PUSHDOWN AUTOMATA

~~Pumping Lemma~~

* To recognize regular language, we construct NFA & DFA

* To recognize CFL, we construct PDA

* PDA is used to recognize context free language (generated by CFG) with one additional capability: a stack on which it can store a string of "stack symbols"

* Like FSM, PDA changes from state to state but also manipulates the stack.

* Our formal notation for a pushdown automaton (PDA) involves seven components.

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

$Q \rightarrow$ finite set of states

$\Sigma \rightarrow$ finite set of i/p symbols.

$\Gamma \rightarrow$ finite stack alphabets.

$\delta \rightarrow$ transition function

$q_0 \rightarrow$ start state

$Z_0 \rightarrow$ start symbol of stack

$F \rightarrow$ set of accepting states.

PUSHDOWN AUTOMATA

* To recognize regular language, we construct NFA & DFA

* To recognize CFL, we construct PDA.

* PDA is used to recognize context Free language (generated by CFG) with one additional capability: a stack on which it can store a string of "stack symbols"

* Like FSM, PDA changes from state to state but also manipulates the stack.

* Our formal notation for a pushdown automaton (PDA) involves seven components.

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

$Q \rightarrow$ finite set of states

$\Sigma \rightarrow$ finite set of i/p symbols

$\Gamma \rightarrow$ finite stack alphabet.

$\delta \rightarrow$ transition function

$q_0 \rightarrow$ start state

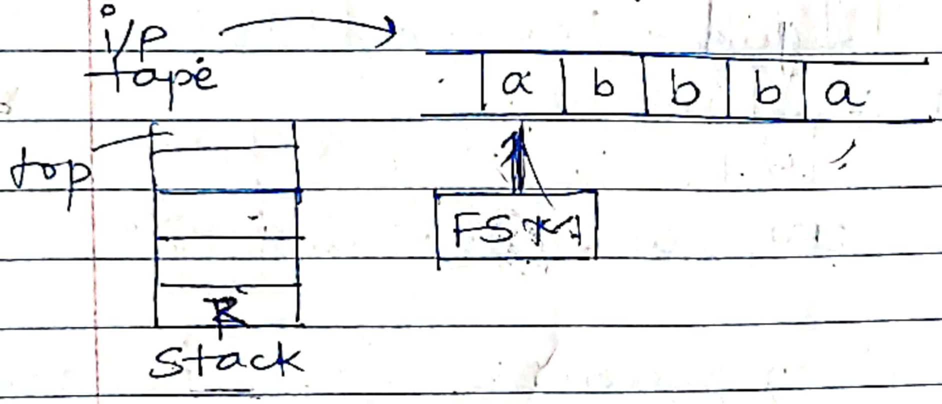
$Z_0 \rightarrow$ start symbol of stack

$F \rightarrow$ set of accepting states.

Reg Lang – if it can be decided if a word is in the Lang with a algo/mac with constant memory by examining all symbols in a word one after other.

All finite languages are regular, but not all reg Lang are finite.

Abstract model of PDA.



a
b
q0
q0

* Components of PDA :-

→ PDA consists of finite set of states, i/p tape, read head and a stack

* Working PDA :

Depending on the state, i/p symbol and the stack top

→ PDA change the state / remain in the same state

→ PDA after reading the i/p symbol, moves the head to the right of current cell.

→ PDA can perform some stack operations.

→ Like FSM, PDA changes from state to state but also manipulates the stack.

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

where,

Q = Finite set of states

Σ = i/p alphabets

Γ = stack alphabets

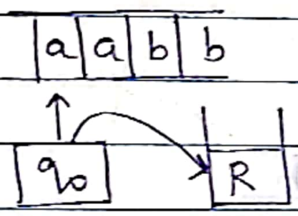
δ = transition function

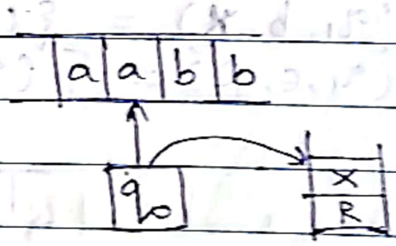
q_0 = start state

Z_0 = initial stack top symbol

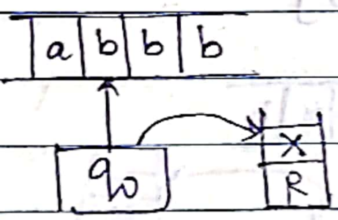
F = Finite set of final states

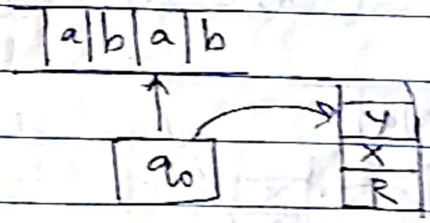
Examples:-



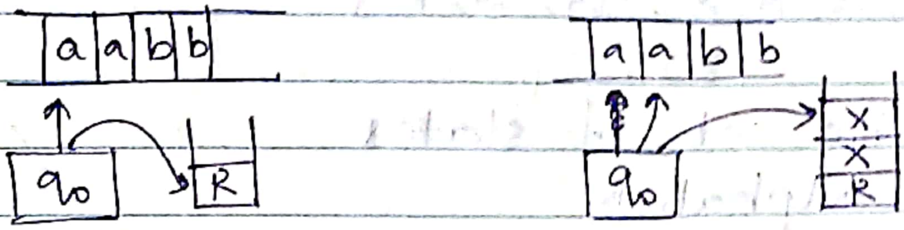


$\delta(q_0, a, R) = \{(q_0, XR)\}$

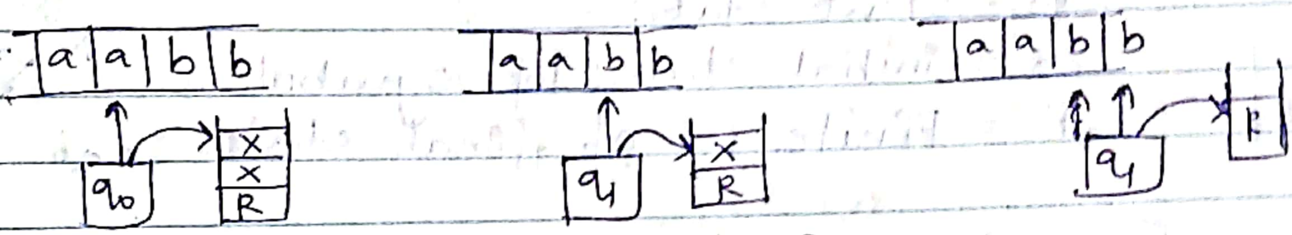




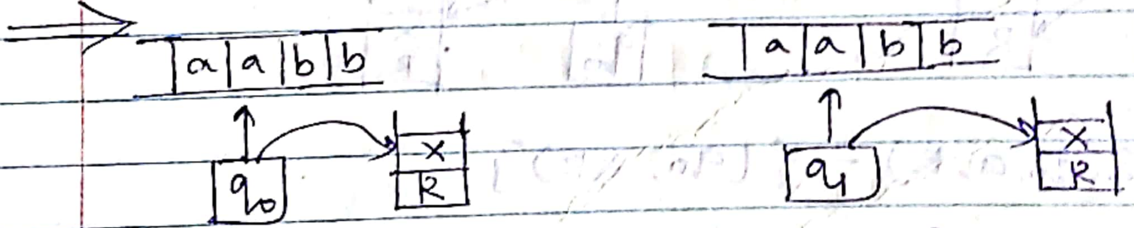
$\delta(q_0, b, X) = \{(q_1, YX)\}$



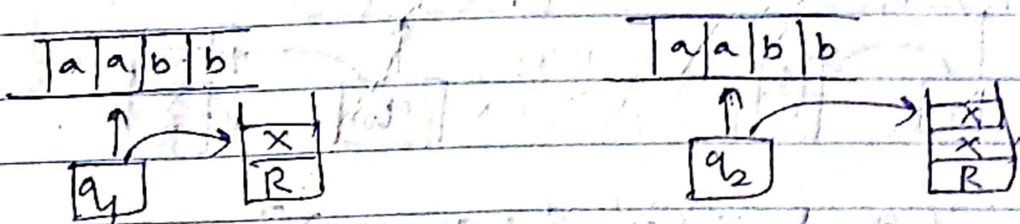
$$\delta(q_0, a, R) = \{(q_0, XXR)\}$$



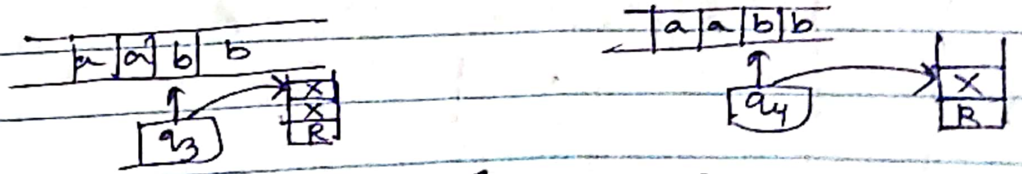
$$\delta(q_0, b, X) = \{(q_1, \epsilon)\}$$
$$\delta(q_1, b, \cancel{X}) = \{(q_1, \epsilon)\}$$
$$\delta(q_1, \epsilon, R) = \{(q_f, R)\}$$



$$\delta(q_0, \epsilon, X) = \{(q_1, X)\}$$

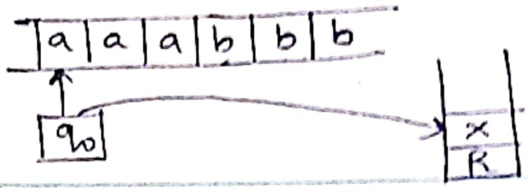


$$\delta(q_1, \epsilon, X) = \{(q_2, XX)\}$$



$$\delta(q_3, \epsilon, X) = \{(q_4, \epsilon)\}$$

Moderator



① Design PDA to recognize $L = \{a^n b^n \mid n \geq 1\}$

⟹ Theory:

For each 'a' push 1 X
For each 'b' pop 1 X

1 Mark

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

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

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

$\Gamma = \{X, R\}$

$q_0 = q_0$

$Z_0 = R$

$F = \{q_f\}$

2 Marks

δ:-

① $\delta(q_0, a, R) = \{(q_0, XR)\}$

$\delta(q_0, a, X) = \{(q_0, XX)\}$

$\delta(q_0, b, X) = \{(q_1, \epsilon)\}$

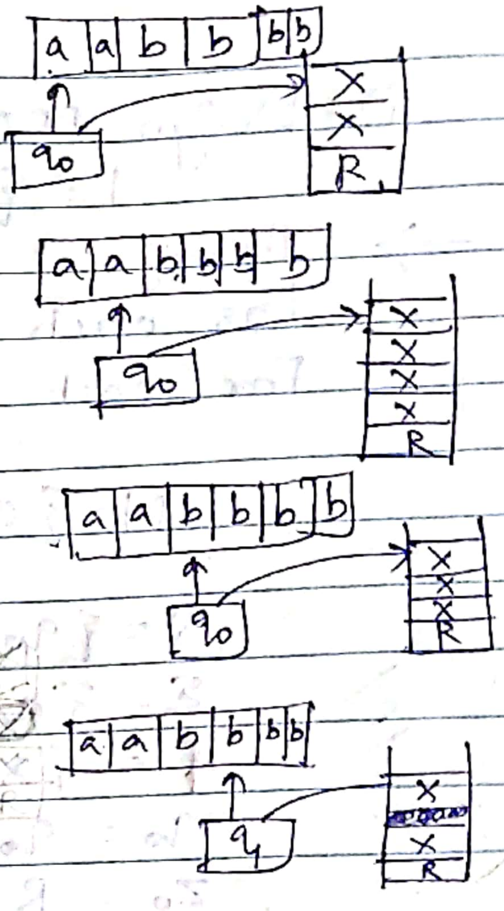
$\delta(q_1, b, X) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, R) = \{(q_f, R)\}$

$(q_0, aabb, R)$	$(q_0, abab, R)$
$\vdash (q_0, abb, XR)$	$\vdash (q_0, bab, XR)$
$\vdash (q_0, bb, XXR)$	$\vdash (q_1, ab, R)$
$\vdash (q_1, b, XR)$	$\vdash$
$\vdash (q_1, \epsilon, R)$	Rejects
$\vdash (q_f, R)$ ∴ accept	

2) Design PDA to recognize

$L = \{a^n b^{2n} \mid n \geq 1\}$



For each 'a' push 2X

For each 'b' pop 1X

$\delta(q_0, a, R) = \{(q_0, XXR)\}$

$\delta(q_0, a, X) = \{(q_0, XXX)\}$

$\delta(q_0, b, X) = \{(q_1, \epsilon)\}$

$\delta(q_1, b, X) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, R) = \{(q_f, R)\}$

3) Design PDA to recognize

$L = \{a^{3n} b^n \mid n \geq 1\}$

For each 3 'a''s push 1X

For each 'b' pop 1X

$\delta(q_0, a, R) = \{(q_1, R)\}$

$\delta(q_1, a, R) = \{(q_2, R)\}$ } start a

$\delta(q_2, a, R) = \{(q_0, XR)\}$

$\delta(q_0, a, X) = \{(q_1, X)\}$

$\delta(q_1, a, X) = \{(q_2, X)\}$ } remaing a

$\delta(q_2, a, X) = \{(q_0, XX)\}$

$\delta(q_0, b, X) = \{(q_3, \epsilon)\}$ first b

$\delta(q_3, b, X) = \{(q_3, \epsilon)\}$ remaing b

$\delta(q_3, \epsilon, R) = \{(q_f, R)\}$

4) Design PDA to recognize

$L = \{a^{2n} b^{3n} \mid n \geq 1\}$

For every 2 'a's push 1X

For every 3 'b's pop 1X

$\delta(q_0, a, R) = \{(q_1, R)\}$

$\delta(q_1, a, R) = \{(q_0, XR)\}$

$\delta(q_0, a, X) = \{(q_1, X)\}$

$\delta(q_1, a, X) = \{(q_0, XX)\}$

$\delta(q_0, b, X) = \{(q_2, X)\}$

$\delta(q_2, b, X) = \{(q_3, X)\}$

$\delta(q_3, b, X) = \{(q_4, \epsilon)\}$

$\delta(q_4, b, X) = \{(q_2, X)\}$

$\delta(q_4, \epsilon, R) = \{(q_f, R)\}$

5) Design a PDA to recognize

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

For every 'a' push 1X

For every 'b' push 1X

For every 'c' pop 1X

$\delta(q_0, a, R) = \{(q_0, XR)\}$

$\delta(q_0, a, X) = \{(q_0, XX)\}$

$\delta(q_0, b, X) = \{(q_1, XX)\}$

$\delta(q_1, b, X) = \{(q_1, XX)\}$

$\delta(q_1, c, X) = \{(q_2, \epsilon)\}$

$\delta(q_2, c, X) = \{(q_2, \epsilon)\}$

$\delta(q_2, \epsilon, R) = \{(q_f, R)\}$

contd...

Pushdown Automata

6) Design a PDA to recognize $L = \{a^n b^{n+1} \mid n \geq 1\}$

For each a push 1X

For each b ★★★ pop 1X $\Longrightarrow$ Pass the first b

δ:-

$\delta(q_0, a, R) = \{(q_0, XR)\}$
$\delta(q_0, a, X) = \{(q_0, XX)\}$
$\delta(q_0, b, X) = \{(q_1, X)\}$
$\delta(q_1, b, X) = \{(q_2, \epsilon)\}$
$\delta(q_2, b, X) = \{(q_2, \epsilon)\}$
$\delta(q_2, \epsilon, R) = \{(q_f, \cancel{R})\}$

VQ ★★★

7) Design PDA to recognize $L = \{\underset{q_0}{a^n} \underset{q_1}{b^m} \underset{q_2}{a^n} \mid m, n \geq 1\}$

For each a push 1X

Bypass all b's

For each a pop 1X

δ:-

$\delta(q_0, a, R) = \{(q_0, XR)\}$
$\delta(q_0, a, X) = \{(q_0, XX)\}$
$\delta(q_0, b, X) = \{(q_1, X)\}$
$\delta(q_1, b, X) = \{(q_1, X)\}$
$\delta(q_1, a, X) = \{(q_2, \epsilon)\}$
$\delta(q_2, a, X) = \{(q_2, \epsilon)\}$

$\delta(q_2, \epsilon, R) = \{(q_f, \epsilon)\}$

//

8) Design a PDA to recognize $L = \{a^m b^n c^n d^m \mid m, n \geq 1\}$

For each a push 1X
for each b push 1Y
for each c pop 1Y
for each d pop 1X

δ:-

$\delta(q_0, a, R) = \{(q_0, XR)\}$
$\delta(q_0, a, X) = \{(q_0, XX)\}$
$\delta(q_0, b, X) = \{(q_1, YX)\}$
$\delta(q_1, b, Y) = \{(q_1, YX)\}$
$\delta(q_1, c, Y) = \{(q_2, \epsilon)\}$
$\delta(q_2, c, Y) = \{(q_2, \epsilon)\}$
$\delta(q_2, d, X) = \{(q_3, \epsilon)\}$
$\delta(q_3, d, X) = \{(q_3, \epsilon)\}$
$\delta(q_3, \epsilon, R) = \{(q_3, \epsilon)\}$

aa

abab
baab aabb
$q_0, a,$

9) Design a PDA to recognize equal number of a's and b's over $\Sigma = \{a, b\}$

S:-

$\delta(q_0, a, R) = \{(q_0, XR)\}$ aabab
$\delta(q_0, a, X) = \{(q_0, XX)\}$
$\delta(q_0, b, X) = \{(q_0, \epsilon)\}$ bbabaaa
$\delta(q_0, b, R) = \{(q_0, YR)\}$ bbaaaab
$\delta(q_0, b, Y) = \{(q_0, YY)\}$
$\delta(q_0, a, Y) = \{(q_0, \epsilon)\}$
$\delta(q_0, \epsilon, R) = \{(q_f, \epsilon)\}$

UC
★★★

10) Design a PDA to recognize $L = \{WCW^R \mid W \in (a, b)^*$ b, bab babb
$W^R \in$ Reverse of $W\}$ $\underbrace{aaba}_{w}$ c $\underbrace{abaa}_{w^R}$

S:-

$\delta(q_0, a, R) = \{(q_0, XR)\}$
$\delta(q_0, b, R) = \{(q_0, YR)\}$

bab c bab

$\delta(q_0, a, X) = \{(q_0, XX)\}$
$\delta(q_0, b, X) = \{(q_0, YX)\}$
$\delta(q_0, b, Y) = \{(q_0, YY)\}$
$\delta(q_0, a, Y) = \{(q_0, XY)\}$

$\delta(q_0, c, R) = \{(q_1, R)\}$ By pass 'c'
$\delta(q_0, c, X) = \{(q_1, X)\}$
$\delta(q_0, c, Y) = \{(q_1, Y)\}$

Scanned with OKEN Scanner

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

$\delta(q_1, a, X) = \{(q_1, \epsilon)\}$
$\delta(q_1, b, Y) = \{(q_1, \epsilon)\}$
$\delta(q_1, \epsilon, R) = \{(q_1, \epsilon)\}$

PDA

11) This is a case of NPDA
Design a PDA to recognize
$L = \{ww^R \mid w \in \{a, b\}^*$
w^R, Reverse of $w\}$

Whenever there is a double letter their are two possibilities

abba

(i) Middle of string is not reached, so we continue pushing resp symbols on stack.
(ii) Middle of string is reached so we start popping the resp. symbols from the stack.

δ:
$\delta(q_0, \epsilon, R) = \{(q_0, \epsilon)\} \Rightarrow$ coz w can be empty

$\delta(q_0, a, R) = \{(q_0, XR)\}$
$\delta(q_0, b, R) = \{(q_0, YR)\}$

$\delta(q_0, a, X) = \{(q_0, XX), (q_1, \epsilon)\}$ ☆
$\delta(q_0, b, X) = \{(q_0, YX)\}$
$\delta(q_0, a, Y) = \{(q_0, XY)\}$
$\delta(q_0, b, Y) = \{(q_0, YY), (q_1, \epsilon)\}$ ☆

$\delta(q_1, a, X) = \{(q_1, \epsilon)\}$

$\delta(q_1, b, Y) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, R) = \{(q_1, \epsilon)\}$

example

$(q_0, \cancel{aabba}\ baab, R)$

$\vdash (q_0, aab, YR)$

$\vdash (q_0, ab, XYR)$

not reached — reached partition

$\vdash (q_0, b, XXYR)$ $\quad\quad$ $\vdash (q_1, b, YR)$

$\vdash (q_0, \epsilon, YXXYR)$ $\quad\quad$ $\vdash (q_1, \epsilon, R)$

accepted

for conversion CFG to PDA

prod^n & pop

PDA to CFG

CFG to PDA

~~Step~~ Step 1 : Express CFG in GNF

Step 2 : PDA $M = \{Q, \Sigma, \Gamma, \delta, q_0, z_0, F\}$

(1) $S \to aSa \mid bSb \mid c$

$S \to aSC_1 \mid bSC_2 \mid c$

$C_1 \to a$

$C_2 \to b$

$Q = \{q_0\}$

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

$\Gamma = \{S, C_1, C_2\}$

$q_0 = q_0$

$Z_0 = S$

$F = \{\}$

δ :-

CFG	PDA
$S \to aSC_1$	$\delta(q_0, a, S) = \{(q_0, SC_1)\}$
$S \to bSC_2$	$\delta(q_0, b, S) = \{(q_0, SC_2)\}$
$S \to c$	$\delta(q_0, c, S) = \{(q_0, \epsilon)\}$
$C_1 \to a$	$\delta(q_0, a, C_1) = \{(q_0, \epsilon)\}$
$C_2 \to b$	$\delta(q_0, b, C_2) = \{(q_0, \epsilon)\}$
	$\delta(q_0, \epsilon, R) = \{(q_f, R)\}$