Lucky | Page No.:
Date : / /

Unit 4 : Parsing and pushdown Automata :

* Pushdown Automata :

A pushdown automata is a way to implement a context-free grammar in a similar way we design DFA for a regular grammar.

A Pushdown automata is a 7 tuple.

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

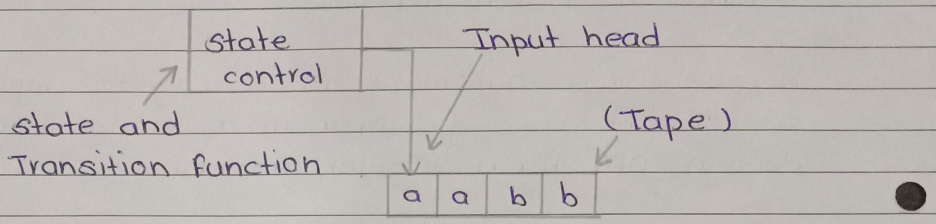
Q = finite set of states.
Σ = set of alphabets
q_0 = Initial state
A = Accepting state
δ = Transition function
Γ = states alphabets
Z_0 = Initial state symbol

Pushdown automata is simply an NFA argumented with an "external stack memory".

Pushdown automata is finite automata with extra memory called stack. Which helps to recognize CFL.

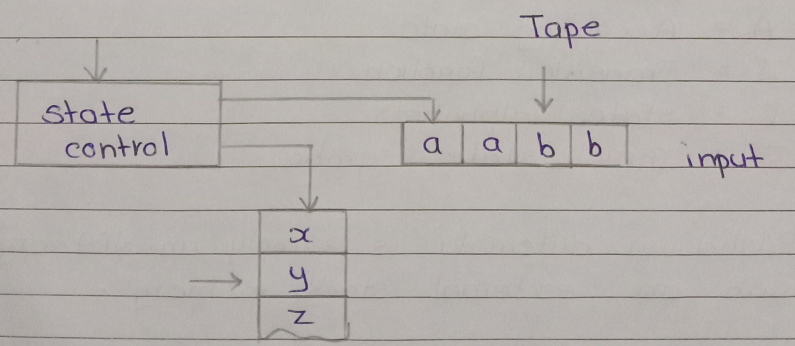
A Pushdown automata is more powerful than FA.

The schematic representation of FA is :



* like NFA-Λ but having stack.

* Stack is empty the automata is stack.



Lucky | Page No.:
Date : / /

stack :

The stack is a strcture in which are can push and remove the items from one end only. It han an infinite size. In PDA, the stack is used to store the items temporarily.

stack does two operations :

① Push : A new symbol is added at the top

② Pop : The top symbol is read and removed.

* Transaction function :

$$\delta : Q \times (\Sigma \cup \{\wedge\}) \times \Gamma \rightarrow Q \times \Gamma^*$$

Lucky | Page No.:
Date : / /

* Acceptance Types

① Language accepted by empty stack

② Language accepted by final stack

③ Language accepted by empty stack and final state.

Greibach Normal form :

A CFG is in GNF if all the production rules satisfy one of the following conditions :

① A start symbol generating ε . for ex :
$S \rightarrow \varepsilon$

② A non-terminal generating a terminal
for ex : $A \rightarrow a$

③ A non-terminal generating a terminal which is followed by any number of non-terminals for example , $S \rightarrow aASB$

Lucky | Page No.:
Date : / /

$S \rightarrow aAB \mid b \mid aB$

$A \rightarrow aA \mid b$

$B \rightarrow cB$

There is only one terminal symbol in front called CNF.

Lucky | Page No.:
Date : / /

* Deterministic PDA :

Let $M = (Q, \Sigma, q_0, A, \delta, z_0, \Gamma)$ be a pushdown automata. M is called as deterministic if M has not more than one

① for any $q \in Q$, $a \in \Sigma \cup \{\wedge\}$ and $x \in \Gamma$ the set $\delta(q, a, x)$ has at most one element.

② for any $q \in Q$ and $x \in \Gamma$ if $\delta(q, \wedge, x) \neq \emptyset$ then $\delta(q, a, x) = \emptyset$ for even $a \in \Sigma$.

$\delta : Q \times (\Sigma \cup \Sigma) \times \Gamma$ to $Q \times \Gamma^*$

③ Read unit :

Read unit of PDA reads words from the cell of i/p tape beginning with first letter in the leftmost cell and then mores to right.

It can't go back.

④ control unit :

Internal state performance

It performs transaction on i/p provided by read unit.

It having 3 states like,
START, ACCEPT, REJECT

⑤ stack :

stack is infinite.

- Always start with empty stack.
- stack can be hold letters of STACK alphabet which can be same as i/p alphabet.
- Initial stack symbol placed on top of stack.

Premitive for writing stack

① Push :

Add i/p symbol on top of stack.

② Pop :

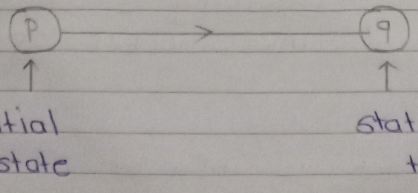
Remove top i/p symbol from top of stack. If empty no change in state.

③ nop :

No any work.

* Diagram and table of PDA

① Transition Diagram :



Lucky | Page No.:
Date : / /

a) (input symbol , top input symbol of stack)

(operation on stack)

b) (P , input symbol , top symbol of stack)
, operation , q

c) (input symbol , Pop old stack symbol |
push new stack symbol)

Pushdown automata is same as DFA with only difference of that it contain auxiliary stack which provide

unlimited memory

The language is recognized by PDA iff it is recognized by context free

Accept both regular and non regular language in the form of stack.

Generally used for parsing and compiler construction

Stack update by pushing or poping

Main components : Input string Read unit
control unit stack

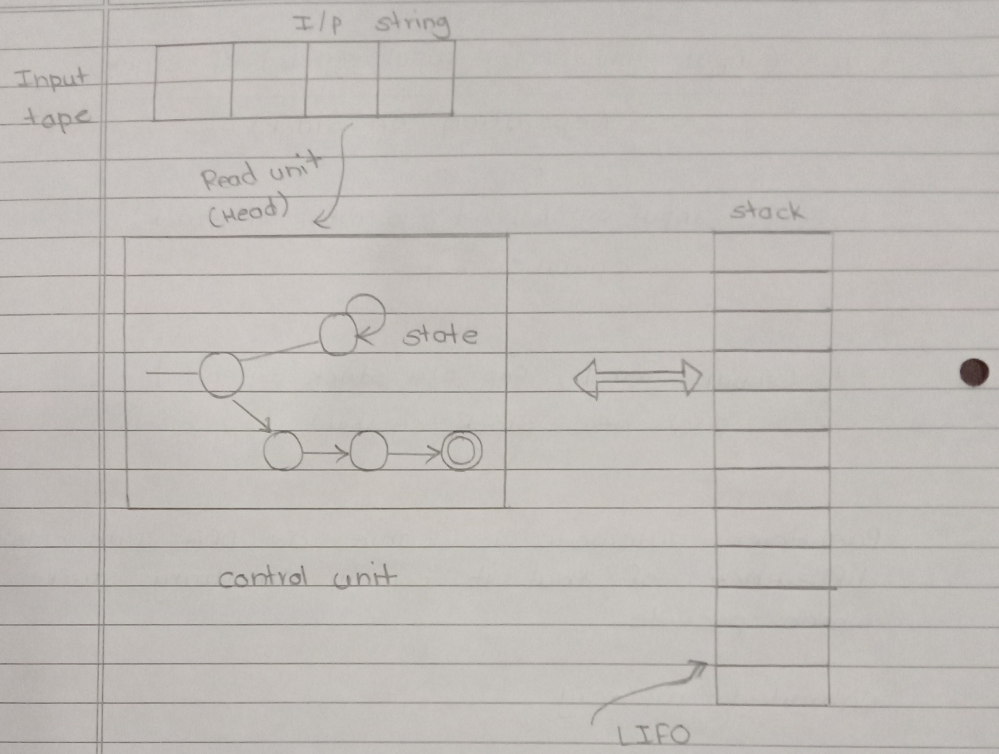


fig : PDA component

① Input Tape :

Infinitely long on which i/p is written

contain sequence of cells

Each cell begin at left and extreme to right end

Hold one input letter or blank symbol ∈

String is written on the tape prior to the beginning of operation of PDA.

Having read only head which always more right and having one cell

Initially

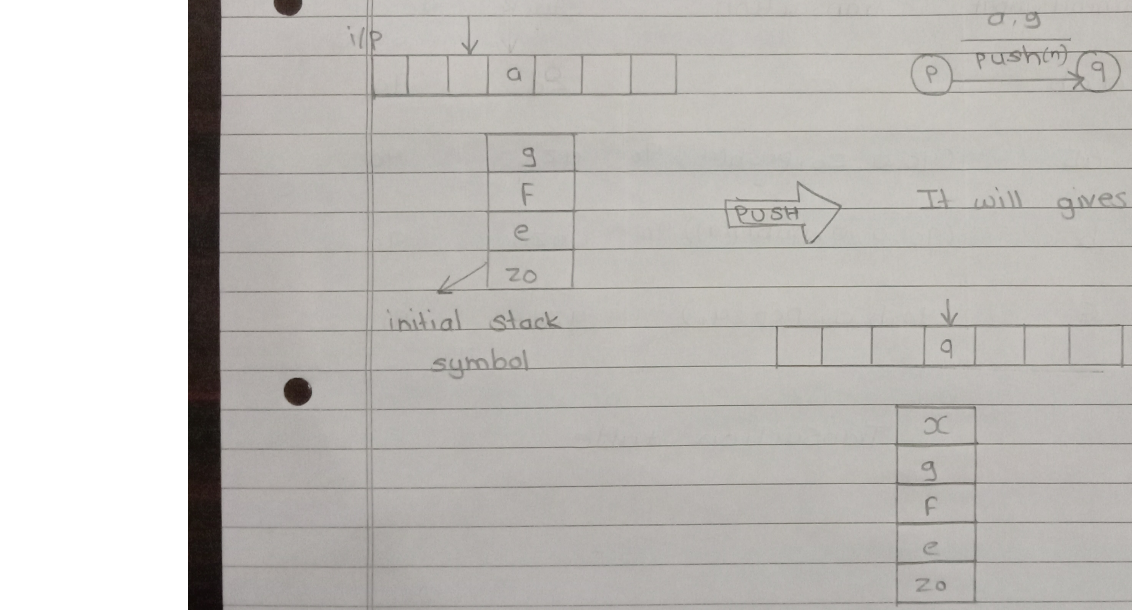


Fig: Transaction Diagram

② Transaction table

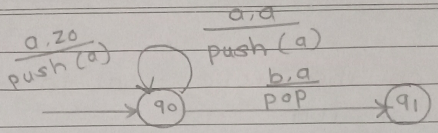


fig : transaction Diagram of PDA

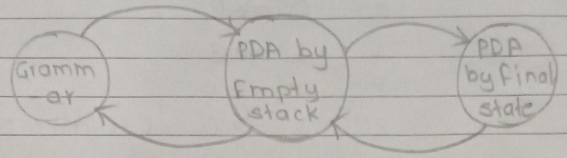
Now transaction table will be as follow,

unread input	Transaction	stack	New state
aab	-	z_0	q_0
ab	$(q_0, a, z_0, push(a)), q_0$	az_0	q_0
b	$(q_0, a, a, push(a)), q_0$	aaz_0	q_0
ϵ	(q_0, b, a, pop, q_1)	az_0	q_1

Transaction table

Lucky | Page No.:
Date : / /

* Equivalence of CFG and PDA



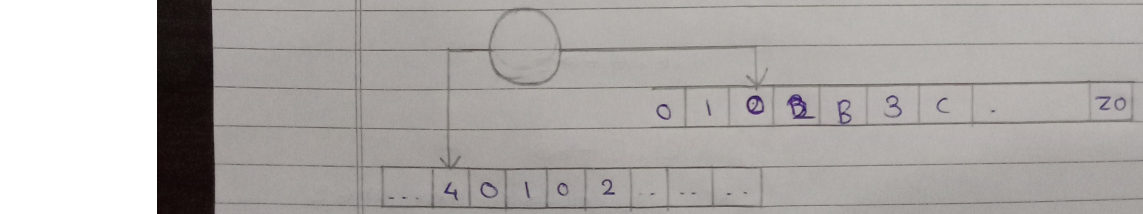
- It shows
  ① CFL is language defined by CFG

  ② The language that are accepted by final state by some PDA

  ③ The language that are accepted by empty stack by some PDA

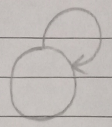
- If we have the rule of following form

  $A \rightarrow 0102B3C$



Lucky | Page No.:
Date : / /

Match terminal symbol to the symbol top



$0 , 0 \to \epsilon$
$1 , 1 \to \epsilon$
$z , z \to \epsilon$
$x , x \to \epsilon$ for all $x \in \Sigma$

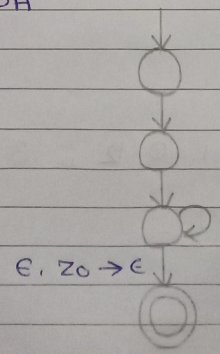
Now top most symbol of stack is terminal symbol.

If i/p and stack are match then pop the symbol from stack and go the next-level

The terminal symbol must be poped from the stack no any push on it.

i.e, $0 , 0 \to \epsilon$
$1 , 1 \to \epsilon$
$z , z \to \epsilon$
$x , x \to \epsilon$

The final PDA



Lucky | Page No.:
Date : / /

While converting CFG to PDA the grammar should be in GNF [Greibach Normal form]

If the production is of the form GNF then we can convert into PDA

$S \rightarrow aAA$
$A \rightarrow as|bs|a$

The grammar will be

$S \rightarrow aAA$
$A \rightarrow as$
$A \rightarrow bS$
$A \rightarrow a$

} GNF

Now equivalent PDA is as follows

$PDA = \{Q, \Sigma, q_0, A, \delta, \Gamma, z_0\}$

Assume $Q = \{q_0, q_1, q_2\}$

$\Sigma = \{a, b\}$

q_0 = Initial state

$A = q_2$ i.e. accepting state

δ = Transaction state

$\Gamma = \{S, A, z_0\}$

z_0 = Initial stack

Step 1 - Push start symbol S onto stack

$\delta(q_0, \epsilon, z_0) = (q_1, Sz_0)$

$S \rightarrow aAA$

$\delta(q_1, a, S) = (q_1, AA)$

$A \rightarrow bS$

$\delta(q_1, b, A) = (q_1, S)$

$A \rightarrow aS$

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

$A \rightarrow a$

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

change $(q_1, \epsilon, z_0) = (q_2, z_0)$
q_1 to final state q_2

$$\delta(q_1, \epsilon, z_0) = (q_2, z_0)$$

Lucky | Page No.:
Date : / /

Ex : $S \rightarrow aA$
$A \rightarrow aABD \mid bB \mid a$
$B \rightarrow b$
$D \rightarrow d$

Initially the starting symbol on stack

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

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

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

$\delta(q_1, b, A) = (q_1, B)$

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

$\delta(q_1, b, B) = (q_1, \epsilon)$

$\delta(q_1, d, D) = (q_1, \epsilon)$

change q_1 to final state q_2

$$\boxed{\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)}$$

Lucky | Page No.:
Date : / /

* Equivalence of CFG and PDA

Two proofs :

① Given CFG to PDA

② Given PDA to show how construct CFG

Ex : ① $G = \{V, \Sigma, S, P\}$

$S \rightarrow BS \mid A$
$A \rightarrow 0A \mid \epsilon$
$B \rightarrow BB \mid 2$

Take leftmost derivation i.e. construct leftmost non terminal symbol

$\rightarrow S$
$\rightarrow BS$
$\rightarrow BBS$
$\rightarrow 2BS$
$\rightarrow 22S$
$\rightarrow 22A$
$\rightarrow 221 \epsilon$
$\rightarrow 221$

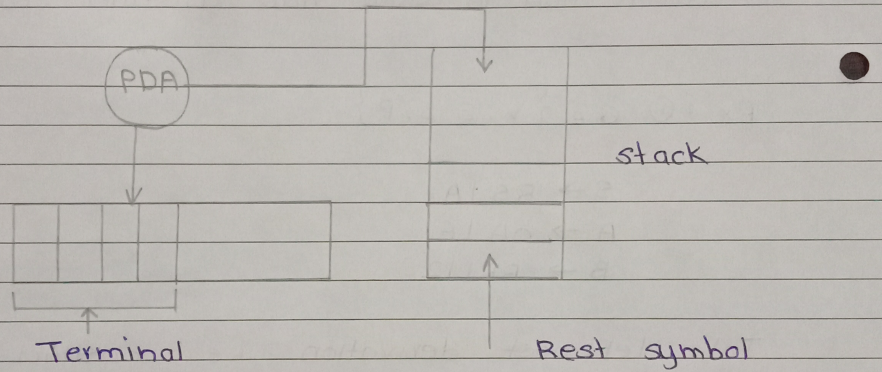
The above all are derived from leftmost derivation hence called as left sentential form

General form

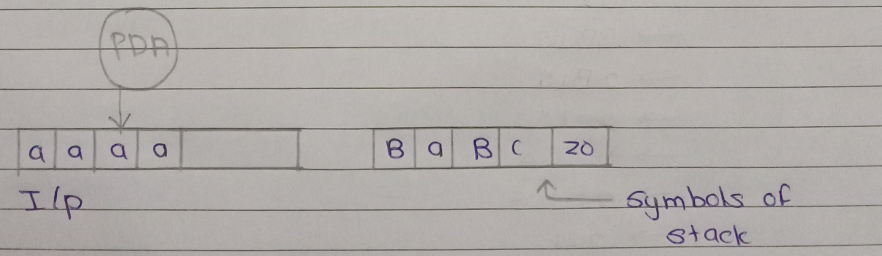
$\omega \Rightarrow$ aaaa BaBC

set of terminal symbol

Rest symbol



Now if we consider the above i/p string then



Now consider

Now take the Rule

$B \rightarrow ASA \mid XBA$ then

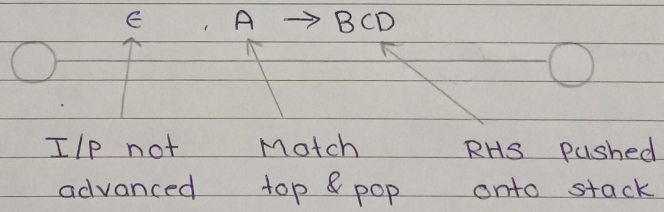
Lucky | Page No.:
Date : / /

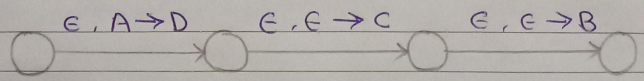
a	a	a	a	A	S	A	X	B	A	a	B	C

Terminal symbol

- Match stack top to a rule
- Pop stack
- Push RHS of rule into stack

If we have rule,
A → BCD add this to PDA





BCD will be pushed in order

B		D
C	not	C
D	like	B
correct		wrong

In above diagram we first pop A and and push D, nothing to pop hence only push C & B

Lucky | Page No.:
Date : / /

* Construct PDA for the CFL

① $S \rightarrow aA$
$A \rightarrow aABD \mid bD$
$B \rightarrow b$
$D \rightarrow d$

Now convert the given CFL into GNF so required GNF.

$S \rightarrow aA$
$A \rightarrow aABD$
$A \rightarrow bD$
$A \rightarrow a$
$B \rightarrow b$
$D \rightarrow d$

Now, we have to define this into PDA

$Q = \{ q_0, q_1, q_2 \}$

$\Sigma = \{ a, b, d \}$

q_0 = initial state

$q_2 \in F$ final state

δ = Transition function

z_0 = initial stack symbol

Lucky | Page No.:
Date : / /

$\Gamma = \{S, A, B, D, Z_0\}$

Assume
$Q = (q_0, q_1, q_2)$

$\Sigma = \{a, b\}$

q_0 = Initial state

$q_2 \in F$ q_2 = final state

δ = Transition function

Z_0 = Initial stack symbol

$\Gamma = \{S, A, Z_0\}$

Now we have to take the δ by selecting starting symbol S

Initially $\delta(q_0, \epsilon, Z_0) = (q_1, SZ_0)$

$\therefore$ by step 1

Grammar	PDA
$S \rightarrow aAA$	$\delta(q_1, a, S) = (q_1, AA)$
$A \rightarrow aS$	$\delta(q_1, a, A) = (q_1, S)$
$A \rightarrow bS$	$\delta(q_1, b, A) = (q_1, S)$
$A \rightarrow a$	$\delta(q_1, a, A) = (q_1, \epsilon)$

State q_1 will be change to final state q_2

$$\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)$$

Step 3 :

In state q_1, on encountering the end of input, if Z_0 is present as the top symbol of ~~status~~ stack then change state to q_2

Lucky | Page No.:
Date : / /

② construct PDA from given CFL

$S \rightarrow aAA$
$A \rightarrow as \mid bS \mid a$

step 1 : convert CFL into GNF

$S \rightarrow aAA$
$A \rightarrow aS$
$A \rightarrow bS$
$A \rightarrow a$

Here it is required GNF

step 2 : convert GNF to PDA

consider $G = (V, \Sigma, S, P)$ is a CFL and

$PDA = (Q, \Sigma, q_0, A, \delta, \Gamma, Z_0)$

Lucky | Page No.:
Date : / /

* Equivalance of PDA and CFL

Given PDA should be NPDA

$$L(G) = L(P)$$

where $L(G)$ = CFL
$L(P)$ = NPDA

* Conversion of CFL to PDA

$G = (V, \Sigma, S, P)$ is CFL

$P = (Q, \Sigma, q_0, A, \delta, \Gamma, Z_0)$ is PDA

steps ① :

Without consuming any input change the state q_0 to q_1 and place the start symbol of G onto stack.

$$S \rightarrow aB$$

Transition

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

initial state — Null transition — initial stack symbol — new state

Lucky | Page No.
Date : / /

② If there is a production in the form $A \rightarrow a\alpha$ then the corresponding transition

δ (Transitions on stack)

$\delta(q_0, \epsilon, z_0) = (q_1, sz_0)$

Grammar	PDA
$S \rightarrow aA$	$\delta(q_1, a, s) = (q_1, A)$
$A \rightarrow aABD$	$\delta(q_1, a, A) = (q_1, ABD)$
$A \rightarrow bB$	$\delta(q_1, b, A) = (q_1, B)$
$A \rightarrow a\epsilon$	$\delta(q_1, a, A) = (q_1, \epsilon)$
$B \rightarrow b\epsilon$	$\delta(q_1, b, B) = (q_1, \epsilon)$
$D \rightarrow d\epsilon$	$\delta(q_1, d, D) = (q_1, \epsilon)$

finally step 3 -

change q_1 to q_2 (As q_2 is any my final state)

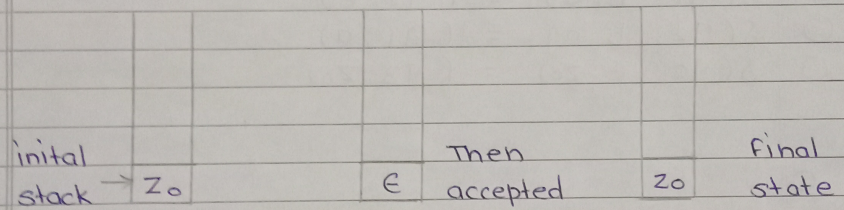
$\delta(q_1, \epsilon, z_0) = (q_2, z_0)$

Lucky | Page No.:
Date : / /

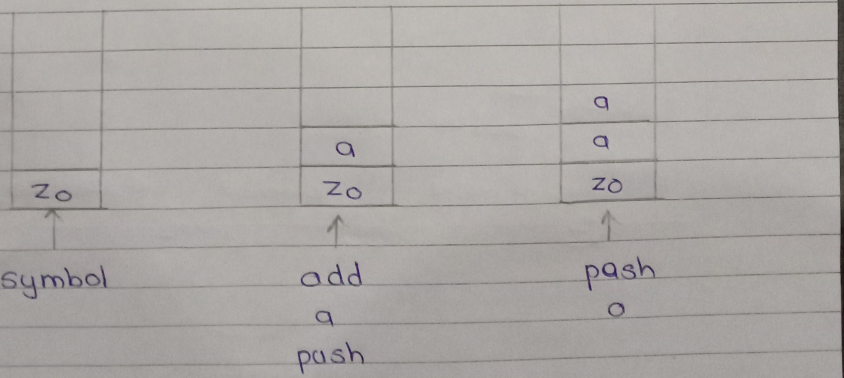
Ex: construct PDA by CFL

$S \rightarrow aSA \mid b$
$A \rightarrow bB \mid b$
$B \rightarrow b$

Initially Empty stack Final stack



The string construction a's and no of b's twice to as follows



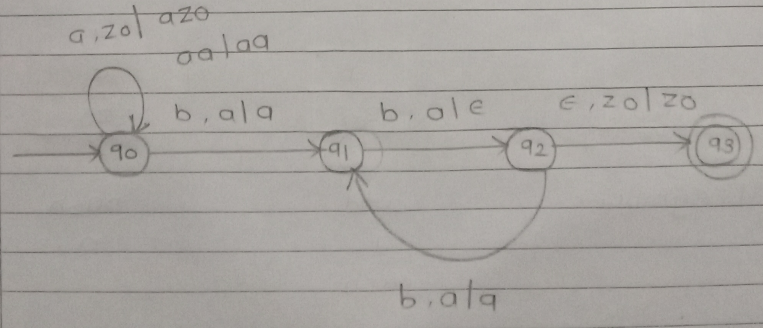
mean we perform , ① Pop
② Nop operation with b's

Transition function,

① $\delta(q_0, a, z_0) = (q_0, a, z_0)$
② $\delta(q_0, a, a) = (q_0, aa)$
③ $\delta(q_0, b, a) = (q_1, a)$
④ $\delta(q_1, b, a) = (q_2, \epsilon)$
⑤ $\delta(q_2, b, a) = (q_1, a)$
⑥ $\delta(q_2, \epsilon, z_0) = (q_3, z_0)$

①

	a			
a	a	a	ϵ	a
z0	z0	z0	z0	z0
push	Push	pop	pop	pop



PDA = { Q, Σ, q_0, δ, Γ, z_0, AIF }

z_0 = Initial stack symbol

Γ = stack symbol

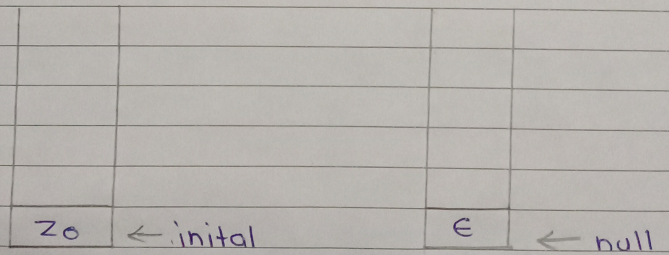
Two way of Acceptance

① Final state :

Top on stack is z_0

② Empty stack :

stack contain ∈ symbol



LIFO

Lucky | Page No.:
Date : / /

Ex : consider a PDA for accept the language $L = \{ a^n b^{2n} \ \forall \ n \geq 1 \}$ by final state

→ consider $n \geq 1$

if $n = 1$ it satisfies
$n > 1$ it satisfies

The string may be $n = 1$

$L = \{ a^1 b^{2 \cdot 1} \} = abb$

② $n > 1$ $n = 2$

$L = \{ a^2 b^{2 \cdot 2} \} = \{ aabbbb \}$

③ $n > 1$ $n = 3$

$L = \{ a^3 b^{3 \cdot 2} \} = \{ aaabbbbbb \}$

We can say that PDA with accept

$L = \{ abb, aabbbb, \ldots\ldots \}$

There are three operation are passible
① Push ③ Nop
② Pop

Ex: consider a PDA for accept the language $L = \{ a^n b^{2n} \ \forall \ n \geq 1 \}$ by final state

→ consider $n \geq 1$

if $n = 1$ it satisfies
$n > 1$ it satisfies

- The string may be $n = 1$

$$L = \{ a^1 b^{2 \cdot 1} \} = abb$$

② $n > 1$ $n = 2$

$$L = \{ a^2 b^{2 \cdot 2} \} = \{ aabbbb \}$$

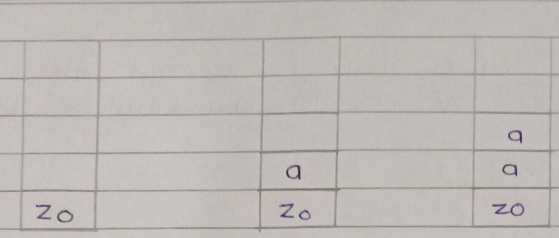
- ③ $n > 1$ $n = 3$

$$L = \{ a^3 b^{3 \cdot 2} \} = \{ aaabbbbbb \}$$

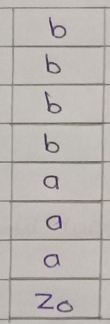
We can say that PDA with accept

$$L = \{ abb, aabbbb, \ldots\ldots \}$$

There are three operation are possible
① Push ③ Nop
② Pop



Initially pusha pusha



ϵ

Lucky | Page No.:
Date : / /

Ex: Given a PDA for accept following Language

$L = \{a^n \cdot b^{2n}\}$ $\forall$ $n \geq 1$ by Final state

→ The starting like { abb, aabbbb ..... }

① Push a
② Push a
③ No operation with b
④ Pop b with a

b	
a	Push a
a	Push a
z_0	Initial stack symbol

① $\delta(q_0, a, z_0) = (q_0, az_0)$

State i/p Top of stack

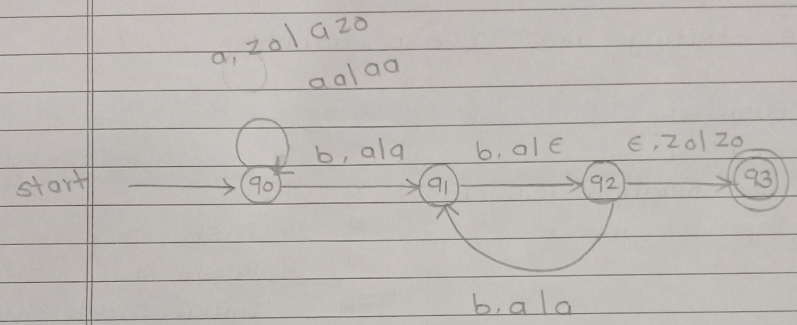
② $\delta(q_0, a, a) = (q_0, aa)$

③ $\delta(q_0, b, a) = (q_1, a)$

④ $\delta(q_1, b, a) = (q_2, \epsilon)$

⑤ $\delta(q_2, b, a) = (q_1, a)$

⑥ $\delta(q_2, \epsilon, z_0) = (q_3, z_0)$ Final state


a, z0 | az0
aa | aa
start
q0
b, a | a
q1
b, a | ∈
q2
∈, z0 | z0
q3
b, a | a

Lucky | Page No.:
Date : / /

* Types of Acceptance in PDA

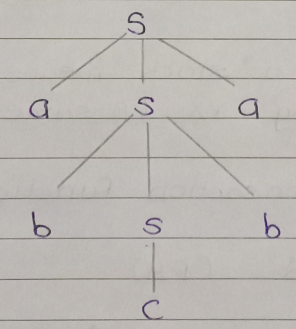
The transaction function of PDA is δ. Denoted by.

$$\delta : Q \times \{\Sigma \cup \{\wedge\}\} \times \Gamma \rightarrow Q \times \Gamma^*$$

Ex. Let G be CFG having $S \rightarrow aSa|bSb|c$ & G generates the language.

$$L = \{x c x^r \mid x \in \{a,b\}^*\}$$

Now CFG will be construct parse tree like.



Hence it will accept abcba string

The above string is palindrome

PDA performs LIFO with the help of Push & Pop

Push - add symbol
pop - remove symbol

The move is depend upon
① current state
② input (next i/p)
③ top symbol of stack

It will gives
① changing states (staying in the same state)

② Replacing the top stack symbol by a string of zero or more symbol

Paping the top symbol on stack means replace it by Λ

Pushing Y onto a stack means replacing the top symbol X by YX (Assume Y on top)

Assume the transaction function

$\delta : Q \times \Sigma \rightarrow Q$ (FA)

$\delta : Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma^*$ (PDA)

For state q, an i/p a & stack symbol X
$\delta(q, a, X) = (p, d)$

state q, with X on top of stack we read a move to state p & replace X by string d.

Everytime remember of if stack is empty then always remember that Z_0 is symbol on stack

If i/p are empty and we want to move to next state then we use $\wedge$ transaction

$$\delta : Q \times \{ \Sigma \cup \{\wedge\} \} \times \Gamma \rightarrow Q \times \Gamma^*$$

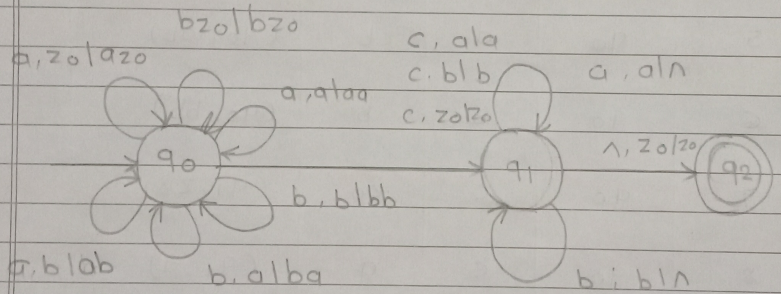


Fig: Pushdown Automata [Transaction diagram]

Move Number	state	Input	stack symbol	Move (s)
1	q_0	a	z_0	(q_0, az_0)
2	q_0	b	z_0	(q_0, bz_0)
3	q_0	a	a	(q_0, aa)
4	q_0	b	a	(q_0, ba)
5	q_0	a	b	(q_0, ab)
6	q_0	b	b	(q_0, bb)
7	q_0	c	z_0	(q_1, z_0)
8	q_0	c	a	(q_1, a)
9	q_0	c	b	(q_1, b)
10	q_1	a	a	(q_1, Λ)
11	q_1	b	b	(q_1, Λ)
12	q_1	Λ	z_0	(q_2, z_0)

[Transaction table]

Lucky | Page No.:
Date : / /

Now take strings : ① abcba
② ab
③ acaa

More Number	Resulting State	unread Input	stack
Initially	q_0	abcba	z_0
1	q_0	bcba	az_0
2	q_0	cba	baz_0
9	q_1	ba	baz_0
11	q_1	a	az_0
10	q_1	-	z_0
12	q_1	-	z_0

Accept

More Number	Resulting state	unread Input	stack
Initially	q_0	acaa	z_0
1	q_0	caa	az_0
8	q_1	aa	az_0
10	q_1	a	z_0
12	q_2	a	z_0

Not accept

If there is no more corresponding to given i/p then it will take '$\wedge$' transaction

$\wedge$ transaction = $\delta(q_0, \wedge, z_0)$

Lucky | Page No.:
Date : / /

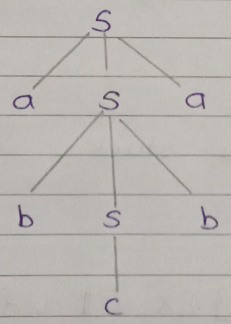
* PDA Example

wcw^R can be represented by cFG

$G = \{V, S, \Sigma, P\}$

$G = \{\{S\}, \{a, b, c\}, S, P\}$

P ⇒ $S \rightarrow aSa$
$S \rightarrow bSb$
$S \rightarrow c$

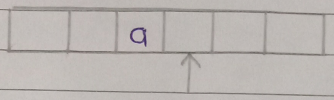


abcba

Tape : Divided into finite no of steps all the symbols are belong to Σ

Stack : All the alphabets belongs to stack Γ

While reading a tape the head of tape moves to right



Tape head

After reading a head goes to right

As we know

$\delta \Rightarrow$ Transition function

$Q \Rightarrow$ Finite sets

$\Sigma \Rightarrow$ Alphabets on tape

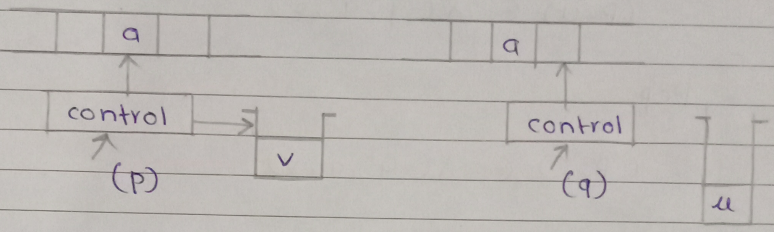
$\epsilon \Rightarrow$ Null symbol

$\Gamma \Rightarrow$ Alphabets on stack

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \rightarrow {}_2Q \times (\Gamma \cup \{\epsilon\})$$

Lucky | Page No.:
Date : / /

* Push & Pop in PDA



$(p, u) \in \delta(q, a, v)$

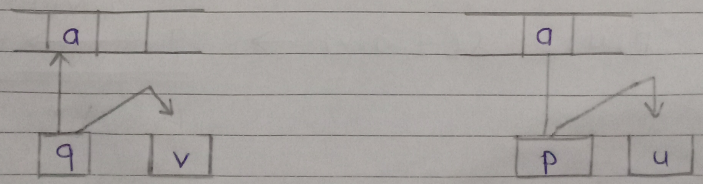
read of tape head $\Rightarrow$ a
stack head read v
Finite control at q

After perform,

tape head more one head right on stack v replaced by u ,

q is replace by p .

if $(p, u) \in \delta(q, \epsilon, v)$



Lucky | Page No.:
Date : / /

Push operation.

$(p, u) \in \delta(q, a, \epsilon)$

Pop

$(p, \epsilon) \in \delta(q, a, v)$

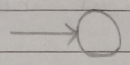
Till the tape head goes rightmost state on type if it accept string then string accepted or it rejected.

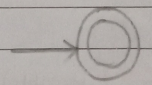
The PDA may 6/7 tuple

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

OR

$(Q, \Sigma, q_0, A, \delta, \Gamma)$

q_0 : initial state 

A Accepting (Final state) 

$(p, u) \in \delta(q, a, v) \rightarrow q \xrightarrow{a, v/u} p$

Lucky | Page No.:
Date : / /

* Ex PDA

wcw^R

i.e,

$L = \{ wcw^R : w \in \{a,b\}^* \}$

All content CFL can be recognized by PDA

$\{a^n b^n\}$ by pigeonhole principle is content free but not regular

where, no of a and b are same

PDA gives unlimited memory with LIFO

Now, $n \geq 0$

If n+1 pigeonhole and n+2 pigeons are there then two pigeons is same hole

Now take,

$x\, y^n z \in L$ with $a^n b^n$

where $y \notin \epsilon$

$x = a^p$

$y = a^q$

$z = a^r b^s$

Lucky | Page No.:
Date : / /

$x y^n z = a^{p+nq+r} b^s$ where $n \geq 0$

From given we can assume no of a's and b's will equal at least for one time so $a^n b^n$ is non regular.