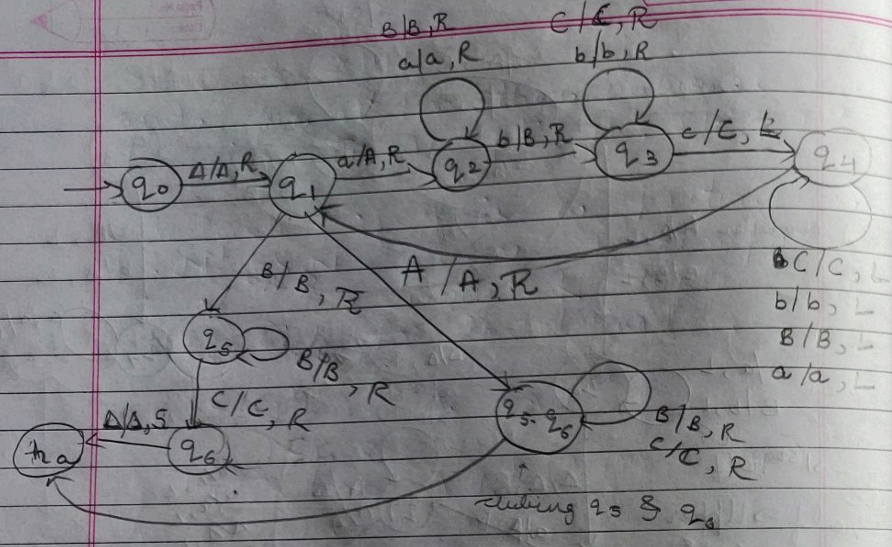


Ex: $L = \{ a^N b^N c^N \mid a, b, c \in \Sigma, N \geq 0 \}$

Δ aabb cc

A A B B C C ⇒ A A B B C $\underline{C}$

A A skip B B skip C C



ex:- $L = \{ a^i b^j \mid i < j,\ a, b \in \Sigma \}$
or $n_a(x) = n_b(x) = n_c(x)$ ← abcabc.

(I) Computing a partial function with a turing machine:

$T = (Q\ \Sigma\ \Gamma\ q_0\ \delta)$ be a turing machine and f be a partial function on Σ^* with values in Γ^*.

We say that T computes f, for $\forall x \in \Sigma^*$, if f is defined, conditions are:
$f : \Sigma^* \to \Gamma^*$

$(q_0, \Delta x)$
$\vdash^*_T (h_a, \Delta f(x))$

and no other x belonging to Σ^* is accepted by T

If f is a partial function on Σ^* to the power k with values in Γ^*, T computes f if for (here it is taking k no. of strings) every k tuple $(x_1, x_2, \ldots x_k)$ at which f is defined, $(q_0, \Delta x_1 \Delta x_2 \Delta \ldots \Delta x_k)$
$\vdash^*_T (h_a, \Delta f(x_1, x_2, \ldots x_k))$ that configuration and no other I/P that is a k tuple of strings is accepted by T.

(II) Computing a numerical function.

Let $T = (Q, \{1\}, \Gamma, q_0, \delta)$ be a turing machine.
If f is a partial fn. from $(N^k \to N)$, T computes f if $\forall n$ for which f is defined,
~~and for~~
$(q_0, \Delta 1^n) \vdash^*_T (h_a, \Delta 1^{f(n)})$

$n=5 \Rightarrow \Delta 11111$
and for $\forall$ ~~no~~ other natural no. n
T fails to accept the I/P : 1^n

Similarly, if f is a partial fn $f: N^k \to N$ then T computes f if for every k tuple $(n_1, n_2, \ldots n_k)$ at which f is defined,

$$(q_0, \Delta 1^{n_1} \Delta 1^{n_2} \Delta \ldots \Delta 1^{n_k}) \vdash_T^* (h_a, \Delta 1^{f(n_1, n_2, \ldots, n_k)})$$

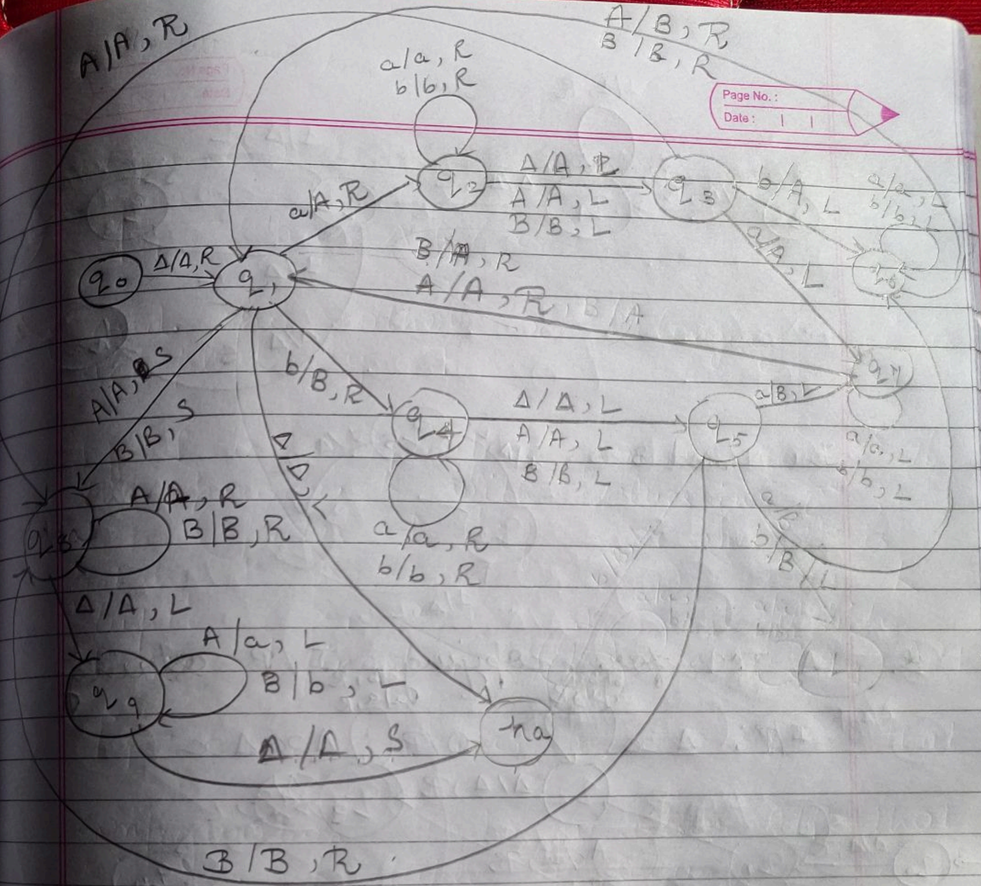
and T fails to accept, if the I/P is any k tuple at which f is not defined.

* Construct a TM for computing reverse of a string.

rev(x) or $x^r : \Sigma^* \to \Sigma^*$

abb ⇒ bba $\quad (q_0, \Delta x) \vdash_T^* (h_a, \Delta x^r)$

$(q_0, \Delta abb) \vdash (q_1, \Delta \underline{a} bb)$
$\vdash (q_2, \Delta A \underline{b} b) \vdash (q_3, \Delta Abb \underline{\Delta})$
$\vdash (q_3, \Delta Ab \underline{b}) \vdash (q_6, \Delta A \underline{b} A)$
$\vdash (q_1, \Delta \underline{B} b A) \vdash (q_4, \Delta BB\underline{A})$
$\vdash (q_5, \Delta B\underline{B}A) \vdash (q_8, \Delta BBA\underline{\Delta})$
$\vdash (q_9, \Delta \underline{B} BA) \vdash (q_9, \underline{\Delta} bbA) \vdash (h_a, \ldots$

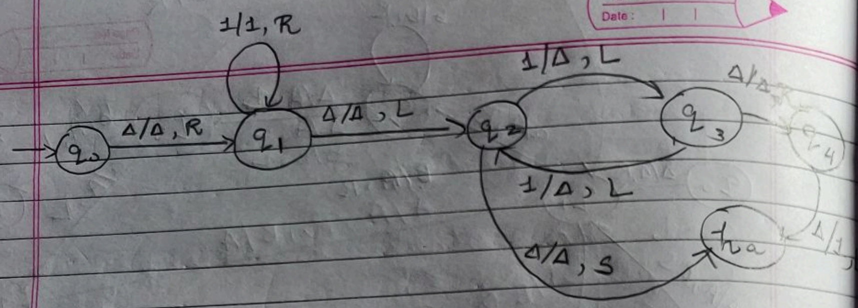


* A Turing Machine to compute N mod 2

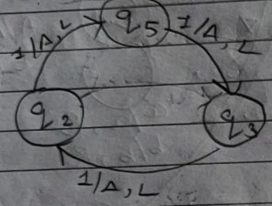
O/P : $\underline{\Delta}$ 1 if N is odd
$\underline{\Delta}$ if N is even

Δ 1 1 1 1 1 Δ

So initially ignoring all 1's reach end & then start erasing pairs of 1's

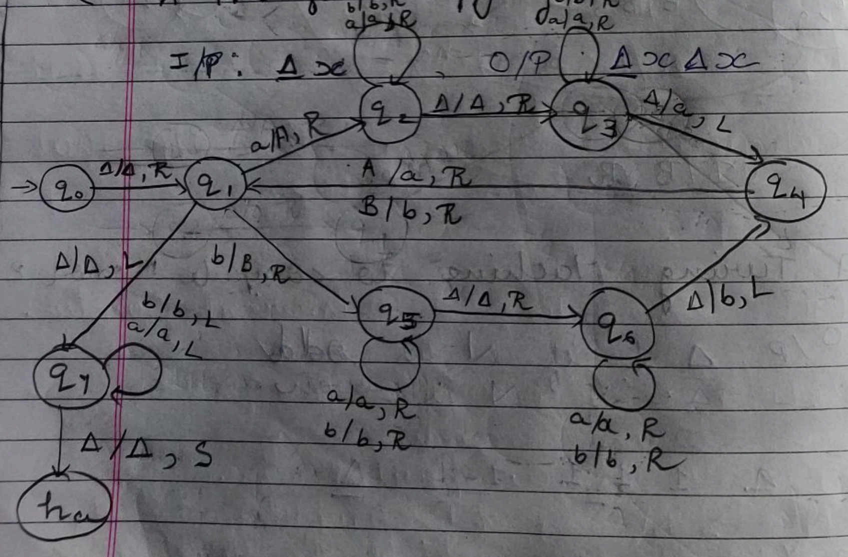


For N mod 3 ⇒



* A TM for copying I/P string

I/P : $\underline{\Delta} x$, O/P $\underline{\Delta} x \Delta x$



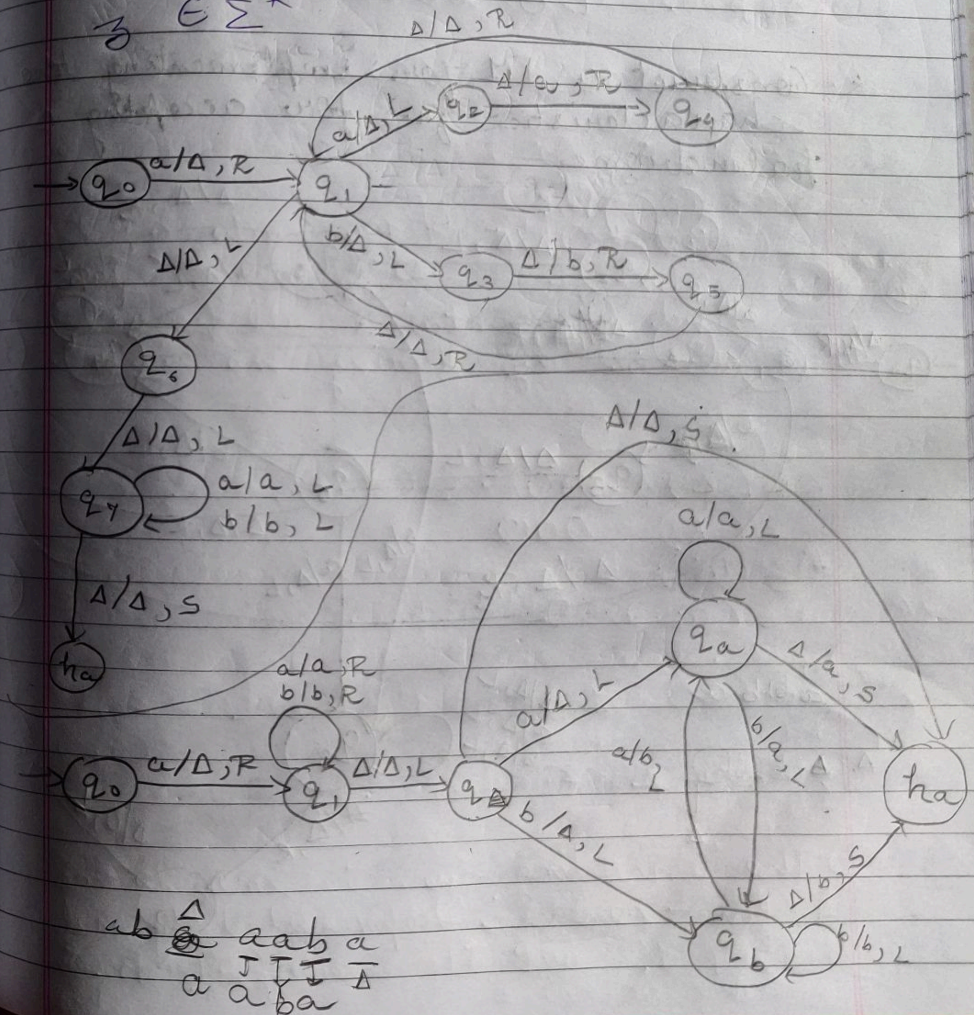
abΔ b b

* A TM for deleting a symbol :-

$(q_0, y\underline{a}z) \vdash^{*} (h_a, y\underline{z})$

$y \in (\Sigma \cup \{\Delta\})$, $a \in (\Sigma \cup \{\Delta\})$

$z \in \Sigma^{*}$

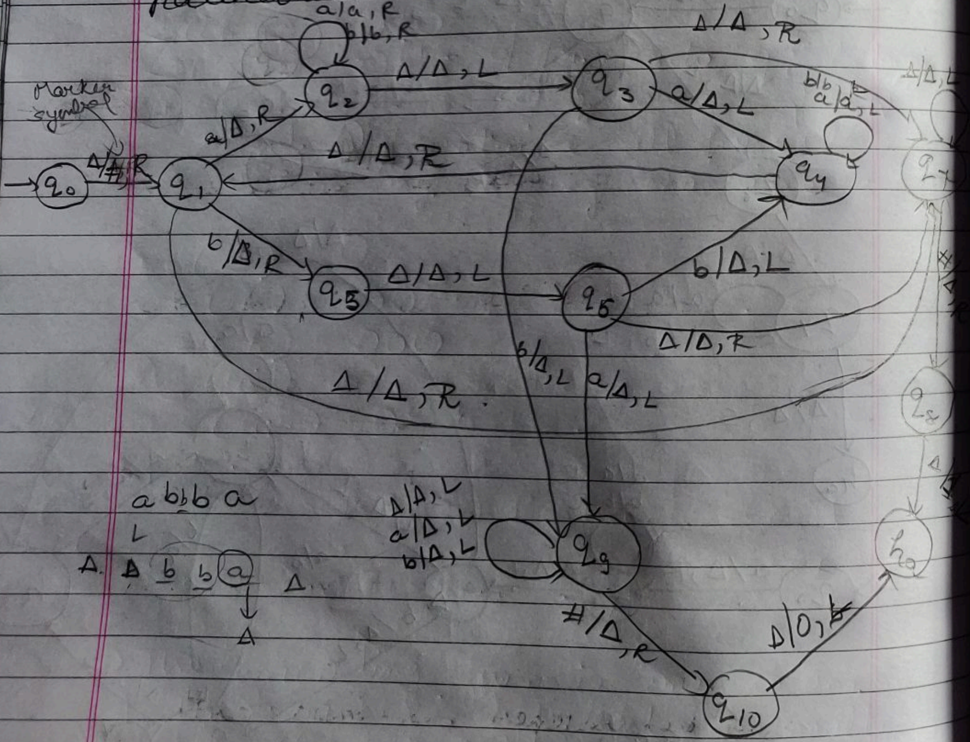


Characteristic function of a set:

Let $L \subseteq \Sigma^*$ $\quad X_L$ ← The characteristic fn. of L

$$X_L(x) = \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{otherwise} \end{cases}$$

* Construct a TM for implementing the characteristic fn for accepting a palindrome.



Universal Turing Machine :-

↳ General Purpose

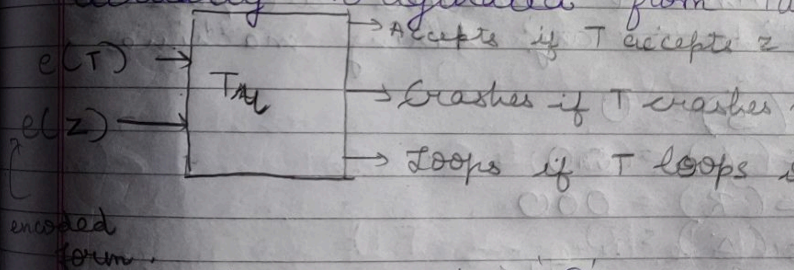
⇒ It's a model of general purpose computer.

⇒ A TM can execute 1 algorithm only, whereas a computer can execute any program.

⇒ Earlier electronic computers were very diff. to reprogram.

⇒ It was used in a better way when people started loading a program as a part of I/P.

⇒ The idea of stored program computer actually originated from Turing.



Encoding Function 'e'

$s(\Delta) = 0$ (It is sequence of 0's separated by 1)

$s(a_i) = 0^{i+1}$ (for $a_i \in S = \{a_1, a_2, \dots\}$),

such that

s.t. for any Turing Machine T, ($\in S$)

i is the integer associated with the symbol 'a'

$e(T) = e(m_1)1\ e(m_2)1 \ldots e(m_k)1$
where the turing machine T has $m_1, m_2, \ldots, m_k$ as distinct moves.

A typical move in a Turing Machine is
$\delta(p, a) = (q, b, D)$

$e(m) = S(p)1\ S(a)1\ S(q)1\ S(b)1\ S(D)1$

$S(h_a) = 0$
$S(h_r) = 00$
$S(q_i) = 0^{i+2}$ [for each q_i belonging to Q where $Q \subseteq Q'$,
And Q' contains all possible states in any turing machine.

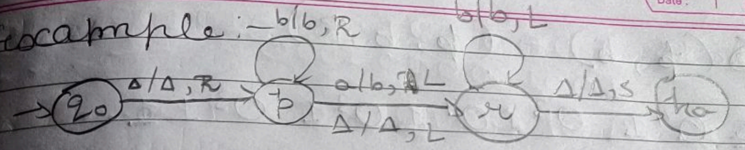
$S(S) = 0$
$S(L) = 00$
$S(R) = 000$
$S(\Delta) = 0$
$S(a_i) = 0^{i+1}$

I/P string which is needed to be processed

$e(z) = 1\ S(z_1)1\ S(z_2)1\ S(z_3)1 \ldots S(z_k)1$
where $z = z_1 z_2 z_3 \ldots z_k$

⇒ The encoding fun. is one to one mapping. A $T(n)$ can be represented by multiple 0's & 1's but it corresponds to ~~only~~ atmost 1 turing machine.

Example:-



Number assigned:

0^{i+1}

Symbols $\{a \to 1,\ b \to 2\}$

$S(\Delta) \to 0$
$i=1$ $S(a) = 00$
$i=2$ $S(b) = 000$

States $\{p \to 2,\ q_0 \to 1,\ r \to 3\}$

$S(h_a) = 0$
$S(h_r) = 00$
$S(q_0) = 0^3 = 000$
$S(p) = 0^4 = 0000$
$S(r) = 0^5 = 00000$

$S(S) = 0$
$S(L) = 00$
$S(R) = 000$

There are 6 distinct moves
$\delta(q_0, \Delta) = (p, \Delta, R)$
$\delta(p, b) = (p, b, R)$
$\delta(p, a) = (r, b, L)$
$\delta(p, \Delta) = (r, \Delta, L)$
$\delta(r, b) = (r, b, L)$
$\delta(r, \Delta) = (h_a, \Delta, S)$

$\delta(p, a) = (q, b, D)$
$e(m) = S(p)1\ S(a)1\ S(q)1\ S(b)1\ S(D)1$

1) Only for 1st move:- $\delta(q_0, \Delta) = (p, \Delta, R)$
$\Rightarrow e(m_1) = 0^4 1\ 0 1\ 0^4 1\ 0 1\ 0^3 1$

2. $\delta(p, b) = (p, b, R)$

$e(m_2) = 0^4 1\ 0^3 1 0^4 1\ 0^3 1\ 0^3 1$

3. $\delta(p, a) = (r, b, L)$

$e(m_3) = 0^4 1\ 0^2 1\ 0^5 1\ 0^3 1 0^2 1$

4. $\delta(p, \Delta) = (r, \Delta, L)$

$e(m_4) = 0^4 1\ 0\ 1\ 0^5\ 1\ 0\ 1\ 0^2\ 1$

5. $\delta(r, b) = (r, b, L)$

$e(m_5) = 0^5 1\ 0^3 1\ 0^5 1\ 0^3 1\ 0^2 1$

6. $\delta(r, \Delta) = (h_a, \Delta, S)$

$e(m_6) = 0^5 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0^4$

$e(T) = e(m_1) 1\ e(m_2) 1 \ldots e(m_k) 1$

Architecture of TU: Universal Turing Machine

It contains 3 Tapes:-

Tape I, Tape II & Tape III

$e(T)e(z)$ → Tape 1 — I/P to TU

Tape II → working Tape (actual simulation is carried out on Tape II)

Tape I: It is the I/P & O/P tape. Initially it contains the I/P string $e(T)e(z)$

Tape 2: It's working tape during simulation of T.

Tape 3: Encoded form of state T is currently in

Initialization steps:

1. To move the string $e(z)$ from Tape 1 to tape 2 (when there are 3 consecutive 1s)
2. To move the encoded form of T's initial state from begining of tape 1 to tape 3.
3. Simulation starts now. The three tape heads are all in square 1.
4. The next move of T is determined by T's state (encoded in Tape 3) and current symbol on T's tape, whose encoding starts in current position on tape 2. In order to simulate this move Tu must search Tape 1 for the 5 Tuple whose 1st & 2nd parts match this state/I/P combination.
5. Once the tuple is found, the last 3 parts tell Tu how to simulate the move.

Tape 1: Δ 0001 01 0000 101000 11 0000 ...
100100000 1000 10011) ...

Tape 2: Δ 01 001 0 01 0001 Δ ...

Tape 3: Δ 0000 Δ ← current state
↳ Encoded form

Tu will

$\delta(p, a) = (r, b, L)$

↓

Tape 2: Δ 01 0 0 1 001 0001 Δ

Tape 3: Δ 00000 Δ
r

Tu is now ~~simulate~~ ready to simulate the next move.

Finally when Tu simulates the halting move of T, assuming that T halts, it does like this:
After Tape 2 is changed appropriately it erases Tape 1 and puts the content of Tape 2 to Tape 1, as tape 1 is the O/P Tape as well, and halts.
(The halting state is detected by Tu when it sees that the 3rd part on Tape 1 is 0, corresponding to $S(h_a) = 0$)

$S \rightarrow \wedge / aSb / \cancel{BSb} / aaSb$

$S \rightarrow Sb$

$aaSb$

$a\,aaSbb$

$aaabb$

Syllabus :- Ch - 3 (3.4) $\uparrow$ Th: 5.2 a

Ch - 5 (5.2, 5.3).

Ch - 6 (6.1).

III - Sessional

1. $L = \{a^i b^j c^k \mid j > i+k,\ i,k \geq 0\}$

$j > i + k$

$j = i + k + m, \quad m \geq 0$

$\cancel{i+k+m}$

$a^i b^{i+k+m} c^k$

$= \underbrace{a^i b^i}_{L_1} \underbrace{b^m}_{L_2} \underbrace{b^k c^k}_{L_3}$

(Concatenation of 3 Languages)

Page No.:
Date:

For $L1 : \{a^i b^i \mid i \geq 0\}$

$S_1 \rightarrow aS_1b \mid \wedge$

For $L2 = \{b^m \mid m > 0\}$

$S_2 \rightarrow bS_2 \mid b$

For $L3 : \{b^k c^k \mid k \geq 0\}$

$S_3 \rightarrow bS_3c \mid \wedge$.

$S \rightarrow S_1 . S_2 . S_3$

$S \rightarrow S_1 S_2 S_3 \rightarrow aS_1b . S_2 . S_3 \rightarrow$

$abS_2S_3 \rightarrow abbS_3 \rightarrow abbbS_3c$

$\rightarrow abbbc$.

* $L = \{a^i b^j c^k \mid i > j + k,\ j, k \geq 0\}$

$i = m + j + k$.

$\therefore a^{m+j+k} . b^j c^k$

$\{a^m a^j a^k b^j c^k\} \quad X = \underbrace{a^m}_{L_3} \underbrace{a^k \underbrace{a^j b^j}_{L_1} c^k}_{L_2}$

a^{m+j+k}

Page No.:
Date:

$L \rightarrow L_3 . L_2$
$\quad\quad\quad\quad\downarrow$
$\quad\quad\quad\quad L_1$

$L_1 = \{a^j b^j \mid j \geq 0\}$

$S_1 \rightarrow aS_1b \mid \wedge$

$L_2 = \{a^k L_1 c^k\}$

$S_2 \rightarrow aS_2c \mid S_1$

$S_3 \rightarrow aS_3 \mid \wedge$

$\therefore S \rightarrow S_3 . S_2$

$aaabc \Rightarrow$

$S \rightarrow S_3 . S_2 \rightarrow aS_2 \rightarrow aaS_2c \rightarrow aaS_1c \rightarrow$
$aaaS_1bc \rightarrow aaabc$

$a^k a^j a^m b^j c^k$

$S_3 \rightarrow aS_3 \mid a$

$S_2 \rightarrow aS_2b \mid S_3$

$S \rightarrow aSc \mid S_2$

$S \rightarrow aSc \rightarrow aS_2c \rightarrow aaS_2bc \rightarrow aaS_3bc$
$\rightarrow aaabc$

5. $L = \{a^i b^j c^k \mid k > i + j,\ i, j \geq 0\}$

$k = m + i + j$

$\therefore a^i b^j c^m c^j c^i$

$S_3 \to c \mid aS_3$

$S_2 \to S_3 \mid bS_2c$

$S_1 \to S_2 \mid aS_1c$

~~$S \to S_1 S_2$~~

4. $L = \{a^i b^j c^k \mid i = j \text{ or } i = k\}$

$L_1 = \{a^i b^i c^k \mid k, i \geq 0\}$

$L_2 = \{a^k b^j c^k \mid k, j \geq 0\}$.

$L = L_1 \cup L_2$

$S \to S_1 \mid S_2$
$S_1 \to AB$
$A \to aAb \mid ab \mid \wedge$
$B \to cB \mid c \mid \wedge$

$S_2 \to aS_2c \mid S_3 \mid \wedge$
$S_3 \to bS_3 \mid \wedge$

"abc" $\to S \to S_1 \to AB \to aAbB \to abB \to abc$

6. $L = \{a^i b^j c^k \mid j = k \text{ or } i = k\}$

$L = \{a^i b^j c^i \mid i, j \geq 0\} \cup \{a^i b^j c^j \mid i, j \geq 0\}$.

$S \to S_1 \mid S_2$

$S_1 \to CA$
$A \to \wedge \mid bAc$.
$C \to aC \mid \wedge$.

$S_2 \to aS_2c \mid S_3 \mid \wedge$
$S_3 \to bS_3 \mid \wedge$

$abbcc \Rightarrow S \to S_1$
$\to CA$
$\to aCA$
$\to aA$
$\to abAc$
$\to abbAcc \to abbcc$.

6. $L = \{a^i b^j c^m d^n \mid i = j \text{ or } m = n\}$

$L_1 = \{a^i b^i c^m d^n \mid i, m, n \geq 0\}$

$L_2 = \{a^i b^j c^m d^m \mid i, j, m \geq 0\}$

~~$L_3 = \{a^i b^i c^n d^m \mid i, n \geq 0\}$~~

H.W

(1) $\{a^i b^j c^m d^n \mid i+m = n+j\}$

(2) $\{a^i b^j c^k \mid j \neq i+k\}$

$\Downarrow$

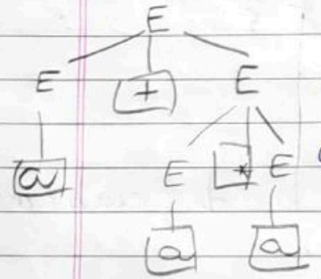
$\{a^i b^j c^k \mid j > i+k\} \cup \{a^i b^j c^k \mid j < i+k\}$

18/2/2020

Ambiguous Grammar:

$E \to E+E \mid E*E \mid a$

"a+a*a" (Left-Most Derivation)



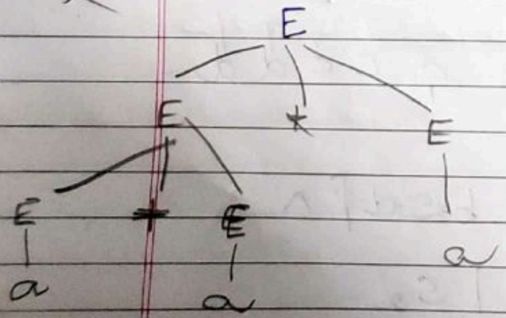
E

$\underline{E}+E$

a+E*E

a+a*a

Left ~~Right~~ Most derivation:



E

E*E

E+E*E

a+E*E

a+a*E

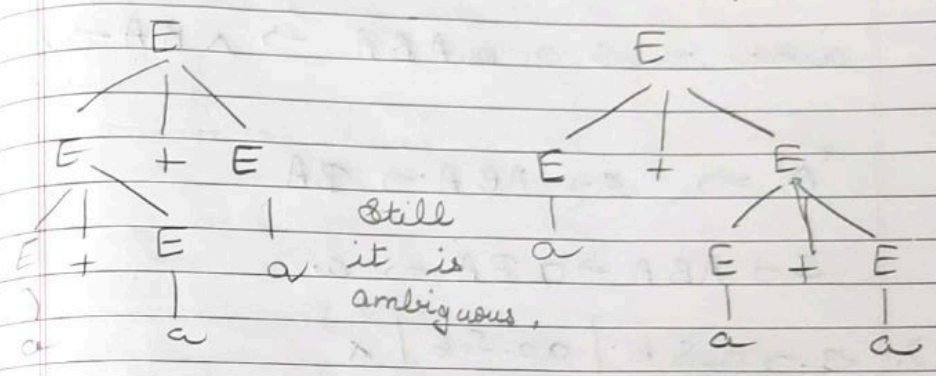
$\boxed{a+a*a}$

For 1 string 2 left most derivations are possible, so it cannot be used in any programming language

"a+a+a"

Show that this is ambiguous:

→ $S \to a \mid Sa \mid bSS \mid SSb \mid SbS$



Still it is ambiguous.

bbbaaaa

$S \to bSS \to bbSSS \to bbbSSSS$

∴ ⇒ bbbaaaa.

abaa:-

$SbS \to abSa \to abaa$.

abaa

$\to Sa \to SbSa \to abaa$

$S \to SS \mid a \mid b$

$SS \to SSSS$

aba →

$S \to SS \to aS \Rightarrow aSS \to aba$

$S \to SS \to SSS \to aba$

2) $S \to ABA$
$A \to aA \mid \wedge$
$B \to bB \mid \wedge$

'a' → $S \to ABA \to \wedge BA \to \wedge\wedge A \to a$

~~'A' ⇒ S → ABA ⇒ BA~~

$S \to ABA \to aBA \Rightarrow a$

3) $S \to aSb \mid aaSb \mid \wedge$
(aaabb).

$S \to aaSb \Rightarrow aa\,aSbb \to aaabb$

$S \to aSb \Rightarrow a\,aaSbb \to aaabb$

4) Dangling "Else" Problem :-

$S \to iEtSeS \mid iEtS \mid a$

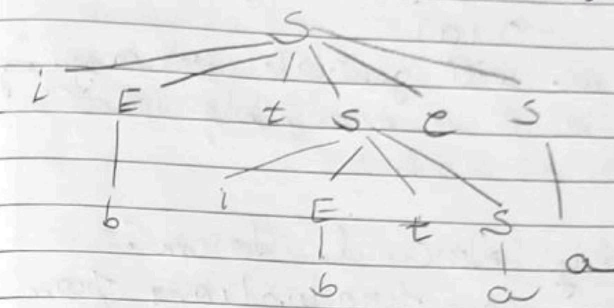
$E \to b$

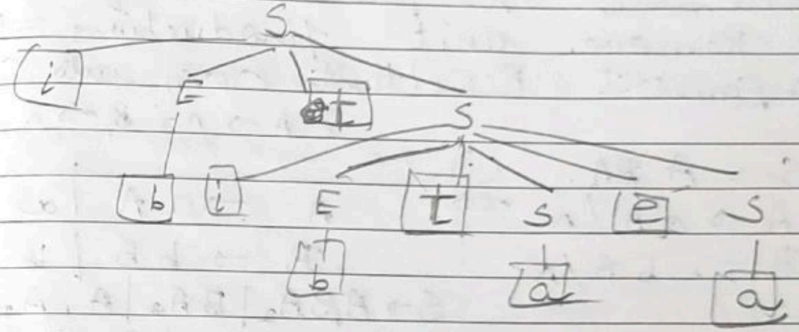
i → if   E → indicates condition
t → then   S → set of statements   e → else

1st → "ibtibtaea"
if E then
  if E then
    a
  else
    a

If E then
  if E then
    a
else
  a

$S \to iEtSeS \Rightarrow iEtiEtSeS$



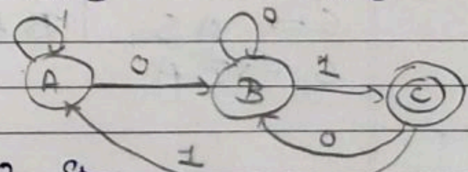


25/2

Regular Expression → CFG

$(0+1)^* 01$: $A \to 0$,
$B \to 1$,
$C \to 0/1$, $D \to CD/\wedge$
$E \to AB$ $S \to DE$

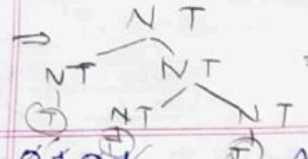
FA → CFG / RG



+ even RG is also a CFG
states = NTs

$A \to 0B$ $B \to 0B$ $C \to 0B$
$A \to 1A$ $B \to 1C$ $C \to 1A$
$B \to 1$ (To terminate the process)

TTT ⇒



⇒ easy to identify number of steps ⇒ $NT \to NT.NT$
$NT \to T$

0101   A , 0B , 01C , 010B.
0101

Always we will generate a language $L - \{\Lambda\}$
⇒ So now if A is accepting state make it non-accepting

* Chomsky Normal Form:-

→ Remove Λ-productions from Grammar
→ Remove unit productions ⇒ $NT \to NT$
→ Convert Resultant CFG into CNF

$\{A \to \Lambda,\ B \to \Lambda\}$

$S \to ABA$
$A \to aA / \Lambda$
$B \to bB / \Lambda$

(i) (From $S \to \Lambda$ there is indirect transaction) ⇒

$$\boxed{\begin{array}{l} A \to aA \mid a \\ B \to bB \mid b \\ S \to A_1BA_2 \mid BA_2 \mid A_1A_2 \mid A_1B \mid A_1 \mid A_2 \mid B \end{array}}$$

(ii) $A \to aA \mid a$
$B \to bB \mid b$
$S \to ABA \mid BA \mid AA \mid AB \mid aA \mid bB \mid a \mid b$

We require single T or NT NT

(iii) $X_a \to a$ $X_b \to b$
$A \to X_aA \mid a$ $B \to X_bB \mid b$

$S \to AS_1 \mid BA \mid AA \mid AB \mid X_aA \mid X_bB \mid a \mid b$
$S_1 \to BA$

Advantage of Chomsky NF ⇒ There will always be binary tree → (better to manage as you can count to no. of steps for derivation)

$S \to AACD$
$A \to aAb / \Lambda$
$C \to aC / a$
$D \to aDa \mid bDb / \Lambda$

(i) ⇒
$$\boxed{\begin{array}{l} A \to aAb \mid ab \\ D \to aDa \mid bDb \mid aa \mid bb \\ S \to AACD \mid C \mid CD \mid ACD \mid AAC \mid AC \end{array}}$$

(ii) $S \to AACD \mid ACD \mid AAC \mid CD \mid AC \mid aC \mid a$
$A \to aAb \mid ab$
$C \to aC \mid a$
$D \to aDa \mid bDb \mid aa \mid bb$

(iii) $X_a \to a$
$X_b \to b$
$C \to X_aC \mid a$
$A \to X_aA_1 \mid X_aX_b$
$A_1 \to AX_b$
$D \to X_aD_1 \mid X_bD_2 \mid X_aX_a \mid X_bX_b$
$D_1 \to DX_a$
$D_2 \to DX_b$

$S \to AS_1 \mid AS_2 \mid AS_3 \mid CD \mid AC \mid X_aC \mid a$
$S_1 \to AS_2$
$S_2 \to CD$
$S_3 \to AC$

$S \to SS \mid (S) \mid \Lambda$.

Step 1 :- $\{S \to \Lambda\}$

$S \to S_1 S_2 \mid S_1 \mid S_2 \mid (S) \mid ()$

$\boxed{S \to SS \mid S \mid (S) \mid ()}$.

Step 2: $S \to SS \mid (S) \mid ()$.

Step 3:- $X_a \to ($.

$X_b \to)$

$S \to SS \mid X_a A \mid X_a X_b$

$A \to S X_b$

Ex. 6.33

$S \to AaA \mid CA \mid BaB$

$A \to aaBa \mid CDA \mid aa \mid DC$

$B \to bB \mid bAB \mid bb \mid aS$

$C \to Ca \mid bC \mid D$

$D \to bD \mid \Lambda$.

Step 1

$D \to bD \mid b$

$C \to Ca \mid bC \mid a \mid b \mid$

$A \to aaBa \mid CD \mid DA \mid CA \mid aa \mid CDA$

H.W 1) $\{a^i b^j c^m d^n \mid i+m = n+j\}$

2) $\{a^i b^j c^k \mid j \neq i+k\}$

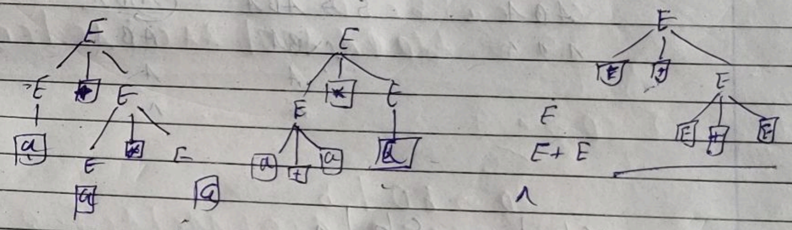
$a^i b^j c^k \mid j > i+k\} \cup \{a^i b^j c^k \mid j < i+k\}$

Ambiguous Grammar

$E \to E+E \mid E*E \mid a \mid \wedge$

"a + a * a"

a+a+a



$S \to a \mid Sa \mid bSS \mid SSb \mid SbS$

abaa

abbaa   abbaa

b   abb

S	S
SbS	Sa
abSa	SbSa
abaa	abaa

S → SS | a | b

S S
SSS

aab

S	S
Sb	aS
SSb	aSS
aab	aab

S → ABA
A → aA | ^
B → bB | ^

"^"

ABA
^^^

S → ABA
^BA
A
^

ABA
AB
A
^

* S → aSb | aaSb | ^

aaabb

aSb	aaSb
aaaSbb	aaaSbb
aaabb	aaabb

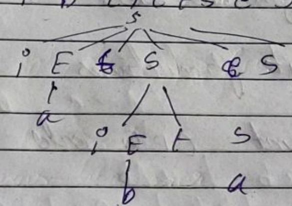
Dangling "Else" Problem

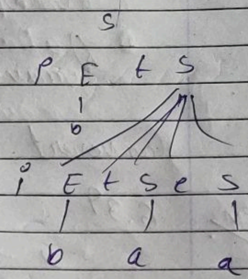
S → iEtSeS | iEtS | a

ibt ibtaea

S → iEtSeS

i b t i E t s e S



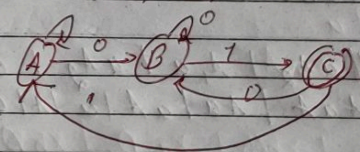


Regular Exp → CFG

$(0+1)^* 01$

A → 0
B → 1
C → 0|1
D → CD|^
E → AB
S → DE

FA → CFG / RG



Even RG is also a CFG state

A → 0B	B → 0B	C → 0B
A → 1A	B → 1C	C → 1A
	B → 1	

Always generate L - {^}

Chomsky Normal form

Remove Λ-productions

Remove unit productions

Convert Resultant CFG into CNF

ex 6.39

$S \to AaA \mid CA \mid BaB$

$A \to aaBa \mid CDA \mid aa \mid DC$

$B \to bB \mid bAB \mid bb \mid aS$

$C \to Ca \mid bC \mid D$

$D \to bD \mid \Lambda$

1) $\{D \to \Lambda\}$

$D \to bD \mid b$

$C \to Ca \mid bC \mid b \mid$

~~$A \to bC \mid DC \mid aa \mid CD$~~

$A \to aaBa \mid DA \mid CA \mid A \mid aa \mid D \mid C$

$S \to A_1aA_2 \mid A_1a \mid aA_2 \mid a \mid A \mid C \mid BaB$

$S \to AaA \mid Aa \mid aA \mid a \mid A \mid C \mid BaB \mid CA$

$S \to ABA$

$A \to aA \mid \Lambda$

$B \to bB \mid \Lambda$

1) Remove null production

$\{A \to \Lambda,\ B \to \Lambda\}$

$\boxed{A \to aA \mid a \quad B \to bB \mid b}$

$S \to A_1BA_2 \mid BA_2 \mid A_1A_2 \mid A_1B \mid A_2 \mid A_1 \mid B$

$\boxed{S \to ABA \mid BA \mid AA \mid AB \mid A \mid B}$

2) $NT \to NT$

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

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

3) $NT \to NT \cdot NT$

$NT \to T$

$X_a \to a$

$X_b \to b$

$A \to X_aA \mid a$

$B \to X_bB \mid b$

$S \to AS_1 \mid BA \mid AA \mid AB \mid X_aA \mid X_bB \mid a \mid b$

$S_1 \to BA$

Date: ____ MON TUE WED THU FRI SAT

$S \to AACD$
$A \to aAb \mid \Lambda$
$C \to aC \mid a$
$D \to aDa \mid bDb \mid \Lambda$

1) Remove Λ $\{A \to \Lambda,\ D \to \Lambda\}$

$$\boxed{\begin{array}{l} A \to ab \mid aAb \\ D \to aa \mid bb \mid aDa \mid bDb \end{array}}$$

$S \to A_1A_2CD \mid A_2CD \mid A_1CD \mid A_1A_2C \mid$
$A_2C \mid A_1C \mid CD \mid C$

$$\boxed{S \to AACD \mid ACD \mid AAC \mid AC \mid CD \mid C}$$

2) $\{ S \to C \}$

$S \to AACD \mid ACD \mid AAC \mid AC \mid CD \mid aC \mid a$

3
$X_a \to xa$
$X_b \to xb$
$C \to X_aC \mid a$
$A \to X_aX_b \mid X_aA_1$
$A_1 \to AX_b$
$D \to X_aX_a \mid X_bX_b \mid X_aD_1 \mid X_bD_2$
$D_1 \to DX_a \mid$
$D_2 \to DX_b$
$S \to AS_1 \mid AS_2 \mid AS_3 \mid CD \mid AC\ X_aC \mid a$
$S_1 \to AS_2$
$S_2 \to CD$
$S_3 \to AC$

Date: ____ MON TUE WED THU FRI SAT

$S \to SS \mid (S) \mid \Lambda$

1) $\{S \to \Lambda\}$

$S \to S_1S_2 \mid S_1 \mid S_2 \mid (S) \mid ()$

$S \to SS \mid S \mid () \mid (S)$

2) $\{S \to S\}$

$S \to SS \mid (S) \mid ()$

3)

$X_a \to ($
$X_b \to)$

$SS \to SS \mid X_aA \mid X_aX_b$
$A \to SX_b$

28/2 Push-Down Automata :-
Stack (Push & Pop Operations)

$L = \{ x / x$ is a palindrome with middle symbol 'c' $\}$

'c'

aca   bacab
bcb   bbcbb
abcba
aacaa

Pop → $\cancel{a}$, Z_0

as soon as c comes check if it is a Pop

$(q_0, abcba, Z_0)$
↓ ①
$(q_0, abcba, aZ_0)$
↓ ③

abcba
b ← Pop
a ← Pop
Z_0

$(q_0, ~~abc~~, baZ_0) \to (q_0, cba, baZ_0)$

$(q_0, abacaba, Z_0)$

	State	I/P	TOS	Move
1)	q_0	a	Z_0	(q_0, aZ_0)
2)	q_0	b	Z_0	(q_0, bZ_0)
3)	q_0	b	a	(q_0, ba)
4)	q_0	a	a	(q_0, aa)
5)	q_0	a	b	(q_0, ab)
6)	q_0	b	b	(q_0, bb)
7)	q_0	c	a	(q_1, a)
~~8)~~	~~q_1~~	~~a~~	~~a~~	~~(q_1, aa)~~
8)	q_0	c	b	(q_1, b)
9)	q_0	c	Z_0	(q_1, Z_0)
10)	q_1	a	a	$(q_1, \wedge)$
11)	q_1	b	b	$(q_1, \wedge)$
12)	q_1	$\wedge$	Z_0	(q_2, Z_0)

Push/pop

Now you are in q_1 state so you cannot push

you don't know which symbol will be there after pop.

(1) $(q_0, c, Z_0) \xrightarrow{9} (q_1, \wedge, Z_0) \xrightarrow{12} (q_2, Z_0)$

$(q_0, ca, Z_0) \xrightarrow{9} (q_1, a, Z_0) \xrightarrow{12} (q_2, a, Z_0)$

Not consumed the whole I/P

2) $(q_0, bbcbb, Z_0) \xrightarrow{2} (q_0, bcbb, bZ_0) \xrightarrow{6} (q_0, cbb, bbZ_0) \xrightarrow{8} (q_1, bb, bbZ_0) \xrightarrow{11} (q_1, b, bZ_0) \xrightarrow{11} (q_1, \wedge, Z_0) \xrightarrow{12} (q_2, Z_0)$

Non-Deterministic

* Push Down Automata for $\{x \in \{a,b\}^* \mid x$ is an odd length Palindrome$\}$

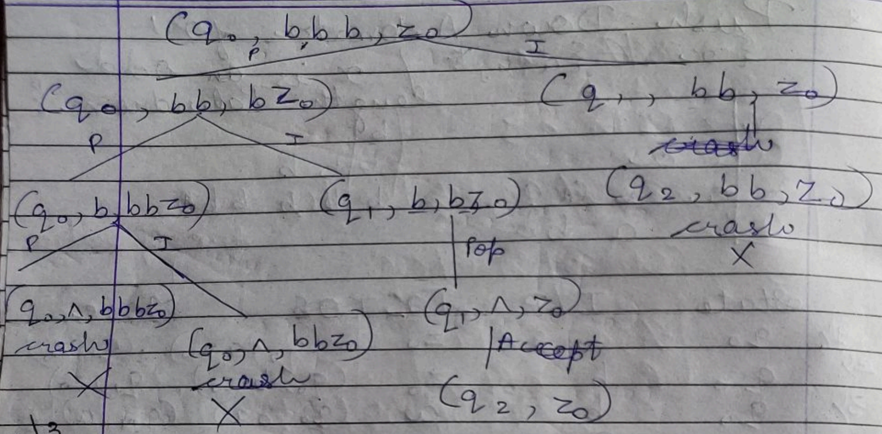
a
a a a
a a a a a
a b a b a

b
b b b
b b b b b

Pop operation

	State	I/P	TOS	Move
1)	q_0	a	Z_0	$(q_0, aZ_0), (q_1, Z_0)$
2)	q_0	b	Z_0	$(q_0, bZ_0), (q_1, Z_0)$
3)	q_0	b	a	$(q_0, ba), (q_1, a)$
4)	q_0	a	a	$(q_0, aa), (q_1, a)$
5)	q_0	a	b	$(q_0, ab), (q_1, b)$
6)	q_0	b	b	$(q_0, bb), (q_1, b)$
7)	q_1	a	a	$(q_1, \wedge)$
8)	q_1	b	b	$(q_1, \wedge)$
9)	q_2	$\wedge$	Z_0	(q_2, Z_0)

(q_0, aba, Z_0)
- Push: (q_0, ba, aZ_0)
  - P: (q_0, a, baZ_0)
    - P: $(q_0, \wedge, abaZ_0)$ crash
    - I: $(q_1, \wedge, baZ_0)$ crash
  - I: (q_1, a, aZ_0)
    - Pop: $(q_1, \wedge, Z_0)$
      - Accept: (q_2, Z_0)
- Ignore: (q_1, ba, Z_0)
  - (q_2, ba, Z_0) crash



13/3

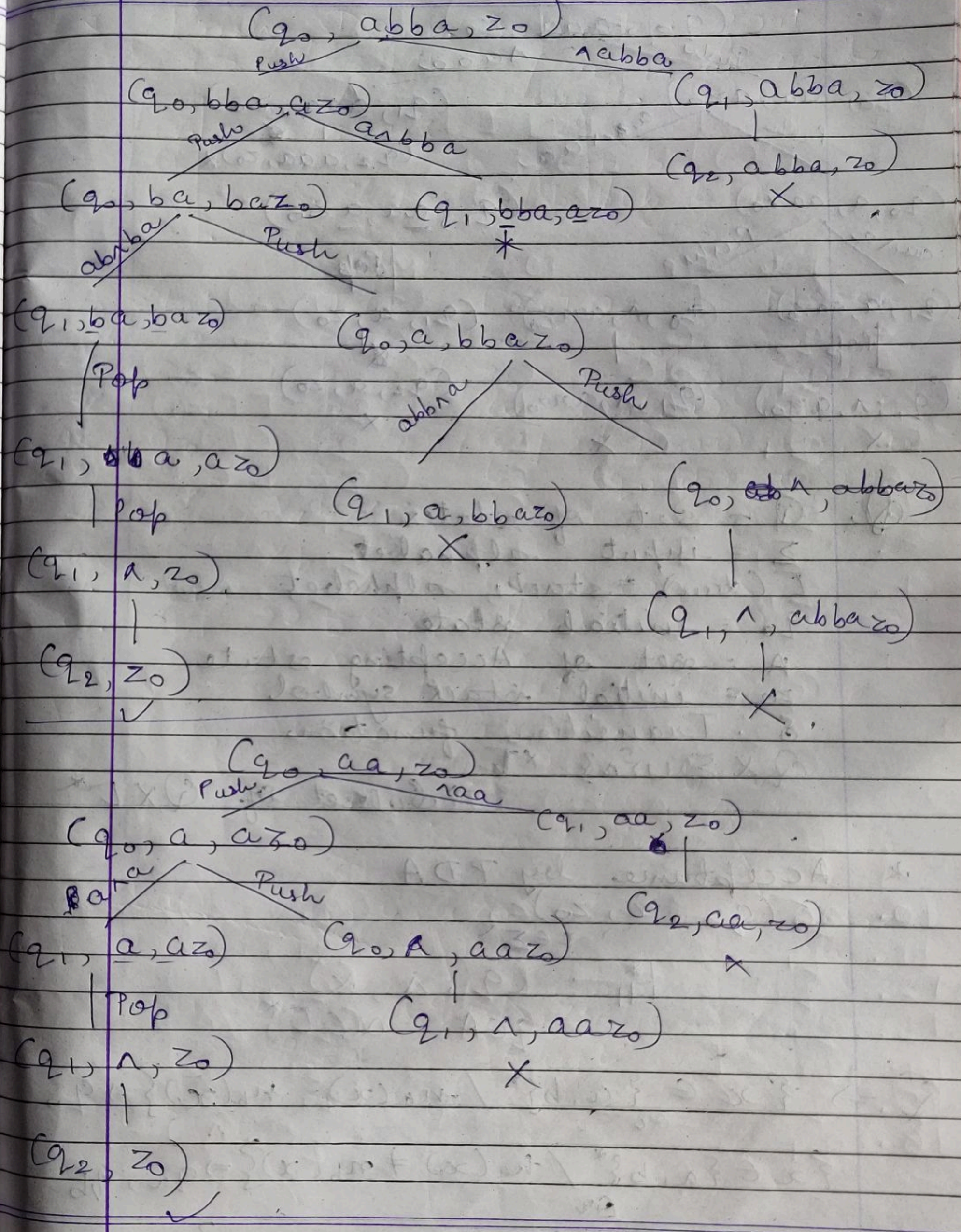
* $L = \{ x \in \{a, b\}^* \,/\, x$ is an even length Palindrome $\}$

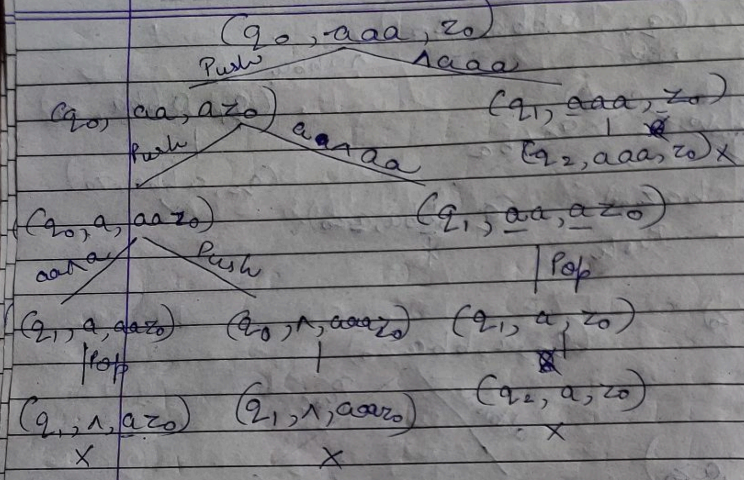
State	I/P	TOS	Move
1) q_0	a	Z_0	$\{q_0, aZ_0\}$
2) q_0	b	Z_0	$\{q_0, bZ_0\}$
3) q_0	a	b	$\{q_0, ab\}$
4) q_0	a	a	(q_0, aa)
5) q_0	b	a	(q_0, ba)
6) q_0	b	b	(q_0, bb)
7) q_0	$\wedge$	Z_0	(q_1, Z_0)
8) q_0	$\wedge$	b	(q_1, b)
9) q_0	$\wedge$	a	(q_1, a)
10) q_1	a	a	$(q_1, \wedge)$
11) q_1	b	b	$(q_1, \wedge)$
12) q_1	$\wedge$	Z_0	(q_2, Z_0)

$(q_0, \wedge, Z_0)$
$(q_1, \wedge, Z_0)$
(q_2, Z_0) = Non-deterministic

Rule 7:
$\delta(q_0, \wedge, Z_0)$
(q_0, a, Z_0)

should not be used but it is being used





Q. Q = set of states
Σ = input alphabet
Γ (tau) = stack alphabet
q_0 = initial state
A = set of Accepting state
z_0 = initial stack symbol
δ : Transition function
$Q \times \Sigma \cup \{\wedge\} \times \Gamma$
$\rightarrow$ Subset of $Q \times \Gamma^*$

* Acceptance by PDA
$\delta(q_0, x, z_0)$
$x \in \Sigma^*$
$\vdash^* (q, \wedge, \alpha)$
$q \in A \quad \alpha \in \Gamma^*$

Q $\Rightarrow \{x \in \{a, b\}^* / n_a(x) = n_b(x)\} \Rightarrow q_0$
or
$\{x \in \{a, b\}^* / n_a(x) \neq n_b(x)\} \Rightarrow q_a, q_b$
Or
$\{x \in \{a, b\}^* / n_a(x) > n_b(x)\} \Rightarrow q_a$

Or
$\{x \in \{a, b\}^* / n_a(x) < n_b(x)\} \Rightarrow q_b$
Ex: aabb, ab, ba, abab, baba, ...

	State	I/P	TOS	Move
1.	q_0	a	$\cancel{z_0}$ z_0	(q_a, z_0)
2.	q_0	b	z_0	(q_b, z_0)
3.	q_a	a	z_0	(q_a, az_0)
4.	q_a	a	a	(q_a, aa)
5.	q_a	b	a	$(q_a, \wedge)$
6.	q_a	b	z_0	(q_0, z_0)
7	q_b	b	z_0	(q_b, bz_0)
8	q_b	b	b	(q_b, bb)
9	q_b	a	b	$(q_b, \wedge)$
10	q_b	a	z_0	(q_0, z_0)

(q_0, aab, z_0)
(q_a, ab, z_0)
(q_a, b, az_0)
$(q_a, \wedge, z_0) \Rightarrow$ Accept

Deterministic PDA:

1). For any $q \in Q$, $a \in \Sigma$, $x \in \Gamma$
$\delta(\underset{\text{state}}{q}, \underset{\text{I/P}}{a}, \underset{\text{TOS}}{x})$ has atmost one rule

2). If $\delta(q, \wedge, x) \neq \emptyset$, then $\delta(q, a, x) = \emptyset$ for every $a \in \Sigma$

Rule (a)

In Even length Palindrome - $(q_0, \wedge, a)$
Rule 3 :- $(q_0, a, a) \Rightarrow (q_0, b, a)$ ×

* $S \to SS \mid [S] \mid (S) \mid \wedge$

$(S) \to (())$ $\quad [()]\,() \to SS$

$[S] \to [[]]$ $\quad [[]]\,[] \to SS$

$[S] \to [()]$ $\quad ()\,[] \to SS$

$\quad ()\,() \to SS$

	State	I/P	TOS	Move
1	q_0	[	z_0	$(q_1, [z_0)$
2	q_0	(	z_0	$(q_1, (z_0)$
3	q_1	(	(	$(q_1, (()$
4.	q_1	[	[	$(q_1, [[)$
5.	q_1	(	[	$(q_1, ([)$
6.	q_1	[	(	$(q_1, [()$
7.	q_1	)	(	$(q_1, \wedge)$
8.	q_1	]	[	$(q_1, \wedge)$
9.	q_1	$\wedge$	z_0	(q_0, z_0)

$(q_0, [], z_0)$

$(q_1,], [z_0)$

$(q_1, \wedge, z_0)$

$(q_0, \wedge; z_0) \to$ Accept

If $\{S\}$

	State	I/P	TOS	Move
10.	q_0	{	z_0	$(q_1, \{z_0)$
11.	q_1	{	{	$(q_1, \{\{)$
12.	q_1	{	[	$(q_1, \{[)$
13.	q_1	{	(	$(q_1, \{()$
14.	q_1	[	{	$(q_1, [\{)$
15.	q_1	(	{	$(q_1, (\{)$
16.	q_1	}	{	$(q_1, \wedge)$

* $S \to [S] \mid (S) \mid \wedge$

9a. q_0, $\wedge$ z_0 $\quad$ pop (z_0) (q_0, z_0)

CFG → NPDA :-

1. $(q_0, \wedge, z_0) \to (q_1, Sz_0)$.

Push

2. → For every NT in G
$\delta(q_1, \wedge, A) \to (q_1, \alpha)$
Here $A \to \alpha$ is a production in G

Pop

3. For every Terminal in G
$\delta(q_1, a, a) \to (q_1, \wedge)$

Acceptance

4. $\delta(q_1, \wedge, z_0) \to (q_2, z_0)$

$S \to a \mid aS \mid bSS \mid SbS \mid SSb$

1.	q_0	$\wedge$	z_0	(q_1, Sz_0)
2.	q_1	$\wedge$	S	(q_1, a) (q_1, aS) (q_1, bSS), (q_1, SbS) (q_1, SSb)
3.	q_1	a	a	$(q_1, \wedge)$
4.	q_1	b	b	$(q_1, \wedge)$
5.	q_1	$\wedge$	z_0	(q_2, z_0)

(q_0, aba, z_0)

(q_1, aba, Sz_0)

(q_1, aba, az_0) $\quad (q_1, aba, SbSz_0)$

$(q_1, aba, abSz_0)$

$(q_1, ba, bSz_0) \to (q_1, a, Sz_0)$

(q_1, a, az_0)

$(q_1, \wedge, z_0)$

(q_2, z_0)