Normal Forms

In context free grammar we know that their is no restriction on right hand side of production rule, however we can convert the given CFG into some mixed notation on right hand side production rule and this is known as Normalization or conversion of CFG into some other standard form.

V. Imp

Remember that while doing this, the meaning of given CFG will not change and will remain the same.

⇒ There are ~~ma~~ two main normal form,

1.> Chomsky Normal Form (CNF)
2.> Greibach Normal Form (GNF)

Chomsky Normal Form (CNF)

This allows following production,

1.> $S \rightarrow \epsilon$ where S is start symbol
2.> $A \rightarrow BC$
3.> $B \rightarrow a$
4.> $C \rightarrow b$

NoTe

- While converting the given CFG to CNF first of all apply simplification of CFG (Removal of useless, unit and ϵ-production)
- After that we can use new non-terminal and can act accordingly
- There can be more than one possible CNF for the given grammar

Ques: For the given ~~CH~~ CFG, Convert this into GNF

CFG: 1.) $S \rightarrow \epsilon$

2.) $S \rightarrow ABC$

3.) $C \rightarrow a$

4.) $B \rightarrow bE$

5.) $E \rightarrow a$

6.) $A \rightarrow a$

Sol^n, Since the given grammar does not require any simplification, hence we will know apply CNF conversion,

1.) $S \rightarrow \epsilon$

2.) $S \rightarrow FC$

3) $F \rightarrow AB$

4) $C \rightarrow a$

5) $B \rightarrow GE$

6) $G \rightarrow b$

7) $E \rightarrow a$

8) $A \rightarrow a$

Equivalent Grammar

If two grammar G & G' represent the same language then both are equivalent grammar that is,

$$L(G) = L(G')$$

⇒ If grammar represent by CFG is denoted as G & the converted grammar into CNF or GNF is G' then again,

$$L(G) = L(G')$$

Ques. Convert the given CFG into CNF.

CFG ⇒ P : 1) $S \rightarrow \epsilon \mid aG$

2) $S \rightarrow ABCD$

3) $A \rightarrow a$

4) $B \rightarrow b$

5) $C \rightarrow aE$

6) $E \rightarrow b$

7) $D \rightarrow F$

8) $F \rightarrow a$

Sol

Applying simplification.

Removing useless production, so we get

1) $S \rightarrow \epsilon$

2) $S \rightarrow ABCD$

3) $A \rightarrow a$

4) $B \rightarrow b$

5) $C \rightarrow aE$

6) $E \rightarrow b$

7) $D \rightarrow F$

8) $F \rightarrow a$

Now removing unit production, so we get

1) $S \rightarrow \epsilon$

2) $S \rightarrow ABCD$

3) $A \rightarrow a$

4) $B \rightarrow b$

5) $C \rightarrow aE$

6) $E \rightarrow b$

7) $D \rightarrow a$

Now the given grammar is simplified, hence we will apply GNF Conversion,

$G' \rightarrow$ 1) $S \rightarrow \epsilon$

2) $S \rightarrow AH$

3) $H \rightarrow BI$

4) $I \rightarrow CD$

5) $A \rightarrow a$

6) $B \rightarrow b$

7) $C \rightarrow JE$

8) $J \rightarrow a$

9) $E \rightarrow b$

10) $D \rightarrow a$

Greibach Normal Form (GNF)

This allow following production,

1) $S \rightarrow \epsilon$

2) $A \rightarrow a$

3) $B \rightarrow aN^*$

Ques: Apply the GNF Conversion for the given CFG,

P: $S \rightarrow Aba$

$S \rightarrow \epsilon$

$S \rightarrow C$

$C \rightarrow a$

$C \rightarrow abD$

Solⁿ, Applying simplification, removing useless production,

1) $S \rightarrow \epsilon$ 3) $C \rightarrow a$

2) $S \rightarrow C$

Camlin Page
Date / /

Now removing unit production,

1) $S \rightarrow \epsilon$
2) $S \rightarrow a$

Now after simplification, applying GNF conversion

1) $S \rightarrow \epsilon$
2) $S \rightarrow a$

Ques Convert given CFG to GNF

P: 1) $S \rightarrow Aaab$
2) $S \rightarrow \epsilon$
3) $S \rightarrow C$
4) $C \rightarrow a$
5) $A \rightarrow a$

Sol^n, Removing unit production,

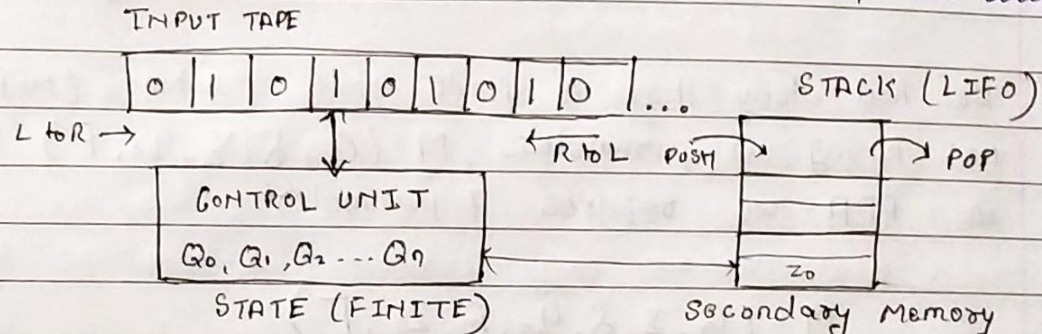
1) $S \rightarrow Aaab$
2) $S \rightarrow \epsilon$
3) $S \rightarrow a$
4) $A \rightarrow a$

Now applying GNF conversion,

1) $S \rightarrow aPQA$
2) $P \rightarrow a$
3) $Q \rightarrow b$
4) $S \rightarrow \epsilon$
5) $S \rightarrow a$
6) $A \rightarrow a$

Push Down Automata (PDA)

- Push Down Automata, PDA is a finite automata having secondary storage or extra storage.
- The secondary storage or extra storage is in the form of STACK (LIFO Property)
- The PDA is for Context Free Language processing.
- A general PDA representation can be shown as below,



⇒ In regular machine (NFA & DFA) only input tape and control unit state was used for defining the transition (δ). However here in PDA transition will involve 3 ~~three~~ parameters. The 3rd extra parameter will be the symbol on to the top of the stack.

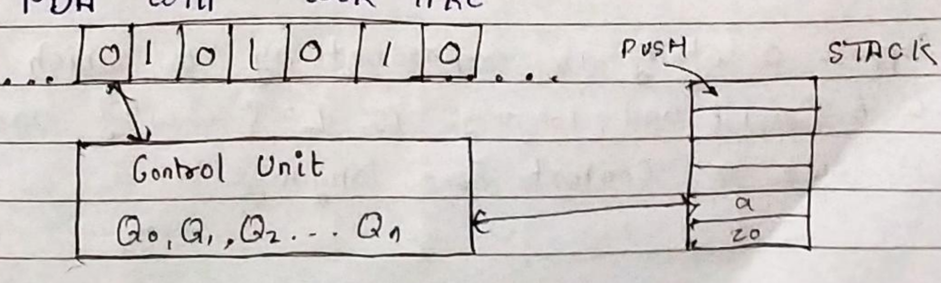
⇒ Hence we can understand the transition state (δ) for above PDA representation as mention below,

$$\delta(Q_0, 0, Z_0) \Rightarrow (Q_1, aZ_0)$$

$$\delta(Q_0, 0, Z_0) \Rightarrow (Q_1, \epsilon)$$

⇒ We very much know that 2 operation PUSH & POP are useful when defining the ~~state~~ Stack.

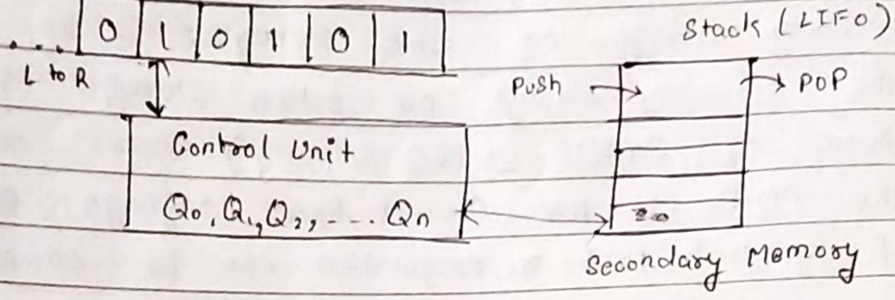
⇒ So, for PUSH operation $\delta(Q_0, 0, Z_0) \Rightarrow (Q_1, aZ_0)$ and PDA will look like



For PoP operation, the transition (δ) can be written as

$$\delta(Q_0, 0, Z_0) = (Q_1, G)$$

& Stack will look like



Imp// We had study that a finite state machine (NFA & DFA) was having 5 parameters, $M = (Q, \delta, \Sigma, q_0, F)$ while in PDA we requires 7 parameter,

$$M = (Q, \Sigma, \delta, q_0, F, Z_0, \underset{\text{TAU}}{\Gamma})$$

where Z_0 is default or initial input symbol of stack and TAU (Γ) is the symbol allowed on stack.

=> We know that certain languages such as,

(1) $L = a^n b^n ; n \geq 0$ can not be expressed or process through finite state machine (DFA & NFA) and can be expressed through push Down Automata and the language are Context Free Language.

2.> $L = \{a^n b^{2n} : n \geq 0\}$
3.> $L = \{0^n 1^n : n \geq 0\}$

} are Context Free language

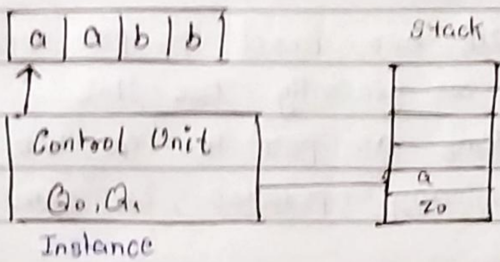
Imp// 4.> Suppose a string is represent by 'w' which belong to $w \in \{0, 1\}$ and language is $L = \{ww^R\}$ and $\Sigma = \{0, 1\}$ is also a Context Free language

5) Palindrom numbers of any length are also a example of context free language.

#) Note: we know that in stack we can have 3 possible operation — PUSH, POP & SKIP or (no operation), let understand one by one,

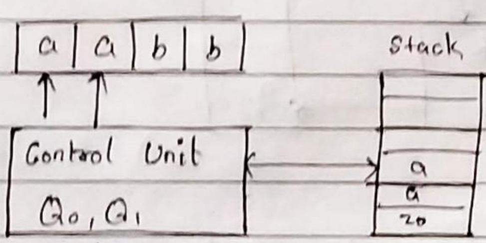
#1 PUSH

(i)



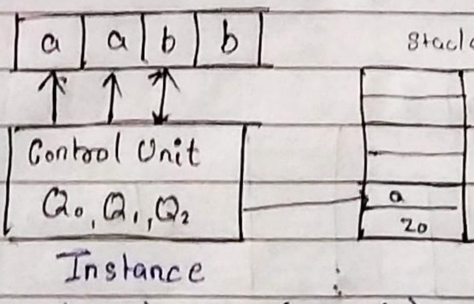
$\delta(Q_0, a, Z_0) = (Q_1, aZ_0)$

(ii) $\delta(Q_0, a, a) = (Q_1, aa)$



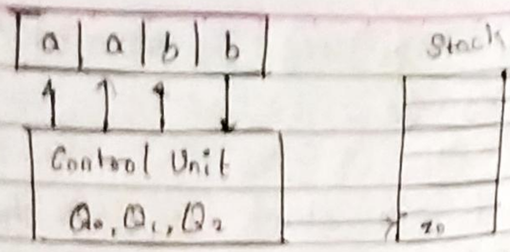
#2 POP

(iii)



$\delta(Q_0, b, a) = (Q_2, \epsilon)$

(iv)



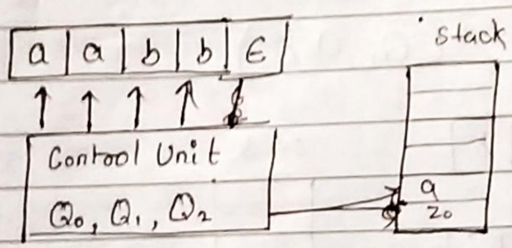
instance

$$\delta(Q_2, b, a) = (Q_2, \epsilon)$$

SKIP Operation

- This is based on our need, as per our convinience for state change we generally use this
- And also some time as per the question when we don't want perform any operation, we can use this.

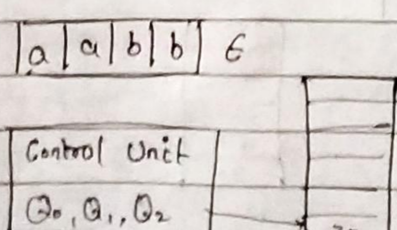
⇒ For the following representation, we can understand skip operation as mentioned below,



Instance

$$\delta(Q_2, \epsilon, Q_0) = (Q_2, Q_0)$$

⇒ Suppose we have a following repⁿ of the PDA then we will have skip operation written as bellow



$$\delta(Q_2, \epsilon, Z_0) = (Q_2, Z_0\}$$

Camlin Page
Date / /

⇒ In general the transition delta (δ) for PDA machine will be expressed as general notation is shown below,

Imp

$$\left[Q \times \Sigma \cup \{\epsilon\} \times \Gamma \Rightarrow Q \times \Gamma^{*} \right]$$

Acceptance of String by PDA

Acceptance by Final State (say Q_f or q_f)
Acceptance by Empty Stack

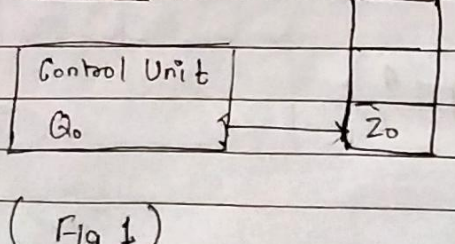
⇒ The first approach we had already studied in the regular grammar, however the other approach is acceptance by empty stack.

NoTe

1. Remember almost all the step are same except the last state by accepting through final state or accepting through empty stack.

2. In both the approaches, the input tape must be empty atleast that is no symbol is remaining.

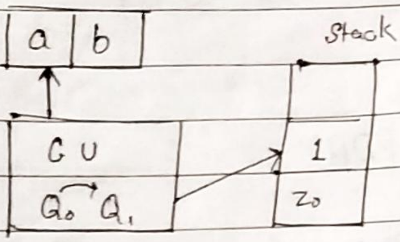
Ques Describe the acceptance by final state for the string $\omega = ab$ through PDA.



(Fig 1)

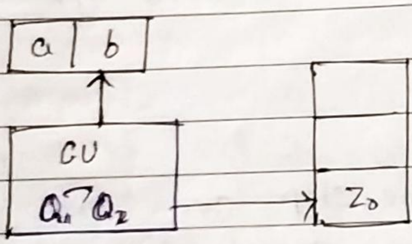
Let $\Gamma = \{Z_0, 1, 0\}$

(i) $\delta(Q_0, a, Z_0) = \{Q_1, 1Z_0)$



(Fig 2)

(ii) $\delta(Q_1, b, 1) = (Q_1, \epsilon)$



(Fig 3)

(iii) $\delta(Q_2, \epsilon, Z_0) = (Q_f, Z_0)$

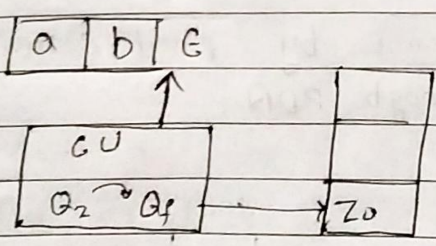
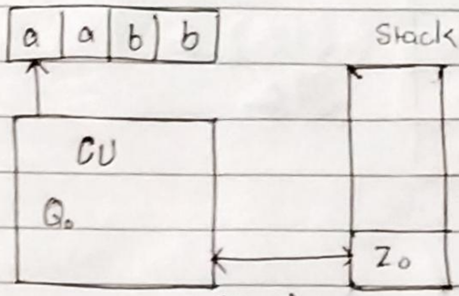


Fig (4)

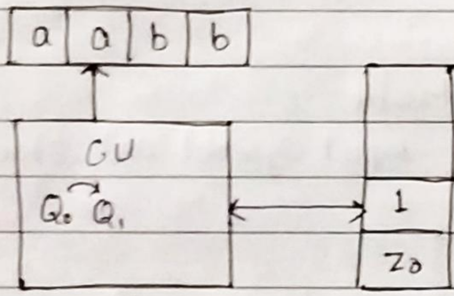
Ques Show the acceptance of string, $w = aabb$ using final state in PDA

Sol^n^,



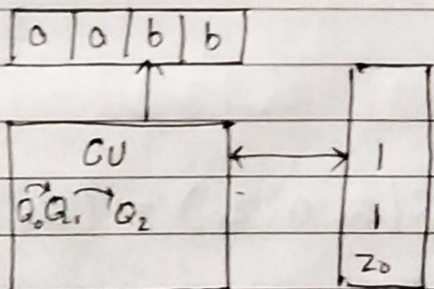
(Fig. 1)

$$\delta(Q_0, a, Z_0) = (Q_1, 1Z_0)$$



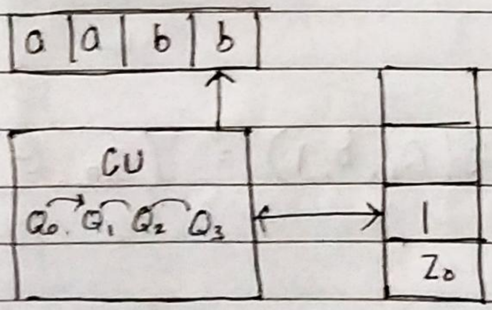
(Fig. 2)

$$\delta(Q_1, a, 1) = (Q_2, 11)$$



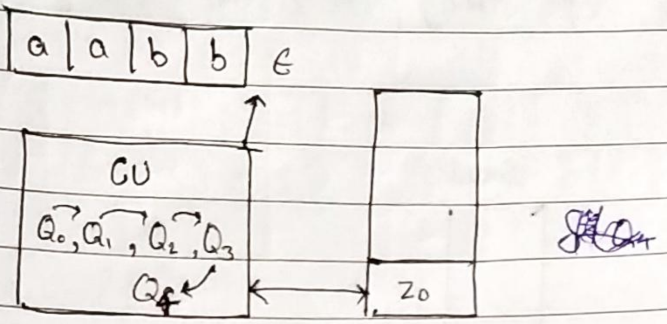
(Fig. 3)

$$\delta(Q_2, b, 1) = (Q_3, \epsilon)$$



(Fig. 4)

$$\delta(Q_3, b, 1) = (Q_4, \epsilon)$$



(Fig. 5)

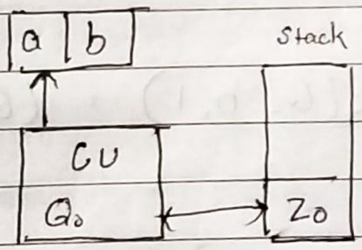
Here we have reach to the final state and input tape is also empty.

Acceptance by Empty Stack

In this approach both input symbol and stack must become empty.

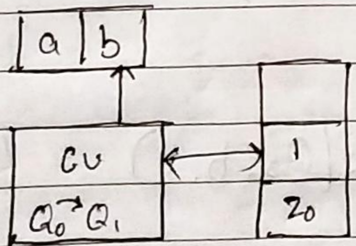
Ques: Show the acceptance by empty stack for string, $w = ab$ using PDA

Solⁿ



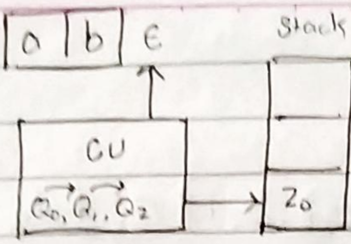
$\delta(Q_0, a, Z_0) = (Q_1, 1Z_0)$

(Fig. 1)



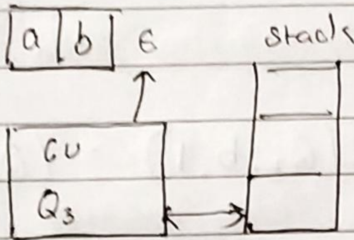
$\delta(Q_1, b, 1) = (Q_2, \epsilon)$

(Fig. 2)



$\delta(Q_2, \epsilon, Z_0) = (Q_3, \epsilon)$

(Fig.3)

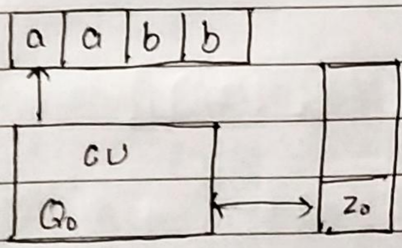


(Fig.4)

Here the input symbol and stack both are empty.

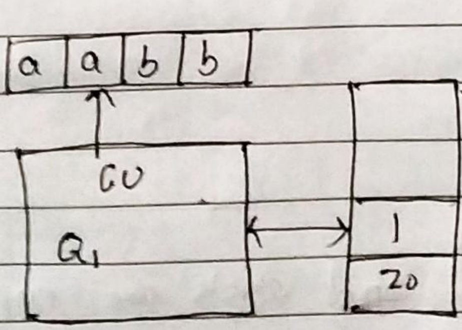
Ques Accept using empty stack for the string, $w = aabb$ using PDA

Sol^n:



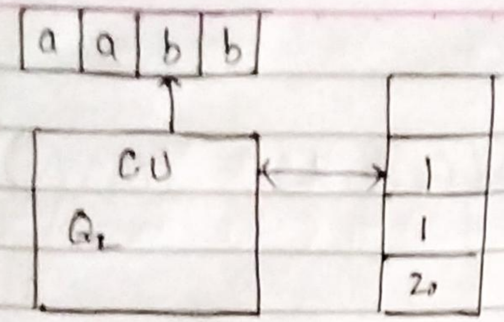
$\delta(Q_0, a, Z_0) = (Q_1, Z_0 1)$

(Fig.1)



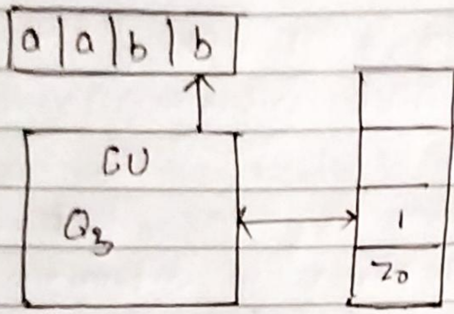
$\delta(Q_1, a, 1) = (Q_2, 1, 1)$

(Fig.2)



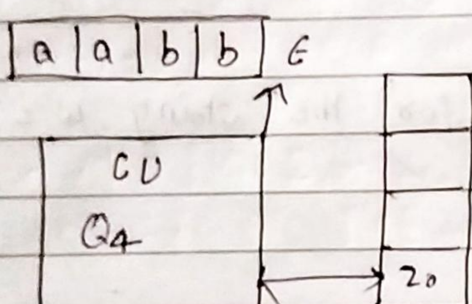
$\delta(Q_2, b, 1) = (Q_3, \epsilon)$

(Fig. 3)



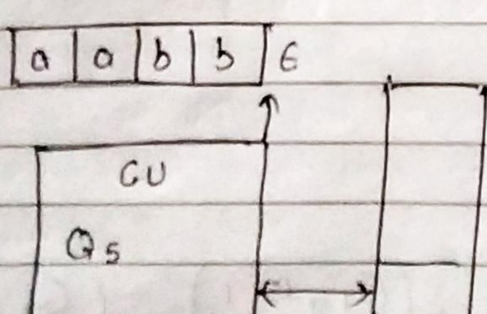
$\delta(Q_3, b, 1) = (Q_4, \epsilon)$

(Fig. 4)



$\delta(Q_4, \epsilon, Z_0) = (Q_5, \epsilon)$

(Fig. 5)



Here both the input symbol and stack are empty.

Camlin Page
Date / /

Ques: Show the acceptance using PDA for the language,
$L = \{a^n b^n : n \geq 1\}$
Write down the state representation and also mention Instantaneous Description (ID) for any assumed string following the given language.

Instantaneous Description (ID)

- We know that we generally process these Context Free language with needs large amount of secondary storage or memory
- In order to process them we require instantaneous description for any small size finite string which is following that limit.
- To show instantaneous description, we use the symbol "$\vdash$" between two consecutive transition.
- ID again describe for acceptance by Final state and acceptance by empty stack.

Ques: For string, w = aabb, following the language
$L = \{a^n b^n : n \geq 1\}$
Show the instantaneous description for acceptance using Final State & Empty Stack.

Soln: For the final state Machine,

$(Q_0, aabb, Z_0) \vdash (Q_1, abb, 1Z_0) \vdash (Q_2, bb, 11Z_0) \vdash$

$(Q_3, b, 1Z_0) \vdash (Q_f, \epsilon, Z_0)$

Now doing the same, for empty stack,

$$(Q_0, aabb, Z_0) \vdash (Q_1, abb, 1Z_0) \vdash (Q_2, bb, 11Z_0)$$
$$\vdash (Q_3, b, 1Z_0) \vdash (Q_4, \epsilon, Z_0) \vdash (Q_5, \epsilon, \epsilon)$$

Ques What do you mean by Deterministic PDA and Non-deterministic PDA?

Soln We had already studied the basic meaning of deterministic & Non-deterministic.

- In non-deterministic we can have more than one or no transition for any input symbol.

Example :- Two PDA machine M & M' are describe below,

For $M = \{Q, \Sigma, \delta, q_0, F, Z_0, \Gamma\}$

$Z_0 = \{0\}$, $\Gamma = \{Z_0, 1\}$

$$\delta(Q_0, a, Z_0) = \{(q_1, \epsilon), (q_2, 1Z_0)\}$$

$$\delta(Q_0, b, 1) = \{Q_2 \epsilon\}$$

=> Here in M, few transition are left out and few are used more than one. Hence M is non-deterministic PDA.

Example (i) $L = \{WW^R : \Sigma = \{0, 1\}\}$

(ii) $L = \{WCW^R : \Sigma = \{0, 1\}\}$

Sol^n, In first one we don't know when w is going to end and when w^R is going to start.
Assuming $w = 100$, So $w^R = 001$
Hence $ww^R = 100001$

⇒ So after process 1st one^n & zero for the next symbol we are not aware whether this is part of w or w^R.

⇒ But^n, in example 2 any thing befor C will be the part of w and after C will be the part of w^R.

Ques: Complete PDA for $L = \{a^n b^n : n \geq 1\}$

Sol^n, Let PDA is represented by M

$M = (Q, \Sigma, \delta, q_0, F, Z_0, \Gamma)$

Let $Z_0 = 0$, $\Gamma = \{Z_0, 0, a\}$

Transition

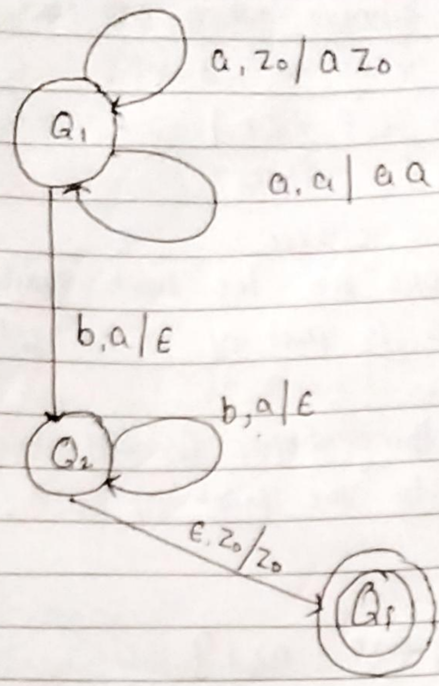
PUSH
$\delta(Q_1, a, Z_0) = (Q_1, aZ_0)$ — Assume Q_1 start as start state
$\delta(Q_1, a, a) = (Q_1, aa)$ — For PUSH, assume Q_1 state

POP
$\delta(Q_1, b, a) = (Q_2, \epsilon)$
$\delta(Q_2, b, a) = (Q_2, \epsilon)$ — For POP, assume Q_2 state

Final state accept.
$\delta(Q_2, \epsilon, Z_0) = (Q_F, Z_0)$

Empty stack acceptance
$\delta(Q_2, \epsilon, Z_0) = (Q_3, \epsilon)$ — Q_f is final state

Q_3 is state on empty stack transition



* PDA representation (acceptance using final State)

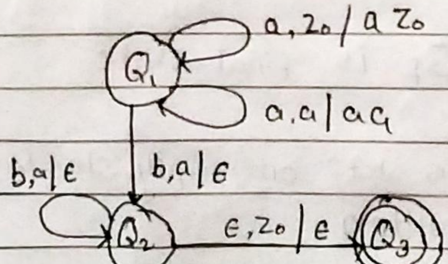
Let $w = aabb \in L$

ID

$(Q_1, aabb, Z_0) \vdash (Q_1, abb, aZ_0) \vdash (Q_1, bb, aaZ_0)$

$\vdash (Q_2, b, aZ_0) \vdash (Q_2, \epsilon, Z_0) \vdash (Q_f, \epsilon, Z_0)$

⟹ The same above question can also be solve using acceptance by empty stack.



* PDA representation (acceptance using empty stack)

ID for $w = aabb \in L$

$(Q_1, aabb, Z_0) \vdash (Q_1, abb, aZ_0) \vdash (Q_1, bb, aaZ_0)$

$\vdash (Q_2, b, aZ_0) \vdash (Q_2, \epsilon, Z_0) \vdash (Q_3\ \epsilon, \epsilon)$

Ques: Design a PDA for $L = \{a^n b^n : n \geq 0\}$

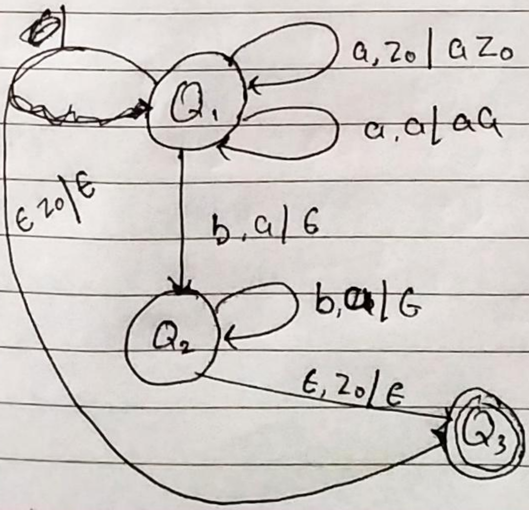
Sol^n/ For this question since empty string (ϵ) can also come hence we should show either of the transaction along with all the previous transition & their two representation are mentioned below

(i) $\delta(Q_1, \epsilon, Z_0) = (Q_3, \epsilon)$

(ii) $\delta(Q_1, \epsilon, Z_0) = (Q_f, Z_0)$

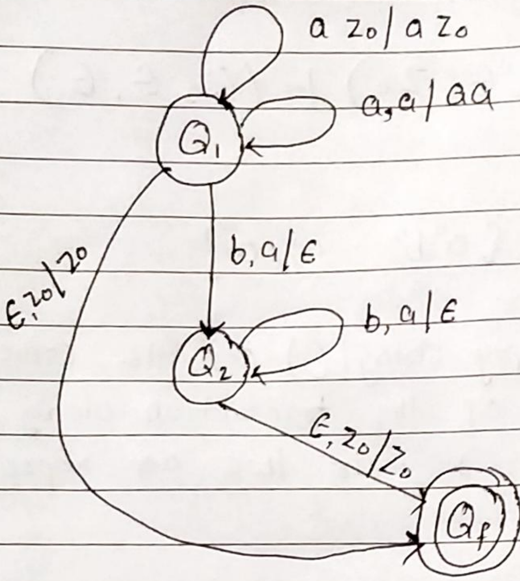
⇒ Here first one is for empty stack acceptance and 2nd one is for final state acceptance.

⇒ Also the PDA state representation will get change and correspondence PDA is show below,



* PDA representation (by empty stack)

Now for the acceptance by final state, the PDA will be like as mention below



* PDA Representation
(By Final State Acceptance)