# Unit 5: Pushdown Automata

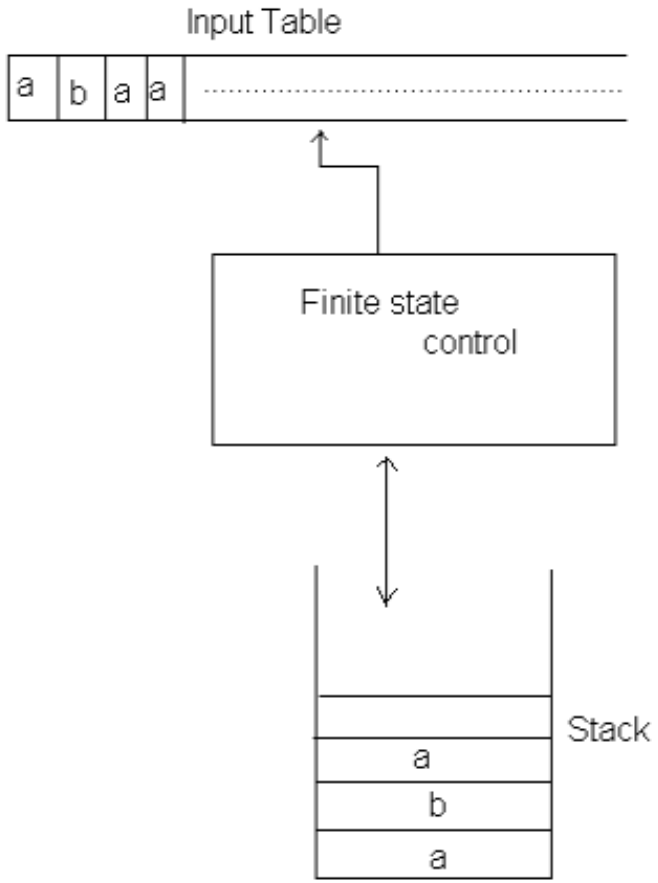
By Prashant Gautam

# Introduction

- The context free languages have a type of automaton that defines them.
- This automaton is called “pushdown automaton” which can be thought as a Є-NFA with the addition of stack.
- The presence of a stack means that, the pushdown automata can remember infinity amount of information.
- However, the pushdown automaton can only access the information on its stack in a Last-in-first-out way.

# Components of PDA

- Thus, PDA is an abstract machine determined by following three things:
  - Input table
  - Finite stack control
  - A stack



# Move in PDA

- Each moves of the machine is determined by three things:
  - The current state
  - Next input symbol
  - Symbol on the top of stack

- The moves consist of
  - Changing state | staying on same state
  - Replacing the stack top by string of zero or more symbols.

# Point to remember

- Popping the top symbol off the stack means replacing it by Є.
- Pushing Y on the stack means replacing stack's top, say X, by YX. Assuming the left end of stack corresponds to stack's top. The single move of the machine contains only one stack operation either push or pop.
- Replacing the stack symbol X by the string α can be accomplished by a sequence of basic moves (a pop followed by sequence of 0 or more pushes).

# Formal Definition

- A PDA P := ( Q,∑,Γ, δ,$q_0$,$Z_0$,F ):
  - Q: states of the ε-NFA
  - ∑: input alphabet
  - Γ : stack symbols
  - δ: transition function
  - $q_0$: start state
  - $Z_0$: Initial stack top symbol
  - F: Final/accepting states

*old state* *input symb.* *Stack top* *new state(s)* *new Stack top(s)*

$$\delta : Q \times \sum \times \Gamma \Rightarrow Q \times \Gamma$$

# Moves in PDA : Formal Definition

- moves of PDA can be interpreted as;
- δ(q, a, z) = {(P1, r1) (p2, r2) …….(pm, rm)}

where q, pi ε Q,

- a ε Σ &
- z ε Γ,
- ri ε Γ*

It means that the PDA in state q with input symbol a and stack top z will go into state pi and replace z by ri and advances a next input symbol.

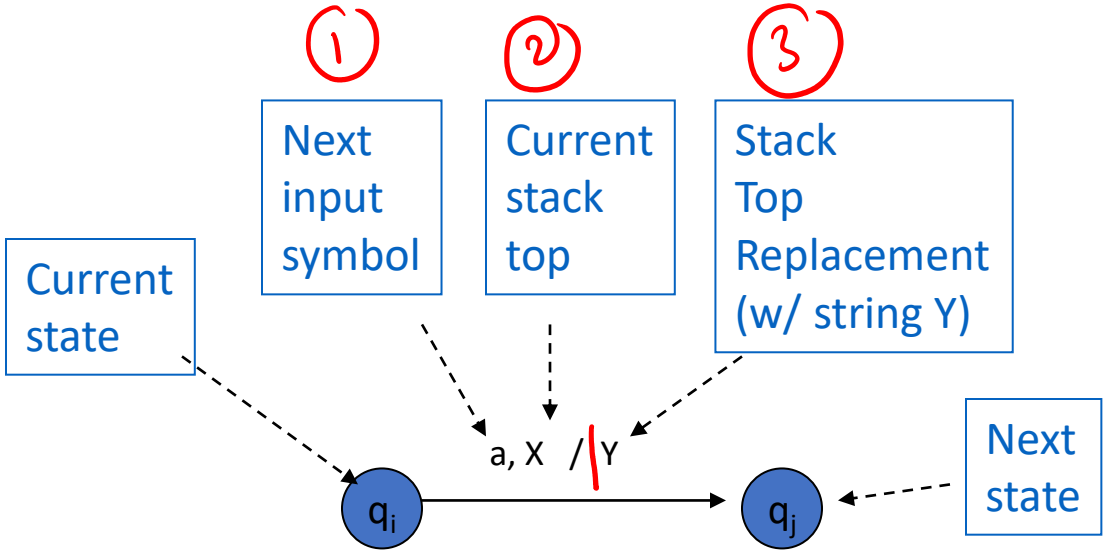
If a = Є

δ(q, Є, z) = (p, Є)

# PDA as a state diagram

$\delta(q_i, a, X) = \{(q_j, Y)\}$

$\delta(q_i, a, X) = (q_j, Y)$



# Construct a PDA that accepts a language over string (a,b) where no.s of a's are equal to no.s of b's

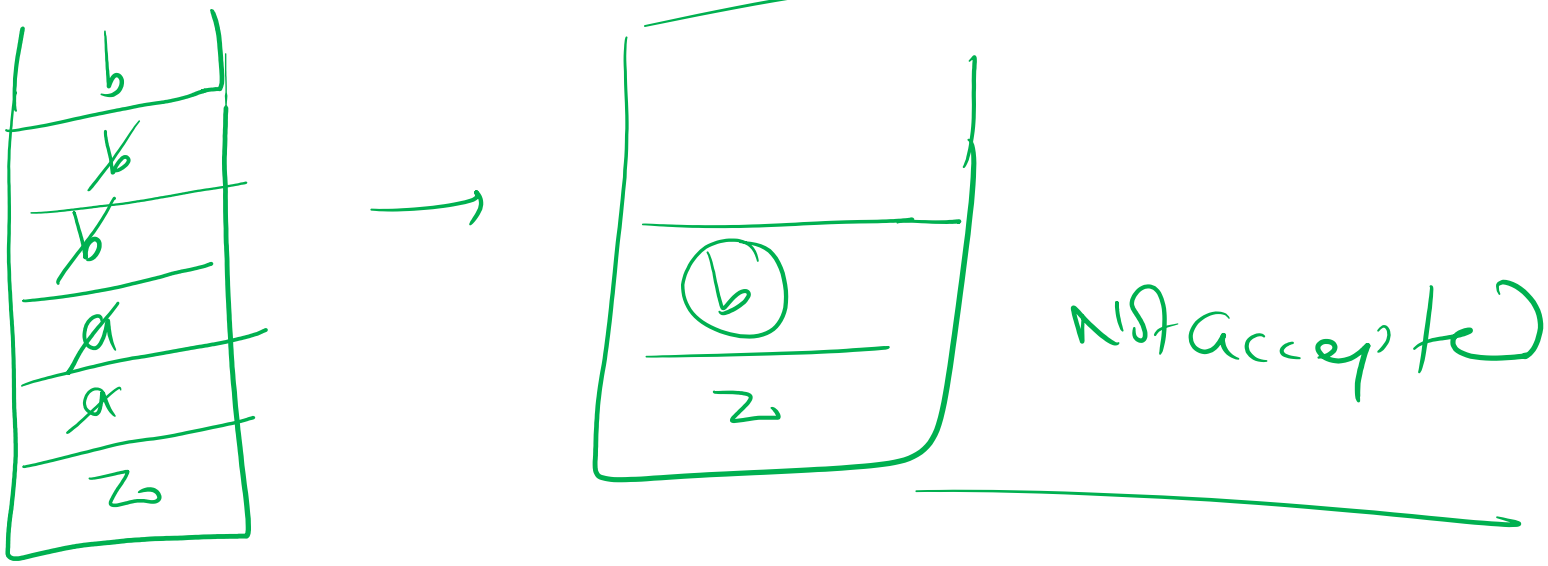
$L = \{\epsilon, ab, ba, abab, baba, abaabb, \ldots\}$

Condition

1. a's no. = b's no.
2. $|w| = 0$

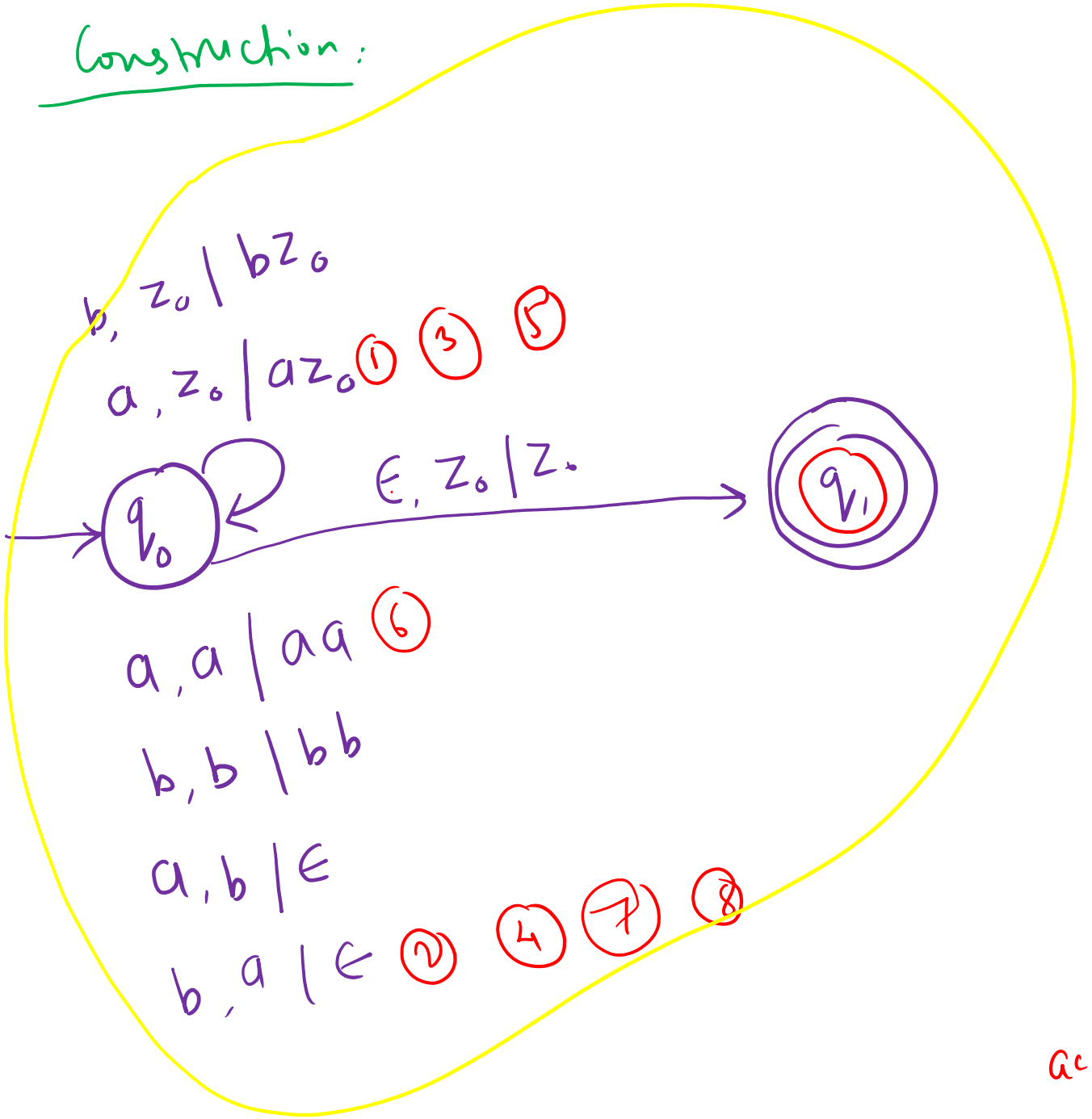
aabbb

Logic

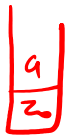



NIf accepted.

Construction:

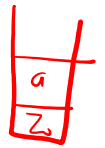



abababbaa ϵ

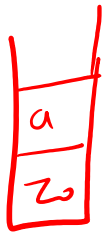



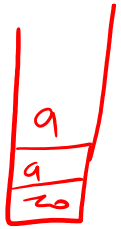


accepted.





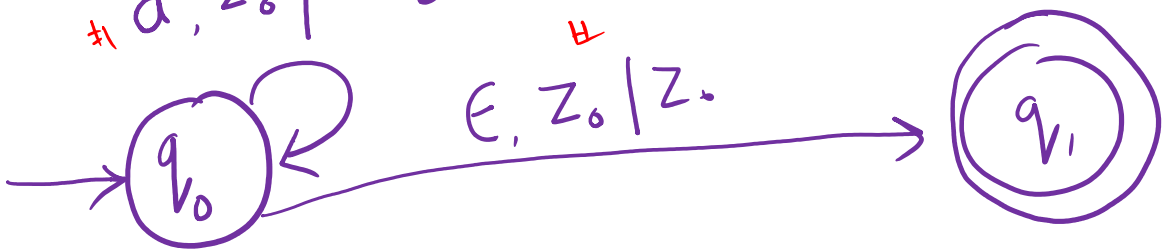
accepted.

## Construction:

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

$q_0, q_1$ — $a, b$ — $a, b, Z_0$ — $q_0$ — $Z_0$ — $q_1$

abaabaabb




Define transition

① $\delta(q_0, a, Z_0) = (q_0, aZ_0)$

② $\delta(q_0, b, Z_0) = (q_0, bZ_0)$

③ $\delta(q_0, a, a) = (q_0, aa)$

④ $\delta(q_0, b, b) = (q_0, bb)$

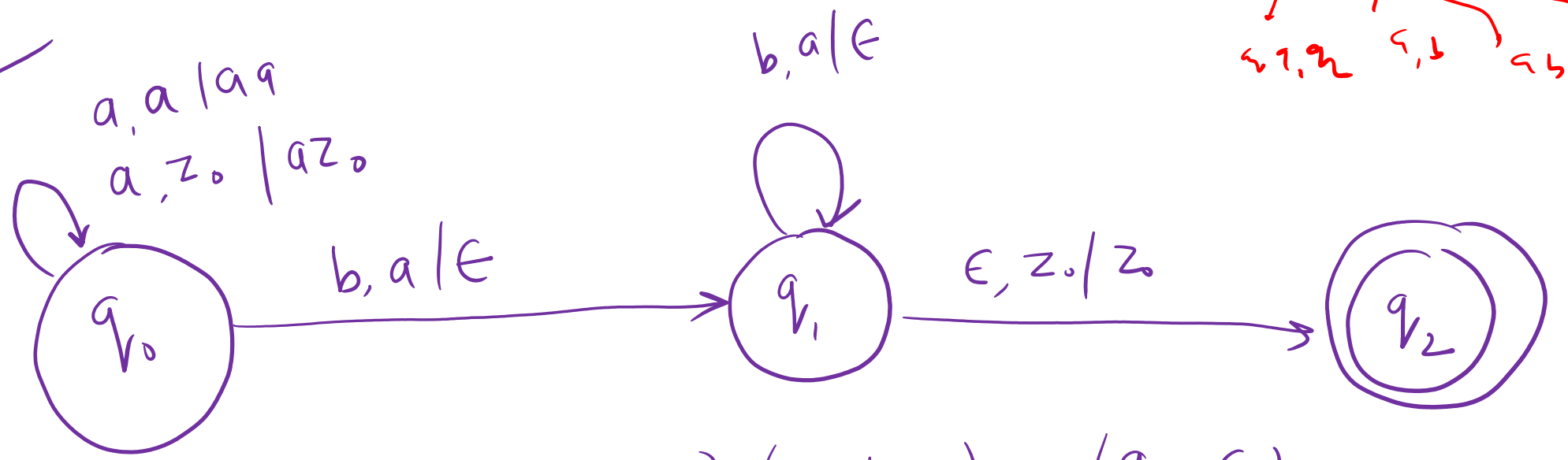
⑤ $\delta(q_0, a, b) = (q_0, \epsilon)$

⑥ $\delta(q_0, b, a) = (q_0, \epsilon)$

⑦ $\delta(q_0, \epsilon, Z_0) = (q_1, Z_0)$

# Construct a PDA that accepts a language = { $a^n b^n$ | n>=1} over string (a,b)

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




① $\delta(q_0, a, z_0) = (q_0, az_0)$

② $\delta(q_0, a, a) = (q_0, aa)$

③ $\delta(q_0, b, a) = (q_1, \epsilon)$

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

⑤ $\delta(q_1, \epsilon, z_0) = (q_2, z_0)$

aabbcc

# Construct a PDA that accepts a language = $\{ a^n b^n c \mid n>=1\}$ over string (a,b,c)

- Do yourself

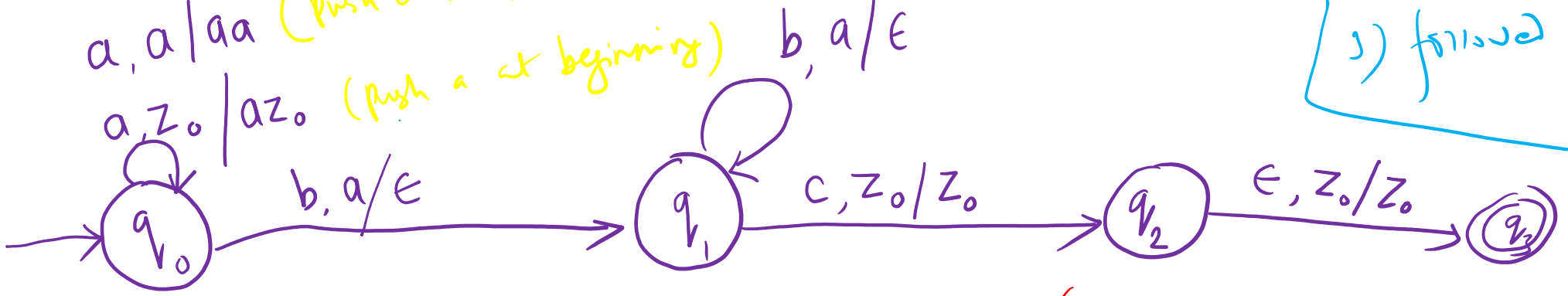
$L = \{ abc, aabbc, aaabbbc, \ldots \}$

Condition:

1) a's no. = b's no. (n) (n)

2) a's followed by b's

3) followed by a single c.




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

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

$\delta(q_0, b, a) = (q_1, \epsilon)$

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

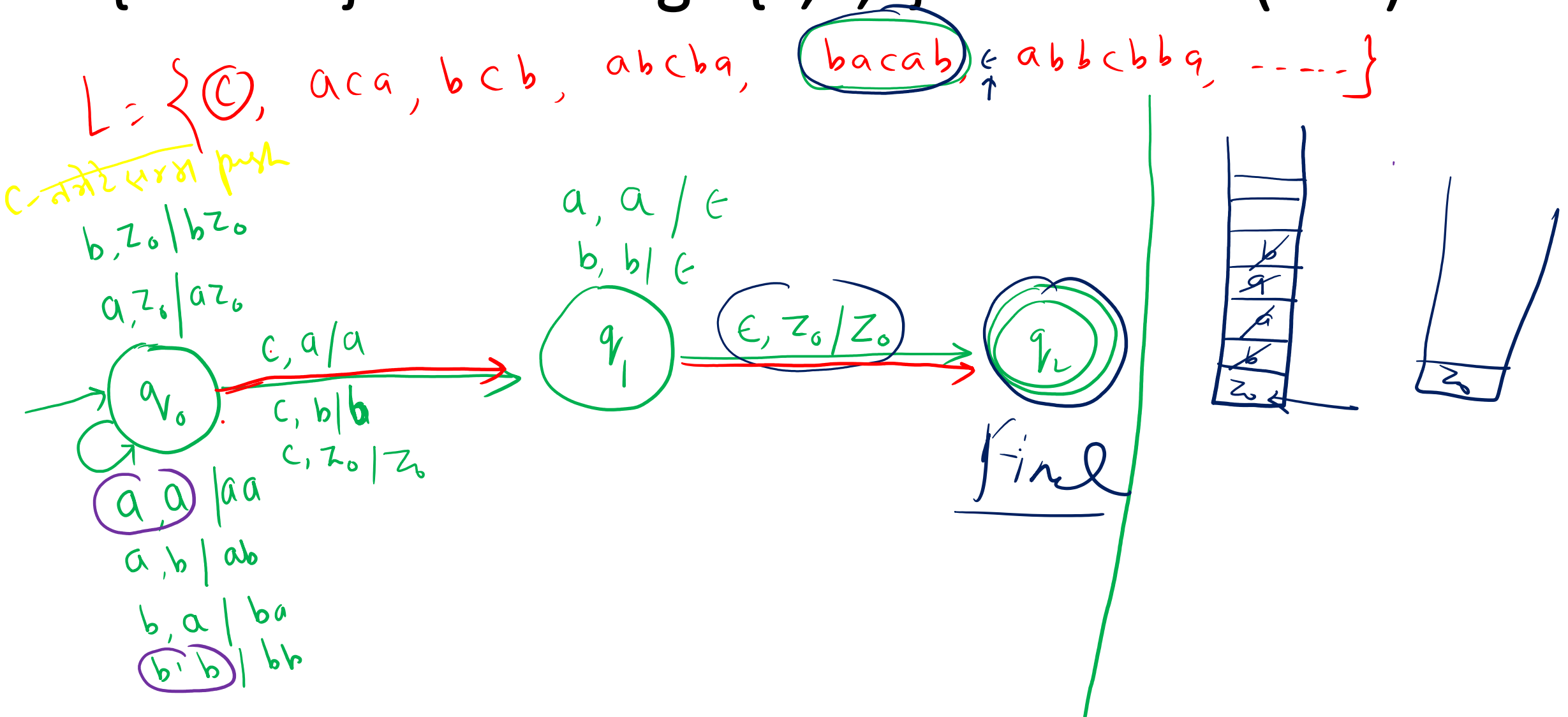
$\delta(q_1, c, Z_0) = (q_2, Z_0)$

$\delta(q_2, \epsilon, Z_0) = (q_3, Z_0)$

# Construct a PDA that accepts a language = $\{ a^n b^n c^m \mid$ m,n>=1$\}$ over string (a,b,c)

- Do yourself!

# Construct a PDA that accepts a language = {Wc$W^R$} over string ={a,b,c} where w =(a+b)*



even length palindrome

# Construct a PDA that accepts a language = $\{WW^R\}$ over string ={a,b} where w =(a+b)*

L = {ε, aa, bb, abba, baab, abbbba, ....}

1) break / middle point

If adjacent symbol same, there is chance of NP.

eg: ab b ... ✗

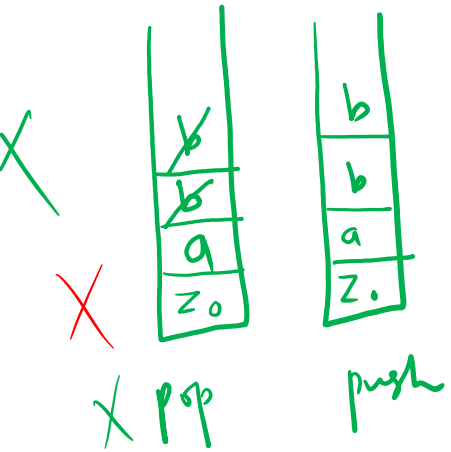
ab a x x

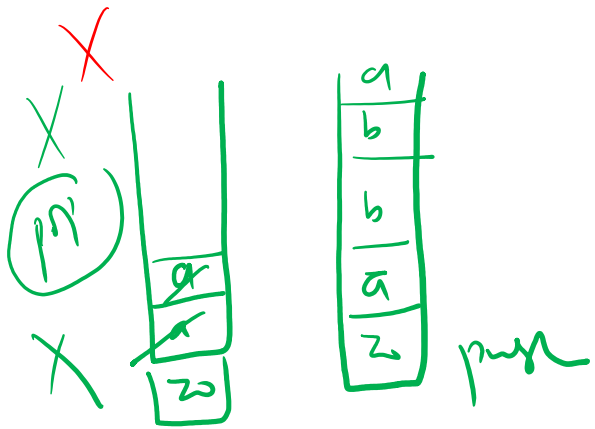
abbbba

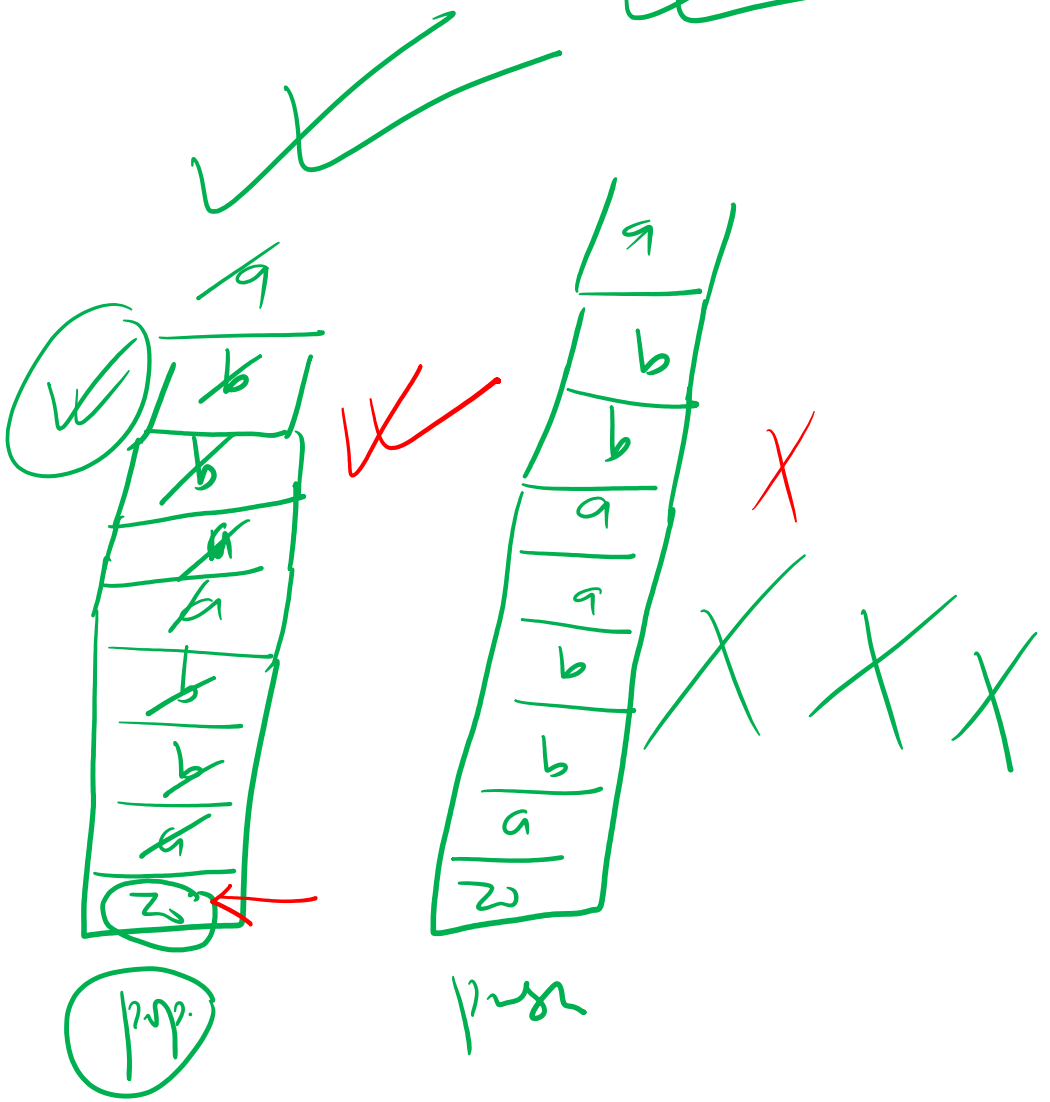
Logic:

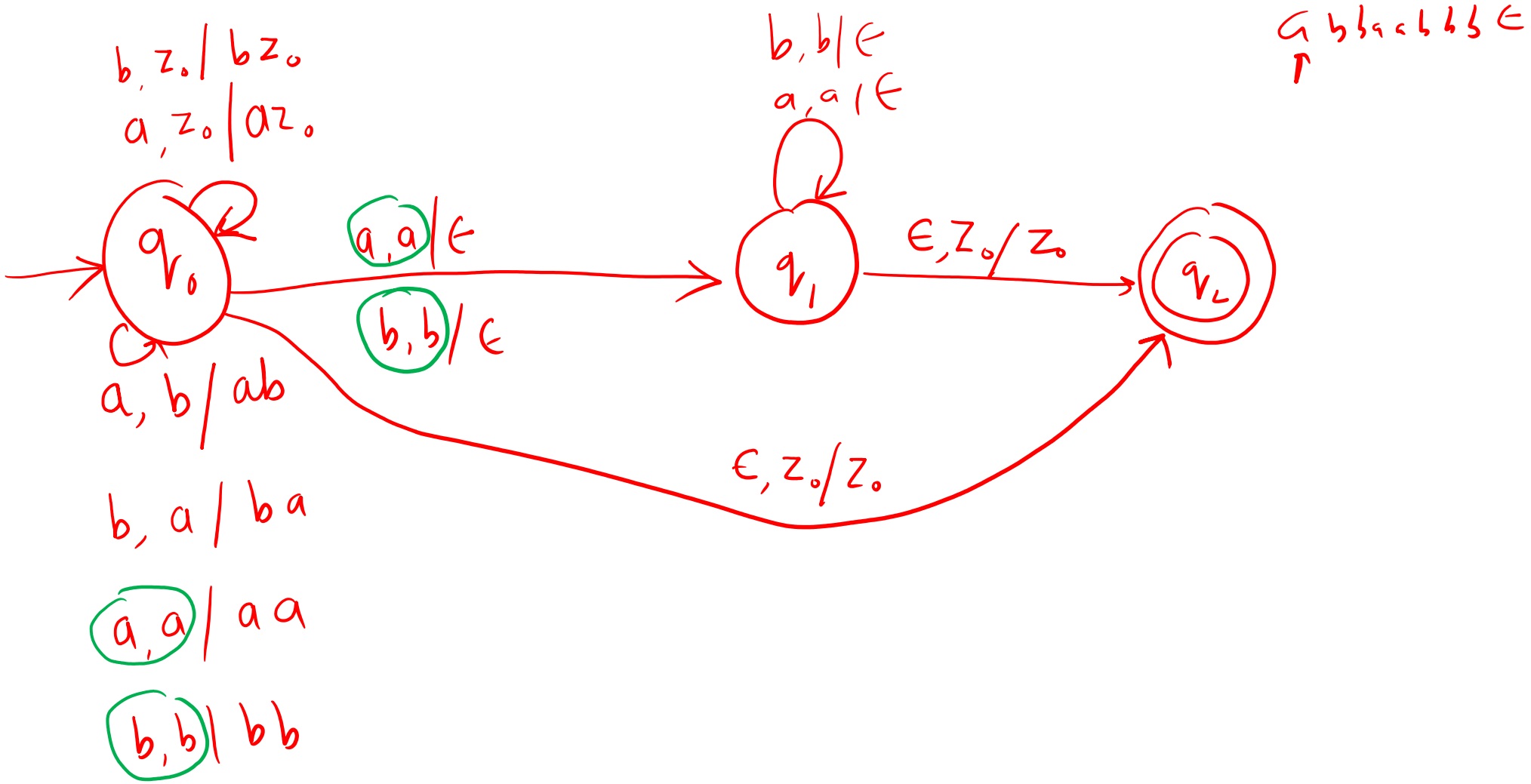
$(q_1, \epsilon, z_0) = (q_2, z_0)$

e.g. a b b a @ b b a ε






a bb a c bb b ∈

WWR NDPDA

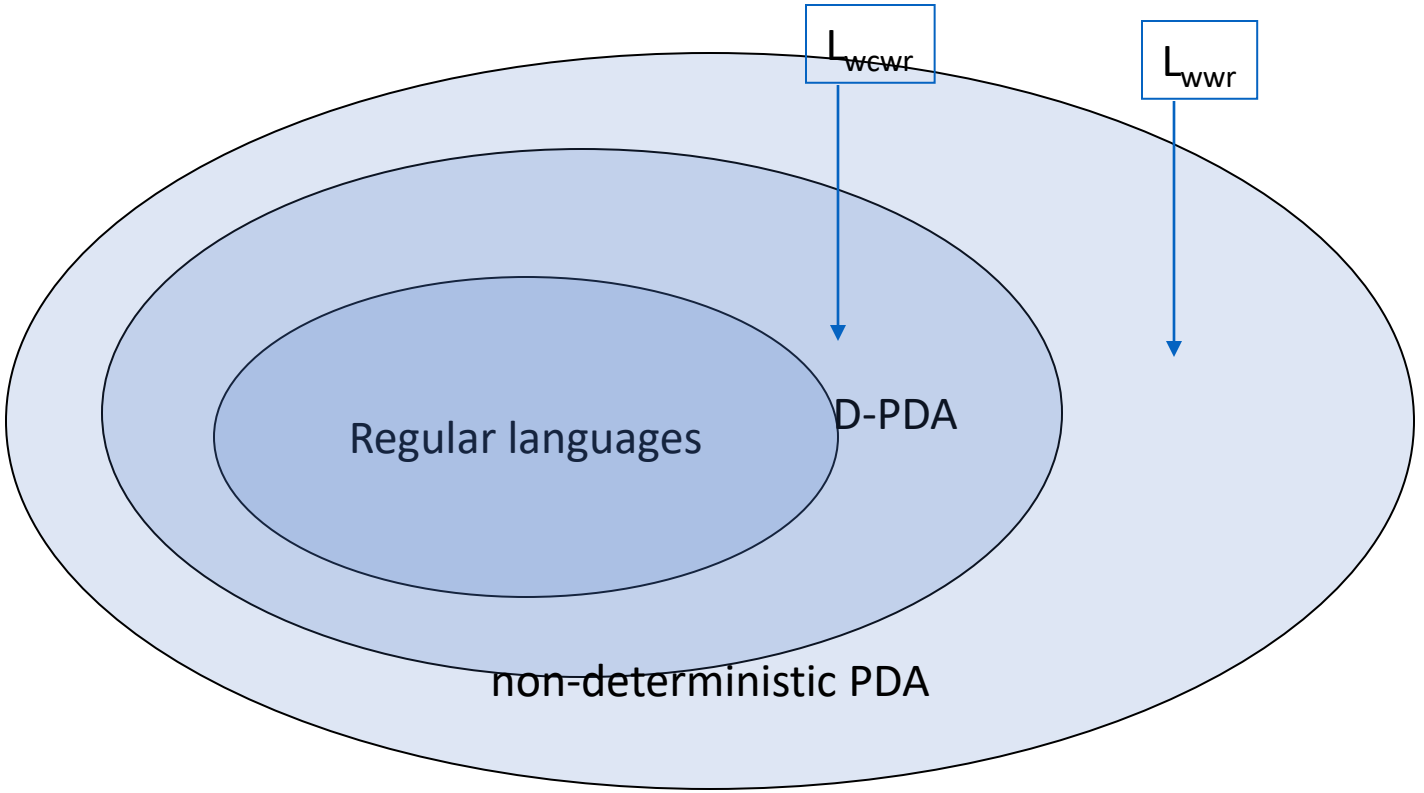
# Deterministic and Non-Deterministic PDA

- **Wc**$W^R$ → Deterministic PDA
- **W**$W^R$ → Non-Deterministic PDA

- See examples in prev. slides.

# PDA vs DPDA vs Regular languages

$ww^R \rightarrow$ DPDA / NDPA



# PDA's Instantaneous Description (ID)

A PDA has a configuration at any given instance: **(q,w,y)**

- q - current state
- w - remainder of the input (i.e., unconsumed part)
- y - current stack contents as a string from top to bottom of stack

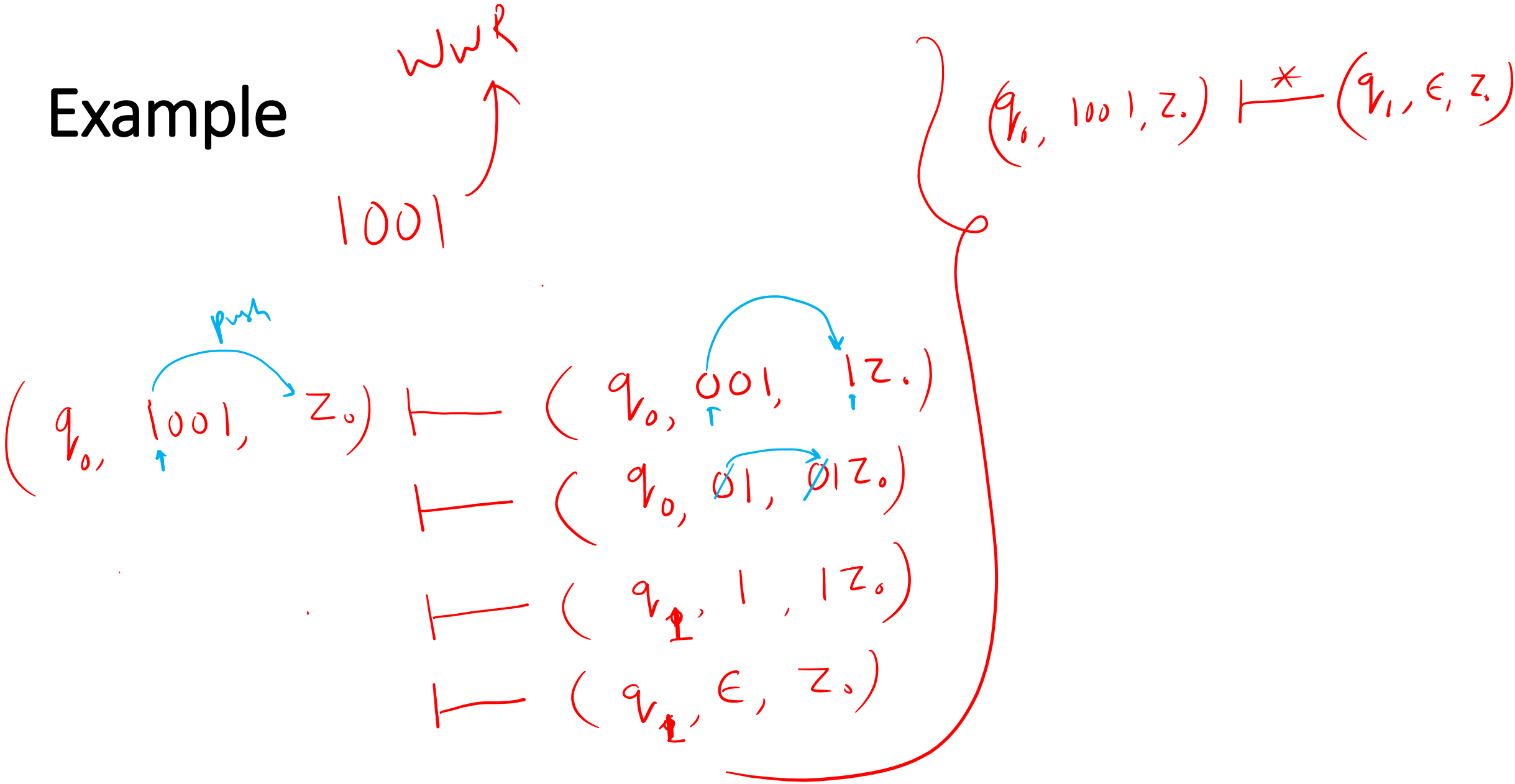
If $\delta(q,a, X)=\{(p, A)\}$ is a transition, then the following are also true:

- $(q, a, X) \vdash (p,\varepsilon A)$
- $(q, aw, XB) \vdash (p,w,AB)$

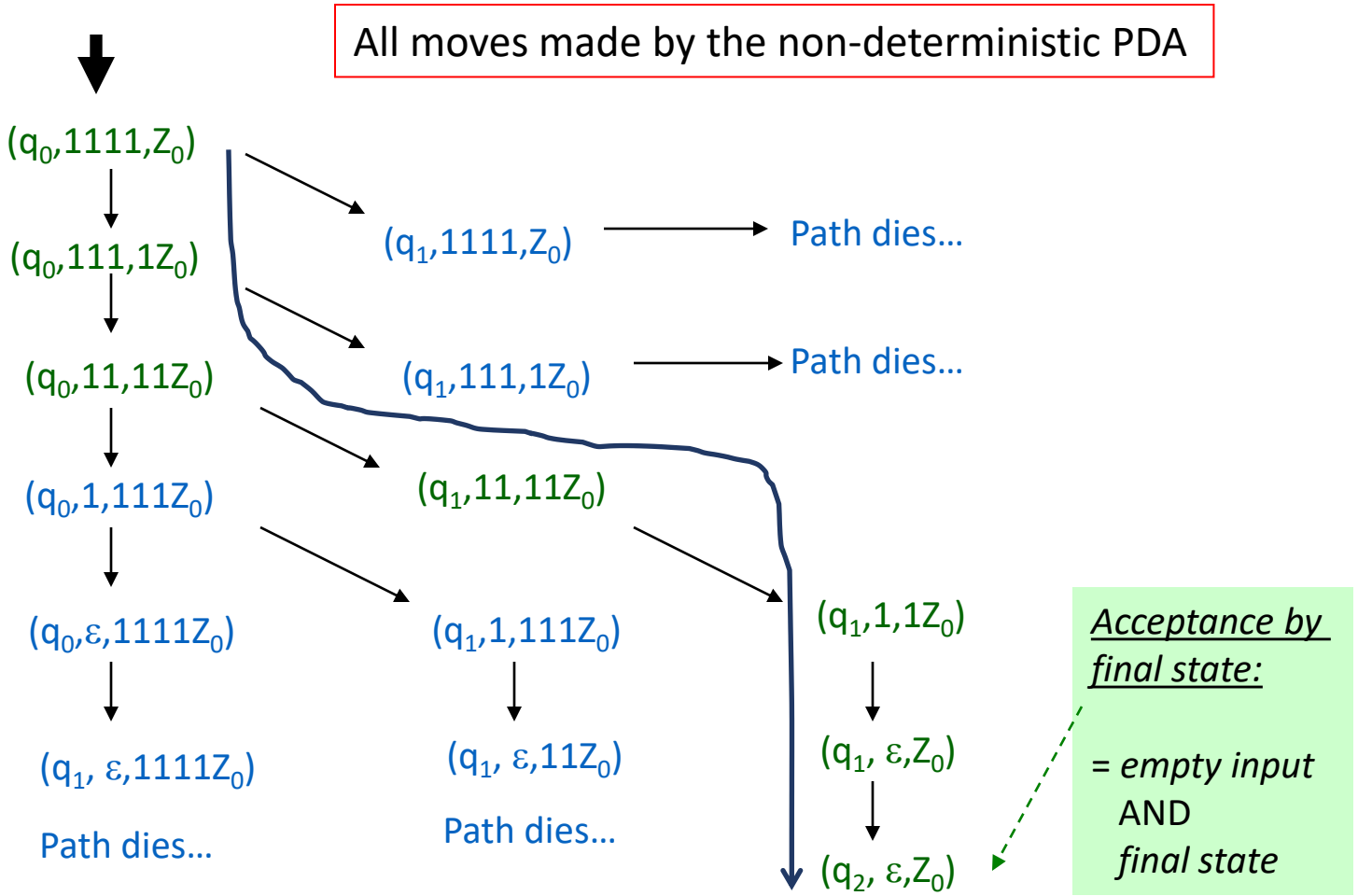
|--- sign is called a "turnstile notation" and represents one move

|---* sign represents a sequence of moves

# Example


wwR
1001
$(q_0, 1001, z_0) \vdash^* (q_1, \epsilon, z_0)$
push
$(q_0, 1001, z_0) \vdash (q_0, 001, 1z_0)$
$\vdash (q_0, 01, 01z_0)$
$\vdash (q_1, 1, 1z_0)$
$\vdash (q_1, \epsilon, z_0)$

# How does the PDA for $L_{wwr}$ work on input "1111"?



There are two types of PDAs that one can design:
those that accept by final state or by empty stack

# Acceptance by…

- *PDAs that accept by **final state**:*
  - For a PDA P, the language accepted by P, denoted by L(P) by *final state*, is:
    - $\{w \mid (q_0,w,Z_0) \vdash^* (q,\varepsilon, A)\}$, s.t., $q \in F$

Checklist:
- input exhausted?
- in a final state?

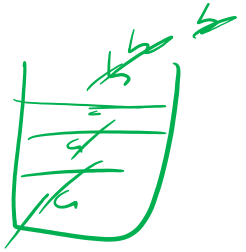
- *PDAs that accept by **empty stack**:*
  - For a PDA P, the language accepted by P, denoted by N(P) by *empty stack*, is:
    - $\{w \mid (q_0,w,Z_0) \vdash^* (q, \varepsilon, \varepsilon)\}$, for any $q \in Q$.

Q) Does a PDA that accepts by empty stack
need any final state specified in the design?

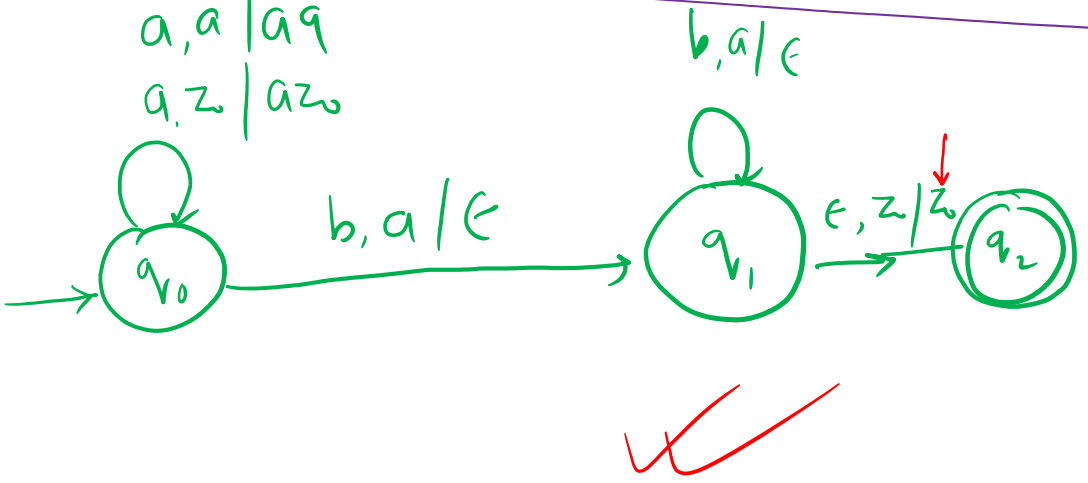
Checklist:
- input exhausted?
- is the stack empty?

# Example

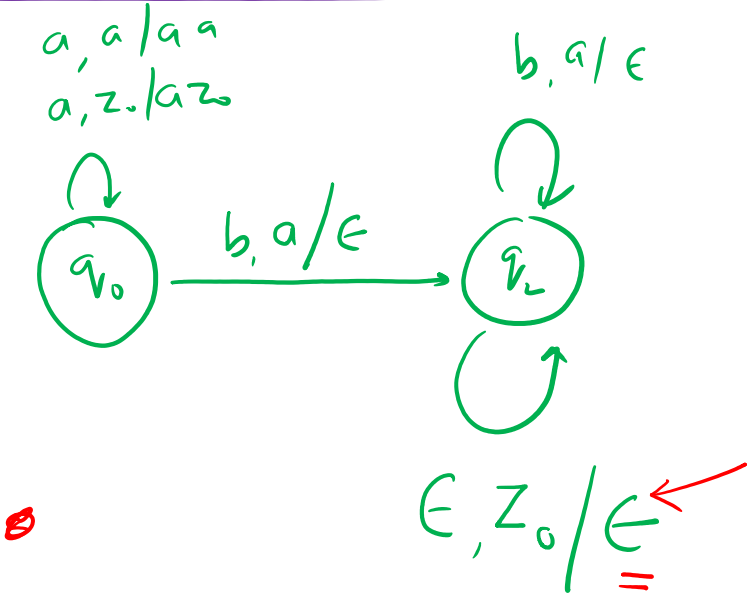
$L = a^n b^n$



final state




empty stack




= stack empty

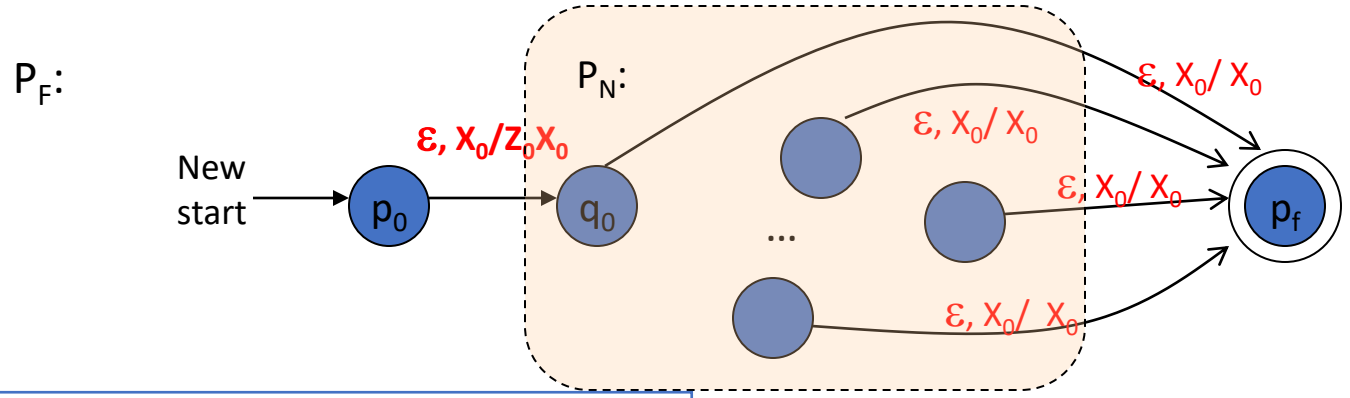
They are equivalent!

# PDAs accepting by final state and empty stack are equivalent

- $P_F$ <= PDA accepting by final state
  - $P_F = (Q_F, \sum, \Gamma, \delta_F, q_0, Z_0, F)$
- $P_N$ <= PDA accepting by empty stack
  - $P_N = (Q_N, \sum, \Gamma, \delta_N, q_0, Z_0)$
- Theorem:
  - *($P_N$ ==> $P_F$)* For every $P_N$, there exists a $P_F$ s.t. $L(P_F)=L(P_N)$
  - *($P_F$ ==> $P_N$)* For every $P_F$, there exists a $P_N$ s.t. $L(P_F)=L(P_N)$

# $P_N$==> $P_F$ construction

- Whenever $P_N$'s stack becomes empty, make $P_F$ go to a final state without consuming any addition symbol
- <u>To detect empty stack in $P_N$:</u> $P_F$ pushes a new stack symbol $X_0$ (not in $\Gamma$ of $P_N$) initially before simultating $P_N$




$P_F = (Q_N \cup \{p_0, p_f\}, \sum, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$

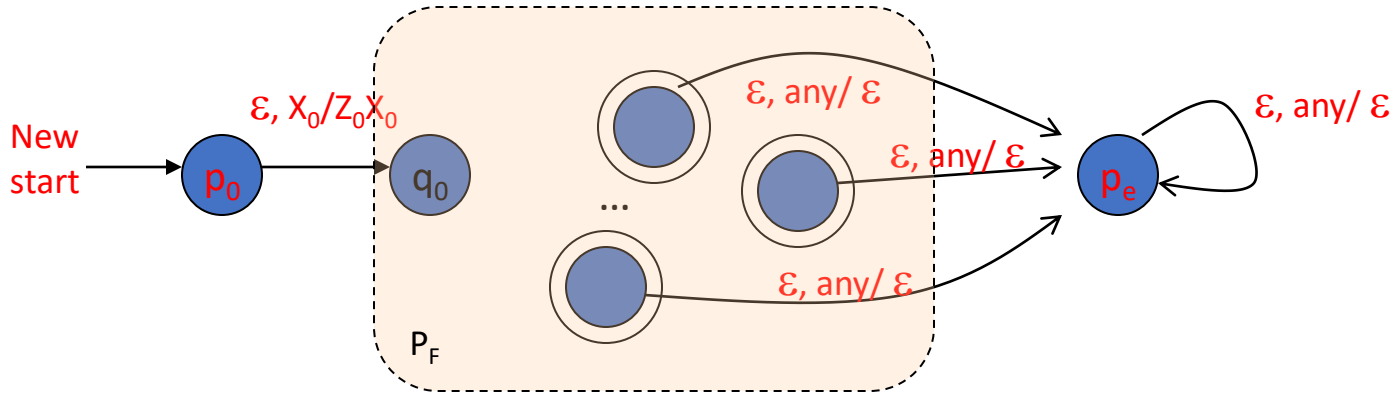
# $P_F$==> $P_N$ construction

- Main idea:
  - Whenever $P_F$ reaches a final state, just make an $\varepsilon$-transition into a new end state, clear out the stack and accept
  - Danger: What if $P_F$ design is such that it clears the stack midway *without* entering a final state?

    ➔ to address this, add a new start symbol $X_0$ (not in $\Gamma$ of $P_F$)

$P_N = (Q \cup \{p_0, p_e\}, \sum, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$

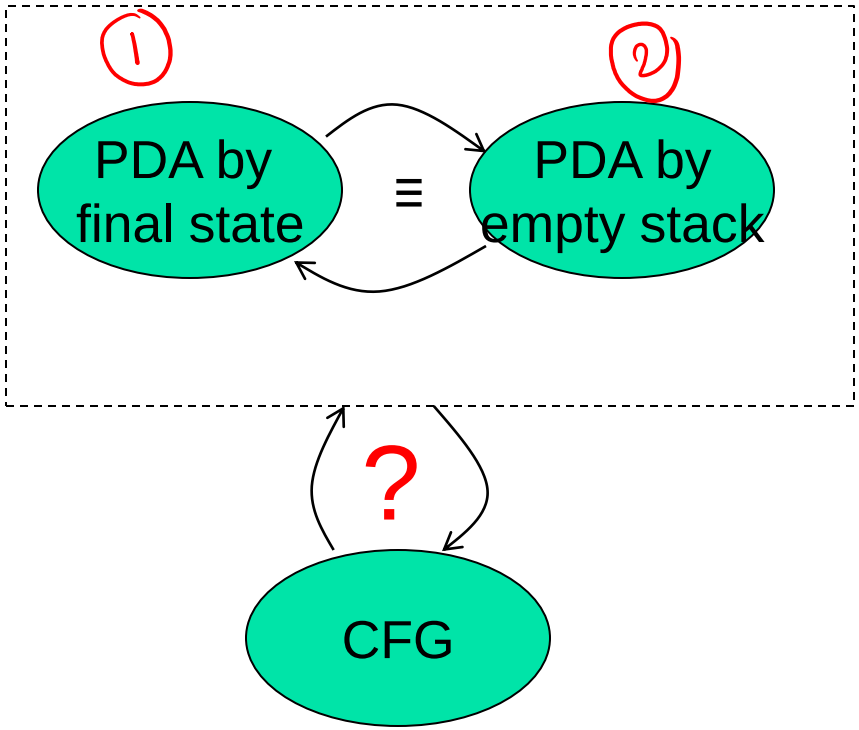
$P_N$:



# Equivalence of PDAs and CFGs

By Prashant Gautam

# CFGs == PDAs ==> CFLs



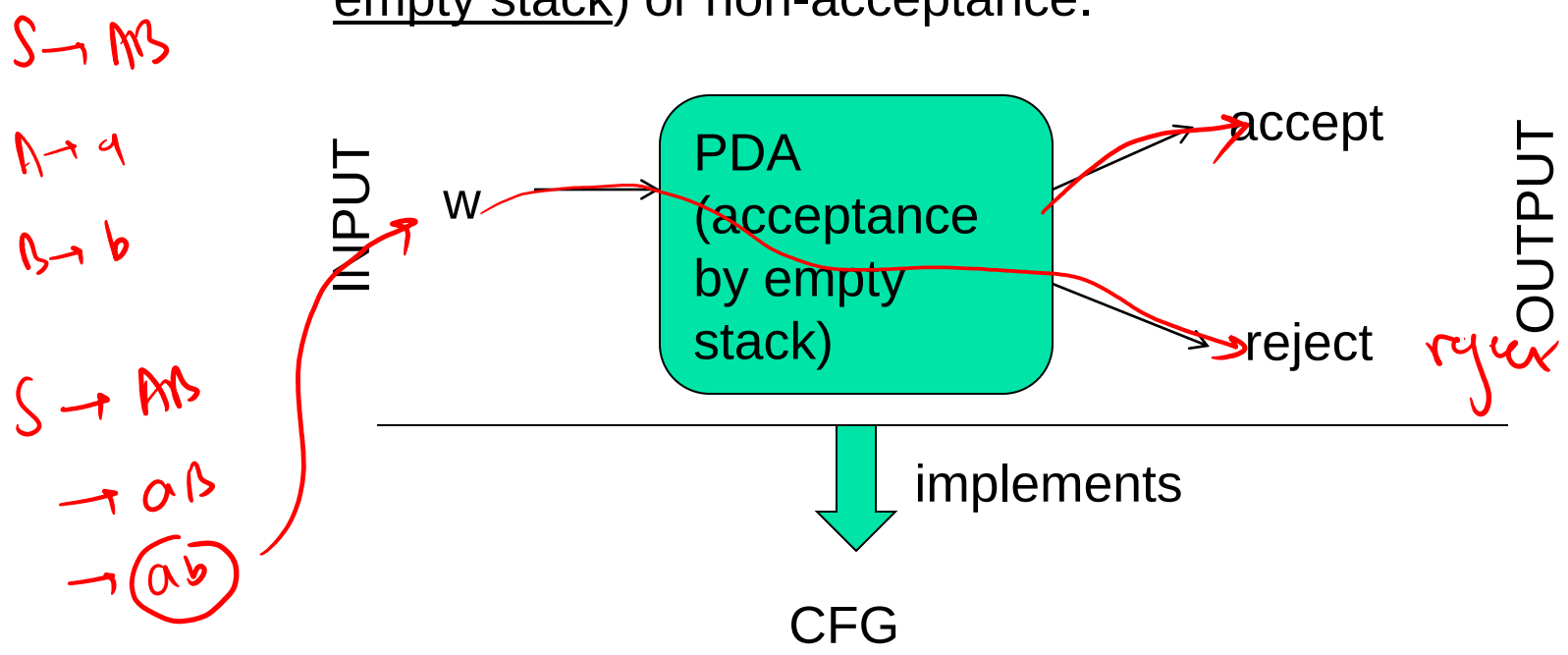

PDA with empty stack

CFG

This is same as: “implementing a CFG using a PDA”

# Converting CFG to PDA

Main idea: The PDA simulates the leftmost derivation on a given w, and upon consuming it fully it either arrives at acceptance (by empty stack) or non-acceptance.

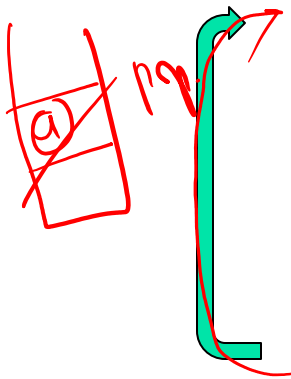


This is same as: “implementing a CFG using a PDA”

# Converting a CFG into a PDA

Main idea: The PDA simulates the leftmost derivation on a given w, and upon consuming it fully it either arrives at acceptance (by empty stack) or non-acceptance.

## Steps:



1. Push the right hand side of the production onto the stack, with leftmost symbol at the stack top
2. If stack top is the leftmost variable, then replace it by all its productions (each possible substitution will represent a *distinct* path taken by the non-deterministic PDA)
3. If stack top has a terminal symbol, and if it matches with the next symbol in the input string, then pop it

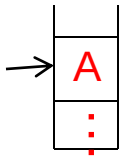
State is inconsequential (only one state is needed)

# Formal construction of PDA from CFG

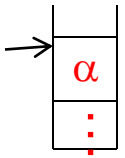
Note: Initial stack symbol (S) same as the start variable in the grammar

- Given: G= (V,T,P,S)
- Output: $P_N = (\{q\}, T, V \cup T, \delta, q, S)$
- δ:
  - For all $A \in V$ , add the following transition(s) in the PDA:
    - $\delta(q, \varepsilon, A) = \{ (q, \alpha) \mid \text{“}A \Rightarrow \alpha\text{”} \in P\}$

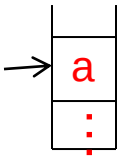
Before:




After:

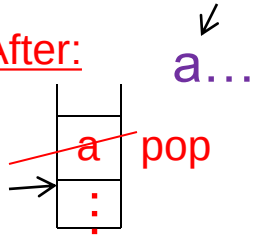



Before:

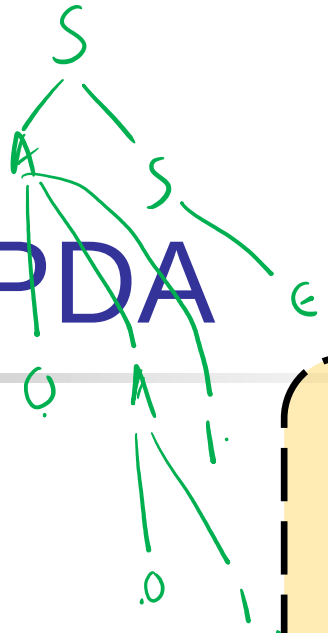



- For all *a* ∈ T, add the following transition(s) in the PDA:
  - $\delta(q, a, a) = \{ (q, \varepsilon) \}$

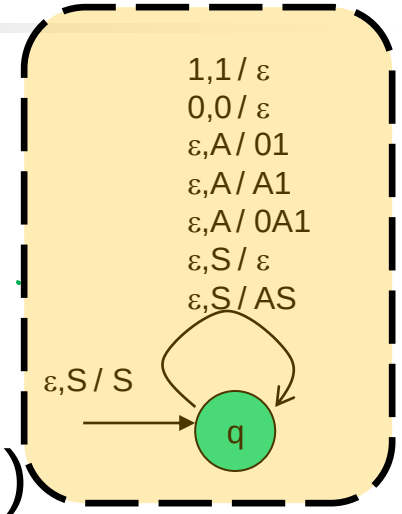
After:



# Example: CFG to PDA




- G = ( {S,A}, {0,1}, P, S)
- P:
  - S ==> AS | ε
  - A ==> 0A1 | A1 | 01
- PDA = ({q}, {0,1}, {0,1,A,S}, δ, q, S)
- δ:
  - δ(q, ε , S) = { (q, AS), (q, ε )}
  - δ(q, ε , A) = { (q,0A1), (q,A1), (q,01) }
  - δ(q, 0, 0) = { (q, ε ) }
  - δ(q, 1, 1) = { (q, ε ) }



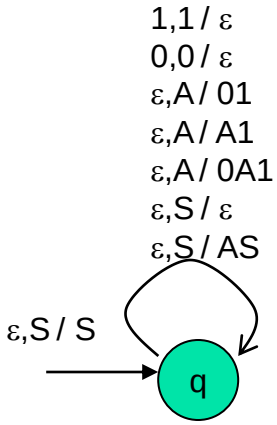

How will this new PDA work?

Lets simulate string 0011

# Simulating string 0011 on the new PDA …

PDA (δ):

δ(q, ε , S) = { (q, AS), (q, ε )}
δ(q, ε , A) = { (q,0A1), (q,A1), (q,01) }
δ(q, 0, 0) = { (q, ε ) }
δ(q, 1, 1) = { (q, ε ) }

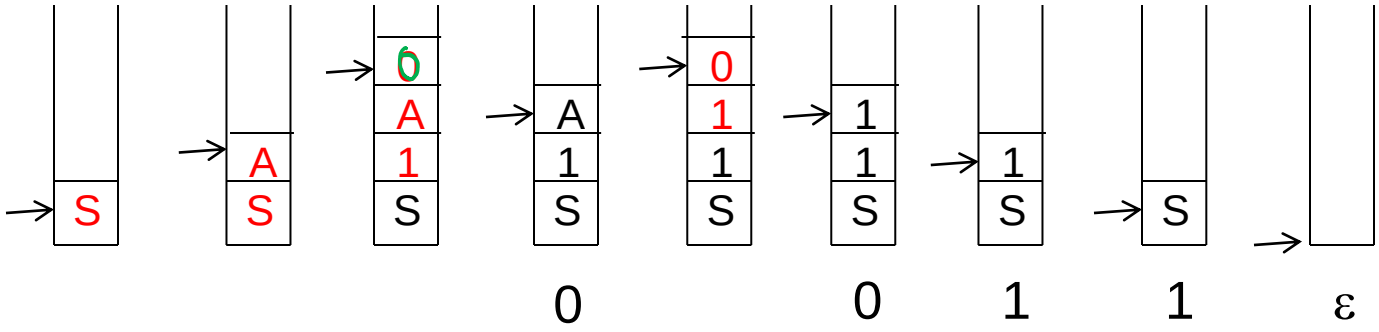



Leftmost deriv.:

S => AS
=> 0A1S
=> 0011S
=> 0011

A → 0A1

Stack moves (shows only the successful path):




Accept by empty stack

# Converting a PDA into a CFG

- Main idea: Reverse engineer the productions from transitions

If $\delta(q,a,Z) \Rightarrow (p, Y_1Y_2Y_3\ldots Y_k)$:

1. State is changed from q to p;
2. Terminal *a* is consumed;
3. Stack top symbol Z is popped and replaced with a sequence of k variables.

- Action: Create a grammar variable called "[qZp]" which includes the following production:

$[qZp] \Rightarrow a[pY_1q_1]\,[q_1Y_2q_2]\,[q_2Y_3q_3]\ldots\,[q_{k-1}Y_kq_k]$

PDA → CFG

CFG = (V, T, P, S)

# Example:

Given: PDA.

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

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

(iii) $\delta(q_0, b, a) = (q_1, a)$

(iv) $\delta(q_1, b, a) = (q_1, a)$

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

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

(1) $S \rightarrow [q_0 Z_0 q_0]$

(2) $S \rightarrow [q_0 Z_0 q_1]$

(i) $\delta(q_0, a, z_0) = (q_0, az_0)$

$A \rightarrow aBA$

$2 \rightarrow 2^2 = 4$ Combination

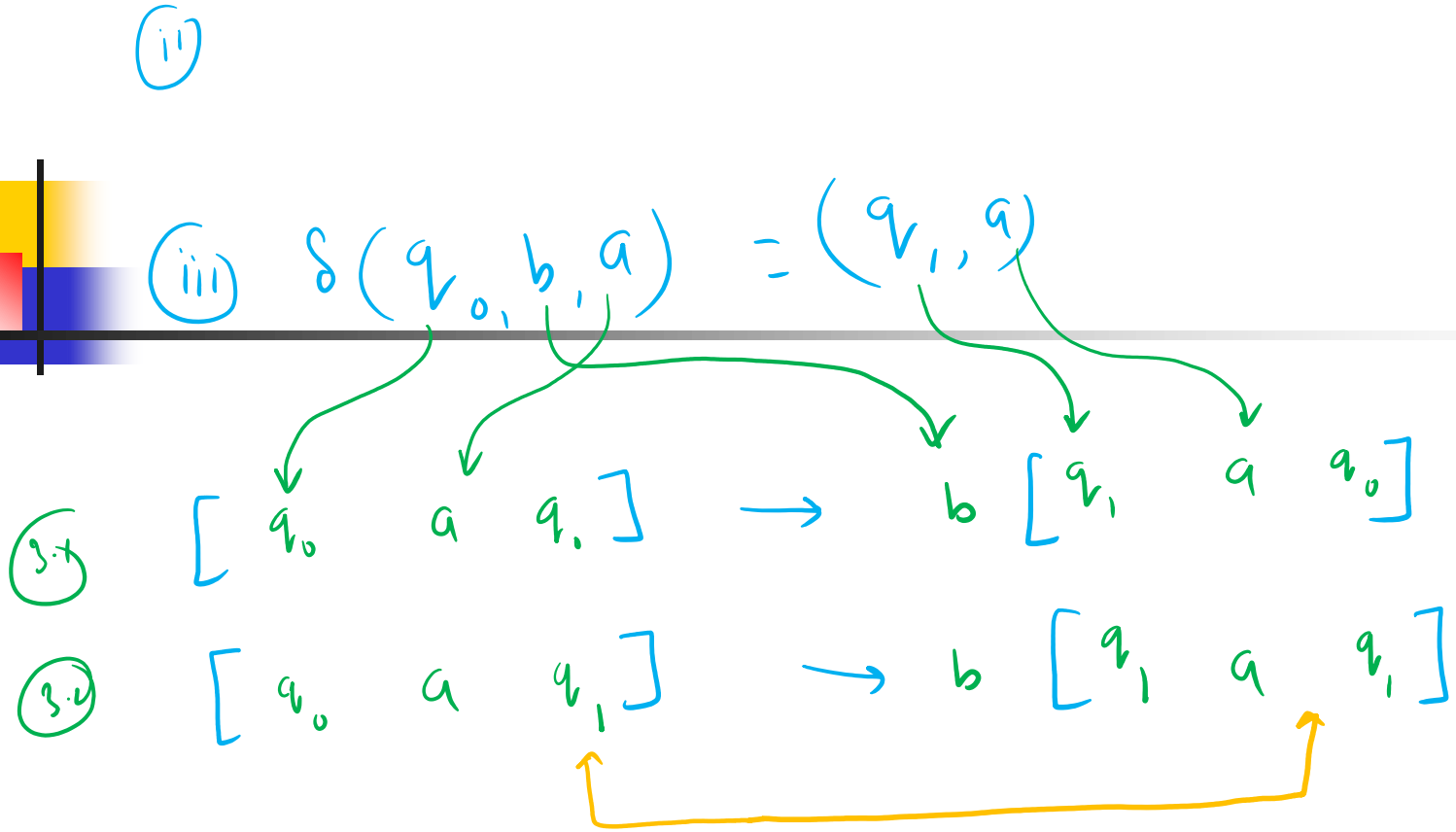
(i) $\delta(q_0, a, z_0) = (q_0, az_0)$

1.1 $[q_0\ z_0\ q_0] \rightarrow a[q_0\ a\ q_0][q_0\ z_0\ q_0]$ (A, B, A)

1.2 $[q_0\ z_0\ q_0] \rightarrow a[q_0\ a\ q_1][q_1\ z_0\ q_0]$

1.3 $[q_0\ z_0\ q_1] \rightarrow a[q_0\ a\ q_0][q_0\ z_0\ q_1]$

1.4 $[q_0\ z_0\ q_1] \rightarrow a[q_0\ a\ q_1][q_1\ z_0\ q_1]$


ii)
iii) δ(q0, b, a) = (q1, a)
3.i) [q0 a q0] → b [q1 a q0]
3.ii) [q0 a q1] → b [q1 a q1]

(iv) $\delta(q_1, b, a) = (q_1, a)$ similar to (iii)

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

$2^0 = 1$

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

(5.1) $[\, q_1 \quad a \quad q_0 \,] \rightarrow q$


vi

# Example: Bracket matching

- To avoid confusion, we will use *b*="(" and *e*=")"

$P_N$: ( $\{q_0\}$, $\{b,e\}$, $\{Z_0,Z_1\}$, $\delta$, $q_0$, $Z_0$ )

1. $\delta(q_0,b,Z_0) = \{ (q_0,Z_1Z_0) \}$
2. $\delta(q_0,b,Z_1) = \{ (q_0,Z_1Z_1) \}$
3. $\delta(q_0,e,Z_1) = \{ (q_0, \varepsilon ) \}$
4. $\delta(q_0, \varepsilon ,Z_0) = \{ (q_0, \varepsilon ) \}$

0. $S \Rightarrow [q_0Z_0q_0]$
1. $[q_0Z_0q_0] \Rightarrow b\ [q_0Z_1q_0]\ [q_0Z_0q_0]$
2. $[q_0Z_1q_0] \Rightarrow b\ [q_0Z_1q_0]\ [q_0Z_1q_0]$
3. $[q_0Z_1q_0] \Rightarrow e$
4. $[q_0Z_0q_0] \Rightarrow \varepsilon$

Let $A=[q_0Z_0q_0]$
Let $B=[q_0Z_1q_0]$

0. $S \Rightarrow A$
1. $A \Rightarrow b\ B\ A$
2. $B \Rightarrow b\ B\ B$
3. $B \Rightarrow e$
4. $A \Rightarrow \varepsilon$

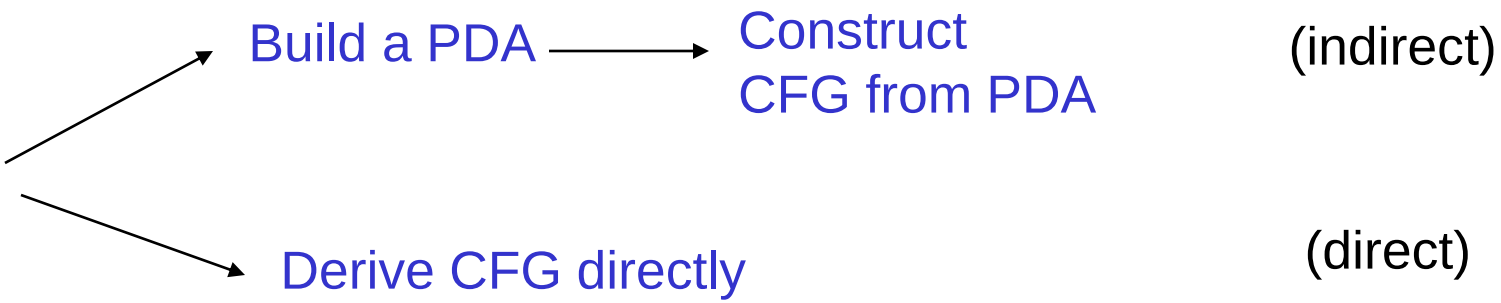
Simplifying,

0. $S \Rightarrow b\ B\ S \mid \varepsilon$
1. $B \Rightarrow b\ B\ B \mid e$

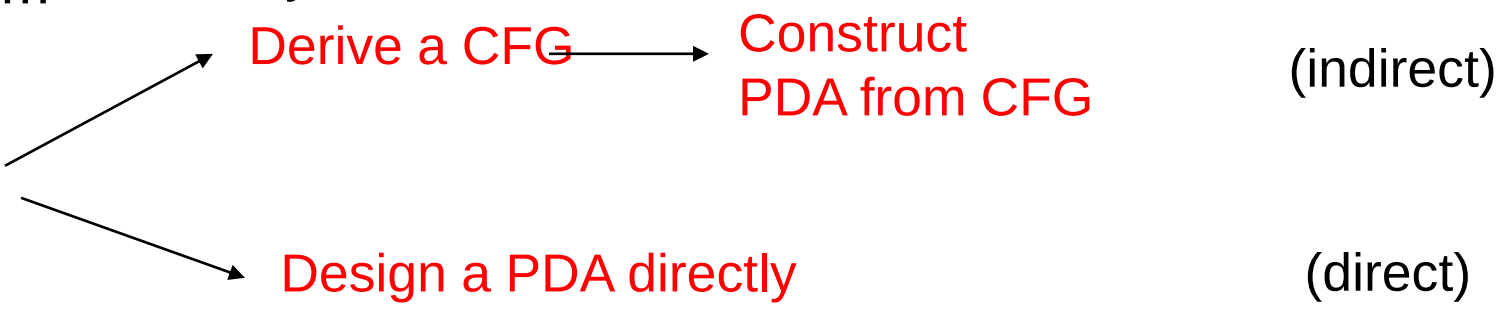
If you were to directly write a CFG:

$S \Rightarrow b\ S\ e\ S \mid \varepsilon$

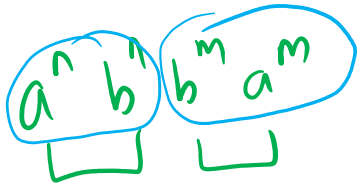
# Two ways to build a CFG




Similarly… Two ways to build a PDA



$a^n b^{n+m} a^m \mid (m,n) >= 1$




$L = \{ abba,\ aabbba,\ abbb\ aa,\ ---- \}$