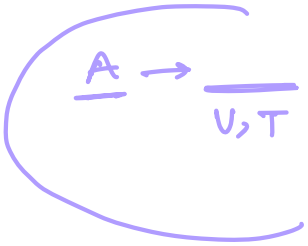
Normal forms of CFG



- CNF (Chomsky Normal form)
- GNF (Greibach Normal form)

Chomsky Normal form (CNF):

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

- A non terminal generating a terminal ($A \to a$)
- A non terminal generating 2 non-terminals ($A \to BC$)
- Start symbol generating ε ($S \to \varepsilon$)

Eg:

$S \to a$
$S \to AZ$
$A \to a$
$Z \to b$

↓

CFG is in CNF

$S \to a$
$S \to aZ$
$Z \to b$

↓

CFG is not in CNF.

Properties:

- For a given grammar, there can be more than 1 CNF
- CNF produces the same language as generated by CFG. (CFG → CNF)
- For generating a string of length n, you require $2n-1$ steps in CNF.

GNF: $S \to AB \mid AS$

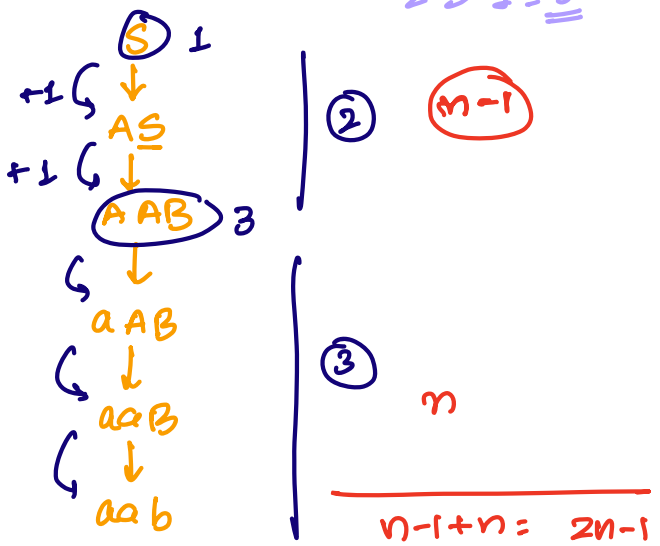
$A \to a$

$B \to b$

"aab" $n=3$

$2n-1$ steps

$2*3-1=5$



- Any CFG that donot have ε in its language has an equivalent GNF.

Conversion from CFG to GNF ?

1. Eliminate start symbol from RHS

$S \to SA$
$A \to a$
.

⟶

$S' \to S$
$S \to SA$
$A \to a$

new start symbol = S'

2. Eliminate null, unit & useless productions

3. $A \to B_1 B_2 B_3 \cdots B_n \quad n > 2$

$A \to B_1 C$

$C \to B_2 B_3 \cdots B_n$

Repeat till 2 variables in RHS

$V \to VV$
$V \to T$
$S \to \varepsilon$

4. $A \to aB$

A → CB
C → a

Eg:

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

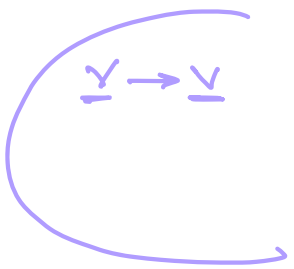
Convert to CNF?

1. $S' \rightarrow S$

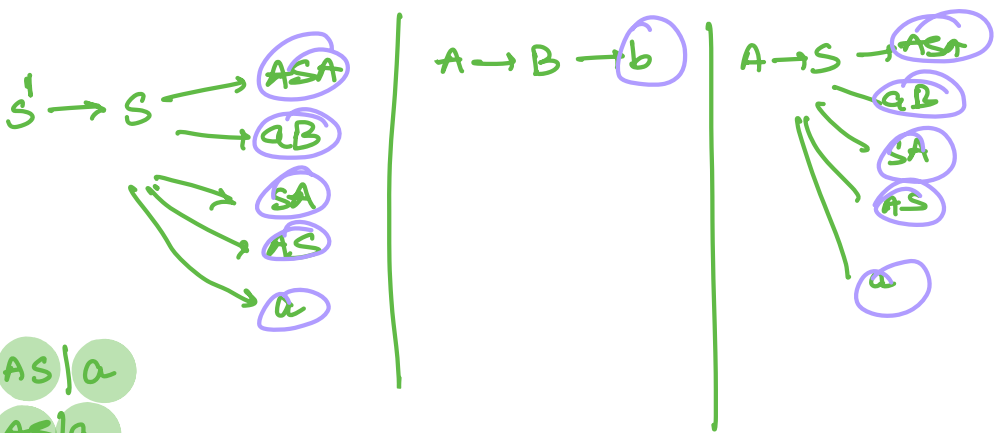
$S \rightarrow ASA \mid aB$
$A \rightarrow B \mid S$
$B \rightarrow b \mid \varepsilon$

2. null productions $A \rightarrow B \rightarrow \varepsilon$ $\{A, B\}$

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

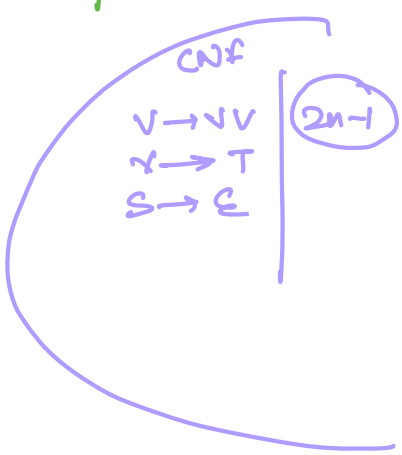


unit



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

all useful symbols



3.

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

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

$A \to b \mid AC \mid aB \mid SA \mid AS \mid a$

$B \to b$

$C \to SA$

4.

$S' \to AC \mid DB \mid SA \mid AS \mid a$

$S \to AC \mid DB \mid SA \mid AS \mid a$

$A \to b \mid AC \mid DB \mid SA \mid AS \mid a$

$B \to b$

$C \to SA$

$D \to a$

CNF form

Eg:

$S \to ASB \mid b$

$A \to aAS \mid a \mid \varepsilon$

$B \to SbS \mid A \mid bb$

Convert CFG to CNF?

1.

$S' \to S \mid b$

$S \to ASB \mid b$

$A \to aAS \mid a \mid \varepsilon$

$B \to SbS \mid A \mid bb$

2.

nullable =

$B \to A \to \varepsilon$

nullable = {A, B}

$S' \to S \mid b$

$S \to ASB \mid SB \mid AS \mid b$

$A \to aAS \mid a \mid aS$

$B \to SbS \mid A \mid bb$

unit

S' → ASB | SB | AS | b
S → ASB | SB | AS | b
A → aAS | a | as
B → SbS | bb | aAS | a | as

B → A → aAS ✓
→ a ✓
→ as ✓

useless X

3.

S' → ASB | SB | AS | b
S → ASB | SB | AS | b
A → aAS | a | as
B → SbS | bb | aAS | a | as

S' → CB | SB | AS | b
S → CB | SB | AS | b
A → aC | a | aS
B → SbS | bb | aC | a | aS
C → AS

S' → CB | SB | AS | b
S → CB | SB | AS | b
A → DC | a | DS
B → SbS | bb | DC | a | DS
C → AS
D → a

S' → CB | SB | AS | b

S → CB | SB | AS | b

A → DC | a | DS

B → SG | ff | DC | a | DS

C → AS

D → a

F → b

G → FS

Converting CFG to GNF:

→ V → T A → a

→ V → T VVV.... A → aBCD....

→ S → ε (epsilon)

Eg: S → aA | bB
B → bB | b ✓ GNF
A → aA | a

S → aA | bB
B → bB | ε GNF ✗
A → aA | ε

- For a given grammar, more than 1 GNF is possible
- Language generated by GNF & by CFG should be same.

Conversion from CFG to GNF:

1. Convert grammar to CNF
2. If left recursion exists, remove it.

3. convert productions to GNF

→ NT → T

NT → T NT NT NT NT ....

S → ε

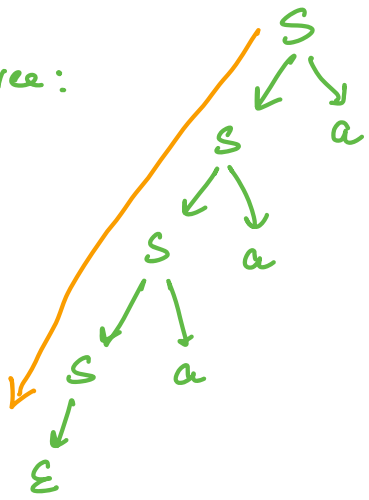
Left Recursion:

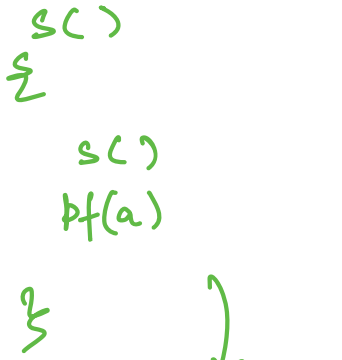
- Production in which left most symbol of RHS = symbol present on LHS.

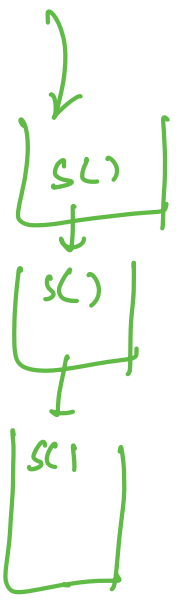
- Grammar having a production with left recursion, such a grammar is called as **Left Recursive Grammar.**

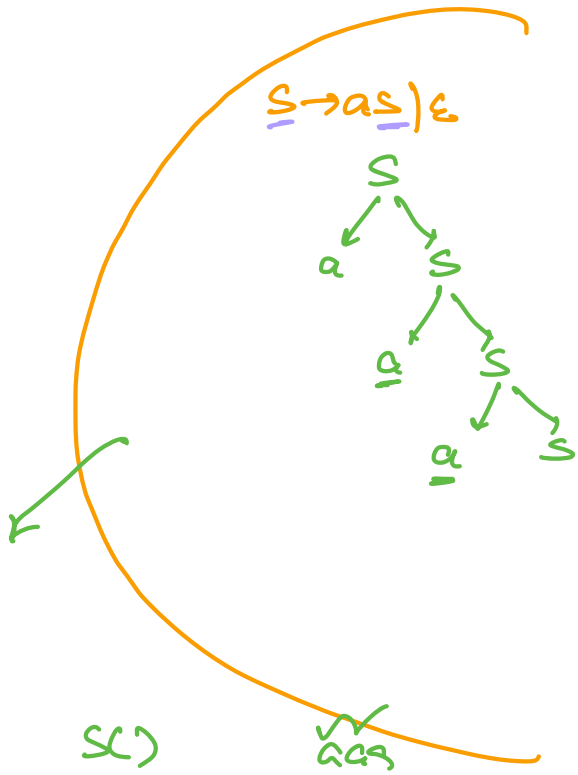
S → Sa | ε

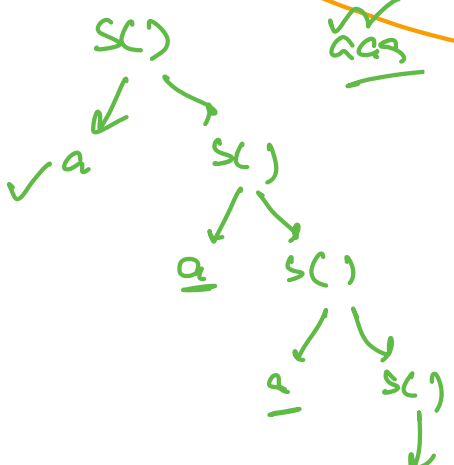
Parse Tree:











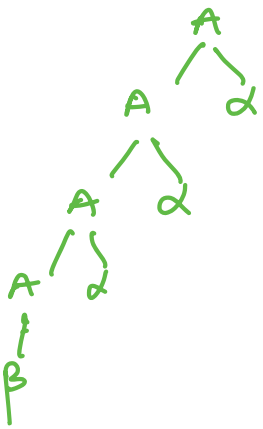
Remove left Recursion

$A \longrightarrow A\alpha \mid \beta$

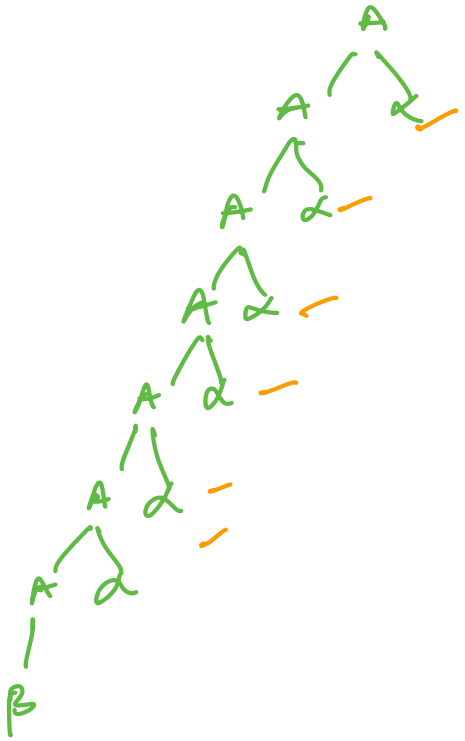
$\downarrow$

$A \rightarrow \beta A'$

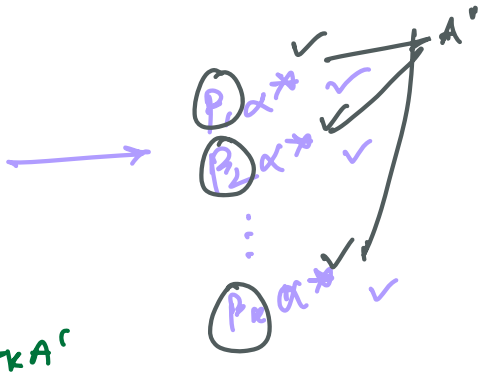
$A' \rightarrow \alpha A' \mid \varepsilon$



language $\beta\alpha^*$



$A \rightarrow A\alpha \mid \beta_1 \mid \beta_2 \ldots\ldots \beta_k$



$\downarrow$

$A \rightarrow \beta_1 A' \mid \beta_2 A' \mid \beta_3 A' \ldots\ldots \mid \beta_k A'$

$A' \rightarrow \alpha A' \mid \varepsilon$

Eg:

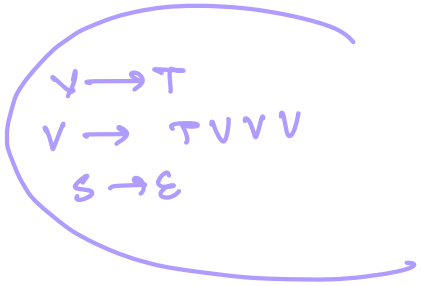
$S \rightarrow XB \mid AA$

$A \rightarrow a \mid SA$

$B \rightarrow b$

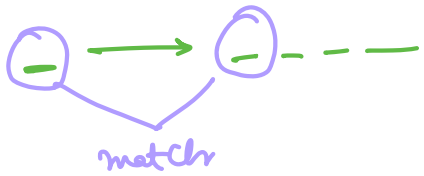
$X \rightarrow a$

CFG $\rightarrow$ GNF ?



1. CNF: $V \to T$, $V \to VV$, $S \to \varepsilon$ ✓

2. left Rec.



3.

$S \to XB \mid AA$

$A \to a \mid SA$

$B \to b$

$X \to a$

↓

$S \to XB \mid AA$

$A \to a \mid XBA \mid AAA$

$B \to b$

$X \to a$

↓

$S \to aB \mid AA$

$A \to a \mid aBA \mid AAA$

$B \to b$

$X \to a$

↓ Removed left Recursion

$S \to aB \mid AA$

$A \to aA' \mid aBAA'$

$A \to A\alpha \mid \beta_1 \mid \beta_2$

↳ $\beta\alpha^*$

$A \to \beta_1 A' \mid \beta_2 A'$

$A' \to \alpha A' \mid \varepsilon$ — new productions

$A \to a \mid aBA \mid AAA$ (β_1, β_2, α)

$A \to aA' \mid aBAA'$

$A' \to AAA' \mid \varepsilon$

$A' \rightarrow AAA' \mid \varepsilon$

$B \rightarrow b$

$X \rightarrow a$

Remove ε production

$S \rightarrow aB \mid AA$

$A \rightarrow aA' \mid aBAA' \mid a \mid aBA$

$A' \rightarrow AAA' \mid AA$

$B \rightarrow b$

$X \rightarrow a$

$S \rightarrow aB \mid aA'A \mid aBAA'A \mid aA \mid aBAA$

$A \rightarrow aA' \mid aBAA' \mid a \mid aBA$

$A' \rightarrow AAA' \mid AA$

$B \rightarrow b$

$X \rightarrow a$

final ans (GNF form)

$S \rightarrow aB \mid aA'A \mid aBAA'A \mid aA \mid aBAA$

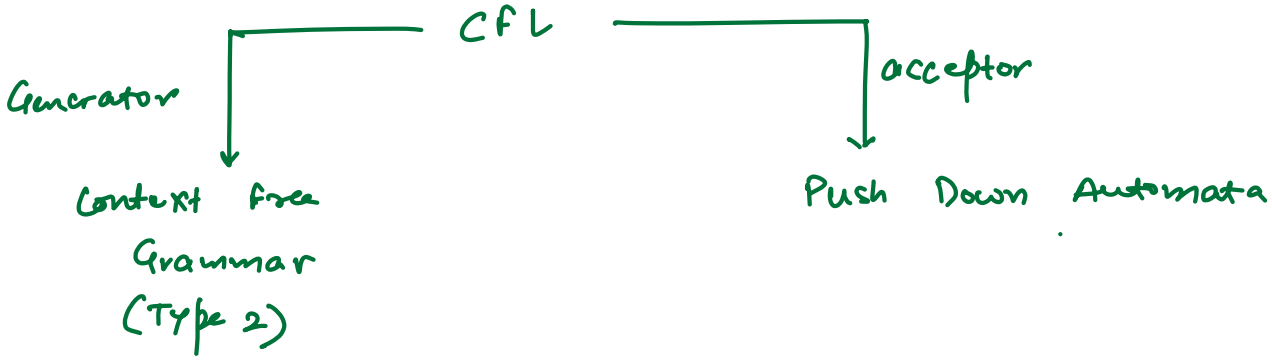
$A \rightarrow aA' \mid aBAA' \mid a \mid aBA$

$A' \rightarrow aA'AA' \mid aBAA'AA' \mid aAA' \mid aBAAA' \mid aA'A \mid aBAA'A \mid aA \mid aBAA$

$B \rightarrow b$

Context Free Languages:

RL → FA
CFL → PDA
memory



Push Down Automata:

$Q, \Sigma, \delta, q_0, Z_0, F, \Gamma$ → Stack alphabet

- Q → finite set of states
- Σ → input alphabet
- δ → Transition function
- q_0 → Initial State
- Z_0 → Bottom of stack
- F → Set of final states

Power
✓ DFA = NFA
DPDA < NPDA

PDA

- Deterministic (DPDA)
- Non Deterministic (NPDA)

$Q \times (\Sigma \cup \varepsilon) \times \Gamma \longrightarrow Q \times \Gamma^*$

$Q \times (\Sigma \cup \varepsilon) \times \Gamma \longrightarrow 2^{(Q \times \Sigma^*)}$

Input alphabet or ε

Symbol from top of stack

You are on same state Q, you see some input alphabet and u decide to go to more than 1 state & push more than 1 symbol on stack

Eg: $a^n b^n \mid n \geq 1$

aabbε

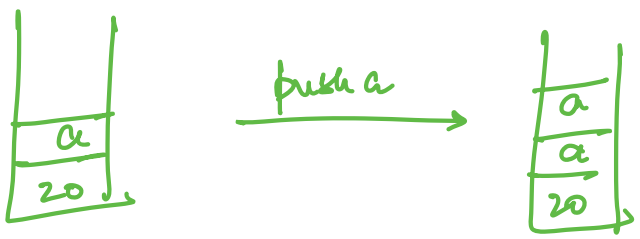
(1)

input alphabet ↑ stack top ↑ new top of stack

a, Z0 / aZ0 → Push

aabbε

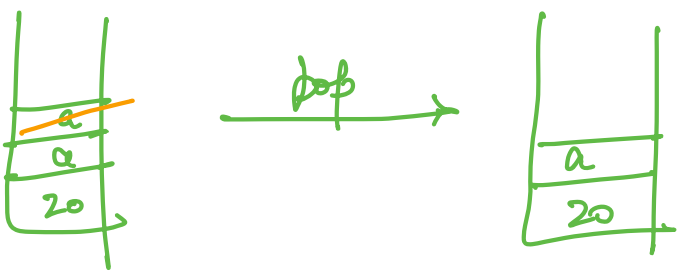
(2)



a, a / aa → Push

aabbε

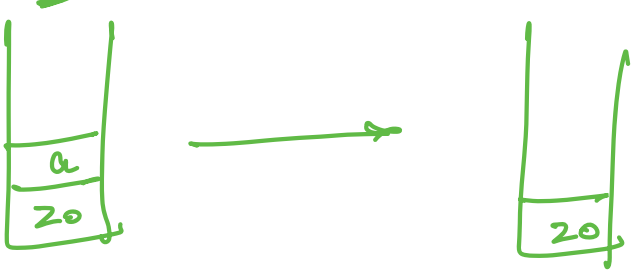
(3)



b, a / ε pop

aabbε

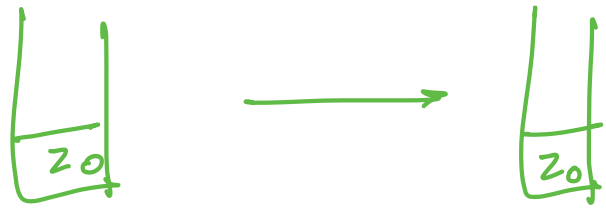
(4)



b, a / ε pop

aabbε

(5)

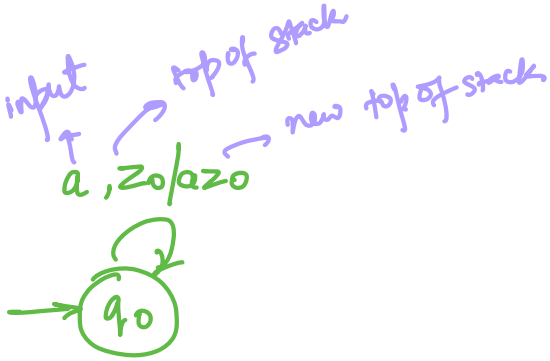


ε, Z0 / Z0 — keep as it is

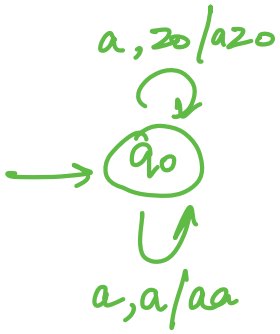
top of stack is Z0 & input is ε

String is accepted

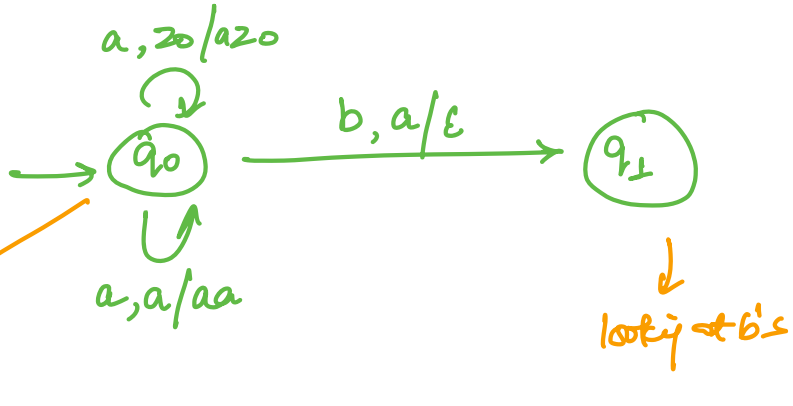
(1)



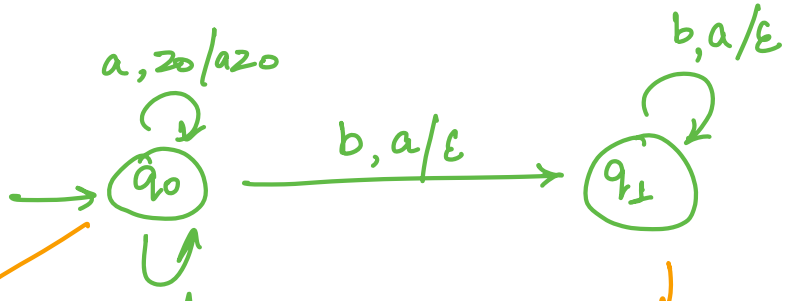
(2)



(3)



(4)

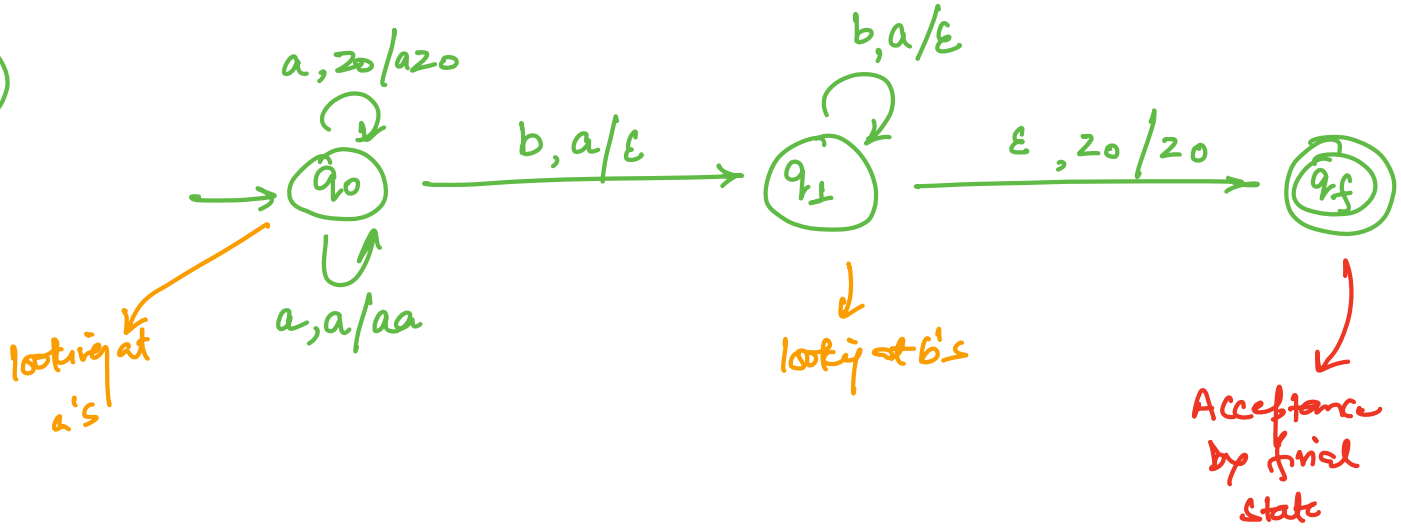


looking at a's

a, a/aa

looking at b's

(5)



bba ?

(A)

$\delta(q_0, a, z_0) \longrightarrow (q_0, az_0)$

new state

new top of stack

state

input symbol

stack top

$\delta(q_0, a, a) \longrightarrow (q_0, aa)$

$\delta(q_0, b, a) \longrightarrow (q_1, \varepsilon)$

(B) $\delta(q_1, \varepsilon, z_0) \longrightarrow (q_f, z_0)$ } Acceptance by final state