PDA

PDA

- Pushdown Automata
- PDA is the Class of Automata associated with CFL
- Finite Automata cannot recognize all context free languages as FA has strictly finite memory whereas the recognition of a CFL may require storing an unbounded amount of information.

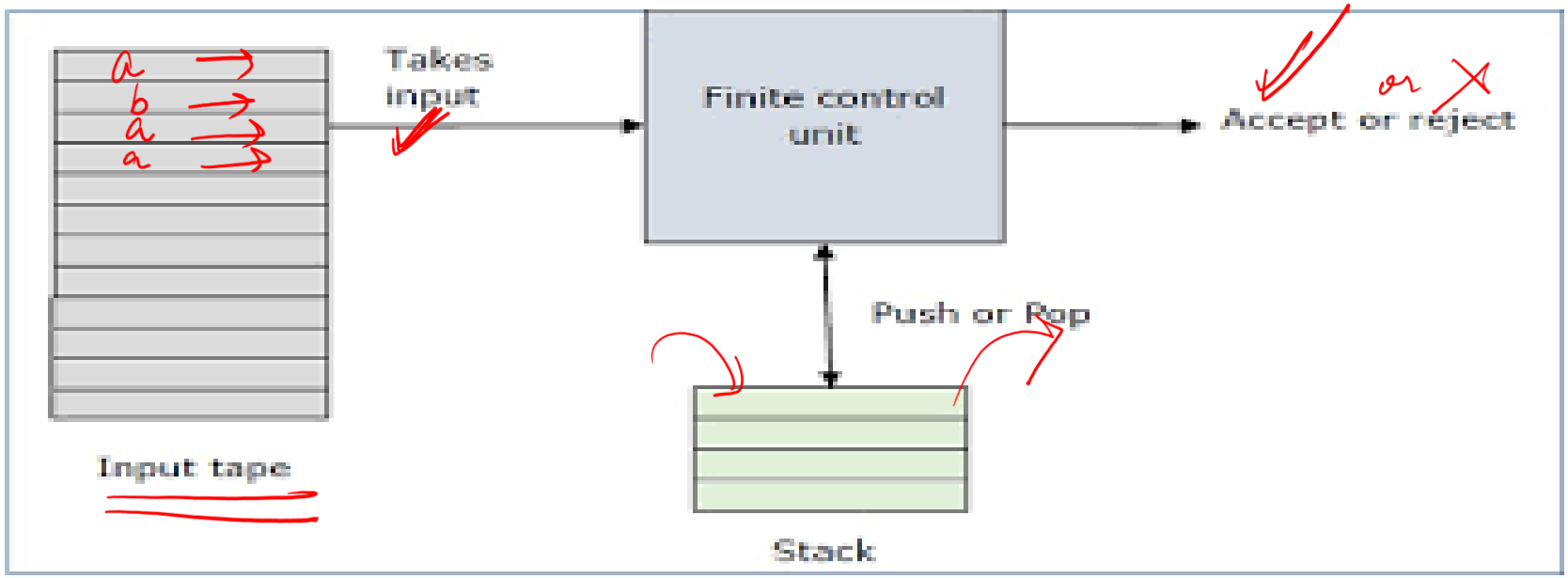
PDA

- Eg-
- $L=\{a^n b^n | n>=0\}$
- When scanning the string, we must check that all a's are precede the first b, we also need to count the number of a's
- Since n is unbounded, this counting cannot be done with a finite memory.
- We want a machine that can count without limit

PDA

- A pushdown automaton is –

"Finite state machine" + "a stack"



- A pushdown automaton has three components –

1) an input tape,
2) a control unit, and
3) a stack with infinite size.

- The stack head scans the top symbol of the stack.

Deterministic PDA: Formal Definition

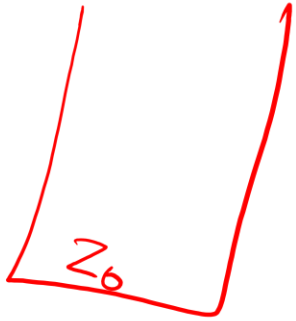
- A list of seven elements is called a 7-tuple,
- A PDA can be defined as a 7-tuple

PDA: Formal Definition

PDA is denoted by 7 Tuple: M=(Q, ∑, Γ, δ, q_0, z, F) where

- Q:Finite set of Internal states of the Control Unit
- ∑:Finite input alphabet
- Γ: Finite Set of Stack Symbols/Pushdown symbols
- δ:Q X (∑ U {ε}) X Γ ->Q X Γ *: Transition function(TF)

  or
- $\delta: Q \times \sum_{\varepsilon} \times \Gamma_{\varepsilon} \rightarrow Q \times \Gamma_{\varepsilon}^{*}$
- q_0: Start state of Control Unit, q0 ∈ Q
- z : Stack start Symbol, z ∈ Γ, Initially present in the stack
- F: Set of Final States/ Accept State, F ⊆ Q



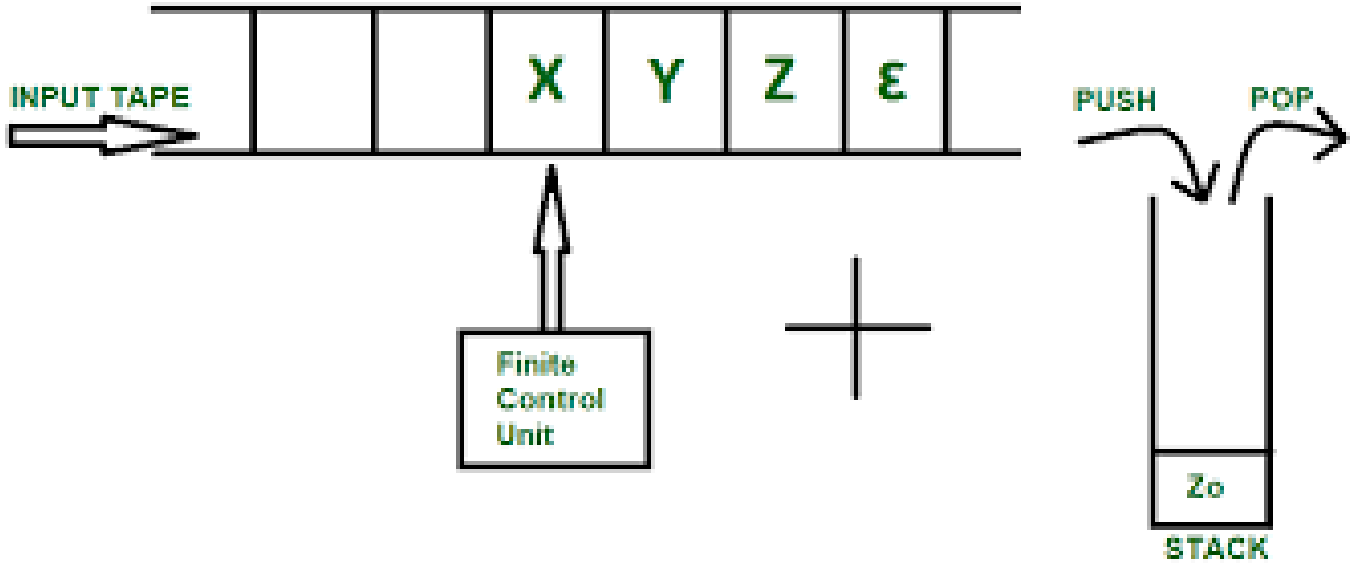
δ Function

δ Function:-

- **δ:Q X (∑ U {ε} X r ->Q X r *: Transition function(TF)**

or

- **δ:Q X $\sum_{\varepsilon}$ X r_{ε} ->Q X r_{ε} ***



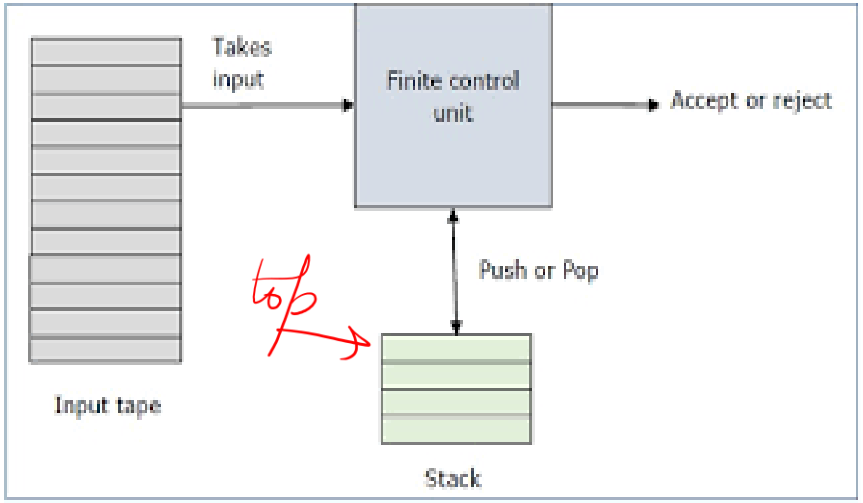
δ Function

δ Function:-

- **δ:Q X (∑ U {ε} X Γ ->Q X Γ *: Transition function(TF)**

or

- **δ:Q X $\sum_{\varepsilon}$ X Γ_{ε} ->Q X Γ_{ε} ***



Domain-

- **The Arguments are :-**

1) **Current State of the control Unit**
2) **Current Input Symbol**
3) **Current Symbol on the Top of the stack**

Range-

- **The Set of pair (q,x) where :-**

1) **q is Next state of the control Unit**
2) **x is a string which is put on the top of the stack , in place of the single symbol there before**

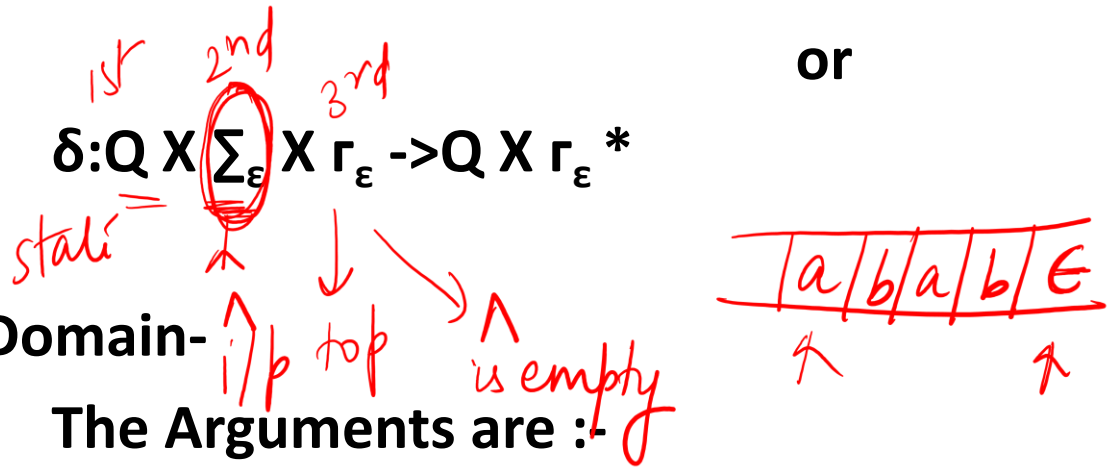
δ Function

δ Function:-

- **δ:Q X (∑ U {ε} X Γ ->Q X Γ *: Transition function(TF)**

or

- **δ:Q X $\sum_{\varepsilon}$ X Γ_{ε} ->Q X Γ_{ε} ***



Domain-

- **The Arguments are :-**

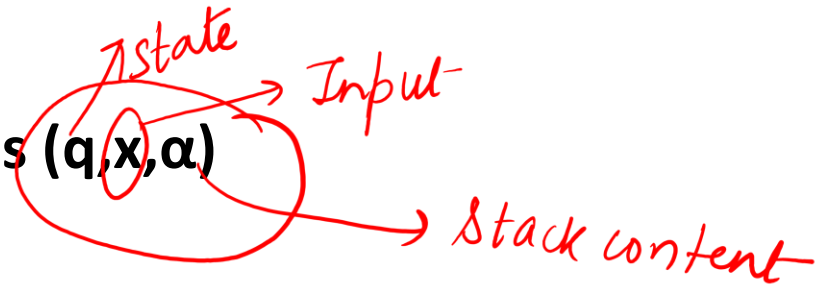
1) **Current State of the control Unit**
2) **Current Input Symbol- Can be ε, indicating a move that does not consume an input symbol, Also called ε or Null Transition**
3) **Current Symbol on the Top of the stack- δ is defined so that it needs a stack symbol, no move is possible if the stack is empty**

Instantaneous Description

Let M=(Q, ∑, r, δ, q_0, z, F) be a pda

An Instantaneous Description(ID) is (q,x,α)

where q ∈ Q, x ∈ $\sum_{\varepsilon}$, α ∈ r_{ε}



1) q is the Current State of the control Unit
2) x is the Unread part of the input string/The part of the input to be processed
   - Say a1,a2,….an
   - The PDA will process in a1,a2,….an order only
3) α are the stack contents with the left most symbol indicating the top pf the stack

- This triplet is called ID

ID

diff from δ rules

Instantaneous Description

- **A move from one ID to another will be denoted by Turnstile notation**

- ⊢ sign is called a “turnstile notation” and represents one move.

Move Relation-

Let M be a PDA

A move relation between IDs are defined as:-

$(q, a1a2\ldots\ldots an, z1z2\ldots.zn) \vdash (q',a2\ldots\ldots an,\beta z2\ldots.zn)$

if $\delta(q,a1,z1)=(q', \beta)$

PDA Example 1

Design a PDA for accepting the language $L=\{a^n b^n \mid n>=1\}$

Logic-

Equal no of a's & Equal no of b's

⇒ i.e. n no of a's & n no of b's

⇒ for every single occurence of 'a'
we will push 'a' onto the stack

⇒ for every single 'b'
we will pop a single 'a' from the stack

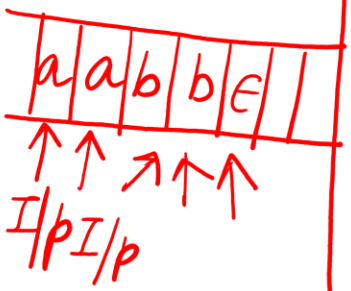
⇒ If we have read all b's & the stack is empty after popping 'a's

⇒ Then we accept the string

PDA Example 1

Design a PDA for accepting the language $L=\{a^n b^n \mid n>=1\}$

Rules-



Input tape

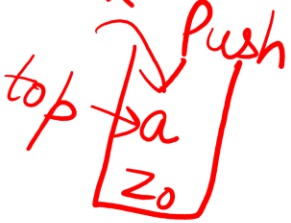
δ Rules

Let us start with "aabb" Transition Rules

$\delta(q_0, a, z_0) = (q_1, a z_0)$

I/p top — new string being inserted — top

top of stack



stack

States

Initial state q_0

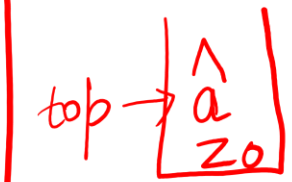
Push operation q_1 state has changed

$\delta(q_1, a, a) = (q_1, aa)$

new string being inserted — top

Push operation q_1 state

$\delta(q_1, b, a) = (q_2, \Lambda)$

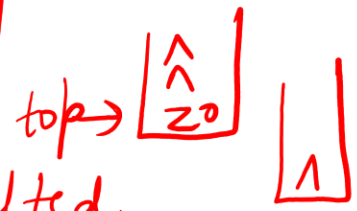


Pop operation change state q_2

$\delta(q_2, b, a) = (q_2, \Lambda)$

Pop again

$\delta(q_2, \in, z_0) = (q_f, \Lambda)$



Move to final state q_f

Stack is empty, string accepted

PDA Example 1

Design a PDA for accepting the language $L=\{a^n b^n \mid n>=1\}$

Simulation-

String "aaabbb"

ID ⇒ Instantaneous Description

turnstile notation

$(q_0, aaabbb, z_0) \vdash (q_1, aabbb, az_0) \vdash (q_1, abbb, aaz_0) \vdash$

$(q_1, bbb, aaaz_0) \vdash (q_2, bb, 1aaz_0) \vdash (q_2, b, \wedge\wedge az_0) \vdash$

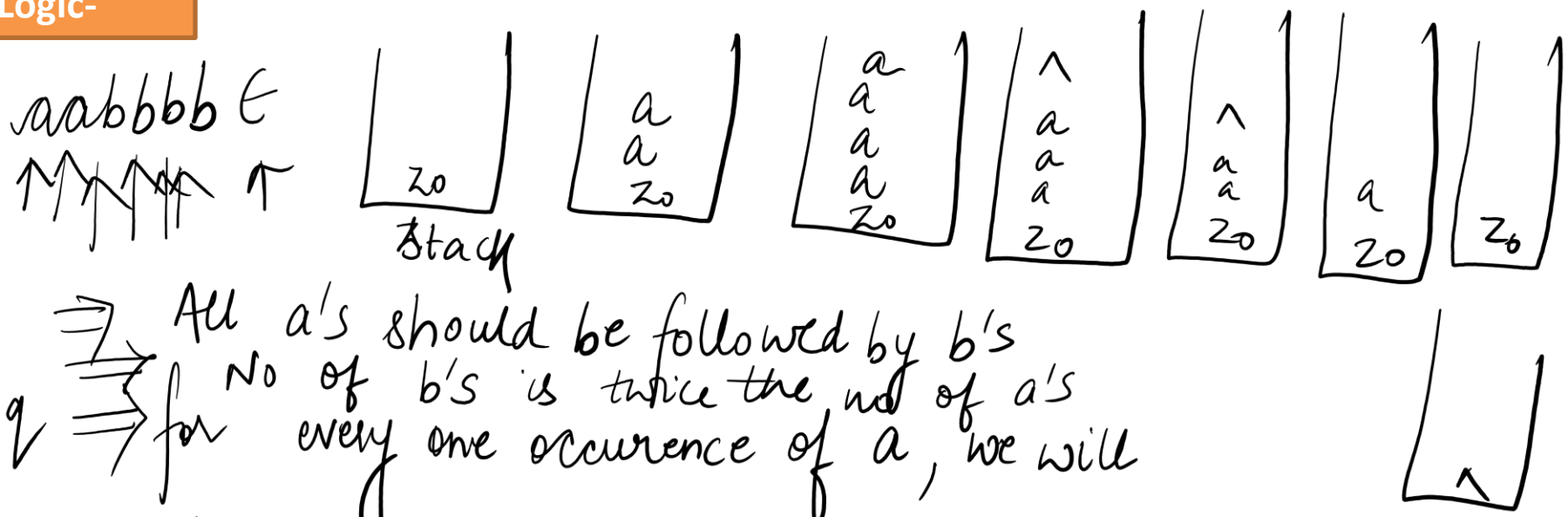
$(q_2, 1, z_0) \vdash (q_f, 1)$

Simulation

PDA Example 2

Design a PDA for accepting the language $L=\{a^n b^{2n} \mid n>=1\}$

Logic-



⇒ All a's should be followed by b's

⇒ No of b's is twice the no of a's

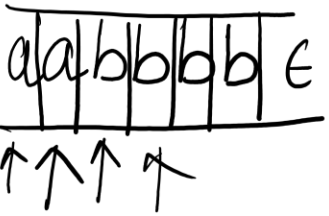
⇒ for every one occurence of a, we will push two a's

⇒ for every one occurence of b, we will pop, single 'b's.

PDA Example 2

Design a PDA for accepting the language $L=\{a^n b^{2n} \mid n>=1\}$

Rules-



δ Rules

$\delta(q_0, a, z_0) = (q_1, aaz_0)$

$\delta(q_1, a, a) = (q_1, aa\,a)$

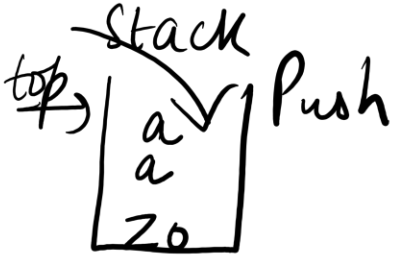
string top inserted

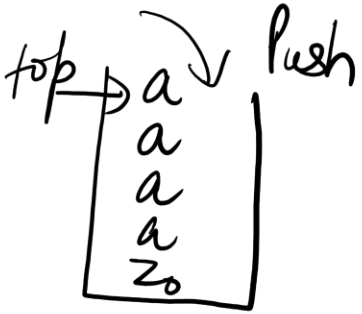
$\delta(q_1, b, a) = (q_2, \wedge)$

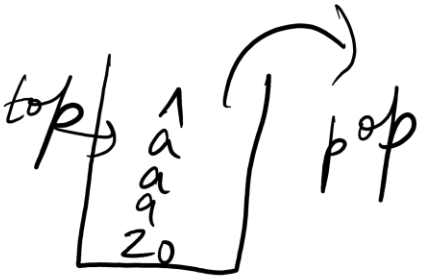
$\delta(q_2, b, a) = (q_2, \wedge)$

$\delta(q_2, \wedge, z_0) = (q_f, \wedge)$

Stack







State

q_0 Initial

Push operation q_1 State

q_1 State

q_2 State

PDA Example 2

Design a PDA for accepting the language $L=\{a^n b^{2n} \mid n>=1\}$

Simulation-

PDA Example 3

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w)=n_b(w)$ }

Logic-

⇒ Equal no of a & bs

⇒ If Initially we read a & stack is empty top = Z_0, Push

⇒ If Initially we read b, top Z_0, Push

⇒ If we read a, top = a, Push

⇒ If we read b, top = b, Push

⇒ If we read a, top = b, Pop

⇒ If we read b, top = a, Pop

⇒ If the string ends, we read ∧, top = Z_0, we Pop & reach final state ⇒ accepted

PDA Example 3

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w)=n_b(w)$ }

Transition Rules-

- **String "aababb"**

① $\delta(q_0, a, z_0) = (q_0, az_0)$ Push

② $\delta(q_0, b, z_0) = (q_0, bz_0)$ Push

→ ③ $\delta(q_0, a, a) = (q_0, aa)$ Push

④ $\delta(q_0, b, b) = (q_0, bb)$ Push

⑤ $\delta(q_0, a, b) = (q_0, \wedge)$ Pop

→ ⑥ $\delta(q_0, b, a) = (q_0, \wedge)$ Pop

⑦ $\delta(q_0, \wedge, z_0) = (q_f, \wedge)$

⇒ δ Rules

Stack is empty

String accepted

PDA Example 3

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w)=n_b(w)$ }

Simulation-

$(q_0, aababb, z_0) \vdash (q_0, ababb, az_0) \vdash (q_0, babb, aaz_0)$

$\vdash (q_0, abb, aaz_0) \vdash (q_0, bb, aaaz_0) \vdash (q_0, b, aaz_0)$

$\vdash (q_0, a, az_0) \vdash (q_0, \wedge, z_0) \vdash (q_f, \wedge)$

String accepted

If p top p

PDA Example 4

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w) > n_b(w)$ }

Logic- no of a's > greater than no of b's

- **String “aababab”**

PDA Example 4

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w) > n_b(w)$ }

Transition Rules-

$\delta(q_0, a, Z_0) = (q_0, aZ_0)$

$\delta(q_0, b, Z_0) = (q_0, bZ_0)$

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

$\delta(q_0, b, b) = (q_0, bb)$

} Push

$\delta(q_0, a, b) = (q_0, \Lambda)$

$\delta(q_0, b, a) = (q_0, \Lambda)$

} Pop

$\delta(q_0, \Lambda, a) = (q_f, a)$

top → a

PDA Example 4

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w) > n_b(w)$ }

aababab

Simulation-

$(q_0, aababab, z_0) \vdash (q_0, ababab, az_0)$

$\vdash (q_0, babab, aaz_0) \vdash (q_0, abab, \wedge az_0) \vdash$

$(q_0, bab, aaz_0) \vdash (q_0, ab, \wedge az_0) \vdash (q_0, b, aaz_0)$

$(q_0, \wedge, az_0) \vdash (q_f, a)$

String accepted

PDA Example 5

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w) < n_b(w)$ }

Logic-

no of b's greater than no of a's

- **String “abbab”**

$$\delta(q_0, \wedge, b) = (q_f, b)$$

PDA Example 5

Design a PDA for accepting the language $L=\{w \mid w \in (a + b)^*$ and $n_a(w)<n_b(w)\}$

Transition Rules-

PDA Example 5

Design a PDA for accepting the language L={w | w ∈ (a + b)* and $n_a(w) < n_b(w)$ }

Simulation-

PDA Example 6

Design a PDA that accepts a string of well formed paranthesis.

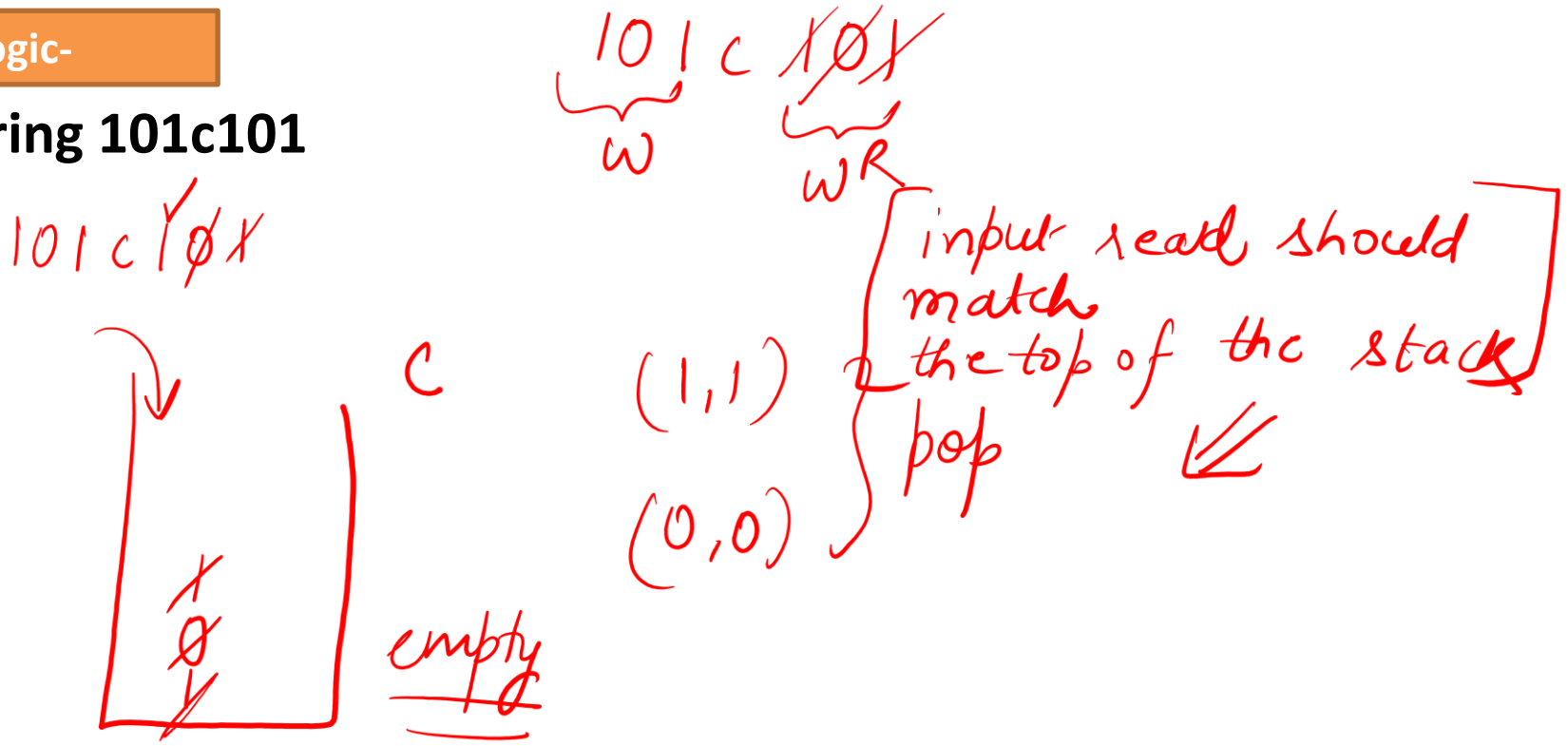
Consider the paranthesis as (,),{,},[,]

PDA Example 7

Design a PDA for language L={wcw^R|w is in (0|1)* and wR is reverse of w}

Logic-

String 101c101



PDA Example 7

Design a PDA for language L={wcw^R | w is in (0|1)* and wR is reverse of w}

Logic-

- The PDA will go on pushing the symbols onto the stack till it encounters c in the input
- It reads c but will not push it onto the stack NOP
- For every symbol read after c, it checks whether it matches with the topmost symbol of the stack
- If input read is 0 ,top=0,pop
- If input read is 1,top=1,pop
- If input ends, we reach zo, replace z0 by null
- Stack empty
- String accepted

PDA Example 7

Design a PDA for language L={wcw^R | w is in (0|1)* and wR is reverse of w}

Transition Rules

$\delta(q_0, 0, z_0) = (q_0, 0z_0)$

$\delta(q_0, 1, z_0) = (q_0, 1z_0)$

} Initially, we will push

Before reading 'c'

$\delta(q_0, 0, 0) = (q_0, 00)$

$\delta(q_0, 0, 1) = (q_0, 01)$

$\delta(q_0, 1, 0) = (q_0, 10)$

$\delta(q_0, 1, 1) = (q_0, 11)$

} Push

We are reading 'c'

String = "c"

10|c|01

$\delta(q_0, c, z_0) = (q_1, z_0)$

$\delta(q_0, c, 0) = (q_1, 0)$

$\delta(q_0, c, 1) = (q_1, 1)$

No operation

NOP

$\Rightarrow$

$\delta(q_1, 0, 0) = (q_1, \wedge)$

$\delta(q_1, 1, 1) = (q_1, \wedge)$

Pop

after reading c

$\delta(q, \wedge, z_0) = (q_f, \wedge)$

Deterministic PDA

Non-Deterministic PDA

NPDA: Formal Definition

NPDA is denoted by 7 Tuple: M=(Q, ∑, r, δ, q_0, z, F) where

- Q:Finite set of Internal states of the Control Unit
- ∑:Finite input alphabet
- r: Finite Set of Stack Symbols/Pushdown symbols
- δ:Q X (∑ U {ε}) X r ->$2^{Q \times r^*}$: Transition function(TF)

  or
- δ:Q X $\sum_\varepsilon$ X r_ε ->$2^{Q \times r^*}$
- q_0: Start state of Control Unit, q0 ∈ Q
- z : Stack start Symbol, z ∈ r, Initially present in the stack
- F: Set of Final States/ Accept State, F ⊆ Q

NDFA

2^Q

Non Determinism-Example 1

Design a PDA for language L={ww^R|w is non-empty even palindromes over {a,b} , wR is reverse of w}

Initial a and b read , when top=z0, it will push

Further, If we read a, top=a, Two Moves possible

1) **Pop and advance input tape head one position**
2) **Push the element read and advance input tape head one position**

Further, If we read b, top=b, Two Moves possible

1) **Pop and advance input tape head one position**
2) **Push the element read and advance input tape head one position**

Non Determinism-Example 1

Even length palindromes

String "baab"

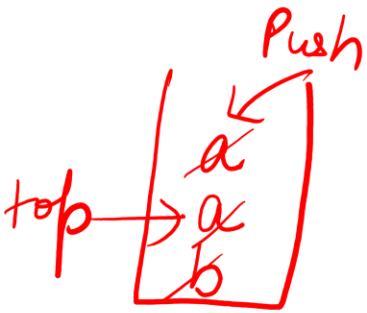
$w : w^R$

$\delta(q, a, a)$

pop

String "baaaab"

$w \; w^R$



$\delta(q, a, a)$

pushing

$\delta(q, a, a)$? pop

$\delta(q, b, b)$? pushing

pop

push

Non-Determinism

NPDA

Non Determinism-Example 1

Design a PDA for language L={ww^R|w is non-empty even palindromes over {a,b} , wR is reverse of w}

δ(q0,a,zo)=(q0,az0)

δ(q0,b,zo)=(q0,bz0)

δ(q0,a,a)={(q0,aa),(q1, ε)}

δ(q0,b,b)={(q0,bb),(q1, ε)}

NDPA

δ(q1,a,a)=(q1, ε)

δ(q1,b,b)=(q1, ε)

δ(q1,ε,z0)=(q1, ε)

Practice